HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

TRẦN ANH VIỆT

NGHIÊN CỨU MỘT SỐ PHƯƠNG PHÁP PHÂN TÍCH DỮ LIỆU TRÊN BẢNG QUYẾT ĐỊNH TRONG HỆ THỐNG DỮ LIỆU LỚN

LUẬN VĂN THẠC SĨ KỸ THUẬT KỸ THUẬT

(*Theo định hướng ứng dụng*)

HÀ NỘI - 2019

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

TRẦN ANH VIỆT

NGHIÊN CỨU MỘT SỐ PHƯƠNG PHÁP PHÂN TÍCH DỮ LIỆU TRÊN BẢNG QUYẾT ĐỊNH TRONG HỆ THỐNG DỮ LIỆU LỚN

Chuyên ngành: Hệ thống Thông tin

Mã số: 8.48.01.04

LUẬN VĂN THẠC SĨ KỸ THUẬT

NGƯỜI HƯỚNG DẪN KHOA HỌC : GS.TS VŨ ĐỨC THI

HÀ NỘI - 2019

LỜI CAM ĐOAN

1) Tôi xin cam đoan luận văn này là sản phẩm nghiên cứu của tôi.

2) Một số định lý, định nghĩa và hệ quả, thuật toán tôi lấy từ nguồn tài liệu chính xác có trích dẫn tên tài liệu và tên tác giả rõ ràng.

3) Chương trình thử nghiệm là của tôi viết và cài đặt.

4) Tôi xin chịu trách nhiệm hoàn toàn về sản phẩm nghiên cứu của mình.

Tác giả

Trần Anh Việt

LỜI CẢM ƠN

Để có thể hoàn thành đề tài luận văn thạc sĩ một cách hoàn chỉnh, bên cạnh sự nỗ lực cố gắng của bản thân còn có sự hướng dẫn nhiệt tình của quý thầy cô, cũng như sự động viên ủng hộ của gia đình và bạn bè trong suốt thời gian học tập nghiên cứu và thực hiện luận văn thạc sĩ.

Tôi xin chân thành bày tỏ lòng biết ơn đến GS.TS **Vũ Đức Thi**, người đã hết lòng giúp đỡ và tạo mọi điều kiện tốt nhất cho tôi hoàn thành luận văn này. Xin gửi lời cảm ơn chân thành nhất của tôi đối với những điều mà Thầy đã dành cho tôi.

Tôi xin chân thành bày tỏ lòng biết ơn của tôi đến toàn thể quý thầy cô đã giảng dạy và truyền đạt kiến thức cho tôi để tôi có thể hoàn thành các môn học trong suốt thời gian học tại Học viện Công nghệ Bưu chính Viễn thông niên khóa 2018-2020 .

Xin chân thành bày tỏ lòng biết ơn đến gia đình, những người đã không ngừng động viên, hỗ trợ và tạo mọi điều kiện tốt nhất cho tôi trong suốt thời gian học tập và thực hiện luận văn.

Cuối cùng, tôi xin chân thành bày tỏ lòng cảm ơn đến các anh chị, các đồng nghiệp đã hỗ trợ cho tôi rất nhiều trong suốt quá trình học tập, nghiên cứu và thực hiện đề tài luận văn thạc sĩ một cách hoàn chỉnh.

Hà nội, tháng 11 năm 2019.

Học viên

Trần Anh Việt

MỤC LỤC

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT

Ký hiệu, từ viết tắt	Diễn giải
IS = (U,A,V,f)	**Hệ thông tin, hệ thông tin đầy đủ**
IIS = (U,A,V,f)	**Hệ thông tin không đầy đủ**
DS =(U,C ∪ D,V,f)	**Bảng quyết định, bảng quyết định đầy đủ**
IDS =(U,C ∪ D,V,f)	**Bảng quyết định không đầy đủ**
$\|U\|$	**Số đối tượng**
$\|C\|$	**Số thuộc tính điều kiện trên bảng quyết định**
$\|A\|$	**Số thuộc tính trong hệ thông tin**
$\underline{B}$ **X**	**B- xấp xỉ dưới của X**
$\overline{B}$ **X**	**Xấp xỉ trên của X**
$BN_B(D)$	**B – Miền biên của D**
$POS_B(D)$	**B- Miền dương của D**
HRED(C)	**Họ tất cả các tập rút gọn Entropy Shannon**
U/B	**Phân hoạch của U sinh bởi tập thuộc tính B**
SĐQH	**Sơ đồ quan hệ**
H(Q/P)	**Entropy Shannon có điều kiện của Q khi đã biết P**
IE(P)	**Entropy liang mở rộng của tập thuộc tính P trong hệ thông tin đầy đủ**
IND(B)	**Quan hệ B không phân biệt**
TB	**Terabyte**
PB	**Petabyte**

BẢNG CÁC THUẬT NGỮ VIẾT TẮT

Thuật ngữ	Tiếng Anh	Tiếng Việt
CNTT	**Information Technology**	**Công nghệ thông tin**
RBDMS	**Relational Database Management System**	**Hệ quản trị cơ sở dữ liệu quan hệ**
GFS	**Google File System**	**Hệ thống tệp tin được phân phối độc quyền của Google**
IoT	**Internet of Thing**	**Internet kết nối vạn vật**
AI	**Artificial Intelligence**	**Trí tuệ nhân tạo**
IDC	**International Data Corporation**	**Tập đoàn dữ liệu quốc tế**
IBM	**International Business Machines**	**Tập đoàn công nghệ máy tính đa quốc gia**
HDFS	**Hadoop Distributed File System**	**Hệ thống file phân tán**

MỞ ĐẦU

1. Lý do chọn đề tài

Các hệ thống dữ liệu lớn cũng như các phương pháp phân tích dữ liệu lớn đã được nhiều nhà khoa học quan tâm nghiên cứu. Hướng phân tích dữ liệu trên các bảng quyết định mà cụ thể là nghiên cứu các bài toán liên quan đến tập rút gọn trên bảng quyết định phát triển rất sôi động có nhiều ứng dụng trong thực tiễn.

Trong những năm gần đây, sự phát triển mạnh mẽ của công nghệ thông tin đã làm cho khả năng thu thập và lưu trữ thông tin của hệ thống thông tin tăng nhanh một cách nhanh chóng. Sự bùng nổ này đã dẫn tới một yêu cầu cấp thiết là cần có những kỹ thuật và công cụ mới để tự động chuyển đổi lượng dữ liệu khổng lồ kia thành các tri thức có ích. Từ đó, các kỹ thuật khai phá dữ liệu đã trở thành một lĩnh vực thời sự của nền công nghệ thông tin thế giới hiện nay nói chung và Việt Nam nói riêng.

Khai phá dữ liệu đang được áp dụng một cách rộng rãi trong nhiều lĩnh vực kinh doanh và đời sống khác nhau: Market tinh, tài chính ngân hàng và bảo hiểm, khoa học kinh tế…Rất nhiều tổ chức và công ty lớn trên thế giới đã áp dụng kỹ thuật khai phá dữ liệu vào các hoạt động sản xuất kinh doanh của mình và thu được nhiều lợi ích to lớn.

Trong lý thuyết tập thô, dữ liệu được biểu diễn thông qua một hệ thông tin $IS=(U,A)$ với U là tập các đối tượng và A là tập thuộc tính. Phương pháp tiếp cận chính của lý thuyết tập thô là dựa trên quan hệ không phân biệt được để đưa ra các tập xấp xỉ dưới và xấp xỉ trên của nó. Xấp xỉ dưới bao gồm các đối tượng chắc chắn thuộc tập đó, còn xấp xỉ trên chứa tất cả các đối tượng có khả năng thuộc về tập đó. Nếu tập xấp xỉ dưới bằng tập xấp xỉ trên thì tập đối tượng cần quan sát là tập rõ. Ngược lại là tập thô. Các tập xấp xỉ là cơ sở để đưa ra các kết luận từ tập dữ liệu. Bảng quyết định là hệ thông tin IS với tập thuộc tính A được chia thành hai tập con khác rỗng rời nhau C và D, lần lượt được gọi là tập thuộc tính điều kiện và tập thuộc tính quyết định. Nói cách khác, $DS=(U,C\cup D)$ với $C\cap D\not=\varnothing$. Bảng

quyết định là mô hình thường gặp trong thực tế, Khi mà giá trị dữ liệu tại các thuộc tính điều kiện có thể cung cấp cho ta thông tin về giá trị của thuộc tính quyết định. Bảng quyết định là nhất quán khi phụ thuộc hàm $C \rightarrow D$ là đúng, trái lại là không nhất quán.

Rút gọn thuộc tính là ứng dụng quan trọng nhất trong lý thuyết tập thô. Mục tiêu của rút gọn thuộc tính là loại bỏ các thuộc tính dư thừa để tìm ra các thuộc tính cốt yếu và cần thiết trong cơ sở dữ liệu. Với bảng quyết định, rút gọn thuộc tính là tập con nhỏ nhất của tập thuộc tính điều kiện bảo toàn thông tin phân lớp của bảng quyết định. Đối với một bảng quyết định có nhiều tập rút gọn khác nhau tuy nhiên trong thực hành thường không đòi hỏi tìm tất cả các tập rút gọn mà chỉ cần tìm được một tập rút gọn tốt nhất theo một tiêu chuẩn đánh giá nào đó là đủ. Vì vậy, mỗi phương pháp rút gọn thuộc tính đều trình bày một thuật toán Heuristic tìm tập rút gọn. Các thuộc tính này giảm thiểu đáng kể khối lượng tính toán, nhờ đó có thể áp dụng đối với các bài toán có khối lượng dữ liệu lớn.

Cho bảng quyết định nhất quán $DS=(U,C \cup \{d\})$, tập thuộc tính $R \subseteq C$ được gọi là tập rút gọn của thuộc tính điều kiện C nếu R là tập tối thiểu thỏa mãn phụ thuộc hàm $R \rightarrow \{d\}$. Xét quan hệ r trên tập thuộc tính $R \subseteq C\{d\}$ được gọi là một tập tối thiểu của thuộc tính $\{d\}$ nếu R là tập thuộc tính tối thiểu thỏa mãn phụ thuộc hàm $R \rightarrow \{d\}$. Do đó, khái niệm tập rút gọn của bảng quyết định tương đương với tập tối thiểu của thuộc tính $\{d\}$ trên quan hệ, và một vài bài toán trên bảng quyết định liên quan đến tập rút gọn có thể được giải quyết bằng một số kết quả liên quan đến tập tối thiểu của một thuộc tính trong cơ sở dữ liệu quan hệ; bao gồm bài toán tìm tập tất cả các thuộc tính rút gọn, bài toán tìm họ tất cả các tập rút gọn, bài toán trích lọc tri thức dưới dạng các phụ thuộc hàm từ bảng quyết định, bài toán xây dựng bảng quyết định từ tập phụ thuộc hàm cho trước. Cho đến nay, hướng tiếp cận này chưa được nhiều tác giả quan tâm nghiên cứu.

Trên bảng quyết định nhất quán, vấn đề nhiên cứu đặt ra là xây dựng các thuật toán có ý nghĩa liên quan đến tập rút gọn sử dụng một số kết quả liên quan đến tập tối thiểu của một thuộc tính trong một cơ sở dữ liệu quan hệ.

2. Tổng quan về vấn đề nghiên cứu

Nhiều chính phủ quốc gia như Hoa Kỳ cũng đã rất quan tâm đến dữ liệu lớn. Trong tháng 3 năm 2012, chính quyền Obama đã công bố một khoản đầu tư 200 triệu USD để khởi động "Kế hoạch Nghiên cứu và Phát triển Big Data", mà đã là một sáng kiến phát triển khoa học và công nghệ chủ yếu thứ hai sau khi "xa lộ thông tin" bắt đầu vào năm 1993. Trong tháng 7 năm 2012 , dự án "Đẩy mạnh công nghệ thông tin Nhật Bản" được ban hành bởi Bộ Nội vụ và Truyền thông Nhật Bản chỉ ra rằng sự phát triển Big Data, nên có một chiến lược quốc gia và các công nghệ ứng dụng nên là trọng tâm. Trong tháng 7 năm 2012, Liên Hiệp Quốc đã đưa ra báo cáo *Big Data cho phát triển*, trong đó tóm tắt cách các chính phủ sử dụng Big Data để phục vụ tốt hơn và bảo vệ người dân của họ như thế nào.

Hiện nay, mặc dù tầm quan trọng của Big Data đã được thừa nhận rộng rãi. Xong vấn đề then chốt trong việc xử lí các hệ thống Big Data là nghiên cứu phát triển các phương pháp phân tích dữ liệu mà thực chất là khai phá các hệ thống dữ liệu lớn để phát hiện tri thức. Luận văn này nghiên cứu tìm hiểu một số phương pháp phân tích dữ liệu liên quan đến các tập rút gọn trên cấu trúc bảng quyết định sử dụng lí thuyết tập thô**.**

3. Mục đích nghiên cứu

Nghiên cứu và tìm hiểu một số nền tảng của hệ thống dữ liệu lớn. Tìm hiểu một số lĩnh vực phân tích tìm các giá trị của hệ thống dữ liệu lớn *(thực chất là khai phá dữ liệu tìm các tri thức)*.

Nghiên cứu và tìm hiểu một số thuật toán liên quan đến tập rút *gọn (tập thuộc tính rút gọn bảo toàn thông tin phân lớp của bảng quyết định)*. Trên cơ sở này tiến hành xây dựng phần mềm thử nghiệm.

4. Đối tượng và phạm vi nghiên cứu

Nghiên cứu và tìm hiểu các tài liệu liên quan đến hệ thống dữ liệu lớn. Phạm vi nghiên cứu tập trung vào các nền tảng của hệ thống dữ liệu lớn bao gồm những định nghĩa, các đặc trưng, sự phát triển của Big Data và những thách thức mà Big Data mang lại. Các phương pháp phân tích dữ liệu nói chung và phân tích dữ liệu

trên các bảng quyết định liên quan đến các tập rút gọn dùng để phân lớp dữ liệu. Các thuật toán cơ bản nhất liên quan đến tập rút gọn trên bảng quyết định nhất quán.

5. Phương pháp nghiên cứu

Ban đầu thu thập tài liệu Thu thập, tổng hợp các tư liệu, bài báo khoa học đã công bố, tham khảo, so sánh và phân tích để tìm ra vấn đề phù hợp phục vụ cho đề tài nghiên cứu; nghiên cứu tìm hiểu các nền tảng của hệ thống dữ liệu lớn, đặc biệt các phương pháp phân tích dữ liệu trên các bảng quyết định. Cuối cùng xây dựng một phần mềm thực nghiệm.

CHƯƠNG 1: NGHIÊN CỨU CÁC NỀN TẢNG CỦA HỆ THỐNG DỮ LIỆU LỚN

1. Nghiên cứu một số nền tảng của hệ thống dữ liệu lớn (BigData)

1.1 Định nghĩa mô tả và các đặc trưng của Dữ liệu lớn(BigData)

Dữ liệu lớn(Big Data) là một khái niệm trừu tượng, là một thuật ngữ cho việc xử lý một tập hợp dữ liệu rất lớn và phức tạp mà các ứng dụng xử lý dữ liệu truyền thống không xử lý được. Dữ liệu lớn thường bao gồm tập hợp dữ liệu với kích thước vượt xa khả năng của các công cụ phần mềm thông thường để thu thập, hiển thị, quản lý và xử lý dữ liệu trong một thời gian có thể chấp nhận được. Kích thước dữ liệu lớn là một mục tiêu liên tục thay đổi. Ngày nay, đã có rất nhiều định nghĩa về Big Data. Ngay như tên gọi là dữ liệu lớn hay dữ liệu khổng lồ thì nó còn có một số đặc trưng khác trong đó xác định sự khác biệt giữa nó và "dữ liệu lớn" hay "dữ liệu rất lớn".

Hiện nay, mặc dù tầm quan trọng của Big Data đã được thừa nhận rộng rãi, nhưng vẫn có nhiều những ý kiến về định nghĩa của nó. Một cách tổng quát có thể định nghĩa rằng Big Data có nghĩa là các bộ dữ liệu không thể được nhận diện, thu hồi, quản lý và xử lý bằng CNTT truyền thống và các công cụ phần mềm/ phần cứng trong một khoảng thời gian có thể chấp nhận được. Phát sinh từ nhiều sự quan tâm, các doanh nghiệp khoa học và công nghệ, các nhà nghiên cứu, các nhà phân tích dữ liệu và các kỹ thuật viên có những định nghĩa khác nhau về Big Data. Sau đây là một số định nghĩa về Big Data mang tới một sự hiểu biết tốt hơn về những ý nghĩa xã hội, kinh tế và công nghệ rộng lớn của Big Data. Như năm 2012 thì phạm vi một vài tá terabytes tới nhiều petabytes dữ liệu. Dữ liệu lớn yêu cầu một tập các kỹ thuật và công nghệ được tích hợp theo hình thức mới để khai phá từ tập dữ liệu đa dạng, phức tạp, và có quy mô lớn. Trong báo cáo nghiên cứu năm 2001 và những diễn giả liên quan, META Group (bây giờ là Gartner) nhà phân tích Doug Laney định nghĩa những thách thức và cơ hội tăng dữ liệu như là 3 chiều, tăng giá trị dữ liệu, tốc độ vào ra của dữ liệu (velocity), và khổ giới hạn của kiểu dữ liệu (variety). Gartner, và nhiều ngành công nghiệp tiếp tục sử dụng mô hình '3Vs' để mô tả dữ

liệu lớn. Trong năm 2012, Gartner đã cập nhật định nghĩa như sau: "Dữ liệu lớn là khối lượng lớn, tốc độ cao và/hoặc loại hình thông tin rất đa dạng mà yêu cầu phương thức xử lý mới để cho phép tăng cường ra quyết định, khám phá bên trong và xử lý tối ưu". Định nghĩa '3Vs' của Gartner vẫn được sử dụng rộng rãi, và trong phù hợp với định nghĩa đồng thuận.là: "Dữ liệu lớn tiêu biểu cho tập thông tin mà đặc điểm như khối lượng lớn (Volume), tốc độ cao(Velocity) và đa dạng (Variety) để yêu cầu phương thức phân tích và công nghệ riêng biệt để biến nó thành có giá trị". Thêm nữa, vài tổ chức đã thêm vào tính xác thực(Veracity) để mô tả về nó, 3Vs đã được mở rộng để bổ sung đặc tính của dữ liệu lớn:

Volume: Khối lượng - dữ liệu lớn không có mẫu; nó chỉ thực hiện và lần theo những gì diễn ra;

Velocity: Tốc độ - dữ liệu lớn thường được xử lý thời gian thực;

Variety: Đa dạng - dữ liệu lớn có thể thu thập từ văn bản, hình ảnh, âm thanh, video, cộng với nó hoàn thành các phần dữ liệu thiếu thông qua tổng hợp dữ liệu;

Machine Learning: Máy học - dữ liệu lớn thường không hỏi tại sao và đơn giản xác định hình mẫu.

Digital footprint: Dấu chân kỹ thuật số - dữ liệu lớn thường là phụ sinh miễn phí của quá trình tương tác kỹ thuật số.

Hiện nay, hệ thống dữ liệu lớn BigData được nhiều nhà khoa học định nghĩa mô tả dựa trên bốn đặc trưng sau đây:

Dung lượng lớn: Có nghĩa là khối lượng dữ liệu cần xử lý cực kỳ lớn

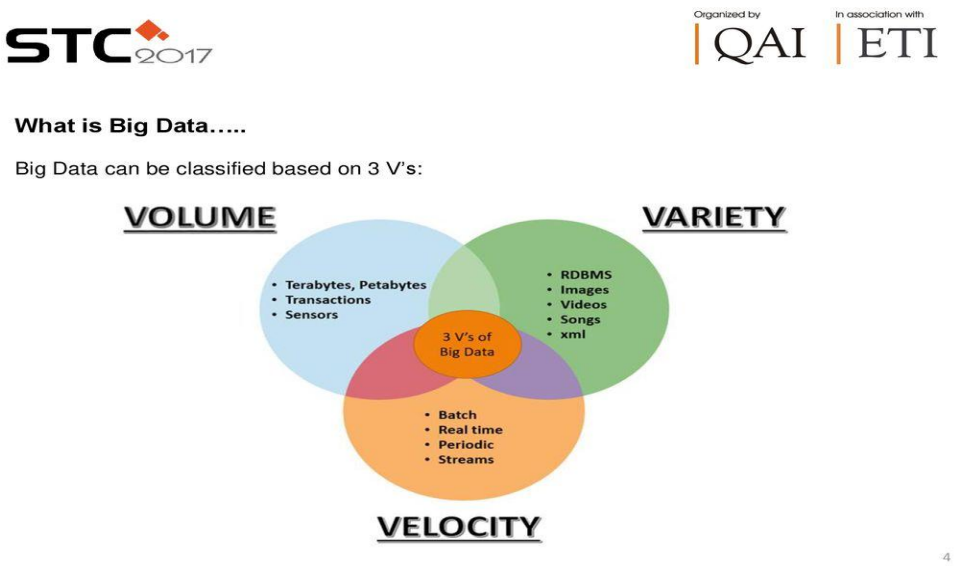
Đa dạng dữ liệu: Phương thức thu thập dữ liệu và các loại dữ liệu rất phong phú bao gồm các dữ liệu có cấu trúc và phi cấu trúc như dữ liệu dạng bảng, đồ thị, loại dữ liệu dạng âm thanh, hình ảnh, video, web, văn bản, dữ liệu di động…;

Tốc độ: Việc thu thập và phân tích dữ liệu phải được tiến hành nhanh chóng và kịp thời(thời gian thực thì càng tốt), để sử dụng một cách tối đa các giá trị của BigData

Tính giá trị: Các phương pháp xử lý của Bigdata phải tìm và phát hiện ra các giá trị, mà thực chất là những tri thức từ các hệ thống dữ liệu lớn này. Đây chính là mục tiêu của các hệ thống dữ liệu lớn.

Năm 2010, Apache Hadoop định nghĩa dữ liệu lớn như "bộ dữ liệu mà không thể thu thập, quản lý và xử lý bởi các máy tính nói chung trong một phạm vi chấp nhận được". Cũng trên cơ sở đó, vào tháng 5 năm 2011, McKinsey & Company, một công ty tư vấn toàn cầu công bố Big Data như một địa hạt mới cho sự đổi mới, cạnh tranh và hiệu suất. Big Data có nghĩa là những bộ dữ liệu mà không có thể được thu lại, lưu trữ và quản lý bởi phần mềm cơ sở dữ liệu cổ điển. Định nghĩa này gồm hai ý nghĩa: Thứ nhất, dung lượng của các tập dữ liệu mà phù hợp với tiêu chuẩn Big Data đang thay đổi và có thể tăng trưởng theo thời gian hoặc với những tiến bộ công nghệ. Thứ hai, dung lượng của các tập dữ liệu mà phù hợp với tiêu chuẩn của Big Data trong các ứng dụng khác nhau trong mỗi ứng dụng. Hiện nay, Big Data thường từ vài TB đến vài PB. Từ định nghĩa của McKinsey & Company, có thể thấy rằng dung lượng của một tập dữ liệu không phải là tiêu chí duy nhất cho Big Data. Quy mô dữ liệu ngày càng phát triển và việc quản lý nó mà không thể xử lý bằng công nghệ cơ sử dữ liệu truyền thống là hai đăng trưng quan trọng tiếp theo.

Dữ liệu lớn đã được định nghĩa từ sớm những năm 2001. Doug Laney, một nhà phân tích của META (nay có tên là công ty nghiên cứu Gartner) định nghĩa những thách thức và cơ hội mang lại của sự tăng trưởng dữ liệu với một mô hình "3Vs", tức là sự gia tăng của dung lượng, tốc độ và tính đa dạng. Mặc dù, mô hình này ban đầu không được sử dụng để xác định Big Data, tuy nhiên Gatrtner cùng nhiều doanh nghiệp khác bao gồm cả IBM và một số cơ sở nghiên cứu của Microsoft vẫn còn sử dụng mô hình "3Vs" để mô tả về dữ liệu lớn trong vòng 10 năm tiếp theo.



Hình 1.1: Mô hình "3Vs" của Big Data

Mô hình "3Vs" được giải thích như sau:

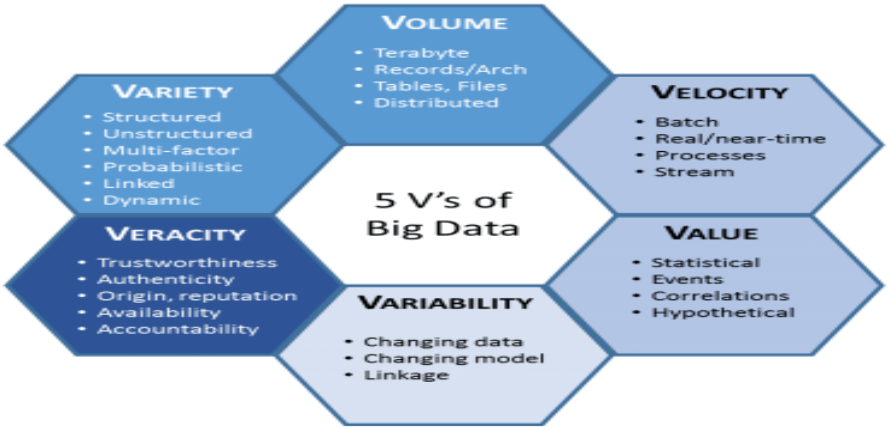
- Dung lượng (Volume): Sự sản sinh và thu thập các dữ liệu lớn, quy mô dữ liệu trở nên ngày càng lớn.

- Tốc độ (Velocity): Tính kịp thời của dữ liệu lớn, cụ thể là việc thu thập và phân tích dữ liệu phải được tiến hành nhanh chóng và kịp thời để sử dụng một cách tối đa các giá trị thương mại của Big Data.

- Tính đa dạng (Variety): Các loại dữ liệu khác nhau bao gồm dữ liệu bán cấu trúc và phi cấu trúc như âm thanh, video, web, văn bản,…cũng như dữ liệu có cấu trúc truyền thống.

Đến năm 2011, định nghĩa về Big Data đã có sự thay đổi khi một báo cáo của IDC đã đưa ra một định nghĩa như sau: "Công nghệ Big Data mô tả một thế hệ mới của những công nghệ và kiến trúc, được thiết kế để lấy ra giá trị kinh tế từ dung lượng rất lớn của một loạt các dữ liệu bằng cách cho phép tốc độ cao trong việc thu thập, khám phá hoặc phân tích". Với định nghĩa này, dữ liệu lớn mang trong mình bốn đặc trưng và được hiểu như một mô hình "4Vs".

Năm 2014, Gartner lại đưa ra một khái niệm mới về Big Data qua mô hình "5Vs" với năm tính chất quan trọng của Big Data.



Hình 1.2: Mô hình 5vs của Big Data

Mô hình "5Vs" được giải thích như sau:

- Khối lượng (Volume): Sự sản sinh và thu thập các dữ liệu lớn, quy mô dữ liệu trở nên ngày càng lớn.

- Tốc độ (Velocity): Tính kịp thời của dữ liệu lớn, cụ thể là việc thu thập và phân tích dữ liệu phải được tiến hành nhanh chóng và kịp thời để sử dụng một cách tối đa các giá trị thương mại của Big Data.

- Tính đa dạng (Variety): Các loại dữ liệu khác nhau bao gồm dữ liệu bán cấu trúc và phi cấu trúc như âm thanh, video, web, văn bản,…cũng như dữ liệu có cấu trúc truyền thống.

- Tính chính xác (Veracity): Tính hỗn độn hoặc tin cậy của dữ liệu. Với rất nhiều dạng thức khác nhau của dữ liệu lớn, chất lượng và tính chính xác của dữ liệu rất khó kiểm soát. Khối lượng dữ liệu lớn sẽ đi kèm với tính xác thực của dữ liệu.

- Giá trị (Value): Đây được coi là đặc điểm quan trọng nhất của dữ liệu lớn. Việc tiếp cận dữ liệu lớn sẽ không có ý nghĩa nếu không được chuyển thành những thứ có giá trị. Giá trị của dữ liệu là đặc điểm quan trọng nhất trong mô hình "5Vs" của Big Data.

Ngoài ra, Viện tiêu chuẩn và kỹ thuật quốc gia của Hoa Kỳ (NIST) định nghĩa "Dữ liệu lớn có nghĩa là các dữ liệu mà dung lượng dữ liệu, tốc độ thu thập hoặc biểu diễn dữ liệu hạn chế khả năng của việc sử dụng các phương pháp quan hệ truyền thống để tiến hành phân tích hiệu quả hoặc các dữ liệu mà có thể được xử lý

một cách hiệu quả với các công nghệ". Định nghĩa này tập trung vào các khía cạnh công nghệ của Big Data. Nó chỉ ra rằng phương pháp hay công nghệ hiệu quả cần phải được phát triển và được sử dụng để phân tích và xử lý dữ liệu lớn.

1.2 Sự phát triển của BigData và các Công nghệ liên quan

Cuối những năm 1970, khái niệm "máy cơ sở dữ liệu" nổi lên, đó là một công nghệ đặc biệt sử dụng cho việc lưu trữ và phân tích dữ liệu. Với sự gia tăng của dung lượng dữ liệu, khả năng lưu trữ và xử lý của một hệ thống máy tính lớn duy nhất trở nên không đủ. Trong những năm 1980, hệ thống "không chia sẻ"- một hệ thống cơ sở dữ liệu song song được đề xuất để đáp ứng nhu cầu của dung lượng dữ liệu ngày càng tăng [14]. Kiến trúc hệ thống không chia sẻ được dựa trên việc sử dụng các cụm và mỗi máy có riêng bộ xử lý, lưu trữ và đĩa cứng. Hệ thống Teradata là hệ thống cơ sở dữ liệu song song thương mại thành công đầu tiên. Ngày 2 tháng 6 năm 1986, một sự kiện bước ngoặt xảy ra khi Teradata giao hệ thống cơ sở dữ liệu song song đầu tiên với dung lượng lưu trữ 1TB cho Kmart để giúp các công ty bán lẻ quy mô lớn tại Bắc Mỹ mở rộng kho dữ liệu [16]. Trong những năm 1990, những ưu điểm của cơ sở dữ liệu song song đã được công nhận rộng rãi trong lĩnh vực cơ sở dữ liệu. Tuy nhiên, Big Data vẫn còn nhiều thách thức phát sinh. Với sự phát triển của dịch vụ Internet, các nội dung chỉ mục và truy vấn đã được phát triển nhanh chóng. Do đó, công cụ tìm kiếm của các công ty đều phải đối mặt với những thách thức của việc xử lý dữ liệu lớn. Google tạo ra mô hình lập trình GFS [16] và MapReduce [17] để đối phó với những thách thức mang lại về việc quản lý và phân tích dữ liệu ở quy mô Internet. Ngoài ra, nội dung được sinh ra bởi người sử dụng, cảm biến và các nguồn dữ liệu phổ biến khác cũng tăng, do đó yêu cầu một sự thay đổi cơ bản về kiến trúc tính toán và cơ chế xử lý dữ liệu quy mô lớn.

Vào tháng 1 năm 2007, Jim Gray là một nhà tiên phong về phần mềm cơ sở dữ liệu đã gọi sự biến đổi là "mô hình thứ tư" [15]. Ông nghĩ rằng cách duy nhất đối phó với mô hình như vậy là phát triển một thế hệ mới các công cụ máy tính để quản lý, trực quan hóa và phân tích dữ liệu khổng lồ. Trong tháng 6 năm 2011, một sự kiện bước ngoặt xảy ra khi EMC/IDC công bố một báo cáo nghiên cứu có tựa đề

Trích xuất giá trị từ sự hỗn độn, đây là lần đầu tiên đưa ra khái niệm và tiềm năng của Big Data. Báo cáo nghiên cứu này gây ra mối quan tâm lớn trong cả công nghiệp và học thuật về Big Data.

Trong vài năm qua, những công ty lớn bao gồm EMC, Oracle, IBM, Microsoft, Google, Amazon, Facebook,… đã bắt đầu các dự án Big Data của họ. Từ năm 2005, IBM đã đầu tư 16 tỷ USD vào 30 sự tiếp nhận liên quan đến dữ liệu lớn. Về học thuật, Big Data cũng chiếm địa vị nổi bật. Trong năm 2008, Nature công bố một vấn đề đặc biệt về Big Data. Năm 2011, Science cũng đưa ra một vấn đề đặc biệt về công nghệ chủ chốt "xử lý dữ liệu" trong Big Data. Năm 2012, Tạp chí Hiệp hội Nghiên cứu châu Âu Tin học và Toán học (ERCIM) đăng một vấn đề đặc biệt về dữ liệu lớn. Vào đầu năm 2012, một báo cáo mang tên *Big Data, Big Impact* trình bày tại diễn đàn Davos ở Thụy Sĩ, đã thông báo rằng Big Data đã trở thành một loại tài sản kinh tế mới, giống như tiền tệ hoặc vàng.

Nhiều chính phủ quốc gia như Mỹ cũng đã rất quan tâm tới dữ liệu lớn. Trong tháng 3 năm 2012, chính quyền Obama đã công bố một khoản đầu tư 200 triệu USD để khởi động "Kế hoạch nghiên cứu và phát triển Big Data". Tháng 7 năm 2012 dự án "Đẩy mạnh công nghệ thông tin Nhật Bản" được ban hành bởi Bộ Nội vụ và Truyền thông Nhật Bản chỉ ra rằng sự phát triển Big Data nên có một chiến lược quốc gia và các công nghệ ứng dụng nên là trọng tâm. Cũng trong thời gian đó, Liên Hiệp Quốc đã đưa ra báo cáo *Big Data cho phát triển*, trong đó tóm tắt cách mà các chính phủ sử dụng Big Data để phục vụ và bảo vệ người dân một cách tốt hơn.

Công ty nghiên cứu thị trường IDC cho thấy doanh thu đến từ thị trường Big Data sẽ tăng lên 16,9 tỷ USD vào năm 2015 và sẽ tiếp tục tăng trưởng kép với tốc độ 27% và đạt đến 32,4 tỷ USD vào năm 2017. Có rất nhiều công nghệ gắn liền với Big Data, phần này sẽ trình bày và giới thiệu một số công nghệ cơ bản liên quan chặt chẽ tới Big Data bao gồm điện toán đám mây(Cloud Computing), Internet Of Things(IoT), trung tâm dữ liệu(Data Centre), Hadoop và Big Data.

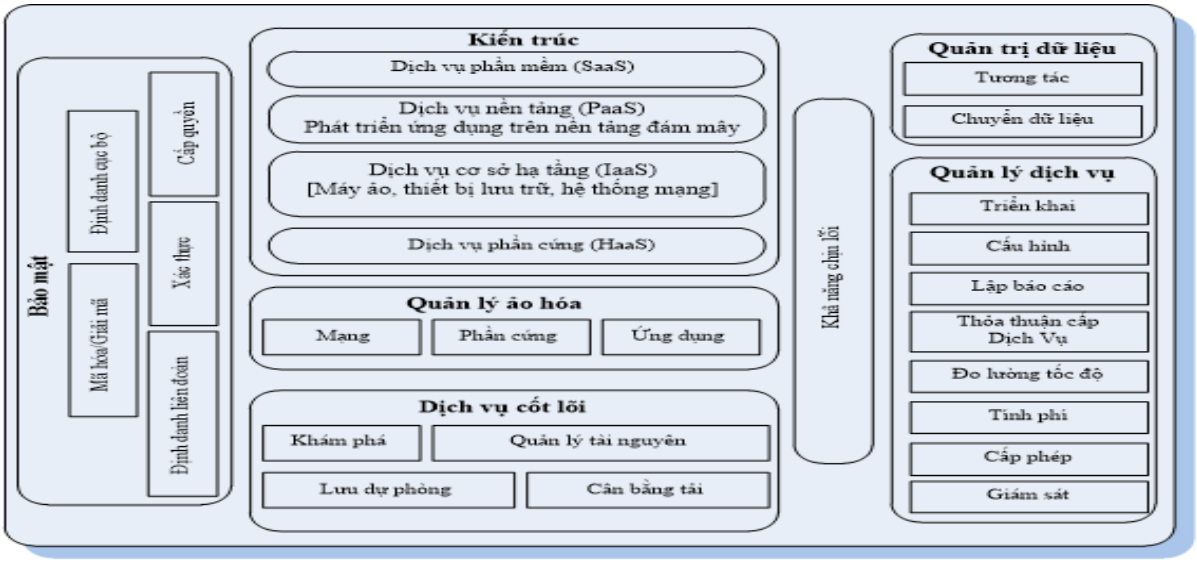
Điện toán đám mây(Cloud computing) và Big Data:

Theo Wikimedia thì điện toán đám mây hay còn gọi là điện toán máy chủ ảo, là mô hình điện toán sử dụng các công nghệ máy tính và phát triển dựa vào mạng Internet. Thuật ngữ "đám mây" ở đây là lối nói ẩn dụ chỉ mạng Internet (dựa vào cách được bố trí của nó trong sơ đồ mạng máy tính) và như một liên tưởng về độ phức tạp của các cơ sở hạ tầng chứa trong nó. Ở mô hình điện toán này, mọi khả năng liên quan đến công nghệ thông tin đều được cung cấp dưới dạng các "dịch vụ", cho phép người sử dụng truy cập các dịch vụ công nghệ từ một nhà cung cấp nào đó "trong đám mây" mà không cần phải có các kiến thức, kinh nghiệm về công nghệ đó, cũng như không cần quan tâm đến các cơ sở hạ tầng phục vụ công nghệ đó. Theo tổ chức IEEE "*Nó là hình mẫu trong đó thông tin được lưu trữ thường trực tại các máy chủ trên Internet và chỉ được được lưu trữ tạm thời ở các máy khách, bao gồm máy tính cá nhân, trung tâm giải trí, máy tính trong doanh nghiệp, các phương tiện máy tính cầm tay,...*". Điện toán đám mây là khái niệm tổng thể bao gồm cả các khái niệm như phần mềm dịch vụ, Web 2.0 và các vấn đề khác xuất hiện gần đây, các xu hướng công nghệ nổi bật, trong đó đề tài chủ yếu của nó là vấn đề dựa vào Internet để đáp ứng những nhu cầu điện toán của người dùng. Ví dụ, dịch vụ Google AppEngine cung cấp những ứng dụng kinh doanh trực tuyến thông thường, có thể truy nhập từ một trình duyệt web, còn các phần mềm và dữ liệu đều được lưu trữ trên các máy chủ.

Ngoài ra, theo IBM thì điện toán đám mây là việc cung cấp tài nguyên máy tính cho người dùng tùy theo mục đích sử dụng thông qua Internet. Nguồn tài nguyên đó có thể là bất cứ thứ gì liên quan đến điện toán và máy tính, ví dụ như phần mềm, phần cứng, hạ tầng mạng cho tới các máy chủ và mạng lưới máy chủ cỡ lớn.

Điện toán đám mây có liên quan chặt chẽ với Big Data. Big Data là đối tượng của hoạt động tính toán chuyên sâu và nhấn mạnh khả năng lưu trữ của mỗi hệ thống đám mây. Mục tiêu chính của hệ thống đám mây là sử dụng tài nguyên tính toán và lưu trữ rất lớn dưới sự quản lý tập trung để cung cấp cho các ứng dụng Big Data khả năng tính toán tốt. Sự phát triển của điện toán đám mây cung cấp các giải pháp cho

việc lưu trữ và xử lý Big Data. Mặt khác, sự xuất hiện của Big Data cũng làm tăng tốc độ phát triển của điện toán đám mây. Các công nghệ lưu trữ phân tán dựa trên điện toán đám mây có thể quản lý Big Data một cách hiểu quả cùng với khả năng tính toán song song của điện toán đám mây có thể nâng cao hiệu quả của việc thu thập và phân tích dữ liệu lớn.



Hình 1.3: Kiến trúc của điện toán đám mây

Hiện nay, có rất nhiều loại dịch vụ điện toán đám mây nhưng nhìn chung đều có những dịch vụ cơ bản sau: Dịch vụ cơ sở hạ tầng (Infrastructure as a Service - IaaS), dịch vụ nền tảng (Platform as a Service - PaaS), dịch vụ phần mềm (Software as a Service - SaaS), dịch vụ phần cứng (Hardware as a Service).

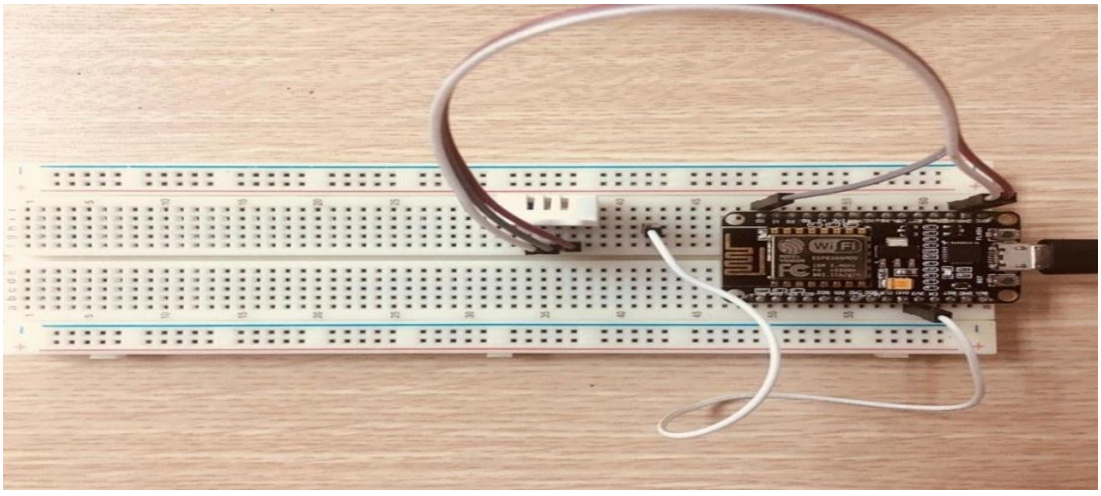
Mặc dù có nhiều công nghệ trùng lặp giữa điện toán đám mây và Big Data, tuy nhiên chúng khác nhau ở hai khía cạnh sau. Đầu tiên, các khái niệm khác nhau ở một mức độ nhất định. Điện toán đám mây biến đổi kiến trúc CNTT trong khi Big Data ảnh hưởng đến các quyết định kinh doanh. Tuy vậy, Big Data cũng phải phụ thuộc vào điện toán đám mây như các cơ sở hạ tầng để hoạt động trơn tru. Thứ hai, Big Data và điện toán đám mây có khách hàng mục tiêu khác nhau. Điện toán đám mây là một công nghệ và sản phẩm nhắm đến Chief Information Officers (CIO) như một giải pháp CNTT tiên tiến. Big Data là một sản phẩm nhắm đến Chief Executive Officers (CEO) người mà chỉ tập trung vào hoạt động kinh doanh. Khi những người

ra quyết định có thể trực tiếp cảm nhận được áp lực cạnh tranh trên thị trường, họ phải đánh bại các đối thủ kinh doanh theo nhiều cách cạnh tranh hơn. Với sự tiến bộ của Big Data và điện toán đám mây hai công nghệ này đã trở thành tất yếu và ngày càng kết hợp chặt chẽ với nhau. Điện toán đám mây với các chức năng tương tự như của máy tính và hệ điều hành, cung cấp tài nguyên cấp hệ thống. Dữ liệu lớn hoạt động trong các cấp độ bên trên được hỗ trợ bởi điện toán đám mây và cung cấp chức năng tương tự như của cơ sở dữ liệu và khả năng xử lý dữ liệu có hiệu quả.

Sự phát triển của Big Data được thúc đẩy bởi sự tăng trưởng nhanh chóng của nhu cầu ứng dụng và điện toán đám mây được phát triển từ công nghệ ảo hóa. Đến một lúc nào đó, các tiến bộ của điện toán đám mây cũng thúc đẩy sự phát triển của Big Data, cả hai sẽ bổ sung cho nhau.

Internet Of Things(IOT) và Big Data:

Mô hình IoT sử dụng một số lượng lớn các bộ cảm biến kết nối mạng được nhúng vào các thiết bị và các máy móc khac nhau trong thế giới thực. Các cảm biến như vậy được triển khai trong các lĩnh vực khác nhau có thể thu thập các loại dữ liệu khác nhau, chẳng hạn như dữ liệu về môi trường, dữ liệu địa lý, dữ liệu thiên văn và dữ liệu logistic. Thiết bị di động, phương tiện vận tải, phương tiện cộng cộng và đồ gia dụng tất cả có thể là những thiết bị thu thập dữ liệu trong IoT.



Hình 1.4: Bộ cảm biến đo độ ẩm và nhiệt độ DHT22 và chip ESP8266MOD

Big Data được tạo ra bởi IoT có các đặc trưng khác so với Big Data nói chung do các loại khác nhau của dữ liệu thu thập được, trong đó các đặc trưng cổ

điển nhất bao gồm sự không đồng nhất, tính đa dạng, tính năng không có cấu trúc, nhiễu và độ dư thừa cao. Mặc dù dữ liệu IoT hiện nay không phải là phần thống trị của Big Data nhưng trong tương lai số lượng cảm biến sẽ đạt một nghìn tỷ (ước tính vào năm 2030 theo dự báo của HP) số lượng cảm biến sẽ đạt một nghìn tỷ và khi đó dữ liệu IoT sẽ là phần quan trọng nhất của dữ liệu lớn. Tập đoàn Intel đã đưa ra một báo cáo trong đó chỉ ra rằng dữ liệu lớn trong IoT có ba tính năng phù hợp với các mô hình dữ liệu lớn: (i) thiết bị đầu cuối phong phú tạo ra khối lượng dữ liệu lớn, (ii) các dữ liệu được tạo ra bởi IoT thường là bán cấu trúc hoạc không có cấu trúc; (iii) dữ liệu của IoT chỉ có ích khi nó được phân tích.

Có một nhu cầu bắt buộc áp dụng Big Data cho các ứng dụng IoT, trong khi sự phát triển của dữ liệu lớn đã sẵn sang hỗ trợ. Việc này đã được công nhận rộng rãi khi hai công nghệ này đều phụ thuộc lẫn nhau và cần được phối hợp để phát triển. Việc triển khai rộng rãi IoT đẩy sự tăng trưởng cao của dữ liệu về cả số lượng và chủng loại từ đó cung cấp cơ hội cho các ứng dụng và phát triển của Big Data. Mặt khác, áp dụng công nghệ dữ liệu lớn vào IoT cũng làm tăng tốc độ tiến bộ nghiên cứu và mô hình kinh doanh của IoT.

Trung tâm dữ liệu(Data centre) và Big data:

Trong mô hình dữ liệu lớn, các trung tâm dữ liệu không chỉ là một nền tảng lưu trữ tập trung dữ liệu, mà còn đảm nhận nhiều trách nhiệm chẳng hạn như thu thập dữ liệu, quản lý dữ liệu, tổ chức dữ liệu và tận dụng các giá trị dữ liệu cùng các chức năng.

Hình 1.5 Hệ thống trung tâm dữ liệu

Các trung tâm dữ liệu chủ yếu tập trung vào dữ liệu. Dữ liệu được tổ chức, quản lý theo mục tiêu và phát triển con đường cốt lõi của trung tâm dữ liệu. Sự xuất hiện của Big Data mang lại những cơ hội phát triển và thách thức lớn cho các trung tâm dữ liệu. Big Data sẽ thúc đẩy sự tăng trưởng bùng nổ của các cơ sở hạ tầng và các phần mềm liên quan của trung tâm dữ liệu. Mạng lưới trung tâm dữ liệu vật lý là nòng cốt hỗ trợ Big Data nhưng hiện nay cơ sở hạ tầng chính mời là điều cần gấp nhất.

Big Data đòi hỏi trung tâm dữ liệu cung cấp nền tảng hỗ trợ mạnh mẽ. Các mô hình Big Data yêu cầu nghiêm ngặt hơn về khả năng lưu trữ và khả năng xử lý, cũng như khả năng truyền tải mạng. Big Data tạo ra cho các trung tâm dữ liệu nhiều chức năng hơn. Trong các mô hình Big Data, trung tâm dữ liệu có trách nhiệm không chỉ tập trung vào các thiết bị phần cứng mà còn tăng cường năng lực mềm như khả năng thu hồi, xử lý, tổ chức, phân tích và ứng dụng của Big Data. Các trung tâm dữ liệu có thể giúp nhân viên kinh doanh phân tích các dữ liệu hiện có, phát hiện ra các vấn đề trong hoạt động kinh doanh và phát triển các giải pháp từ Big Data.

Hadoop và Big data:

Hadoop là một Apache framework mã nguồn mở được viết bằng Java, cho phép xử lý phân tán (distributed processing) các tập dữ liệu lớn trên các cụm máy tính (clusters of computers) thông qua mô hình lập trình đơn giản. Hadoop được

thiết kế để mở rộng quy mô từ một máy chủ đơn giản sang hàng ngàn máy tính khác có tính toán và lưu trữ cục bộ (local computation and storage).

Hadoop được sử dụng rộng rãi trong các ứng dụng Big Data trong công nghiệp, ví dụ như lọc thư rác, tìm kiếm mạng, phân tích luồng clicks hay khuyến cáo xã hội,…Ngoài ra, các nghiên cứu học thuật đáng kể hiện nay dựa trên Hadoop. Tháng 6 năm 2012, Yahoo chạy Hadoop trên 42,000 máy chủ tại bốn trung tâm dữ liệu để hỗ trợ các sản phẩm và dịch vụ của mình. Cũng trong thời gian đó, Facebook thông báo rằng cụm Hadoop của họ có thể xử lý 100PB dữ liệu mà dữ liệu này có thể tăng 0,5 PB mỗi ngày như trong tháng 11 năm 2012. Ngoài ra, nhiều công ty cung cấp Hadoop thương mại bao gồm Cloudera, IBM, MapR,…

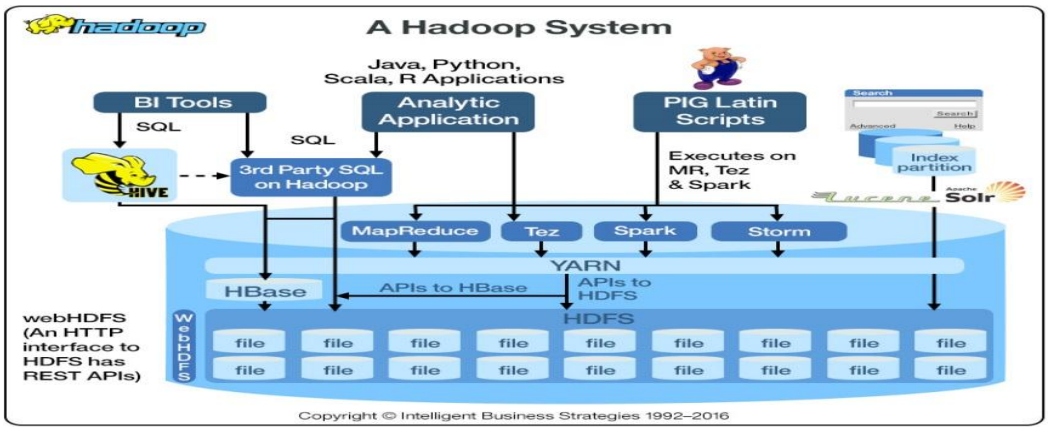
Về kiến trúc, Hadoop gồm 4 module:

+ *Hadoop Common:* Đây là các thư viện và tiện ích cần thiết của Java để các module khác sử dụng. Những thư viện này cung cấp hệ thống file và lớp OS trừu tượng, đồng thời chứa các mã lệnh Java để khởi động Hadoop.

+ *Hadoop YARN*: Đây là framework để quản lý tiến trình và tài nguyên của các cluster.

+ *Hadoop Distributed File System (HDFS):* Đây là hệ thống file phân tán cung cấp truy cập thông lượng cao cho ứng dụng khai thác dữ liệu.

+ *Hadoop MapReduce:* Đây là hệ thống dựa trên YARN dùng để xử lý song song các tập dữ liệu lớn.



Hình 1.6 Kiến trúc hệ thống Hadoop

Trong số các máy móc và hệ thống công nghiệp hiện đại, các cảm biến được triển khai rộng rãi để thu thập thông tin cho việc theo dõi môi trường và dự báo sự cố. Bahga và những cộng sự của mình đã đề xuất một framework cho việc tổ chức dữ liệu và cơ sở hạ tầng điện toán đám mây gọi là CloudView [19]. CloudView sử dụng kiến trúc hỗn hợp, các node địa phương và các cụm dữ điều khiển từ xa dựa trên Hadoop để phân tích dữ liệu máy tính tạo ra. Các node địa phương được sử dụng cho các dự báo thời gian thực các sự cố, các cụm dựa trên Hadoop được dùng để phân tích offline.

1.3 Các thách thức đối với BigData

Với sự gia tăng một cách mạnh mẽ của dữ liệu trong kỷ nguyên Big Data đã mang tới những thách thức rất lớn về việc thu thập, lưu trữ, quản lý và phân tích dữ liệu. Hệ thống quản lý và phân tích dữ liệu truyền thống được dựa trên hệ thống quản lý cơ sở dữ liệu quan hệ (RDBMS). Tuy nhiên, RDBMS chỉ áp dụng cho các dữ liệu có cấu trúc, khác với những dữ liệu bán cấu trúc hoặc không có cấu trúc. Ngoài ra, RDBMS đang ngày càng sử dụng nhiều phần cứng đắt tiền. Các RDBMS truyền thống không thể xử lý dung lượng rất lớn và không đồng nhất của Big Data. Cộng đồng nghiên cứu đã đề xuất một số giải pháp theo các quan điểm khác nhau. Đối với các giải pháp lưu trữ vĩnh viễn và quản lý các tập dữ liệu qui mô lớn không có trật tự, hệ thống tập tin được phân phối và cơ sở dữ liệu NoSQL là những lựa chọn tốt. Những frameworks lập trình như vậy đã đạt được thành công lớn trong các bài toán xử lý cụm, đặc biệt đối với lập thứ hạng trang web (webpage ranking). Nhiều ứng dụng dữ liệu lớn có thể được phát triển dựa trên những công nghệ hoặc nền tảng cách mạng này.

Các thách thức chính mà Big Data mang lại:

- *Biểu diễn dữ liệu:* Nhiều bộ dữ liệu có mức độ không đồng nhất trong kiểu, cấu trúc, ngữ nghĩa, tổ chức, độ chi tiết và khả năng tiếp cận. Biểu diễn dữ liệu nhằm mục đích để làm cho dữ liệu có ý nghĩa hơn cho phân tính máy tính và sự giải thích của người dùng. Tuy nhiên, việc biểu diễn dữ liệu không đúng cách sẽ làm giảm giá trị ban đầu của dữ liệu và thậm chí có thể gây cản trở cho việc phân tích

dữ liệu. Biểu diễn dữ liệu hiệu quả sẽ phản ánh cấu trúc, lớp và kiểu dữ liệu cũng như các công nghệ tích hợp, để cho phép hoạt động hiệu quả trên các tập dữ liệu khác nhau.

- *Giảm sự dư thừa và nén dữ liệu:* Giảm sự dư thừa và nén dữ liệu là cách hiệu quả để giảm chi phí gián tiếp của toàn bộ hệ thống trên tiền đề rằng các giá trị tiềm năng của dữ liệu không bị ảnh hưởng. Ví dụ, hầu hết các dữ liệu được tạo ra bởi các mạng cảm biến là rất cần thiết, trong đó có thể được logic và nén ở các đơn đặt hàng của các cường độ.

- *Quản lý vòng đời của dữ liệu:* Vòng đời của dữ liệu là chuỗi các giai đoạn mà một đơn vị dữ liệu từ thế hệ ban đầu được thu thập, lưu trữ đến khi bị xóa bỏ và kết thúc vòng đời hữu ích của nó. So với tiến bộ của hệ thống lưu trữ tương ứng, cảm biến và máy tính đang tạo ra dữ liệu với quy mô và tốc độ chưa từng có. Điều này đã tạo ra rất nhiều thách thức, một trong số đó là hệ thống lưu trữ hiện đại không thể hỗ trợ dữ liệu lớn như vậy. Vì vậy, một nguyên tắc quan trọng liên quan đến các giá trị phân tích cần được phát triển để quyết định dữ liệu nào sẽ được lưu trữ và dữ liệu nào sẽ được loại bỏ.

- *Cơ chế phân tích:* Hệ thống phân tích Big Data sẽ xử lý khối lượng dữ liệu không đồng nhất trong mọt thời gian giới hạn. Tuy nhiên, RDBMS truyền thống được thiết kế với sự thiếu khả năng thay đổi và khả năng mở rộng, do đó không thể đáp ứng các yêu cầu về hiệu suất. Cơ sở dữ liệu không quan hệ đã chỉ ra những lợi thế riêng của mình trong việc xử lý dữ liệu phi cấu trúc và bắt đầu trở thành đề tài chủ đạo trong phân tích Big Data. Mặc dù vậy, vẫn còn một số vấn đề về cơ sở dữ liệu không quan hệ trong hoạt động và những ứng dụng cụ thể của chúng. Điều này dẫn tới việc cần tìm một giải pháp thỏa hiệp giữa RDBMS và cơ sở dữ liệu không quan hệ. Ví dụ, một số doanh nghiệp đã sử dụng một kiến trúc cơ sở dữ liệu hỗn hợp mà tích hợp những ưu điểm của cả hai loại cơ sở dữ liệu như Facebook và Taobao.

- *Bảo mật dữ liệu:* Hầu như các nhà cung cấp dịch vụ hoặc chủ sở hữu dịch vụ Big Data có thể không duy trì và phân tích một cách hiệu quả các tập dữ liệu lớn

như vậy vì khả năng hạn chế của họ. Họ phải dựa vào các chuyên gia hoặc các công cụ để phân tích dữ liệu như vậy, làm tăng rủi ro bảo mật.

- *Quản lý năng lượng:* Năng lượng tiêu thụ của hệ thống máy tính lớn đã thu hút nhiều sự quan tâm từ cả quan điểm kinh tế và môi trường. Với sự gia tăng của dung lượng dữ liệu và nhu cầu phân tích, xử lý, lưu trữ và truyền tải thì Big Data chắc chắn sẽ tiêu thụ ngày càng nhiều năng lượng điện. Vì vậy, cơ chế kiểm soát và quản lý điện năng tiêu thụ cấp hệ thống sẽ được thành lập với Big Data trong khi khả năng mở rộng và khả năng tiếp cận được đảm bảo.

- *Khả năng mở rộng và thay đổi:* Hệ thống phân tích Big Data phải hỗ trợ tập dữ liệu hiện tại và tương lai. Thuật toán phân tích phải có khả năng xử lý các tập dữ liệu ngày càng mở rộng và phức tạp hơn.

- *Sự hợp tác*: Phân tích các dữ liệu lớn là một nghiên cứu liên ngành, trong đó yêu cầu các chuyên gia trong các lĩnh vực khác nhau hợp tác để thu thập các dữ liệu. Một kiến trúc mạng lưới Big Data toàn diện phải được thiết lập để giúp các nhà khoa học và kỹ sư trong các lĩnh vực khác nhau truy cập các loại dữ liệu khác nhau và sử dụng đầy đủ chuyên môn của họ, phối hợp để hoàn thành các mục tiêu phân tích.

1.4 Các phương pháp tiền xử lý dữ liệu cho BigData

Do dữ liệu được thu thập từ nhiều nguồn và thủ công nên có nhiều sai sót. Người ta chia giai đoạn thu thập và tiền xử lí dữ liệu thành các công đoạn như: lựa chọn dữ liệu, làm sạch, làm giàu, mã hóa dữ liệu. Các công đoạn được thực hiện theo trình tự đưa ra được một cơ sở dữ liệu thích hợp cho các giai đoạn sau. Tuy nhiên, tùy từng dữ liệu cụ thể mà quá trình trên được điều chỉnh cho phù hợp vì người ta đưa ra một phương pháp cho mọi loại dữ liệu.

Chọn lọc dữ liệu: Đây là bước chọn lọc các dữ liệu có liên quan trong các nguồn dữ liệu khác nhau. Các thông tin được chọn lọc sao cho có chứa nhiều thông tin liên quan tới lĩnh vực cần phát hiện tri thức đã xác định trong giai đoạn xác định vấn đề.

Làm sạch dữ liệu: Dữ liệu thực tế, đặc biệt dữ liệu lấy từ nhiều nguồn khác

nhau thường không đồng nhất. Do đó còn có biện pháp xử lí để đưa về một cơ sở dữ liệu thống nhất phục vụ cho khai thác. Nhiệm vụ làm sạch dữ liệu thường bao gồm: Điều hoà dữ liệu, xử lí các giá trị khuyết, xử lí nhiễu và các ngoại lệ.

Làm giàu dữ liệu: Việc thu nhập dữ liệu đôi khi không đảm bảo tính đầy đủ của dữ liệu. Một số thông tin quan trọng có thể thiếu hoặc không đầy đủ. Chẳng hạn, dữ liệu về khách hàng lấy từ một nguồn bên ngoài không có hoặc không đầy đủ thông tin về thu nhập. Nếu thông tin về thu nhập là quan trọng trong quá trình khai thác dữ liệu để phân tích hành vi khách hàng thì rõ ràng là ta không thể chấp nhận đưa các dữ liệu khuyết thiếu vào được.

Mã hóa: Các Phương pháp dùng để chọn lọc, làm sạch, làm giàu dữ liệu sẽ được mã hóa dưới dạng các thủ tục, chương trình hay tiện ích nhằm tự động hóa việc kết xuất, biến đổi và di chuyển dữ liệu. Các hệ thống con đó có thể được thực thi định kỳ làm tươi dữ liệu phục vụ cho việc phân tích**.**

1.5 Các hướng ứng dụng chính của BigData

Các tổ chức ngày càng sử dụng rộng rãi Big data và các ứng dụng có liên quan trong các lĩnh vực khác nhau, nhằm giảm thiểu rủi ro, hỗ trợ tổ chức trong việc quản lý hoạt động hàng ngày cũng như ra quyết định. Những ứng dụng của Big Data bao gồm:

Ứng dụng của Big Data trong các doanh nghiệp: Tìm ý muốn của khách hàng trong quá trình mua hàng. Làm sâu sắc hơn những phần người trước dịnh tính Biến từ định tính thành định lượng.

Ứng dụng của IoT dựa trên Big data: IOT không chỉ là 1 nguồn quan trọng của dữ liệu lướn, mà cũng là 1 trong những thị trường chính của những ứng dụng dữ liệu lớn. Vì sự đa dạng cao của các đối tượng, các ứng dụng của IoT cũng phát triển không ngừng. Trong kỷ nguyên của IoT, các cảm biến được nhúng vào trong các thiết bị di động như điện thoại di động, ô tô, và máy móc công nghiệp góp phần vào việc tạo và chuyển dữ liệu, dẫn đến sự bùng nổ của dữ liệu có thể thu thập được. Các doanh nghiệp giao thông vận tải, vận chuyển thường rất có kinh nghiệm

trong các ứng dụng của Big Data và IOT. Thành phố thông minh là 1 lĩnh vực nghiên cứu hot dựa trên các ứng dụng của dữ liệu IoT.

Ứng dụng của mạng xã hội trực tuyến theo định hướng dữ liệu lớn: Mạng xã hội trực tuyến là 1cấu trúc xã hội được cấu thành bởi các cá nhân dựa trên 1 mạng thông tin xã hội. Ứng dụng bao gồm mạng lưới phân tích quan điểm của công chúng, thu thập tình báo mạng và phân tích, marketing mạng xã hội, hỗ trợ ra quyết định của chính phủ, giáo dục trực tuyến… Những ứng dụng cơ bản của dữ liệu lớn từ mạng xã hội trực tuyến bao gồm: Các ứng dụng dựa trên nội dung: ngôn ngữ và văn bản là 2 hình thức quan trọng nhất của 1 thể hiện trong mạng xã hội. Thông qua việc phân tích ngôn ngữ và văn bản, có thể phân biệt được sở thích người dùng, cảm xúc, quan tâm và nhu cầu…

Các ứng dụng dựa trên cấu trúc: trong mạng xã hội, người dùng được biểu diễn như là các nút trong khi mối quan hệ xã hội, quan tâm và sở thích… tổng hợp các mối quan hệ giữa người sử dụng thành 1 cấu trục cụm. Nói chung, các ứng dụng dữ liệu lớn từ mạng xã hội trực tuyến có thể giúp hiểu rõ hơn về hành vi của người sử dụng và nắm vững các quy luật của accs hoạt động kinh tế xã hội từ 3 khía cạnh:

Cảnh báo sớm: Để nhanh chóng đối phó với cuộc khủng khoảng nếu có bằng chứng phát triển bất thường trong sự việc sử dụng các thiết bị và dịch vụ điện tử.

Giám sát thời gian thực: Phản hồi và theo dõi thời gian thực: có được phản hồi nhóm chống lại 1 số hoạt động xã hội dựa trên giám sát thời gian thực.

Ứng dụng trong y tế và chăm sóc sức khỏe: Dữ liệu y tế và chăm sóc sức khỏe được sinh ra liên tục nhanh chóng phát triển thành dữ liệu phức tạp, chứa các giá trị thông tin phong phú đa dạng. Big data có tiềm năng không giới hạn cho việc lưu trữ hiệu quả, xử lý, truy vấn và phân tích các dữ liệu y tế. Các ứng dụng của dữ liệu lớn y tế sẽ ảnh hưởng lớn đến các hoạt động chăm sóc sức khỏe.

Trí tuệ tập hợp: Nghiên cứu về dữ liệu cung cấp bởi 1 tập thể để đưa ra quyết định, dự đoán tốt hơn. Và lĩnh vực này được đem áp dụng cho mạng XH và phát huy tác dụng 1 cách đột phá.

2. Nghiên cứu một số lĩnh vực phân tích của Big Data

Phân tích dữ liệu đóng một vai trò hướng dẫn rất lớn trong việc xây dựng kế hoạch phát triển cho một quốc gia, sự hiểu biết về nhu cầu khách hàng trong thương mại và dự đoán xu hướng thị trường cho các doanh nghiệp. Phân tích dữ liệu lớn có thể được coi như các kỹ thuật phân tích cho một dạng đặc biệt của dữ liệu. Do đó, nhiều phương pháp phân tích dữ liệu truyền thống vẫn có thể được sử dụng để phân tích dữ liệu lớn, những phương pháp đó bắt nguồn từ thống kê và khoa học máy tính.

Phương pháp	Mô tả	Sử dụng
Bloom Filter	Bloom Filter bao gồm một loạt các hàm băm. Nguyên tắc của Bloom Filter là để lưu trữ các giá trị băm của dữ liệu khác với dữ liệu chính nó bằng cách sử dụng một mảng bit, mà bản chất là một chỉ số bitmap sử dụng hàm để tiến hành lưu trữ và nén dữ liệu.	Bloom Filter có hiệu quả không gian cao và tốc độ truy vấn cao.
Băm	Là một phương pháp mà chủ yếu biến đổi dữ liệu thành các giá trị số có chiều dài cố định ngắn hơn hoặc thành các giá trị chỉ số	Băm có những lợi thế như đọc, ghi nhanh và tốc độ truy vấn cao nhưng khó có hàm băm âm thanh
Đánh chỉ mục	Một chỉ mục là một cấu trúc riêng biệt trong cơ sở dữ liệu, nó được tạo ra bằng câu lệnh CREATE INDEX. Nó cần có không gian lưu trữ riêng	Đánh chỉ mục luôn là một phương pháp hiệu quả để giảm các chi phí của đọc, ghi ổ

	trên thiết bị lưu trữ (đĩa cứng) và có một phần bản sao của dữ liệu của bảng được lập chỉ mục. Điều này có nghĩa rằng việc tạo ra một chỉ mục là có sự dư thừa về dữ liệu. Tạo một chỉ mục không thay đổi dữ liệu của các bảng; nó chỉ tạo một cấu trúc dữ liệu mới và nó trỏ đến bảng ban đầu.	đĩa, cải thiện chèn, xóa, sửa đổi, tốc độ truy vấn trong cả cơ sở dữ liệu quan hệ truyền thống quản lý các dữ liệu có cấu trúc lẫn các công nghệ khác quản lý các dữ liệu bán cấu trúc và phi cấu trúc. Tuy nhiên, đánh chỉ mục có một bất lợi là nó có chi phí phụ thêm để lưu trữ các tập tin chỉ mục và cần được duy trì tự động khi dữ liệu được cập nhật\
Tính toán song song	Tính toán song song đề cập đến việc sử dụng đồng thời nhiều tài nguyên tính toán để hoàn thành một tác vụ tính toán. Ý tưởng cơ bản của nó là để phân tách một vấn đề và gắn chúng cho một số tiến trình riêng biệt để thực hiện một cách độc lập, do đó đạt được sự xử lý đồng thời.	Hiện nay, một số mô hình tính toán song song cổ điển bao gồm MPI (Message Passing Interface), Mapreduce và Dryad.

Bảng 1.1 Các phương pháp phân tích Big Data

Big Data có thể được dùng để phân tích trong nhiều lĩnh vực như: Bán lẻ, ngân hàng, dịch vụ chăm sóc sức khỏe, viễn thông, giải trí, bảo hiểm, giao thông,

giáo dục… theo 4 tiêu chí là: tối ưu hóa hoạt động, tăng trải nghiệm với khách hàng, tạo ra dịch vụ mới và quản trị rủi ro.

Phần trình bày sau đây sẽ đề cập đến một số khía cạnh việc sử dụng Big Data trong phân tích thực tế của một số lĩnh vực như lĩnh vực bán lẻ trong kinh doanh, lĩnh vực giáo dục của các tổ chức hay doanh nghiệp.

Lĩnh vực bán lẻ:

Dynamic Pricing (điều chỉnh giá linh hoạt): Thay vì chỉ áp dụng theo phương thức truyền thống là dựa vào cung cầu và hạn sử dụng của sản phẩm. Big data cho phép thay đổi giá dựa vào các yếu tố như thời tiết, địa điểm, lịch sử mua sắm của khách hàng. Amazon dùng Big data và thay đổi giá sản phẩm sau mỗi 10 phút, Walmart thay đổi giá 50.000 lần trong 1 tháng và giúp tăng doanh thu khoảng 26%.

*Phân tích giỏ hàng (basket analysis): T*rước đây thường dựa trên lịch sử các đơn hàng. Ví dụ: người dùng mua bỉm Merries hay mua kèm sữa Glico, từ đó các hãng bán lẻ có thể thiết kế gian hàng để bỉm Merries và sữa Glico gần nhau, hoặc khi khách hàng mua bỉm Merries sẽ khuyến nghị mua sữa. Với Big data, có thể thêm nhiều điều kiện khác để phân tích như thời gian mua hàng trong ngày, thời gian khách hàng mua sắm, thời tiết, thậm chí là loại nhạc được bật trong siêu thị hay thời gian chờ đợi để thanh toán.

Phân tích bỏ giỏ hàng (shopping cart defection): Có một số tính toán là khi người dùng vào website thì chỉ có 57% sẽ click chọn sản phẩm, và chỉ khoảng 5% là thêm vào giỏ hàng, tuy nhiên một nửa trong số này không tiến hành thanh toán. Big data dựa trên việc kết hợp các sản phẩm mà người dùng xem hoặc thêm vào giỏ hàng rồi từ đó dự đoán khả năng bỏ giỏ hàng, trong trường hợp đó việc khuyến mai giảm giá hoặc thêm voucher có thể giúp giảm khả năng bỏ giỏ hàng.

Tăng trải nghiệm khách hàng:

Hệ thống khuyến nghị: dựa trên lịch sử mua hoặc xem sản phẩm để đưa ra khuyến nghị sản phẩm tiếp theo mà khách hàng quan tâm. Những framework như Spark MLLib hoặc cơ sở dữ liệu dạng đồ thì như Titan, Neo4j cho phép triển khai

những thuật toán khuyến nghị theo cả hướng phân tán lẫn phân tích dạng đồ thị để nhận ra mối liên hệ ẩn giữa các nhóm khách hàng.

Duy trì khách hàng trung thành: Với sự phát triển của mạng xã hội, các diễn đàn, các website đánh giá, có thể tri ân (tích điểm, giảm giá) cho khách hàng nếu như họ có những nhận xét tích cực về sản phẩm, thương hiệu.

Tạo ra dịch vụ mới:

Điều chỉnh giá linh hoạt: Big data có thể tạo ra counter – dynamic pricing cho phép khách hàng quyết định thời điểm để mua hàng với giá tốt nhất. Ví dụ như startup Farecast (tích hợp trong Bing search) phân tích khoảng 200 tỷ vé máy bay để tìm ra thời điểm mua giá vé rẻ nhất cho khách hàng.

Kiếm tiền từ dữ liệu bán lẻ: Các hãng bán lẻ có thể bán thông tin ngược trở lại cho nhà cung cấp để nhà cung cấp có thể thay đổi chiến lược marketing hoặc sản xuất.

Quản trị rủi ro:

Phát hiện gian lận: Công cụ Big data có thể phát hiện những gian lận như dùng thẻ tín dụng ăn cắp để mua hàng trong thời gian thực.

Lĩnh vực giáo dục:

Tối ưu hóa hoạt động: Bằng việc thu thập, phân tích thông tin về sự nghiệp, mức lương, địa vị xã hội cũng những người đã tốt nghiệp các ngành học để đưa ra những cải tiến cho các ngành phù hợp hơn.

Tăng trải nghiệm khách hàng:

Cá nhân hóa giáo dục trực tuyến: dựa trên thông tin về lịch sử học, các môn học yêu thích, thời gian học… của học viên để cá nhân hóa bài giảng giúp cải thiện kết quả học tập. Tạo ra framework để làm các báo cáo phân tích dự đoán: tìm ra các biến chung dự đoán tình trạng bỏ học của học viên bằng việc kết hợp các cơ sở dữ liệu.

Tạo dịch vụ mới:

Đào tạo các nhà khoa học dữ liệu: với sự bùng nổ của dữ liệu, tất cả các ngành nghề đều cần đến các nhà khoa học dữ liệu để phân tích dự đoán trên các lĩnh

vực đó, vì vậy đào tạo các nhà khoa học dữ liệu là lĩnh vực mới cần thiết trong giáo dục.

Quản trị rủi ro:

Phát hiện gian lận báo cáo khoa học: Với sự phát triển của phân tích ngôn ngữ tự nhiên (NLP), các trang mạng xã hội, diễn đàn có thể giúp việc phát hiện những gian lận trong báo cáo khoa học.

Trên đây là một vài ví dụ áp dụng Big data trong hai ngành giáo dục và bán lẻ. Tương tự, trong bất cứ lĩnh vực nào cũng có thể đi theo 4 tiêu chí như trên để tìm ra các ứng dụng của Big data phù hợp nhằm tăng chất lượng dịch vụ, năng suất lao động.

3. Kết luận chương

Chương này trình bày một số nghiên cứu nền tảng về Big Data, nêu ra các định nghĩa mô tả và đặc trưng của Big Data, sự phát triển của Big Data cho đến thời điểm hiện tại cũng như các công nghệ liên quan, thách thức đối với Big Data. Ngoài ra, nội dung trong chương này còn trình bày các phương thức tiền xử lý dữ liệu cho Big Data và một số hướng ứng dụng của nó, đồng thời cũng trình bày một số lĩnh vực phân tích của Big Data mà bản chất là việc phân tích và khai phá dữ liệu để tìm tri thức.

CHƯƠNG 2: NGHIÊN CỨU MỘT SỐ CÁC PHƯƠNG PHÁP PHÂN TÍCH DỮ LIỆU TRÊN BẢNG QUYẾT ĐỊNH

2.1 Nghiên cứu khái quát hướng khai phá dữ liệu sử dụng lý thuyết tập thô

Lý thuyết tập thô do Z.Pawlak đề xuất vào những năm đầu thập niên 80 của thế kỷ XX, được xem là công cụ hữu hiệu để giải quyết các bài toán phân lớp, phát hiện luật,… chứa dữ liệu mơ hồ không chắc chắn. Từ khi xuất hiện, lý thuyết tập thô đã được sử dụng hiệu quả trong các bước của quá trình khai phá dữ liệu và khám phá tri thức, bao gồm tiền xử lý dữ liệu, trích lọc các tri thức tiềm ẩn và đánh giá kết quả thu được. Việc sử dụng lý thuyết tập thô vào khai phá dữ liệu thu hút được sự quan tâm của nhà khoa học. Một trong những nhánh quan trọng của nghiên cứu này là nghiện cứu việc rút gọn thuộc tính trên bảng quyết định. Mục tiêu của việc rút gọn trên bảng quyết định là tìm tập thuộc tính rút gọn mà bảo toàn thông tin phân lớp của bảng quyết định. Với bảng quyết định cho trước số lượng các tập rút gọn là hàm mũ theo số thuộc tính. Trong thực tế không đòi hỏi phải tìm tất cả các tập rút gọn mà chỉ cần tìm được một tập rút gọn tốt nhất theo một tiêu chuẩn đánh giá nào đó là đủ. Mỗi phương pháp rút gọn thuộc tính đều đưa ra định nghĩa tập rút gọn và xây dựng thuật toán tìm tập rút gọn tốt nhất theo tiêu chuẩn đánh giá chất lượng phân lớp của thuộc tính, còn gọi là độ quan trọng của thuộc tính. Một số phương pháp được nhiều người quan tâm là phương pháp sử dụng miền dương, phương pháp sử dụng Entropy Shanon, phương pháp sử dụng Entropy Liang

2.1.1 Những khái niệm cơ bản trong lý thuyết tập thô

Hệ thông tin là công cụ biểu diễn tri thức dưới dạng một bảng dữ liệu gồm p cột ứng với p thuộc tính và n hàng ứng với n đối tượng.

Định nghĩa 1.1. Hệ thông tin là một bộ tứ *IS=(U,A,V,f)* trong đó *U* là một tập hữu hạn, khác rỗng các đối tượng, *A* là một tập hữu hạn, khác rỗng các thuộc tính, $V=\bigcup_{a\in A} V_a$ với V_a là tập giá trị các thuộc tính $a\in A$; $f: U\times A\to V_a$ là hàm thông

tin, $\forall a \in A, u \in U \ f(u, a) \in V_a$.

Với mọi $u \in U$, $a \in A$ ta ký hiệu giá trị thuộc tính a tại đối tượng u là $a(u)$ thay vì $f(u,a)$. Nếu $B=\{b_1,b_2, ...,b_k\} \subseteq A$ là một tập con các thuộc tính thì ta ký hiệu bộ các giá trị $b_i(u)$ bởi $B(u)$. Như vậy, nếu u và v là hai đối tượng, thì ta viết $B(u)=B(v)$, nếu $b_i(u)=b_i(v)$ với mọi $i = 1,...,k$.

Cho hệ thông tin $IS=(U,A,V,f)$, nếu tồn tại $u \in U$ và $a \in A$ sao cho $a(u)$ thiếu giá trị (missing value) thì IS được gọi là *hệ thông tin không đầy đủ*, trái lại IS được gọi là *hệ thông tin đầy đủ*. Ta tự hiểu hệ thông tin đầy đủ được gọi tắt là *hệ thông tin*.

Xét hệ thông tin $IS=(U,A\ V,f)$. Mỗi tập con các thuộc tính $P \subseteq A$ xác định một quan hệ hai ngôi trên U, ta ký hiệu $IND(P)$, xác định bởi $IND(P)=\{(u,v) \in U \times U \mid \forall a \in P, a(u)=a(v)\}$. $IND(P)$ là quan hệ P – *không phân biệt được*. Dễ thấy rằng $IND(P)$ là một quan hệ tương đương trên U. Nếu $(u,v) \in IND(P)$ thì hai đối tượng u và v không phân biệt được bởi các thuộc tính trong P. Quan hệ tương đương $IND(P)$ xác định một phân hoạch U/P chứa đối tượng u là $[u]_p$, khi đó $[u]_p=\{v \in U/(u,v) \in IND(P)\}$.

Định nghĩa 1.2. [2] Cho hệ thông tin $IS=(U,A,V,f)$, và $P, Q \subseteq A$.

1) Phân hoạch U/P và phân hoạch U/Q là như nhau (viết $U/P=U/Q$), khi và chỉ khi $\forall u \in U, [u]_P =[u]_Q$.

2) Phân hoạch U/P mịn hơn phân hoạch U/Q là như nhau *(viết* $U/P \sqsubseteq U/Q$), khi và chỉ khi $\forall u \in U, [u]_P \subseteq [u]_Q$.

Tính chất 1.1 [2] *Xét hệ thông tin* $IS=(U,A,V,f)$ *và* $P, Q \subseteq A$.

1) *Nếu* $P \subseteq Q$ *thì* $U/Q \leq U/P$, mỗi lớp của U/P là một lớp hoạch hợp của một số lớp thuộc U/Q.

2) Với mọi $u \in U$ ta có $[u]_{P \cup Q} = [u]_p \cap [u]_Q$.

2.1.2 Mô hình tập thô truyền thống

Cho hệ thông tin IS=(U,A,V,f), và tập đối tượng X⊆U. Với một tập thuộc tính B⊆A cho trước chúng ta có các lớp tương đương của phân hoạch U/B, thế thì một tập đối tượng X có thể biểu diễn thông qua các lớp tương đương này như thế nào?

Trong lý thuyết tập thô, để biểu diễn X thông qua các lớp tương đương của U/B (còn gọi là biểu diễn X bằng tri thức có sẵn trong B), người ta xấp xỉ X bởi hợp của một số hữu hạn các lớp tương đương của U/B. Có hai cách xấp xỉ tập đối tượng X thông qua tập thuộc tính B, được gọi là B xấp xỉ dưới và B xấp xỉ trên của X, ký hiệu lần lượt là $\underline{B}X$ và $\overline{B}X$ được xác định như sau: $\underline{B}X$ ={u∈U /[u]B⊆X}, $\overline{B}X$ ={u∈U /[u]B∩X ≠ ∅};

Tập $\underline{B}X$ bao gồm tất cả các phần tử của U chắc chắn thuộc vào X, còn tập $\overline{B}X$ bao gồm các phần tử của *U* có thể thuộc vào *X* dựa trên tập thuộc tính *B*. Từ hai tập xấp xỉ nêu trên, ta định nghĩa các tập. $BN_B(X) = \overline{B}X - \underline{B}X$ *: B miền biên* của *X, U-* $\overline{B}X$ *:* B *miền ngoài* của *X*. *B miền biên* của *X* là tập chứa các đối tượng có thể *thuộc hoặc không thuộc X,* còn *B miền ngoài* của *X* chứa các đối tượng *chắc chắn không thuộc X*. Sử dụng các lớp của phân hoạch *U/B*, các xấp xỉ dưới và trên của *X* có thể viết lại $\underline{B}X = \cup\{Y \in U/B \mid Y \subseteq X\}$, $\overline{B}X = \cup\{Y \in U/B \mid Y \cap X \neq \emptyset\}$.

Trong trường hợp $BN_B(X) = \emptyset$ thì *X* được gọi là *tập chính xác* (exact set), ngược lại *X* được gọi là *tập thô* (rough set). Với *B,D⊆A*, ta gọi *B* - miền dương của *D* là tập được xác định như sau $POS_B(D) = \bigcup_{X \in U/D} (\underline{B}X)$. Rõ ràng $POS_B(D)$ là tập tất cả các đối tượng *u* sao cho với mọi *v∈U* mà *u*(*B*)=*v(B)* ta đều có *u*(*D*)= *v*(*D*). Nói cách khác, $POS_B(D) = \{u \in U \mid [u]_B \subseteq [u]_D\}$.

Ví dụ 1.1 Xét hệ thông tin biểu diễn các triệu chứng cúm của bệnh nhân như sau

U	Đau đầu	Thân nhiệt	Cảm cúm

U_1	Có	Bình thường	Không
U_2	Có	Cao	Có
U_3	Có	Rất cao	Có
U_4	Không	Bình thường	Không
U_5	Không	Cao	Không
U_6	Không	Rất cao	Có
U_7	Không	Cao	Có
U_8	Không	Rất cao	Không

Bảng 2.1 Bảng thông tin về bệnh cúm

Ta có: $U/\{Đau\ đầu\}=\{\{u_1, u_2, u_3\}, \{u_4, u_5, u_6, u_7, u_8\}\}$,

$U/\{Thân\ nhiệt\}=\{\{u_1, u_4\}, \{u_2, u_5, u_7\}, \{u_3, u_6, u_8\}\}$,

$U/\{Cảm\ cúm\}=\{\{u_1, u_4, u_5, u_8\}, \{u_2, u_3, u_6, u_7, \}\}$,

$U/\{Đau\ đầu, Cảm\ cúm\}=\{\{u_1\}, \{u_2, u_3\}, \{u_4, u_5, u_8\}, \{u_6, u_7\}\}$.

Như vậy, các bệnh nhân u_2, u_3 không phân biệt được về đau đầu và cảm cúm, nhưng phân biệt được về thân nhiệt.

Các lớp không phân biệt được *bởi* $B=\{Đau\ đầu, Thân\ nhiệt\}$ là:

$\{u_1\}, \{u_2\}, \{u_3\}, \{u_4\}, \{u_5, u_7\}, \{u_6, u_8\}$.

Đặt $X=\{u/u\ (Cảm\ cúm)= Có\}=\{u_2, u_3, u_6, u_7\}$. *Khi đó:*

$\underline{B}X=\{u_2, u_3\}$,

$\overline{B}X=\{u_2,u_3,u_5,u_6,u_7,u_8\}$. *Như vậy, B miền biên của X là tập hợp*

$BN_B(X)=\{u_5,u_6,u_7,u_8\}$. Nếu *đặt* $D=\{Cảm\ cúm\}$ thì:

$U/D=\{X_1=\{u_1,u_4,u_5,u_8\}; X_2=\{u_2,u_3,u_6,u_7\}\}$,

$\underline{B}X_1=\{u_1,u_4\}$, $\overline{B}X_2=\{u_2,u_3\}$, $POS_B(D)=\bigcup_{X\in U/D}(\underline{B}X)=\{u_1,u_2,u_3,u_4\}$.

Với các khái niệm của tập xấp xỉ đối với phân hoạch U/B, các tập thô được chia thành bốn lớp cơ bản:

1) Tập X là B - *xác định thô* nếu $\underline{B}X\neq\emptyset$ và $\overline{B}X\neq U$

2) Tập X là B - *không xác định trong* nếu $\underline{B}X=\emptyset$ và $\overline{B}X\neq U$

3) Tập X là B - *không xác định ngoài* nếu $\underline{B}X \neq \emptyset$ và $\overline{B}X$ =U

4) Tập X là B - *không xác định hoàn toàn* nếu $\underline{B}X = \emptyset$ và $\overline{B}X$ =U.

Bảng quyết định đầy đủ:

Một lớp đặc biệt của hệ thông tin có vai trò quan trọng trong nhiều ứng dụng là bảng quyết định. Bảng quyết định là một hệ thông tin *DS* với tập thuộc tính *A* được chia thành hai tập khác rỗng rời nhau *C* và *D*, lần lượt được gọi là tập thuộc tính điều kiện và tập thuộc tính quyết định. Tức là $DS=(U,C\cup D,V,f)$ với $C \cap D=\emptyset$.

Xét bảng quyết định $DS=(U,C\cup D,V,f)$ với giả thiết mọi $u\in U$, $\forall d\in D$, $d(u)$ đầy đủ giá trị, nếu tồn tại $u\in U$ và $c\in C$ sao cho $c(u)$ thiếu giá trị thì *DS* được gọi là bảng quyết định không đầy đủ, trái lại *DS* được gọi là bảng quyết định đầy đủ. Trong luận văn này, bảng quyết định đầy đủ được gọi tắt là bảng quyết định.

Bảng quyết định *DS* được gọi là nhất quán nếu *D* phụ thuộc vào *C*, tức là với mọi $u,v\in U$, $C(u)=C(v)$ kéo theo $D(u)=D(v)$. Ngược lại thì gọi là không nhất quán hay mâu thuẫn. Theo định nghĩa miền dương, bảng quyết định là nhất quán khi và chỉ khi $POS_C(D)=U$. Trong trường hợp bảng không nhất quán thì $POS_C(D)$ chính là tập con cực đại của *U* sao cho phụ thuộc hàm $C \rightarrow D$ đúng.

Tập rút gọn và tập lõi:

Trong bảng quyết định, các thuộc tính điều kiện được phân thành 3 nhóm: thuộc tính lõi (core attribute), thuộc tính rút gọn (reductive attribute) và thuộc tính dư thừa (redundant attribute). Thuộc tính lõi là thuộc tính không thể thiếu trong việc phân lớp chính xác tập dữ liệu. Thuộc tính lõi xuất hiện trong tất cả các tập rút gọn của bảng quyết định. Thuộc tính dư thừa là những thuộc tính mà việc loại bỏ chúng không ảnh hưởng đến việc phân lớp tập dữ liệu, thuộc tính dư thừa không xuất hiện trong bất kỳ rút gọn nào của bảng quyết định. Thuộc tính rút gọn là thuộc tính xuất hiện trong một tập rút gọn nào đó của bảng quyết định. Chúng ta sẽ đưa ra các định nghĩa chính xác trong phần tiếp theo.

Định nghĩa 1.3. [11] (tập lõi dựa trên miền dương) Cho bảng quyết định $DS=(U,C\cup D,V,f)$. Thuộc tính $c\in C$ được gọi là không cần thiết (dispensable) trong *DS* dựa trên miền dương nếu $POS_C(D)=POS_{(C-\{d\})}(D)$. Ngược lại, c được gọi là cần

thiết (indispensable). Tập tất cả các thuộc tính cần thiết trong DS được gọi là tập lõi dựa trên miền dương và được ký hiệu là *PCORE (C)*. Khi đó, thuộc tính cần thiết chính là thuộc tính lõi.

Theo định nghĩa 1.3 thuộc tính không cần thiết được gọi là thuộc tính dư thừa hoặc thuộc tính rút gọn.

Định nghĩa 1.4. [11] (tập rút gọn dựa trên miền dương) Cho bảng quyết định $DS=(U,C\cup D,V,f)$. Và tập thuộc tính $R\subseteq C$.

Nếu 1) $POS_R(D)=POS_C(D)$

2) $\forall r\in R, POS_{R-\{r\}}(D)\neq POS_C(D)$

Thì R là một tập rút gọn của C dựa trên miền dương.

Tập rút gọn định nghĩa như trên còn gọi là tập rút gọn Pawlak. Ký hiệu *PRED(C)* là họ tất cả các tập rút gọn Pawlak của C. Khi đó $PCORE(C)=\bigcup_{R\in PRED(C)} R$.

Định nghĩa 1.5. Cho bảng quyết định $DS=(U,C\cup D,V,f)$. Và $a\in C$. Ta nói rằng a là thuộc tính rút gọn của DS nếu tồn tại 1 tập rút gọn $R\in PRED(C)$ sao cho $a\in R$.

Định nghĩa 1.6. Cho bảng quyết định $DS=(U,C\cup D,V,f)$. Và $a\in C$. Ta nói rằng a là thuộc tính dư thừa của DS nếu $a\in C-\bigcup_{R\in PRED(C)} R$.

Ví dụ 1.2. Xét bảng quyết định về bệnh cúm như sau

U	**Mệt mỏi**	**Đau đầu**	**Đau cơ**	**Thân nhiệt**	**Cảm cúm**
u_1	Có	Có	Có	Bình thường	Không
u_2	Có	Có	Có	Cao	Có
u_3	Có	Có	Có	Rất cao	Có
u_4	Có	Không	Có	Bình thường	Không
u_5	Có	Không	Không	Cao	Không
u_6	Có	Không	Có	Rất cao	Có

Bảng 2.2 Bảng quyết định về bệnh cúm

Bảng này có hai tập rút gọn là R_1=*{Đau cơ, Thân nhiệt}* và R_2=*{Đau đầu, thân nhiệt}*. Như vậy tập lõi là *PCORE(C)={Thân nhiệt}* và Thân nhiệt là thuộc lõi

duy nhất. Các thuộc tính không cần thiết bao gồm:

+ Thuộc tính Mệt mỏi là thuộc tính dư thừa vì không tham gia vào rút gọn nào.

+ Hai thuộc tính Đau đầu và Đau cơ là hai thuộc tính rút gọn vì đều có mặt trong một tập rút gọn. Hai thuộc tính này đều không cần thiết theo nghĩa là, từ bảng dữ liệu, có thể loại bỏ một trong hai thuộc tính này mà vẫn chuẩn đoán đúng bệnh. Tức là

$$POS_{\{Đau\ cơ,\ Thân\ nhiệt\}}(\{Cảm\ cúm\}) = POS_C(\{Cảm\ cúm\})$$

$$POS_{\{Đau\ đầu,\ Thân\ nhiệt\}}(\{Cảm\ cúm\}) = POS_C(\{Cảm\ cúm\}).$$

2.2 Nghiên cứu phân tích một số thuật toán liên quan đến tập rút gọn trong bảng quyết định rút gọn nhất quán:

2.2.1 Đặt vấn đề

Bảng quyết định trong các bài toán thực tế thường chứa một số thuộc tính dư thừa thực sự, thuộc tính dư thừa là những thuộc tính mà việc loại bỏ chúng không ảnh hưởng gì đến việc phân lớp tập đối tượng. Sự có mặt của các thuộc tính này làm cho độ phức tạp tính toán của bài toán khai phá dữ liệu tăng lên rất lớn. Việc loại bỏ các thuộc tính này trước khi thực hiện các nhiệm vụ khai phá dữ liệu có ý nghĩa thực tiễn cao trong bối cảnh dữ liệu ngày càng lớn, ngày càng đa dạng và phức tạp.

Thuộc tính dư thừa thực sự là thuộc tính không xuất hiện trong bất kỳ tập rút gọn nào và thuộc tính rút gọn là thuộc tính xuất hiện trong một tập rút gọn nào đó. Khi đó, bài toán tìm tập tất cả các thuộc tính dư thừa thực sự tương đương với bài toán tìm tập tất cả các thuộc tính rút gọn. Để giải quyết bài toán này, phương pháp tiếp cận thông thường là tìm họ tất cả các tập rút gọn của bảng quyết định, sau đó tìm phép hợp giữa các tập rút gọn. Tuy nhiên, cách tiếp cận này không khả thi với các bảng dữ liệu kích thước lớn vì độ phức tạp thời gian của thuật toán tìm họ tất cả các tập rút gọn của bảng quyết định là hàm mũ đối với số thuộc tính điều kiện.

Trong phần này, chúng tôi trình bày một số thuật toán liên quan đến các thuộc tính rút gọn của bảng quyết định nhất quán có độ phức tạp thời gian là đa thức.

2.2.2 Thuật toán tìm tất cả các thuộc tính rút gọn

Trong cơ sở dữ liệu quan hệ, Demetrovics J. và Thi V.D [6] đã chứng minh bổ đề quan trọng sau.

Bổ đề 1. [2] Giả sử $\mathcal{K}$ là một hệ Sperner trên R, khi đó $\bigcup_{k\in\mathcal{K}} K$ =R- $\bigcap_{k\in\mathcal{K}^{-1}} K$. Trên quan hệ r, do $\mathcal{K}_a^r$ là hệ Sperner trên R nên áp dụng Bổ đề 1 ta có bổ đề sau.

Bổ đề 2. Cho r là một quan hệ trên R và a∈R, khi đó $\bigcup_{K\in\mathbf{K}_a^r} K = R - \bigcap_{K\in(\mathbf{K}_a^r)^{-1}} K$

Cho bảng quyết định nhất quát *DS* =(U,$C\cup\{d\}$,V,f) với U ={ $u_1, u_2,...,u_m$ }.

Xét quan hệ r={ $u_1, u_2,...,u_m$ } trên tập thuộc tính R=$C\cup\{d\}$, từ khái niệm tập rút gọn của bảng quyết định nhất quán và tập tối thiểu của một thuộc tính trên quan hệ được trình bầy ở trên ta có *PRED*(C)= κ_d^r -{d}, với *PRED*(C) là họ tất cả các tập rút gọn Pawlak của C trong *DS* và k_d^r là họ tập tối thiểu của thuộc thính d trên r. Do đó, nếu kí hiệu *REAT*(C) là tập tất cả các thuộc tính rút gọn của C thì *REAT*(C)= $\bigcup_{R\in PRED(C)} R$=$\left(\bigcup_{R\in k_d^r} R\right)$-{$d$}

Thuật toán 1[2]: Tìm tập tất cả các thuộc tính rút gọn.

Đầu vào: Bảng quyết định *DS*=(U,C,$\cup\{d\}$,V,f) với $POS_C(\{d\})=U$, C={ c_1, c_2,..., c_n }, U ={ $u_1, u_2,...,u_m$ }

Đầu ra: *REAT* (C) là tập tất cả các thuộc tính rút gọn của C.

Xét quan hệ r={ $u_1, u_2,...,u_m$ } trên tập thuộc tính R=$C\cup\{d\}$.

Bước 1. từ r ta tính hệ bằng nhau ε_r ={ E_{ij} :1≤i<j≤m} với E_{ij} ={a∈R:a(u_i)=a(u_j)}.

Bước 2. Từ ε_r ta xây dựng tập M_d ={$A\in\varepsilon_r$: $d\notin A \nexists B\in\varepsilon_r$: $d\notin B, A\subset B$}

Bước 3. Xây dựng tập $V = R - \bigcap_{K\in M_d} K$

Bước 4. Đặt *REAT (C)* =V-{d} .

Tập *REAT*(C) được xây dựng tập tất cả các thuộc tính rút gọn của C.

Chứng minh: Theo cách xây dựng μ_d tại Bước 2 và theo công thức tính bao đóng

của tập thuộc tính trên quan hệ, $\forall A \in \mu_d$ ta có $A_r^+ = A$ và A không chứa d nên A_r^+ không chứa d, suy ra $A \to \{d\} \notin F^+$. Mặt khác, nếu tồn tại B sao cho $A \subset B$ thì xảy ra hai trường hợp: (1) Nếu B không chứa d thì $B_r^+ = R$; (2) Nếu B chứa d thì hiển nhiên B_r^+ chứa d. Cả hai trường hợp ta đều có B_r^+ chứa d hay $B \to \{d\} \in F^+$. Do đó $\mu_d = MAX(F^+, d)$ với $MAX(F^+, d) = \{A \subseteq R : A \to \{d\} \notin F^+, A \subset B \Rightarrow B \to \{d\} \in F^+\}$. Theo [3], $MAX(F^+, d) = (K_d^r)^{-1}$ với K_d^r là họ các tập tối thiểu của thuộc tính d trên quan hệ r. Do đó $\mu_d = (K_d^r)^{-1}$. Tại Bước 3 kết hợp với Bổ đề 2 ta có:

Ta có $V = R - \bigcap_{K \in M_d} K = R - \bigcap_{K \in (\mathbf{K}_d^r)^{-1}} K = \bigcup_{K \in \mathbf{K}_d^r} K$

Tại Bước 4 ta có $REAT(C) = V - \{d\} = \left(\bigcup_{k \in k_d{}^r} K \right) - \{d\} = \bigcup_{R \in PRED(C)} R$.

Do đó theo định nghĩa , *REAT (C)* là tập tất cả các thuộc tính rút gọn của C.

Độ phức tạp thời gian của thuật toán:

Với m là số đối tượng và n là thuộc tính điều kiện, độ phức tạp thời gian để tính hệ bằng nhau ε_r tại *bước 1 là* $O(m^2 n)$. Tại *Bước 2*, hệ bằng nhau ε_r có tối đa m^2 phần tử. Do đó, độ phức tạp thời gian để tính tập μ_d là $O(m^4 n)$. Vì vậy, độ phức tạp thời gian của thuật toán là $O(m^4 n)$. Độ phức tạp này là đa thức theo số hàng và số cột của bảng quyết định *DS*.

Hệ quả 2.1. *Cho trước bảng quyết định nhất quán* $DS = (U, C \cup \{d\}, V, f)$ *và thuộc tính a, tồn tại thuật toán xác định thuộc tính a là thuộc tính rút gọn hay không với thời gian đa thức theo số hàng và số cột của DS.*

Hệ quả 2.2. *Cho trước bảng quyết định nhất quán* $DS = (U, C \cup D, V, f)$ *và thuộc tính a, tồn tại thuật toán xác địnhj thuộc tính a là thuộc tính dư thừa thực sự hay không với thời gian đa thức theo số hàng và số cột của DS.*

2.2.3 Thuật toán tìm một tập rút gọn

Thuật toán 2. [2]: Tính bao đóng của tập thuộc tính trong quan hệ.

Đầu vào: $r = \{h_1, \ldots, h_m\}$ là quan hệ trên $R = \{a_1, \ldots, a_n\}$ và $A \subseteq R$.

Đầu ra: A_r^+

Bước 1: Từ r xây dựng tập $E_r = \{E_{ij} : m \geq j \geq i \geq 1\}$ với $E_{ij} = \{a \in R$ và $h_i(a) = h_j(a)\}$.

Bước 2: Xây M =$\{B$: tồn tại E_{ij} để $B = E_{ij}\}$.

Bước 3:

Đặt $A_r^+ = \cap B$ nếu tồn tại $B \in M : A \subseteq B$. Ngược lại $A_r^+ = R$.

Có thể thấy rằng thuật toán này có độ phức tạp tính toán là đa thức với n và m.

Có thể thấy rằng $A \to B \in F^+$ trong quan hệ *r* khi và chỉ khi $B \subseteq A_r^+$.

Ví dụ 3: Cho quan hệ r = {h_1, h_2, h_3, h_4, h_5, h_6, h_7} trên tập thuộc tính R = {a, b, c, d, e}, cụ thể như sau.

a	**b**	**c**	**D**	**e**
0	0	1	1	0
1	1	0	0	0
0	0	1	0	1
1	0	0	0	1
0	1	0	1	0
1	0	1	1	1
1	0	0	0	0

Bảng 2.3 Bảng dữ liệu tính bao đóng

Hãy tính $\{a, b\}_r^+$. Dễ thấy,

E_{12} = e, E_{13} = abc, E_{14} = b, E_{15} = ade,

E_{16} = bcd, E_{17} = be, E_{23} = d, E_{24} = acd, E_{25} = bce, E_{26} = a, E_{27} = ade,

E_{34} = bde, E_{35} = a, E_{45} = c, E_{46} = abe, E_{47} =abcd,

E_{56} = d, E_{57} = ce, E_{67} = ab.

E_r ={e, abc, b, ade, bcd, be, d, acd, bce, a, ade, bde, a, c, abe, abcd, d, ce, ab}

M = {d, abc, b, ade, bcd, be, acd, bce,a bde, c, abe, abcd, ce, ab}.

Khi đó $\{a, b\}_r^+ = \{abc\} \cap \{abcd\} \cap \{ab\} = \{ab\}$.

Trên cơ sở thuật toán tìm một tập tối thiểu của một thuộc tính trong [1], ta đưa ra thuật toán sau:

Trên cơ sở thuật toán tìm một tập tối thiểu của một thuộc tính trong [3], ta đưa ra thuật toán sau:

Thuật toán 3[2]: Thuật toán tìm một rút gọn trong bảng quyết định nhất quán.

Đầu vào: Bảng quyết định $DS = (U, C \cup \{d\}, V, f\})$ với

$POS_C(\{d\}) = U$, $C = \{c_1, c_2, ..., c_n\}$, $U = \{u_1, u_2, ..., u_m\}$.

Đầu ra: A là tập rút gọn của *DS*.

Xét quan hệ $r = \{u_1, u_2, ..., u_m\}$ trên tập thuộc tính $R = C \cup \{d\}$.

Bước 1. Từ *r* ta tính hệ bằng nhau

$\varepsilon_r = \{E_{ij} : 1 \le i < j \le m\}$ với $E_{ij} = \{a \in R : a(u_i) = a(u_j)\}$.

Bước 2. Từ E_r xây M = {B: Tồn tại E_{ij} để B = E_{ij}}.

Bước 3. Đặt L(0) = C.

Bước i + 1. Đặt $L(i + 1) = L(i) - a_{i+1}$ nếu $\{d\} \subseteq \{L(i) - a_{i+1}\}_r^+$. Ngược lại, đặt L(i+1) = L(i).

Khi đó A = L(n).

Có thể thấy thuật toán này có độ phức tạp tính toán là đa thức với n và m. Ta cũng có thể thấy rằng, nếu hoán vị các phần tử của C, ta có thể nhận được một rút gọn khác của DS.

Trên cơ sở bảng quyết định trong ví dụ 4 sau đây, tìm một rút gọn cho bảng này.

Ví dụ 4: Cho bảng quyết định $DS = (U, C \cup \{d\}, V, f\})$ với U = {u_1, u_2, u_3, u_4, u_5} và C = {a, b, c} cụ thể như sau.

a	b	C	d
1	0	1	1
0	1	1	0
1	1	1	0
1	0	0	0
1	1	0	1

Bảng 2.4 Bảng dữ liệu đầu vào tìm một tập rút gọn

Như vậy, ta có:

E_{12} = c, E_{13} = ac, E_{14} = ab, E_{15} = ad,

E_{23} = bcd, E_{24} = d, E_{25} = b,

E_{34} = ad, E_{35} = ab, E_{45} = ac,

E_r = {c, ac, ab, ad, bcd, d, b, ad, ab, ad}.

M = {c, ac, ab, ad, bcd, d, b}.

L(0) = abc.

L(1) = bc vì $\{d\} \subseteq \{bc\}_r^+ = bcd$.

L(2) = bc vì $\{c\}_r^+ = c$ và L(3) = L(2) = bc vì $\{b\}_r^+ = b$.

Như vậy tập {b, c} là một rút gọn của bảng quyết định trên.

2.2.4 Thuật toán tìm họ tất cả các tập rút gọn

Cho bảng quyết định nhất quán: $DS = (U, C \cup \{d\}, V, f\})$ với $U = \{u_1, u_2, ..., u_m\}$. Theo nội dung trình bày ở trên ta có $PRED(C) = \mathbf{K}_d^r - \{d\}$ với *PRED(C)* là họ tất cả các tập rút gọn Pawlak của *C* và $\mathbf{K}_d^r$ là họ các tập tối thiểu của thuộc tính *d* trên *r*. Từ kết quả này, xây dựng thuật toán tìm họ tất cả các tập rút gọn Pawlak của *C*, gọi tắt là họ các tập rút gọn của *C*.

Thuật toán 4. [1]: Tìm tập các khoá tối thiểu từ tập các phản khoá.

Đầu vào: Cho K = $\{B_1, \ldots, B_m\}$ là một hệ Sperner trên R.

Đầu ra: H mà H^{-1} = K.

Trong [1], đã chứng minh rằng độ phức tạp của thuật toán này là hàm mũ đối với lực lượng của tập các thuộc tính.

Trong [4], ta có kết quả sau.

Thuật toán 5[2]: Tìm họ tất cả các tập rút gọn của tập thuộc tính điều kiện.

Đầu vào: Bảng quyết định $DS = (U, C \cup \{d\}, V, f\})$ với $POS_C(\{d\}) = U$, $C = \{c_1, c_2, \ldots, c_n\}$, $U = \{u_1, u_2, \ldots, u_m\}$.

Đầu ra: *PRED(C).*

Xét quan hệ $r = \{u_1, u_2, \ldots, u_m\}$ trên tập thuộc tính $R = C \cup \{d\}$.

Bước 1. Từ *r* xây dựng quan hệ bằng nhau:

$\varepsilon_r = \{E_{ij} : 1 \le i < j \le m\}$ với $E_{ij} = \{a \in R : a(u_i) = a(u_j)\}$.

Bước 2. Từ ε_r xây dựng tập

$M_d = \{A \in \varepsilon_r : d \notin A \nexists B \in \varepsilon_r : d \notin B, A \subset B\}$.

Bước 3. Sử dụng Thuật toán 4 tính tập $\mathbf{K}$ từ tập M_d (M_d = $\mathbf{K}^{-1}$).

Bước 4. Đặt *PREA(C)* = $\mathbf{K}$ - {*d*}.

Trong [2, 142-143] và tại Bước 4, *PRED(C)* = $\mathbf{K}_d^r$ - {*d*} là họ tất cả các tập rút gọn của bảng quyết định.

Vì phải sử dụng Thuật toán 2, có thể thấy, độ phức tạp tính toán của Thuật toán 3 là hàm mũ đối với số thuộc tính của bảng quyết định.

Ví dụ 6: Cho bảng quyết định

$DS = (U, C \cup \{d\}, V, f\})$ với U = {u_1, u_2, u_3, u_4, u_5} và C = {a, b, c, e, f, d} cụ thể như sau.

a	**b**	**c**	**e**	**f**	**d**
0	0	0	0	0	0
0	1	0	0	0	1
2	0	2	0	0	2
3	0	0	3	0	3
4	0	0	0	4	4

Bảng 2.5 Bảng dữ liệu đầu vào tìm họ tất cả các tập rút gọn

Có thể thấy, E_{12} = acef, E_{13} = bef, E_{14} = bcf, E_{15} = bce,

E_{23} = ef, E_{24} = cf, E_{25} = ce,

E_{34} = bf, E_{35} = be, E_{45} = bc.

Từ các E_{ij} trên ta thấy M_d = {acef, bef, bcf, bce}.

Sử dụng Thuật toán 4 ta tính được tập K từ K^{-1} với $K^{-1} = M_d$.

Có thể thấy K = {abc, bcef}. Vậy:

PRED(C) = K – {d} = {abc, bcef}.

Bổ đề 3. [2] Cho bảng quyết định nhất quán $DS = (U, C \cup \{d\}, V, f\})$ với $C = \{c_1, c_2, \ldots, c_n\}$, $U = \{u_1, u_2, \ldots, u_m\}$.

Xét quan hệ $r = \{u_1, u_2, \ldots, u_m\}$ trên tập thuộc tính $R = C \cup \{d\}$.

Đặt $\varepsilon_r = \{E_{ij} : 1 \le i < j \le m\}$ với $E_{ij} = \{a \in R : a(u_i) = a(u_j)\}$.

Đặt $M_d = \{A \in \varepsilon_r : d \notin A, \nexists B \in \varepsilon_r : d \notin B, A \subset B\}$.

Thì $M_d = \left(\mathbf{K}_d^r\right)^{-1}$. Ở đây $\mathbf{K}_d^r$ là họ các tập tối thiểu của thuộc tính $\{d\}$ trên quan hệ r.

Bổ đề 4[2]. Cho trước bảng quyết định $DS = (U, C \cup \{d\}, V, f\})$ thì $\left(\mathrm{K}_d^r\right)^{-1}$ là hệ Sperner trên C. Ngược lại, nếu K là hệ Sperner trên C thì tồn tại một bảng quyết định nhất quán: $DS = (U, C \cup \{d\}, V, f\})$ để $K = \left(\mathrm{K}_d^r\right)^{-1}$

Bổ đề 5[2]. Cho trước $R = \{a_1, a_2, \ldots, a_n\}$ là tập thuộc tính không rỗng bất kỳ. Khi đó luôn tồn tại một hệ Sperner K trên R sao cho lực lượng của K là hàm mũ đối với n và lực lượng của K^{-1} là tuyến tính với n.

Định lí 1[2]. Bài toán cho trước bảng quyết định nhất quán $\mathrm{DS} = (\mathrm{U}, \mathrm{C} \cup \{\mathrm{d}\}, \mathrm{V}, \mathrm{f}\})$, việc tìm toàn bộ các rút gọn của *DS* có độ phức tạp tính toán là hàm mũ theo lực lượng của *A*.

2.2.5 Thuật toán tìm bảng quyết định không dư thừa

Cho trước bảng quyết định $\mathrm{DS} = (\mathrm{U}, \mathrm{C} \cup \{\mathrm{d}\}, \mathrm{V}, \mathrm{f}\})$. Ta gọi DS là bảng quyết định nhất quán không dư thừa nếu với mọi $\mathrm{u} \in \mathrm{U}$, ta đặt $\mathrm{U}' = \mathrm{U} \setminus \mathrm{u}$ và $\mathrm{DS}' = (\mathrm{U}', \mathrm{C} \cup \{\mathrm{d}\}, \mathrm{V}, \mathrm{f}\})$ ta có $\mathrm{PRED}_{\mathrm{U}}(\mathrm{C}) \neq \mathrm{PRED}_{\mathrm{U}'}(\mathrm{C})$. Ở đây $\mathrm{PRED}_{\mathrm{U}}(\mathrm{C})$ là kí pháp tập tất cả các rút gọn của DS. Có nghĩa là nếu ta bỏ đi một đối tượng (dòng) bất kỳ của DS thì tập tất cả các rút gọn sẽ thay đổi.

Chúng ta trình bày thuật toán sau: Tìm một bảng quyết định nhất quán không dư thừa từ một bảng quyết định nhất quán cho trước.

Đầu vào: Bảng quyết định nhất quán $DS = (U, C \cup \{d\}, V, f\})$, U = $\{u_1, \ldots, u_m\}$.

$PRED_U(C)$ là tập tất cả các rút gọn của DS.

Đầu ra: U' là bảng quyết định nhất quán không dư thừa.

Bước 1. Xét quan hệ r = $\{u_1, \ldots, u_m\}$ trên tập thuộc tính $R = C \cup \{d\}$.

Đặt $E_r = \{E_{ij} : 1 \le i < j \le m\}$ với $E_{ij} = \{a \in R : a(u_i) = a(u_j)\}$.

Đặt $M_U^d = \{A \in \varepsilon_r : d \notin A, \nexists B \in \varepsilon_r : d \notin B, A \subset B\}$.

M_U^d được gọi là tập bằng nhau cực đại trên U của d.

Bước 2. Đặt N(0) = U = $\{u_1, \ldots, u_m\}$

Bước 3. Tính:

$N(i + 1) = N(i) - u_{i+1}$ nếu $M^d_{N(i)-u_{i+1}} = M^d_U$. Ngược lại N(i + 1) = N(i)

Ở đây $M^d_{N(i)-u_{i+1}}$ là tập bằng nhau cực đại trên $N(i) - u_{i+1}$ của d.

Bước cuối. Đặt U' = N(m).

Có thể thấy rằng M_U^d được tính bằng thời gian đa thức. Vì thế, dễ thấy rằng thuật toán trên có độ phức tạp tính toán đa thức với số dòng và số cột của bảng quyết định DS. Mặt khác, chúng ta có thể chứng minh tính đúng đắn của thuật toán trên bằng phương pháp quy nạp. Có thể thấy, nếu hoán vị thứ tự các phần tử của U thì chúng ta sẽ nhận được một bảng quyết định nhất quán không dư thừa khác.

Ví dụ 7: Cho bảng quyết định $DS = (U, C \cup \{d\}, V, f\})$ với U = $\{u_1, u_2, u_3, u_4, u_5\}$ và C = {a, b, c, e} cụ thể như sau:

a	**b**	**c**	**e**	**d**
0	1	1	1	0
1	1	0	1	0
0	0	1	0	1
1	0	0	1	1
0	1	0	1	0

Bảng 2.6 Bảng dữ liệu đầu vào tìm bảng quyết định không dư thừa

Để đơn giản kí pháp, ta quy ước {a, b} được viết là ab. Như vậy, ta có E_{12} = bed, E_{13} = ae, E_{14} = e, E_{15} = abed, E_{23} = Φ, E_{24} = ace, E_{25} = bced, E_{34} = bd, E_{35} = a, E_{45} = ce,

E_r ={bed, ae, e, abed, Φ, ace, bced, bd, a, ce}.

Để dễ kí pháp đặt E_r = E(U) là tập các tập bằng nhau trên U.

Dễ thấy M_U^d={ace}. Đặt N(0) = {u_1, u_2, u_3, u_4, u_5}

Tính N(0) – u_1 = {u_2, u_3, u_4, u_5}, có thể thấy E(N(0) – u_1) = {Φ, ace, bced, bd, a, ce}

Và $M^d_{N(0)-u_1} = M^d_U$. Do đó N(1) = {u_2, u_3, u_4, u_5}. Tính N(1) – u_2 = {u_3, u_4, u_5}. Có thể thấy $M^d_{N(1)-u_2} \neq M^d_U$. Vì thế N(2) = N(1). Tính N(2) – u_3 = {u_2, u_4, u_5}, có thể thấy E(N(2) – u_3) = {ace, bced, ce} và $M^d_{N(2)-u_3} = M^d_U$. Do đó N(3) = {u_2, u_4, u_5}. Tính N(3) – u_4 = {u_2, u_5}. Có thể thấy $M^d_{N(3)-u_4} \neq M^d_U$. Vì thế N(4) = N(3). Cuối cùng, chúng ta tính N(4) – u_5 = {u_2, u_4}, có thể thấy E(N(4) – u_5)={ace} và $M^d_{N(4)-u_5} = M^d_U$. Do đó N(5) = N(4) – u_5 = {u_2, u_4}. Chúng ta đặt U' = N(5) = {u_2, u_4}. Vậy, chúng ta có bảng quyết định nhất quán DS' = (U' = {u_2, u_4}, Ơa, b, c, d}, V, f) là bảng quyết định nhất quán không dư thừa.

2.3 Kết luận chương

Chương 2 trình bày về một số tính chất và rút gọn thuộc tính trên bảng quyết định. Thuật toán tìm tập tất cả các thuộc tính rút gọn, họ tất cả các tập rút gọn, xây dựng các phụ thuộc hàm từ bảng quyết định nhất quán, thuật toán xây dựng bảng quyết định từ tập phụ thuộc hàm.

CHƯƠNG 3: THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH THỬ NGHIỆM

3.1 Đặt vấn đề

Như đã trình bày trong chương 2 từ một bảng quyết định nhất quán ban đầu, nếu có các thuộc tính dư thừa, ta rút gọn bằng cách loại bỏ các thuộc tính dư thừa đó đi. Việc loại bỏ các thuộc tính dư thừa này giúp bài toán trở nên đơn giản hơn mà vẫn không ảnh hưởng tới việc phân lớp đối tượng. Nội dung thuật toán được trình bày trong mục 2.2.2 chương 2.

3.2 Yêu cầu phần mềm nền tảng và cấu hình phần cứng máy PC

3.2.1 Yêu cầu phần mềm nền tảng

Chương trình "Tìm tập thuộc tính rút gọn" được viết bằng ngôn ngữ C#, sử dụng file txt chứa dữ liệu đầu vào theo định dạng định sẵn. Yêu cầu tối thiểu của hệ thống khi sử dụng chương trình:

- ✓ Cài đặt .Net Framework phiên bản 4.6.1 trở lên
- ✓ Cài đặt Visual Studio phiên bản 2017 trở lên

Phiên bản .Net Framework 4.6.1 hỗ trợ hệ điều hành Windows 7 SP1, Windows 8, Windows 8.1, Windows 10, Windows Server 2008 R2 SP1, Windows Server 2012, Windows Server 2012 R2.

3.2.2 Cấu hình phần cứng máy PC

Máy tính chạy tối thiểu hệ điều hành Windows 7 SP1 (with latest Windows Updates): Home Premium, Professional, Enterprise, Ultimate. Bộ xử lý 1.8 GHz Dual-core hoặc cao hơn. Ram 2GB hoặc cao hơn. Ổ cứng(HDD) 500GB. Video card tối thiểu720p (1280 by 720).

3.3 Giới thiệu chương trình và cách sử dụng

3.3.1 Cấu trúc chương trình

Program: Khởi tạo khung giao diện chương trình, có chức năng thiết lập các tham số cơ bản cho giao diện hiển thị.

Các hàm chính:

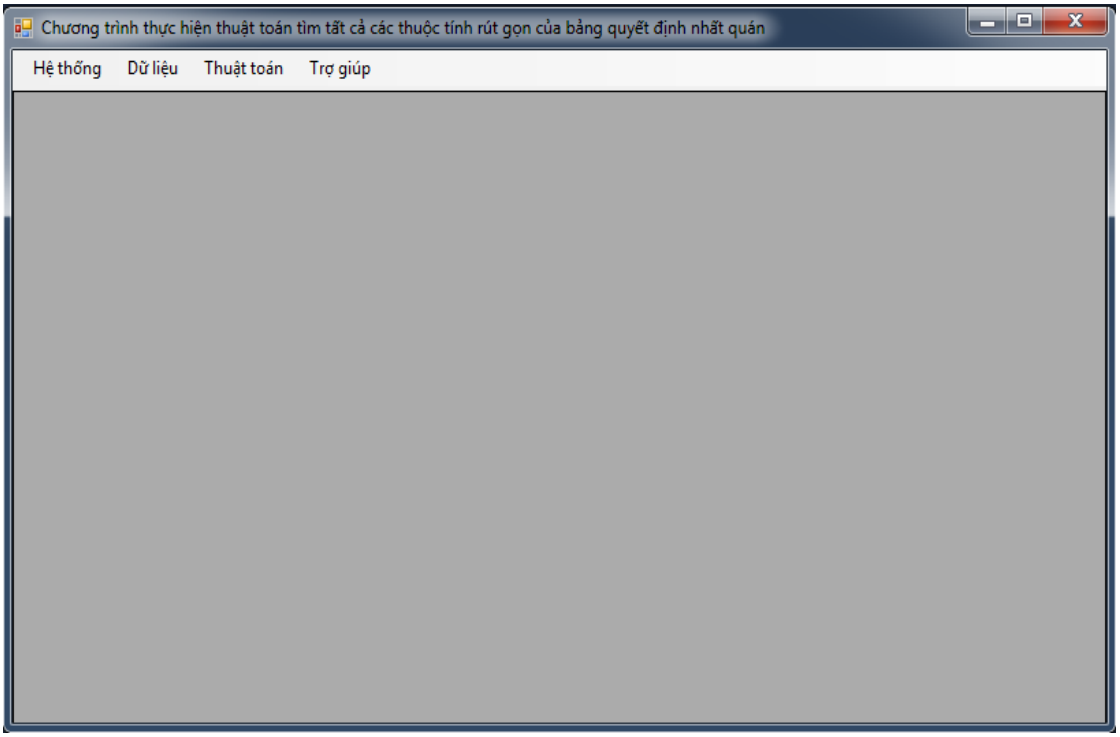
Các hàm	Diễn giải
`private void loadingToolStripMenuItem_Click(object sender, EventArgs e)`	Đọc dữ liệu từ file đầu vào, hiển thị dữ liệu ra giao diện chương trình
`private void findAllReductAttributesToolStripMenuItem_Click(object sender, EventArgs e)`	Tìm tất cả các thuộc tính rút gọn, quá trình thực thi hàm gọi tới hàm con makeConsistent() và calculateEqualitySet()
`private void private void checkConsistent()`	Kiểm tra bảng dữ liệu đầu là nhất quán hay không. Nếu không nhất quán đưa ra thông báo, nếu nhất quán thì hiển thị dữ liệu bảng nhất quán ra giao diện chương trình.
`private void calculateEqualitySet()`	Tìm các thuộc tính rút gọn và các thuộc tính dư thừa.

Bảng 3.1 Bảng mô tả các hàm chương trình tìm tất cả các tập rút gọn trên bảng quyết định nhất quán

3.3.2 Giới thiệu chương trình

Sao chép thư mục chương trình vào thư mục bất kỳ trên ổ cứng máy PC. Chạy file **FindAllReductAttribute.exe** để mở chương trình. Giao diện chính của chương trình như sau:

Giao diện của chương trình chính



Hình 3.1 Giao diện chương trình chính tìm tất cả các tập rút gọn trên bảng quyết định nhất quán

Chương trình có 3 phần chính:

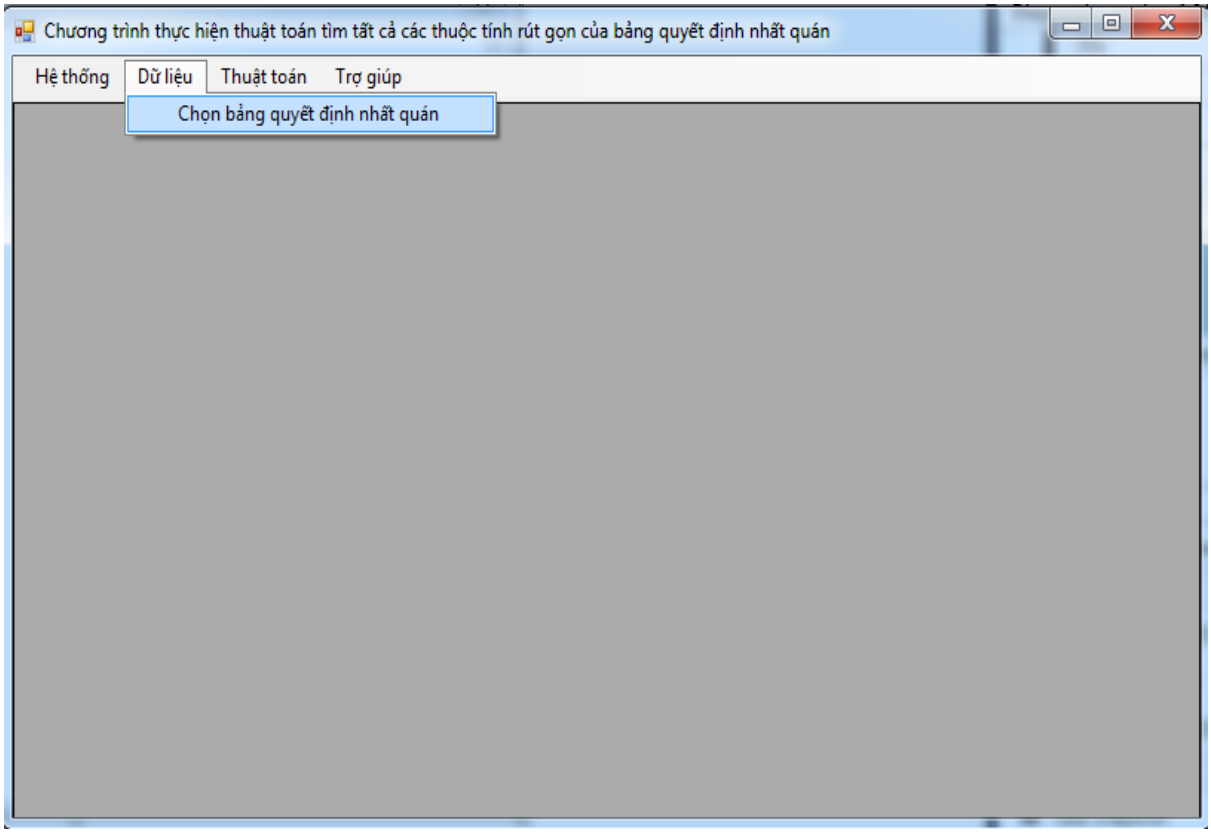
+ Phần 1: Các tab chức năng của chương trình *(Hệ thống / Dữ liệu / Thuật toán /Trợ giúp)*

+ Phần 2: Đầu vào chương trình *(Tab Dữ liệu)*

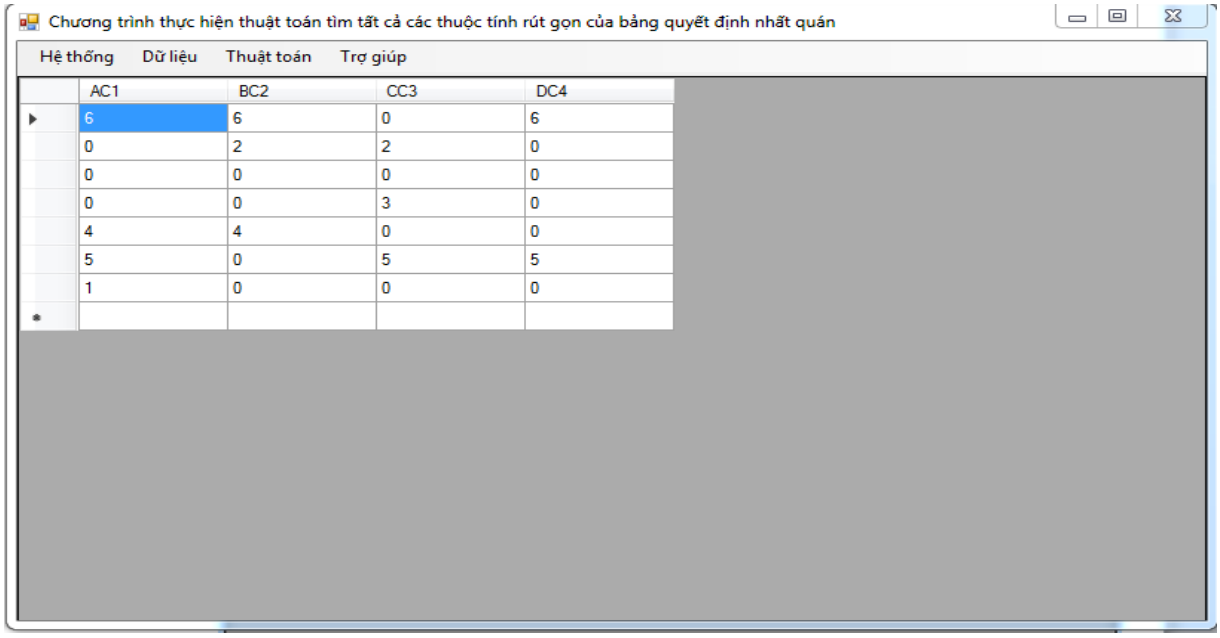
+ Phần 3: Thực hiện thuật toán *(Tab Thuật toán)*

Để thực hiện thuật toán, từ giao diện chương trình chính ta thực hiện theo các bước sau:

1. Chọn tab **"Dữ liệu"** từ giao diện chương trình chính để nhập dữ liệu đầu vào cho chương trình. Chương trình sẽ yêu cầu chọn file dữ liệu đầu vào để thực hiện cho bước tính toán tìm tập thuộc tính rút gọn ở bước sau.

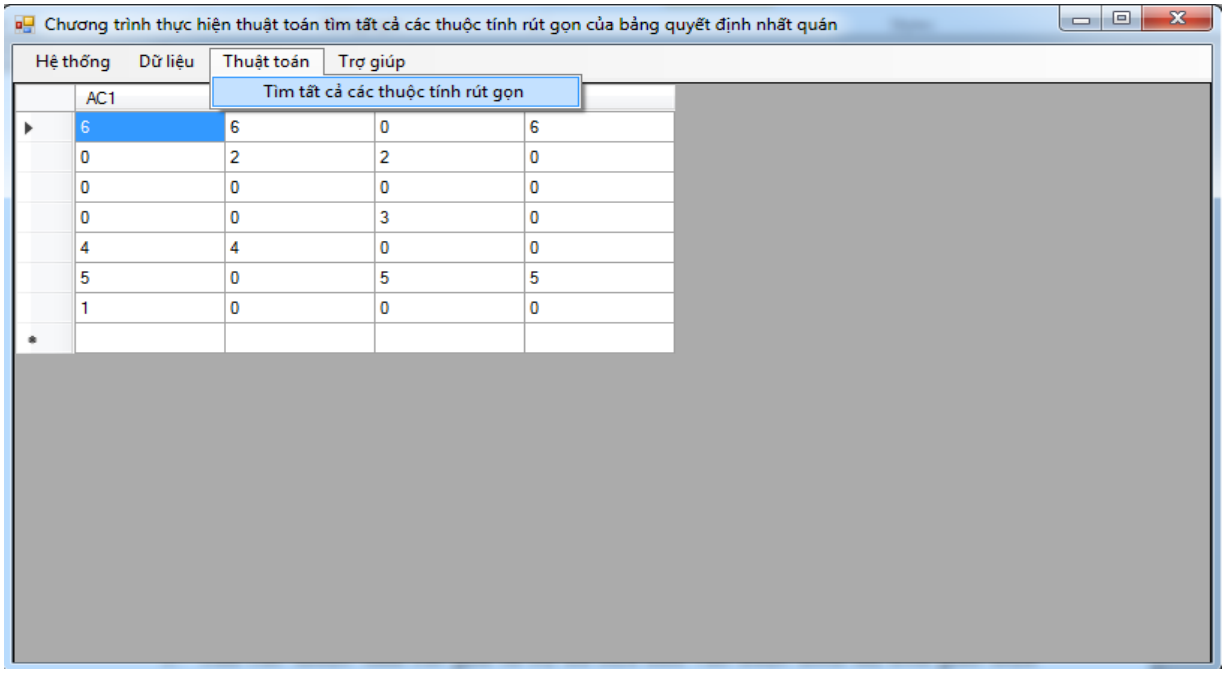


Hình 3.2 Chọn file dữ liệu đầu vào cho chương trình



Hình 3.3 Giao diện chương trình hiển thị dữ liệu đầu vào

2. Tìm các thuộc tính rút gọn từ bộ dữ liệu đầu vào được hiển thị trên giao diện chương trình. Chọn Tab "Thuật toán", sau đó chọn chức năng "Tìm tất cả các thuộc tính rút gọn" để thực hiện tìm các thuộc tính rút gọn.



Hình 3.4 Tìm tất cả các thuộc tính rút gọn

3.4 Thực hiện thuật toán với bộ dữ liệu Flu, EXAMPLE1, EXAMPLE

3.4.1 Bộ dữ liệu Flu

Cho bảng quyết định $DS=(U,C\cup\{c_3\},V,f)$

với $U=\{u_1,u_2,u_3,u_4,u_5,u_6,u_7,u_8\}$, $C=\{c_1,c_2\}$.

Trong đó: C_1 – đau đầu, C_2 – Thân nhiệt, C_3 – Cảm cúm

U	**C_1**	**C_2**	**C_3**
U_1	Yes	Normal	No
U_2	Yes	High	Yes
U_3	Yes	Very High	Yes
U_4	No	Normal	No
U_5	No	High	No
U_6	No	Very High	Yes
U_7	No	High	Yes
U_8	No	Very High	No

Bảng 3.2 Triệu chứng cúm của bệnh nhân

Dữ liệu đầu vào:

- Bảng quyết định $DS=(U,C\cup\{C3\},V,f)$,

- Các đối tượng U =$\{u_1,u_2,u_3,u_4,u_5,u_6,u_7,u_8\}$,

- Các thuộc tính C =$\{c_1,c_2,c_3,\}$.

Kết quả: Tập các thuộc tính rút gọn.

*** Thực hiện thuật toán:**

a) Kiểm tra xem bảng quyết định có nhất quán không

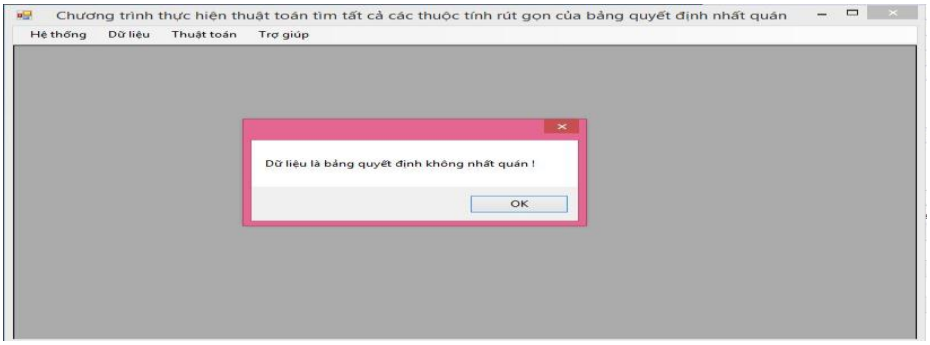
Xấp xỉ dưới của $\underline{B}X$ (Yes) = {u2,u3},

Xấp xỉ dưới của $\underline{B}X$ (No) = {u1,u4};

=>POSc(D)= $\bigcup_{X\subset U/D}(\underline{B}X)$ Không bằng U: Vậy bảng quyết định là không nhất quán.

b) Các bước thực hiện thuật toán

Vì bảng quyết định không nhất quán nên kết thúc thuật toán.



Hình 3.5 Kết quả của bộ dữ liệu Flu

3.4.2 Bộ dữ liệu "EXAMPLE1"

Xét bảng quyết định có các thuộc tính và các đối tượng như sau:

U	AC_1	BC_2	CC_3	DC_4
U_1	6	6	0	6
U_2	0	2	2	0
U_3	0	0	0	0
U_4	0	0	3	0
U_5	0	4	0	4
U_6	5	0	5	5
U_7	1	0	0	1

Bảng 3.3 Bảng quyết định bộ dữ liệu Example1

Dữ liệu đầu vào:

- Bảng quyết định $DS=(U,C\cup\{DC_4\},V,f)$,

- Các đối tượng $U=\{u_1,u_2,u_3,u_4,u_5,u_6,u_7\}$,

- Các thuộc tính $C=\{AC_1,BC_2,CC_3,DC_4\}$.

Kết quả: Tập các thuộc tính rút gọn

*** Thực hiện thuật toán:**

a) Kiểm tra xem bảng quyết định có nhất quán không

Xấp xỉ dưới của $\underline{B}X(0)$ = {U3, U4, U2},

Xấp xỉ dưới của $\underline{B}X(1)$= {U7},

Xấp xỉ dưới của $\underline{B}X(4)$ = {U5},

Xấp xỉ dưới của $\underline{B}X(5)$ = {U6},

Xấp xỉ dưới của $\underline{B}X(6)$ = {U1}.

=> $POSc(C_4)=\bigcup_{X\subset U/D}(\underline{B}X_i)=U$: Vậy bảng quyết định nhất quán.

b) Các bước thực hiện thuật toán

+ Bước 1: Hệ bằng nhau

ε_r = {{CC_3},{AC_1, DC_4},{ AC_1},{AC_1, BC_2, DC_4},{AC_1, CC_3},{ BC_2},{BC_2, CC_3}}.

+ Bước 2: Tập M_d = {{ AC_1, CC_3},{ BC_2, CC_3}}.

+ Bước 3: Tập $\bigcap_{K\in M_d} K$ = { CC_3},

$$\text{Tập } V=R-\bigcap_{K\in M_d}K=R-\bigcap_{K\in(\mathbf{K}_d^r)^{-1}}K=\{AC_1,BC_2,DC_4\}.$$

+ Bước 4: REAT(C) = V - {DC_4} = { AC_1, BC_2},

=> Hoàn thành thuật toán (thuộc tính rút gọn l { AC_1, BC_2}).

Hình 3.6 Kết quả khi thực hiện thuật toán với bộ dữ liệu Example1

3.4.3 Bộ dữ liệu "EXAMPLE"

Xét bảng quyết định có các thuộc tính và các đối tượng như sau:

U	*a*	*b*	*c*	*d*
u_1	6	6	0	6
u_2	0	2	2	0
u_3	0	0	0	0
u_4	0	0	3	0
u_5	4	4	0	0
u_6	5	0	5	5
u_7	1	0	0	0

Bảng 3.4 Bảng quyết định bộ dữ liệu Example

Dữ liệu đầu vào:

- Bảng quyết định $DS=(U,C\cup\{d\},V,f)$,

- Các đối tượng U =$\{u_1, u_2, u_3, u_4, u_5, u_6, u_7\}$,

- Các thuộc tính $C = \{a, b, c, d\}$.

Kết quả: Tập các thuộc tính rút gọn

*** Thực hiện thuật toán:**

a) Kiểm tra xem bảng quyết định có nhất quán không

Xấp xỉ dưới của $\underline{B}X$ (0) = {u3,u4,u2,u7,u5},

Xấp xỉ dưới của $\underline{B}X$ (5) = {u6},

Xấp xỉ dưới của $\underline{B}X$ (6) = {u1},

=> POSc(D) = $\bigcup_{X \subset U/D} (\underline{B}X) = U$: Vậy bảng quyết định là nhất quán.

b) Các bước thực hiện thuật toán

+ Bước 1: Hệ bằng nhau

ε_r = {{c},{a,d},{d},{a,b,d},{c,d},{b},{b,c,d},{b,d}}.

+ Bước 2: Tập M_d = {{c},{b}}.

+ Bước 3: Tập $\bigcap_{K \in M_d} K$ = {}.

$$\text{Tập } V = R - \bigcap_{K \in M_d} K = R - \bigcap_{K \in (\mathbf{K}_d^r)^{-1}} K = \{a,b,c,d\}$$

+ Bước 4: REAT(C) = V - {d} = {a,b,c}

=> Hoàn thành thuật toán (*thuộc tính rút gọn là {a,b, c}*).

Hình 3.7 Kết quả tìm các tập rút gọn với bộ dữ liệu Example

3.5 Kết luận chương

Chương này đã hướng dẫn cài đặt chương trình và các kết quả thử nghiệm

của chương trình tìm tập thuộc tính rút gọn. Đưa ra một số giao diện chính và cách sử dụng khi chạy chương trình…

KẾT LUẬN VÀ ĐỀ NGHỊ

1. Kết quả đạt được của luận văn

Khai phá dữ liệu là một trong những kỹ thuật quan trọng, mang tính thời sự không chỉ đối với Việt Nam mà của cả nền công nghệ thông tin toàn cầu hiện nay. Với sự bùng nổ thông tin dữ liệu toàn cầu, trong mọi mặt của đời sống xã hội cùng với sự phát triển và ứng dụng ngày càng rộng rãi của công nghệ thông tin trong mọi lĩnh vực đã khiến cho nhu cầu xử lý những khối dữ liệu khổng lồ để phát hiện ra những thông tin, tri thức hữu ích cho người sử dụng một cách tự động, nhanh chóng và chính xác. Một trong những phương pháp quan trọng của kỹ thuật khai phá dữ liệu mà đề tài đi tìm hiểu để làm cơ sở dữ liệu cho một số thuật toán rút gọn trên bảng quyết định nhất quán. Trong khoảng thời gian không dài đề tài đã tổng kết những kiến thức cơ bản nhất để phục vụ cho việc nghiên cứu một số thuật toán liên quan đến tập rút gọn trên bảng quyết định nhất quán. Có thể nói đề tài là một tài liệu tham khảo khá khá đầy đủ, rõ ràng về các kiến thức cơ bản trong khi nghiên cứu một số thuật toán liên quan đến tập rút gọn trên bảng quyết định nhất quán. Thông qua đó đã cài đặt thuật toán **"Tìm tập tất cả các thuộc tính rút gọn trên bảng quyết định nhất quán"** và chạy thử chương trình trên máy PC nhiều lượt với các bộ dữ liệu khác nhau.

2. Hướng nghiên cứu tiếp theo

Trên cơ sở những nghiên cứu đã được trình bày trong luận văn, tiếp tục nghiên cứu sâu hơn một số thuật toán liên quan tới tập rút gọn trên bảng quyết định nhất quán. Nhằm loại bỏ các thuộc tính dư thừa không cần thiết mà vẫn bảo toàn thông tin bài toán. Thông qua việc loại bỏ các thuộc tính dư thừa, các bài toán khai phá dữ liệu trở nên đơn giản hơn, phù hợp với giai đoạn hiện nay... Trong quá trình học tập, tìm hiểu và nghiên cứu cùng với khoảng thời gian làm luận văn, tôi đã cố gắng tập trung tìm hiểu và tham khảo các tài liệu liên quan. Tuy nhiên do thời gian và điều kiện nghiên cứu có hạn nên không tránh khỏi những thiếu sót, rất mong

nhận được sự nhận xét và những đóng góp ý kiến của quý thầy cô giáo và những ai quan tâm để luận văn được hoàn thiện hơn.

TÀI LIỆU THAM KHẢO

Tiếng Việt

[1] Vũ Đức Thi(1997). Cơ sở dữ liệu – Kiến thức và thực hành, Nhà xuất bản Thống kê, Hà Nôi.

[2] Vũ Đức Thi, Công nghệ tri thức, Nhà xuất bản khoa học tự nhiên và công nghệ, tái bản lần thứ nhất, tháng 10 năm 2018.

[3] Vũ Đức Thi (2018). Một vấn đề thuật toán liên quan đến tập rút gọn trong bảng quyết định nhất quán. Kỷ yếu về hội nghị quốc gia " Nghiên cứu cơ bản và ứng dụng CNTT" lần thứ XI, Hà Nội, 8/2018, tr 150 – 157.

[4] Vũ Đức Thi, Nguyễn Long Giang (2011). Một số phương pháp rút gọn thuộc tính trong bảng quyết định dựa trên ENTROPY cải tiến. Tạp chí Tin học và điều khiển. T 27, S 2 , tr. 166 – 175.

[5] Vũ Đức Thi, Nguyễn Long Giang. Thuật toán tìm tất cả các tập rút gọn trong bảng quyết định. Tạp chí Tin học và điều khiển T 27, S 3, tr. 211-218.

[6] Nguyễn Long Giang, Vũ Đức Thi (2011), "Some Problems Concering Condition Attributes and Reducts in Decision Tables", *Proceeding of the Fifsh National Symposium "Fundamental and Applied Information Technology Research"* (*FAIR*), Bien Hoa, Dong Nai, pp.142 – 152.

[7] Hoàng Thị Lan Giao (2007), "Khía cạnh đại số logic phát hiện luật theo tiếp cận tập thô*", Luận án tiến sĩ toán học*, Viện công nghệ thông tin.

Tiếng Anh

[8] Agrawal R., Imielinski T., Swami A.(1993). Mining association rules between sets of items in large database. Procceedings of the ACM SIGMOD conference, Washington DC, USA, pp.207-216.

[9] Demetrovics J., Thi V. D. Duong T.H.(2015). An Algorithm to mine normalized weighted sequential patterns using prefix-projeted database. SERDICA J. of computing. Bulgarian Academy of Sciences V.9.N 2.pp.111-118.

[10] Dinh V. V., Thi V. D., Giang N. L. (2014). Generallized Discernibility function Based Attribute Reduction Incomplete Decision Systems, SERDICA Journal of Computing, Bulgarial Academy of Sciences, V. 7,N_o4, pp.374-388.

[11] Giang N. L., Thi V. D.(2012). Some problems Concerning Condition Attributes and Reducts in Decision Tables, Proceeding of the fifth National Symposium "Fundamental and Applied Information Technology Research" (FAIR), Dong Nai, Viet Nam, pp.142-152.

[12] Pawlak Z. (1991), Rough sets: *Theoretical Aspects of Reasoning About Data*, Kluwer Aca-demic Publishers.

[13] Pawlak Z. (1998), "Rough set theory and its applications to data analysis", *Cysbernetics and systems 29*, pp. 661-688.

[14] DeWitt D., Gray J. (1992). *Parallel database systems: the future of high performance database systems*. Commun ACM 35(6):85-98.

[15] Walter T. (2009). *Teradata past, present, and future. UCI ISG lecture series on scalable data managemen*t.

[16] Ghemawat S., Gobioff H., Leung S-T. (2003). *The google file sys-tem. In: ACM SIGOPS Operating Systems Review*, vol 37. ACM, pp 29-43.

[17] Dean J., Ghemawat S. (2008). *Mapreduce: simplified data process-ing on large clusters*. Commun ACM 51(1):107-113.

[18] Hey AJG., Tansley S., Tolle KM. et al (2009). *The fourth paradigm: data-intensive scientific discovery*.

[19] Bahga A, Madisetti VK (2012). *Analyzing massive machine maintenance data in a computing cloud*. IEEE Transac Parallel Distrib Syst 23(10): 1831-1843.

PHỤ LỤC

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace FindAllReductAttribute
{
   public partial class MainForm : Form
   {
      private OpenFileDialog openFileDialog1;
      private DataTable dataTable1;
      private string[] head;
      private List<String> header1;
      private List<String> header2;
      private List<String> reductAttributes;
      private List<String> deleteAttributes;
      public MainForm()
      {
         InitializeComponent();
         dataTable1 = new DataTable();
         dataTable1.Rows.Clear();
      }
```

```
private void loadingToolStripMenuItem_Click(object sender, EventArgs e)
{
   openFileDialog1 = new OpenFileDialog()
   {
      FileName = "Select a text file",
      Filter = "Text files (*.txt)|*.txt",
      Title = "Open text file"
   };
   if (openFileDialog1.ShowDialog() == DialogResult.OK)
   {
      try
      {
         var filePath = openFileDialog1.FileName;
         string[] textData = System.IO.File.ReadAllLines(filePath);
         string[] headers = textData[0].Split(',');
         head = textData[0].Split(',');
         header1 = new List<String>();
         header2 = new List<String>();
         for(int i=0;i<headers.Length;i++)
         {
            header1.Add(headers[i].Substring(0,1));
            header2.Add(headers[i]);
         }

         //Create and populate DataTable
         dataTable1 = new DataTable();
         dataTable1.Rows.Clear();
         foreach (string header in headers)
            dataTable1.Columns.Add(header, typeof(string), null);
```

```
                for (int i = 1; i < textData.Length; i++)
                    dataTable1.Rows.Add(textData[i].Split(','));
                DataTable dt2 = new DataTable();
                dataGridView1.DataSource = dt2;
                dataGridView1.Refresh();
                dataGridView1.DataSource = dataTable1;
                checkConsistent();
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Security error.\n\nError message:
{ex.Message}\n\n" +
                $"Details:\n\n{ex.StackTrace}");
            }
        }
    }
    private void exitToolStripMenuItem_Click(object sender, EventArgs e)
    {
        this.Close();
    }
    private void aboutToolStripMenuItem_Click(object sender, EventArgs e)
    {
        About about = new About();
        about.ShowDialog();
    }
    private void findAllReductAttributesToolStripMenuItem_Click(object
sender, EventArgs e)
    {
        if (dataGridView1.Rows.Count > 0)
```

```
            {
                calculateEqualitySet();
                for(int i=0;i<reductAttributes.Count;i++)
                {
                    dataGridView1.Refresh();
            dataGridView1.Columns[reductAttributes[i]].DefaultCellStyle.ForeColor =
Color.Green;
                }
                for(int i=0;i<deleteAttributes.Count;i++)
                {

    dataGridView1.Columns[deleteAttributes[i]].DefaultCellStyle.ForeColor =
    Color.Red;
                }
            }
            else
            {
                MessageBox.Show("Chưa nhập dữ liệu đầu vào. Chọn 'Dữ liệu' từ
    Menu để nhập! ");
            }
        }

        private void checkConsistent()
        {
            List<int> lstRM = new List<int>();
            int iCount = dataGridView1.Rows.Count - 1;
            int iMax = dataGridView1.Rows[0].Cells.Count;
            bool bC = false;
            bool bD = false;
```

```
for (int i=0;i<iCount-1;i++)
{
   for(int j=i+1;j<iCount;j++)
   {
      bool bE = false;
      for(int k=0;k<iMax;k++)
      {

if(!dataGridView1.Rows[i].Cells[k].Value.ToString().Equals(dataGridView1.Ro
ws[j].Cells[k].Value.ToString()))
            {
               break;
            } else
            {
               if(k.Equals(iMax - 2) &&
(!dataGridView1.Rows[i].Cells[iMax-
1].Value.ToString().Equals(dataGridView1.Rows[j].Cells[iMax-
1].Value.ToString())) )
               {
                  bD = true;
               }
               if(k.Equals(iMax-1))
               {
                  bE = true;
               }
               continue;
            }
         }
         if(bE)
```

```
        {
            if(lstRM.IndexOf(j) < 0)
            {
                lstRM.Add(j);
            }
            bC = true;
        }
    }
}
DataTable dt = new DataTable();
foreach (DataGridViewColumn column in dataGridView1.Columns)
    dt.Columns.Add(column.Name);
for (int i=0;i<iCount-1;i++)
{
    if(lstRM.IndexOf(i) < 0)
    {
        dt.ImportRow(((DataTable)dataGridView1.DataSource).Rows[i]);
    }
}
DataTable dt00 = new DataTable();
dataGridView1.DataSource = dt00;
dataGridView1.Refresh();
if(!bD)
{
    dataGridView1.DataSource = dt;
    dataGridView1.Refresh();
} else
{
    MessageBox.Show("Dữ liệu là bảng quyết định không nhất quán !");
```

```
    }
}

private void calculateEqualitySet()
{
    DateTime dt = DateTime.Now;
    int iMax = dataGridView1.Rows[0].Cells.Count;
    List<List<String>> lstGetall = new List<List<String>>();
    List<String> lstAll = new List<String>();
    StringBuilder sb1 = new StringBuilder();
    int iCount = dataGridView1.Rows.Count -1;
    List<int> lstI = new List<int>();
    List<String> lstErr = new List<String>();
    //n,n+1
    for (int i=0;i<iCount-1;i++)
    {
        for(int i1=i+1;i1<iCount;i1++)
        {
            StringBuilder sb = new StringBuilder();
            lstErr = new List<String>();
            for (int j=0;j<iMax;j++)
            {
                if
(dataGridView1.Rows[i].Cells[j].Value.ToString().Equals(dataGridView1.Rows
[i1].Cells[j].Value.ToString()))
                {
                    lstErr.Add(header1[j]);
                }
            }
```

```
                bool bl = false;
                if(!bl)
                {
                    if(lstGetall.IndexOf(lstErr) < 0)
                    {
                        lstGetall.Add(lstErr);
                    }
                    string xv = string.Join("-", lstErr);
                    if( (lstAll.IndexOf(xv) < 0) && (xv.Trim().Length > 0) )
                    {
                        if(xv.ToUpper().IndexOf(header1[header1.Count-
1].ToUpper()) < 0)
                        {
                            lstAll.Add(xv);
                        }
                    }
                }
            }
        }
        string[] arrALL = lstAll.ToArray();
        Array.Sort(arrALL, (y, x) => x.Length.CompareTo(y.Length));
        List<String> lstMD = new List<String>();
        for(int i=0;i<arrALL.Length;i++)
        {
            string[] arrItem = arrALL[i].Split('-');
            bool blA = false;
            for(int j=0;j<lstMD.Count;j++)
            {
                for(int k=0;k<arrItem.Length;k++)
```

```
        {
            if(lstMD[j].IndexOf(arrItem[k]) >=0 )
            {
                if(k.Equals(arrItem.Length-1))
                {
                    blA = true;
                }
                continue;
            } else
            {
                break;
            }
        }
        if(blA)
        {
            break;
        }
    }
    if(!blA)
    {
        lstMD.Add(arrALL[i]);
        sb1.Append("\n" + arrALL[i]);
    }
}
List<int> lstFA = new List<int>();
deleteAttributes = new List<String>();
if(lstMD.Count > 0)
{
    for(int j=0;j<iMax-1;j++)
```

```
    {
        bool blC = false;
        for (int i = 0; i < lstMD.Count; i++)
        {
            if (lstMD[i].IndexOf(header1[j].Substring(0, 1)) >= 0)
            {
                if(i.Equals(lstMD.Count-1)) {
                    blC = true;
                }
                continue;
            } else
            {
                break;
            }
        }
        if(blC)
        {
            lstFA.Add(j);
            deleteAttributes.Add(header2[j]);
        }
    }
}
string xyz = string.Join("-", lstFA);
List<String> lstRA = new List<String>();
reductAttributes = new List<String>();
for(int i=0;i<iMax-1;i++)
{
    if(!xyz.Contains(i.ToString())) {
        lstRA.Add(header1[i]);
```

```
                reductAttributes.Add(header2[i]);
            }
        }
        DateTime dt1 = DateTime.Now;
        List<String> lstRS = new List<String>();
        lstRS.Add("KẾT QUẢ TÌM TẤT CẢ CÁC THUỘC TÍNH RÚT GỌN:");
        lstRS.Add("---------------------------------");
        lstRS.Add("Số lượng thuộc tính điều kiện: " + (header1.Count-1).ToString());
        for(int i=0;i<iMax-1;i++)
        {
            lstRS.Add(header2[i] + " (" + header1[i] + ")");
        }
        lstRS.Add("---------------------------------");
        lstRS.Add("Số lượng thuộc tính quyết định: 1");
        lstRS.Add(header2[header2.Count - 1] + " (" + header1[header1.Count-1] + ")");
        lstRS.Add("---------------------------------");
        lstRS.Add("Số lượng đối tượng: " + (dataGridView1.Rows.Count-1).ToString());
        lstRS.Add("---------------------------------");
        lstRS.Add("Tập Md: ");
        lstRS.Add("{");
        for (int i=0;i<lstMD.Count;i++)
        {
            if(i<(lstMD.Count-1))
            {
                lstRS.Add("    {" + lstMD[i].Replace("-", ",") + "},");
```

```
            } else
            {
               lstRS.Add("    {" + lstMD[i].Replace("-", ",") + "}");
            }
         }
         lstRS.Add("}");
         lstRS.Add("---------------------------------");
         lstRS.Add("Thuộc tính rút gọn: " + (reductAttributes.Count).ToString());
         for (int i=0;i<reductAttributes.Count;i++)
         {
            lstRS.Add(reductAttributes[i] + " (" +
reductAttributes[i].Substring(0,1) + ")" );
         }
         lstRS.Add("---------------------------------");
         lstRS.Add("Thuộc tính dư thừa: " + (deleteAttributes.Count).ToString());
         for(int i=0;i<deleteAttributes.Count;i++)
         {
            lstRS.Add(deleteAttributes[i] + " (" +
deleteAttributes[i].Substring(0,1) + ")");
         }
         lstRS.Add("---------------------------------");
         lstRS.Add("Thời gian thực hiện thuật toán: " + (dt1 -
dt).TotalSeconds.ToString() + "giây");
         lstRS.Add("---------------------------------");
         FormResult fr = new FormResult(lstRS.ToArray());
         fr.ShowDialog();
      }
}
}
```