

# Giải thuật K-Means xử lý song song với mô hình MapReduce

Trần Thiên Thành

Khoa Công nghệ thông tin

Đại học Quy Nhơn

Email: [thanhtranthien@gmail.com](mailto:thanhtranthien@gmail.com)

Nguyễn Thị Như

Khoa Khoa học Tự nhiên và Công nghệ

Trường Đại học Tây Nguyên

Email: [nhudoan2002@gmail.com](mailto:nhu doan2002@gmail.com)

Tóm tắt nội dung – K-Means là giải thuật phân cụm dữ liệu khá nổi tiếng và được sử dụng phổ biến trong lĩnh vực khai phá dữ liệu, nó cho phép chia n đối tượng thành k cụm sao cho tổng bình phương khoảng cách giữa các đối tượng đến tâm nhóm là nhỏ nhất. Tuy nhiên, phương pháp này còn nhiều hạn chế do việc tính khoảng cách giữa các đối tượng đến các tâm và việc xác định lại tâm được thực hiện lặp lại nhiều lần khiến giải thuật mất nhiều thời gian xử lý và khó triển khai trên tập dữ liệu lớn. Nhằm cải tiến nhược điểm trên, trong bài báo này, chúng tôi chọn giải pháp triển khai phân cụm dữ liệu bằng giải thuật K-Means dựa trên mô hình lập trình song song MapReduce được cài đặt trên hệ thống Hadoop. Cuối bài báo chúng tôi đưa ra một số kết quả thực nghiệm cho thấy giải thuật phân cụm K-Means trên mô hình MapReduce đạt được hiệu suất cao hơn khi phân loại tự động dữ liệu lớn và nó chứng tỏ tính hiệu quả và tính chính xác của giải thuật.

Keywords – Data mining, K-Means, MapReduce;

## I. MỤC MỞ ĐẦU

Phân cụm dữ liệu là một phương thức khai phá dữ liệu quan trọng. Việc phân tích gom cụm là tìm hiểu giải thuật và phương thức phân loại đối tượng. Một cụm là tập các đối tượng tương tự nhau hoặc là tập các thực thể hoặc nhóm các đối tượng mà các thực thể trong cùng cụm phải giống nhau; các thực thể ở các cụm khác nhau thì không giống nhau. Mỗi thực thể có thể có nhiều thuộc tính hoặc tính năng giống nhau được đo sự giống nhau dựa trên các thuộc tính hoặc tính năng giống nhau.

Cùng với việc phát triển nhanh chóng của Internet, một lượng lớn tài liệu cần được xử lý trong một thời gian ngắn. Việc lưu trữ và tính toán dữ liệu lớn trong hệ thống phân tán là một phương thức đang được triển khai. Trong tính toán phân tán thì cần chia nhỏ công việc để mỗi công việc được xử lý trên một máy tính.

MapReduce là mô hình lập trình song song bắt nguồn từ lập trình chức năng và được Google đề xuất để xử lý lượng dữ liệu lớn trong môi trường phân tán. Dự án Hadoop cung cấp hệ thống file phân tán (HDFS) và hỗ trợ mô hình MapReduce. Hadoop cho phép các ứng dụng làm việc với hàng ngàn nodes với hàng petabyte dữ liệu.

Trong bài báo này, chúng tôi đề cập đến việc triển khai giải thuật gom cụm K-means trên mô hình lập trình song song MapReduce dựa trên nền tảng Hadoop [4] và đưa ra một số kết quả thử nghiệm gom cụm phân tán dựa trên mô hình này.

## II. MÔ HÌNH MAPREDUCE

MapReduce là mô hình lập trình được sử dụng để tính toán tập dữ liệu lớn. Một tiến trình xử lý MapReduce cơ bản có thể tính toán đến terabytes hoặc petabyte dữ liệu trên hệ thống được kết nối thành cụm các nodes. Dữ liệu được chia thành các mảnh nhỏ rồi đưa vào các nodes độc lập, vì vậy số lượng và kích thước của các mảnh phụ thuộc vào số nodes được kết nối trong mạng [4].

Các bước Map và Reduce được thiết kế tách biệt, riêng rẽ và hoàn toàn độc lập. Mỗi bước Map và Reduce được thực hiện song song trên các cặp dữ liệu (key, value). Do đó, chương trình được chia thành hai giai đoạn riêng biệt là Map và Reduce [1].

Bộ ánh xạ (Mapper): xử lý tập dữ liệu đầu vào dưới dạng (key, value) và tạo ra tập dữ liệu trung gian là cặp (key, value). Tập dữ liệu này được tổ chức cho hoạt động của Reduce. Bộ ánh xạ Map thực hiện các bước như sau:

**Bước 1:** Ánh xạ cho mỗi nhóm dữ liệu đầu vào dưới dạng (key, value).

**Bước 2:** Thực thi việc Map, xử lý cặp (key, value) để tạo (key, value) mới, công việc này được gọi là chia nhóm, tức là tạo các giá trị liên quan tương ứng với cùng các từ khóa.

**Bước 3:** Đầu ra của bộ ánh xạ được lưu trữ và định vị cho mỗi bộ Reducer. Tổng các khối và số công việc reduce là như nhau.

Bộ Reducer: Trộn tất cả các phần tử value có cùng key trong tập dữ liệu trung gian do Map tạo ra để tạo thành tập trị nhỏ hơn và quá trình này được lặp lại cho tất cả các giá trị key khác nhau. Việc truyền dữ liệu được thực hiện giữa Map và Reduce.

Lập trình viên cần cài đặt chính xác (key, value), MapReduce sẽ gom cụm một cách tự động và chính xác các values với cùng key. Các bước của bộ Reducer:

**Bước 1 (Trộn):** Đầu vào của Reducer là đầu ra của Mapper. Giai đoạn này, MapReduce sẽ gán khối liên quan cho bộ Reducer.

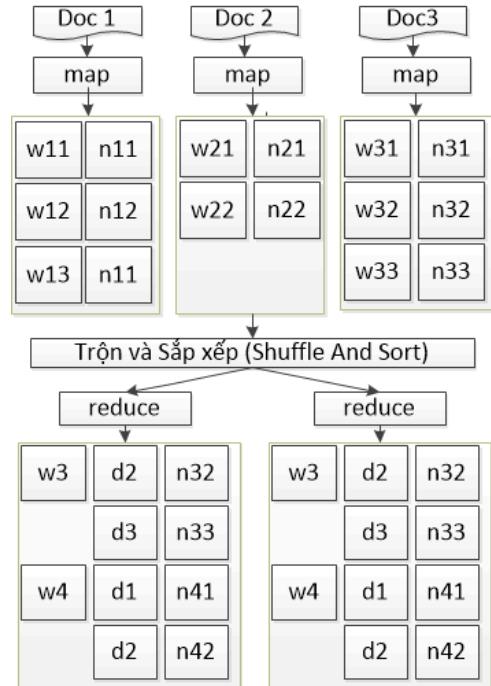
**Bước 2 (Sắp xếp):** Giai đoạn này, đầu vào của bộ Reducer được gom theo key (do đầu ra của bộ ánh xạ khác nhau có thể có cùng key). Hai giai đoạn Trộn và sắp xếp được đồng bộ hóa.

**Bước 3:** Tạo bộ so sánh để nhóm các keys trung gian lần hai nếu quy luật gom nhóm khác với trước đó.

Trong một tiến trình của MapReduce, các công việc Map chạy song song, các công việc Reduce chạy song song. Tuy nhiên, các công việc Map và Reduce được thực hiện tuần tự, tức là Map phải hoàn thành và chuyển dữ liệu cho Reduce. Dữ liệu đầu vào và đầu ra được lưu trữ trong hệ thống file.

### III. GIẢI THUẬT K-MEANS

Có nhiều giải thuật và kỹ thuật gom cụm dữ liệu được sử dụng. Giải thuật gom cụm K-Means được sử dụng để phân loại tập dữ liệu phi cấu trúc hoặc bán cấu trúc, là một trong những phương thức phân loại dữ liệu phổ dụng và hiệu quả do tính đơn giản và khả năng kiểm soát tập dữ liệu. Giải thuật K-Means [5] lấy tham số đầu vào k và chia một tập n đối tượng thành k cụm dựa vào việc đo độ tương tự. Trị trung bình hay các tâm là tổng độ đo tương tự của các đối tượng dữ liệu trong cùng cụm. Các đối tượng còn lại sẽ được gán cho cụm gần nhất,



Hình 1. Quá trình xử lý các tài liệu trên mô hình MapReduce

dựa vào khoảng cách tới các tâm khác nhau, sau đó tính lại các tâm. Quá trình này được lặp lại cho đến khi hàm điều kiện hội tụ.

Giải thuật được mô tả như sau:

Input: số cụm k và n tài liệu.

Output: k cụm.

**Bước 1:** chọn ngẫu nhiên k tài liệu từ n tài liệu để khởi tạo làm tâm các cụm

**Bước 2:** tính khoảng cách từ các tài liệu còn lại đến các tâm của các cụm, gán mỗi tài liệu đó cho cụm gần nhất.

**Bước 3:** tính toán và điều chỉnh tâm các cụm.

**Bước 4:** lặp bước 2 và bước 3 cho đến khi hàm điều kiện hội tụ. Kết thúc chương trình.

Tuy nhiên, giải thuật này tồn tại một số nhược điểm như việc tính khoảng cách từ một phần tử đến tâm và việc tính toán và điều chỉnh tâm các cụm được thực hiện lặp lại sau mỗi bước gán một phần tử cho một cụm dẫn đến tiêu tốn nhiều tài nguyên hệ thống và thời lượng chạy chương trình sẽ lâu. Do vậy, giải thuật này phù hợp với lượng dữ liệu vừa và nhỏ, để triển khai cho tập dữ liệu lớn thì nó cần được cải thiện và thực hiện trên mô hình phù hợp hơn để hạn chế các nhược điểm nói trên [6].

#### IV. PHÂN CỤM DỮ LIỆU VỚI GIẢI THUẬT K-MEANS SONG SONG TRÊN MÔ HÌNH MAPREDUCE

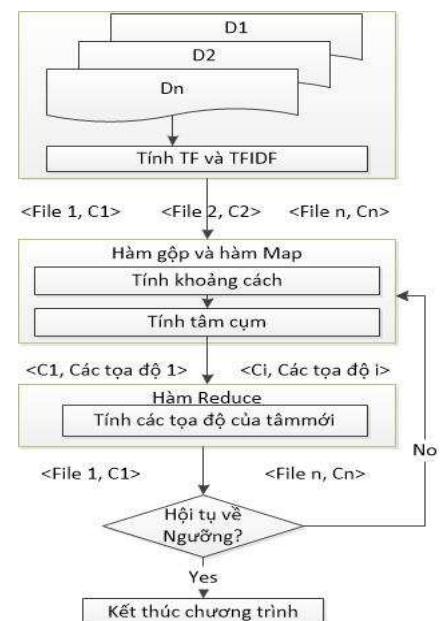
Giải thuật K-Means dựa trên mô hình MapReduce làm việc như sau:

**Bước 1.** Giai đoạn đầu là tiền xử lý tài liệu. Chia tập tài liệu D thành m tập con. Có hai công việc MapReduce trong giai đoạn này, đầu tiên là phải tính toán các tham số cho bước tiếp theo như đếm từ, tính TF,.., tiếp theo là tính TFIDF (con số thể hiện mức độ quan trọng của từ trong một tài liệu trên tập các tài liệu) của mỗi tài liệu. Kết thúc giai đoạn này tài liệu được đánh chỉ số cũng như vector trọng số của nó cũng đã hoàn chỉnh, đã chọn được k tài liệu làm tâm, xác định kích thước mảnh dữ liệu, ngừng hội tụ để chương trình kết thúc.

**Bước 2:** Giai đoạn thứ hai là hàm map, đọc dữ liệu đầu vào và tính khoảng cách tới mỗi tâm. Với mỗi tài liệu, nó tạo ra cặp <key (chỉ số cụm), value(tọa độ của tài liệu)>. Rất nhiều dữ liệu được tạo ra trong giai đoạn này, hàm gộp có thể được sử dụng để giảm kích thước trước khi gửi đến Reduce.

Hàm trộn được mô tả như sau:

Hàm trộn tính trị trung bình của các tọa độ cho mỗi id cụm, cùng với số tài liệu. Tất cả dữ liệu cùng cụm hiện tại được gửi tới một reducer.



Hình 2. Quá trình xử lý song song giải thuật K-Means

**Bước 3:** Giai đoạn thứ 3 là hàm reduce, hàm này tính tọa độ mới cho tâm các cụm. Dữ liệu đầu ra này được ghi vào tập tin của cụm bao gồm: số lần lặp, id cụm, các tọa độ của tâm cụm, kích thước của cụm.

**Bước 4:** Cuối cùng các tọa độ của cụm mới được so sánh với các tọa độ ban đầu. Nếu hàm điều kiện hội tụ thì chương trình kết thúc và ta tìm được các cụm. Nếu không, sử dụng các tâm của cụm mới thực hiện và lặp lại từ bước 2 đến bước 4.

## V. THỦ NGHIỆM

Chúng tôi thực hiện thử nghiệm giải thuật K-Means trên cùng hệ thống máy tính với cùng bộ dữ liệu mẫu là các tài liệu tin tức của Reuters tại <http://www.daviddlewis.com/resources/testcollections/reuters21578/reuters21578.tar.gz>. Điểm khác biệt là hệ thống triển khai chạy giải thuật k-means gốc, sau đó tiến hành trên mô hình Hadoop Mapreduce cho một node và sau đó là 2 nodes.

Chúng tôi cài đặt giải thuật K-Means và K-Means trên MapReduce (MRKmeans) trên máy laptop Dell Intel Core i7-2620M [CPU@2.70GHzx4](#), RAM 4GB.

Dữ liệu thử nghiệm là tập dữ liệu Reuters có kích thước 16.7 MB chứa 21.578 item (mỗi item tương ứng là một tài liệu) và các tài liệu được lưu trong 22 tập tin.

Giải thuật K-Means gốc được cài đặt trên ngôn ngữ C# và chạy trên cùng cấu hình máy.

Giải thuật MRKmeans được triển khai trên hệ điều hành Ubuntu 14.04 LTS 64 bit, java 1.7.0\_75 và SSH, Hadoop 2.6.0, Maven [3], Mahout 3.2.1 [2]

Chúng tôi sử dụng Apache Mahout để thực hiện quá trình phân cụm như các tham số tiền xử lý dữ liệu: tách và đánh chỉ mục làm cơ sở để tạo vectors cho các tài liệu, sau đó sử dụng trình điều khiển Hadoop đã tích hợp trong Mahout để chạy giải thuật K-Means với các tham số k=20 [2]:

### Bước 1: Tiền xử lý dữ liệu

Chương trình sẽ trích xuất dữ liệu trong 22 tập tin ban đầu thành 21.578 tập tin tài liệu và đánh số thứ tự cho các tài liệu đó.

Tiếp theo là tạo vector cho tài liệu từ tập tài liệu vừa được trích xuất ở trên bằng cách:

- Tạo tài liệu theo định dạng tuân tự gồm cặp <tên-file, nội\_dung>
- Dữ liệu trên được sử dụng để tạo vector cho tài liệu dựa trên trọng số TFIDF. Kết quả trả về là (docID, TF-IDF Vector)

Từ kết quả trên, tiến hành chọn k=20 vector ngẫu nhiên làm tâm khởi điểm.

**Bước 2:** Dữ liệu đầu vào cho hàm Map là 21.578 records. Tính khoảng cách từ các vector (21.558) đến các tâm (20), việc tính toán này được phân phối để thực hiện song song. Kết quả bước này là 20 cặp (key, value) tương ứng (chỉ số cụm, tọa độ các tài liệu).

**Bước 3:** Tính lại tọa độ các tâm. Dữ liệu đầu vào cho hàm Reduce là 20 cặp (key, value) do Map chuyển qua, dữ liệu được phân chia để thực hiện song song trên cả 2 node sau đó gộp lại. Kết quả hàm này ghi lại số lần lặp, chỉ số cụm, tọa độ của cụm, kích thước cụm.

**Bước 4:** So sánh tọa độ cụm mới với tọa độ cụm trước đó. Nếu điều kiện hội tụ thỏa mãn (ngưỡng hội tụ =0.5) thì dừng chương trình và cho kết quả 20 cụm là 20 tập tin chứa tọa độ của các vector của cụm. Ngược lại, điều kiện hội tụ chưa thỏa mãn thì lặp lại quá trình Map và Reduce từ bước 2 đến bước 4.

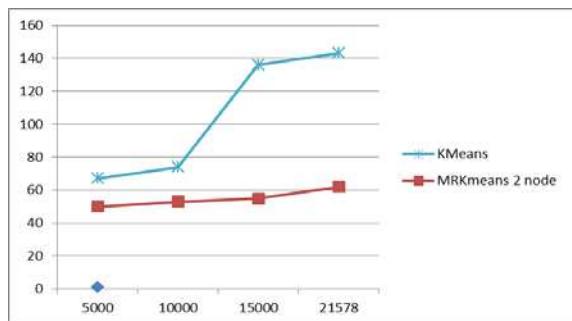
Giải thuật được chạy nhiều lần với cùng tập dữ liệu và cùng số cụm, kiểm tra kết quả phân cụm và thời gian xử lý.

Triển khai hệ thống Hadoop Mapreduce trên 2 node có cùng cấu hình phần cứng, một máy vừa đóng vai trò máy chủ chứa dữ liệu và chịu trách nhiệm điều phối hoạt động được gọi là NameNode, máy còn lại đóng vai trò DataNode sẽ thực thi nhiệm vụ do máy chủ gán.

Kết quả của việc chạy giải thuật K-Means và MRKmeans trên tập dữ liệu Reuters được thể hiện như sau:

Table 1. Bảng thời gian chạy các giải thuật Kmeans và MRKmeans

Số mẫu (tài liệu)	Kmeans (giây)	MRKmeans 2 node (giây)
5.000	67	50
10.000	98	53
15.000	145	55
21.578	178	62



Hình 3. So sánh thời gian chạy của hai thuật toán Kmeans và MRKmeans

## VI. Kết luận

Từ kết quả nghiên cứu các thuật toán K-Means, thuật toán ban đầu không thể thực hiện với tập dữ liệu lớn hàng terabyte. Do đó, chúng tôi đề xuất giải pháp phân cụm dữ liệu văn bản lớn trên mô hình MapReduce, với kỹ thuật này hiệu suất, thời lượng và tính ổn định của việc phân cụm dữ liệu lớn được cải thiện và hiệu quả hơn nhiều so với giải thuật K-Means ban đầu. Kết quả thực nghiệm cũng cho thấy tính hiệu quả của việc phát triển giải thuật K-Means trên mô hình MapReduce so với thuật toán K-Means ban đầu.

## TÀI LIỆU THAM KHẢO

- [1] J. D. a. S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," 2004.
- [2] "Apache Mahout," [Online]. Available: <http://mahout.apache.org>. [Accessed 15 4 2015].
- [3] "Maven," [Online]. Available: <http://maven.apache.org/>. [Accessed 10 4 2015].
- [4] W. Tom, Hadoop\_The definitive Guide - 3rd edition, O'reilly, 2012.
- [5] J. A. Hartigan. and. M. A. Wong, "A k-means clustering algorithm," *Applied Statistics*, vol. 28, pp. 100-108, 1979.
- [6] D. M. a. N. S. Kehar Singh, "Evolving limitations in K-means algorithm in data mining and their removal," *International Journal of Computational Engineering & Management*, vol. 12, 2011.