

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

VIỆN TRÍ TUỆ NHÂN TẠO

____***)____

**BÁO CÁO MÔN HỌC
KỸ THUẬT VÀ CÔNG NGHỆ DỮ LIỆU LỚN**

ĐỀ TÀI

**HỆ THỐNG ĐỀ XUẤT PHIM DỰA TRÊN
ITEM COLLABORATIVE FILTERING VÀ HADOOP MAPREDUCE**

Nhóm sinh viên thực hiện:

1. Nguyễn Mạnh Cường - 22022516
2. Bùi Đức Mạnh - 22022602
3. Lê Việt Hùng - 22022666

Giảng viên hướng dẫn:

TS. Trần Hồng Việt
ThS.Ngô Minh Hương

HÀ NỘI, 5/2024

MỞ ĐẦU

Công nghệ Big data đã đạt đến đỉnh cao trong việc thực hiện các chức năng của nó. Trong tháng 8/2015 Big data đã vượt ra khỏi bảng xếp hạng những công nghệ mới nổi Cycle Hype của Gartner và tạo ra một tiếng vang lớn cho xu hướng công nghệ của thế giới. Big data chứa trong mình rất nhiều thông tin quý giá mà nếu mà trích xuất thành công, nó sẽ giúp rất nhiều trong nhiều lĩnh vực như y tế, giao thông, giáo dục, ...

Chính vì thế những framework giúp việc xử lý BIGDATA cũng đang ngày càng được xử lý và phát triển mạnh. Một trong những công nghệ cốt lõi cho việc lưu trữ và truy cập số lượng lớn dữ liệu là Hadoop - một framework giúp lưu trữ và xử lý Big data áp dụng MapReduce.

Từ đó, chúng em đã chọn đề tài: "**Hệ thống đề xuất phim dựa trên Item Collaborative Filtering và Hadoop Mapreduce**" để làm báo kết thúc môn học của mình.

Báo cáo gồm 4 chương

- **Chương 1:** Tổng quan về dữ liệu lớn
- **Chương 2:** Xây dựng hệ thống gợi ý phim bằng thuật toán Item-based Collaborative Filtering
- **Chương 3:** Ứng dụng Hadoop MapReduce với Item-based Collaborative Filtering trong Recommendation System
- **Chương 4:** Kết luận và hướng phát triển

Mục lục

MỞ ĐẦU	2
1 TỔNG QUAN VỀ DỮ LIỆU LỚN	4
1.1 Định nghĩa	4
1.2 Nguồn hình thành của dữ liệu lớn	4
1.3 Đặc trưng 5V của dữ liệu lớn	5
1.4 Tổng quan về Hadoop	5
1.5 Tổng quan về MapReduce	6
2 XÂY DỰNG HỆ THỐNG GỢI Ý PHIM BẰNG THUẬT TOÁN ITEM-BASED COLLABORATIVE FILTERING	7
2.1 Giới thiệu thuật toán	7
2.2 Triển khai thuật toán	7
2.3 Ví dụ minh họa thuật toán	8
3 ỨNG DỤNG HADOOP MAPREDUCE VỚI ITEM-BASED COLLABORATIVE FILTERING TRONG RECOMMENDATION SYSTEM	10
3.1 Ý tưởng MapReduce Item-based Collaborative Filtering	10
3.2 Lưu đồ của thuật toán MapReduce Item-based Collaborative Filtering	10
3.3 Triển khai thuật toán MapReduce Item-based Collaborative Filtering	11
3.4 Demo thuật toán MapReduce Item-based Collaborative Filtering	14
4 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	17
4.1 Kết luận	17
4.2 Hướng phát triển	17
SẢN PHẨM CỦA NHÓM	17
TÀI LIỆU THAM KHẢO	18
NHIỆM VỤ CỦA CÁC THÀNH VIÊN	18

Chương 1

TỔNG QUAN VỀ DỮ LIỆU LỚN

1.1 Định nghĩa

Dữ liệu lớn, hay còn được gọi là Big Data, là thuật ngữ chỉ đến tập hợp dữ liệu có kích thước lớn hoặc phức tạp đến mức các phương pháp truyền thống không đủ để xử lý. Theo định nghĩa của Wikipedia, dữ liệu lớn là một khái niệm chỉ đến các tập dữ liệu có quy mô hoặc độ phức tạp đáng kể, đặc biệt là khi các phương pháp truyền thống gặp khó khăn trong việc xử lý chúng.

Tổ chức nghiên cứu Gartner định nghĩa dữ liệu lớn là những nguồn thông tin có đặc điểm chung là kích thước lớn, tốc độ xử lý nhanh và định dạng dữ liệu đa dạng. Để khai thác hiệu quả dữ liệu lớn, cần áp dụng các phương pháp và công nghệ xử lý mới để thực hiện các quyết định, khám phá thông tin mới và tối ưu hóa các quy trình.

1.2 Nguồn hình thành của dữ liệu lớn

Dữ liệu lớn được tạo thành chủ yếu từ 6 nguồn chính, bao gồm:

- **Dữ liệu hành chính:** Phát sinh từ các hoạt động chương trình của tổ chức, bao gồm cả chính phủ và phi chính phủ. Ví dụ như hồ sơ y tế điện tử tại các cơ sở y tế, hồ sơ bảo hiểm, thông tin ngân hàng.
- **Dữ liệu thương mại:** Xuất phát từ các giao dịch giữa các thực thể kinh doanh. Ví dụ như giao dịch thẻ tín dụng, giao dịch trực tuyến, bao gồm cả các giao dịch từ thiết bị di động.
- **Dữ liệu từ cảm biến:** Bao gồm thông tin từ các thiết bị cảm biến như máy chụp hình vệ tinh, cảm biến đường, cảm biến khí hậu.
- **Dữ liệu từ thiết bị theo dõi:** Bao gồm thông tin thu thập từ các thiết bị theo dõi như điện thoại di động, GPS.
- **Dữ liệu từ hành vi trực tuyến:** Bao gồm thông tin từ các hoạt động trực tuyến như tìm kiếm sản phẩm, dịch vụ hoặc thông tin khác, cũng như việc đọc các trang web.
- **Dữ liệu từ ý kiến và quan điểm:** Bao gồm thông tin về ý kiến, quan điểm của cá nhân hoặc tổ chức trên các nền tảng mạng xã hội và các phương tiện truyền thông khác.



Hình 1.1: Minh họa nguồn gốc của dữ liệu

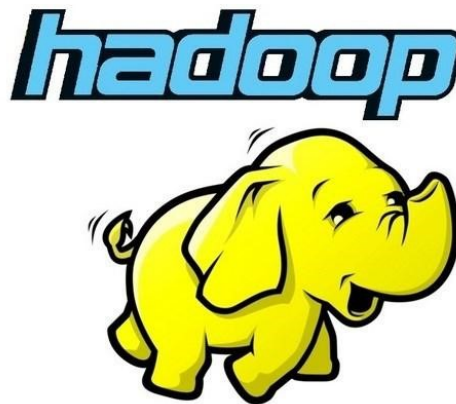
1.3 Đặc trưng 5V của dữ liệu lớn

Dữ liệu lớn có 5 đặc điểm cơ bản, được mô hình hóa qua mô hình 5V:

- **Volume (Khối lượng):** Dữ liệu lớn thường có khối lượng lớn, từ hàng terabyte đến petabyte. Để lưu trữ dữ liệu này, chúng ta phải sử dụng các công nghệ đám mây mới.
- **Velocity (Tốc độ):** Dữ liệu lớn được tạo ra và truy cập với tốc độ cực kỳ nhanh, đôi khi trong thời gian thực. Điều này đặc biệt quan trọng trong các lĩnh vực như Internet, Tài chính, Y tế.
- **Variety (Đa dạng):** Dữ liệu lớn không chỉ là dữ liệu có cấu trúc mà còn bao gồm các loại dữ liệu phi cấu trúc như tài liệu văn bản, hình ảnh, video, dữ liệu từ cảm biến vật lý. Việc phân tích và liên kết các loại dữ liệu này là thách thức lớn.
- **Veracity (Độ tin cậy/Chính xác):** Một trong những vấn đề phức tạp nhất của dữ liệu lớn là độ tin cậy và chính xác của dữ liệu. Với sự phát triển mạnh mẽ của phương tiện truyền thông xã hội và mạng xã hội, việc đảm bảo tính chính xác của dữ liệu trở nên khó khăn hơn.
- **Value (Giá trị):** Giá trị là yếu tố quan trọng nhất của dữ liệu lớn. Trước khi đầu tư vào dữ liệu lớn, chúng ta cần xác định rõ giá trị mà thông tin có thể mang lại. Dự báo chính xác và ứng dụng hiệu quả dữ liệu lớn có thể mang lại lợi ích lớn, giảm chi phí và tăng hiệu suất trong nhiều lĩnh vực.

1.4 Tổng quan về Hadoop

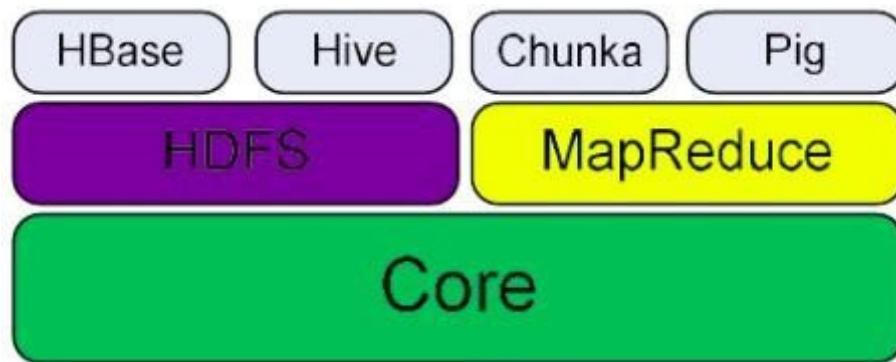
Hadoop là một framework nguồn mở của Apache, viết bằng Java, cho phép phát triển các ứng dụng phân tán xử lý dữ liệu lớn miễn phí. Nó được thiết kế để mở rộng từ một máy chủ đơn sang hàng ngàn máy tính khác có tính toán và lưu trữ cục bộ.



Hình 1.2: Biểu tượng của Hadoop

Hadoop có cấu trúc liên kết master-slave, với một node master và nhiều node slave. Node master gán tác vụ và quản lý tài nguyên, trong khi node slave lưu trữ dữ liệu thực. Hadoop gồm ba lớp chính:

- **HDFS (Hadoop Distributed File System):** Hệ thống file phân tán của Hadoop, cung cấp khả năng lưu trữ dữ liệu lớn trên nhiều node.
- **MapReduce:** Mô hình lập trình và xử lý dữ liệu song song của Hadoop.
- **YARN (Yet Another Resource Negotiator):** Một hệ thống quản lý tài nguyên trong Hadoop.



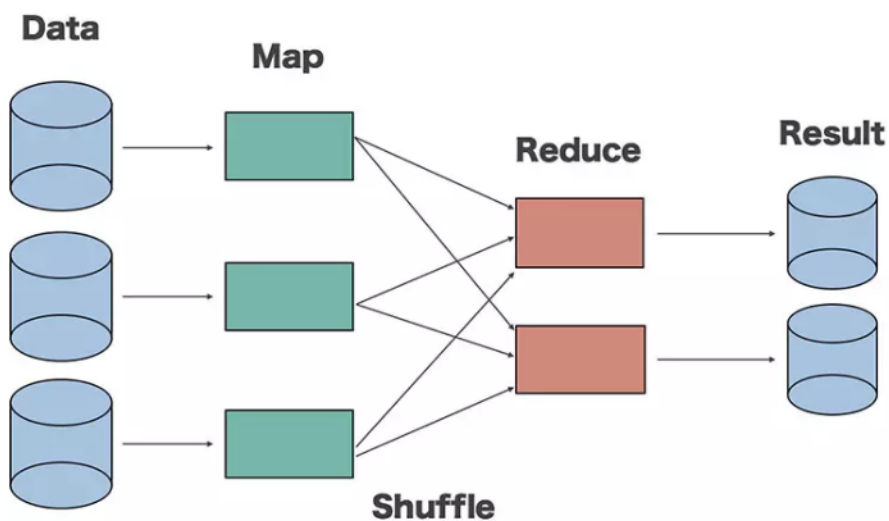
Hình 1.3: Thành phần của Hadoop

1.5 Tổng quan về MapReduce

MapReduce là một framework dùng để viết các ứng dụng xử lý song song một lượng lớn dữ liệu có khả năng chịu lỗi cao xuyên suốt hàng ngàn cluster (cụm) máy tính.

MapReduce thực hiện 2 chức năng chính đó là:

- **Map:** Sẽ thực hiện đầu tiên, có chức năng tải, phân tích dữ liệu đầu vào và chuyển đổi thành tập dữ liệu theo cặp key/value.
- **Reduce:** Sẽ nhận kết quả đầu ra từ tác vụ Map, kết hợp dữ liệu lại với nhau thành tập dữ liệu nhỏ hơn, tạo ra kết quả cuối cùng.



Hình 1.4: MapReduce

Chương 2

XÂY DỰNG HỆ THỐNG GỢI Ý PHIM BẰNG THUẬT TOÁN ITEM-BASED COLLABORATIVE FILTERING

2.1 Giới thiệu thuật toán

Thuật toán Item - based Collaborative Filtering (CF) là một phương pháp phổ biến trong hệ thống gợi ý, được sử dụng để đưa ra các đề xuất cá nhân hóa cho người dùng.

Thuật toán này phân tích ma trận người dùng - sản phẩm để xác định mối quan hệ giữa các sản phẩm khác nhau và sau đó sử dụng các mối quan hệ này để tính toán gián tiếp các đề xuất cho người dùng.

Phương pháp này giúp cải thiện hiệu quả và chất lượng của hệ thống gợi ý so với các thuật toán CF dựa trên người dùng.

2.2 Triển khai thuật toán

Triển khai thuật toán Item - based Collaborative Filtering bao gồm các bước chính sau đây:

Bước 1: Xây dựng ma trận đồng xuất hiện

Tạo ra ma trận đồng xuất hiện C kích thước $n \times n$, trong đó $C[i][j]$ là số lượng người dùng đã đánh giá cả sản phẩm i và j .

Bước 2: Chuẩn hóa ma trận

Tạo ra ma trận chuẩn hóa S kích thước $n \times n$, trong đó:

$$S[i][j] = \frac{C[i][j]}{\sum_{m \in M} C[m][j]}$$

Trong đó M là tập hợp tất cả các sản phẩm.

Bước 3: Xây dựng ma trận đánh giá

Tạo ra ma trận đánh giá R kích thước $n \times m$, trong đó:

$$R[i][j] = \sum_{m \in M} S[i][m] \times U[m][j]$$

Trong đó M là tập hợp tất cả các sản phẩm và thay $U[i][j] = ?$ thành $U[i][j] = 0$

Bước 4: Triển khai với Hadoop MapReduce

Để triển khai thuật toán này trên một tập dữ liệu lớn, chúng ta sử dụng Hadoop MapReduce để phân chia và xử lý dữ liệu song song. Quá trình triển khai bao gồm các bước sau:

- **Mapper:** Đọc vào các cặp key/value biểu diễn dữ liệu người dùng - sản phẩm, tính toán các cặp sản phẩm tương tự và lưu trữ các kết quả trung gian.
- **Reducer:** Tập hợp các kết quả từ Mapper, tính toán độ tương đồng giữa các sản phẩm và tổng hợp các dự đoán cuối cùng.

2.3 Ví dụ minh họa thuật toán

Để minh họa cho quá trình hoạt động của thuật toán CF dựa trên Item, chúng ta xem xét một ví dụ đơn giản:

Dữ liệu đầu vào: Ma trận người dùng - sản phẩm với các đánh giá từ 1 - 5. (? là sản phẩm mà người dùng đó chưa đánh giá)

	Sản phẩm 1	Sản phẩm 2	Sản phẩm 3	Sản phẩm 4
Người dùng A	5	3	?	4
Người dùng B	3	?	2	3
Người dùng C	4	?	4	?
Người dùng D	?	3	?	5
Người dùng E	?	?	?	?

Bước 1: Xây dựng ma trận đồng xuất hiện

Dựa trên dữ liệu đầu vào, chúng ta xây dựng ma trận đồng xuất hiện C như sau: (Ví dụ ta thấy rằng có 2 người cùng đánh giá sản phẩm 1 và 3 là Người dùng B và C nên $C[1][3] = C[3][1] = 2$)

	Sản phẩm 1	Sản phẩm 2	Sản phẩm 3	Sản phẩm 4
Sản phẩm 1	3	1	2	2
Sản phẩm 2	1	2	0	2
Sản phẩm 3	2	0	2	1
Sản phẩm 4	2	2	1	3

Bước 2: Chuẩn hóa ma trận

Dựa trên ma trận C, chúng ta chuẩn hóa thành ma trận S như sau: (Tổng cột sản phẩm 1 là 8 nên $S[1][1] = \frac{C[1][1]}{8} = \frac{3}{8} = 0.375$)

	Sản phẩm 1	Sản phẩm 2	Sản phẩm 3	Sản phẩm 4
Sản phẩm 1	0.375	0.2	0.4	0.25
Sản phẩm 2	0.125	0.4	0	0.25
Sản phẩm 3	0.25	0	0.4	0.125
Sản phẩm 4	0.25	0.4	0.2	0.375

Bước 3: Xây dựng ma trận đánh giá

Dựa trên ma trận S và ma trận đầu vào U, ta tính được ma trận R như sau: (Ví dụ ta tính đánh giá của người dùng A cho sản phẩm 3 là $R[1][3] = 5 \times 0.4 + 3 \times 0 + 0 \times 0.4 + 4 \times 0.2 = 2.8$)

	Sản phẩm 1	Sản phẩm 2	Sản phẩm 3	Sản phẩm 4
Người dùng A	3.25	3.8	2.8	3.5
Người dùng B	2.375	1.8	2.6	2.125
Người dùng C	2.5	0.8	3.2	1.5
Người dùng D	1.625	3.2	1.0	2.625
Người dùng E	?	?	?	?

Người dùng E chưa đánh giá sản phẩm nào nên ta không thể tính được đánh giá của người đó.

Kết luận

Dựa trên tính toán ở trên, dự đoán đánh giá của người dùng A cho Sản phẩm 3 là khoảng 2.8. Qua ví dụ minh họa này, chúng ta thấy rằng:

- Thuật toán CF dựa trên Item giúp xác định mối quan hệ giữa các sản phẩm dựa trên đánh giá của người dùng.
- Việc tính toán độ tương đồng giữa các sản phẩm và sau đó sử dụng nó để tính toán dự đoán đánh giá giúp cải thiện chất lượng gợi ý.
- Thuật toán này có thể mở rộng để xử lý dữ liệu lớn bằng cách sử dụng Hadoop MapReduce, giúp phân chia và xử lý dữ liệu song song, tối ưu hóa hiệu suất và tốc độ xử lý.

Chương 3

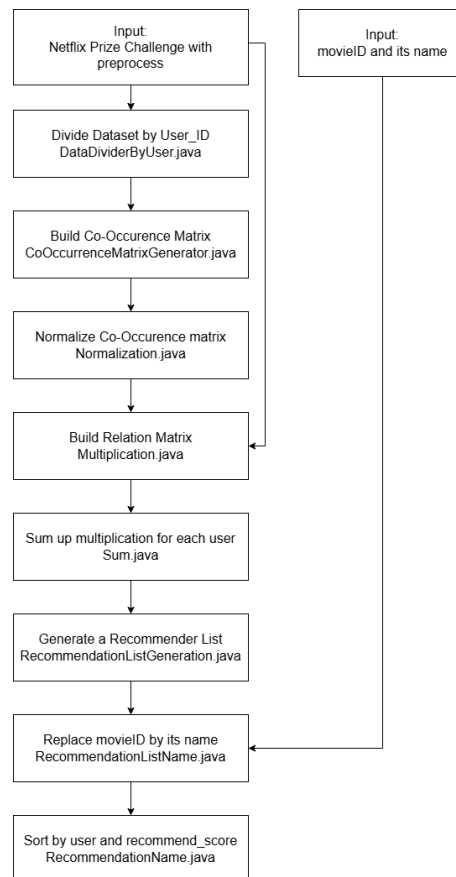
ỨNG DỤNG HADOOP MAPREDUCE VỚI ITEM-BASED COLLABORATIVE FILTERING TRONG RECOMMENDATION SYSTEM

3.1 Ý tưởng MapReduce Item-based Collaborative Filtering

Nhiệm vụ: MapReduce chia quá trình tính toán thành các bước nhỏ hơn (Map và Reduce), cho phép xử lý song song dữ liệu lớn.

Ý tưởng: Phân tán việc tính toán ma trận đồng xuất hiện và ma trận đánh giá qua nhiều máy tính
→ giảm thời gian xử lý và tăng hiệu suất
→ giải quyết các thách thức về kích thước dữ liệu và độ phức tạp của tính toán trong hệ thống gợi ý

3.2 Lưu đồ của thuật toán MapReduce Item-based Collaborative Filtering



Hình 3.1: Lưu đồ thuật toán Mapreduce Item-based Collaborative Filtering

3.3 Triển khai thuật toán MapReduce Item-based Collaborative Filtering

Giải pháp Mapreduce cho Item-based Collaborative Filtering

- **Dữ liệu đầu vào:** Là danh sách các hàng lưu dưới dạng file .txt. Mỗi hàng chứa thông tin về người dùng, sản phẩm và đánh giá của người dùng đó cho sản phẩm đó, cách nhau bởi dấu phẩy, được chuyển sang kiểu key-value làm đầu vào cho thuật toán
- **Triển khai:**
 1. Biểu diễn dữ liệu. Dữ liệu lưu trữ dưới dạng list các hàng. Mỗi hàng chứa thông tin về người dùng, sản phẩm và đánh giá của người dùng đó cho sản phẩm đó, cách nhau bởi dấu phẩy.
 2. Lưu trữ phân tán dữ liệu. Dữ liệu được chia thành các phần nhỏ và lưu trữ trên nhiều máy tính khác nhau.
 3. Trên mỗi máy tính, trong mỗi vòng lặp, thực hiện đọc vào từng dòng, gửi lại kết quả cho reducer để tính độ lợi thông tin của từng thuộc tính trong từng phần dữ liệu.
 4. Thực hiện gọi đệ quy xác định nút gốc và nút lá tương ứng. Cập nhật lại nút, cho đến khi đạt hội tụ sau mỗi vòng lặp.

⇒ *Dữ liệu cần phân lớp:* Là danh sách các hàng lưu trên file .txt. được chuyển sang kiểu key/value làm đầu ra cho thuật toán.

- **Mô hình cơ bản của MapReduce:**
 - Map (KeyIn, ValIn) \rightarrow List(KeyInt, ValInt)
 - Reduce (KeyInt, List(ValInt)) \rightarrow List(KeyOut, ValOut)
- **Áp dụng cho thuật toán Item-based Collaborative Filtering:**
 - Xây dựng lớp DataDividedByUser
 - Xây dựng lớp CoOccurrenceMatrixGenerator
 - Xây dựng lớp Normalize
 - Xây dựng lớp Multiplication
 - Xây dựng lớp Sum
 - Xây dựng lớp RecommendationListGenerator
 - Xây dựng lớp RecommendationListName
 - Xây dựng lớp RecommendationName

Bước 1: Xây dựng lớp DataDividerByUser

- **Xử lý:** Nhóm movie và rating tương ứng theo user
- **Mapper**
 - **Input:** user, movie, rating
 - **Output:** key là user, value là movie:rating
- **Reducer**
 - **Input:** key là user, value là movie:rating
 - **Output:** user movie1:rating1, movie2:rating2, ...

Bước 2: Xây dựng lớp CoOccurrenceMatrixGenerator

- **Xử lý:** Đếm số lần mỗi cặp movie được đánh giá bởi cùng 1 người
- **Mapper**
 - **Input:** user movie1:rating1, movie2:rating2, ...
 - **Output:** key là movie1:movie2, value là 1
- **Reducer**
 - **Input:** key là movie1:movie2, value là 1
 - **Output:** movie1:movie2 count (trong đó count là số lần cặp movie đó được đánh giá bởi cùng 1 người)

Bước 3: Xây dựng lớp Normalize

- **Xử lý:** Chuẩn hóa từng đơn vị của ma trận đồng xuất hiện bằng cách chia giá trị đó cho tổng giá trị của cột tương ứng trong ma trận
- **Mapper**
 - **Input:** movie1:movie2 count
 - **Output:** key là movie1, value là movie2:count
- **Reducer**
 - **Input:** key là movie1, value là movie2:count
 - **Output:** movie1 movie2=relation (với relation là giá trị sau khi xử lý)

Bước 4: Xây dựng lớp Multiplication

- **Xử lý:** Nhân relation với rating tương ứng với mỗi key là movie
- **CooccurrenceMapper**
 - **Input:** movie1 movie2=relation
 - **Output:** key là movie1, value là movie2=relation
- **RatingMapper**
 - **Input:** user, movie, rating
 - **Output:** key là movie, value là user:rating
- **Reducer**
 - **Input:** key là movie, value là movie1=relation, movie2=relation, ..., userA:rating, userB:rating, ...
 - **Output:** user:movie result (với $result = relation \times rating$)

Bước 5: Xây dựng lớp Sum

- **Xử lý:** Tính tổng các result theo key là user:movie
- **Mapper**
 - **Input:** user:movie result
 - **Output:** key là user:movie, value là result
- **Reducer**
 - **Input:** key là user:movie, value là result
 - **Output:** user:movie sum (với sum là tổng các result)

Bước 6: Xây dựng lớp RecommendationListGenerator

- **Xử lý:** Lấy ra k movie có giá trị sum lớn nhất với mỗi user được sắp xếp giảm dần (ở đây chúng em lấy k=5)
- **Mapper**
 - **Input:** user:movie sum
 - **Output:** key là user, value là movie:sum
- **Reducer**
 - **Input:** key là user, value là movie:sum
 - **Output:** user movie:sum

Bước 7: Xây dựng lớp RecommendationListName

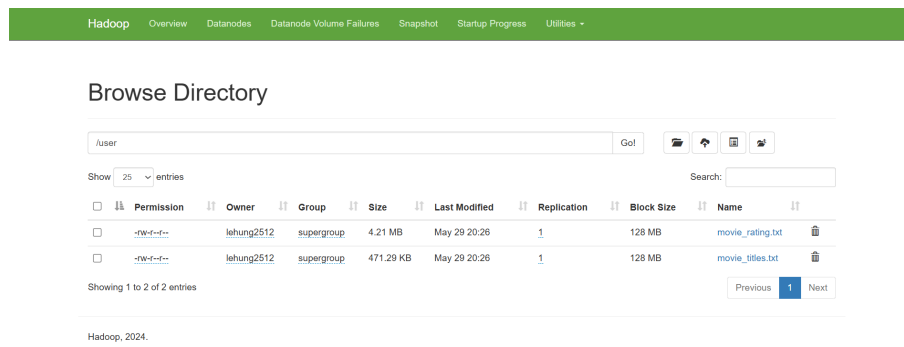
- **Xử lý:** Thay thế movie trong RecommendationListGenerator bằng tên movie tương ứng trong movie_titles.txt
- **RecommendationListMapper**
 - **Input:** user movie:sum
 - **Output:** key là movie, value là user=movie=sum
- **TitlesMapper**
 - **Input:** movie,name
 - **Output:** key là movie, value là movie:name
- **Reducer**
 - **Input:** key là movie, value là userA=movie1=sum, userB=movie2=sum, ..., movie1:name1, movie2:name2, ...
 - **Output:** user name:sum

Bước 8: Xây dựng lớp RecommendationName

- **Xử lý:** Sắp xếp lại RecommendationListName theo userID tăng dần và sum giảm dần
- **Mapper**
 - **Input:** user name:sum
 - **Output:** key là user, value là name:sum
- **Reducer**
 - **Input:** key là user, value là name:sum
 - **Output:** user name

3.4 Demo thuật toán MapReduce Item-based Collaborative Filtering

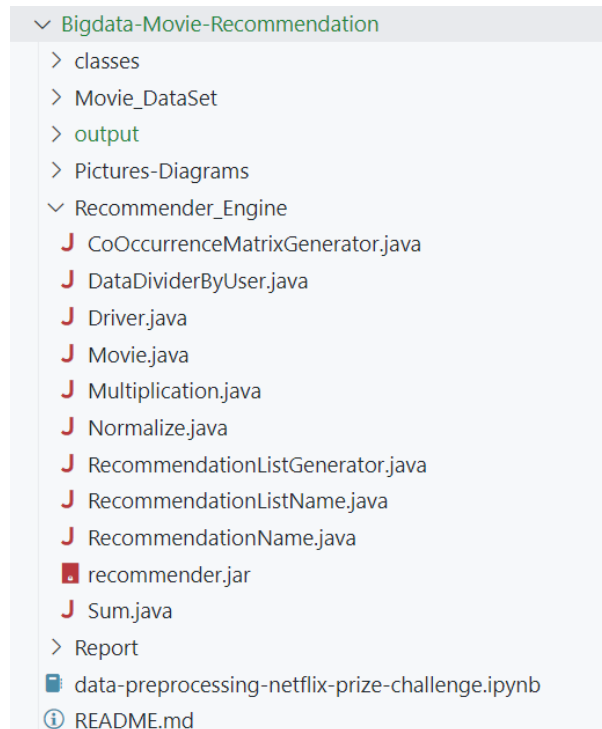
Demo cài đặt hadoop thành công



Hình 3.2: Demo cài đặt Hadoop thành công

Demo Chương trình

1. Cấu trúc thư mục của Project



Hình 3.3: Cấu trúc thư mục của Project

2. Kết quả chương trình

```
2024-05-20 22:08:12,062 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_175078811395_0021
2024-05-20 22:08:12,063 INFO mapreduce.JobSubmitter: Executing ucdm tasks: []
2024-05-20 22:08:12,064 INFO impl.VersionInfoImpl: Submitted application application_175078811395_0021
2024-05-20 22:08:12,457 INFO mapreduce.Job: The url to track the job: http://web-cs-110880.prwv/application_175078811395_0021/
2024-05-20 22:08:12,657 INFO mapreduce.Job: Running job: job_175078811395_0021
2024-05-20 22:08:12,112 INFO mapreduce.Job: Job job_175078811395_0021 running in uber mode : false
2024-05-20 22:08:12,112 INFO mapreduce.Job: map 0% reduce 0%
2024-05-20 22:08:17,106 INFO mapreduce.Job: map 100% reduce 0%
2024-05-20 22:08:17,106 INFO mapreduce.Job: map 100% reduce 100%
2024-05-20 22:08:17,106 INFO mapreduce.Job: Job job_175078811395_0021 completed successfully
2024-05-20 22:08:19,223 INFO mapreduce.Job: Counters: 54
File System Counters
  FILE: Number of bytes read=34128408
  FILE: Number of bytes written=60202059
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  HDFS: Number of bytes read=5523208
  HDFS: Number of bytes written=66103004
  HDFS: Number of read operations=0
  HDFS: Number of large read operations=0
  HDFS: Number of write operations=1
  HDFS: Number of bytes read ensure-codes=0
Job Counters
  Launched map tasks=1
  Launched reduce tasks=1
Data-local map tasks=1
Total time spent by all maps in occupied slots (ms)=3051
Total time spent by all reducers in occupied slots (ms)=3732
Total time spent by all map tasks (ms)=3051
Total time spent by all reduce tasks (ms)=3732
Total vcore-milliseconds taken by all map tasks=3051
Total vcore-milliseconds taken by all reduce tasks=3732
Total megabyte-milliseconds taken by all map tasks=3738204
Total megabyte-milliseconds taken by all reduce tasks=3821168
Map-Reduce Framework
  Map input records=1212318
  Map output records=1211318
  Map output bytes=10103778
  Map output materialized bytes=34338608
  Input split bytes=101
  Combine input records=0
  Combine output records=0
  Reduce input groups=4015
  Reduce shuffle bytes=34338608
  Reduce input records=1212318
  Reduce output records=126575
  Spilled Records=242406
```

Hình 3.4: Kết quả chương trình

3. File kết quả đầu ra

```
1 7 What the #*$! Do We Know!?
2 7 Sick
3 7 Full Frame
4 7 Immortal Beloved
5 7 Character
6 307 What the #*$! Do We Know!?
7 307 Sick
8 307 Full Frame
9 307 Immortal Beloved
10 307 Character
11 424 Immortal Beloved
12 424 My Favorite Brunette
13 424 Character
14 424 Screammers
15 424 Inspector Morse 31
16 462 7 Seconds
17 462 Never Die Alone
18 462 Chump Change
19 462 Strange Relations
20 462 Screammers
```

Hình 3.5: File kết quả đầu ra

4. Jobtracker (Trong phiên bản mới của Hadoop, Jobtracker đã chuyển thành ResourceManager và ApplicationMaster)

17:49:29/05/2024 All Applications Logged in as: mcdo

hadoop

Cluster

Cluster Metrics

Cluster Nodes

Cluster Nodes Metrics

Scheduler Metrics

Scheduler Type

Capacity Scheduler

Scheduler

Tools

Cluster Metrics

Apps Submitted: 11, Apps Pending: 0, Apps Running: 11, Apps Completed: 0, Containers Running: 1, Used Resources: 1 memory 0 B, VCore 0, Total Resources: 1 memory 0 B, VCore 0, Reserverd Resources: 1 memory 0 B, VCore 0, Physical Mem Used %: 74, Physical VCore Used %: 0

Cluster Nodes

Active Nodes: 0, Decommissioning Nodes: 0, Decommissioned Nodes: 0, Lost Nodes: 0, Unhealthy Nodes: 0, Relocated Nodes: 0, Shutdown Nodes: 0

Scheduler Metrics

Scheduler Type: Capacity Scheduler, Scheduling Resource Type: memory 0 B, VCore 0, Minimum Allocation: 1 memory 0 B, VCore 0, Maximum Allocation: 0, Maximum Cluster Application Priority: 0, Scheduler Busy %: 0, RM Dispatcher EventQueue Size: 0, Scheduler Dispatcher EventQueue Size: 0

Search

ID	User	Name	Application Type	Application Tags	Queue	Application Priority	Start Time	Launch Time	Finish Time	State	Final Status	Running Containers	Allocated CPU V-Cores	Allocated Memory MB	Allocated GPUs	Reserved CPU V-Cores	Reserved Memory MB	Reserved GPUs	% of Queue	% of Cluster	Progress	Tracking ID	Blacklisted Nodes
application_171697927316_0011	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:42:17 +0700 2024	Wed May 29 17:42:25 +0700 2024	Wed May 29 17:42:33 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0
application_171697927316_0010	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:41:50 +0700 2024	Wed May 29 17:42:08 +0700 2024	Wed May 29 17:42:12 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0
application_171697927316_0009	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:41:29 +0700 2024	Wed May 29 17:41:58 +0700 2024	Wed May 29 17:41:52 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0
application_171697927316_0008	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:41:09 +0700 2024	Wed May 29 17:41:13 +0700 2024	Wed May 29 17:41:22 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0
application_171697927316_0007	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:40:59 +0700 2024	Wed May 29 17:41:08 +0700 2024	Wed May 29 17:41:03 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0
application_171697927316_0006	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:40:59 +0700 2024	Wed May 29 17:41:08 +0700 2024	Wed May 29 17:41:03 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0
application_171697927316_0005	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:40:59 +0700 2024	Wed May 29 17:41:08 +0700 2024	Wed May 29 17:41:03 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0
application_171697927316_0004	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:40:59 +0700 2024	Wed May 29 17:41:08 +0700 2024	Wed May 29 17:41:03 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0
application_171697927316_0003	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:39:59 +0700 2024	Wed May 29 17:40:08 +0700 2024	Wed May 29 17:40:03 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0
application_171697927316_0002	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:39:59 +0700 2024	Wed May 29 17:40:08 +0700 2024	Wed May 29 17:40:03 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0
application_171697927316_0001	mcdo	recommender.jar	MAPREDUCE		root default	0	Wed May 29 17:38:59 +0700 2024	Wed May 29 17:39:08 +0700 2024	Wed May 29 17:39:03 +0700 2024	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.0	0.0	<div></div>	History	0

Showing 1 to 11 of 11 entries

First Previous 1 Next Last

Hình 3.6: Jobtracker

5. Đánh giá

- Chương trình đã chạy thành công thuật toán MapReduce Item-based Collaborative Filtering.
- Kết quả chạy khớp với kết quả đã thực hiện thủ công trước đó
→ có triển vọng mở rộng với tập dữ liệu lớn hơn

Chương 4

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

4.1 Kết luận

Big data mang đến cho các tổ chức và doanh nghiệp nhiều cơ hội, thách thức và tài nguyên giá trị. Mô hình MapReduce giúp giải quyết vấn đề này bằng cách chia công việc xử lý thành nhiều khối nhỏ, phân tán chúng qua các nút tính toán và sau đó thu thập lại kết quả. Trong đề tài này, chúng em đã áp dụng mô hình MapReduce trên Hadoop, một framework nguồn mở, để xây dựng mô hình dự đoán phim dựa trên thuật toán Item-based Collaborative Filtering.

Hoàn thành đề tài "Hệ thống đề xuất phim dựa trên Item Collaborative Filtering và Hadoop MapReduce", nhóm em đã đạt được những kết quả sau:

- Hiểu tổng quan về Big Data, Hadoop và MapReduce
- Hiểu về thuật toán Item-based Collaborative Filtering
- Triển khai ý tưởng và giải pháp cho việc sử dụng Hadoop MapReduce trong việc triển khai thuật toán Item-based Collaborative Filtering
- Xây dựng lưu đồ thuật toán và triển khai thành công chương trình demo
- Đánh giá chương trình.
- Do hạn chế về công nghệ nên việc phân tích dữ liệu vẫn còn ở mức nhỏ.
- Chương trình demo mới chỉ áp dụng đúng cho dữ liệu đó, dữ liệu khác chưa hỗ trợ được.

4.2 Hướng phát triển

- Áp dụng kiến thức về Big data, apache hadoop, cải tiến và xây dựng ứng dụng phân tích dữ liệu lớn hơn và vào nhiều lĩnh vực khác.
- Trong quá trình hoàn thành bài tập lớn, nhóm em đã cố gắng tìm hiểu và tham khảo các tài liệu liên quan. Tuy nhiên, thời gian có hạn nên chúng em sẽ không tránh khỏi những thiếu sót, rất mong nhận được sự đóng góp ý kiến của thầy và các bạn để báo cáo và kỹ năng của chúng em ngày được hoàn thiện hơn.

SẢN PHẨM CỦA NHÓM

Link GitHub báo cáo của nhóm

Link Slides thuyết trình của nhóm

Link Video demo hệ thống

TÀI LIỆU THAM KHẢO

https://github.com/thviet79/Bigdata_Project_Recommender_System

https://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html

<https://www.ra.ethz.ch/cdstore/www10/papers/pdf/p519.pdf>

<https://github.com/coffee183/Movie-Recommendation-System/tree/master/input>

NHIỆM VỤ CỦA CÁC THÀNH VIÊN

Sinh viên	Nhiệm vụ	Đóng góp
Nguyễn Mạnh Cường	<ul style="list-style-type: none">• Tìm hiểu về Hadoop MapReduce• Tìm hiểu về thuật toán Item-based Collaborative Filtering• Điều hành hoạt động của nhóm• Làm slide và tham gia viết báo cáo	33.3%
Bùi Đức Mạnh	<ul style="list-style-type: none">• Tìm hiểu về Hadoop MapReduce• Tìm hiểu về thuật toán Item-based Collaborative Filtering• Chỉnh sửa code và chạy chương trình• Làm slide và tham gia viết báo cáo• Thuyết trình	33.3%
Lê Việt Hùng	<ul style="list-style-type: none">• Tìm hiểu về Hadoop MapReduce• Tìm hiểu về thuật toán Item-based Collaborative Filtering• Viết, chỉnh sửa và test code• Viết báo cáo và tham gia làm slide	33.3%

Bảng 4.1: Bảng công việc