

KHAI PHÁ DỮ LIỆU

Bài 3. Luật kết hợp

Giáo viên: TS. Trần Mạnh Tuấn

Bộ môn: Hệ thống thông tin

Khoa: Công nghệ thông tin

Email: tmtuan@tlu.edu.vn

Điện thoại: 0983.668.841

Nội dung

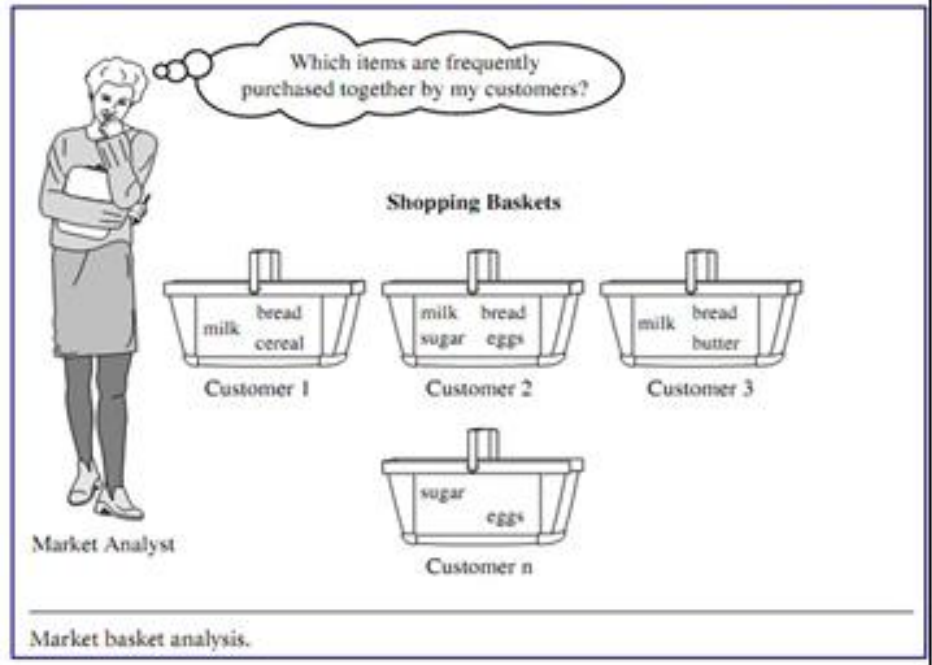
- ❖ Tổng quan
- ❖ Phát biểu bài toán
- ❖ Một số thuật giải
 - Thuật giải Apriori
 - Thuật giải AprioriTid
 - Thuật giải FP_Growth
 - ✓ Thuật toán 1: Simple algorithm
 - ✓ Thuật toán 2: Fast algorithm
 - ✓ Thuật toán 3: Tìm luật đơn giản

Tổng quan

Bài toán phân tích giỏ hàng



Ide Dasar



Bài toán phân tích giỏ hàng

Những mặt hàng nào thường được khách hàng mua cùng nhau trong cùng 1 lần mua hàng?

- Thiết kế gian hàng.
- Lên kế hoạch bán giảm giá cho mặt hàng/nhóm mặt hàng.
- Lên kế hoạch tiếp thị/các chiến lược quảng cáo.
- .v.v.

Tổng quan

Tiếp thị chéo



Tổng quan

Tiếp thị chéo

Đắc Nhân Tâm - Cuốn Sách Hay Nhất Của Mọi Thời Đại Đưa Bạn Đến Thành Công - Sách Vinabook.com - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites Media Print Mail News RSS Feeds Links



★★★★★ (11 người đánh giá)

Số trang: 320
Kích thước: 14.5x20.5 cm
Trọng lượng: 370 gram
(Chi tiết về phí vận chuyển)

Hình thức bìa: Bìa mềm
Ngày xuất bản: 09 - 2008
Số lần xem: 87539

Giá bìa: 48.000 VNĐ
Giá bán: **48.000 VNĐ**
Giảm giá: (0%)


Vietnam đồng

In trang này

Gửi cho bạn bè

Xếp hạng: 12 (trong những cuốn Sách bán chạy)

Sách nên mua kèm với sách này
Nên mua cuốn sách trên cùng với cuốn **Quảng Gánh Lo Đi Và Vui Sống - Những Ý Tưởng Tuyệt Vời Để Sống Thanh Thản Và Hạnh Phúc** - Nxb Trẻ



Giá bán tất cả: **96.000 VNĐ**

Thêm tất cả vào giỏ hàng

Khách hàng mua cuốn sách trên cũng từng mua một trong những cuốn sách dưới đây



Người Giỏi Không Phải Là Người Làm ... Tác giả: Donna M. Genett
Giá bán: 24.000 VNĐ



Quảng Gánh Lo Đi Và Vui Sống - Những Ý ... Tác giả: Dale Carnegie
Giá bán: 48.000 VNĐ



Lập Bản Đồ Tư Duy - Công Cụ Tư Duy ... Tác giả: Tony Buzan
Giá bán: 24.000 VNĐ



Nguyên Lý 80/20 - Bí Quyết Làm Ít Được ... Tác giả: Richard Koch
Giá bán: 70.000 VNĐ



Bài Giảng Cuối Cùng - The Last Lecture ... Tác giả: Randy Pausch
Giá bán: 58.000 VNĐ

Trang: 1 / 10

Done Internet

Tổng quan

- ▣ Phân tích dữ liệu giỏ hàng (basket data analysis)
- ▣ Tiếp thị chéo (cross-marketing)
- ▣ Thiết kế catalog (catalog design)
- ▣ Phân loại dữ liệu (classification) và gom cụm dữ liệu (clustering) với các mẫu phổ biến
- ▣ ...

Tổng quan

- ❖ Luật kết hợp (LKH) là một hướng quan trọng trong KPDL.
- ❖ Giúp ta tìm được các mối liên hệ giữa các mục dữ liệu/thuộc tính (items) của DL.
- ❖ Tìm các luật kết hợp ‘quý hiếm’ và mang nhiều thông tin từ CSDL tác nghiệp là một trong những hướng tiếp cận chính của lĩnh vực khai phá dữ liệu.

Tổng quan

- ❖ VD luật kết hợp: “80 % khách hàng mua máy điện thoại di động thì mua thêm simcard, 30 % có mua cả máy điện thoại di động lẫn simcard”.
- ❖ “mua máy điện thoại di động” là vế trái (tiền đề) của luật, còn “mua simcard” là vế phải (kết luận) của luật.
- ❖ Các số 30% là độ hỗ trợ của luật (support - số phần trăm các giao dịch chứa cả vế trái và vế phải), 80% là độ tin cậy của luật (confidence - số phần trăm các giao dịch thoả mãn vế trái thì cũng thoả mãn vế phải).

Tổng quan

Các hướng tiếp cận trong khai phá LKH

❖ LKH nhị phân (Binary association rule):

- Các items chỉ được quan tâm là có hay không xuất hiện trong CSDL giao tác (Transaction database) chứ không quan tâm về Mức độ hay tần xuất xuất hiện.
- Thuật giải Apriori.

❖ LKH có thuộc tính số và thuộc tính hạng mục

- Dùng các phương pháp rời rạc hoá chuyển về dạng nhị phân để có thể áp dụng các thuật giải đã có.

Tổng quan

Các hướng tiếp cận trong khai phá LKH

- ❖ LKH tiếp cận theo hướng tập thô (Mining association rules base on rough set):
 - Tìm kiếm LKH dựa trên lý thuyết tập thô.
- ❖ LKH nhiều mức (Multi-level association rules):
 - Với cách tiếp cận LKH thế này sẽ tìm kiếm thêm những luật có dạng: mua máy tính PC \Rightarrow mua hệ điều hành Window AND mua phần mềm văn phòng Microsoft Office,....

Tổng quan

Các hướng tiếp cận trong khai phá LKH

❖ LKH mờ (fuzzy association rules):

- Với những khó khăn gặp phải khi rời rạc hoá các thuộc tính số, LKH mờ khắc phục hạn chế đó và chuyển luật kết hợp về một dạng gần gũi hơn.

❖ LKH với thuộc tính được đánh trọng số (Association rule with weighted items):

- Các thuộc tính được đánh trọng số theo mức độ xác định nào đó.
- Nhờ vậy, thu được những luật “ hiếm ”(tức là có độ hỗ trợ thấp nhưng mang nhiều ý nghĩa).

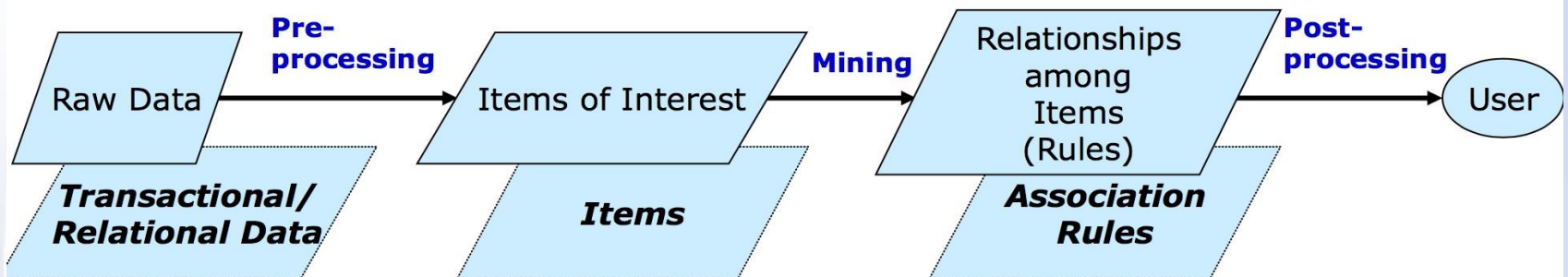
Tổng quan

Các hướng tiếp cận trong khai phá LKH

- ❖ LLKH song song (Parallel mining of association rule).
 - Nhu cầu song song hoá và xử lý phân tán là cần thiết vì kích thước DL ngày càng lớn.

Tổng quan

□ Quá trình khai phá luật kết hợp



Transaction	Items_bought
2000	A, B, C
1000	A, C
4000	A, D
5000	B, E, F
...	

A, B, C, D, F,
...

$A \rightarrow C$ (50%, 66.6%)
...

Bài toán phân tích giỏ thị trường

Phát biểu bài toán

- ❖ Cho $I = \{I_1, I_2, \dots, I_n\}$ là một tập các mục (mặt hàng, .v.v.).
- ❖ Cho D là một tập các giao dịch mà mỗi giao dịch T là một tập các mục, $T \subseteq I$.
- ❖ Mỗi giao dịch có một mã định danh riêng gọi là TID .
- ❖ Cho A là một tập các mục (mặt hàng). Một giao dịch T được gọi là chứa A khi và chỉ khi $A \subseteq T$.
- ❖ *Một luật kết hợp được diễn đạt dưới hình thức $A \Rightarrow B$, với $A \subset I, B \subset I$, và $A \cap B = \emptyset$*
- ❖ **Ý nghĩa:** Khi xuất hiện A thì B cũng xuất hiện (với xác suất nào đó)

Phát biểu bài toán

- ❖ VD1: Bảng 1 mô tả CSDL tác vụ, A, C, D, T, W là các mục: T_i ($T_i = 1, 2, 3, 4, 5, 6$) là các tác vụ.
- ❖ Mỗi giá trị của mục dữ liệu (Item) thể hiện thuộc tính xuất hiện hay không xuất hiện (nhận giá trị 0) trong tác vụ.

<i>Tập các tác vụ</i>	<i>Các mục dữ liệu</i>
1	A C T W
2	C D W
3	A C T W
4	A C D W
5	A C D T W
6	C D T

TID	A	C	D	T	W
T_1	1	1	0	1	1
T_2	0	1	1	0	1
T_3	1	1	0	1	1
T_4	1	1	1	0	1
T_5	1	1	1	1	1
T_6	0	1	1	1	0

Phát biểu bài toán

- ❖ Hai thông số quan trọng của luật kết hợp là độ hỗ trợ/độ phổ biến (s) và độ tin cậy (c).
- ❖ Định nghĩa 1: Độ hỗ trợ (support) của tập X trong CSDL D là tỷ lệ phần trăm các bản ghi chứa tập X với tổng số các giao dịch có trong CSDL

$$Support(X) = \frac{count(X)}{|D|}$$

- ❖ Định nghĩa 2: Độ hỗ trợ (support) của $X \Rightarrow Y$ là tỷ lệ phần trăm các bản ghi $X \cup Y$ với tổng số các giao dịch có trong CSDL.

$$Support(X \Rightarrow Y) = support(X \cup Y)$$

$$support(X \Rightarrow Y) = P(X \cup Y)$$

- ❖ Định nghĩa 3: Độ tin cậy (confidence) của $X \Rightarrow Y$ là tỷ lệ phần trăm của số giao dịch có chứa $X \cup Y$ với số giao dịch có chứa X.

$$Confidence(X \Rightarrow Y) = support(X \cup Y) / support(X)$$

$$confidence(X \Rightarrow Y) = P(Y|X)$$

Phát biểu bài toán

- ❖ Luật kết hợp thường được đánh giá dựa trên 2 độ đo là **độ hỗ trợ** và **độ tin cậy**.
- ❖ Tìm tất cả các luật có độ hỗ trợ và độ tin cậy lớn hơn ngưỡng xác định trước.
 - Ngưỡng của độ hỗ trợ là **minsup**
 - Ngưỡng của độ tin cậy là **minconf**.
- ❖ VD: Khi phân tích giỏ hàng của người mua hàng: 80% khách hàng mua sữa thì cũng mua bánh mì, 30% thì mua cả hai thứ .
 - Trong đó “mua sữa ”là tiền đề còn “mua bánh mì ”là kết luận của luật. Con số 30% là độ hỗ trợ của luật còn 80% là độ tin cậy của luật.

Phát biểu bài toán

Phát biểu bài toán

- ❖ Khai phá LKH là bài toán tìm tất cả các luật dạng $X \Rightarrow Y$ với $(X, Y \in I, \text{ và } X \cap Y = \emptyset)$ thỏa mãn độ hỗ trợ và độ tin cậy tối thiểu.
 - $\text{Support}(X \Rightarrow Y) \geq \text{minsup}$
 - $\text{Confidence}(X \Rightarrow Y) \geq \text{minconf}$

Phát biểu bài toán

- ❖ Định nghĩa 4: Nếu tập X có $\text{support}(X) \geq \text{minsup}$ thì X gọi là tập phổ biến (Frequent itemset). Kí hiệu các tập này là FI.
- ❖ Luật kết hợp tin cậy $r = X \Rightarrow Y$ được gọi là luật chính xác nếu $\text{Confidence}(r) = 1$ và được gọi là xấp xỉ nếu $\text{Confidence}(r) < 1$.

Phát biểu bài toán

- ❖ Ví dụ 2: Trong CSDL bảng 1, tất cả các tập phổ biến với độ hỗ trợ cực tiểu là 0.5 (hay 50%) và tất cả các luật với độ tin cậy cực tiểu là 0,8 (hay 80%).

Độ hỗ trợ	Tập mục	Độ tin cậy	Tập tất cả các luật
100% (6) 83% (5) 67% (4) 50% (3)	C W, CW. A, D, T, AC, AW, CD, CT, ACW. AT, DW, TW, ACT, ATW, CTW, CDW, ACTW.	1,0 (100%)	<u>Các luật chính xác:</u> $A \Rightarrow C(4/4)$, $A \Rightarrow CW(4/4)$, $A \Rightarrow W(4/4)$, $D \Rightarrow C(4/4)$, $T \Rightarrow C(4/4)$, $AC \Rightarrow W(4/4)$, $W \Rightarrow C(5/5)$, $AW \Rightarrow C(4/4)$, $AT \Rightarrow C(3/3)$, $AT \Rightarrow W(3/3)$, $DW \Rightarrow C(3/3)$, $TW \Rightarrow A(3/3)$, $TW \Rightarrow C(3/3)$, $AT \Rightarrow CW(3/3)$, $TW \Rightarrow AC(3/3)$, $ACT \Rightarrow W(3/3)$, $ATW \Rightarrow C(3/3)$, $CTW \Rightarrow A(3/3)$.
		0,8 (80%)	<u>Các luật kết hợp xấp xỉ</u> $W \Rightarrow A(4/5)$, $C \Rightarrow W(5/6)$, $W \Rightarrow AC(4/5)$

Phát biểu bài toán

- ❖ Ngữ nghĩa của luật kết hợp: Luật kết hợp $r = X \Rightarrow Y$ có độ hỗ trợ s và độ tin cậy c . Có nghĩa là đối với CSDL đã cho có $s\%$ các tác vụ chứa cả hai tập mục dữ liệu X, Y ; trong đó có $c\%$ các tác vụ chứa tập mục dữ liệu X cũng sẽ chứa tập mục dữ liệu Y .
- ❖ Ví dụ 3 : Xét luật $AW \Rightarrow C$ trong VD 2 thì tập mục dữ liệu ACW có độ hỗ trợ là 67% , có độ tin cậy là 100%
- ❖ Có thể diễn giải như sau:
 - Có 67% những vụ mua sắm mua cả 3 mặt hàng A, C, W .
 - 100% những vụ mua sắm có mua A, W cũng mua C .

Phát biểu bài toán

- ❖ Quá trình tìm các LKH gồm 2 pha:
 - Pha 1: Tìm tất cả các tập phổ biến (tìm FI) trong CSDL T.
 - Pha 2: Sử dụng tập FI để sinh ra các quy tắc luật

Phát biểu bài toán

Từ phân tích trên chia thành hai bài toán con

- ❖ Bài toán 1: Khám phá tất cả các tập phổ biến theo ngưỡng MINSUP cho trước. Gồm các thuật giải:
 - Apriori
 - AprioriTid
 - FP_Growth

Phát biểu bài toán

- ❖ Bài toán 2: Tìm luật, gồm hai bước:
 - B1: Khám phá các LKH theo ngưỡng MINCONF cho trước
 - Thuật giải 1: Simple algorithm
 - Thuật giải 2: Fast algorithm
 - Thuật giải 3: Tìm luật đơn giản
 - B2: Loại luật thừa
 - Dùng quy luật loại bỏ luật thừa
 - Phương pháp lọc dùng mẫu đơn giản

Phát biểu bài toán

- ❖ Thách thức chính trong khai phá các tập mục thường xuyên từ một tập dữ liệu lớn chính là việc tạo ra một lượng cực lớn các tập mục thỏa mãn độ hỗ trợ tối thiểu (min_sup), đặc biệt khi min_sup được cho giá trị cực nhỏ.
- ❖ Điều này xảy ra bởi vì một tập mục được coi là thường xuyên nếu các tập con của nó cũng là những tập mục thường xuyên. Như vậy một tập mục dài sẽ chứa một số tổ hợp các tập mục con thường xuyên ngắn hơn.

Thuật giải Apriori

- ❖ Do Apriori do Rakesh Agrawal, Tomasz Imielinski, Arun Swami đề xuất [1993].
- ❖ Tìm giao dịch t có độ hỗ trợ và độ tin cậy thoả mãn lớn hơn một giá trị ngưỡng nào đó.
 - Thuật giải được tĩa bớt những tập ứng cử viên có tập con không phổ biến trước khi tính độ hỗ trợ.

Thuật giải Apriori

Ý tưởng thuật giải

- ❖ Tạo các tập 1_itemset: từ các item trên CSDL, ta xác định độ hỗ trợ s cho từng item dựa vào CSDL đã mã hóa, loại đi các item có $s < \text{minsup}$.
- ❖ Tạo các tập 2_itemset: xác định độ hỗ trợ s cho tập gồm 2 item, loại đi các item có $s < \text{minsup}$.
- ❖ ...
- ❖ Tạo các tập k_itemset: xác định độ hỗ trợ s cho tập gồm k item, loại đi các item có $s < \text{minsup}$.

Thuật giải Apriori

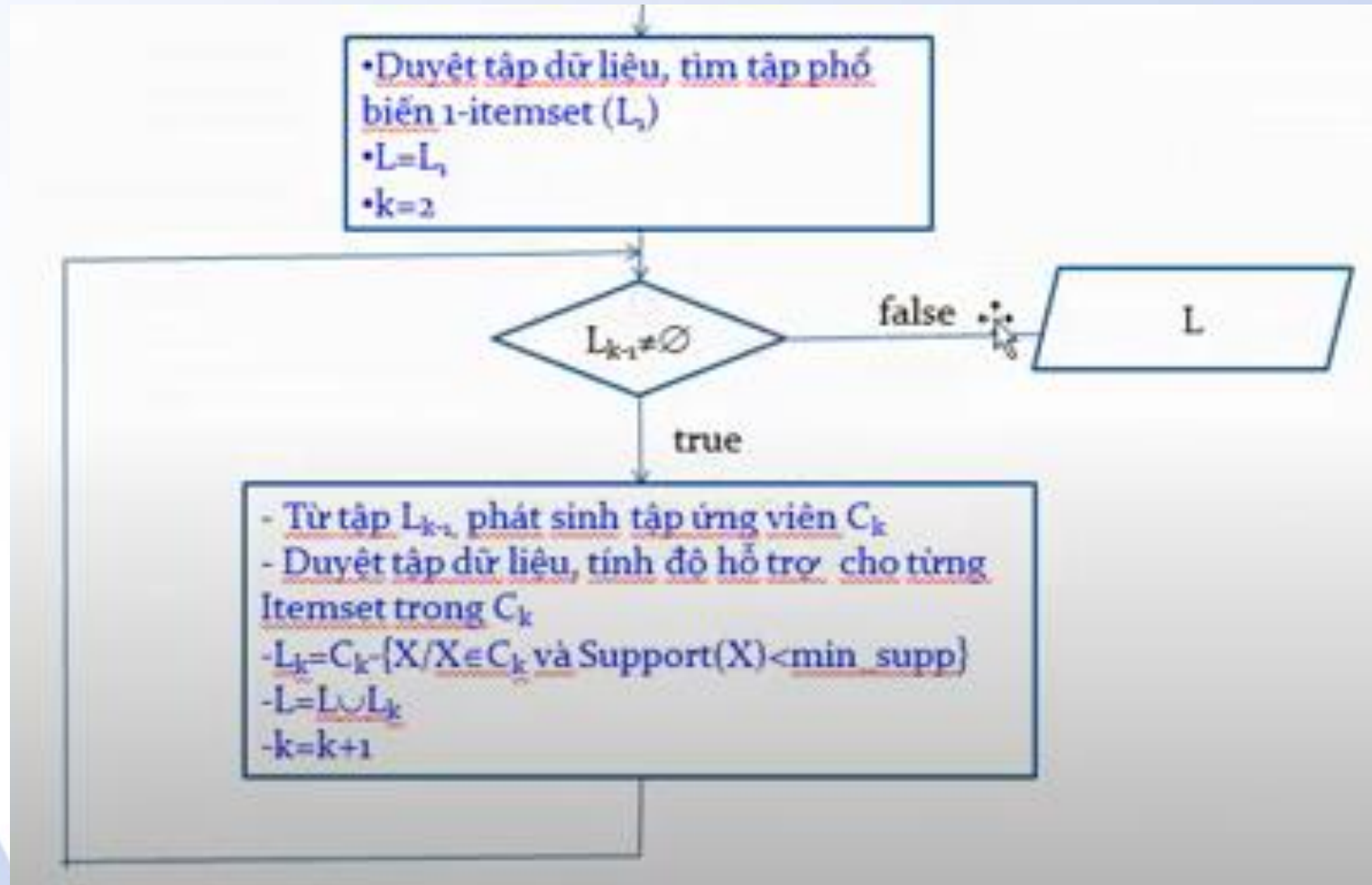
Thuật giải

Có 2 bước chính:

- ❖ B1: Sinh ra tập Itemset phổ biến.
- ❖ B2: Tìm ra luật.
- ❖ Apriori dùng để giảm các thuộc tính, loại bỏ các thuộc tính không cần thiết.
- ❖ Apriori cần 2 tham số là minsup và minconf, minsup dùng để sinh ra tập các itemsets phổ biến còn minconf dùng để tìm luật.
- ❖ Input: CSDL giao dịch D, ngưỡng minsup.
- ❖ Output: Các tập phổ biến.

Thuật giải Apriori

Thuật giải



Thuật giải Apriori

Thuật giải

- ❖ *Tính chất Apriori*: Tất cả các tập con không rỗng của một tập mục thường xuyên cũng thường xuyên.
- ❖ Tính chất Apriori được sử dụng để tìm L_k dựa trên L_{k-1} thông qua quy trình 2 bước (kết nối và loại bỏ)

Thuật giải Apriori

Thuật giải

Algorithm: Apriori. Find frequent itemsets using an iterative level-wise approach based on candidate generation.

Input:

- D , a database of transactions;
- min_sup , the minimum support count threshold.

Output: L , frequent itemsets in D .

Method:

```
(1)   $L_1 = \text{find\_frequent\_1-itemsets}(D);$ 
(2)  for ( $k = 2; L_{k-1} \neq \phi; k++$ ) {
(3)     $C_k = \text{apriori\_gen}(L_{k-1});$ 
(4)    for each transaction  $t \in D$  { // scan  $D$  for counts
(5)       $C_t = \text{subset}(C_k, t);$  // get the subsets of  $t$  that are candidates
(6)      for each candidate  $c \in C_t$ 
(7)         $c.\text{count}++;$ 
(8)    }
(9)     $L_k = \{c \in C_k | c.\text{count} \geq min\_sup\}$ 
(10) }
(11) return  $L = \cup_k L_k;$ 
```


Thuật giải Apriori

Thuật giải Apriori_Gen

- ❖ Mục đích: tìm C_k – sinh các tập mục ứng cử là ứng cử viên cho các tập $k_itemset$ và xóa các tập mục không phổ biến theo điều kiện minsup.
- ❖ Input: L_{k-1} , tập mục $(k-1)_itemset$ phổ biến. minsup, độ hỗ trợ tối thiểu.
- ❖ Output: C_k , tập ứng cử viên k -itemset

Thuật giải Apriori

Thuật giải Apriori_Gen

```
procedure apriori_gen( $L_{k-1}$ :frequent  $(k-1)$ -itemsets)
(1)   for each itemset  $l_1 \in L_{k-1}$ 
(2)     for each itemset  $l_2 \in L_{k-1}$ 
(3)       if  $(l_1[1] = l_2[1]) \wedge (l_1[2] = l_2[2])$ 
            $\wedge \dots \wedge (l_1[k-2] = l_2[k-2]) \wedge (l_1[k-1] < l_2[k-1])$  then {
(4)          $c = l_1 \bowtie l_2$ ; // join step: generate candidates
(5)         if has_infrequent_subset( $c, L_{k-1}$ ) then
(6)           delete  $c$ ; // prune step: remove unfruitful candidate
(7)         else add  $c$  to  $C_k$ ;
(8)       }
(9)   return  $C_k$ ;

procedure has_infrequent_subset( $c$ : candidate  $k$ -itemset;
                                $L_{k-1}$ : frequent  $(k-1)$ -itemsets); // use prior knowledge
(1)   for each  $(k-1)$ -subset  $s$  of  $c$ 
(2)     if  $s \notin L_{k-1}$  then
(3)       return TRUE;
(4)   return FALSE;
```

Thuật giải Apriori

Ví dụ

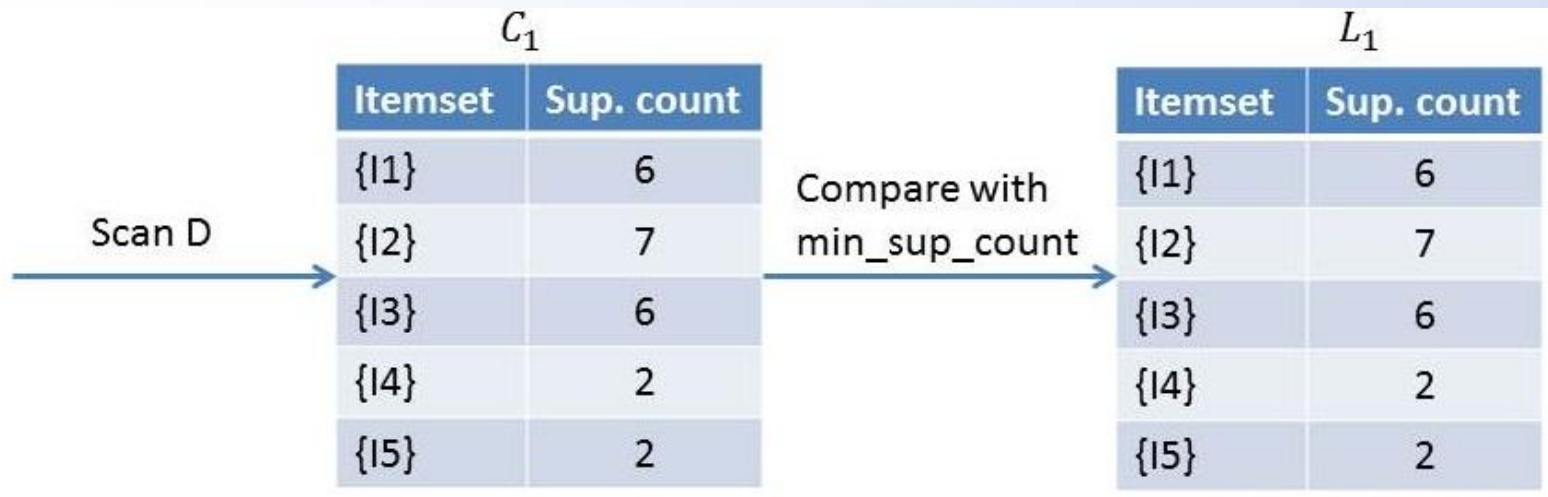
- Giả sử thiết lập giá trị $min_sup_count = 2$
- Tương ứng với $min_sup = 2/9 = 22\%$
- Tập các 1-itemset L_1 xác định bằng cách đếm tần suất xuất hiện trong cơ sở dữ liệu giao dịch.

TID	Danh mục
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

Thuật giải Apriori

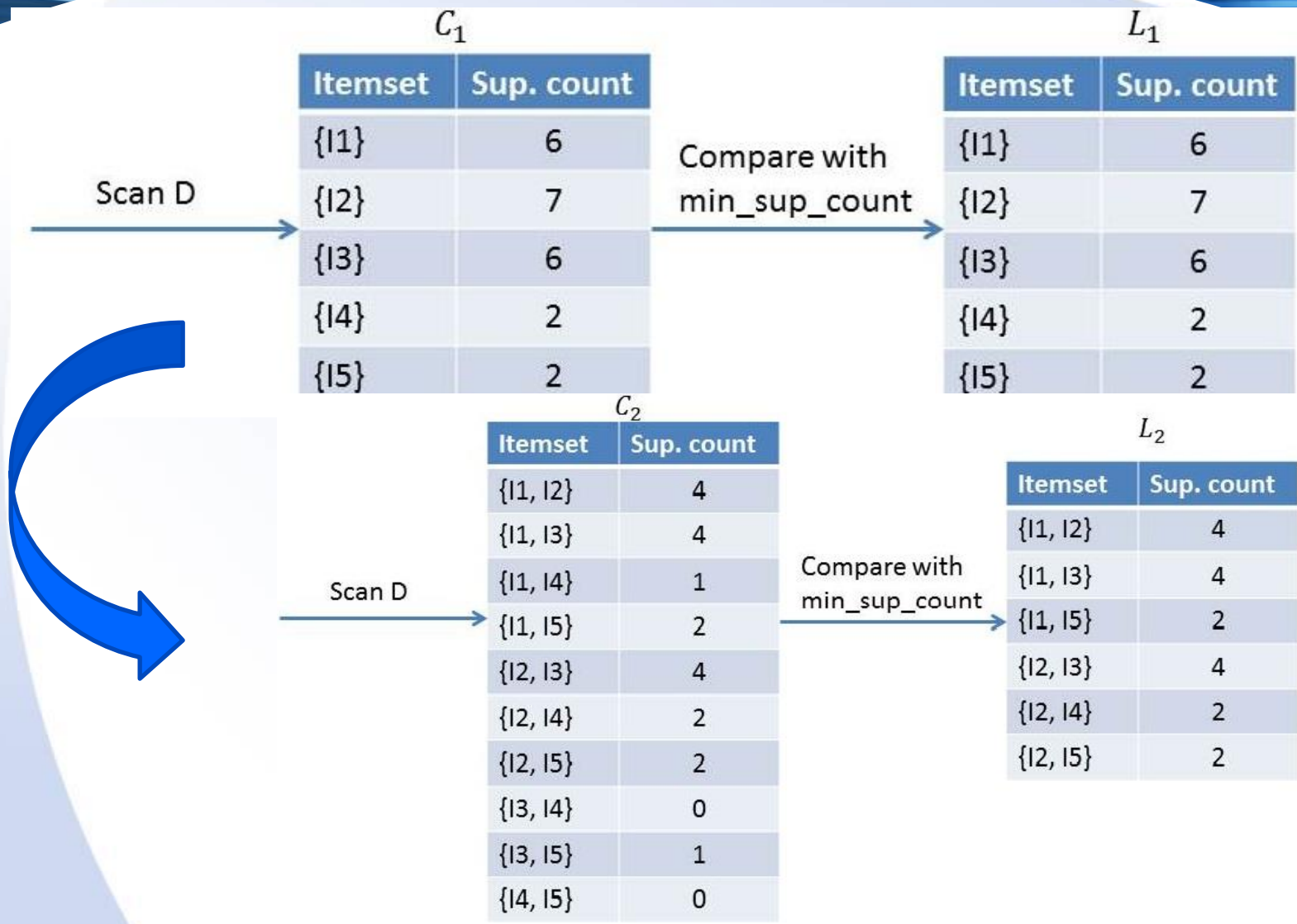
Ví dụ

TID	Danh mục
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3



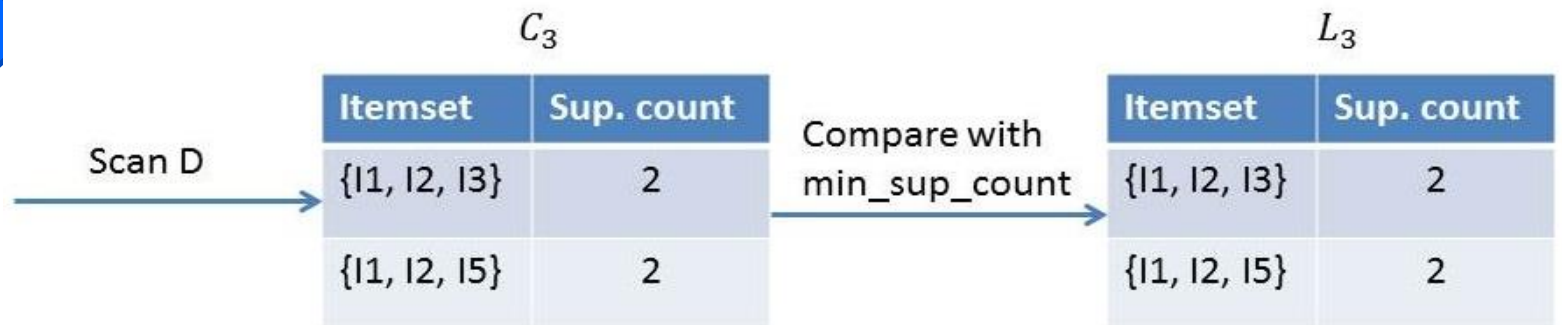
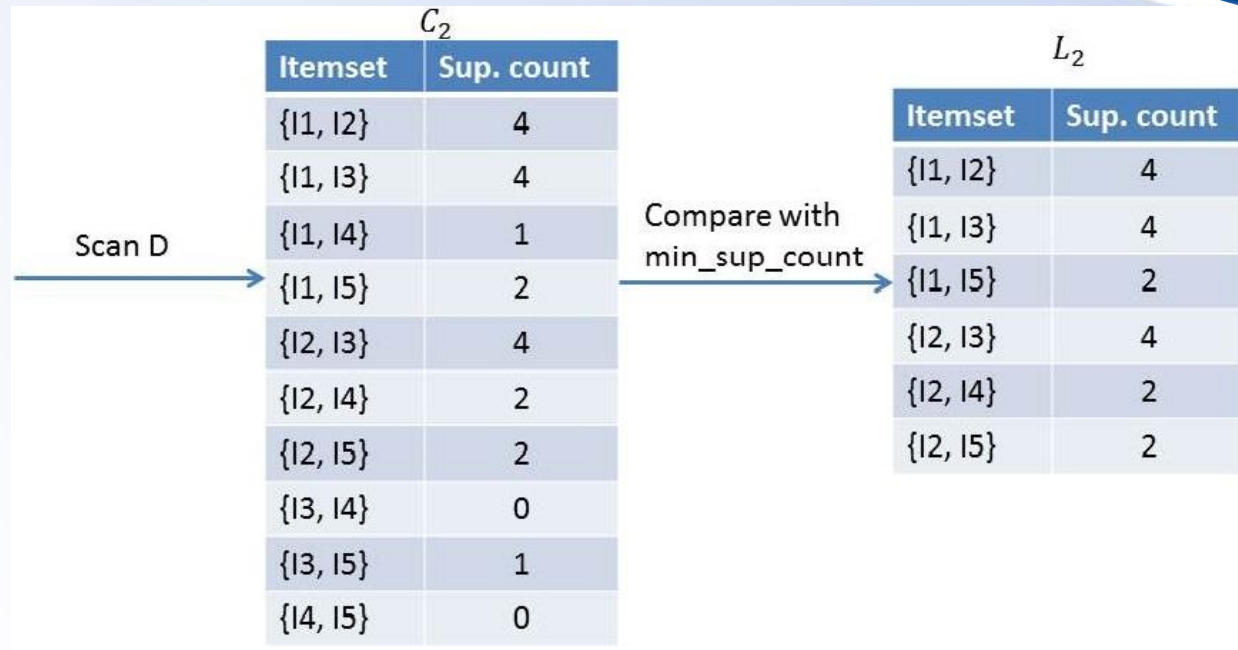
Thuật giải Apriori

Ví dụ



Thuật giải Apriori

Ví dụ



Thuật giải AprioriTID

- ❖ Thuật giải này cũng sử dụng hàm Apriori_Gen để sinh ra các tập ứng cử viên cho mỗi giai đoạn.
- ❖ Không dùng CSDL D để đếm các support với các giai đoạn $k > 1$ mà sử dụng tập C^k .
- ❖ Mỗi phần tử của C^k có dạng $\langle Tid, \{X_k\} \rangle$, trong đó mỗi X_k là một tập phổ biến $k_itemset$ tiềm năng trong giao dịch Tid .
- ❖ Khi $k = 1$, C^k tương ứng với D, trong đó mỗi item i được coi là một itemset $\{i\}$.
- ❖ Với $k > 1$, C^k được sinh ra bởi $C^{k+} = \langle t.Tid, C_t \rangle$. Phần tử của C^k tương ứng với giao dịch t là $\langle t.Tid, \{c \in C_t \mid c \text{ chứa trong } t\} \rangle$.

Thuật giải AprioriTID

- ❖ Nếu một giao dịch không chứa bất kỳ tập ứng viên $k_itemset$ nào thì C^k sẽ không có một điểm vào nào cho giao dịch này.
- ❖ Số lượng điểm vào trong C^k có thể nhỏ hơn số giao dịch trong CSDL, đặc biệt với k lớn.
- ❖ Với các giá trị k khá lớn, mỗi điểm vào có thể nhỏ hơn giao dịch tương ứng vì một số ứng viên đã được chứa trong giao dịch.
- ❖ Với các giá trị k nhỏ, mỗi điểm vào có thể lớn hơn giao dịch tương ứng vì một điểm vào trong C^k bao gồm tất cả các ứng viên $k_itemset$ được chứa trong giao dịch.

Thuật giải AprioriTID

Ý tưởng thuật giải

- ❖ Tạo các tập 1_itemset: từ các item trên CSDL
- ❖ Tính độ hỗ trợ cho từng item đưa vào CSDL đã mã hóa, loại đi các item có độ hỗ trợ nhỏ hơn minsup.
- ❖ Tạo các tập 2_itemset: tính độ hỗ trợ cho tập gồm 2 item dựa vào tập C_{1_N} loại đi các item có độ hỗ trợ nhỏ hơn minsup.
- ❖
- ❖ Tạo các tập k_itemset: tính độ hỗ trợ cho tập gồm k item dựa vào tập C_{k-1_N} loại đi các item có độ hỗ trợ nhỏ hơn minsup.

Thuật giải AprioriTID

Ý tưởng thuật giải

- ❖ Các khái niệm trong AprioriTid cũng tương tự như Apriori, chỉ thêm tập C_{k_N} .
- ❖ Là cải tiến của thuật giải Apriori, ở chỗ: sau khi dùng CSDL D đếm support cho các itemset 1 item tạo ra L_1 , thuật giải tạo thêm tập C_{k_N} , dựa vào tập C_{k_N} này tính support cho các itemset từ 2 item trở lên tương ứng.

Thuật giải AprioriTID

Thuật giải

Input: Cơ sở dữ liệu giao dịch D, ngưỡng minsup.

Output: Các tập phổ biến.

Procedure AprioriTid

1. Quét CSDL D để tìm các tập mục phổ biến 1_itemset, L_1 .
2. $C_1' = \text{CSDL } D$
3. for($k=2$; $L_{k-1} \neq \emptyset$; $k++$) {
4. $C_k = \text{Apriori_Gen}(L_{k-1}, \text{minsup})$; //sinh ứng cử k-itemset
5. $C_k' = \emptyset$
6. for(mỗi một tác vụ $t \in C_{k-1}$) {
7. $C_t = \{c \in C_k \mid (c - c[k]) \in t.\text{Set_of_ItemSets} \wedge (c - c[k-1] \in t.\text{Set_of_ItemSets})\}$
//xác định tập ứng viên trong C_k chứa trong giao dịch với định danh t.Tid
8. for(mỗi một ứng cử $c \in C_t$)
9. $c.\text{count}++$;
10. if($C_t \neq \emptyset$) then $C_k += \langle t.\text{Tid}, C_t \rangle$; //thêm item t vào bảng C_k'
11. $L_k = \{c \in C_k \mid c.\text{count} \geq \text{minsup}\}$
12. return $L = \bigcup_{i=1}^k L_i$

Thuật giải AprioriTID

Nhận xét

- ❖ Apriori tốt hơn Apriori_Tid [Agrawal 1994] khi tập CSDL lớn
- ❖ Trong trường hợp tập C_k tương đối nhỏ thì Apriori_Tid thực hiện tốt hơn Apriori.

Thuật giải AprioriTID

Ví dụ: min_sup_count=2

TID	Các mục dữ liệu
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

TID	Tập các tập mục dữ liệu
100	{{1}, {3}, {4}}
200	{{2}, {3}, {5}}
300	{{1}, {2}, {3}, {5}}
400	{{2}, {5}}

Tập mục	Số tác vụ hỗ trợ
{1}	2
{2}	3
{3}	3
{5}	3

C_2

Tập mục	Số tác vụ hỗ trợ
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

C'_2

TID	Tập các tập mục dữ liệu
100	{{1 3}}
200	{{2 3}, {2 5}, {3 5}}
300	{{1 2}, {1 3}, {1 5}, {2 3}, {2 5}, {3 5}}
400	{{2 5}}

L_2

Tập mục dữ liệu	hỗ trợ
{1 3}	2
{2 3}	3
{2 5}	3
{3 5}	2

Thuật giải AprioriTID

Ví dụ

C_3		C'_3		L_3	
<i>Tập mục dữ liệu</i>	<i>Số tác vụ hỗ trợ</i>	<i>TID</i>	<i>Tập các tập mục dữ liệu</i>	<i>Tập mục dữ liệu</i>	<i>Số tác vụ hỗ trợ</i>
$\{2\ 3\ 5\}$	2	200 300	$\{\{2\ 3\ 5\}\}$ $\{\{2\ 3\ 5\}\}$	$\{2\ 3\ 5\}$	2

Thuật giải FP_Growth

- ❖ FP_Growth sử dụng một cấu trúc dữ liệu gọi là FP_tree (Frequent Pattern tree).
- ❖ FP_tree là một thể hiện cô đọng các thông tin có liên quan đến thông tin thể hiện tính thường xuyên của các tập mục trong CSDL.
- ❖ Mỗi nhánh của cây FP_tree thể hiện một tập mục phổ biến, và các nút dọc theo các nhánh được lưu trữ theo thứ tự giảm dần của tính phổ biến tương ứng với các mục, các mục ở lá của cây có tính phổ biến thấp nhất.

Thuật giải FP_Growth

- ❖ Cây FP_tree có một bảng header kết hợp với nó.
- ❖ Bảng header lưu các mục cùng với số lần xuất hiện của nó trong CSDL theo thứ tự giảm dần của tính phổ biến (mỗi mục chiếm một dòng của bảng).
- ❖ Mỗi mục của bảng chứa một nút đầu danh sách liên kết với tất cả các nút của cây FP_tree mà nút đó có tên trùng với tên của nó.
- ❖ FP_growth chỉ duyệt CSDL 2 lần để khai phá tất cả các tập mục phổ biến. Lần 1 để xác định tần xuất của từng tập mục trong CSDL. Lần 2 để xây dựng cây FP_tree.

Thuật giải FP_Growth

1) Cấu trúc cây FP

❖ Cấu trúc của cây FP_tree:

- Gốc cây được tạo với nhãn là null.
- Các liên kết trên cây: Liên kết giữa nút có tên mục giống nhau.
- Cấu trúc của một nút (trừ nút gốc) gồm các thành phần:
 - Tên mục
 - Bộ đếm (counter)
 - Liên kết (node link) đến nút tiếp theo trên cây có cùng tên mục

Thuật giải FP_Growth

2) Xây dựng cây FP

- ❖ Quá trình xây dựng cây FP gồm 2 bước:
- ❖ Bước 1: Quét CSDL lần 1, tìm tất cả các mục và tần xuất của nó.
 - Chèn các mục có độ hỗ trợ lớn hơn hoặc bằng độ hỗ trợ tối thiểu cùng với tần xuất của nó vào bảng Header theo thứ tự giảm dần của tần xuất.
- ❖ Bước 2: Quét CSDL lần 2, mỗi một giao dịch được quét.
 - Loại bỏ mục có độ hỗ trợ nhỏ hơn minsup và sắp xếp lại các mục theo thứ tự giảm dần của tần xuất.

Thuật giải FP_Growth

- ❖ Nếu phần đầu của tập mục GD này không trùng với mọi phần đầu của GD đã xét thì nó được chèn vào cây như một nhánh và bộ đếm của mỗi nút ban đầu là 1. Tạo liên kết từ bảng Header đến các mục tương ứng.
- ❖ Nếu tập mục của GD đang xét, có phần đầu trùng với phần đầu của GD nào đó, mà GD này đã được tạo nhánh trên cây, thì phần đầu của GD đang xét sẽ được chia sẻ với phần đầu nhánh thể hiện GD đã xét, với mỗi nút trên đoạn nhánh chia sẻ bộ đếm được tăng lên 1 đơn vị, phần còn lại với mỗi mục sẽ được tạo một nút và được nối liền với nhánh được chia sẻ ở phần đầu.

Thuật giải FP_Growth

- ❖ Bộ đếm lưu trữ số giao dịch thể hiện bởi nhánh cây xuất phát từ nút gốc đến nút đó.
- ❖ Cây FP_tree chứa đựng tất cả các thông tin về tần xuất của các mục trong CSDL, việc khai phá CSDL lúc này trở về khai phá cây FP_tree.

Thuật giải FP_Growth

Thuật giải

Input: Cơ sở dữ liệu giao dịch D.

Output: Cây FP_tree

Procedure Insert_Tree(string[p], Tree có gốc T)

1) If T có nút con N mà N.itemname = p Then N.Count ++

2) ELSE

3) Tạo nút mới N

4) N.itemname = p, N.Count = 1;

5) Liên kết bảng từ p đến N

6) If p khác rỗng Then

7) Insert_tree(N, p);

P: là mục đầu tiên trong danh sách các tập mục P của giao dịch đang xét.

Thuật giải FP_Growth

3) Phương pháp tìm tập phổ biến từ cây FP

- ❖ Từ cấu trúc cây FP, xét một số thuộc tính quan trọng:
 - HeadNodeLink: Nhờ thuộc tính này, khi xét các item phổ biến trong L1, ta có thể truy xuất đến vị trí đầu tiên của nút trong cây có tên giống với tên L1.item.
 - NodeLink: Nhờ thuộc tính này nên với bất kỳ item phổ biến i thuộc L1, ta có thể xác định được tất cả các tập phổ biến có chứa item i dựa vào các liên kết của nút i trong cây.

Thuật giải FP_Growth

- ❖ Thuật giải tìm các tập phổ biến từ cây FP
- ❖ Input: Cây FP.
- ❖ Output: Tập các tập phổ biến.
- ❖ Procedure FrequentItem_FPTree(Tree T)
 - 1) Duyệt L1 theo thứ tự các item có độ hỗ trợ từ thấp đến cao (duyệt ngược lại trong L1)
 - 2) Với mỗi item $i \in L1$
 - 3) TìmDuongDi (i, SoDD);// Có được MangDuongDi, SoDD
 - 4) TimTapPhoBien (i, MangDuongDi, SoDD)

Thuật giải FP_Growth

- ❖ Thủ tục TimDuongDi
- ❖ Mục đích: Tìm tất cả đường đi trong cây có chứa item i ,
- ❖ Input: Item I , $SoDD = 0$
- ❖ Output: MangDuongDi: là mảng các đường đi trong cây FP có chứa item i .
- ❖ SoDD: số đường đi trong cây có chứa item i .

Thuật giải FP_Growth

Procedure TimDuongDi (Item i, string MangDuongDi)

- 1) VT = i.HeadNodeLink; //Từ liên kết HeadNodeLink, xác định được vị trí đầu tiên //của nút có tên item giống i.Item
- 2) N = nút ở vị trí hiện hành; //Di chuyển đến nút ở vị trí VT trong cây
- 3) Link = 1
- 4) j = 1
- 5) Trong khi Link \neq 0 {
- 6) Support = N.Support
- 7) DuongDi = \emptyset
- 8) N1 = N
- 9) Duyệt ngược cây FP từ vị trí VT {
- 10) If N1.TenNutCha \neq <Null> then
- 11) DuongDi = DuongDi + N1.TenNutCha
- 12) VT = Tim(N1.TenNutCha, VT); //Tìm ngược tự vị trí VT trong cây có tên item là //N1.TenNutCha, trả về vị trí tìm được
- 13) N1 = nút ở vị trí hiện hành)
- 14) Else
- 15) Thoát khỏi vòng lặp duyệt cây}
- 16) MangDuongDi(j) = DaoNguoc(DuongDi)
- 17) j = j+1
- 18) Link = N.NodeLink; //Tìm nút gần nhất trong cây có tên cùng tên với nút hiện hành
- 19) VT = Link
- 20) Di chuyển đến nút ở vị trí VT trong cây
- 21) N = nút ở vị trí hiện hành}
- 22) SoDD = j-1

Thuật giải FP_Growth

- ❖ Thủ tục TimTapPhoBien(Item i, string MangDuongDi, int soDD)
- ❖ Input: i: Item phổ biến một phần tử i. MangDuongDi: các đường đi trong cây chứa item i. SoDD: số đường đi trong cây chứa item i.
- ❖ Output: Tập các tập phổ biến.

Thuật giải FP_Growth

Procedure TimTapPhoBien(Item i, string MangDuongDi, int soDD)

- 1) $j = 1$
- 2) $t = 1$
- 3) while $j \leq \text{SoDD}$ {
- 4) Với mỗi kết hợp j phần tử trong MangDuongDi {
- 5) PhanTuChung = **TimPhanTuChung** (i, soPTChung)
- 6) If Support(PhanTuChung) \geq Minsup then
- 7) $t = 1$
- 8) do while $t \leq \text{SoPTChung}$ {
- 9) Với mỗi kết hợp t phần tử $k = \{k_1, k_2, \dots, k_t\}$
- 10) Tạo ra tập phổ biến(k, i)}}
- 11) $j = j + 1$ }

Thuật giải FP_Growth

❖ Thủ tục TimPhanTuChung

- Tìm phần tử chung giữa j phần tử trong kết hợp, nếu có item giống nhau thì $\text{PhanTuChung} = \text{PhanTuChung} + \text{Item}$,
- Support của PhanTuChung bằng tổng Support của các item trong kết hợp j phần tử này.

❖ Nhận xét:

- ❖ Ưu điểm của cây FP: tạo khả năng UD cho CSDL lớn,
- ❖ Giảm thời gian thực hiện do:
 - Cấu trúc dữ liệu đơn giản, đầy đủ.
 - Giảm số lần duyệt cơ sở dữ liệu.
 - Xây dựng và tính toán trên cây FP là cơ bản.

Thuật giải FP_Growth

Ví dụ

- Giả sử thiết lập giá trị $min_sup = 50\%$


TID	items bought
100	{f, a, c, d, g, i, m, p}
200	{a, b, c, f, l, m, o}
300	{b, f, h, j, o}
400	{b, c, k, s, p}
500	{a, f, c, e, l, p, m, n}

Thuật giải FP_Growth

Ví dụ

- ❖ Bước 1 - Nén cơ sở dữ liệu giao dịch gốc vào cây FP-tree
 1. Quét cơ sở dữ liệu một lần, tìm các tập phổ biến *1-itemsets*.
 2. Sắp xếp các tập phổ biến tìm được theo thứ tự giảm dần của độ phổ biến (tần số).

TID	items bought
100	{f, a, c, d, g, i, m, p}
200	{a, b, c, f, l, m, o}
300	{b, f, h, j, o}
400	{b, c, k, s, p}
500	{a, f, c, e, l, p, m, n}


 $minSup=0.5$

item	frequency
f	4
c	4
a	3
b	3
m	3
p	3

Sắp xếp các sản phẩm
theo thứ tự giảm dần của
độ tần số.

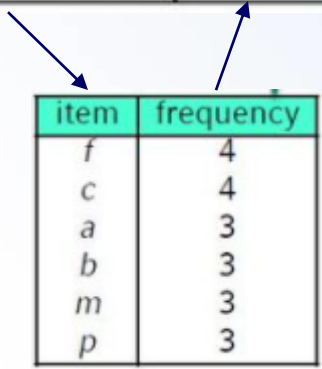
Thuật giải FP_Growth

Ví dụ

- ❖ Bước 1 - Nén cơ sở dữ liệu giao dịch gốc vào cây FP-tree
- 3. Quét lại cơ sở dữ liệu lần 2, xây dựng một cây FP-tree bắt đầu với hạng mục phổ biến nhất trong mỗi giao dịch.

TID	items bought	(ordered) frequent items
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}
200	{a, b, c, f, l, m, o}	{f, c, a, b, m}
300	{b, f, h, j, o}	{f, b}
400	{b, c, k, s, p}	{c, b, p}
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}

Với mỗi giao dịch chỉ giữ lại các hạng mục phổ biến ở trong giao dịch đó và sắp xếp chúng theo thứ tự giảm dần của tần số



item	frequency
f	4
c	4
a	3
b	3
m	3
p	3

Thuật giải FP_Growth

Ví dụ

- ❖ Bước 1 - Nén cơ sở dữ liệu giao dịch gốc vào cây FP-tree
- 3. Quét lại cơ sở dữ liệu lần 2, xây dựng một cây FP-tree bắt đầu với hạng mục phổ biến nhất trong mỗi giao dịch.

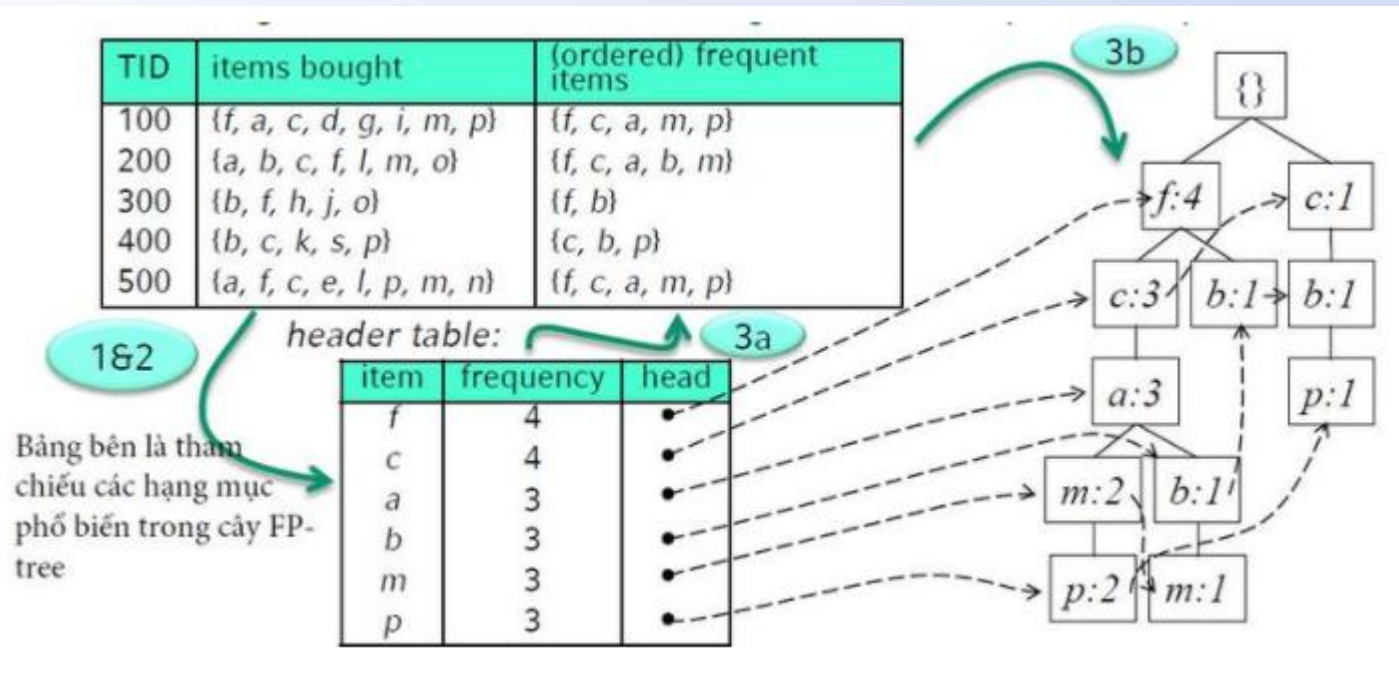
Với mỗi giao dịch mới xây dựng lại ở trên, xây dựng một đường đi tương ứng trong FP-tree:

- Nếu một đường đi có tiền tố chung tồn tại: tăng tần số của các nút trên đường đi này và nối thêm hậu tố
- Ngược lại, tạo một nhánh mới

Thuật giải FP_Growth

Ví dụ

- ❖ Bước 1 - Nén cơ sở dữ liệu giao dịch gốc vào cây FP-tree
- 3. Quét lại cơ sở dữ liệu lần 2, xây dựng một cây FP-tree bắt đầu với hạng mục phổ biến nhất trong mỗi giao dịch.



Thuật giải FP_Growth

Ví dụ

- ❖ Bước 2 - Các bước chính để khai thác các tập phổ biến trên cây FP- tree - cây FP -tree có điều kiện
- Duyệt từng hạng mục phổ biến (*1-itemsets*) theo thứ tự tăng dần của tần số (*p, m, b, a, c, f*). Với mỗi hạng mục, xây dựng cơ sở mẫu điều kiện và các cây *FP-tree có điều kiện* tương ứng của nó:

$\{item\} item \text{ in } (1\text{-itemsets})(1\text{-itemsets})$

$\{\backslash\text{Rightarrow}\} \Rightarrow \{conditional\} conditional \{pattern\text{-base}\}$

$pattern\text{-base} \{\backslash\text{Rightarrow}\} \Rightarrow conditional conditional \{FP\text{-Tree}\} FP\text{-Tree}$

Thuật giải FP_Growth

Ví dụ

- ❖ Bước 2 - Các bước chính để khai thác các tập phổ biến trên cây FP- tree - cây FP -tree có điều kiện
 - Bắt đầu với hạng mục p , cơ sở mẫu điều kiện của nó là tất cả các đường đi tiền tố của cây FP-Tree khi duyệt từ nút gốc $\{\}$ đến nút p , các đường đi này chính là $fcam:2$ và $cb:1$ (trong đó số theo sau là số lần xuất hiện của nút p tương ứng với mỗi tiền tố đó).
 - Xây dựng cây ***FP-Tree có điều kiện*** từ mẫu trên bằng cách trộn tất cả các đường đi và giữ lại các nút có tần số ≥ 3 do $min_sup = 0.5$

Thuật giải FP_Growth

Ví dụ

- ❖ Bước 2 - Các bước chính để khai thác các tập phổ biến trên cây FP- tree - cây FP -tree có điều kiện

Item	Cơ sở mẫu điều kiện	FP-Tree điều kiện	Các mẫu phổ biến
p	{fcam:2, cb:1}	{c:3}-p	p, cp
m	{fca:2, fcab:1}	{f:3, c:3, a:3}-m	m, fm, cm, am, fcm, cam, fam, fcam
b	{fca:1, f:1, c:1}	∅	b
a	{fc:3}	{f:3, c:3}-a	a, fa, ca, fca
c	{f:3}	{f:3}-c	c, fc
f	∅	∅	f

Khái phá các luật kết hợp theo ngưỡng MINCONF

- ❖ **Ý tưởng:** Ứng với một frequent itemset l , tìm những tập con khác rỗng của l .
- ❖ Với tập con a , đưa ra luật dạng $a \rightarrow (l - a)$ nếu tỉ số $\text{support}(l)/\text{support}(a) \geq \text{minconf}$.
- ❖ Mọi tập con của a đều có độ hỗ trợ lớn hơn hoặc bằng độ hỗ trợ của a .
- ❖ **VD:** AB có support là 5, thì A là con của AB phải có độ hỗ trợ ≥ 5 .
- ❖ Độ tin cậy của luật dạng $a \rightarrow (l - a)$ là:
$$\text{Support}(l)/\text{support}(a) \geq \text{minconf}.$$
- ❖ Nếu tập con a của l không đưa ra được luật thỏa minconf thì các tập con của a cũng không thể tạo ra một luật thỏa minconf được.

Thuật giải 1: Simple algorithm

- ❖ Cải tiến thủ tục xử lý bằng cách sinh ra các tập con của mục lớn theo kiểu đệ qui ưu tiên độ sâu.
 - ❖ VD: với tập mục ABCD, đầu tiên chúng ta xét tập con ABC, sau đó đến AB,...
- ❖ **Nếu tập a không sinh ra được luật thì không cần xét đến các tập con của a nữa** (nếu một luật không thoả mãn với tập cha a thì cũng không thoả mãn với tập con của nó)
 - ❖ Chẳng hạn: nếu luật $ABC \rightarrow D$ không đủ độ tin cậy thì ta không cần xét đến luật $AB \rightarrow CD$.

Thuật giải 1: Simple algorithm

- ❖ Điều này có thể CM như sau:
- ❖ Nếu luật $a \rightarrow (1-a)$ không thoả mãn độ tin cậy, tức là: $\text{conf}(a \rightarrow (1-a))$ nhỏ hơn minconf , thế thì với bất kỳ tập con b nào của a ta có:
- ❖ Vì $b \subset a$ nên $\text{supp}(b) \geq \text{supp}(a)$, do vậy:

$$\text{conf}(b \rightarrow (1-b)) = \frac{\text{supp}(l)}{\text{supp}(b)} \leq \frac{\text{supp}(l)}{\text{supp}(a)} = \text{conf}(a \rightarrow (1-a)) < \text{minconf}$$

- ❖ Tức là độ tin cậy của luật $b \rightarrow (1-b)$ cũng nhỏ hơn minconf

Thuật giải 1: Simple algorithm

Thuật giải simple có thể mô tả như sau:

\forall frequent $k_itemsets$ lk , $k \geq 2$ do

call GenRules(lk , lk)

Thủ tục GenRules

Mục đích: phát sinh tất cả các luật hợp lệ dạng $a \sim \Rightarrow (lk - a \sim)$, với $a \sim \subset am$

Procedure GenRules (lk : frequent $k_itemset$, am : frequent $m_itemset$)

1) $A = \{(m-1)_itemsets\ am-1 \mid am-1 \subset am\}$

2) Forall $am-1 \in A$ do begin

3) $conf = support(lk) / support(am-1)$

4) if($conf \geq minconf$) then begin

5) output $am-1 \rightarrow (lk - am-1)$ //với confidence = $conf$ và support = $sup(lk)$

6) if($m-1 > 1$) then

7) call GenRules(lk , $am-1$)

8) end

9) end

Thuật giải 2: Fast algorithm

- ❖ Thuật giải 2 là cải tiến của thuật giải 1.
 - ❖ Nếu xảy ra luật với tập con thì cũng xảy ra luật với tập cha.
 - VD: nếu luật $AB \rightarrow CD$ có đủ độ tin cậy thì luật $ABC \rightarrow D$ cũng đủ độ tin cậy.
- 1) forall frequent k_itemset L_k , $k \geq 2$
 - 2) $H1 = \{\text{Tập vế phải của các luật có 1 item ở vế phải}\}$
 - 3) Call Ap_GenRule(L_k , $H1$)
 - 4) end

Thuật giải 2: Fast algorithm

Procedure Ap_GenRule (Lk: Frequent k_itemset, Hm: tập về phải của các luật có m item ở về phải)

if $(m+1 < k)$ then

$H_{m+1} = \text{Apriori_Gen}(H_m)$

 với mọi $hm+1 \in H_{m+1}$

$\text{conf} = \text{support}(L_k) / \text{support}(L_k - hm+1)$

 if $(\text{conf} \geq \text{minconf})$

 Xuất ra luật $(L_k - hm+1) \rightarrow (hm+1)$ với support là $\text{support}(L_k)$,
 confidence là Conf

 else

 xóa $hm+1$ khỏi H_{m+1}

 endif

 gọi hàm $\text{GenRule}(L_k, H_1)$

endif

Thuật giải 3: Tìm luật đơn giản

- ❖ Nếu một luật chứa tập a ở vế phải thỏa ngưỡng minconf thì mọi luật chứa $a \sim$ ở vế phải cũng thỏa ngưỡng minconf với mọi $a \sim \subset a$
- ❖ NX: nếu phải tìm tất cả các luật kết hợp có thể có thì chỉ cần tìm những luật có 1 item ở vế phải là đủ.
- ❖ Tất cả các luật kết hợp có hơn 1 item ở vế phải đều có thể suy ra từ các luật có 1 item ở vế phải.

Thuật giải 3: Tìm luật đơn giản

- ❖ Ký hiệu s là tập luật gồm tất cả những luật kết hợp có 1 item ở vế phải thỏa ngưỡng minsup và minconf cho trước.
- ❖ Thuật giải tìm tập luật đơn giản S
- ❖ 1. Tìm tất cả các tập frequent itemset thỏa minsup.
- ❖ 2. Đối với từng frequent itemset $X: li_1, li_2, \dots, li_k$ kiểm tra tất cả các luật có vế phải có 1 thuộc tính $r: X - li_j \rightarrow li_j, j = 1 \dots k$. Nếu thỏa minconf thì cho ra luật r

Thuật giải 3: Tìm luật đơn giản

- ❖ Tập luật s chứa đựng tất cả thông tin của tập các luật AR, nhưng có kích thước bé hơn tập AR.
- ❖ Nên tìm tập luật đơn giản s (thay vì AR) vì:
- ❖ Số lượng luật cần lưu lại giảm đáng kể, thường giảm từ 10% - 50%.
- ❖ Giảm đáng kể thời gian và tài nguyên tiêu tốn trong lúc tìm luật khi chỉ tìm luật đơn giản.
- ❖ Mọi luật kết hợp đều có thể được suy dẫn từ tập luật đơn giản.
- ❖ Chỉ tập trung vào các luật ta quan tâm chứ không phải chìm ngập trong tập tất cả các luật kết hợp.

Loại luật thừa, tìm tập luật quan tâm

- ❖ Phương pháp dùng quy luật loại bỏ luật thừa
- ❖ Phương pháp lọc dùng mẫu đơn giản

Phương pháp dùng quy luật loại bỏ luật thừa

- ❖ Có ba tập luật cần quan tâm.
- ❖ Tập luật kết hợp
- ❖ $AR = \{X \Rightarrow Y \mid \sup(X \Rightarrow Y) \geq \text{minsup} \text{ và } \text{conf}(X \Rightarrow Y) \geq \text{minconf}\}$
- ❖ Đây là tất cả những luật có được do áp dụng thuật giải khi tìm luật kết hợp.

Phương pháp dùng quy luật loại bỏ luật thừa

- ❖ Tập luật đặc trưng
- ❖ $RR = \{ (X \Rightarrow Y) \in AR \mid \neg \exists (X' \Rightarrow Y') \in AR, (X = X') \wedge (X \cup Y \subset X' \cup Y') \vee (X \times X' \supset X \wedge Y = X' \cup Y') \}$.
- ❖ Với mọi luật $X \Rightarrow Y$ (được sinh ra từ itemset $X \cup Y$) đã có trong tập AR , tập luật RR gồm những luật trong tập AR loại bỏ các loại luật như sau:
- ❖ Luật sinh ra itemset $(X' \cup Y')$ chứa itemset $(X \cup Y)$ và có cùng vế trái với luật $X \Rightarrow Y$.
- ❖ Luật sinh ra từ $(X' \cup Y') = (X \cup Y)$ và luật có vế trái là con của X

Phương pháp dùng quy luật loại bỏ luật thừa

- ❖ Tập luật gồm các luật về trái nhỏ nhất, về phải lớn nhất
- ❖ $MMR = \{r: (X \Rightarrow Y) \in AR \mid \neg \exists r': (X' \Rightarrow Y') \in AR, r' \neq r \text{ và } X' \subseteq X \text{ và } Y' \supseteq Y\}$
- ❖ Với luật mọi luật $X \Rightarrow Y \in AR$, tập MMR gồm những luật trong tập AR loại bỏ luật có tính chất sau:
Luật có về trái là con của X và có về phải chứa Y.

Phương pháp dùng quy luật loại bỏ luật thừa

- ❖ Đối với ba tập luật trên, ta CM được mối quan hệ sau:
 $MMR \subseteq RR \subseteq AR$
- ❖ Thuật giải tìm tập luật MMR
 - $MMR = AR$
 - While ($\exists r': (X' \Rightarrow Y') \in AR, r' \neq r$ và $X' \subseteq X$ và $Y' \supseteq Y$)
 - $MMR = MMR - rhhhh$

Phương pháp lọc dùng mẫu đơn giản

- ❖ Lớp các luật IR (hoặc ngay cả các luật vô ích) có thể được mô tả bởi các mẫu (template). Mẫu là một sự tổng quát hóa một lớp các luật kết hợp.
- ❖ Một mẫu có dạng như sau: $A_1, \dots, A_k \Rightarrow A_{k+1}$
- ❖ A_i là tên thuộc tính hoặc tên lớp hoặc là một biểu thức có dạng $C+$ hoặc C^* với C là tên của một lớp.
 - $C+$ và C^* tương ứng là “một hoặc nhiều” và “0 hoặc nhiều” thể hiện của lớp C .
 - Luật: $B_1, \dots, B_h \Rightarrow B_{h+1}$ thỏa mẫu khi luật được xem là thể hiện của mẫu.

Phương pháp lọc dùng mẫu đơn giản

- ❖ Phương pháp này dùng cách biểu diễn luật trên sự phân loại mà người dùng định nghĩa dựa trên các thuộc tính của dữ liệu dùng để khai thác luật.
- ❖ Trong phương pháp này, người dùng tự nhập vào tiêu chuẩn của luật cần tìm thông qua mẫu thể hiện luật mà họ quan tâm.

Trao đổi, câu hỏi?