**MINING ERASABLE ITEMSETS WITH CONSTRAINTS**

Nguyễn Hà Giang và Võ Đình Bảy

Khoa Công nghệ Thông tin, Trường Đại học Công nghệ Tp. HCM (HUTECH)

**Đã bổ sung**

**TÓM TẮT**

Khai thác tập có thể xóa là một phân nhánh của khai thác dữ liệu, tập có thể xóa hỗ trợ các nhà quản lý sản xuất lập ra kết hoạch cho việc tạo ra các sản phẩm mới. Các công việc khai thác liên quan đến tập có thể xóa được bắt đầu nghiên cứu từ năm 2009 do Deng và các cộng sự của ông đề xuất. Từ đó cho đến nay đã có rất nhiều các công trình nghiên cứu của các nhóm tác giả khác nhau nhằm phát triển các thuật toán để nâng cao quá trình khai thác tập dữ liệu có thể xóa này. Các thuật toán thuật toán thường mất rất nhiều thời gian và bộ nhớ để khai thác. Điều này ảnh hưởng đến các hệ thống hỗ trợ kế hoạch sản xuất. Trong phần nghiên cứu này, chúng tôi phát triển thuật toán khai thác dữ liệu có thể xóa dựa trên ràng buộc. Kết quả của giải thuật này sẽ tập trung trên các dữ liệu quan tâm, không khai thác dàn trải trên tất cả tập dữ liệu. Do vậy quá trình khai phá sẽ hiệu quả hơn.

**ABSTRACT**

Mining erasable itemsets is a variant of data mining, erasable itemsets support production managers establish plans for the creation of new products. The first research about mining erasable itemset was published in 2009 by Deng et al. Since then there have been a lot of research works of different authors to develop advanced algorithms to mining erasable itemsets. The algorithm usually takes a lot of time and memory to mine. This affects systems support production planning. In this study, we developed an algorithm for mining erasable itemset with constrainsts. The result of this algorithm will focus on certain itemsets, not mine spread across all itemsets. Thus the process of mining more effective.

1. **TỔNG QUAN**

Khám phá tri thức và khai thác dữ liệu là quá trình khám phá những tri thức tiềm ẩn, thú vị tồn tại trong một tập dữ liệu lớn. Quá trình này kết hợp này nhiều phương pháp như trí tuệ nhân tạo, học máy, thống kê, v.v. Nhiều vấn đề trong khai thác dữ liệu đã thu hút các nhà khoa học quan tâm như khai thác luật kết hợp [8], ứng dụng của luật kết hợp [7], phân lớp [1], v.v. Khai thác tập mục là phương pháp nền tảng để giải quyết các bài toán trên [9]. Rất nhiều phương pháp để khai thác tập phổ biến được đề xuất như thuật toán Apriori [8], thuật toán FP-growth [5], Eclat [2].

Năm 2009, Deng cùng các đồng sự [12] đã đưa ra khái niệm tập có thể xóa (EI - erasable itemset) và bài toán khai thác EI. Vấn đề này bắt nguồn từ việc lập kế hoạch sản xuất của một nhà máy sản xuất rất nhiều loại sản phẩm. Mỗi sản phẩm được tạo từ một số các danh mục và thu lợi cho nhà máy một khoản lợi nhuận. Để sản xuất tất cả các sản phẩm, nhà máy phải mua và lưu trữ toàn bộ danh mục. Trong tình hình khủng hoảng kinh tế, nhà máy không còn khả năng về tài chính để mua và lưu trữ toàn bộ danh mục như thường lệ; do đó, những người quản lý phải cân nhắc sửa đổi kế hoạch sản xuất để đảm bảo sự ổn định của nhà máy. Vấn đề khai thác EI là tìm toàn bộ tập danh mục có thể loại bỏ mà không ảnh hưởng lớn đến lợi nhuận của nhà máy. Bài toán này được sử dụng gợi ý cho những người quản lý tạo ra các kế hoạch sản xuất mới thích hợp hơn.

Một ứng dụng khác của bài toán khai thác EI, giả sử một nhà máy sản xuất một sốsản phẩm khác nhau. Những người quản lý trong quá trình nghiên cứu tìm thấy được tìm năng phát triển của một sản phẩm mới; tuy nhiên, để phát triển sản phẩm mới này cần một số vốn nhất định, và nhà máy lại không muốn mở rộng qui mô sản xuất hiện tại. Trong trường hợp này, những người quản lý có thể sử dụng bài toán khai thác EI, sau đó, thay thế các EI bằng một số danh mục mới để sản xuất sản phẩm mới mà vẫn kiểm soát được lợi nhuận của nhà máy. Với bài toán khai thác EI này, những người quản lý có thể tìm ra những sản phẩm có lợi nhuận tốt nhất cho nhà máy mà không làm ảnh hưởng tới sự ổn định của nhà máy.

Mặc dù đã có nhiều thuật toán được phát triển gần đây cho bài toán khai thác tập có thể xóa nhưng việc khai thác tập có thể xóa có ràng buộc chưa được phát triển. Chẳng hạn, làm thế nào để khai thác các tập có chứa tập *X*? Trong bài báo này, chúng tôi phát triển một phương pháp hiệu quả để khai thác tập có thể xóa dựa trên ràng buộc dạng *Y* = {*X* ⊆ *I* | *X* là tập có thể xóa và *X* ⊇ *C*} trong đó *I* là tập tất cả các mục có trong cơ sở dữ liệu và *C* ⊆ *I*. Việc khai thác tập *Y* như trên có thể thực hiện đơn giản bằng cách khai thác tất cả các tập có thể xóa, sau đó lọc ra các tập thỏa ràng buộc. Tuy nhiên, cách tiếp cận này không hiệu quả cả về thời gian khai thác lẫn bộ nhớ lưu trữ. Vì vậy, chúng tôi đề xuất một giải pháp khai thác tập *Y* không cần kiểm tra ràng buộc.

1. **CÁC NGHIÊN CỨU LIÊN QUAN**

Cho *I* = {*i*1, *i*2,..., *im*} là một tập hợp toàn bộ các danh mục đại diện cho các thành phần của sản phẩm. Một cơ sở dữ liệu sản phẩm ký hiệu bằng *DB* = {*P*1, *P*2,..., *Pn*}, trong đó mỗi *Pi* (1≤ *i* ≤ *n*) là một sản phẩm. Mỗi sản phẩm được biểu diễn bằng 〈*Items*, *Val*〉, trong đó *Items* là toàn bộ danh mục tạo nên sản phẩm *Pi* và *Val* là lợi nhuận thu được khi kinh doanh sản phẩm *Pi*. Một tập *X* ⊆ *I* được gọi là một tập danh mục, và một tập danh mục với *k* danh mục thì được gọi là một *k*-danh-mục.

Dữ liệu ở Bảng 2.1 (*DBe*) được sử dụng làm ví dụ trong suốt luận văn trong đó {*a, b, c, d, e, f, g, h*} là tập hợp các danh mục tạo ra các sản phẩm {*P*1, *P*2,..., *P*11}.

#### Bảng 2.1: Dữ liệu ví dụ (*DBe*)

|  |  |  |
| --- | --- | --- |
| **Sản phẩm** | ***Items*** | ***Val* (Đô-la)** |
| *P*1 | *a*, *b*, *c* | 2,100 |
| *P*2 | *a*, *b* | 1,000 |
| *P*3 | *a*, *c* | 1,000 |
| *P*4 | *b*, *c*, *e* | 150 |
| *P*5 | *b*, *e* | 50 |
| *P*6 | *c*, *e* | 100 |
| *P*7 | *c*, *d*, *e*, *f*, *g* | 200 |
| *P*8 | *d*, *e*, *f*, *h* | 100 |
| *P*9 | *d*, *f* | 50 |
| *P*10 | *b*, *f*, *h* | 150 |
| *P*11 | *c*, *f* | 100 |

Để hiểu rõ bài toán khai thác EI, hai định nghĩa cơ bản trong [11] được trình bày lại như sau:

**Định nghĩa 2.1.** Gọi *X* (⊆ *I*) là một tập danh mục. Lợi nhuận của *X* được tính bằng công thức sau:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

Lợi nhuận của tập danh mục *X* là tổng lợi nhuận của những sản phẩm chứa ít nhất một danh mục trong *X*.

Ví dụ gọi *X* = {*ac*} là một tập danh mục. Ta có {*P*1, *P*2, *P*3, *P*4, *P*6, *P*7, *P*11} là các sản phẩm chứa {*a*}, {*c*}, hoặc {*ac*}. Do đó, (*X*) = *P*1*.Val* + *P*2.*Val* + *P*3.*Val* + *P*4.*Val* + *P*6.*Val* + *P*7*.Val* + *P*11*.Val* = 4.650 đô-la.

**Định nghĩa 2.2.** Cho trước ngưỡng *ξ* và cơ sở dữ liệu *DBe*, gọi *T* là tổng lợi nhuận của nhà máy. Một tập danh mục *X* là EI nếu:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

trong đó *T* được tính bằng công thức sau:

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

Tổng lợi nhuận của nhà máy là tổng lợi nhuận của toàn bộ sản phẩm. Xét *DBe*, ta có *T* = 5.000 đô-la. Một tập danh mục *X* được gọi là EI nếu (*X*) ≤ *T* × *ξ*.

Ví dụ, cho *ξ* = 16%, theo Định nghĩa 2.1, (*e*) = 600 đô-la và *e* được gọi là một EI vì (*e*) = 600 ≤ 5.000 × 16% = 800. Điều này có nghĩa là nhà máy không cần mua và lưu giữ *e*. Trong tình huống này, nhà máy không sản xuất các sản phẩm *P*4, *P*5, *P*6, *P*7, và *P*8, nhưng vẫn duy trì được lợi nhuận (lớn hơn hoặc bằng 5.000 × 16% = 4.000 đô-la).

Từ Định nghĩa 2.1 và 2.2, bài toán khai thác EI là tìm toàn bộ tập danh mục có (*X*) nhỏ hơn hoặc bằng *T* × *ξ*.

Hiện tại, có ba hướng được tập trung nghiên cứu bao gồm:

1. Khai thác tập có thể xóa: gồm các thuật toán đã được phát triển như: META [12], VME [13], MERIT [14], dMERIT+[10], và MEI [11]. Trong đó MEI là thuật toán được đánh giá là thuật toán hiệu quả nhất cho bài toán khai thác tập EI.
2. Khai thác Top-rank-k EI: gồm các thuât toán VM [15], và dVM [3]. Trong đó thuật toán dVM được đánh giá hiệu quả hơn VM do sử dụng cấu trúc dPID\_List để tính nhanh giá trị gain của tập itemset.
3. Khai thác tâp EI đóng: có một thuật toán đã phát triển để khai thác tập dữ liệu có thể xóa đóng là MECP [4].
4. **KHAI PHÁ TẬP DỮ LIỆU CÓ THỂ XÓA VỚI RÀNG BUỘC**

## Định nghĩa

Trong phần 2 đã định nghĩa các khái niệm liên quan đến khai thác tập dữ liệu có thể xóa. Việc khai thác tập dữ liệu có thể xóa với ràng buộc có thể được định nghĩa dựa trên các khái niệm trên.

**Định nghĩa 3.1**. Cho trước tập *C*, gọi là tập ràng buộc, trong đó *C* ⊆ *I*. Việc khai phá tập có thể xóa với ràng buộc *C* là quá trình tìm tất cả các tập danh mục *X*, với:

|  |  |  |
| --- | --- | --- |
|  | *Y* = {*X* ⊆ *I |*  ∧ *C* ⊆ *X*} | (4) |

Ví dụ: Xét *DBe*, ta có *T* = 5.000 đô-la, cho *ξ* = 16%, giả sử cần khai thác tập có thể xóa thỏa ràng buộc *C*, trong đó *C* = {*ed*}. Do tập *C* có 2 itemset nên ta chỉ xét các tập ràng buộc *C* có 2 phần tử, trong trường hợp này chỉ có tập {*ed*}, dựa trên định nghĩa 2.2, tính được *g*({*ed*}) = P4.Val+P5.Val+P6.Val+P7.Val+P8.Val+P9.Val = 150 + 50 + 100 + 200 + 100 + 50 = 650 < *T* × *ξ* = 800.

Với tập 3 itemset, ta xét {*edh*} và {*edg*}, trong đo *g*({*edh*}) = 800, *g*({*edg*}) = 650. Do cả hai *g*({*edh*}) và *g*({*edg*}) đều nhỏ hơn 800, nên là các tập có thể xóa.

Với tập 4 itemset chỉ có duy nhất 1 tập {*edhg*}, ta tính được *g*({*edhg*}) là 800, do vậy {*edhg*} là tập có thể xóa.

Do vậy, tập dữ liệu có thể xóa với ràng buộc *C*={*ed*} trong *DBe* là: {*ed*}, {*edh*}, {*edg*}, và {*edhg*}.

## Cách tiếp cận khai thác tập dữ liệu có thể xóa với ràng buộc.

Dựa trên MEI [11] có thể đề xuất 3 mức khai thác tập dữ liệu có thể xóa dựa trên ràng buộc như sau:

1. Áp dụng MEI để khai thác tất cả các tập dữ liệu có thể xóa với ngưỡng *ξ*. Sau đó lọc ra các EI thỏa điều kiện ràng buộc *C*.
2. Trong quá trình áp dụng MEI để khai thác EI, khi một tập có thể xóa được tạo ra, chúng ta kiểm tra nó có thỏa ràng buộc hay không? Nếu thỏa thì bổ sung vào danh sách các itemset thỏa ràng buộc.
3. Cải tiến thuật toán MEI để khai tác tập dữ liệu có thể xóa dựa trên ràng buộc ngay từ ban đầu. Thuật toán chỉ tập trung mở rộng với các itemset thỏa điều kiện ràng buộc.

Trong bài báo này, chúng tôi đề xuất áp dụng mức 3 để khai thác. Trong cách tiếp cận này, sẽ dựa trên MEI, tuy nhiên chỉ tập trung trong việc mở rộng các itemset đã thỏa điều kiện ràng buộc cho trước.

## Thuật toán CMEI

Thuật toán sử dụng chiến thuật chia-để-trị, cấu trúc dPidset và các định lý liên quan để khai thác EI với ràng buộc *C*. Đầu tiên, thuật toán xuất phát với tập ràng buộc *C* cho trước, trong đó *C* là tập k-itemset với k = |*C*|, kết hợp với tập EI1 \ *C* để tạo ra các tập (k+1)-EI tương tự MEI. Cụ thể, với những phần tử tạo ra mà có lợi nhuận nhỏ hơn *T* × *ξ*, thuật toán sẽ: (i) thêm chúng vào kết quả; (ii) kết hợp chúng với nhau để tạo ra EI*k*+2. Sau bước này ta có tập EI*k*+2, ở bước tiếp theo sẽ lấy phần tử đầu tiên trong tập này kết hợp với các phần tử còn lại để tạo tập k+3 itemset ứng viên, v.v. Quá trình Được thực hiện tương tự cho tất cả những lớp tương đương khác được tạo ra từ tập EI*k*+2.

Thuật toán CMEI được mô tả như sau:

Input: tập *DBE*, ngưỡng *ξ* và tập ràng buộc *C*

Output: tập *EIC*, tập dữ liệu có thể xóa thỏa ràng buộc *C*.

1. Duyệt *DBE* để xác định *T, G*, và *EI’1 (EI’1 = EI1\C)* với các pidset tương ứng. Tính *g(C),* nếu *g(C)* > *T**ξ*. Kết thúc thuật toán với *EInext* là rỗng.
2. Sắp xếp *EI’1* theo thứ tự giảm dần của kích thước pidset.
3. *EInext*
4. for k 1 to |*|* - 1 do
5. *E*.items = *C* ∪ [k].Item
6. (.pidset, Gain)
7. Sub\_pidsets(.pidset,[k].pidset)
8. *E*.gain = *C*.gain + Gain
9. if .gain ≤ *T**ξ* then
10. *EInext*
11. *EIC*
12. if |*EInext* |>1
13. call Expand\_E(*EInext*)

procedure Expand\_E(*EIv*)

1. for k ← 0 to |*EIv*| - 2 do
2. *EInext*
3. for j (k+1) to - 1 do
4. .Items = [k].Items ∪ [j].Items
5. (.pidset, Gain)
6. Sub\_pidsets([k].pidset, [j].pidset)
7. *E*.gain = [k].gain + Gain
8. if .gain ≤ T*ξ* then
9. *EInext*
10. *EIC*
11. if |*EInext* |>1
12. call Expand\_E(*EInext*)

Trong giải thuật CMEI có dùng thủ tục Sub\_pidsets được mô tả trong MEI [11].

## Minh họa

Quá trình thực thi của thuật toán CMEI trên toán trên *DBe* với *ξ* = 16% và *C* là {*ed*} được mô tả qua các bước chính sau:

1. CMEI duyệt *DBe* để xác định *T* = 5.000 đô-la; mảng index of gain, ; và *EI’1* = { *f*, *g*, *h*} kèm các pidset của nó. Do *g(C)* = 800 nên *C* là itemset có thể xóa.
2. *EI’1* theo thứ tự giảm dần của lượng lượng pidset của chúng. Sau bước sắp xếp này, thứ tự mới của *EI’1* là {*f*, *g, h*};
3. CMEI kết hợp *C* với từng item trong *EI’1* để tạo các itemset ứng viên như: {*edf*},{*edg*}, và {*edh*}. Do chỉ có {*edg*}, và {*edh*} thỏa điều kiện là EI nên được đưa vào tập kết quả và tập để mở rộng ở bước tiếp theo.
4. CMEI sử dụng thủ tục Expand\_E để mở rộng ở các mức lớn hơn. Quá trình kết thúc khi không thể mở rộng thêm.

Hình 4.1 minh họa cây được xây dựng từ thuật toán CMEI.

{}

***ed***×456789

650

*f*×7891011

600

*g*×7

200

*h*×810

250

*edf*×1011

900

***edg***×∅

650

***edh***×10

800

***edgh***×10

800

Hình 4.1. Cây minh họa CMEI.

Để có được cây như hình 4.1, thì tập {*ed*} ban đầu được kết hợp với *EI’1* là {*f, g, h*} như hình 4.2.

{}

***ed***×456789

650

*f*×7891011

600

*g*×7

200

*h*×810

250

Hình 4.2: Kết hợp tập ràng buộc với *EI’1.*

Mỗi nút trên cây theo hình 4.2 gồm tập itemset, tập PID, và tổng lợi nhuận của itemset. Tập {*ed*} sẽ kết hợp với ba itemset còn lại {*f*},{*g*}, {*h*} để tạo ra các itemset như hình 4.3.

{}

***ed***×456789

650

*f*×7891011

600

*g*×7

200

*h*×810

250

*edf*×1011

900

***edg***×∅

650

***edh***×10

800

Hình 4.3: Các itemset được kết hợp.

Mỗi nút ở cấp thứ 2, gồm thông tin tập itemset, tập dPidset của itemset và tổng giá trị lợi nhuận của itemset. Do {*edf*} có giá trị lợi nhuận lớn hơn *T* × *ξ*, nên bị loại, chỉ còn 2 itemset {*edg*} và {*edh*} kết hợp để tạo ra nút cuối cùng như hình 4.4.

{}

***ed***×456789

650

*f*×7891011

600

*g*×7

200

*h*×810

250

*edf*×1011

900

***edg***×∅

650

***edh***×10

800

***edgh***×10

800

Hình 4.4: Cây phát sinh ở bước sau cùng.

Kết quả tập dữ liệu có thể xóa với ràng buộc {*ed*} trong tập dữ liệu *DBe*. là các tập {*ed*: 650}, {*edg*:650}, {*edh*:800} và {*edgh*:800}.

## Đánh giá giải thuật CMEI

Trong bài báo này, chúng tôi đã phát triển một phương pháp khai thác tất cả các tập có thể xóa không thỏa ràng buộc mà không cần kiểm tra ràng buộc do tận dụng quan hệ thứ tự trên tập 2*I*. Ngoài ra, CMEI tận dụng tất cả các ưu điểm của MEI để khai thác hiệu quả EI. Với cách tiếp cận kể trên, CMEI tỉa sớm đáng kể không gian tìm kiếm trên cây tìm kiếm và vì vậy, nó giảm đáng kể thời gian khai thác cũng như không gian lưu trữ so với phương pháp khai thác dựa trên MEI rồi lọc các tập có thể xóa thỏa ràng buộc.

1. **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

Bài báo đề xuất một phương pháp khai thác tập có thể xóa thỏa ràng buộc *C* cho trước. Chúng tôi cũng đã đề nghị một thuật toán khai thác tập có thể xóa thỏa ràng buộc mà không cần kiểm tra ràng buộc dựa trên quan hệ thứ tự của các tập. Cách tiếp cận này được đánh giá hiệu quả so với việc khai thác toàn bộ tập EI rồi chọn ra các tập EI với ràng buộc.

Trong tương lai, chúng tôi sẽ tiến hành cài đặt thuật toán và so sánh với cách tiếp cận khai thác tất cả các tập có thể xóa sau đó lọc các tập thỏa ràng buộc. Ngoài ra, chúng tôi cũng sẽ mở rộng sang các loại ràng buộc khác. Bên cạnh đó, chúng tôi cũng sẽ mở rộng sang khai thác tập có thể xóa đóng thỏa ràng buộc.

**TÀI LIỆU THAM KHẢO**

1. B. Liu, W. Hsu, Y. Ma: Integrating classification and association rule mining. In SIGKDD’98, (1998), pp. 80–86.
2. B. Vo, T.P. Hong, B. Le: DBV-Miner: A dynamic bit-vector approach for fast mining frequent closed itemsets. Expert Systems with Applications, 39(8), (2012), pp. 7196-7206.
3. G. Nguyen, T. Le, B. Vo, B. Le: A new approach for mining top-rank-k erasable itemsets. In ACIIDS’14, (2014), Bangkok, Thailand, pp. 73–82.
4. G. Nguyen, T. Le, B. Vo, B. Le: Discovering erasable closed patterns. In ACIIDS’15, Bali, Indonesia, (2015) pp. 368-376.
5. J. Han, J. Pei, Y. Yin: Mining frequent patterns without candidate generation. In SIGMOD’00, (2000), pp. 1–12.
6. M. Zaki, K. Gouda: Fast vertical mining using diffsets. In SIGKDD’03, (2003), pp. 326–335.
7. M.J. Abdi , D. Giveki: Automatic detection of erythemato-squamous diseases using PSO-SVM based on association rules. Engineering Applications of Artificial Intelligence, 26(1), (2013), pp. 603-608.
8. R. Agrawal, R. Srikant: Fast algorithms for mining association rules. In VLDB’94, (1994), pp. 487–499.
9. R. Agrawal, T. [Imielinski](http://dl.acm.org/author_page.cfm?id=81100164746&coll=DL&dl=GUIDE&CFID=85460486&CFTOKEN=14292318), A. [Swami](http://dl.acm.org/author_page.cfm?id=81100152152&coll=DL&dl=GUIDE&CFID=85460486&CFTOKEN=14292318). Database mining: A performance perspective. IEEE Transactions on Knowledge and Data Engineering, 5(6), (1993), pp. 914-925.
10. T. Le, B. Vo, F. Coenen: An efficient algorithm for mining erasable itemsets using the difference of NC-Sets. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, Manchester, UK, (2013), pp. 2270–2274.
11. T. Le, B. Vo: MEI: An efficient algorithm for mining erasable itemsets. Engineering Applications of Artificial Intelligence, 27, (2014), pp. 155–166.
12. Z.H. Deng, G. Fang, Z. Wang, X. Xu: Mining erasable itemsets. In ICMLC’09, (2009) pp. 67–73.
13. Z.H. Deng, X.R. Xu: An efficient algorithm for mining erasable itemsets. In ADMA’10, (2010), pp. 214–225.
14. Z.H. Deng, X.R. Xu: Fast mining erasable itemsets using NC\_sets. Expert Systems with Applications, 39(4), (2012), pp. 4453–4463.
15. Z.H. Deng: Mining top-rank-k erasable itemsets by PID\_lists. International Journal of Intelligent Systems, 28(4), (2013), pp. 366–379.