ỨNG DỤNG MÔ HÌNH MAPREDUCE VÀO BÀI TOÁN TÌM KIẾM NHỮNG KHÁCH HÀNG CÓ CÙNG NHU CẦU SẢN PHẨM TRONG THƯƠNG MẠI ĐIỆN TỬ

Trần Xuân Hiệp, Phan Thị Thanh Thủy*
Trường Đại học Phú Yên
Ngày nhận bài: 07/05/2020; Ngày nhận đăng: 28/05/2021

Tóm tắt

Tìm kiếm những khách hàng có cùng nhu cầu sản phẩm đang trở nên rất quan trọng trong lĩnh vực thương mại điện tử. Những người có cùng mối quan tâm, thông tin cá nhân phù hợp và những tương tác của họ với cộng đồng là nguồn dữ liệu quan trọng được chọn để phân tích. Bài báo này giới thiệu phương pháp phân tích hồ sơ cá nhân bằng cách áp dụng các giải thuật phổ biến vào mô hình Map-Reduce dựa trên Hadoop framework. Đây là phương pháp nhằm tạo ra công cụ hiệu quả, tiết kiệm thời gian và công sức tìm ra danh sách tối ưu những người có cùng sở thích để phục vụ doanh nghiệp tư vấn sản phẩm cho khách hàng.

Từ khoá: *MapReduce, Hadoop, so trùng ngữ nghĩa, phân tích ngữ nghĩa*

1. Giới thiệu

Sự phát triển của cuộc cách mạng công nghiệp 4.0 trong những năm gần đây đã tạo ra cơ hội hơn cho lĩnh vực thương mại điện tử. Trước nhu cầu ngày càng cao của khách hàng, những bài toán, giải pháp nhằm nâng cao chất lượng dịch vụ ngày càng trở nên cấp thiết và phức tạp. Vấn đề tìm kiếm khách hàng có cùng nhu cầu về sản phẩm trong thương mại điện tử một cách hiệu quả, nhanh chóng trên một tập dữ liệu rất lớn là một trong những bài toán luôn đặt ra nhiều vấn đề cho các nhà kinh doanh thương mại điện tử. Trên cơ sở này chúng ta có thể phát triển phương pháp xây dựng hồ sơ cá nhân (proflile) chứa các thông tin của khách hàng một cách khoa học, từ đó xây dựng phương pháp so trùng các thông tin cá nhân để tìm ra sự tương đồng theo từng nhóm sản phẩm.

Việc tìm kiếm những người có cùng mối quan tâm một lĩnh vực bao gồm hai vấn đề: làm thế nào để mô tả sơ lược mối quan tâm sản phẩm của một người và làm thế nào để tính toán sự tương tự giữa các hồ sơ cá nhân về lĩnh vực quan tâm. Phương pháp phổ biến nhất được sử dụng để tính độ tương tự là phép tính cosin (Rajesh Thiagarajan, Geetha Manjunath, and M. Stumptner, 2008). Trong phương pháp này, những profile được trình bày như là một tập những từ khóa (có trọng số). Sự tương đồng được tính bởi phép tính cosine của hai vectơ được đại diện bởi hai tập từ khoá được đánh giá cao. Phương pháp này thì đơn giản để thực hiện, nhưng nó chỉ đề cập đến sự phù hợp cú pháp của từ và không có khả năng trong những trường hợp đòi hỏi so sánh ngữ nghĩa của từ. Để đáp ứng nhu cầu cho việc phù hợp ngữ nghĩa chúng ta cần mô tả một cách rõ ràng mối quan hệ giữa các từ. Những mối quan hệ giữa các từ hoặc các thuật ngữ cũng có thể được suy tính hoàn toàn, bằng cách sử dụng phương pháp phân tích ngữ nghĩa tiềm ẩn (T. K. Landauer, P. W. Foltz, and D. Laham, 1998).

* Email: thuycdsppy@yahoo.com

Việc tra cứu trong kho dữ liệu khổng lồ chứa thông tin cá nhân trong cộng đồng này bằng cách lựa chọn từng người thực chất là một việc tốn rất nhiều thời gian, đồng thời khả năng tìm được người thích hợp không cao. Với sự bùng nổ thông tin hiện nay, việc xử lý và lưu trữ khối dữ liệu khổng lồ ngày một phát triển đã trở thành một thách thức lớn với nhiều vấn đề đặt ra. Nhưng thực tế này đang dần thay đổi khi các kỹ thuật xử lý phân tán xuất hiện, đáng kể nhất là MapReduce.

Mục đích quan trọng nhất của MapReduce là tăng tốc độ thực thi xử lý dữ liệu trên tập hợp dữ liệu lớn (Jeffrey Dean and Sanjay Ghemawat, 2004). Mô hình MapReduce dựa trên hai hàm Map và Reduce. MapReduce là một giải pháp tốt cho các dạng bài toán xử lý khối lượng dữ liệu phát sinh khổng lồ với các tác vụ phân tích và tính toán phức tạp, trong các lĩnh vực như khai thác dữ liệu, phân tích và mô phỏng. Bài báo này trình bày phương pháp nghiên cứu và kết quả thử nghiệm việc áp dụng giải thuật tìm kiếm những người có cùng sở thích trong cộng đồng khoa học dựa trên mô hình MapReduce.

2. Phương pháp nghiên cứu

2.1. MapReduce

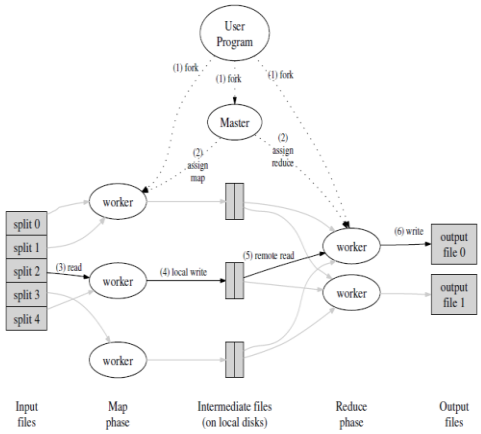
2.1.1 Mô hình MapReduce

Nhiệm vụ tính toán trong mô hình MapReduce nhận dữ liệu vào và ra đều là một tập hợp lớn các cặp khóa/giá trị. Mô hình này dựa trên 2 bước cơ bản:

- Map: nhận dòng dữ liệu thô và xử lý, tạo ra các cặp khóa/giá trị. Thư viện MapReduce sẽ tự động nhóm các giá trị có cùng khóa lại với nhau, và chuyển chúng đến các hàm Reduce.

- Reduce: nhận một khóa và các giá trị tương ứng với khóa đó dưới dạng list, sẽ tiếp tục xử lý các giá trị này để tạo thành kết quả cuối cùng tùy theo yêu cầu công việc. Thư viện MapReduce tiếp nhận các kết quả này và xuất ra kết quả cuối cùng của bài toán dưới các dạng dữ liệu khác nhau.

Trong mô hình MapReduce, dữ liệu vào được chia thành nhiều mảnh nhỏ, mỗi mảnh được xử lý độc lập bởi các hàm Map. Kết quả xử lý của các hàm Map sẽ lại được phân thành nhiều tập hợp khác nhau, sắp xếp và chuyển tiếp đến các hàm Reduce. Hình 1 sau minh họa cách làm việc trên.



Hình 1: *Mô hình hoạt động MapReduce*

2.1.2 Hadoop

Hadoop là dự án chủ chốt của hãng Apache nhằm cung cấp công cụ và hỗ trợ cho các hệ thống tính toán phân tán. Dựa trên ý tưởng MapReduce của Google, Hadoop tự xử lý tất cả các khâu phức tạp để bảo đảm hệ thống hoạt động thông suốt và nhường việc định nghĩa tính toán luận lý của ứng dụng lại cho lập trình viên.

Hai thành phần chính của dự án Hadoop là framework MapReduce cung cấp khả năng tính toán phân tán, và hệ thống file HDFS (Hadoop Distributed File System). Framework MapReduce của Hadoop được xem là sự thiết lập ý tưởng cùng tên của Google trên nền ngôn ngữ Java, vì vậy các khái niệm cũng như mô hình hoạt động là hoàn toàn giống nhau.

- *Hadoop Distributed File System (HDFS)*

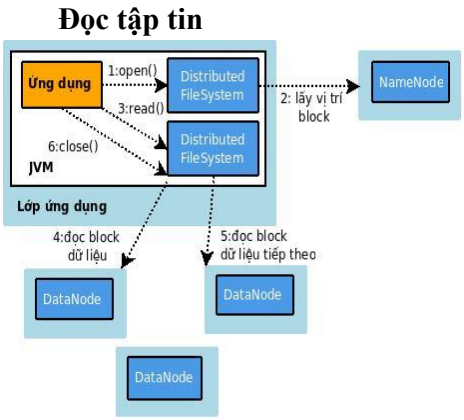
HDFS được thiết kế để giải quyết các vấn đề mà các hệ thống tập tin phân tán đang gặp phải:

- Khả năng lưu trữ lên đến hàng terabyte hoặc petabyte tùy thuộc vào số lượng máy tham gia vào hệ thống. HDFS cũng hỗ trợ kích thước tập tin lớn hơn nhiều so với NFS.

- HDFS tạo ra sự tin cậy cao khi lưu trữ dữ liệu. Nếu có một máy bị mất liên lạc vì một lý do nào đó, dữ liệu vẫn có thể được cung cấp toàn vẹn đến máy con.

- Do nằm trong gói sản phẩm Hadoop, HDFS cung cấp khả năng tích hợp cao nhất đến mô hình MapReduce, cho phép dữ liệu được đọc và ghi nội bộ trong quá trình thực hiện tính toán với khả năng cao nhất.

- *Cơ chế truy xuất dữ liệu*



Hình 2. *Quá trình đọc tập tin trong HDFS*

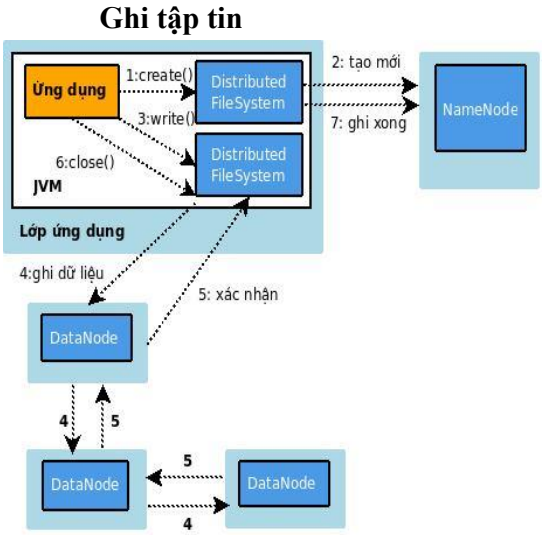
Bước 1: Ứng dụng gọi hàm open để mở tập tin cần đọc trên HDFS thông qua một đối tượng Distributed FileSystem.

Bước 2: Distributed FileSystem gọi đến Namenode bằng RPC để lấy vị trí các block dữ liệu đầu tiên trong tập tin cần đọc. Đối với mỗi block, Namenode trả về địa chỉ của tất cả các Datanode có chứa dữ liệu. Đối tượng Distributed FileSystem đồng thời trả về một đối tượng DFSInputStream cho ứng dụng.

Bước 3 + 4: Ứng dụng gọi hàm read(), đối tượng DFSInputStream kết nối đến Datanode gần nó nhất có chứa block dữ liệu đầu tiên, lấy dữ liệu và truyền về ứng dụng.

Bước 5: Khi đã truyền hết block dữ liệu, đối tượng DFSInputStream đóng kết nối với datanode hiện tại và tìm kiếm Datanode tốt nhất cho block tiếp theo.

Bước 6: Sau khi đọc hết dữ liệu trong tập tin, client gọi hàm close() để đóng tất cả các kết nối với Namenode và Datanode.



Hình 3. *Quá trình ghi tập tin trong HDFS*

Bước 1: Ứng dụng gọi hàm create() để tạo đối tượng Distributed FileSystem.

Bước 2: Distributed FileSystem gọi đến NameNode thông qua phương thức RPC để tạo một tập tin mới và chưa có block nào liên quan đến nó. NameNode sẽ thực hiện nhiều bước kiểm tra để chắc chắn rằng tập tin này chưa tồn tại và ứng dụng có đủ quyền để tạo mới tập tin này trên hệ thống.

Bước 3: Ứng dụng ghi dữ liệu qua đối tượng DFSOutputStream (được trả về bởi Distributed FileSystem). Dữ liệu được tách thành các gói nhỏ và xếp trong một hàng đợi.

Bước 4: Với mỗi block dữ liệu cần được ghi, DFSOutputStream yêu cầu NameNode trả về một danh sách các

DataNode để lưu trữ. Dữ liệu từ ứng dụng chỉ cần được ghi lên DataNode gần nhất, các DataNode sẽ tự động gửi tiếp dữ liệu nó vừa ghi đến node tiếp theo trong danh sách.

Bước 5: Khi DataNode đã hoàn tất việc ghi dữ liệu, một thông điệp sẽ được truyền đến Node trước đó. Đứng ở phía ứng dụng, việc ghi được xem là hoàn tất khi tất cả các Datanode đều ghi dữ liệu thành công.

Bước 6: Sau khi việc ghi dữ liệu hoàn tất, ứng dụng đóng kết nối với DataNode.

Bước 7: Ứng dụng thông báo cho NameNode việc ghi tập tin đã kết thúc.

2.2. Giải pháp sử dụng MapReduce để giải quyết vấn đề so trùng

2.2.1 Cơ sở dữ liệu

Tập dữ liệu đầu vào được xây dựng dưới dạng tập tin, và được tổ chức như sau:

Thông tin về sản phẩm được lưu trong file "product.dat" và có định dạng sau: *ArticleID:Title:Atract:Keyword*

Thông tin về thành viên xem sản phẩm và đánh giá sản phẩm được lưu trong file "user.dat" và có thông tin định dạng sau: *UserID::ArticleID:ratting:timeStamp*

Cấu trúc của thư mục được tạo trên hệ thống file Hadoop (HDFS) như sau:

Thư mục *input* là nơi lưu trữ các tập tin dữ liệu được sao chép trực tiếp từ hệ thống lưu trữ các sản phẩm, đồng thời là dữ liệu đầu vào cho tác vụ MapReduce.

Thư mục *output* sẽ chứa đầu ra của mỗi tác vụ MapReduce, ngay sau khi một tác vụ MapReduce kết thúc, các kết quả sẽ chuyển đổi và lưu vào cơ sở dữ liệu và hiển thị cho người dùng ở ứng dụng đầu cuối.

2.2.2. Xây dựng giải thuật so trùng dựa trên mô hình lập trình MapReduce

Bước 1: Dữ liệu đầu vào bao gồm các tập tin chứa thông tin sản phẩm, thông tin khách hàng. Đầu tiên, dữ liệu được đẩy vào hệ thống ở dạng tập tin HDFS. Dữ liệu được lọc cho kết quả dữ liệu mới theo yêu cầu xử lý của giải thuật. Kết quả thu được là tập vector các sản phẩm gồm từ khóa và trọng số của nó. Quá trình xử lý này được thực hiện bởi một quy trình MapReduce như sau:

- Dữ liệu đầu là các sản phẩm được đưa vào hàm Map để lọc ra các từ khóa và tần xuất xuất hiện của nó trong sản phẩm.
- Hàm Reduce sẽ gom lại và trả ra kết quả là danh sách các từ khóa và trọng số tương ứng của trong sản phẩm.

Công việc 1 (Job 1): Tần xuất từ trong tài liệu

Setup(stopwords)

Mapper

- Input: (productid,product)
- Xử lý: duyệt và đếm số lượng từng từ trong sản phẩm bỏ qua stopwords, số, từ bắt đầu ký tự đặc biệt.
- Output: ((word,productid),n)

Reducer

- Output: ((word,productid),n)

Công việc 2 (Job 2): Đếm số từ trong tài liệu

Mapper

- Input: ((word,productid),n)
- Output: (productid,(word,n))

Reducer

- Output: (productid,(word,n))

Bước 2: Thực hiện so trùng giữa các vectơ người dùng để tìm ra kết quả những người cùng sở thích bằng phương pháp so trùng profile:

Setup(userList,matrix(user/productList), matrix(productid,(keyword,w)))

Từ tập dữ liệu đầu vào chúng ta xây dựng được danh sách người dùng, người dùng đã xem sản phẩm nào, sản phẩm và trọng số của các từ khóa trên tất cả các sản phẩm.

Mapper

- Input: (userid(a),productid)

o Duyệt 1 nữa danh sách người dùng B, tính độ tương quan của userid(a) với mỗi B[i]

o Output: ((userid1,userid2),w)

Reducer

o Output: ((userid1,userid2),w)

Kết quả được sắp sếp lại và lấy ra danh sách những người dùng có độ tương tự cao nhất.

2.2.3 Minh họa giải thuật so trùng

Giả sử ta xét một tập dữ liệu gồm 9 sản phẩm (C1-C9). Các sản phẩm này có mối quan hệ với tập từ khóa (k1-k7) xác định trước được biểu diễn ở ma trận sau (giá trị biểu diễn là số lần xuất hiện của từ khóa).

Bảng 1. *Tập sản phẩm và từ khóa*

	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**
k1	n_{11}	n_{12}	n_{13}	n_{14}	n_{15}	n_{16}	n_{17}	n_{18}	n_{19}
k2	n_{21}	n_{22}	n_{23}	n_{24}	n_{25}	n_{26}	n_{27}	n_{28}	n_{29}
k3	n_{31}	n_{32}	n_{33}	n_{34}	n_{35}	n_{36}	n_{37}	n_{38}	n_{39}
k4	n_{41}	n_{42}	n_{43}	n_{44}	n_{45}	n_{46}	n_{47}	n_{48}	n_{49}
k5	n_{51}	n_{52}	n_{53}	n_{54}	n_{55}	n_{56}	n_{57}	n_{58}	n_{59}
k6	n_{61}	n_{62}	n_{63}	n_{64}	n_{65}	n_{66}	n_{67}	n_{68}	n_{69}
k7	n_{71}	n_{72}	n_{73}	n_{74}	n_{75}	n_{76}	n_{77}	n_{78}	n_{79}

Từ ma trận trên áp dụng công thức tính toán độ tương đồng (E. Gabrilovich and S. Markovitch, 2007, Rajesh Thiagarajan, Geetha Manjunath, and M. Stumptner, 2008) tính trọng số xuất hiện từ khóa thông qua tần số xuất hiện trong sản phẩm tính được từ ma trận trên, kết quả thu được 1 tập vectơ các sản phẩm chứa trọng số xuất hiện của từ khóa dưới dạng sau:

$$C=\{(w_0,k_0),\ldots,(w_i,k_i)\}$$

Kết hợp kết quả có được ở trên với tập dữ liệu những người dùng tham gia vào các sản phẩm được biểu diễn sau:

$$U=\{(w_0,C_0),\ldots,(w_i,C_i)\}$$

Từ 2 tập vectơ chúng tôi xây dựng một ma trận biểu diễn quan hệ giữa người dùng và từ khóa theo trọng số xuất hiện từ khóa trong sản phẩm.

Bảng 2. *Tập người dùng và từ khóa*

	k1	**k2**	**k3**	**k4**	**k5**	**k6**	**k7**
u1	w_{11}	w_{12}	w_{13}	w_{14}	w_{15}	w_{16}	w_{17}
u2	w_{21}	w_{22}	w_{23}	w_{24}	w_{25}	w_{26}	w_{27}
u3	w_{31}	w_{32}	w_{33}	w_{34}	w_{35}	w_{36}	w_{37}
u4	w_{41}	w_{42}	w_{43}	w_{44}	w_{45}	w_{46}	w_{47}
u5	w_{51}	w_{52}	w_{53}	w_{54}	w_{55}	w_{56}	w_{57}
u6	w_{61}	w_{62}	w_{63}	w_{64}	w_{65}	w_{66}	w_{67}

Áp dụng phương pháp so trùng tìm ra danh sách khách hàng tương đồng nhau sở thích thông qua phép tính độ tương tự của ma trận trên và độ tương tự của các từ khóa từ ma trận từ khóa và sản phẩm ta được kết quả sau:

Bảng 3. *Tập biểu diễn mối quan tâm giữa các người dùng*

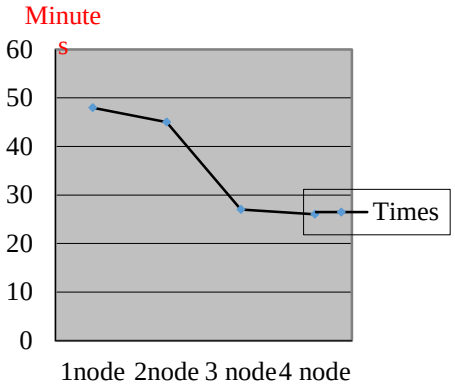
	u1	**u2**	**u3**	**u4**	**u5**	**u6**
u1	1	w_{12}	w_{13}	w_{14}	w_{15}	w_{16}
u2	w_{21}	1	w_{23}	w_{24}	w_{25}	w_{26}
u3	w_{31}	w_{32}	1	w_{34}	w_{35}	w_{36}
u4	w_{41}	w_{42}	w_{43}	1	w_{45}	w_{46}
u5	w_{51}	w_{52}	w_{53}	w_{54}	1	w_{56}
u6	w_{61}	w_{62}	w_{63}	w_{64}	w_{65}	1

Kết quả thu được ta lọc chọn ra những người có trọng số cao nhất được xem ra là những người có cùng mối quan tâm.

3. Kết quả và thảo luận

3.1. Triển khai và kết quả đạt được

Hệ thống được triển khai thử nghiệm trên 4 máy tính (CPU: 2.4Ghz, RAM: 4GB HDD: 200GB, Hadoop-0.20.2, Java JDK 1.6, Ubuntu 10.4 OS), sử dụng cơ sở dữ liệu cho 5000 sản phẩm và 10000 người dùng. Kết quả đo được theo thời gian thực thi như sau:



Hình 4. *Thời gian thực thi của giải thuật*

Từ kết quả đạt được, chúng tôi rút ra một số nhận xét sau:

- Để có thể tìm chính xác những người có cùng sở thích cần phải có tập từ khóa đầy đủ và tốt được trích rút ra từ các sản phẩm.
- Số lượng node lớn thì thời gian thực thi giảm đáng kể. Khi số lượng node đủ lớn cho tập dữ liệu thì việc tăng số lượng node không làm giảm đáng kể thời gian thực thi.

3.2. Kết luận

Bài báo này đi sâu vào nghiên cứu thuật toán so trùng và cách tổ chức lưu trữ dữ liệu trong một hệ thống phân tán lấy trọng tâm là MapReduce. Điều quan trọng nhất rút ra được là hiệu quả của thuật toán so trùng được xây dựng trên mô hình lập trình MapReduce, công thức tính toán mà MapReduce đưa ra có thể áp dụng được cho khá nhiều bài toán của hệ thống lớn trong thực tế. Hệ thống tự động tìm kiếm những người có cùng mối quan tâm trong thương mại điện tử thông qua các sản phẩm họ đã tương tác với hệ thống❑

TÀI LIỆU THAM KHẢO

E. Gabrilovich and S. Markovitch. (2007). Computing semantic relatedness using wikipedia-based explicit semantic analysis, presented at IJCAI.

Jeffrey Dean and Sanjay Ghemawat. (2004). MapReduce: Simplied Data Processing on Large Clusters.

Jeffrey Dean and Sanjay Ghemawat. (2004). Distributed Programming with MapReduce.

T. K. Landauer, P. W. Foltz, and D. Laham. (1998). An Introduction to Latent Semantic Analysis, Discourse Processes, vol. 25, pp. 259-284.

Rajesh Thiagarajan, Geetha Manjunath, and M. Stumptner. (2008). Finding Experts By Semantic Matching of User Profiles, presented at Personal Identification and Collaborations: Knowledge Mediation and Extraction.

Application of Map-Reduce model to search the customers of similar demands for products in e-commerce

Tran Xuan Hiep, Phan Thi Thanh Thuy*
Phu Yen University
**Email: thuycdsppy@yahoo.com*
Received:May 07, 2020; Accepted: May 28, 2021

Abstract

Finding and collaborating with customers of similar demands for products has become more and more important in the e-commerce. The essential data storage such as people with similar concerns, suitable personal profiles and their interactions in the community is chosen for analysis. This paper introduces the method of matching personal profiles by applying popular algorithms to the Map-Reduce model which is based on Hadoop framework. The purpose of this method is to create an effective tool which can help potential users save their time in finding an optimal list of people with similar interests. Then they can help companies to do better in consulting business to their customers.

Key words: *Map-Reduce, Hadoop, profile matching, semantic analysis*