Số hồ sơ: .....

TÊN BÀI BÁO

Tác giả
co quan của tac gia, email,..

Tóm tắt:

(
Phần ghi chú của mình:

Bố cục của bài báo:

I. **Đặt vấn đề** (hay giới thiệu tổng quan):
Tình hình thực tế dẫn đến vấn đề trong bài báo,
Cách giải quyết của các tác giả có trước cho đến khi viết bài báo này,
Hướng giải quyết của tác giả,
Các ứng dụng có thể
II. **Cơ sở lý luận** (nhưng căn cứ lý thuyết của vấn đề):
III. **Phương pháp nghiên cứu** (và giải quyết vấn đề của tác giả)
IV. **Thực nghiệm và kết quả**
V. **Kết luận và khuyến nghị**
(bên dưới các mục lớn : I, II, III,.. là 1. 2. ,.. dưới 1, 2, 3, là a, b, .. nhớ đánh số trang)
Tài liệu tham khảo (có trích dẫn trong bài viết của tác giả: phải tuân theo quy định)
[1] ABC TEN (năm) *Tên tài liệu*. Nhà Xbản, trang số..

1 Vì đây không phải là báo cáo hay tài liệu diễn giải nên không phải nhắc **Định nghĩa**,... mà phải viết, chẳng hạn: "*Tập P có các đặc tính ... được gọi là tập Presssure set*" , và có thể nêu kèm ví dụ nếu thật sự cần.
Vd, trong định nghĩa 1:

Xét không gian các vector s = (s_1, s_2, …, s_m) là các vector nhị phân có m chiều (s_i = 0 hoặc s_i = 1 $\forall i$ = 1, …, m)

Ta nói vector s1 = ($s1_1$, $s1_2$, …, $s1_m$) *bao phủ* vector s2 = ($s2_1$, $s2_2$, …, $s2_m$) nếu $\forall s2_i$ =1 thì $s1_i$ = 1, i = 1, …, m

Ví dụ: Với s1 = (1, 1, 1, 0) và s2 = (0, 1, 1, 0) thì ta nói s1 *bao phủ* s2

Em nên viết:

Cho s = (s_1, s_2, …, s_m) là *vector nhị phân* m chiều có các thành phần là 0 hoặc 1. Lúc đó vector nhị phân s1 = ($s1_1$, $s1_2$, …, $s1_m$) *bao phủ* vector nhị phân s2 = ($s2_1$, $s2_2$, …, $s2_m$) nếu với mỗi $s2_i$ =1 thì cũng có $s1_i$ = 1, trong đó i $\in$ {1, …, m}. Chẳng hạn: vector s1 = (1, 1, 1, 0) bao phủ s2 = (0, 1, 1, 0).

Không cần đánh chỉ mục các định nghĩa, mệnh đề, định lý Mà chỉ đánh số, ví dụ: Định lý 1. Mệnh đề 2.,..
Các định nghĩa, mệnh đề,.. cần được gắn vào các tiểu mục, để sau náy tham chiếu: ví dụ: theo Mệnh đề 2 tại mục 2.a ta có …

2 Việc trình bày các ma trận 0,1 : nên co hẹp vùng giấy, chẳng hạn: dùng bảng

1	1	1	0		1	1	1	0
0	1	1	1		0	1	1	1

1	1	1	0		1	1	1	0
0	1	1	1		0	1	1	1
0	0	1	1		0	0	1	1

để trình bày Hình chữ nhật tối đại ((1,1,1,0);2), ((0,1,1,0);4)

3 Tránh tuyệt đối dùng đại từ "tôi, chúng tôi, .." mà phải thay bằng dạn g thụ động, chẳng hạn đoạn:

Ta có các vector nhị phân-m chiều u, v và z như sau:

$u = (u_1\ u_2 \dots u_m)$

$v = (v_1\ v_2 \dots v_m)$

$z = (z_1\ z_2 \dots z_m)$

Theo mệnh đề 1 ta đã biết 2 phần tử bất kỳ của tập P luôn khác nhau, suy ra $u \neq v$

Trước tiên, ta sẽ chứng minh phát biểu trên đúng khi u và v chỉ khác nhau tại 1 vị trí (còn m-1 vị trí còn lại giống nhau)

Nên viết lại như sau:

Theo mệnh đề 1, các vector nhị phân $u = (u_1\ u_2 \dots u_m)$, $v = (v_1\ v_2 \dots v_m)$ và $z = (z_1\ z_2 \dots z_m)$ trong P nên $u \neq v$.

. . .

Trước hết **cần** chứng minh phát biểu trên đúng khi u và v chỉ khác nhau tại 1 vị trí, m-1 vị trí còn lại đều như nhau.

hoặc:

Không mất tính tổng quát khi ta giả sử $u_k = 0$ và $v_k = 1$. ---> Không mất tính tổng quát có thể giả thiết $u_k = 0$ và $v_k = 1$.

V.V..

4. Về trích dẫn: vdụ

2. Các phương pháp tìm tập phổ biến hiện nay:

- Phương pháp sinh ứng viên: Apriori do Agrawal đề xuất và các thuật toán dựa Apriori như: AprioriTID, AprioriHybrid, DIC, DHP, PHP, …
- Phương pháp không sinh ứng viên:
  - ✓ Zaki: dựa vào cây IT-tree và phần giao của các Tidset để tính độ phổ biến.
  - ✓ J. Han: dựa vào cây FP-tree để khai thác tập phổ biến.
  - ✓ Ngoài ra, còn có một số phương pháp được đưa ra như: Lcm, DCI, …
- Phương pháp song song hoá

Nên viết lại như sau:

2. Một vài phương pháp tìm tập phổ biến:

- Phương pháp sinh ứng viên: Apriori, do Agrawal đề xuất trong [?] và các thuật toán dựa Apriori như: AprioriTID, AprioriHybrid, DIC, DHP, PHP, … trong [?]
- Phương pháp không sinh ứng viên, chẳng hạn: Phương pháp của Zaki: dựa vào cây IT-tree và phần giao của các Tidset để tính độ phổ biến, [?]; hoặc của J. Han: dựa vào cây FP-tree để khai thác tập phổ biến, [?]; hay các phương pháp Lcm, DCI, … đã trình bày trong [?]
- Phương pháp song song hoá, xem chẳng hạn trong [?]

Các dấu ? là số của tài liệu trong tham chiếu, không dùng dấu check!

5. Về ứng dụng:

Viết như vậy chưa đạt. Cần nêu rõ :

- Tình huống cần áp dụng :,
- Cách giải quyết : theo một số tác giả klhác đã làm
- Ứng dụng Phương pháp do bài báo đế xuất :,

- Kết quả ứng dụng và so sánh với các tác giả, ppháp trước

6. Về trình bày một giải thuật:
Cần theo format sau:
// tên giải thuật: mục đích của giải thuật.
// Input
// output
// Nội dung giải thuật
..
..
7. Về Kết quả, Kết luận:

- Kết quả: chỉ là kết quả ứng dụng giải thuật vào bài toán cụ thể của phần IV,
Lưu ý: kết quả đã làm trong Lvăn cao học cũng là 1 thử nghiệm, ý kiến hội đồng trao đổi là những phẩm bình cần lưu ý khi nói kết quả ứng dụng ,..

- Kết luận và khuyến nghị: (Tổng quát hơn kết quả ứng dụng),
Bài báo đã trình bày, đề xuất gì; qua vài ứng dụng đạt được gì; cái gì cần làm sáng tỏ hơn
Từ đó đưa ra các khuyến nghị: cần lưu ý gì, cần nghiên cứu phát triển thêm gì,…

)

I. CƠ SỞ TOÁN HỌC

1. Định nghĩa 1: Bao phủ

Xét không gian các vector s = (s_1, s_2, …, s_m) là các vector nhị phân có m chiều (s_i = 0 hoặc s_i = 1 $\forall i$ = 1, …, m)

Ta nói vector s1 = ($s1_1$, $s1_2$, …, $s1_m$) *bao phủ* vector s2 = ($s2_1$, $s2_2$, …, $s2_m$) nếu $\forall s2_i$ =1 thì $s1_i$ = 1, i = 1, …, m

Ví dụ: Với s1 = (1, 1, 1, 0) và s2 = (0, 1, 1, 0) thì ta nói s1 *bao phủ* s2

2. Định nghĩa 2: Mẫu

Xét tập hợp S có n vector s1, s2, ..., sn (S = {s1, s2, ..., sn})

Xét vector u = (u_1, u_2, …, u_m)

Nếu trong S, tồn tại k (k ∈ N, k ≤ n) vector sao cho k vector này đều bao phủ u thì ta nói (u, k) là một *mẫu* của tập hợp S (với u được gọi là *dạng* và k được gọi là *tần suất*).

Ví dụ: Với tập S = {(1,1,1,0), (0,1,1,1), (0,1,1,0), (0,0,1,0), (0,1,0,1)} thì ta nói ((0,1,1,0),2) là một mẫu của tập S

3. Định nghĩa 3: Mẫu cực đại

Cho (u, k) (u = (u_1, u_2, …, u_m), k ∈ N) là một mẫu của tập hợp S có n vector.

Ta nói (u; k) là *mẫu cực đại* của S nếu: !∃k'∈ N, k ≤ k' ≤ n mà (u; k') là một mẫu của S

Ví dụ: Với tập S = {(1,1,1,0), (0,1,1,1), (0,1,1,0), (0,0,1,0), (0,1,0,1)} thì ta nói ((0,1,1,0),3) là một mẫu cực đại của tập S

4. Định nghĩa 4: Đại diện

Tập hợp tất cả **các mẫu cực đại mà dạng của nó không bị bao phủ bởi dạng của một mẫu cực đại khác** của tập S được gọi là *đại diện* của S.

Nếu ta sắp n vector (m chiều) của tập S thành ma trận nhị phân n dòng x m cột thì ta sẽ thấy mỗi phần tử của tập đại diện là một *dạng hình chữ nhật tối đại* với *chiều cao* cực đại trong tập S

Ví dụ: Với tập S = {(1,1,1,0), (0,1,1,1), (1,1,1,0), (0,1,1,1), (0,0,1,1)}

Sắp các phần tử của S thành ma trận nhị phân 5x4

1	**1**	**1**	0
0	**1**	**1**	1
1	**1**	**1**	0
0	**1**	**1**	1

0 0 1 1

Ta có ((0,1,1,0), 4) là một dạng hình chữ nhật tối đại của S với chiều cao là 4

Tập đại diện của S là: {((1,1,1,0);2), ((0,1,1,0);4), ((0,0,1,0);5), ((0,1,1,1);2), ((0,0,1,1);3)}

Hình chữ nhật tối đại ((1,1,1,0);2)

1 **1** **1** 0
0 1 1 1
1 **1** **1** 0
0 1 1 1
0 0 1 1

Hình chữ nhật tối đại ((0,1,1,0);4)

1 **1** **1** 0
0 **1** **1** 1
1 **1** **1** 0
0 **1** **1** 1
0 0 1 1

Hình chữ nhật tối đại ((0,0,1,0);5)

1 1 **1** 0
0 1 **1** 1
1 1 **1** 0
0 1 **1** 1
0 0 **1** 1

Hình chữ nhật tối đại ((0,1,1,1);2)

1 1 1 0
0 **1** **1** **1**
1 1 1 0
0 **1** **1** **1**
0 0 1 1

Hình chữ nhật tối đại ((0,0,1,1);3)

1 1 1 0
0 1 **1** **1**
1 1 1 0
0 1 **1** **1**
0 0 **1** **1**

Chú ý: Mẫu cực đại ((1,1,0,0);2) không nằm trong đại diện của S vì dạng của nó bị bao phủ bởi dạng của mẫu cực đại ((1,1,1,0);2)

5. Định nghĩa 5: Phép toán $\cap$

Cho 2 phần tử thuộc tập S có n vector nhị phân-m chiều: s1 = ($s1_1$, $s1_2$, …, $s1_m$) và s2 = ($s2_1$, $s2_2$, …, $s2_m$)

s1 $\cap$ s2 = z (= (z_1, z_2, …, z_m)) với z_k = min ($s1_k$, $s2_k$), k = 1, .. , m

6. Định nghĩa 6: Phép toán $\underset{\cap}{}$

Cho 2 phần tử (u1;a) (1) và (u2;b) (2) với u1 và u2 là các vector nhị phân-m chiều, a và b là các số tự nhiên

(1) $\underset{\cap}{}$ (2) ta được phần tử mới (z; c) với z = u1 $\cap$ u2 và c = a + b

7. Định nghĩa 7: Phép toán $\subseteq$

Cho 2 phần tử (u1;a) (1) và (u2;b) (2) với u1 và u2 là các vector nhị phân-m chiều, a và b là các số tự nhiên

(1) $\subseteq$ (2) khi u1 = u2 và a $\leq$ b

8. Mệnh đề 1:

Cho tập S có n vector nhị phân-m chiều và P là đại diện của S thì 2 phần tử bất kỳ trong P không trùng nhau.

Điều này là hiển nhiên vì theo định nghĩa 4, các phần tử trong tập đại diện là các mẫu cực đại và các dạng của chúng không bao phủ nhau

9. Định lý 1:

Cho tập S có n vector nhị phân-m chiều và P là đại diện của S

Gọi z là một vector nhị phân-m chiều mới cần bổ sung vào S

Gọi (u;a), (v;b) là 2 phần tử thuộc P (với u và v là các vector nhị phân-m chiều, a và b là các số tự nhiên)

(u;a) $\underset{\cap}{}$ (z;1) = (t;a+1) (với t = u $\cap$ z)

(v;b) $\underset{\cap}{}$ (z;1) = (d;b+1) (với d = v $\cap$ z)

thì ta có 3 trường hợp sau:

(u;a) $\subseteq$ (t;a+1) , (u;a) $\subseteq$ (d;b+1) , t = d

(u;a) $\subseteq$ (t;a+1) , (u;a) !$\subseteq$ (d;b+1) , t != d

(u;a) !$\subseteq$ (t;a+1) , (u;a) !$\subseteq$ (d;b+1)

Chứng minh:

Để chứng minh điều trên ta chỉ cần chứng minh:

u = t , u = d , t = d

u = t , u $\neq$ d , t $\neq$ d

u $\neq$ t , u $\neq$ d

Ta có các vector nhị phân-m chiều u, v và z như sau:

u = (u_1 u_2 … u_m)

v = (v_1 v_2 … v_m)

z = (z_1 z_2 … z_m)

Theo mệnh đề 1 ta đã biết 2 phần tử bất kỳ của tập P luôn khác nhau, suy ra $u \neq v$

Trước tiên, ta sẽ chứng minh phát biểu trên đúng khi u và v chỉ khác nhau tại 1 vị trí (còn m-1 vị trí còn lại giống nhau)

Giả sử vị trí khác nhau là k, vậy xét 2 giá trị u_k và v_k

Không mất tính tổng quát khi ta giả sử $u_k = 0$ và $v_k = 1$. Vậy giá trị z_k có 2 tình huống là $z_k = 0$ hoặc $z_k = 1$

Khi $z_k = 0$

$u_k \cap z_k = \min(u_k , z_k) = \min(0 , 0) = 0$

$v_k \cap z_k = \min(v_k , z_k) = \min(1 , 0) = 0$

Vậy:

$(u_1\ u_2 \dots u_k \dots u_m) \cap (z_1\ z_2 \dots z_k \dots z_m)$

$= (u_1\ u_2 \dots 0 \dots u_m) \cap (z_1\ z_2 \dots 0 \dots z_m)$

$= (t_1\ t_2 \dots 0 \dots t_m)$ (với $t_i = u_i \cap z_i$, i = 1..m, i≠k)

$(v_1\ v_2 \dots v_k \dots v_m) \cap (z_1\ z_2 \dots z_k \dots z_m)$

$= (v_1\ v_2 \dots 1 \dots v_m) \cap (z_1\ z_2 \dots 0 \dots z_m)$

$= (t_1\ t_2 \dots 0 \dots t_m)$ (với $t_i = v_i \cap z_i$, i = 1..m, i≠k. Chú ý u và v giống nhau tại m-1 vị trí, khác tại vị trí k)

Rõ ràng, trong trường hợp này, u có thể bằng với vector được tạo ra khi $u \cap z$ và cũng có thể bằng với vector được tạo ra khi $v \cap z$ nhưng 2 vector này là một.

Khi $z_k = 1$

$u_k \cap z_k = \min(u_k , z_k) = \min(0 , 1) = 0$

$v_k \cap z_k = \min(v_k , z_k) = \min(1 , 1) = 1$

Vậy:

$(u_1\ u_2 \dots u_k \dots u_m) \cap (z_1\ z_2 \dots z_k \dots z_m)$

$= (u_1\ u_2 \dots 0 \dots u_m) \cap (z_1\ z_2 \dots 1 \dots z_m)$

$= (t_1\ t_2 \dots 0 \dots t_m)$ (với $t_i = u_i \cap z_i$, i = 1..m, i ≠ k)

$(v_1\ v_2 \dots v_k \dots v_m) \cap (z_1\ z_2 \dots z_k \dots z_m)$

$= (v_1\ v_2 \dots 1 \dots v_m) \cap (z_1\ z_2 \dots 1 \dots z_m)$

$= (t_1\ t_2 \dots 1 \dots t_m)$ (với $t_i = v_i \cap z_i$, i = 1..m, i ≠ k. Chú ý u và v giống nhau tại m-1 vị trí, khác tại vị trí k)

Rõ ràng, trong trường hợp này, u chỉ có thể bằng với vector được tạo ra khi $u \cap z$ và hoàn toàn không thể bằng với vector được tạo ra khi $v \cap z$.

Hiển nhiên là vẫn có khả năng u không thể bằng với vector được tạo ra khi $u \cap z$ cũng như không thể bằng với vector được tạo ra khi $v \cap z$

Vậy trường hợp 2 vector u, v khác nhau tại 1 vị trí thì phát biểu trên là đúng

Giả sử phát biểu trên đúng cho trường hợp 2 vector u, v khác nhau tại l vị trí, ta sẽ chứng minh nó cũng đúng cho trong trường hợp 2 vector u, v khác nhau tại l+1 vị trí

Theo giả thiết ta có 3 tình huống (khi xét l vị trí khác nhau của u, v) là

Tình huống 1: u = t, u = d, t = d

Tình huống 2: u = t, u ≠ d, t ≠ d

Tình huống 3: u ≠ t, u ≠ d

Khi xét l+1 vị trí khác nhau nghĩa là ta đã xét l vị trí và thêm 1 vị trí khác nhau nữa

Khi xét riêng cho 1 vị trí khác nhau, ta cũng có 3 tình huống như trên (đã được chứng minh ở trên)

Vậy khi xét l+1 vị trí khác nhau ta thấy có thể xảy ra 9 khả năng

(u = t, u = d, t = d) kết hợp với (u = t, u = d, t = d) cho ra (u = t, u = d, t = d)

kết hợp với (u = t, u ≠ d, t ≠ d) cho ra (u = t, u ≠ d, t ≠ d)

kết hợp với (u ≠ t, u ≠ d) cho ra (u ≠ t, u ≠ d)

(u = t, u ≠ d, t ≠ d) kết hợp với (u = t, u = d, t = d) cho ra (u = t, u ≠ d, t ≠ d)

kết hợp với (u = t, u ≠ d, t ≠ d) cho ra (u = t, u ≠ d, t ≠ d)

kết hợp với (u ≠ t, u ≠ d) cho ra (u ≠ t, u ≠ d)

(u ≠ t, u ≠ d) kết hợp với (u = t, u = d, t = d) cho ra (u ≠ t, u ≠ d)

kết hợp với (u = t, u ≠ d, t ≠ d) cho ra (u ≠ t, u ≠ d)

kết hợp với (u ≠ t, u ≠ d) cho ra (u ≠ t, u ≠ d)

Ta thấy với l+1 vị trí khác nhau của u và v thì cũng xảy ra đúng 3 tình huống như trên.

Vậy phát biểu trên đúng cho mọi trường hợp

10. Thuật toán 1: Tìm đại diện mới của S khi S được bổ sung thêm 1 phần tử mới

Cho tập S có n vector nhị phân-m chiều và P là đại diện của S

Gọi z là một vector nhị phân-m chiều mới cần bổ sung vào S

Thuật toán sau sẽ giúp tìm ra đại diện mới của S

Procedure FindOutP(P, z)

M = ∅ (M là tập các phần tử mới sẽ xuất hiện trong P)
flag1 = 0
flag2 = 0
For each p ∈ P do
    m = p ∩ (z;1)
    If m ≠ 0
        if p ⊆ m then P = P \ {p}
        if (z;1) ⊆ m then flag1 = 1
        For each m' ∈ M do
            if m' ⊆ m then
                M = M \ {m'}
                break for
            endif
            if m ⊆ m' then
                flag2 = 1
                break for
            endif
        endfor
    else
        flag2 = 1
    endif
endfor
if flag1 = 0 then P = P ∪ {(z;1)}
if flag2 = 0 then M = M ∪ {m}
P = P ∪ M
M = ∅
return P

11. Định lý 2:

Cho tập S có n vector nhị phân-m chiều.

Nếu áp dụng thuật toán 1 lần lượt cho n phần tử của S ta sẽ tìm ra được đại diện P đầy đủ của S.

Chứng minh:

Gọi s_i (i = 1..n) là n phần tử của S

Để chứng minh thuật toán đúng, ta cần chứng minh thuật toán sẽ cho ra tất cả các tập có dạng: $s\_i_1 \cap \ldots \cap s\_i_k$, với $1 \le i_1 < \ldots < i_k \le n$

Ta sẽ chứng minh bằng phương pháp quy nạp

Với n=1, thuật toán hiển nhiên đúng

Giả sử thuật toán đúng với n

Xét n+1

Xét một tập X có dạng $X = s\_i_1 \cap \ldots \cap s\_i_k$, với $1 \le i_1 < \ldots < i_k \le n+1$

Ta thấy có 2 trường hợp:

TH1: $i_k < n+1 \Rightarrow i_k \le n \Rightarrow$ Theo giả thuyết quy nạp, X đã có ở bước thứ n

TH2: $i_k = n+1$

X sẽ có dạng: $X = Y \cap s\_n+1$, với $Y = s\_i_1 \cap \ldots \cap s\_i_{k-1} \Rightarrow$ Y ở trong bước thứ n

Vì ở bước thứ n+1 sẽ chứa những tập có dạng $s\_n+1 \cap W$, với W ở bước thứ n

$\Rightarrow$ X sẽ chứa trong bước thứ n+1

II. ỨNG DỤNG VÀO DATA MINING

1. Bài toán Khai thác Luật Kết Hợp:

Bài toán khai thác luật kết hợp được đưa ra vào năm 1993 bởi Agrawal. Từ khi nó được giới thiệu, bài toán khai thác luật kết hợp nhận được rất nhiều sự quan tâm của nhiều nhà khoa học. Ngày nay việc khai thác các luật như thế vẫn là một trong những phương pháp khai thác mẫu phổ biến nhất trong việc khám phá tri thức và khai thác dữ liệu.

Khai thác luật kết hợp là tiến trình khám phá các tập giá trị thuộc tính xuất hiện phổ biến trong các đối tượng dữ liệu. Từ tập phổ biến có thể tạo ra các luật kết hợp giữa các giá trị thuộc tính nhằm phản ánh khả năng xuất hiện đồng thời các giá trị thuộc tính trong tập các đối tượng. Một luật kết hợp $X \rightarrow Y$ phản ánh sự xuất hiện của tập X dẫn đến sự xuất hiện đồng thời tập Y. Trong CSDL bán hàng, một luật kết hợp tiêu biểu như sau:

Có 67% khách hàng mua bia 333, rượu Nàng Hương thì mua bánh tôm Cầu Tre

Luật kết hợp giúp các nhà hoạch định hiểu rõ xu thế bán hàng, tâm lý khách hàng, … từ đó đưa ra các chiến lược bố trí mặt hàng, kinh doanh, tiếp thị, tồn kho, …

Khai thác luật kết hợp được chia làm hai giai đoạn:

- Tìm tất cả các tập phổ biến thỏa ngưỡng phổ biến.
- Tìm tất cả các luật thỏa ngưỡng tin cậy.

2. Các phương pháp tìm tập phổ biến hiện nay:

- Phương pháp sinh ứng viên: Apriori do Agrawal đề xuất và các thuật toán dựa Apriori như: AprioriTID, AprioriHybrid, DIC, DHP, PHP, …

- Phương pháp không sinh ứng viên:
  - ✓ Zaki: dựa vào cây IT-tree và phần giao của các Tidset để tính độ phổ biến.
  - ✓ J. Han: dựa vào cây FP-tree để khai thác tập phổ biến.
  - ✓ Ngoài ra, còn có một số phương pháp được đưa ra như: Lcm, DCI, …
- Phương pháp song song hoá

3. Những thách thức được đặt ra hiện tại:

- Thách thức lớn nhất là khi làm việc với một cơ sở dữ liệu (CSDL) có nhiều biến động, đặc biệt là khi CSDL có nhu cầu bổ sung thêm hóa đơn ta không cần phải chạy lại thuật toán từ đầu
- Một số thuật toán hiệu quả nhưng cơ sở toán học và cách cài đặt của chúng lại khá phức tạp
- Hạn chế của bộ nhớ vật lý. Do đó một thách thức cũng khá lớn là làm thế nào để có thể lưu trữ được ngữ cảnh khai thác dữ liệu một cách hiệu quả nhất (ít tốn bộ nhớ nhất) kết hợp với việc lưu trữ luôn những tập phổ biến cần thiết

Trong báo cáo này, ta sẽ áp dụng cơ sở toán học ở phần I để giải quyết phần nào những thách thức trên

4. Ngữ cảnh khai thác dữ liệu (KTDL):

Cho tập ***O*** là tập hữu hạn khác rỗng các hoá đơn và ***I*** là tập hữu hạn khác rỗng các mặt hàng, ***R*** là một quan hệ hai ngôi giữa ***O*** và ***I*** sao cho với $o \in O$ và $i \in I$, $(o,i) \in R \Leftrightarrow$ hoá đơn ***o*** có chứa mặt hàng ***i***. Ngữ cảnh KTDL là bộ ba ***(O,I,R)***.

Bảng 1 là một ví dụ về bảng chi tiết hoá đơn và bảng 2 là ngữ cảnh KTDL được tạo từ bảng 1

Ma trận ngữ cảnh KTDL

Cho bảng dữ liệu ***chi tiết hóa đơn*** gồm hai thuộc tính là mã hóa đơn và mã hàng. Gọi ***O*** là tập các hoá đơn, ***I*** là tập các mặt hàng và ***R*** là một quan hệ hai ngôi giữa ***O*** và ***I***, $R \subseteq OxI$, trong đó $(o,i) \in R$ nếu và chỉ nếu hoá đơn ***o*** có chứa mặt hàng ***i***.

Quan hệ hai ngôi ***R*** được biểu diễn bằng một ma trận nhị phân trong đó dòng thứ ***i*** ứng với hoá đơn o_i và cột thứ ***j*** ứng với mặt hàng i_j. Ma trận này được gọi là ma trận biểu diễn ngữ cảnh KTDL. Bảng 2 là ma trận nhị phân biểu diễn ngữ cảnh KTDL tương ứng với bảng dữ liệu ***chi tiết hoá đơn*** trong bảng 1

Bảng 1. *Một ví dụ về bảng chi tiết hoá đơn*

Mã hoá đơn	Mã hàng
o1	i1
o1	i2
o1	i3
o2	i2
o2	i3
o2	i4
o3	i2
o3	i3
o3	i4
o4	i1
o4	i2
o4	i3
o5	i3
o5	i4

Bảng 2. *Ma trận nhị phân biểu diễn ngữ cảnh KTDL*

	i1	i2	i3	i4
o1	1	1	1	0
o2	0	1	1	1
o3	0	1	1	1
o4	1	1	1	0
o5	0	0	1	1

5. Áp dụng thuật toán ở phần I:

Thực chất của thuật toán, tại mỗi bước, chính là tìm ra những dạng hình chữ nhật mới được hình thành bằng cách thực hiện thao tác giao các bit nhị phân giữa dòng dữ liệu tương ứng của bước đó với những dạng hình chữ nhật đã được tìm thấy ở các bước trước (được lưu lại trong tập P). Tuy nhiên, ở bước đầu tiên, ta phải "tranh thủ" dùng phương pháp gom nhóm để nhanh chóng lấy được những dạng hình chữ nhật "hiển nhiên" và "chiều cao" của chúng.

Tiếp theo, nếu dạng hình chữ nhật mới được hình thành bao trùm (cùng dạng nhưng lại có chiều cao lớn hơn) lên một dạng hình chữ nhật đã có trong tập P thì ta sẽ tiến hành loại bỏ dạng hình chữ nhật cũ này. Nghĩa là ta phải tiến hành thao tác rà soát và loại bỏ những phần tử cũ hiện có của tập P nếu chúng chứa trong những phần tử mới được tạo ra qua phép giao.

Ngoài ra, ta cũng phải loại bỏ các dạng hình chữ nhật mới được tạo ra nếu chúng bị bao trùm lẫn nhau. Nghĩa là lại phải tiến hành thao tác rà soát và loại bỏ những phần tử mới vừa được tạo ra qua phép giao nếu chúng chứa trong

một trong những phần tử cũng vừa được tạo ra qua phép giao.

Tóm lại, sau mỗi bước, ta sẽ luôn có được một tập P với các phần tử khác nhau từng đôi một, nghĩa là ta thu được các dạng hình chữ nhật tối đại khác nhau tồn tại trong cơ sở dữ liệu tính từ dòng dữ liệu đầu tiên cho đến dòng dữ liệu tương ứng với bước hiện hành.

6. Ví dụ:

Ma trận nhị phân biểu diễn ngữ cảnh khai thác dữ liệu, minsupp là 40%

	i1	i2	i3	i4
o1	1	1	1	0
o2	0	1	1	1
o3	0	1	1	1
o4	1	1	1	0
o5	0	0	1	1

Mục tiêu cuối cùng là xây dựng tập P chứa tất cả các dạng hình chữ nhật tối đại của ma trân nhị phân biểu diễn ngữ cảnh khai thác dữ liệu

Đầu tiên: Ta thực hiện gom nhóm để có được bảng sau

Dạng (vector nhị phân)	**Tần suất**
1110	**2**
0111	**2**
0011	**1**

Bước 1:

Xét dòng 1

Đưa dòng 1 vào P ta có :

P={(1110; 2)}

Bước 2:

Xét dòng 2

Lấy dòng 2 lần lượt giao với các phần tử hiện có trong P để được các phần tử mới của P. Với mỗi thao tác thực hiện phép toán giao, đồng thời ta sẽ xét xem các phần tử cũ hiện tại của P có chứa trong các phần tử mới vừa được tạo ra hay không để tiến hành loại bỏ ngay phần tử cũ này. Tương tự, ta cũng xét xem dòng đang xét (dòng 2) có chứa trong các phần tử mới được tạo ra hay không để đánh dấu loại bỏ, không đưa dòng đang xét vào P. Ta có:

(1110; 2) $\cap$ (0111; 2) = (0110; 4) //ta thấy (1110; 2) và (0111; 2) đều không chứa trong (0110; 4)

Lúc này P chứa

P={(1110; 2) **(1)** //phần tử cũ của P

(0110; 4) **(2)** //phần tử mới tạo được qua phép giao

(0111; 2) **(3)** //dòng 2 được đưa vào P vì không chứa trong phần tử mới tạo ra qua phép toán $\cap$

}

Bước 3:

Xét dòng 3

Lấy dòng 3 lần lượt giao với các phần tử trong P để được các phần tử mới của P, đồng thời tiến hành xét loại ngay khi giao, ta có:

(1110; 2) $\cap$ (0011; 1) = (0010; 3)

(0110; 4) $\cap$ (0011; 1) = (0010; 5)

(0111; 2) $\cap$ **(0011; 1)** = (0011; 3) //loại (0011; 1) vì $\subseteq$ (0011; 3)

Lúc này S chứa

P={(1110; 2) **(1)**

(0110; 4) **(2)**

(0111; 2) **(3)**

~~(0010; 3) **(4)**~~ **//phần tử mới bị** $\subseteq$ 1 phần tử mới khác là (5)

(0010; 5) (5)

(0011; 3) (6)

}

Sau khi xét loại bớt các phần tử mới chứa lẫn nhau, lúc này P còn:

P={(1110; 2) **(1)**

(0110; 4) **(2)**

(0111; 2) **(3)**

(0010; 5) (4)

(0011; 3) (5)

}

Lúc này 5 phần tử của P chính là các dạng hình chữ nhật tối đại đại diện của ma trận nhị phân biểu diễn ngữ cảnh khai thác dữ liệu của bài toán, chiều cao của các dạng hình chữ nhật này là số lượng các hoá đơn ở phần sau (tần suất). Và tập phổ biến tối đại (để tìm được luật) thỏa minsupp=40% (2/5) là:

{{i2, i3, i4}(2/5); {i1, i2, i3}(2/5); {i2, i3}(4/5); {i3, i4}(3/5)**}**

7. Các kết quả đạt được:

- Giải quyết được trường hợp bổ sung thêm dữ liệu (thêm hóa đơn hoặc thêm các mặt hàng mới) vào bài toán mà không cần chạy lại thuật toán từ đầu
- Đồng thời cũng rất hiệu quả khi cần xóa hoặc sửa (là thao tác xóa đi và thêm mới) thông tin các hóa đơn (nếu áp dụng phương pháp lưu trữ chỉ số các hóa đơn)
- Thuật toán dễ cài đặt và có độ phức tạp thấp ($n2^{2m}$, với n là số lượng các hóa đơn và m là số lượng các mặt hàng)
- Tập P thu được cuối cùng chính là đại diện cho ngữ cảnh của bài toán và ta chỉ cần lưu trữ lại tập P này mà không cần phải lưu lại toàn bộ ngữ cảnh bài toán đôi khi rất lớn
- Giải quyết được trường hợp bộ nhớ yếu, không đủ để giải quyết những bài toán với ngữ cảnh quá lớn. Vì với thuật toán này ta hoàn toàn có thể phân đoạn ngữ cảnh để giải quyết từng đoạn một
- Có thể song song hoá thuật toán một cách đơn giản

TÊN TIẾNG ANH CỦA BÀI BÁO

Abstract: *A abb. . .*