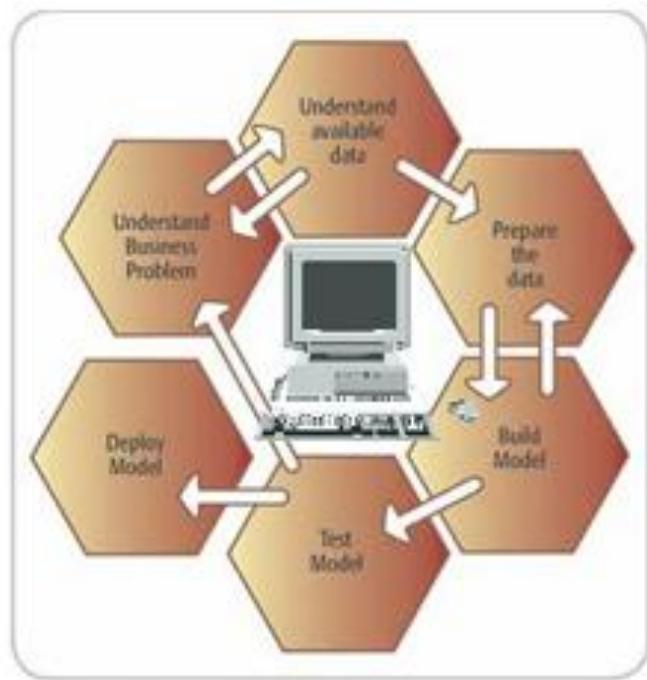




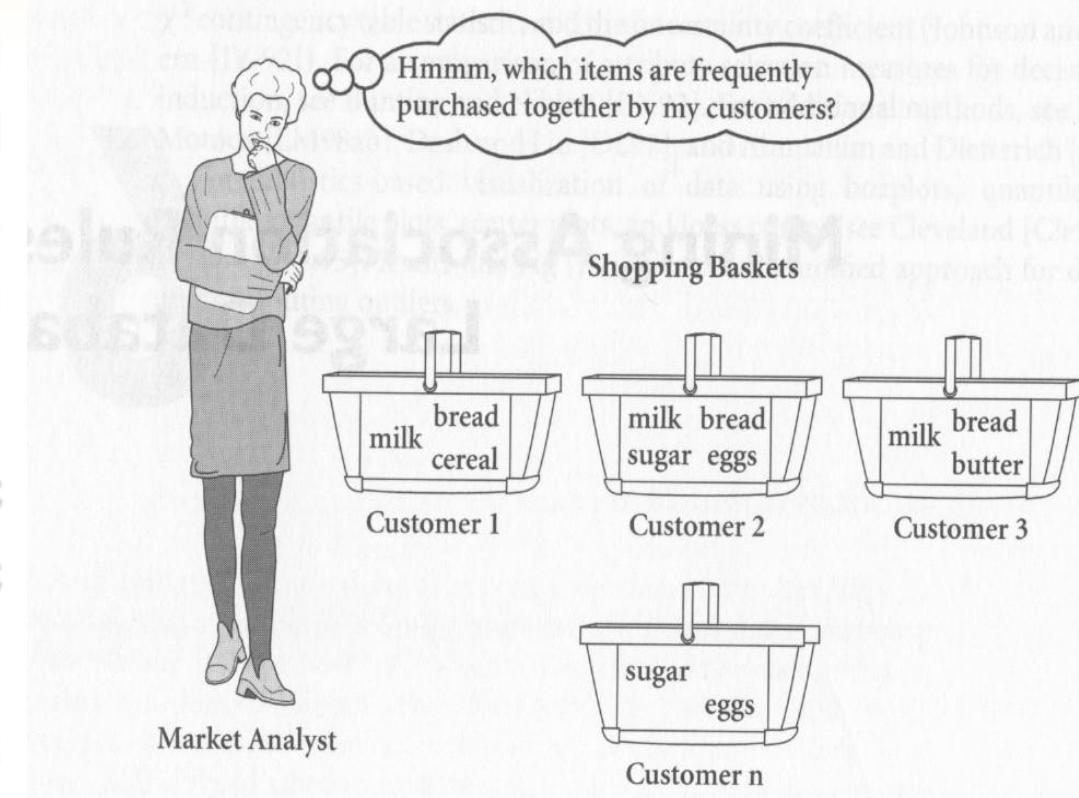
Chương 2

LUẬT KẾT HỢP (Association Rules)



Bài toán phân tích giỏ hàng

- Phân tích việc mua hàng của khách hàng bằng cách tìm ra những “mối kết hợp” giữa những mặt hàng mà khách đã mua.
- Bài toán được Agrawal thuộc nhóm nghiên cứu của IBM đưa ra vào năm 1994.



Luật kết hợp: Cơ sở

Khai phá luật kết hợp:

- Tìm tần số mẫu, mối kết hợp, sự tương quan, hay các cấu trúc nhân quả giữa các tập đối tượng trong các cơ sở dữ liệu giao tác, cơ sở dữ liệu quan hệ, và những kho thông tin khác.

Tính hiểu được: dễ hiểu

Tính sử dụng được: Cung cấp thông tin thiết thực

Tính hiệu quả: Đã có những thuật toán khai thác hiệu quả

Các ứng dụng:

- Phân tích bán hàng trong siêu thị, cross-marketing, thiết kế catalog, loss-leader analysis, gom cụm, phân lớp, ...

Luật kết hợp: Cơ sở

Định dạng thể hiện đặc trưng cho các luật kết hợp:

- khăn \Rightarrow bia [0.5%, 60%]
- mua:khăn \Rightarrow mua:bia [0.5%, 60%]
- “**Nếu** mua khăn **thì** mua bia trong 60% trường hợp. Khăn và bia được mua chung trong 0.5% dòng dữ liệu.”

Các biểu diễn khác:

- mua(x, "khăn") \Rightarrow mua(x, "bia") [0.5%, 60%]
- khoa(x, "CS") \wedge học(x, "DB") \Rightarrow điểm(x, "A") [1%, 75%]

Luật kết hợp: Cơ sở

khăn \Rightarrow bia [0.5%, 60%]

```
graph TD; A((khăn)) --- B((bia)); B --- C(( )); B --- D(( ));
```

“**NẾU** mua khăn
THÌ mua bia
trong 60% trường hợp
trên 0.5% dòng dữ liệu”

Tiền đề, về trái luật

Mệnh đề kết quả, về phải luật

Support, độ hỗ trợ/ủng hộ (“trong bao nhiêu phần trăm dữ liệu thì những điều ở về trái và về phải cùng xảy ra”)

Confidence, độ mạnh (“nếu về trái xảy ra thì có bao nhiêu khả năng về phải xảy ra”)

2.1 CỘC KHỐI NỘI MÔN

Cho $I = \{I_1, I_2, \dots, I_m\}$ là tập các đơn vị dữ liệu. Cho D là tập các giao tác, mỗi giao tác T là tập các đơn vị dữ liệu sao cho $T \subseteq I$

Định nghĩa 1: Ta gọi giao tác T chứa X , với X là tập các đơn vị dữ liệu của I , nếu $X \subseteq T$

Định nghĩa 2: Một luật kết hợp là một phép suy diễn có dạng $X \rightarrow Y$, trong đó $X \subset I$, $Y \subset I$ và $X \cap Y = \emptyset$

Định nghĩa 3: Ta gọi luật $X \rightarrow Y$ có mức xác nhận(support) là s trong tập giao tác D , nếu có s% giao tác trong D chứa $X \cup Y$.

Ký hiệu: $\text{Supp}(X \rightarrow Y) = s$

2.1 CÁC KHÁI NIỆM (Tiếp)

Định nghĩa 4: Ta gọi luật $X \rightarrow Y$ là có độ tin cậy c (Confidence) trên tập giao tác D,

Ký hiệu: $c = \text{Conf}(X \rightarrow Y) = \text{Supp}(X \rightarrow Y) / \text{Supp}(X)$

Nhận xét: Các xác nhận và độ tin cậy chính là các xác suất sau:

$\text{Supp}(X \rightarrow Y) = P(X \cup Y)$: Xác suất của $X \cup Y$ trong D

$\text{Conf}(X \rightarrow Y) = P(Y/X)$: Xác suất có điều kiện

Định nghĩa 5: Cho trước $\text{Min_Supp} = s_0$ và $\text{Min_Conf} = c_0$

Ta gọi luật $X \rightarrow Y$ là xảy ra nếu thỏa:

$\text{Supp}(X \rightarrow Y) > s_0$ và $\text{Conf}(X \rightarrow Y) > c_0$

Ví dụ 1: Xét CSDL sau

Ngày	T_ID	Các đơn vị dữ liệu					
D ₁	t ₁	A			D	E	
	t ₂	A					F
	t ₃	A	B		D	E	
D ₂	t ₄	A	B	C		E	
	t ₅				D		F
	t ₆	A		C	D	E	
D ₃	t ₇		B		D	E	
	t ₈	A			D		F
	t ₉		B	C		E	
D ₄	t ₁₀		B	C		E	F
	t ₁₁		B	C			F
	t ₁₂	A			D		

Ta có:

$$\text{Supp}(A \rightarrow D) = 5/12 = 41.66\%, \text{Conf}(A \rightarrow D) = 5/7$$

$$\text{Supp}(B \rightarrow D) = 2/12 = 17\%, \text{Conf}(B \rightarrow D) = 2/6 = 33.3\%$$

$$\text{Supp}(D \rightarrow F) = 2/12 \text{ và } \text{Conf}(D \rightarrow F) = 2/7 = 28.5\%$$

$$\text{Supp}(F \rightarrow D) = 2/12 \text{ và } \text{Conf}(F \rightarrow D) = 2/5$$

$$\text{Supp}(AC \rightarrow E) = 17\% \text{ Conf}(AC \rightarrow E) = 100\%$$

$$\text{Supp}(E \rightarrow AC) = 17\% \text{ Conf}(E \rightarrow AC) = 2/7 = 28.5\%$$

2.1 Thuật toán Apriori

Nhận xét 1:

* Hai bước chính của bài toán khai thác dữ liệu dựa trên các luật kết hợp:

1. Tạo ra tất cả tập đơn vị dữ liệu thường xuyên xảy ra (thoả ngưỡng là Min_Sup).
2. Từ tập các đơn vị dữ liệu thường xuyên xảy ra $Y = \{I_1, I_2, \dots, I_k\}$ với $k \geq 2$, sinh ra các luật tạo ra từ các đơn vị dữ liệu này bằng cách tóm các tập con của mỗi tập đơn vị dữ liệu và tính các độ tin cậy của chúng như trên.

2.1 Thuật toán Apriori

Cách tiếp cận của thuật toán Apriori dựa trên nhận xét sau: Nếu bất kỳ tập k-đvdl nào là không phỗ biến thì bất kỳ tập $(k+1)$ -đvdl chưa chúng cũng sẽ không phỗ biến, và ngược lại: Nếu bất kỳ tập k-đvdl nào là phỗ biến thì mọi tập con của nó là phỗ biến.

2.1 Thuật toán Apriori

Ký hiệu:

- Ta gọi số đơn vị dữ liệu trong một tập hợp là số các phần tử của chúng và tập có k phần tử là k-đơn vị dữ liệu
- Gọi L_k : Tập hợp các tập phỏ biến gồm các k -đvdl. Mỗi phần tử gồm 2 trường: i) các đơn vị dữ liệu và ii) đếm số lần xuất hiện.
- C_k : Tập hợp các tập ứng viên k - đơn vị dữ liệu. Mỗi phần tử gồm 2 trường: i) các đơn vị dữ liệu và ii) đếm số lần xuất hiện.

Thuật toán Apriori dựa trên các thủ tục sau

Procedure 1: Tạo ra các tập phỗ biến

Begin

$L_1 = \{\text{tập phỗ biến 1-đvdl}\};$

for ($k = 2; L_{k-1} \neq \emptyset; k++$) do

begin

$C_k = \text{Tạo ra tập ứng viên từ } (L_{k-1});$

for mỗi giao tác $t \in D$ do

begin

$C_t = \text{Tập con của } (C_k) \text{ chứa } t$

for mỗi $c \in C_t$ do $c.count++;$

end

$L_k = \{c \in C_k \mid c.count \geq \text{Min_Supp}^* |D|\}$

end

Return $(\cup_k L_k);$

end

Thuật toán Apriori dựa trên các thủ tục sau

Procedure 2: Tìm ra tất cả các luật kết hợp

begin

 Result = \emptyset

 for mỗi tập xảy ra thường xuyên $X \in L$ do

 begin

 for mỗi $a \subset X$ sao cho $a \neq \emptyset$ do

 if(Mức xác nhận(X)/Mức xác nhận(a) \geq Min_Conf)

 then Result = Result \cup { $a \rightarrow (X-a)$ }

 end

 return Result

end

Trong ví dụ 1, với $\text{Min_Conf} = c_0 = 70\%$ và $\text{Min_Supp} = s_0 = 40\%$

- Ta có tập L gồm các tập đơn vị dữ liệu xảy ra thường xuyên như sau:

$$L = \{\{A\}, \{B\}, \{C\}, \{D\}, \{E\}, \{F\}, \{AD\}, \{BE\}, \{CE\}, \{DE\}\}$$

Có các luật kết hợp như sau:

$A \rightarrow D$ với $c=71.42\%$ và $s=41.66\%$

$D \rightarrow A$ với $c=71.42\%$ và $s=41.66\%$

$B \rightarrow E$ với $c=83.33\%$ và $s=41.66\%$

$E \rightarrow B$ với $c=71.42\%$ và $s=41.66\%$

Luật kết hợp: Cơ sở

Mức xác nhận tối thiểu σ / S_0 : (minsupp)

- Cao \Rightarrow ít tập phần tử (itemset) phổ biến
 \Rightarrow ít luật hợp lệ rất thường xuất hiện
- Thấp \Rightarrow nhiều luật hợp lệ hiếm xuất hiện

Độ tin cậy tối thiểu γ / C_0 : (minconf)

- Cao \Rightarrow ít luật nhưng tất cả “gần như đúng”
- Thấp \Rightarrow nhiều luật, phần lớn rất “không chắc chắn”

Giá trị tiêu biểu: $\sigma = 2 - 10 \%$, $\gamma = 70 - 90 \%$

Luật kết hợp: Cơ sở

Giao tác:

– Dạng quan hệ

$\langle \text{Tid}, \text{item} \rangle$

$\langle 1, \text{item1} \rangle$

$\langle 1, \text{item2} \rangle$

$\langle 2, \text{item3} \rangle$

Dạng kết

$\langle \text{Tid}, \text{itemset} \rangle$

$\langle 1, \{\text{item1}, \text{item2}\} \rangle$

$\langle 2, \{\text{item3}\} \rangle$

Item và itemsets:

phần tử đơn lẻ và tập phần tử

Support của tập I: số lượng giao tác có chứa I

Min Support σ : ngưỡng cho support

Tập phần tử phổ biến: có độ ủng hộ (support) $\geq \sigma$

Luật kết hợp: Cơ sở

Cho: (1) CSDL các giao tác, (2) mỗi giao tác là một danh sách mặt hàng được mua (trong một lượt mua của khách hàng)
Frequent item sets

ID của giao tác	Hàng mua
100	A,B,C
200	A,C
400	A,D
500	B,E,F

Tập phổ biến	support
{A}	3 or 75%
{B} và {C}	2 or 50%
{D}, {E} và {F}	1 or 25%
{A,C}	2 or 50%
Các cặp khác	max 25%

Tìm: tất cả luật có support $\geq \text{minsupport}$

- If min. support 50% and min. confidence 50%, then
 $A \Rightarrow C$ [50%, 66.6%], $C \Rightarrow A$ [50%, 100%]

Tạo ứng viên Apriori

Nguyên tắc Apriori:

Những tập con của tập phỏng biến cũng phải phỏng biến

$$L_3 = \{abc, abd, acd, ace, bcd\}$$

Tụ kết: $L_3 * L_3$

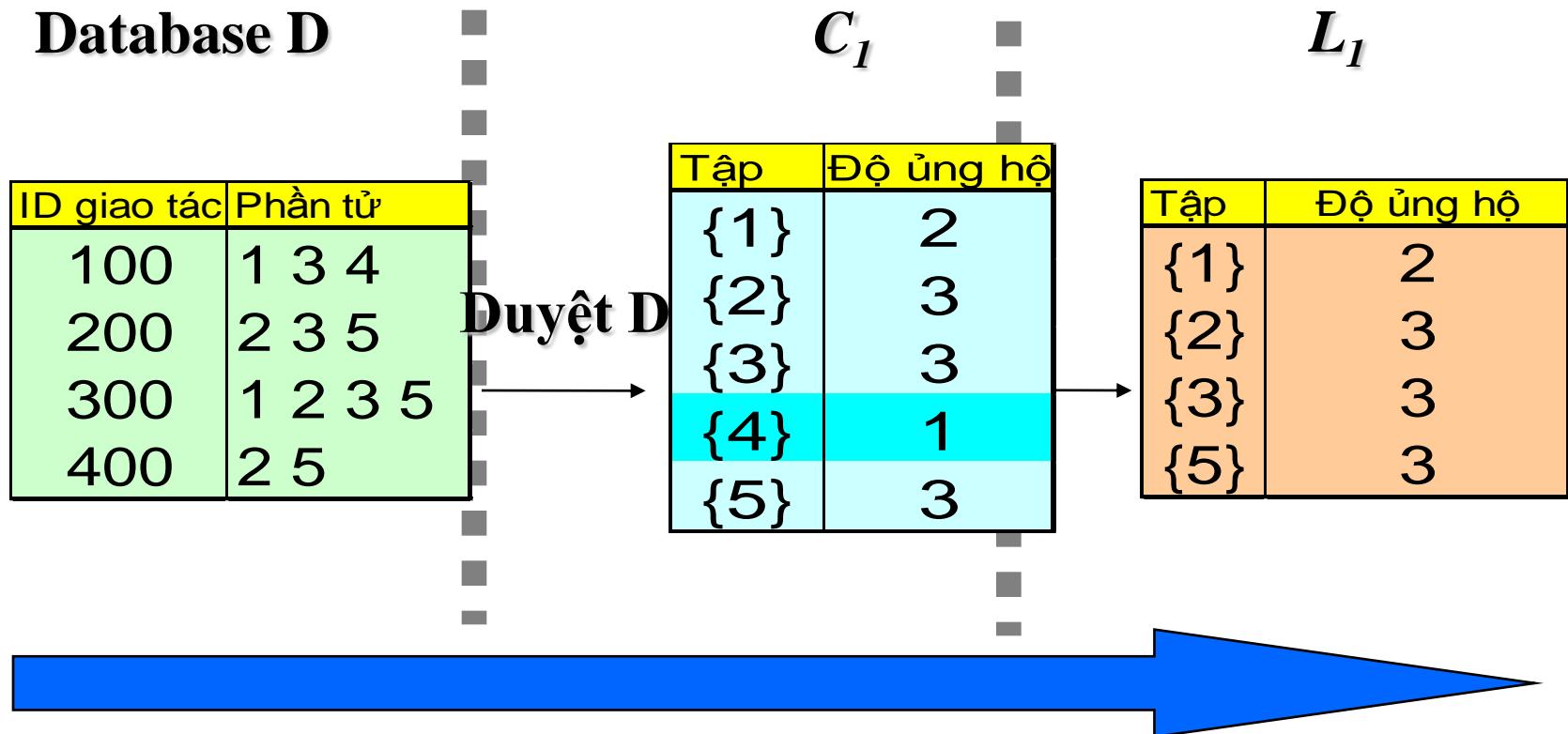
- $abcd$ từ abc và abd
- $acde$ từ acd và ace

Rút gọn:

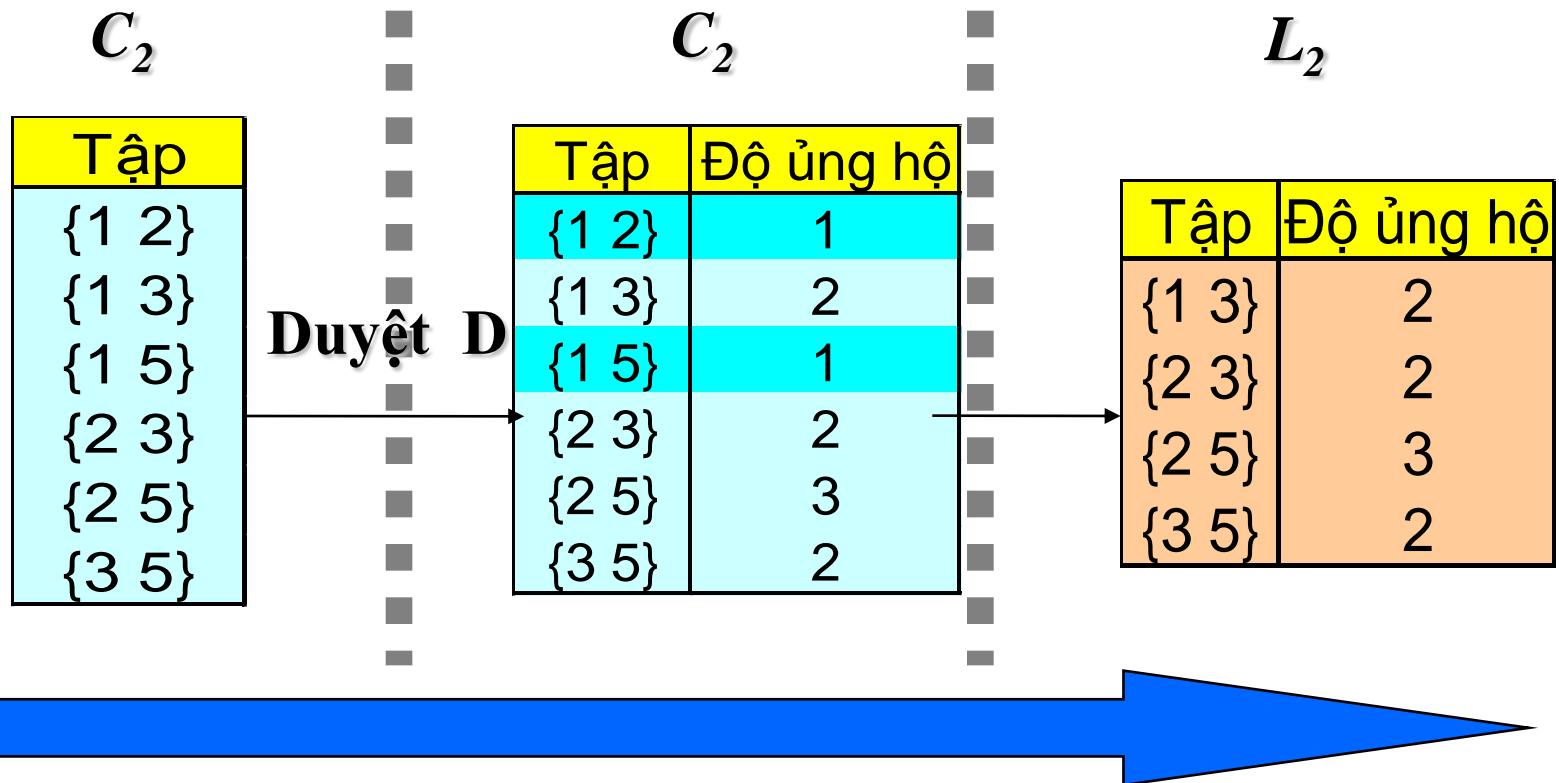
- $acde$ bị loại vì ade không có trong L_3

$$C_4 = \{abcd\}$$

Ví dụ về Apriori (1/6)



Ví dụ về Apriori (2/6)



Ví dụ về Apriori (3/6)

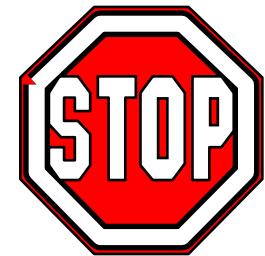
C_3

Tập
{2 3 5}

Duyệt D

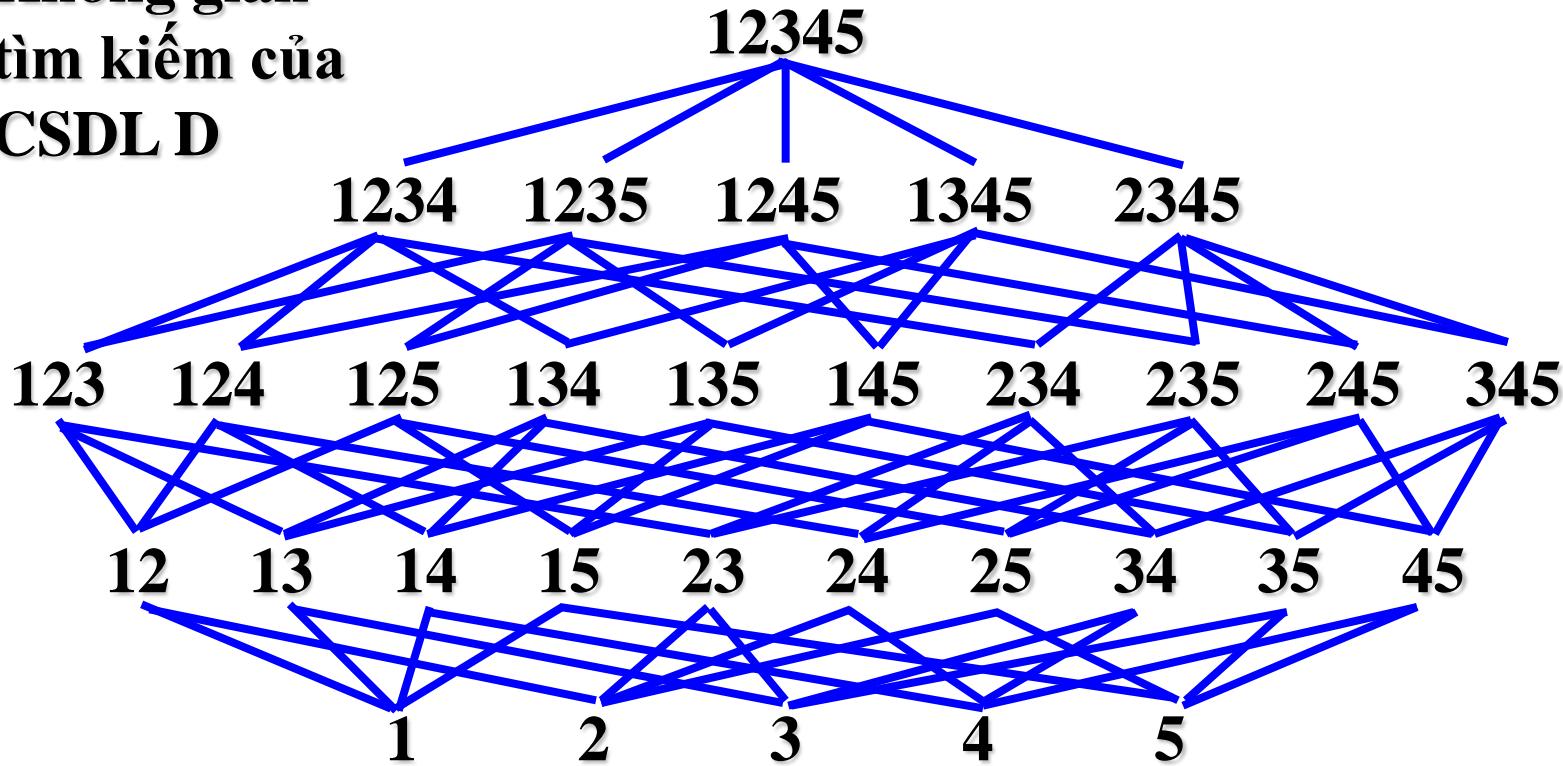
L_3

Tập	Độ ủng hộ
{2 3 5}	2



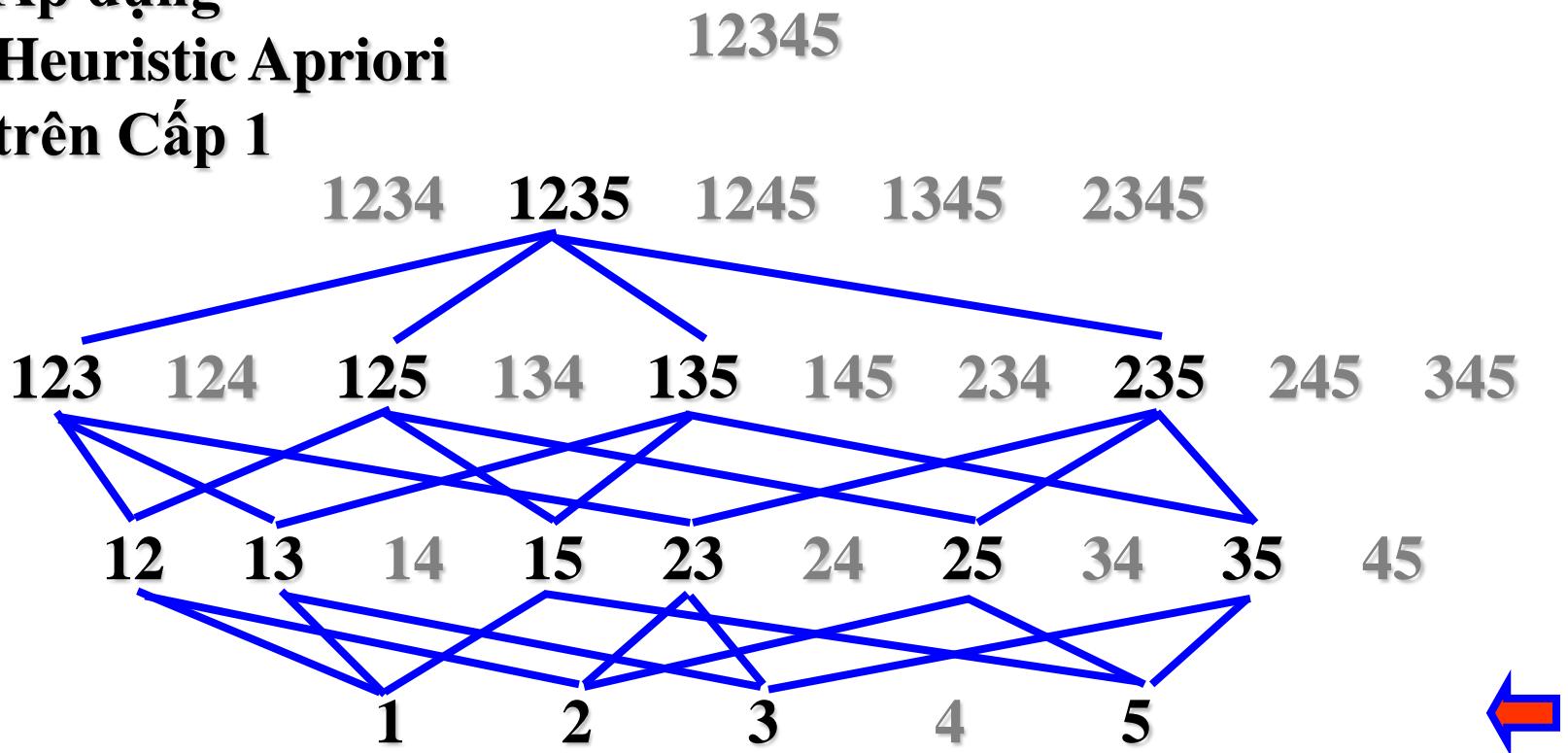
Ví dụ về Apriori (4/6)

Không gian
tìm kiếm của
CSDL D



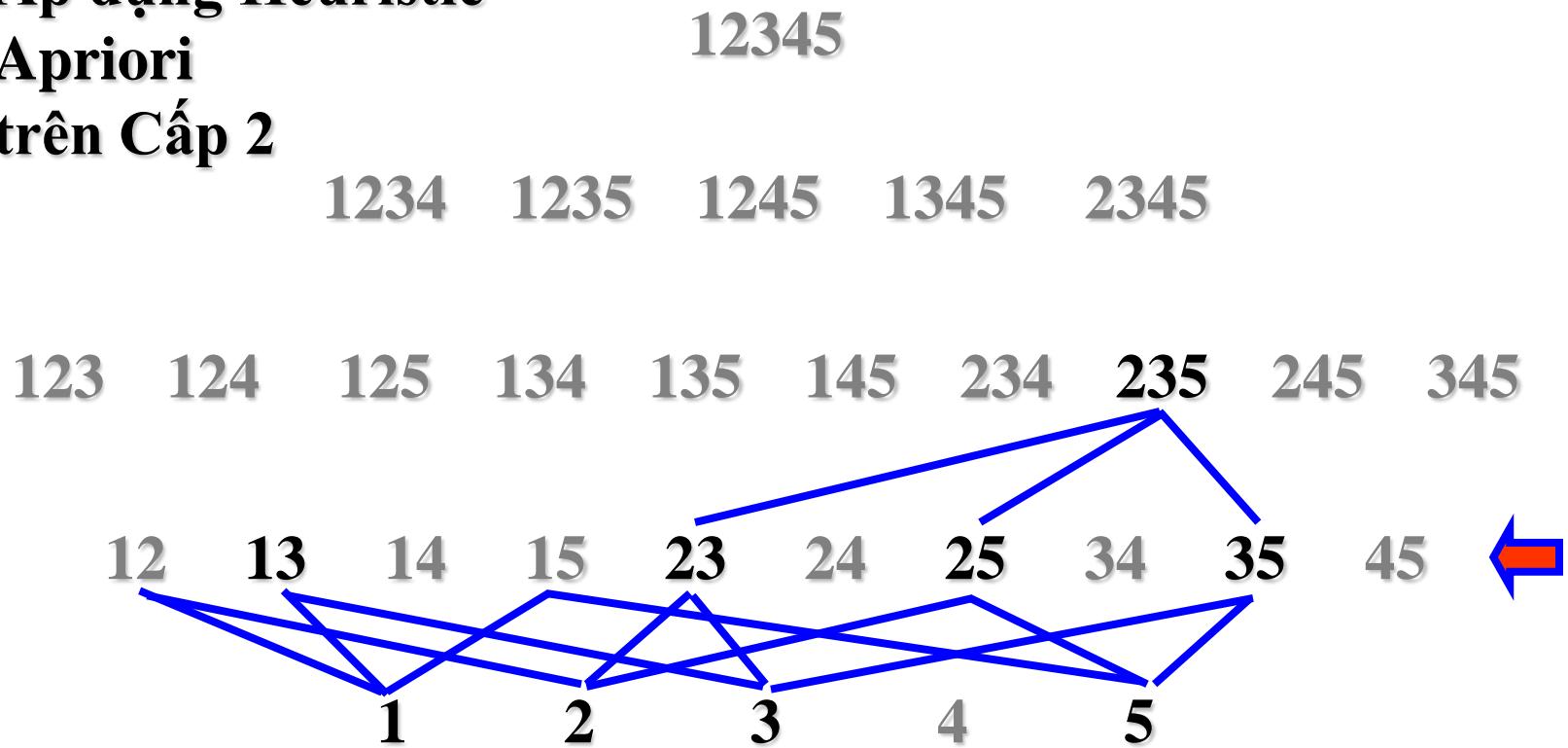
Ví dụ về Apriori (5/6)

Áp dụng
Heuristic Apriori
trên Cấp 1



Ví dụ về Apriori (6/6)

Áp dụng Heuristic
Apriori
trên Cấp 2



Tập phổ biến tối đa (maximal frequent sets).

Tập phổ biến

Tập phổ biến tối đa (maximal frequent sets).

Định nghĩa: M là tập phổ biến tối đa nếu M là tập phổ biến và không tồn tại tập phổ biến S khác M mà $M \subset S$

Thuật toán Apriori đã đủ nhanh?

Phần cốt lõi của thuật toán Apriori: FP tree

- Dùng các tập phỏ biến kích thước ($k - 1$) để tạo các tập phỏ biến kích thước k ứng viên
- Duyệt CSDL và đối sánh mẫu để đếm số lần xuất hiện của các tập ứng viên trong các giao tác

Tình trạng nghẽn cổ chai của thuật toán Apriori: việc tạo ứng viên

- Các tập ứng viên đồ sộ:
 - 10^4 tập phỏ biến kích thước 1 sẽ tạo ra 10^7 tập ứng viên kích thước 2
 - Để phát hiện một mẫu phỏ biến kích thước 100, ví dụ $\{a_1, a_2, \dots, a_{100}\}$, cần tạo $2^{100} \approx 10^{30}$ ứng viên.
- Duyệt CSDL nhiều lần:
 - Cần duyệt $(n + 1)$ lần, n là chiều dài của mẫu dài nhất

Thuật toán Apriori đã đủ nhanh?

Thực tế:

- Đối với tiếp cận Apriori căn bản thì số lượng thuộc tính trên dòng thường khó hơn nhiều so với số lượng dòng giao tác.
- Ví dụ:
 - 50 thuộc tính mỗi cái có 1-3 giá trị, 100.000 dòng (không quá tệp)
 - 50 thuộc tính mỗi cái có 10-100 giá trị, 100.000 dòng (hơi tệp)
 - 10.000 thuộc tính mỗi cái có 5-10 giá trị, 100 dòng (quá tệp...)
- Lưu ý:
 - Một thuộc tính có thể có một vài giá trị khác nhau
 - Các thuật toán luật kết hợp có đặc trưng là xem một cặp thuộc tính-giá trị là một thuộc tính (2 thuộc tính mỗi cái có 5 giá trị => "10 thuộc tính")

Cách khắc phục vấn đề ?

Cải thiện hiệu quả của TT Apriori

Đếm tập dựa vào kỹ thuật băm:

- Một tập kích thước k có hashing bucket count tương ứng nhỏ hơn giới hạn thì không thể phổ biến.

Thu nhỏ giao tác:

- Một giao tác không chứa tập phổ biến kích thước k nào thì không cần xét đến ở các lần duyệt tiếp theo.

Chia nhỏ:

- Tập nào có khả năng phổ biến trong DB thì sẽ phổ biến trong ít nhất một phần chia của DB.

Lấy mẫu:

- Khai thác trên tập con của dữ liệu được cho, ngưỡng của độ ủng hộ thấp hơn + một phương thức để xác định tính đầy đủ

Thuật toán FP-Tree

Ý tưởng: Dùng đệ quy để gia tăng độ dài của mẫu phổ biến dựa trên cây FP và các mẫu được phân hoạch

Phương pháp thực hiện:

- Với mỗi item phổ biến trong Header Table, xây dựng cơ sở điều kiện và cây điều kiện của nó
- Lặp lại tiến trình trên với mỗi cây điều kiện mới được tạo ra
- Cho tới khi cây điều kiện được tạo ra là cây rỗng hoặc chỉ bao gồm một đường đi đơn thì ngừng. Mỗi tổ hợp con các item trên đường đi đơn được tạo ra sẽ là một tập phổ biến

Thuật toán FP-Tree

Các bước xây dựng cây FP-Tree

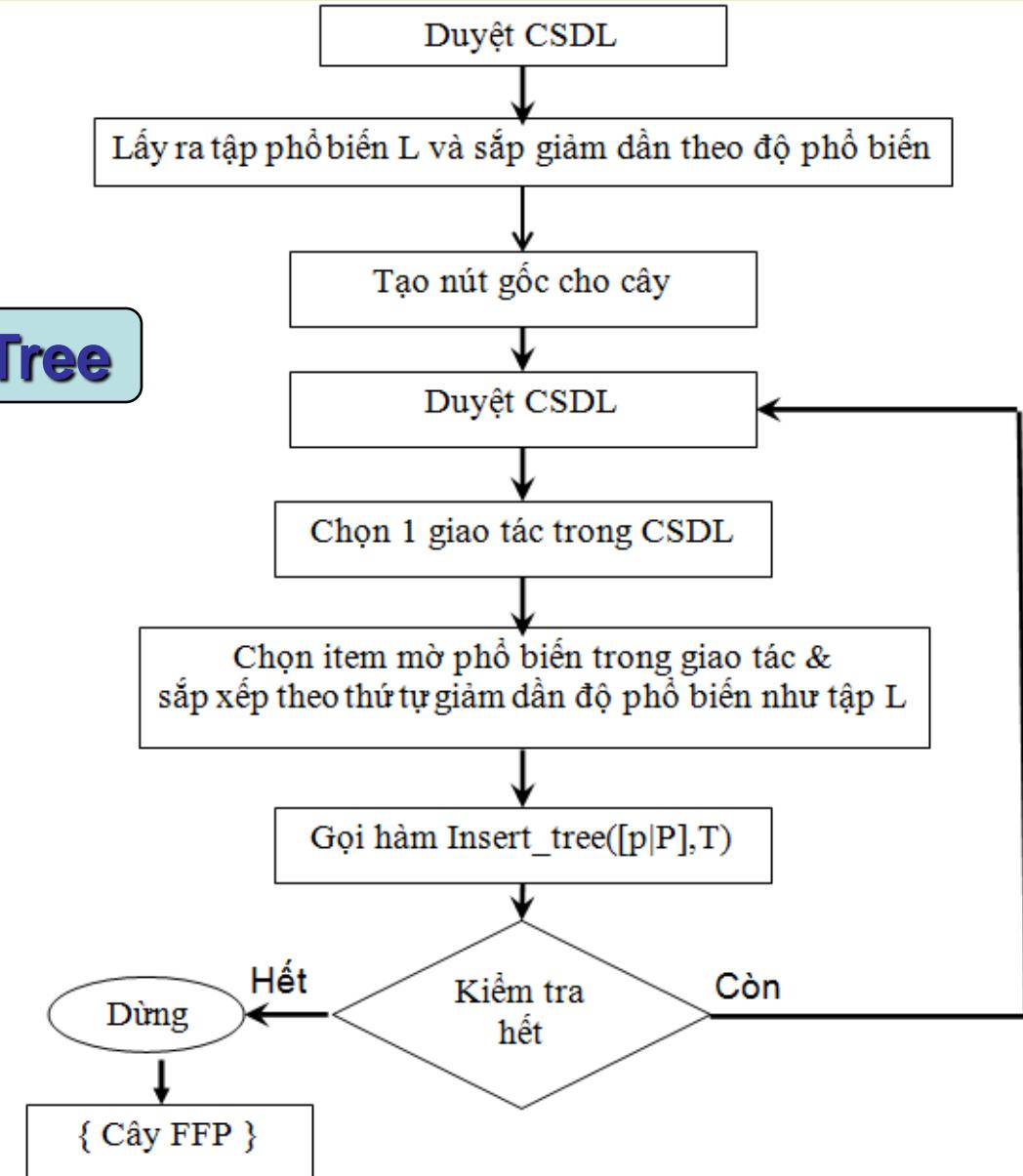
Bước 1: Duyệt CSDL, xác định tập F các item phổ biến một phần tử, sau đó loại bỏ các Item không thỏa ngưỡng minsup. Sắp xếp các item trong tập F theo thứ tự giảm dần của độ phổ biến, ta được tập kết quả là L.

Bước 2: Tạo nút gốc cho cây T, và tên của nút gốc sẽ là Null. Sau đó duyệt CSDL lần thứ hai. Ứng với mỗi giao tác trong CSDL ta thực hiện 2 công việc sau:

- Chọn các item phổ biến trong các giao tác và sắp xếp chúng theo thứ tự giảm dần độ phổ biến trong tập L
- Gọi hàm Insert_tree([p|P],T) để đưa các item vào trong cây T

Thuật toán FP-Tree

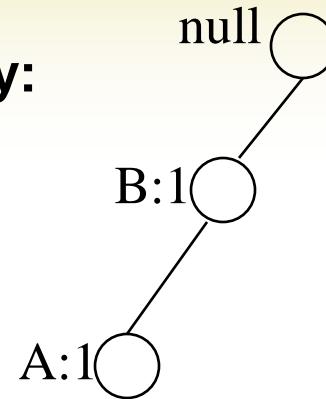
Xây dựng cây FP-Tree



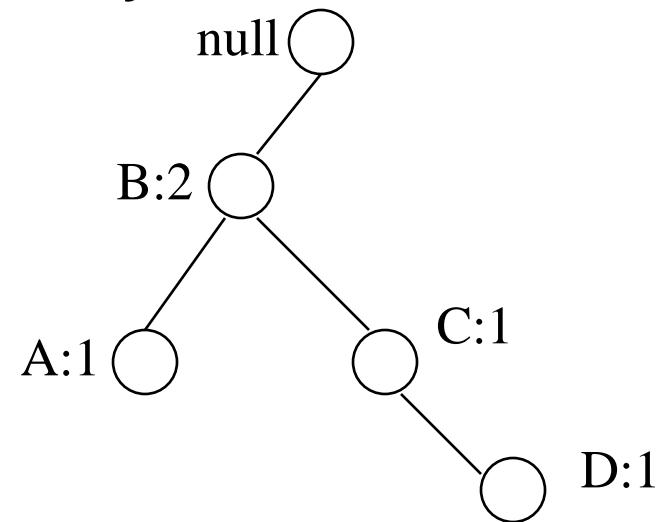
Thuật toán FP-Tree

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

Thêm TID=1 vào cây:



Thêm TID=2 vào cây:



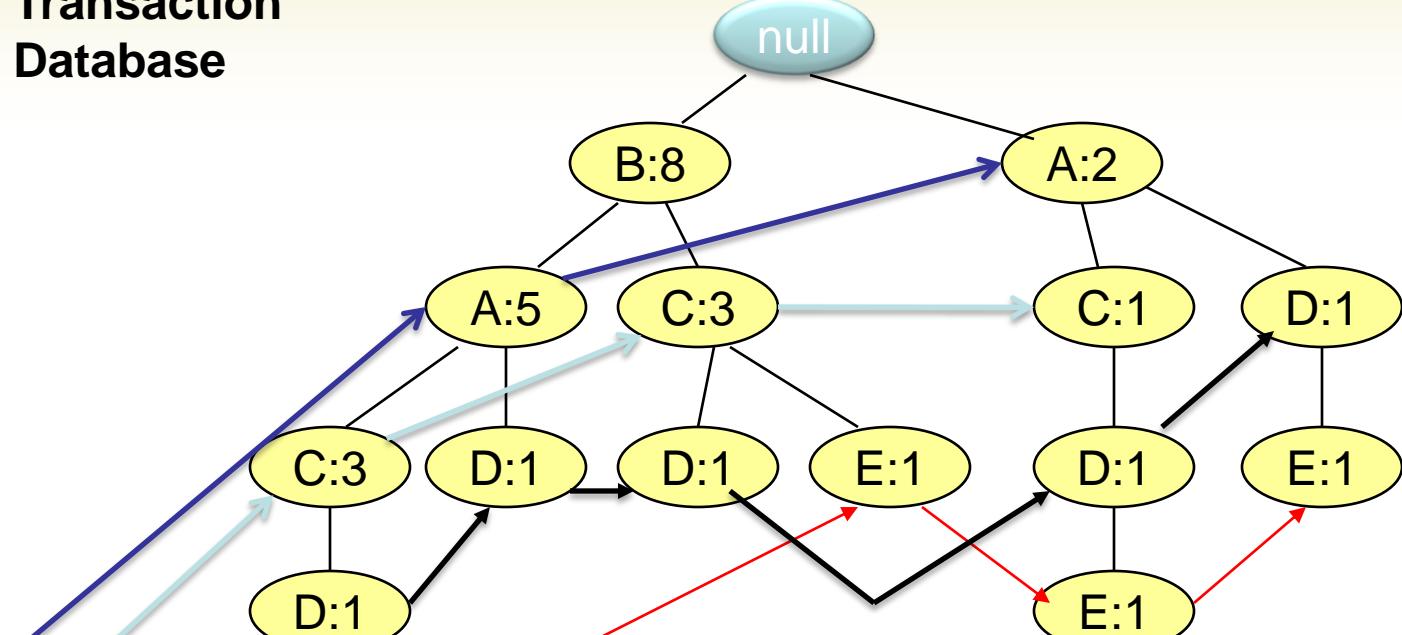
Thuật toán FP-Tree

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

Transaction Database

Header table

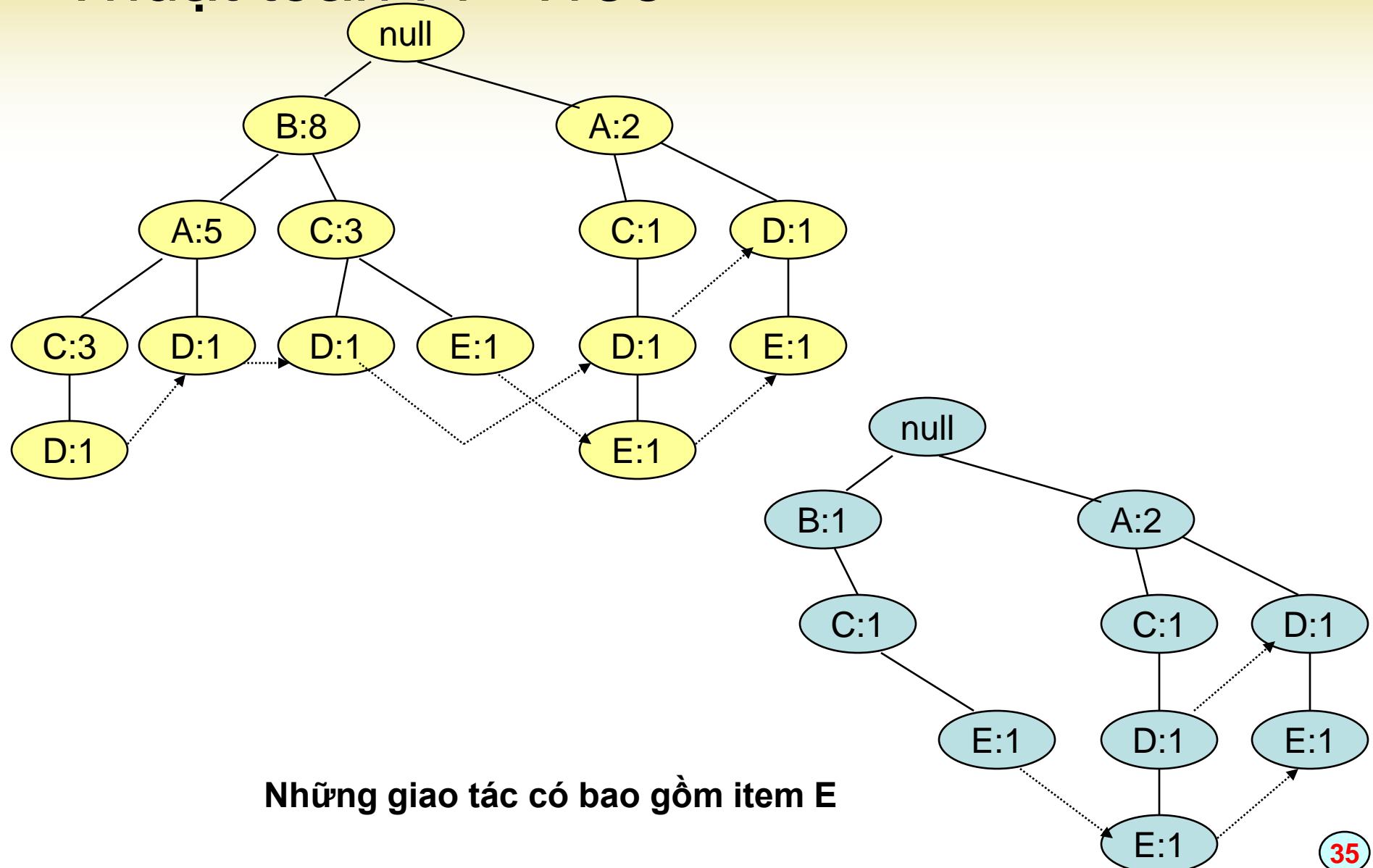
B	8
A	7
C	7
D	5
E	3



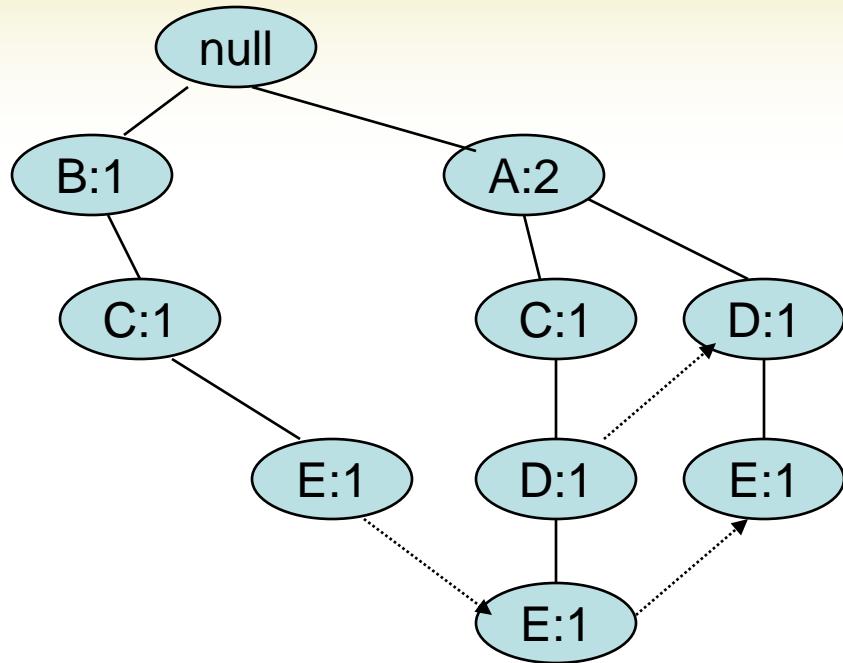
Sau khi thêm các giao tác vào, ta được cây như hình trên

Dựa trên cây, ta lập bảng header để tạo liên kết giữa các node có cùng item

Thuật toán FP-Tree



Thuật toán FP-Tree

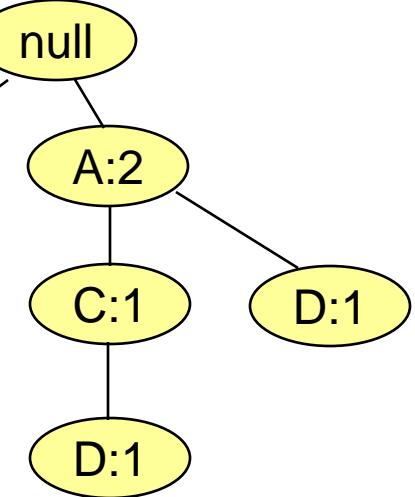


(New) Header table

A	2
C	2
D	2

Cây điều kiện
cho item E

Item B bị loại
bỏ do
 $\text{support}(B)=1$
nhỏ hơn
 $\text{minsup}=2$.

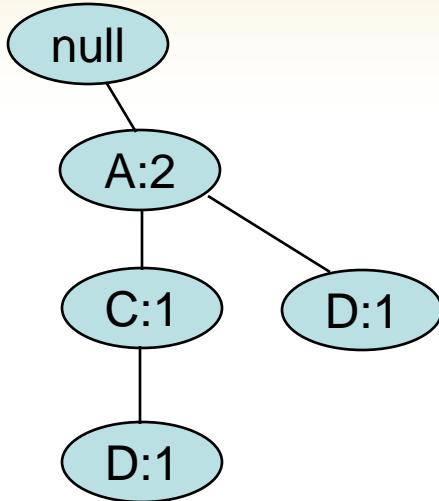


Với mỗi nhánh cây bao gồm E.

- Loại bỏ E
- Thêm vào cây mới
- Xây dựng lại bảng Header cho cây mới

Tiếp tục thực hiện đệ quy các
thao tác cho đến khi trên cây chỉ
còn một đường đi đơn
Item phổ biến: E(3)

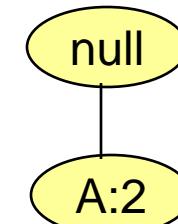
Thuật toán FP-Tree



(New) Header table

A	2
---	---

Cây điều kiện cho
tập item DE



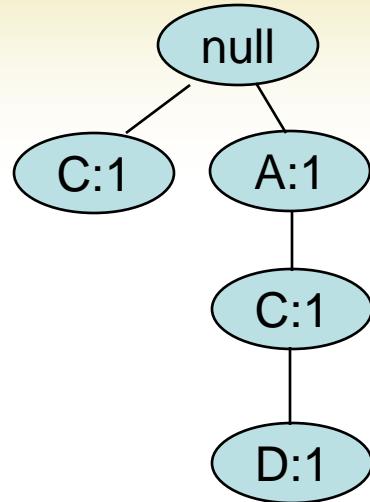
Các tập phổ biến sau khi kết thúc tiến
trình đệ quy do cây chỉ còn một đường đi

Tập phổ biến: DE(2), ADE(2)

Tập các đường đi bắt đầu với E và
kết thúc với D.

Lần lượt thêm từng đường dẫn vào
cây mới sau khi đã loại bỏ D

Thuật toán FP-Tree



(New) Header table



Kết thúc quá trình đệ quy do cây rỗng.

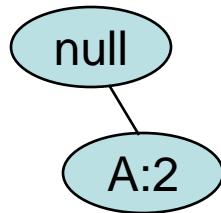
Tập phổ biến: CE(2)

Tập các đường dẫn bắt đầu từ E và kết thúc với C.

Thêm lần lượt từng nhánh vào cây mới (sau khi đã loại bỏ C)

Thuật toán FP-Tree

(New) Header table



Quá trình đệ quy kết thúc do cây rỗng.

Tập phổ biến: AE(2)

Tập các đường đi bắt đầu từ E và
kết thúc với A.

Thêm lần lượt từng đường đi vào
cây mới (sau khi loại bỏ A).

Thuật toán FP-Tree

Procedure FFP-growth(Tree, α)

{

- (1) Nếu Tree có chứa một đường đi đơn P
- (2) Thì với mỗi cách kết hợp γ của các nút trong đường đi P thực hiện
- (3) phát sinh tập mẫu $\tilde{\gamma}U\alpha$, support = min(support của các nút trong γ);
- (4) ngược lại ứng với mỗi A_i trong thành phần của Tree thực hiện

{

- (5) - phát sinh tập mẫu $\beta=A_iU\alpha$ với độ phổ biến support = $A_i.support$;
- (6) - xây dựng cơ sở điều kiện cho β và sau đó xây dựng cây FP Tree β theo điều kiện của β ;
- (7) - Nếu Tree $\beta \neq \emptyset$
- (8) thì gọi lại hàm FFP-growth(Tree β , β)

}

}

Tạo luật kết hợp

Ghi nhớ 1:

- Tạo các tập phổ biến thì chậm (đặc biệt là các tập kích thước 2)
- Tạo các luật kết hợp từ các tập phổ biến thì nhanh

Ghi nhớ 2:

- Khi tạo các tập phổ biến, ngưỡng độ ủng hộ được sử dụng
- Khi tạo luật kết hợp, ngưỡng độ tin cậy được sử dụng

Thực tế, việc tạo các tập phổ biến và tạo các luật kết hợp thật sự chiếm thời gian bao lâu?

- Xét một ví dụ nhỏ trong thực tế...
- Các thử nghiệm được thực hiện với Citum 4/275 Alpha server có bộ nhớ chính 512 MB & Red Hat Linux release 5.0 (kernel 2.0.30)

Chọn những luật tốt nhất?

Tập kết quả thường rất lớn, cần chọn ra những luật tốt nhất dựa trên:

- Các độ đo khách quan:

Hai các đo phổ biến:

☆ *support*; và

⌚ *confidence*

- Các độ đo chủ quan (Silberschatz & Tuzhilin, KDD95)

Một luật (mẫu) là tốt nếu

☆ *gây bất ngờ* (gây ngạc nhiên cho user); và/hoặc

⌚ *có thể hoạt động* (user có thể dùng nó để làm gì đó)

Những kết quả này sẽ được dùng trong các quá trình khám phá tri thức (KDD)

Luật Boolean và luật định lượng

Luật kết hợp Boolean so với định lượng (tùy vào loại giá trị được dùng)

- **Boolean:** Luật liên quan đến mối kết hợp giữa sự có xuất hiện và không xuất hiện của các phần tử (ví dụ "có mua A" hoặc "không có mua A")

mua=SQLServer, mua=DMBook \Rightarrow mua=DBMiner [2%,60%]

$mua(x, "SQLServer") \wedge mua(x, "DMBook") \rightarrow mua(x, "DBMiner") [0.2\%, 60\%]$

- **Định lượng:** Luật liên quan đến mối kết hợp giữa các phần tử hay thuộc tính định lượng

tuổi=30..39, thu nhập=42..48K \Rightarrow mua=PC [1%, 75%]

$tuổi(x, "30..39") \wedge thu\ nh\acute{a}p(x, "42..48K") \rightarrow mua(x, "PC") [1\%, 75\%]$

Các luật định lượng

Các thuộc tính định lượng: ví dụ: tuổi, thu nhập, chiều cao, cân nặng

Các thuộc tính phân loại: ví dụ: màu sắc của xe

CID	chieu cao	cân nặng	thu nhập
1	168	75,4	30,5
2	175	80,0	20,3
3	174	70,3	25,8
4	170	65,2	27,0

Vấn đề: có quá nhiều giá trị khác nhau cho các thuộc tính định lượng

Giải pháp: chuyển các thuộc tính định lượng sang các thuộc tính phân loại (chuyển qua không gian rời rạc)

Các luật một chiều và nhiều chiều

Các mối kết hợp một chiều và nhiều chiều

- **Một chiều:** Các thuộc tính hoặc tập thuộc tính trong luật chỉ quy về một đại lượng (ví dụ, quy về "mua")
Bia, khoai tây chiên ⇒ bánh mì [0.4%, 52%]
mua(x, "Bia") ^ mua(x, "Khoai tây chiên") →
mua(x, "Bánh mì") [0.4%, 52%]
- **Nhiều chiều:** Các thuộc tính hoặc thuộc tính trong luật được quy về hai hay nhiều đại lượng (ví dụ: "mua", "thời gian giao dịch", "loại khách hàng")
Trong ví dụ sau là: quốc gia, tuổi, thu nhập

Các luật nhiễu chiềу

C ID	qu o c g ia	tu o i	th u n h a p
1	Ý	50	th a p
2	Ph á p	40	c a o
3	Ph á p	30	c a o
4	Ý	50	trung b ình
5	Ý	45	c a o
6	Ph á p	35	c a o

CÁC LUẬT:

- quốc gia** = Pháp \Rightarrow **thu nhập** = cao [50%, 100%]
thu nhập = cao \Rightarrow **quốc gia** = Pháp [50%, 75%]
tuổi = 50 \Rightarrow **quốc gia** = Ý [33%, 100%]

Các luật một cấp và nhiều cấp

Các mối kết hợp một cấp và nhiều cấp

- **Một cấp:** Mối kết hợp giữa các phần tử hay thuộc tính của cùng một cấp khái niệm (ví dụ cùng một cấp của hệ thống phân cấp)
Bia, Khoai tây chiên ⇒ Bánh mì [0.4%, 52%]
- **Nhiều cấp:** Mối kết hợp giữa các phần tử hay thuộc tính của nhiều cấp khái niệm khác nhau (ví dụ nhiều cấp của hệ thống phân cấp)
Bia:Karjala, Khoai tây chiên:Estrella:Barbeque ⇒ Bánh mì [0.1%, 74%]

Các luật kết hợp nhiều cấp

Khó tìm những mẫu tốt ở cấp quá gần gốc

- độ ủng hộ cao = quá ít luật
- độ ủng hộ thấp = quá nhiều luật, không tốt nhất

Tiếp cận: suy luận ở cấp khái niệm phù hợp

Một dạng phổ biến của tri thức nền là một thuộc tính có thể được tổng quát hóa hay chi tiết hóa dựa vào **cây khái niệm**

Các luật kết hợp nhiều cấp: những luật phối hợp các mối kết hợp với cây các khái niệm

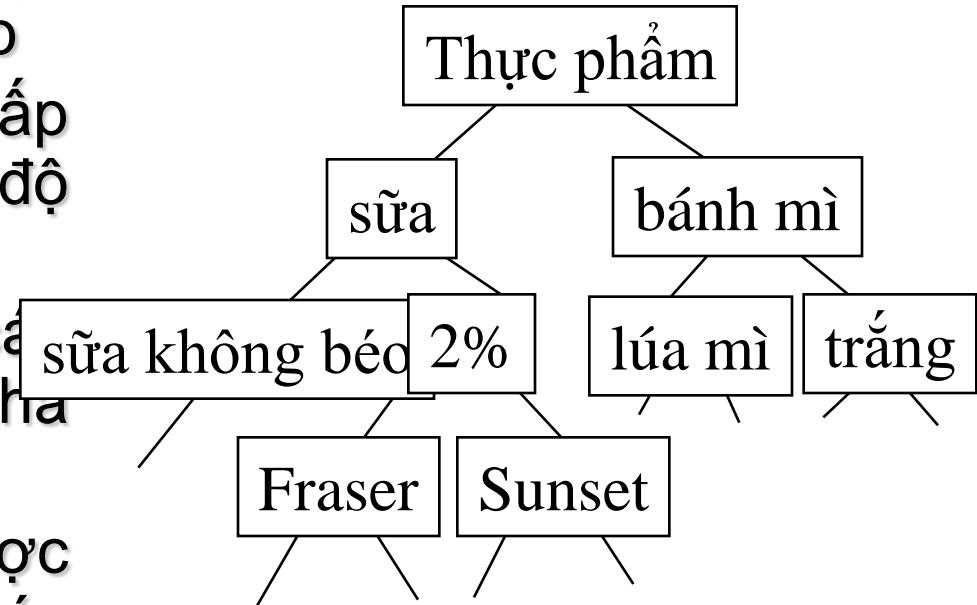
Các luật kết hợp nhiều cấp

Các phần tử thường tạo thành các cây phân cấp

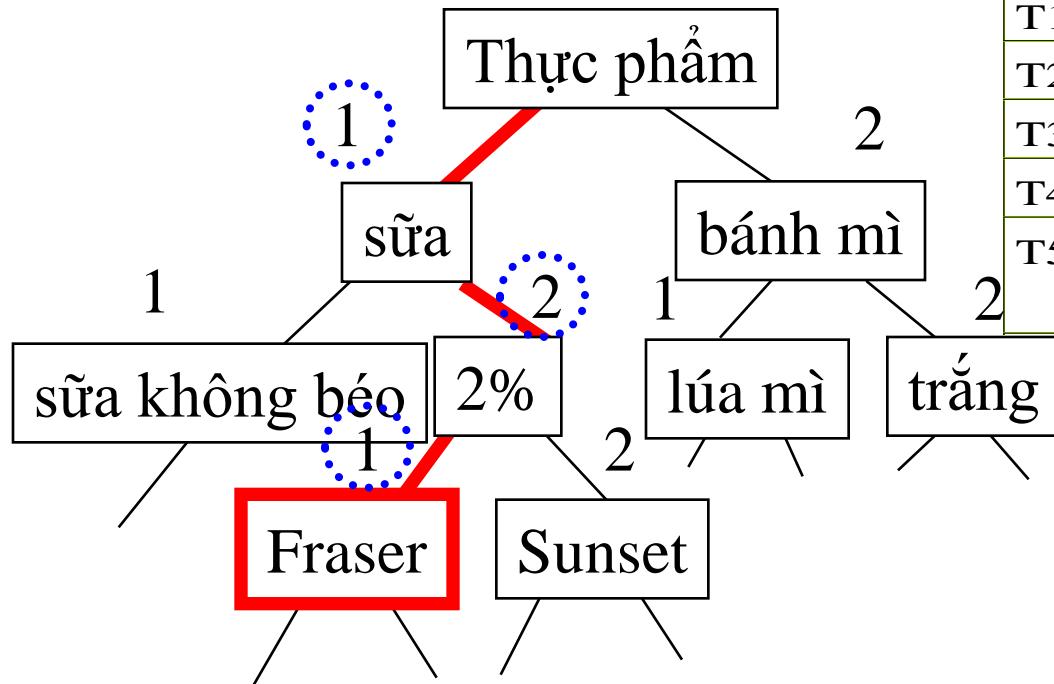
Các phần tử ở cấp thấp hơn được cho là có độ ủng hộ thấp hơn

Các luật về các tập ở cấp thấp thích hợp sẽ không hữu ích

CSDL giao tác có thể được mã hóa dựa trên các chiều và các cấp



Các luật kết hợp nhiều cấp



ID giao tác	Mã hàng
T1	{111, 121, 211, 221}
T2	{111, 211, 222, 323}
T3	{112, 122, 221, 411}
T4	{111, 121}
T5	{111, 122, 211, 221, 413}

Các luật kết hợp nhiều cấp

Tiếp cận trên-xuống, tiến theo chiều sâu:

- Trước tiên tìm những luật mạnh ở cấp cao:
sữa → bánh mì [20%, 60%]
- Sau đó tìm những luật “yếu hơn” ở cấp thấp hơn của chúng:
sữa 2% → bánh mì lúa mì [6%, 50%]

Khai thác thay đổi trên các luật kết hợp nhiều cấp:

- Các luật kết hợp trên nhiều cấp khác nhau:
sữa → *bánh mì lúa mì*
- Các luật kết hợp với nhiều cây khái niêm:
sữa → *bánh mì Wonder*

Các luật kết hợp nhiều cấp

Tổng quát hóa/chuyên biệt hóa giá trị của các thuộc tính...

- ...từ chuyên biệt sang tổng quát: support của các luật tăng (có thêm những luật mới hợp lệ)
- ...từ tổng quát sang chuyên biệt: support của các luật giảm (có những luật trở thành không hợp lệ, độ ủng hộ của chúng giảm xuống nhỏ hơn ngưỡng qui định)

Bậc quá thấp => quá nhiều luật và quá thô sơ

Pepsi light 0.5l bottle ⇒ Taffel Barbeque Chips
200gr

Bậc quá cao => các luật không hay

Food ⇒ Clothes

Lọc luật thừa

Có những luật có thể là dư thừa do đã có các mối quan hệ “tổ tiên” giữa các phần tử

Ví dụ (sữa có 4 lớp con):

- sữa \Rightarrow bánh mì lúa mì [độ ủng hộ = 8%, độ tin cậy = 70%]
- sữa 2% \Rightarrow bánh mì lúa mì [độ ủng hộ = 2%, độ tin cậy = 72%]

Ta nói luật thứ nhất là tổ tiên của luật thứ hai

Một luật là dư thừa nếu độ ủng hộ của nó gần với giá trị “mong đợi”, dựa trên tổ tiên của luật

- Luật thứ hai ở trên có thể là dư thừa

Khai thác dựa trên ràng buộc

Khai thác cả giga-byte dữ liệu theo cách thăm dò, có tương tác?

- Điều này có khả thi không? - Bằng cách sử dụng tốt các ràng buộc!

Các loại ràng buộc nào có thể dùng trong khai thác dữ liệu?

- **Ràng buộc dạng tri thức:** phân lớp, kết hợp,
- **Ràng buộc dữ liệu:** những câu truy vấn dạng SQL
 - Tìm những cặp sản phẩm được bán chung tại VanCouver tháng 12/98
- **Những ràng buộc về kích thước/cấp bậc:**
 - Có liên quan về vùng, giá, nhãn hiệu, loại khách hàng
- **Những ràng buộc về sự hấp dẫn:**
 - Những luật mạnh ($\text{min_support} \geq 3\%$, $\text{min_confidence} \geq 60\%$)
- **Những ràng buộc luật** (xem slide sau)

Ràng buộc luật

Có hai loại ràng buộc luật:

- **Ràng buộc dạng luật: khai thác theo siêu luật (meta-rule)**
 - Metarule: $P(X, Y) \wedge Q(X, W) \rightarrow \text{lấy}(X, \text{"database systems"})$
 - Luật đối sánh: $\text{tuổi}(X, \text{"30..39"}) \wedge \text{thu nhập}(X, \text{"41K..60K"}) \rightarrow \text{lấy}(X, \text{"database systems"}).$
- **Ràng buộc trên nội dung luật: tạo câu truy vấn dựa trên ràng buộc (Ng, et al., SIGMOD'98)**
 - $\text{sum}(\text{LHS}) < 100 \wedge \text{min}(\text{LHS}) > 20 \wedge \text{count}(\text{LHS}) > 3 \wedge \text{sum}(\text{RHS}) > 1000$

Ràng buộc luật

Ràng buộc 1-biến và ràng buộc 2-biến (Lakshmanan, et al. SIGMOD'99):

- **1-biến**: Ràng buộc chỉ hạn chế trên một bên (L/R) của luật, ví dụ;
 - $\text{sum}(\text{LHS}) < 100 \wedge \text{min}(\text{LHS}) > 20 \wedge \text{count}(\text{LHS}) > 3 \wedge \text{sum}(\text{RHS}) > 1000$
- **2-biến**: Ràng buộc hạn chế trên cả hai bên (L và R) của luật.
 - $\text{sum}(\text{LHS}) < \text{min}(\text{RHS}) \wedge \text{max}(\text{RHS}) < 5 * \text{sum}(\text{LHS})$

Tóm tắt

Khai thác luật kết hợp:

- Quan trọng nhất trong KDD
- Khái niệm khá đơn giản nhưng ý tưởng của nó cung cấp cơ sở cho những mở rộng và những phương pháp khác
- Nhiều bài báo đã được công bố về đề tài này

Đã có nhiều kết quả hấp dẫn

Hướng nghiên cứu lý thú:

- Phân tích mối kết hợp trong các dạng dữ liệu khác: dữ liệu không gian, dữ liệu đa phương tiện, dữ liệu thời gian thực,
...