Tạp chí Khoa học và Công nghệ ***50*** *(6) (2012) 679-703*

MỘT SỐ VẤN ĐỀ TÍNH TOÁN LIÊN QUAN ĐẾN CƠ SỞ DỮ LIỆU VÀ KHAI PHÁ DỮ LIỆU

Vũ Đức Thi

Viện Công nghệ thông tin, Viện KHCNVN, 18 Hoàng Quốc Việt, Cầu Giấy, Hà Nội

Email: *vdthi@ioit.ac.vn*

Đến Tòa soạn: 17/12/2012; Chấp nhận đăng: 23/12/2012

TÓM TẮT

Cơ sở dữ liệu và khai phá dữ liệu là những hướng phát triển rất quan trọng trong lĩnh vực công nghệ thông tin (CNTT). Về thực chất dữ liệu đóng vai trò nền tảng nhất trong quá trình xử lí thông tin trên hệ thống máy tính. Lí thuyết cơ sở dữ liệu và việc ứng dụng lí thuyết này vào thực tiễn đã được phát triển và đạt được nhiều thành tựu ngay từ những năm 80 thế kỉ trước. Về bản chất lí thuyết cơ sở dữ liệu cung cấp cho chúng ta những tri thức quan trọng nhất liên quan đến vấn đề tổ chức, thiết kế và xây dựng các hệ thống quản trị cơ sở dữ liệu. Trên nền tảng những kết quả đạt được trong lí thuyết này, các hãng máy tính của thế giới như IBM, Microsoft, Oracle, Apple … đã xây dựng những hệ thống quản trị cơ sở dữ liệu thương mại bán khắp nơi trên thị trường toàn cầu như SQL, Oracle, IBM DB2. Về một khía cạnh nào đó, hiện nay, trong mọi hoạt động nhân loại đã tích lũy một khối lượng khổng lồ dữ liệu. Tuy vậy, tri thức thì lại quá nhỏ bé. Chính vì thế, hiện nay, hướng nghiên cứu về phát hiện tri thức từ dữ liệu (Knowledge Discovery from Data) là một hướng phát triển rất mạnh mẽ. Một khâu đặc biệt then chốt trong quá trình phát hiện tri thức từ dữ liệu này là khai phá dữ liệu (Data Mining) để thu nhận tri thức. Do đó, hướng nghiên cứu về các phương pháp khai phá dữ liệu là một hướng rất cơ bản trong lĩnh vực CNTT. Trong bài báo này, chúng tôi trình bày một số kết quả nền tảng về vấn đề tính toán, thực chất là vấn đề thuật toán, trong lĩnh vực cơ sở dữ liệu và khai phá dữ liệu.

Từ khóa: cơ sở dữ liệu, khai phá dữ liệu, hệ thống quản trị cơ sở dữ liệu, phát hiện tri thức từ dữ liệu, vấn đề tính toán, thuật toán

1. MỞ ĐẦU

Cơ sở dữ liệu (CSDL) là một trong những lĩnh vực được tập trung nghiên cứu và phát triển của công nghệ thông tin, nhằm giải quyết các bài toán quản lí, tìm kiếm thông tin trong những hệ thống lớn, đa dạng, phức tạp cho nhiều người sử dụng trên máy tính điện tử. Cùng với sự ứng dụng mạnh mẽ công nghệ thông tin vào đời sống xã hội, kinh tế, quốc phòng ...Việc nghiên cứu CSDL đã và đang phát triển ngày càng phong phú và hoàn thiện. Từ những năm 70, mô hình dữ liệu quan hệ do E.F. Codd đưa ra với cấu trúc hoàn chỉnh đã tạo lên cơ sở nền tảng cho các vấn đề nghiên cứu lí thuyết về CSDL. Với ưu điểm về tính cấu trúc đơn giản và khả năng hình thức hoá phong phú, CSDL quan hệ dễ dàng mô phỏng các hệ thống thông tin đa dạng trong thực tiễn, tạo điều kiện lưu trữ thông tin tiết kiệm, có tính độc lập dữ liệu cao, dễ sửa đổi, bổ sung

cũng như khai thác dữ liệu. Mặt khác, việc khai thác và áp dụng các kĩ thuật tổ chức và sử dụng bộ nhớ cho phép việc cài đặt các CSDL quan hệ đưa lại hiệu quả cao và làm cho CSDL quan hệ chiếm ưu thế hoàn toàn trên thị trường.

Nhiều hệ quản trị CSDL dựa trên mô hình dữ liệu quan hệ đã được xây dựng và đưa vào sử dụng rộng rãi như: DBASE, FOXBASE, FOXPRO, PARADOX, ORACLE, MEGA, IBM DB2, SQL.

Mô hình dữ liệu quan hệ đặt trọng điểm hàng đầu không phải là khai thác các tiềm năng của máy mà ở sự mô tả trực quan dữ liệu theo quan điểm của người dùng, cung cấp một mô hình dữ liệu đơn giản, trong sáng, chặt chẽ, dễ hiểu và tạo khả năng tự động hoá thiết kế CSDL quan hệ. Có thể nói lí thuyết thiết kế và cài đặt CSDL, nhất là mô hình dữ liệu quan hệ đã phát triển ở mức độ cao và đạt được những kết quả sâu sắc. Hàng loạt vấn đề đã được nghiên cứu giải quyết như:

- Lí thuyết thiết kế CSDL, các phương pháp tách và tổng hợp các sơ đồ quan hệ theo tiêu chuẩn không tổn thất thông tin hay bảo toàn tính nhất thể của các ràng buộc trên dữ liệu .

- Các loại ràng buộc dữ liệu, cấu trúc và các tính chất của chúng, ngữ nghĩa và khả năng áp dụng phụ thuộc dữ liệu ví dụ như phụ thuộc hàm, phụ thuộc đa trị, phụ thuộc kết nối, phụ thuộc lôgic...

- Các vấn đề tối ưu hoá: ở mức vật lí trong việc tổ chức quản lí các tệp; ở mức đường truy nhập với các tệp chỉ số hay các danh sách sắp xếp; ở mức lôgic trên cơ sở rút gọn các biểu thức biểu diễn các câu hỏi, ...vv.

Trong bài báo này chúng tôi trình bày một số vấn đề thuật toán phục vụ việc thiết kế tổng thể các hệ thống CSDL hiện nay.

Sự phát triển nhanh chóng các ứng dụng công nghệ thông tin và Internet vào nhiều lĩnh vực đời sống xã hội, quản lí kinh tế, khoa học kĩ thuật, đã tạo ra nhiều cơ sở dữ liệu khổng lồ. Để khai thác hiệu quả nguồn thông tin từ các cơ sở dữ liệu lớn, hỗ trợ tiến trình ra quyết định, bên cạnh các phương pháp khai thác thông tin truyền thống, các nhà nghiên cứu đã phát triển các phương pháp tìm kiếm các tri thức.

Theo đánh giá của IBM, các phương pháp khai thác thông tin truyền thống chỉ thu được khoảng 80 % thông tin từ cơ sở dữ liệu, phần còn lại bao gồm các thông tin mang tính khái quát, thông tin có tính quy luật vẫn còn đang tiềm ẩn trong dữ liệu. Lượng thông tin này tuy nhỏ nhưng là những thông tin cốt lõi và cần thiết cho tiến trình ra quyết định.

Khai phá dữ liệu (KPDL) là một lĩnh vực quan trọng của ngành CNTT. Đây là một trong những lĩnh vực phát triển rất sôi động của CNTT.Trên thực tế, hiện có nhiều phương pháp KPDL như phân cụm dữ liệu, cây quyết định, thống kê, mạng nơron, phân lớp dữ liệu, phương pháp sinh luật kết hợp, phương pháp sử dụng lí thuyết tập thô,... Trong bài báo này chúng tôi trình bày một số vấn đề tính tóan liên quan đến hai phương pháp rất nền tảng của KPDL là phương pháp sinh luật kết hợp và phương pháp sử dụng lí thuyết tập thô.

Cho đến nay có rất nhiều tác giả đã nghiên cứu và phát triển phương pháp sinh luật kết hợp. Kể từ khi Agrawal [1] đề xuất lần đầu vào năm 1993 đến nay, khai phá tập mục thường xuyên đã có hàng trăm kết quả nghiên cứu được công bố. Trong quá trình sinh luật kết hợp, khai phá tập mục thường xuyên đóng vai trò then chốt nhất. Khai phá tập mục thường xuyên đã có nhiều cách thức mở rộng và ứng dụng, từ thay đổi phương pháp luận đến thay đổi đa dạng các kiểu dữ liệu, mở rộng các nhiệm vụ khai phá và đa dạng các ứng dụng mới. Năm 2003, Tao và các đồng sự đề xuất việc sinh luật kết hợp có trọng số [2]. Trên cơ sở thuật tóan Apriori họ đã đưa ra một thuật tóan tìm tập mục thường xuyên có trọng số. Năm 2008, Khan và các đồng sự đã mở rộng

phương pháp này để sinh luật kết hợp [3]. Một số tác giả đã nghiên cứu trên các cơ sở dữ liệu giao tác gia tăng [10,23], thực chất là tập các mục và tập các giao tác đều cho phép thay đổi. Một hướng nghiên cứu khác là ứng dụng lí thuyết tập mờ trong việc sinh luật kết kết hợp cũng được nhiều tác giả quan tâm [9,21].

Mô hình khai phá tập mục thường xuyên cơ bản có nhiều ứng dụng trong thực tế nhưng có những hạn chế, không đáp ứng đầy đủ yêu cầu của người sử dụng. Để đáp ứng yêu cầu của thực tiễn, một số hướng mở rộng bài toán đã được quan tâm nghiên cứu. Một hướng mở rộng bài toán có rất nhiều ứng dụng là quan tâm đến cấu trúc dữ liệu và mức độ quan trọng khác nhau của các mục dữ liệu, các thuộc tính trong cơ sở dữ liệu. Theo hướng này, từ bài toán khai phá tập mục thường xuyên ban đầu, nhiều nhà nghiên cứu đề xuất các mô hình mở rộng: khai phá tập mục cổ phần cao, đánh giá sự đóng góp của tập mục dữ liệu trong tổng số các mục dữ liệu của cơ sở dữ liệu; khai phá tập mục lợi ích cao, đánh giá lợi ích mà tập mục dữ liệu mang lại trong cơ sở dữ liệu [34, 35].

Trên thế giới, các kết quả nghiên cứu về khai phá tập mục cổ phần cao, khai phá tập mục lợi ích cao đã được công bố nhiều từ các nhóm nghiên cứu tại một số trường đại học ở Mỹ, Canada, Úc, Đài Loan, Singapore [19, 35]. Đã có các hội thảo quốc tế riêng về khai phá dữ liệu dựa trên lợi ích (Workshop on Utility-Based Data Mining): hội thảo lần thứ nhất tổ chức tại Chicago, Illinois, Mỹ vào tháng 8 năm 2005, lần thứ hai tổ chức cùng với hội thảo về khám phá tri thức tại Mỹ vào tháng 8 năm 2006 [25, 35]. Khai phá tập mục lợi ích cao là sự khái quát của khai phá cổ phần cao và thực sự là một lĩnh vực đang thu hút nhiều nhà nghiên cứu tham gia.

Lí thuyết tập thô do Z. Pawlak [27] đề xuất vào những năm đầu thập niên tám mươi của thế kỉ hai mươi - 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, lí 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 lí số liệu, trích lọc các tri thức tiềm ẩn trong dữ liệu và đánh giá kết quả thu được. Việc sử dụng lí thuyết tập thô vào khai phá dữ liệu thu hút nhiều nhà khoa học. Một trong những nhánh quan trọng của hướng nghiên cứu này là nghiên cứu việc rút gọn thuộc tính trên bảng quyết định. Mục tiêu của rút gọn thuộc tính trong bảng quyết định là tìm tập thuộc tính rút gọn (gọi tắt là *tập 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 có thể là hàm số mũ theo số thuộc tính điều kiện. Tuy nhiên, trong thực hành 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 đưa ra định nghĩa tập rút gọn và xây dựng thuật toán heuristic tìm một 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 đáng chú ý là: phương pháp sử dụng miền dương [4, 27], phương pháp sử dụng entropy Shannon [36], phương pháp sử dụng entropy Liang [23, 26].

2. MỘT SỐ KHÁI NIỆM CƠ BẢN

2.1. Một số khái niệm về cơ sở dữ liệu

Một cơ sở dữ liệu là một hệ thống các file dữ liệu, mỗi file này có cấu trúc bản ghi khác nhau, nhưng về mặt nội dung có quan hệ với nhau. Một hệ quản trị cơ sở dữ liệu là một hệ thống quản lí và điều hành các file dữ liệu. Trên thực tế có nhiều mô hình dữ liệu. Song mô hình dữ liệu quan hệ do E.F. Codd đề xuất đã phát triển mạnh mẽ nhất kể cả về mặt lí thuyết lẫn ứng dụng trong thực tiễn.

Mô hình dữ liệu quan hệ là một công cụ rất tiện lợi để mô tả cấu trúc lôgic của các cơ sở dữ liệu. Như vậy, ở mức lôgic mô hình này bao gồm các file được biểu diễn dưới dạng các bảng. Do đó đơn vị của CSDL quan hệ là một bảng, trong đó các dòng của bảng là các bản ghi dữ liệu cụ thể, còn tên các cột là các thuộc tính.

Theo cách nhìn của người sử dụng thì một cơ sở dữ liệu quan hệ là một tập hợp các bảng biến đổi theo thời gian.

Trong mục này, chúng ta trình bày những khái niệm cơ bản về mô hình dữ liệu quan hệ. Những khái niệm này có thể tìm thấy trong [8,15,16,17,20].

Định nghĩa 1. (Quan hệ, bảng)

Cho $R = \{a_{1, \dots,} a_n\}$ là một tập hữu hạn và không rỗng các thuộc tính. Mỗi thuộc tính a_i có miền giá trị là D_{ai}. Khi đó r là một tập các bộ $\{h_1, ..., h_m\}$ được gọi là một quan hệ trên R với h_j (j = 1,...m) là một hàm:

$h_j: R \rightarrow \cup D_{ai}$

$a_i \in R$

sao cho: $h_j (a_j) \in D_{ai}$

Chúng ta có thể biểu diễn quan hệ r thành bảng sau:

	a_1	a_2		a_n
h_1	$h_1(a_1)$	$h_1(a_2)$	.....	$h_1(a_n)$
h_2	$h_2(a_1)$	$h_2(a_2)$	.....	$h_2(a_n)$
.	.....			
h_m	$h_m(a_1)$	$h_m(a_2)$	.....	$h_m(a_n)$

Định nghĩa 2. (Phụ thuộc hàm)

Cho $R = \{a_1,...,a_n\}$ là tập các thuộc tính, $r = \{h_1,...,h_m\}$ là một quan hệ trên R, và $A, B \subseteq R$. Khi đó chúng ta nói A xác định hàm cho B hay B phụ thuộc hàm vào A trong r (Kí pháp $A \xrightarrow{f}_{r} B$) nếu

$(\forall h_i,h_j \in r)((\forall a \in A)(h_i(a)= h_j(a)) \Rightarrow (\forall b \in B) (h_i(b)=h_j(b)))$

Đặt $F_r = \{ (A,B): A,B \subseteq R, A \xrightarrow{f}_{r} B \}$. Lúc đó F_r được gọi là họ đầy đủ các phụ thuộc hàm của r.

Khái niệm phụ thuộc hàm miêu tả một loại ràng buộc (phụ thuộc dữ liệu) xảy ra tự nhiên nhất giữa các tập thuộc tính. Dù hiện nay đã có nhiều loại phụ thuộc dữ liệu được nghiên cứu, song về cơ bản các hệ quản trị cơ sở dữ liệu lớn sử dụng phụ thuộc hàm.

Định nghĩa 3.

Phụ thuộc hàm (PTH) trên tập các thuộc tính R là một dãy kí tự có dạng $A \rightarrow B$, ở đây $A,B \subseteq R$. Chúng ta nói PTH $A \rightarrow B$ đúng trong quan hệ r if $A \xrightarrow{f}_{r} B$.

Định nghĩa 4. (Hệ tiên đề của Armstrong)

Giả sử R là tập các thuộc tính và kí pháp P(R) là tập các tập con của R. Cho $Y \subseteq P(R) \times P(R)$. Chúng ta nói Y là một họ f trên R nếu đối với mọi $A, B, C, D \subseteq R$

(1) $(A,A) \in Y$,

(2) $(A,B) \in Y, (B,C) \in Y \Rightarrow (A,C) \in Y$,

(3) $(A,B) \in Y, A \subseteq C, D \subseteq B \rightarrow (C,D) \in Y$,

(4) $(A,B) \in Y, (C,D) \in Y \Rightarrow (A \cup C, B \cup D) \in Y$.

Rõ ràng, F_r là một họ f trên R.

Trong [7] A. A. Armstrong đã chứng minh một kết quả rất quan trọng như sau: Nếu Y là một họ f bất kì thì tồn tại một quan hệ r trên R sao cho $F_r = Y$.

Kết quả này cùng với định nghĩa của phụ thuộc hàm chứng tỏ rằng hệ tiên đề Armstrong là đúng đắn và đầy đủ.

Mặt khác, hệ tiên đề này cho ta những đặc trưng của họ các phụ thuộc hàm, mà các đặc trưng này không phụ thuộc vào các quan hệ (bảng) cụ thể. Nhờ có hệ tiên đề này các công cụ của toán học được áp dụng để nghiên cứu làm sáng tỏ cấu trúc lôgic của mô hình dữ liệu quan hệ. Đặc biệt chúng ta sử dụng công cụ thuật toán để thiết kế các công đoạn xây dựng các hệ quản trị cơ sở dữ liệu.

Định nghĩa 5. (Sơ đồ quan hệ)

Chúng ta gọi sơ đồ quan hệ (SĐQH) s là một cặp <R,F>, ở đây R là tập các thuộc tính và F là tập các phụ thuộc hàm trên R. Kí pháp F^+ là tập tất cả các PTH được dẫn xuất từ F bằng việc áp dụng các qui tắc trong Định nghĩa 4.

Đặt $A^+ = \{a: A \rightarrow \{a\} \in F^+\}$. A^+ được gọi là bao đóng của A trên s.

Có thể thấy rằng $A \rightarrow B \in F^+$ nếu và chỉ nếu $B \subseteq A^+$.

Tương tự chúng ta đặt $A_r^+ = \{a: A \xrightarrow[r]{f} \{a\}\}$. A_r^+ được gọi là bao đóng của A trên r.

Theo [7] chúng ta có thể thấy nếu s = <R,F> là sơ đồ quan hệ thì có quan hệ r trên R sao cho $F_r = F^+$. Quan hệ r như vậy chúng ta gọi là quan hệ Armstrong của s.

Trong trường hợp này hiển nhiên các PTH của s đúng trong r.

Định nghĩa 6. (Khoá)

Giả sử r là một quan hệ , s = <R,F> là một sơ đồ quan hệ, và $A \subseteq R$. Khi đó A là một khoá của r (tương ứng là một khoá của s, một khoá của Y) nếu $A \xrightarrow[r]{f} R$ $(A \rightarrow R \in F^+)$. Chúng ta gọi A là một khoá tối tiểu của r (tương ứng của s) nếu

- A là một khoá của r (s),

- Bất kì một tập con thực sự của A không là khoá của r (s).

Chúng ta kí pháp K_r, (K_s) tương ứng là tập tất cả các khoá tối tiểu của r (s).

Chúng ta gọi K (ở đây K là một tập con của P(R)) là một hệ Sperner trên R nếu với mọi $A,B \in K$ kéo theo $A \subseteq B$).

Có thể thấy K_r, K_s là các hệ Sperner trên R.

Định nghĩa 7.

Giả sử K là một hệ Sperner trên R. Chúng ta định nghĩa tập các phản khoá của K, kí pháp là K^{-1}, như sau:

$K^{-1} = \{A \subset R: (B \in K) \Rightarrow (B \subseteq A) \text{ and } (A \subset C) \Rightarrow (\exists B \in K)(B \subseteq C)\}$

Dễ thấy K^{-1} cũng là một hệ Sperner trên R.

Tập phản khoá đóng vai trò rất quan trọng trong quá trình nghiên cứu cấu trúc lôgic của các họ phụ thuộc hàm, khóa, dạng chuẩn, quan hệ Armstrong, đặc biệt đối với các bài toán tổ hợp trong mô hình dữ liệu quan hệ.

Trong [14] người ta đã nêu ra rằng nếu s = <R,F> là một sơ đồ quan hệ trên R, thì K_s là hệ Sperner trên R. Ngược lại, nếu K là một hệ Sperner bất kì trên R, thì tồn tại một sơ đồ quan hệ s sao cho $K_s = K$.

Định nghĩa 8.

Cho r là một quan hệ trên R. Chúng ta đặt $E_r = \{E_{ij}: 1 \le i \le j \le |r|\}$, ở đây $E_{ij} = \{a \in R: h_i(a) = h_j(a)\}$. E_r được gọi là hệ bằng nhau của r.

Đặt $M_r = \{ A \in P(R): \exists E_{ij} = A, \exists E_{pq}: A \subset E_{pq}\}$. Khi đó chúng ta gọi M_r là hệ bằng nhau cực đại của r.

Sau này ta sẽ thấy hệ bằng nhau và hệ bằng nhau cực đại được dùng rất nhiều trong các thuật toán thiết kế.

Mối quan hệ giữa lớp các quan hệ và lớp các phụ thuộc hàm đóng một vai trò quan trọng trong quá trình nghiên cứu cấu trúc lôgic của lớp các phụ thuộc hàm.

Định nghĩa 9.

Cho trước r là một quan hệ r và F là một họ f trên R. Chúng ta nói rằng r là thể hiện họ F nếu $F_r = F$. Chúng ta cũng có thể nói r là một quan hệ Armstrong của F.

2.2. Một số khái niệm liên quan đến khai phá dữ liệu

2.2.1. Một số khái niệm liên quan đến sinh luật kết hợp

Khai phá tập mục thường xuyên là bài toán có vai trò quan trọng trong nhiều nhiệm vụ khai phá dữ liệu. Khai phá tập mục thường xuyên được biết đến ban đầu là một trong những bài tóan quan trọng của khai phá luật kết hợp được giới thiệu bởi Agrawal vào năm 1993 khi phân tích cơ sở dữ liệu bán hàng của siêu thị [8], phân tích sở thích mua của khách hàng bằng cách tìm ra những mặt hàng khác nhau được khách hàng mua cùng trong một lần mua. Những thông tin như vậy sẽ giúp người quản lí kinh doanh tiếp thị chọn lọc và thu xếp không gian bày hàng hợp lí hơn, giúp cho kinh doanh hiệu quả hơn.

Khai phá luật kết hợp là phát hiện những mối quan hệ giữa các giá trị dữ liệu trong cơ sở dữ liệu, các mối quan hệ đó chính là các luật kết hợp.

Việc sinh luật kết hợp có hai bước: bước thứ nhất, tìm các tập mục thường xuyên thỏa mãn ngưỡng độ hỗ trợ tối thiểu *minsup* cho trước, bước thứ hai, từ các tập mục thường xuyên tìm được, sinh ra các luật kết hợp thỏa mãn ngưỡng độ tin cậy *minconf* cho trước. Mọi khó khăn của bài toán khai phá luật kết hợp tập trung ở bước thứ nhất, đó là khai phá tất cả các tập mục thường xuyên thỏa mãn ngưỡng độ hỗ trợ cho trước.

Sinh luật kết hợp là một kỹ thuật quan trọng của khai phá dữ liệu. Mục tiêu là phát hiện những mối quan hệ giữa các giá trị dữ liệu trong cơ sở dữ liệu.

Sau đây chúng tôi trình bày một số khái niệm cơ bản liên quan bài toán khai phá tập mục thường xuyên.

Cơ sở dữ liệu giao tác

Định nghĩa 1.

Cho tập các mục (item) $I = \{i_1, i_2, ..., i_n\}$. Một giao tác (transaction) T là một tập con của *I*, $T \subseteq I$. Cơ sở dữ liệu giao tác là một tập các giao tác $DB = \{T_1, T_2, ..., T_m\}$. Mỗi giao tác được gán một định danh *TID*. Một tập mục con $X \subseteq I$, gồm k mục phân biệt được gọi là một k-tập mục. Giao tác *T* gọi là chứa tập mục *X* nếu $X \subseteq T$.

Ma trận giao tác: Cơ sở dữ liệu giao tác $DB = \{T_1, T_2, ..., T_m\}$ trên tập các mục (item) $I = \{i_1, i_2, ..., i_n\}$ được biểu diễn bởi ma trận nhị phân $M = (m_{pq})_{m \times n}$, ở đó:

$$m_{pq} = \begin{cases} 1 \text{ khi } i_q \in T_p \\ 0 \text{ khi } i_q \notin T_p \end{cases}$$

Tập mục thường xuyên và luật kết hợp

Định nghĩa 2. Cho tập mục $X \subseteq I$. Ta gọi độ hỗ trợ (Support) của *X* trong cơ sở dữ liệu giao tác *DB*, kí hiệu sup(*X*), là tỷ lệ phần trăm các giao tác chứa *X* trên tổng số các giao tác trong *DB*, tức là: $\sup(X) = \dfrac{|\{T \in DB \mid T \supseteq X\}|}{|DB|}$

Ta có: $0 \le \sup(X) \le 1$ với mọi tập mục $X \subseteq I$.

Định nghĩa 3. Cho tập mục $X \subseteq I$ và ngưỡng hỗ trợ tối thiểu (minimum support) $minsup \in [0,1]$ (được xác định trước bởi người sử dụng). *X* được gọi là tập mục thường xuyên (frequent itemset hoặc large itemset) với độ hỗ trợ tối thiểu *minsup* nếu $\sup(X) \ge minsup$, ngược lại *X* gọi là tập mục không thường xuyên.

Định nghĩa 4. Một luật kết hợp là một biểu thức dạng $X \to Y$, trong đó *X* và *Y* là các tập con của *I*, $X \cap Y = \emptyset$; *X* gọi là tiền đề, *Y* gọi là kết luận của luật.

Luật kết hợp có hai thông số quan trọng là *độ hỗ trợ* và *độ tin cậy*.

Định nghĩa 5. Độ hỗ trợ (Support) của một luật kết hợp $X \to Y$, kí hiệu là $\sup(X \to Y)$, là độ hỗ trợ của tập mục $X \cup Y$, $\sup(X \to Y) = \sup(X \cup Y)$.

Như vậy độ hỗ trợ của luật kết hợp $X \to Y$ chính là xác suất $P(X \cup Y)$ của sự xuất hiện đồng thời của *X* và *Y* trong một giao tác.

Ta có: $0 \le \sup(X \to Y) \le 1$.

Định nghĩa 6. Độ tin cậy (Confidence) của một luật $X \to Y$, kí hiệu $\text{conf}(X \to Y)$, là tỷ lệ phần trăm giữa số giao tác chứa $X \cup Y$ và số giao tác chứa *X* trong cơ sở dữ liệu *DB*.

$$\text{conf}(X \rightarrow Y) = \frac{\sup(X \cup Y)}{\sup(X)}$$

Độ tin cậy của luật kết hợp $X \rightarrow Y$ chính là xác suất có điều kiện P(*Y/X*) :

$$\text{P}(Y / X) = \frac{\left|\{T \in DB \mid X \subseteq T \wedge Y \subseteq T\}\right|}{\left|\{T \in DB \mid X \subseteq T\}\right|} = \frac{\left|\{T \in DB \mid X \cup Y \subseteq T\}\right|}{\left|\{T \in DB \mid X \subseteq T\}\right|} = \frac{\sup(X \cup Y)}{\sup(X)}$$

và ta có $0 \leq \text{conf}(X \rightarrow Y) \leq 1$.

Các luật thoả mãn cả hai ngưỡng độ hỗ trợ tối thiểu (*minsup*) và độ tin cậy tối thiểu (*minconf*), tức thỏa mãn $\sup(X \rightarrow Y) \geq minsup$ và $\text{conf}(X \rightarrow Y) \geq minconf$, được gọi là luật kết hợp mạnh.

Tính chất cơ bản của tập mục thường xuyên

Cho cơ sở dữ liệu giao tác *DB* và ngưỡng độ hỗ trợ tối thiểu *minsup.* Các tập mục thường xuyên có các tính chất sau :

(1) Nếu *X, Y* là các tập mục và $X \subseteq Y$ thì $\sup(X) \geq \sup(Y)$.

(2) Nếu một tập mục là không thường xuyên thì mọi tập cha của nó cũng không thường xuyên.

(3) Nếu một tập mục là thường xuyên thì mọi tập con khác rỗng của nó cũng là tập mục thường xuyên.

Tính chất (3) được gọi là tính chất Apriori, tính chất này là cơ sở để rút gọn không gian tìm kiếm các tập mục thường xuyên.

Cho cơ sở dữ liệu giao tác *DB*, ngưỡng độ hỗ trợ tối thiểu *minsup* và ngưỡng độ tin cậy tối thiểu *minconf.*

Yêu cầu: Tìm tất cả các luật kết hợp $X \rightarrow Y$ trên cơ sở dữ liệu *DB* sao cho $\sup(X \rightarrow Y) \geq minsup$ và $\text{conf}(X \rightarrow Y) \geq minconf$.

Bài toán khai phá luật kết hợp này được gọi là bài toán cơ bản hay bài toán nhị phân, vì ở đây, giá trị của mục dữ liệu trong cơ sở dữ liệu là 0 hoặc 1 (xuất hiện hay không xuất hiện).

Bài toán khai phá luật kết hợp được chia thành hai bài toán con. Bài toán thứ nhất là tìm tất cả các tập mục thỏa mãn độ hỗ trợ tối thiểu cho trước, tức là tìm tất cả các tập mục thường xuyên. Bài toán thứ hai là sinh ra các luật kết hợp từ các tập mục thường xuyên đã tìm được thỏa mãn độ tin cậy tối thiểu cho trước.

Bài toán thứ hai được giải quyết như sau : giả sử đã tìm được *X* là tập mục thường xuyên, ta sinh ra các luật kết hợp bằng cách tìm $\forall Y \subset X$, kiểm tra độ tin cậy của luật $X \setminus Y \rightarrow Y$ có thỏa mãn độ tin cậy tối thiểu không. Bài toán thứ hai này đơn giản, mọi khó khăn nằm ở bài toán thứ nhất, hầu hết các nghiên cứu về luật kết hợp đều tập trung giải quyết bài toán thứ nhất là tìm các tập mục thường xuyên.

2.2.2. Một số khái niệm liên quan đến lí 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. Một cách hình thức, hệ thông tin được định nghĩa như sau.

Định nghĩa 1. Hệ thông tin là một bộ tứ $IS=(U,A,V,f)$ trong đó U là tập hữu hạn, khác rỗng các đối tượng; A là tập hữu hạn, khác rỗng các thuộc tính; $V=\bigcup_{a\in A}V_a$ với V_a là tập giá trị của thuộc tính $a\in A$; $f:U\times A\to V_a$ là hàm thông tin, $\forall a\in A, u\in U \quad f(u,a)\in V_a$.

Với mọi $u\in U$, $a\in A$, ta kí 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 kí 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*.

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, kí hiệu là $IND(P)$, 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 trên U, kí hiệu là $U/IND(P)$ hay U/P. Kí hiệu lớp tương đương trong phân hoạch U/P chứa đối tượng u là $[u]_P$, khi đó $[u]_P=\{v\in U \mid (u,v)\in IND(P)\}$.

Định nghĩa 2. [4,27] Cho hệ thông tin $IS=(U,A,V,f)$ và $P,Q\subseteq A$. Ta nói:

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 (viết $U/P\preceq U/Q$) khi và chỉ khi $\forall u\in U,\ [u]_P\subseteq[u]_Q$.

Cho hệ thông tin $IS=(U,A,V,f)$ và tập đối tượng $X\subseteq U$. Với một tập thuộc tính $B\subseteq A$ 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 lí 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 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, kí hiệu là lượt là $\underline{B}X$ và $\overline{B}X$, được xác định như sau:

$$\underline{B}X = \{u \in U \mid [u]_B \subseteq X\},\ \overline{B}X = \{u \in U \mid [u]_B \cap X \neq \varnothing\}.$$

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

$$BN_B(X) = \overline{B}X - \underline{B}X : \textit{B-miền biên của } X,\quad U - \overline{B}X : \textit{B-miền ngoài của } X.$$

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

$$\underline{B}X = \bigcup\{Y \in U/B \mid Y \subseteq X\},\ \overline{B}X = \bigcup\{Y \in U/B \mid Y \cap X \neq \varnothing\}.$$

Trong trường hợp $BN_B(X) = \varnothing$ thì *X* được gọi là *tập chính xác (exact set)*, ngược lại *X* được gọi là *tập thô (rough set)*.

Với $B, D \subseteq A$, 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 $v \in U$ mà $u(B) = v(B)$ ta đều có $u(D) = v(D)$. Nói cách khác, $POS_B(D) = \{u \in U \mid [u]_B \subseteq [u]_D\}$

Một lớp đặc biệt của các 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 $C \cap D = \varnothing$.

Xét bảng quyết định $DS = (U, C \cup D, V, f)$ với giả thiết $\forall u \in U, \forall d \in D$, $d(u)$ *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 bài báo 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 hàm 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 $C \to D$ đúng.

3. KẾT QUẢ NGHIÊN CỨU

3.1. Cơ sở dữ liệu

Cho trước quan hệ r và hệ Sperner K trên R. Chúng ta nói rằng r thể hiện K nếu $K_r = K$. Những kết quả sau có thể thấy tại [28, 14].

Định lí 1. Giả sử K là một hệ Sperner không rỗng, r là một là một quan hệ trên R. Khi đó r thể hiện K nếu và chỉ nếu $K^{-1} = M_r$, ở đây M_r là hệ bằng nhau cực đại của r.

Cho trước s = <R,F> là một sơ đồ quan hệ trên R, K_s là tập tất cả các khoá tối tiểu của s. Kí pháp K_s^{-1} là tập các phản khoá của s. Từ Định lí 1 chúng ta có kết quả sau.

Hệ quả 1. Cho trước s = <R,F> là một sơ đồ quan hệ và r là một quan hệ trên R. Khi đó $K_r = K_s$ nếu và chỉ nếu $K_s^{-1} = M_r$, ở đây M_r là hệ bằng nhau cực đại của r.

Định nghĩa 1. Giả sử r là một quan hệ trên R và K_r là tập của tất cả các khoá tối tiểu của r. Chúng ta nói rằng a là một thuộc tính cơ bản của r nếu tồn tại một khoá tối tiểu K ($K \in K_r$) để a là một phần tử của K.

Nếu a không thoả mãn tính chất trên thì a là thuộc tính thứ cấp.

Chúng ta có thể thấy các thuộc tính cơ bản và thứ cấp đóng một vai trò quan trọng trong việc chuẩn hoá các sơ đồ quan hệ và các quan hệ.

Người ta đã chứng minh kết quả sau

Cho trước một sơ đồ quan hệ s = <R,F> và một thuộc tính a. Bài toán xác định a là thuộc tính cơ bản hay không là bài toán NP- đầy đủ.

Có nghĩa rằng cho đến nay không có một thuật toán có độ phức tạp thời gian đa thức để giải quyết bài toán này. Tuy vậy, chúng ta chỉ ra rằng đối với quan hệ thì bài toán này được giải bằng một thuật toán thời gian đa thức.

Trước tiên chúng ta chứng minh kết quả sau [1, 3].

Định lí 2. Giả sử *K* là một hệ Sperner trên R thì

$$\cup K = R - \cap K^{-1}.$$

Trên cơ sở Định lí 1 và Định lí 2 chúng ta chỉ ra rằng đối với một quan hệ, thì vấn đề về thuộc tính cơ bản có thể là giải quyết bằng một thuật toán thời gian đa thức.

Đầu tiên chúng ta xây dựng một thuật toán xác định tập các thuộc tính cơ bản của quan hệ cho trước.

Thuật toán 1.

Vào: r = $\{h_1, ..., h_m\}$ là một quan hệ trên R

Ra: V là tập tất cả thuộc tính cơ bản của r

Bước 1: Từ r chúng ta xây dựng một tập $E_r = \{E_{ij} : m \geq j > i \geq 1\}$ và $E_{ij} = \{ a \in R: h_j(a) = h_j(a) \}$

Bước 2: Từ E_r chúng ta xây dựng tập

$M = \{B \in P(R)$: Tồn tại $E_{ij} \in E_r$: $E_{ij} = B\}$

Bước 3: Từ M xây dựng tập $M_r = \{ B \in M$: Với mọi $B' \in M$: $B \not\subset B'\}$

Có thể thấy rằng M_r tính được bằng một thuật toán thời gian đa thức.

Bước 4: Xây dựng tập $V = R - \cap M_r$.

Rõ ràng $m.(m+1)/2 \geq |E_r| \geq |M| \geq |M_r|$. Bởi vậy thời gian tính của Thuật toán 1 là một đa thức theo số hàng và số cột của r.

Như vậy là tồn tại thuật toán đối với một quan hệ r cho trước, xác định một thuộc tính bất kì là cơ bản hay không với thời gian tính đa thức theo số hàng và cột của r.

Mối quan hệ giữa quan hệ Armstrong và sơ đồ quan hệ

Việc xây dựng quan hệ Armstrong của một sơ đồ quan hệ cho trước và ngược lại từ quan hệ cho trước ta xây dựng một SĐQH sao cho quan hệ cho trước này là quan hệ Armstrong của nó có vai trò rất quan trọng trong việc phân tích cấu trúc lôgic của mô hình dữ liệu quan hệ cả trong thiết kế lẫn trong ứng dụng. Đã có nhiều tác giả nghiên cứu vấn đề này. Trong mục này chúng tôi trình bày hai thuật toán giải quyết bài toán trên và đưa ra việc đánh giá các thuật toán này cũng như đánh giá độ phức tạp của bài toán trên.

Trong [13, 31] chúng tôi đã trình bày các kết quả sau:

Định lí 3. Tồn tại một thuật toán để tìm SĐQH s = <R,F> từ một quan hệ r cho trước sao cho $F^+ = F_r$.

Ngược lại

Định lí 4. Tồn tại một thuật tóan để tìm một quan hệ r từ SĐQH s = <R,F> cho trước sao cho $F^+ = F_r$.

Định lí 5. Độ phức tạp thời gian cho việc tìm kiếm một quan hệ Armstrong của một SĐQH cho trước là hàm số mũ theo số lượng của các thuộc tính.

Định lí 6. Độ phức tạp thời gian cho việc tìm kiếm một SĐQH s = <R,F> từ một quan hệ r cho trước sao cho $F_r = F^+$ là hàm số mũ theo số lượng các thuộc tính.

Về chuẩn hóa dữ liệu

Việc chuẩn hoá các quan hệ cũng như các sơ đồ quan hệ đóng một vai trò cực kì quan trọng trong việc thiết kế các hệ quản trị cơ sở dữ liệu trên mô hình dữ liệu của Codd. Nhờ có chuẩn hoá các quan hệ và các sơ đồ quan hệ chúng ta tránh được việc dư thừa dữ liệu và tăng tốc độ của các phép toán xử lí quan hệ [15,17,29].

Chúng ta định nghĩa các dạng chuẩn như sau.

Cho $r = \{h_1,...,h_m\}$ là quan hệ trên $R = \{a_1 ...., a_n\}$

Định nghĩa 1. (Dạng chuẩn 1 - 1NF):

r là dạng chuẩn 1 nếu các phần tử của nó là sơ cấp.

Khái niệm sơ cấp hiểu ở đây là giá trị $h_i(a_j)$ (i=1,...,m; j=1,...,n) không phân chia được nữa.

Định nghĩa 2 (Dạng chuẩn 2 - 2NF)

r là dạng chuẩn 2 nếu:

- r là dạng chuẩn 1

- $A \rightarrow \{a\} \notin F_r$ đối với mọi khoá tối thiểu K, $A \subset K$ và a là thuộc tính thứ cấp.

Định nghĩa 3. (Dạng chuẩn 3 - 3NF):

r là dạng chuẩn 3 nếu:

$A \rightarrow \{a\} \notin F_r$ đối với A mà $A^+ \neq R$, $a \notin A$, $a \notin \cup K$

Định nghĩa 4. (Dạng chuẩn Boye-Codd - BCNF)

r là dạng chuẩn của Boye-Codd nếu:

$A \to \{a\} \notin F_r$ đối với A mà $A^+ \neq R$, $a \notin A$

Qua định nghĩa, ta có thể thấy dạng chuẩn

BCNF là 3NF và 3NF là 2NF. Tuy vậy, chúng ta có thể đưa ra các ví dụ chứng tỏ có quan hệ là 2NF nhưng không là 3NF và có quan hệ là 3NF nhưng không là BCNF.

Nói cách khác là lớp các quan hệ BCNF là lớp con thực sự của lớp các quan hệ 3NF và lớp các quan hệ 3NF này lại là lớp con thực sự của lớp các quan hệ 2NF.

Đối với s = <F,R> thì các dạng chuẩn 2NF, 3NF, BCNF trong đó ta thay F_r bằng F^+.

Dạng chuẩn 2NF

Định lí 1. Giả sử s = <R, F> là sơ đồ quan hệ. Đặt $M_s = \{A - a; a \in A, A \in K_s\}$, và F_n là tập tất cả các thuộc tính thứ cấp của s. Đặt $l_s = \{B: B = C^+, C \in M_s\}$. Khi đó ta có các tương đương sau:

(1) s là 2NF.

(2) Với mỗi $C \in M_s$: $C^+ \cap F_n = \varnothing$;

(3) Với mỗi $B \in l_s$ và $a \in F_n$: $(B - a)^+ = B - a$.

Từ định lí 1 trực tiếp suy ra kết quả sau

Hệ quả 1. Giả sử s = (R, F) là một sơ đồ quan hệ. Kí pháp F_n là tập tất cả những thuộc tính thứ cấp của s, và $G_s = \{B - F_n: B \in K_s^{-1}\}$. Khi đó nếu đối với mọi $C \in G_s$: $C^+ = C$ thì s là 2NF.

Dạng chuẩn 3NF

Định lí 2. Cho s = <R, F> là một sơ đồ quan hệ. Đặt F_n là tập tất cả các thuộc tính thứ cấp của s. Khi đó s là 3NF nếu và chỉ nếu $\forall B \in K_s^{-1}$, $a \in F_n$: $(B - a)^+ = B - a$.

Định lí 3. Giả sử r là một quan hệ trên R. Khi đó r là 3NF nếu và chỉ nếu với mọi $A \in E_r$, $a \in A$ và a là thuộc tính thứ cấp thì $\{A- a\}_r^+ = A- a$, ở đây E_r là hệ bằng nhau của r.

Từ Định lí 3 ta có hệ quả sau

Hệ quả 2. Giả sử s là một sơ đồ quan hệ trên R. Khi đó s là 3NF nếu và chỉ nếu với mọi A: $A^+ = A$, $a \in A$ và a là thuộc tính thứ cấp thì $\{A - a\}^+ = A- a$.

Dạng chuẩn BCNF

Trong mục này, chúng ta đưa ra một số các đặc trưng của dạng chuẩn BCNF cho sơ đồ quan hệ và quan hệ.

Định lí 4. Cho s = <R, F> là một sơ đồ quan hệ. Đặt F_n là tập tất cả các thuộc tính thứ cấp của s. Khi đó s là BCNF nếu và chỉ nếu $\forall B \in K_s^1$, $a \in B$: $(B - a)^+ = B - a$.

Định lí 5. Giả sử r là một quan hệ trên R. Khi đó r là BCNF nếu và chỉ nếu với mọi $A \in M_r$, $a \in A$ thì $\{A- a\}_r^+ = A- a$, ở đây M_r là hệ bằng nhau cực đại của r.

Trên cơ sở các định lí đã trình bày ở các mục trên, chúng ta xây dựng các thuật toán để xác định dạng chuẩn cho các quan hệ hoặc sơ đồ quan hệ cho trước.

Đầu tiên chúng ta xây dựng thuật toán xác định một quan hệ cho trước có là 3NF hay không.

Thuật toán 1.

Đầu vào: r = $\{h_1, ..., h_m\}$ là một quan hệ trên R

Đầu ra: r là 3NF ?

Bước 1: Từ r chúng ta xây dựng một tập $E_r = \{E_{ij} : m \geq j > i \geq 1\}$, ở đây $E_{ij} = \{ a \in R: h_j(a) = h_j(a)\}$.

Bước 2: Từ E_r chúng ta xây dựng một tập $M = \{B \in P(R)$: Tồn tại $E_{ij} \in E_r: E_{ij} = B\}$.

Bước 3: Từ M xây dựng tập $M_r = \{ B \in M$: Với mọi $B' \in M: B \not\subset B'\}$.

Có thể thấy rằng M_r tính được bằng một thuật toán thời gian đa thức.

Bước 4: Xây dựng tập $V = \cap M_r$.

Bước 5: r là 3NF nếu với mọi $B \in M_r$, $a \in V$: $\{B - a\}_r^+ = B - a$. Ngược lại r không là 3NF.

Trên cơ sở Định lí 5 chúng ta xây dựng thuật toán dưới đây

Thuật toán 2.

Đầu vào: r = $\{h_1, ..., h_m\}$ là một quan hệ trên R

Đầu ra: r là BCNF ?

Bước 1: Từ r chúng ta xây dựng một tập $E_r = \{E_{ij} : m \geq j > i \geq 1\}$ và $E_{ij} = \{a \in R: h_j(a) = h_j(a)\}$

Bước 2: Từ E_r chúng ta xây dựng một tập $M = \{B \in P(R)$: Tồn tại $E_{ij} \in E_r: E_{ij} = B\}$

Bước 3: Từ M xây dựng tập $M_r = \{B \in M$: Với mọi $B' \in M: B \not\subset B'\}$. Có thể thấy rằng M_r tính được bằng một thuật toán thời gian đa thức.

Bước 4: r là BCNF nếu với mọi $B \in M_r$, $a \in B$: $\{B - a\}_r^+ = B - a$. Ngược lại r không là BCNF.

Chúng ta có thể thấy thuật toán dưới đây

Thuật toán 3.

Đầu vào: s = <R,F> là một sơ đồ quan hệ trên R, với

F = $\{ A_1 \rightarrow B_1, ..., A_m \rightarrow B_m \}$

Đầu ra: s là BCNF ?

Bước 1: Nếu $A_1 \rightarrow B_1$ là phụ thuộc hàm không tầm thường và $A_1^+ \# R$ thì dừng và kết luận s không là BCNF. Ngược lại thì chuyển sang bước tiếp theo.

.....

Bước m: Giống như bước 1 nhưng đối với $A_m \rightarrow B_m$.

Bước m+1: s là BCNF.

Định lí 4. Cho trước một quan hệ r và một sơ đồ quan hệ s. Khi đó đều tồn tại một thuật toán có độ phức tạp thời gian đa thức theo kích thước của r (s) để kiểm tra r (s) có là BCNF hay không.

Định lí 5. Cho trước r là một quan hệ trên R. Khi đó tồn tại một thuật toán có độ phức tạp thời gian đa thức để kiểm tra r có là 3NF hay không.

Tuy vậy, đối với đầu vào là s thì đây lại là bài toán NP đầy đủ.

Có nghĩa là cho đến nay, độ phức tạp thời gian của bài toán này không là đa thức.

Với trường hợp 2NF, các câu hỏi tương tự cho cả r lẫn s còn là bài toán mở (Chúng tôi phỏng đoán có độ phức tạp thời gian là hàm mũ trở lên).

3.2. Về khai phá tập mục thường xuyên sinh luật kết hợp

Bài toán cơ bản khai phá luật kết hợp do Agrawal và đồng sự đề xuất. Mục tiêu của bài toán là phát hiện các tập mục thường xuyên, từ đó tạo các luật kết hợp. Trong mô hình của bài toán này, giá trị của mỗi mục dữ liệu trong một giao tác là 0 hoặc 1, tức là chỉ quan tâm mục dữ liệu có xuất hiện trong giao tác hay không. Bài toán cơ bản này có nhiều ứng dụng, tuy vậy, do tập mục thường xuyên chỉ mang ngữ nghĩa thống kê nên nó chỉ đáp ứng được phần nào nhu cầu của thực tiễn.

Nhằm khắc phục hạn chế của bài toán cơ bản khai phá luật kết hợp, nhiều nhà nghiên cứu đã mở rộng bài toán theo nhiều hướng khác nhau. Năm 1998, Hilderman và các cộng sự đề xuất bài toán khai phá tập mục cổ phần cao [19]. Trong mô hình này, giá trị của mục dữ liệu trong giao tác là một số, số đó có thể là số nguyên (như số lượng đã bán của mặt hàng). Cổ phần (hay đóng góp) của một tập mục là số đo tỷ lệ đóng góp của tập mục trong cơ sở dữ liệu. Khai phá tập mục cổ phần cao là khám phá tất cả các tập mục có cổ phần không nhỏ hơn ngưỡng quy định bởi người sử dụng.

Trong bài toán cơ bản, các thuật toán khám phá được xây dựng theo phương pháp tìm kiếm từng bước. Cơ sở của các thuật toán là tính chất Apriori của tập mục thường xuyên (hay còn gọi là tính chất phản đơn điệu – Anti monotone). Trong mô hình khai phá tập mục cổ phần cao, tính chất này không còn đúng nữa. Vì vậy việc rút gọn không gian tìm kiếm không thể thực hiện được như đối với khai phá tập mục thường xuyên. Trong [22,25], các tác giả đã đề nghị một số thuật toán khai phá tập mục cổ phần cao như các thuật toán ZP, ZSP, SIP, FSM,.... Trong đó, thuật toán FSM trong [22] là một thuật toán nhanh, cho phép khám phá tất cả các tập mục cổ phần cao trong cơ sở dữ liệu giao tác cho trước.

Trong [6,32] chúng tôi đề xuất khái niệm "tập mục cổ phần theo giao tác cao" và chứng minh nó có tính chất phản đơn điệu (anti monotone), có thể ứng dụng vào nhiều thuật toán khai phá tập mục thường xuyên đã có để tìm được tập mục cổ phần theo giao tác cao, từ đó tìm ra tập mục cổ phần cao. Sử dụng ý tưởng này, chúng tôi đề xuất thuật toán AFSM (*Advanced FSM)* dựa trên các bước của thuật toán FSM với phương pháp mới tỉa hiệu quả hơn các tập mục ứng viên.

Như phần trên đã trình bày, ràng buộc cổ phần không có tính chất phản đơn điệu như tập mục thường xuyên, đây chính là trở ngại của bài toán khai phá tập mục cổ phần cao. Để khắc phục điều này, luận án đề xuất khái niệm "giá trị theo giao tác của tập mục", "tập mục cổ phần theo giao tác cao" và chứng minh tập mục cổ phần theo giao tác cao có tính chất phản đơn điệu, do đó có thể sử dụng để tỉa các tập mục ứng viên.

Định nghĩa 1: Cho tập mục *X*, db_X là tập các giao tác chứa *X*. Giá trị theo giao tác (*transaction*

measure value) của tập mục *X*, kí hiệu *tmv(X)*, là tổng giá trị của tất cả các giao tác chứa tập mục *X* , tức là $tmv(X) = Tmv(db_X) = \sum_{T_q \in db_X} tmv(T_q)$.

Định nghĩa 2: Tập mục *X* được gọi là *tập mục cổ phần theo giao tác cao* nếu $tmv(X) \geq min\_lmv$. Trường hợp ngược lại, *X* được gọi là *tập mục cổ phần theo giao tác thấp*.

Mệnh đề 1: Tập mục cổ phần theo giao tác cao có tính chất phản đơn điệu (Anti Monotone).

Chứng minh:

Xét hai tập mục *X, Y* sao cho $Y \subset X$, ta chứng minh nếu *Y* là tập mục cổ phần theo giao tác thấp thì *X* cũng là tập mục cổ phần theo giao tác thấp.

Ta có $Y \subset X$ nên $db_Y \supseteq db_X$, do đó

$$tmv(Y) = Tmv(db_Y) \geq Tmv(db_X) = tmv(X).$$

Nếu *Y* là tập mục cổ phần theo giao tác thấp, tức là $tmv(Y) < min\_lmv$ thì $tmv(X) \leq tmv(Y) < min\_lmv$, *X* cũng là tập mục cổ phần theo giao tác thấp. □

Mệnh đề 1 cho biết các tập mục cổ phần theo giao tác cao có tính chất phản đơn điệu như tính chất của tập mục thường xuyên, do đó có thể sử dụng tính chất này để tỉa các ứng viên khi khai phá.

Mệnh đề 2: Nếu tập mục *X* là tập mục cổ phần cao thì *X* cũng là tập mục cổ phần theo giao tác cao.

Chứng minh: Kí hiệu db_X là tập các giao tác chứa tập mục *X*, ta có:

$$lmv(X) = \sum_{T_q \in db_X} imv(X, T_q) = \sum_{T_q \in db_X} \sum_{i_p \in X} mv(i_p, T_q) \leq \sum_{T_q \in db_X} \sum_{i_p \in T_q} mv(i_p, T_q) = tmv(X)$$

Do đó, nếu *X* là tập mục cổ phần cao, tức $lmx(X) \geq min\_lmv$, thì *X* cũng là tập mục cổ phần theo giao tác cao vì $tmv(X) \geq lmx(X) \geq min\_lmv$. □

Từ Mệnh đề 2 có thể suy ra tập các tập mục cổ phần cao chứa trong tập các tập mục cổ phần theo giao tác cao. Theo Mệnh đề 1, các tập mục cổ phần theo giao tác cao có tính chất phản đơn điệu như tập mục thường xuyên, do đó ta có thể áp dụng một số thuật toán khai phá tập mục thường xuyên đã có (như các thuật toán kiểu Apriori, thuật toán tìm kiếm theo chiều sâu FP-growth,...), thay số lần xuất hiện của tập mục bởi giá trị theo giao tác của tập mục thì sẽ nhận được kết quả khai phá là các tập mục cổ phần theo giao tác cao. Khi đó ta chỉ cần duyệt lại cơ sở dữ liệu để tính giá trị đóng góp thực sự của các tập mục cổ phần theo giao tác cao để nhận được các tập mục cổ phần cao.

Từ các cơ sở lí thuyết đã trình bày, chúng tôi đề xuất thuật toán AFSM như sau:

Thuật toán AFSM( )

Input: Cơ sở dữ liệu giao tác *DB*, ngưỡng cổ phần *minShare (s%).*

Output: Tập *HS* gồm các tập mục cổ phần cao.

Method:

1. *k:=1, HS_1:=∅, C_1:=I*;
2. **for each** *T*∈*DB* // duyệt cơ sở dữ liệu *DB*
3. tính *lmv(i_p)* và $tmv(i_p)$ cho $\forall i_p \in C_1$;
4. **for each** $i_p \in C_1$
5. **if** $tmv(i_p) < min\_lmv$ **then**
6. $C_1 := C_1 \setminus \{i_p\}$
7. **else if** *lmv(i_p)* ≥*min_lmv* **then**
8. $HS_1 := HS_1 \cup \{i_p\}$;
9. $RC_1 := C_1$;
10. **repeat**
11. $k := k + 1$;
12. for each X_p, X_q $\in RC_{k-1}$
13. C_k :=Apriori-gen(X_p, X_q);
14. **for each** *T*∈*DB* // duyệt cơ sở dữ liệu *DB*
15. tính *lmv(X)* và $tmv(X)$ cho $\forall X \in C_k$;
16. **for each** $X \in C_k$
17. **if** $tmv(X) < min\_lmv$ **then**
18. $C_k := C_k \setminus \{X\}$
19. **else if** *lmv(X)*≥*min_lmv* **then**
20. $HS_k := HS_k \cup \{X\}$;
21. $RC_k := C_k$;
22. **until** $C_k = \varnothing$;
23. **return** $HS = \cup HS_k$;

Khai phá tập mục lợi ích cao là sự mở rộng, tổng quát hóa của khai phá tập mục cổ phần cao. Mô hình khai phá tập mục lợi ích cao được Yao và cộng sự đề xuất [34, 35]. Trong mô hình khai phá tập mục lợi ích cao, giá trị của mục dữ liệu trong giao tác là một số (như số lượng đã bán của mặt hàng, gọi là giá trị khách quan), ngoài ra còn có bảng lợi ích cho biết lợi ích mang lại khi bán một đơn vị hàng đó (gọi là giá trị chủ quan, do người quản lí kinh doanh xác định). Lợi ích của một tập mục là số đo lợi nhuận mà tập mục đó đóng góp trong cơ sở dữ liệu, nó có thể là tổng lợi nhuận, là tổng chi phí của tập mục. Khai phá tập mục lợi ích cao là khám phá tất cả các tập mục có lợi ích không nhỏ hơn ngưỡng lợi ích tối thiểu quy định bởi người sử dụng.

Trong [34, 35], Hong Yao và Howard Hamilton đề xuất phương pháp khai phá và các chiến lược tỉa dựa trên các tính chất của ràng buộc lợi ích, thể hiện trong hai thuật toán Umining và

Umining H. Các thuật tỉa mà hai thuật toán này áp dụng có khả năng thu gọn phần nào tập ứng viên, tuy vậy có những nhược điểm nên hiệu quả không cao.

Trong [25], Liu đưa ra khái niệm lợi ích của giao tác và lợi ích của tập mục tính theo lợi ích của các giao tác chứa nó gọi là lợi ích TWU (Transaction-weighted Utilization). Lợi ích theo giao tác TWU có tính chất phản đơn điệu như tính chất của tập mục thường xuyên và tập tất cả các tập mục lợi ích cao chứa trong tập tất cả các tập mục lợi ích TWU cao. Y. Liu đề xuất thuật toán hiệu quả gồm hai pha để khai phá tập mục lợi ích cao. Thuật toán rút gọn không gian tìm kiếm nhờ áp dụng tính chất phản đơn điệu của lợi ích TWU. Tuy nhiên, thuật toán thực hiện kém hiệu quả khi khai phá các tập dữ liệu dày và mẫu dài vì tốn nhiều thời gian cho việc sinh ra khối lượng khổng lồ các tập mục ứng viên và tính lợi ích TWU của nó trong mỗi lần duyệt cơ sở dữ liệu. Thuật toán phải duyệt cơ sở dữ liệu nhiều lần, số lần duyệt bằng với chiều dài của mẫu dài nhất tìm được, do đó, khi số mục dữ liệu lớn thì khối lượng tính toán là vô cùng lớn.

Trong [11], A. Erwin và đồng sự đề xuất các thuật toán CTU-Mine và CTU-PRO khai phá tập mục lợi ích cao theo cách phát triển các mẫu trên cấu trúc cây. Thuật toán CTU-Mine khai phá hiệu quả hơn thuật toán Hai pha chỉ trong cơ sở dữ liệu dày với ngưỡng lợi ích thấp. Thuật toán CTU-PRO có cải tiến so với thuật toán CTU-Mine nên khai phá hiệu quả hơn thuật toán Hai pha và thuật toán CTU-Mine.

Trong [32] chúng tôi đề xuất ba thuật toán khai phá tập mục lợi ích cao dựa trên cấu trúc cây đơn giản hơn và cách khai phá không đệ quy. Các thuật toán đề xuất sử dụng cấu trúc cây FP-tree được Han, Wang và Yin giới thiệu năm 2000 trong [18], cách khai phá cây FP-tree không đệ quy bởi cấu trúc cây COFI-tree do Mohammad El-Hajj và Osmar R. Zaiane đề xuất năm 2003 trong [12]. Hai thuật toán đầu sử dụng cấu trúc cây FP-tree để xây dựng cây chứa thông tin của các giao tác, sau đó khai phá cây này để tìm các tập mục lợi ích cao. Thuật toán thứ ba chuyển đổi dữ liệu thành dạng ma trận và lưu ở bộ nhớ ngoài, sau khi đã chuyển đổi sang dạng biểu diễn mới, có thể khai phá với các ngưỡng lợi ích khác nhau. Thuật toán thứ ba này có thể khai phá được các tập dữ liệu rất lớn vì hầu như toàn bộ dữ liệu đặt tại bộ nhớ ngoài, chỉ đưa vào bộ nhớ trong một phần nhỏ của dữ liệu để khai phá. Ba thuật toán đề xuất thực hiện khai phá hiệu quả vì các lí do: 1) Số lần duyệt cơ sở dữ liệu ít, 2) Không sinh ra khối lượng khổng lồ các tập mục ứng viên, giảm chi phí tính toán và 3) Sử dụng tiết kiệm bộ nhớ.

3.3. Lí thuyết tập thô

Trong bảng quyết định, nhiều phương pháp rút gọn thuộc tính đã được công bố. Mỗi phương pháp đều đưa ra định nghĩa *tập rút gọn* của phương pháp đó dựa trên một độ đo. Ở đây, trong bài báo này chúng tôi chỉ trình bày ba định nghĩa cơ bản về tập rút gọn.

Định nghĩa 1. [27] Cho bảng quyết định $DS = (U, C \cup D)$ 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, gọi tắt là *tập rút gọn miền dương*. Kí hiệu $PRED(C)$ là họ tất cả các tập rút gọn miền dương.

Tập rút gọn dựa trên độ đo entropy Shannon có điều kiện do G.Wang và các cộng sự [36] đề xuất.

Cho bảng quyết định $DS = (U, C \cup D)$. Giả sử $U/C = \{C_1, C_2, ..., C_m\}$, $U/D = \{D_1, D_2, ..., D_n\}$. Entropy Shannon có điều kiện của *D* khi đã biết *C* được định nghĩa bởi

$$H\left(D\middle|C\right)=-\sum_{i=1}^{m}\frac{\left|C_i\right|}{\left|U\right|}\sum_{j=1}^{n}\frac{\left|C_i\cap D_j\right|}{\left|C_i\right|}\log_2\frac{\left|C_i\cap D_j\right|}{\left|C_i\right|}$$

trong đó $\left|X\right|$ kí hiệu lực lượng của tập X và với quy ước $0.\log_2 0=0$.

Định nghĩa 2. [36] Cho bảng quyết định $DS=\left(U,C\cup D\right)$ và tập thuộc tính $R\subseteq C$. Nếu

1) $H\left(D\middle|R\right)=H\left(D\middle|C\right)$

2) $\forall r\in R, H(D\middle|R-\{r\})\neq H(D\middle|C)$

thì R là một rút gọn của C dựa trên entropy Shannon có điều kiện, gọi tắt là *tập rút gọn Entropy Shannon*. Kí hiệu $HRED\left(C\right)$ là họ tất cả các tập rút gọn Entropy Shannon.

Trong [23], Jiye Liang và các cộng sự đã đưa ra một định nghĩa mới về entropy, chúng tôi gọi là entropy Liang.

Định nghĩa 3. [23] Cho bảng quyết định $DS=\left(U,C\cup D\right)$. Giả sử $U/C=\{C_1,C_2,....,C_m\}$, $U/D=\{D_1,D_2,...,D_n\}$. Entropy Liang có điều kiện của D khi đã biết C được định nghĩa bởi

$$E(D|C)=\sum_{i=1}^{n}\sum_{j=1}^{m}\frac{\left|D_i\cap C_j\right|}{\left|U\right|}\frac{\left|D_i^c-C_j^c\right|}{\left|U\right|}$$

với $D_i^c=U-D_i, C_j^c=U-C_j$

Dựa trên entropy Liang có điều kiện, Luo Ping và các cộng sự [26] định nghĩa tập rút gọn của bảng quyết định.

Định nghĩa 4. [26] Cho bảng quyết định $DS=\left(U,C\cup D\right)$ và tập thuộc tính $R\subseteq C$. Nếu

1) $E\left(D\middle|R\right)=E\left(D\middle|C\right)$.

2) $\forall r\in R, E(D\middle|\left(R-\{r\}\right))\neq E(D\middle|C)$.

thì R là một rút gọn của C dựa trên entropy Liang có điều kiện, gọi tắt là *tập rút gọn Entropy Liang*. Kí hiệu $ERED\left(C\right)$ là họ tất cả các tập rút gọn Entropy Liang.

Ngoài ba định nghĩa tập rút gọn nêu trên, một số định nghĩa khác về tập rút gọn cũng được một số tác giả đề xuất.

Thông thường, mỗi phương pháp rút gọn thuộc tính đều đưa ra định nghĩa tập rút gọn của phương pháp. Trong bảng quyết định nhất quán, các tập rút gọn là như nhau. Trong bảng quyết định không nhất quán, chúng ta có kết quả sau:

Mối liên hệ giữa ba tập rút gọn này là: Nếu R_E là một *tập rút gọn Entropy Liang* thì tồn tại một *tập rút gọn Entropy Shannon* R_H và một *tập rút gọn miền dương* R_P sao cho $R_P\subseteq R_H\subseteq R_E$.

Trong các bài toán thực tế, bảng quyết định thường chứa các đối tượng không nhất quán (là các đối tượng bằng nhau trên tập thuộc tính điều kiện nhưng khác nhau trên tập thuộc tính

quyết định). Tuy nhiên, tùy thuộc vào lớp bài toán cần giải quyết mà ta có thể chuyển bảng quyết định không nhất quán về bảng quyết định nhất quán qua bước tiền xử lí số liệu bằng cách loại bỏ các đối tượng không nhất quán.

Như đã trình bày trong mục trên, bảng quyết định $DS=\left(U,C\cup\left\{d\right\},V,f\right)$ là nhất quán khi và chỉ khi phụ thuộc hàm $C\to\left\{d\right\}$ đúng và *B* là một tập rút gọn của *C* nếu *B* là tập tối thiểu thỏa mãn phụ thuộc hàm $B\to\left\{d\right\}$. Trong cơ sở dữ liệu quan hệ, với quan hệ *r* trên tập thuộc tính *R* thì B là một tập tối thiểu của thuộc tính $d\in R, d\notin B$ nếu *B* là tập tối thiểu thỏa mãn phụ thuộc hàm $B\to\left\{d\right\}$ [17]. Do đó, khái niệm tập rút gọn của bảng quyết định tương đương với khái niệm tập tối thiểu của thuộc tính $\left\{d\right\}$ trên quan hệ.

Với các bảng quyết định nhất quán, chúng tôi trình bày một số thuật toán liên quan đến tập rút gọn sử dụng một số thuật toán và một số kết quả liên quan đến tập tối thiểu của một thuộc tính trong cơ sở dữ liệu quan hệ.

Bảng quyết định trong các bài toán thực tế thường chứa một số thuộc tính dư thừa thực sự, 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.

Như đã trình bày, trong bảng quyết định *thuộc tính dư thừa thực sự* là thuộc tính không xuất hiện trong bất kì 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 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 đề xuất một thuật toán tìm tập tất cả các thuộc tính rút gọn của bảng quyết định nhất quán có độ phức tạp thời gian là đa thức.

Trong cơ sở dữ liệu quan hệ, chúng tôi [16] đã chứng minh bổ đề quan trọng sau.

Bổ đề 1. [16] *Giả sử* K *là một hệ Sperner trên R, khi đó* $\bigcup_{K\in K}K=R-\bigcap_{K\in K^{-1}}K$.

Trên quan hệ *r*, do 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 $a\in R$, *khi đó*

$$\bigcup_{K\in K_a^r}K=R-\bigcap_{K\in\left(K_a^r\right)^{-1}}K$$

Cho bảng quyết định nhất quán $DS=\left(U,C\cup\left\{d\right\},V,f\right)$ với $U=\left\{u_1,u_2,...,u_m\right\}$. Xét quan hệ $r=\left\{u_1,u_2,...,u_m\right\}$ trên tập thuộc tính $R=C\cup\left\{d\right\}$, từ khái niệm *tập rút gọn* của bảng quyết định nhất quán và *tập tối thiểu của một thuộc tính* trên quan hệ ta có

$PRED(C) = K_d^r - \{d\}$, với $PRED(C)$ là họ tất cả các tập rút gọn Pawlak của C trong DS và K_d^r là họ các tập tối thiểu của thuộc tính 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. 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 $E_r = \{E_{ij} : 1 \le i < j \le m\}$ với $E_{ij} = \{a \in R : a(u_i) = a(u_j)\}$.

Bước 2. Từ E_r ta xây dựng tập $M_d = \{A \in E_r : d \notin A \not\exists B \in E_r : d \notin B, A \subset B\}$.

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 là tập tất cả các thuộc tính rút gọn của C.

Chứng minh

Theo cách xây dựng M_d tại *Bước 2* và theo công thức tính bao đóng của tập thuộc tính trên quan hệ, $\forall A \in M_d$ 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 $M_d = MAX(F^+, d)$ với $MAX(F^+, d) = \{A \subseteq R : A \to \{d\} \notin F^+, A \subset B \Rightarrow B \to \{d\} \in F^+\}$. Theo [17], $MAX(F^+, d) = (K_d^r)^{-1}$ với K_d^r là họ các tập tối thiểu của thuộc tính d trên quan hệ r. Do đó $M_d = (K_d^r)^{-1}$. Tại *Bước 3* kết hợp với *Bổ đề 5.2* ta có

$$V = R - \bigcap_{K \in M_d} K = R - \bigcap_{K \in (K_d^r)^{-1}} K = \bigcup_{K \in K_d^r} K.$$

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 1

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

Từ thuật toán 1 ta thu được các hệ quả sau:

Hệ quả 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.*

Về mặt lí thuyết, trong nhiều trường hợp chúng ta cần tìm tất cả các tập rút gọn trong bảng quyết định nhất quán. Chúng tôi đã xây dựng ba thuật toán sau:

- *Thuật toán* tìm họ tất cả các tập rút gọn của bảng quyết định nhất quán với độ phức tạp thời gian là hàm mũ.

- *Thuật toán* xây dựng các phụ thuộc hàm từ bảng quyết định nhất quán với độ phức tạp thời gian là hàm mũ. Ý nghĩa của thuật toán này là xây dựng một *công cụ hình thức* để biểu diễn tất cả các quyết định dưới dạng các phụ thuộc hàm từ bảng quyết định nhất quán cho trước, *không quan tâm đến dữ liệu cụ thể*.

- *Thuật toán* xây dựng bảng quyết định thỏa mãn tập phụ thuộc hàm cho trước với độ phức tạp thời gian là hàm mũ. Ý nghĩa của thuật toán này là khẳng định tính đúng đắn của việc suy diễn và ra quyết định trên các phụ thuộc hàm. Nghĩa là có thể thực hiện suy diễn và ra quyết định trên các tri thức dưới được biểu diễn dưới dạng phụ thuộc hàm mà không quan tâm đến bảng dữ liệu cụ thể.

Các kết quả này đã được công bố trong [5,33].

Lời cám ơn. Tôi xin chân thành cám ơn Ban Biên tập Tạp chí Khoa học và Công nghệ đã mời tôi viết bài báo này. Do khuôn khổ có hạn của bài báo, tôi chỉ có thể trình bày một phần những kết quả đã đạt được trong lĩnh vực cơ sở dữ liệu và khai phá dữ liệu. Một số kết quả trình bày trong bài báo này là những kết quả làm việc của các nghiên cứu sinh của tôi, TS. Nguyễn Long Giang, TS Nguyễn Huy Đức, TS Nguyễn Hoàng Sơn trong nhóm nghiên cứu "Cơ sở dữ liệu và khai phá dữ liệu " của Viện Công nghệ thông tin.

TÀI LIỆU THAM KHẢO

1. Agrawal R., Imielinski T., Swami A. - Mining association rules between sets of items in large databases, Proceedings of the ACM SIGMOD conference, Washington DC, USA, 1993, pp. 207-216.
2. Tao F., Murtagh F., Farid M. - Weighed Association Rule Mining Using Weighted Support and sighificance Framework, SIGKDD,2003, pp. 61-666.

3. Khan M. S., Muyeba M., Coenen F. - A weighted utility framework for mining association rule, Proc. IEEE European Modeling Symporium 2008, 2008, pp. 87-92.
4. Nguyễn Long Giang, Vũ Đức Thi - Một số phương pháp rút gọn thuộc tính trong bảng quyết định dựa trên ENTROPY cải tiến, Tạp chí Tin học và Điều khiển học **27** (2) (2011) 166-175.
5. Nguyễn Long Giang, Vũ Đức Thi - Thuật toán tìm tất cả các rút gọn trong bảng quyết định, Tạp chí Tin học và Điều khiển học **27** (3) (2011) 211-218.
6. Vũ Đức Thi, Nguyễn Huy Đức - Một số kỹ thuật hiệu quả tỉa các tập mục ứng viên trong khai phá tập mục lợi ích cao, Kỉ yếu Hội thảo quốc gia "Một số vấn đề chọn lọc của Công nghệ thông tin và truyền thông – Biên Hòa, 2010, tr. 214-232.
7. Armstrong W. W. - Dependency Structures of Database Relationships, Information Processing 74, Holland publ. Co. 1974, pp. 580-583.
8. Vũ Đức Thi - Cơ sở dữ liệu - Kiến thức và thực hành, Nhà xuất bản Thống kê, Hà Nội, 1997.
9. Chen C. H., Hong T. P., Tseng V. S. - An improved approach to find membership funtions and multiple minimum supports in fuzzy data mining, Expert Systems with Application **36** (2009) 10016-10024.
10. Chen H., Li T., Qiao S., Ruan D. - A Rough set based dynamic maintenance approach for approximations in coarsening an refining attribue values, International Journal of intelligent systems **25** (2010) 1005-1026.
11. Erwin A., Gopalan R. P., and Achuthan N. R. - CTU-Mine: An Efficient High Utility Itemset Mining Algorithm Using the Pattern Growth Approach, Paper presented at the *IEEE 7th International Conferences on Computer and Information Technology*, Aizu Wakamatsu, Japan.
12. El-Hajj M. and Zaiane Osmar R. - Inverted matrix: Efficient discovery of frequent items in large datasets in the context of interactive mining, In *Proc. 2003 Int'l Conf. on Data* Mining and Knowledge Discovery (ACM SIGKDD) 2003, 2007, pp. 109-118.
13. Demetrovics J. and Thi V. D. - Some remarks on generating Armstrong and inferring functional dependencies relation, Acta Cybernetica **12** (2003) 167-180.
14. Demetrovics J, Thi V. D. - Some computational problems related to the functional dependency in the relational datamodel, Journal Acta Scien. Mathematics, Hungary, **57** (1993) 627-638.
15. Demetrovics J, Thi V. D. - Some results about normal forms for functional dependency in the relational datamodel,. Journal of Discrete Applied Mathematics, North Holland **69** (1996) 61-74.
16. Demetrovics J, Thi V. D. - Describing Candidate Keys by Hypergraphs, J. Computers and Artificial Intelligence **18**(2) (1999) 191-207.
17. Demetrovics J, Thi V. D. - Some Computational problems related to Boyce-Codd nornal form , Ann. Univ. Sci. Budapest. Sect. Comput. Hungary **19** (2000) 19-130.
18. Han J., Pei J., and Yin Y. - Mining frequent patterns without candidate generation, In ACM SIGMOD Intl. Conference on Management of Data, 2003, pp. 1-12.
19. Hilderman R. J., Carter C. L., Hamilton H. J., and Cercone N. - Mining association rules from market basket data using share measures and characterized itemsets, Intl. Journal of Artificial Intelligence Tools **7** (2003) 189-220.

20. Vũ Đức Thi - Giáo trình Cơ sở dữ liệu nâng cao, Nhà xuất bản Đại học Thái Nguyên, 2010.
21. Khan M. S., M. Muyeba M., Coenen F. - Fuzzy Weighted Association Rule Mining with Weighted Support and Confidence Framework, Proc. 1st In workshop on Algorithems for Large _ Scale Information Processing in knowledge Discovery (ALSIP 2008), Held in conjunction with PAKDD 2008 (Japan), 2008, pp. 52-64.
22. Li Y. C., Yeh J. S., Chang C. C. - A fast algorithm for mining share-frequent itemsets, Lecture Notes in Computer Science, Springer-Verlag, Germany 3399, 2005, pp. 417-428.
23. Liang J. Y, Chin K. S., Dang C. Y., Richard C. M. YAM - New method for measuring uncertainty and fuzziness in rough set theory, International Journal of General Systems **31** (2002) 331-342.
24. Liu D., Li T., Ruan D., Zou W. - An incremental approach for inducing knowledge from dynamic information systems, Funda. Inform. **94** (2009) 245-260.
25. Liu Y., W. Liao K., and Choudhary A. - A fast high utility itemsets mining algorithm", in Proc. 1st Intl. Conf. on Utility-Based Data Mining, Chicago Illinois, 2009, USA, pp. 90-99,.
26. Luo P., He Q. and Shi Z. Z. -Theoretical study on a new information entropy and its use in attribute reduction, ICCI (2005) pp. 73-79.
27. Pawlak Z. -Rough sets, International Journal of Computer and Information Sciences, **11** (5) (1982) 341-356.
28. Vũ Đức Thi - Thuật toán trong tin học, Nhà xuất bản Khoa học kỹ thuật, Hà Nội, 1999.
29. Thi V. D. Sơn N. H. - Some problems related to keys and Boyce-Codd normal form, Acta Cybernet, Hungary **16** (3) (2004) 473-483.
30. Thi V.D., Son N.H. - Some results related to dense families of database relations. Acta Cybernet, Hungary **17** (1) (2005) 173-182.
31. Thi V. D., Son N. H. - On Armstrong relations for strong dependencies. Acta Cybernet, Hungary **17** (3) (2006) 521-531.
32. Thi V. D., Duc N. H. - Mining High Utility Itemsets in Massive Transactional Database, Acta Cybernetica **20** (2011) 331-346.
33. Thi V. D., Giang N. L. - A Method to Construct a Decision Table from a relation scheme, Jounal of Cybernetics and Information Technology, Bugarian Academy of Sciences **3** (2011) 32-41.
34. Yao H., Hamilton H. J. - Mining Itemsets Utilities from Transaction Databases, Data and Knowledge Engeneering **59** (3) (2006).
35. Yao H., Hamilton H. J., Geng L. - A Unified Framework for Utility Based Measures for Mining Itemsets, UBDM'06 Philadelphia, Pennsylvania, USA, 2006.
36. Wang G. Y. - Algebra view and information view of rough sets theory, In*:* Dasarathy BV,editor. Data mining and knowledge discovery: Theory, tools, and technology III, Proceedings of SPIE, 2001, pp. 200-207.

ABSTRACT

SOME COMPUTATIONAL PROBLEMS RELATED TO DATABASE AND DATA MINING

Vu Duc Thi

Institute of Information Technology, VAST, 18 Hoang Quoc Viet, Cau Giay, Hanoi, Vietnam

Email: *vdthi@ioit.ac.vn*

Database and data mining are very important development in information technology (IT). In essence the data play a fundamental role in the processing of information on computer systems. Database theory and the practical applications of this theory have been developed and achieved many accomplishments since the 80th of last century. Essentially, database theory provides us with the most important knowledge related to organizational issues, design and construction of the database management system. On the basis of the results obtained in this theory, computer companies such as IBM, Microsoft, Oracle, Apple has built the database management system trade all over the world market requirements, such as SQL, Oracle, IBM DB2. In some aspects, the present, in all activities of mankind has accumulated a huge amount of data. However, knowledge is too small. Therefore, the current research directions for knowledge discovery from data is a very powerful development. A particularly critical stage in the process of knowledge discovery from data is data mining to acquire knowledge. Hence, research on data mining methods is a very basic directions in IT. In this paper, we present some main results related to the computational problems, in fact algorithmic problems, in the field of database and data mining.

Keywords: database, data mining, database management system, knowledge discovery from data, computational problem, algorithm.