
# KHAI PHÁ LUẬT KẾT HỢP

Giảng viên: Đặng Thị Thu Hiền, Nguyễn Tu Trung
BM HTTT, Khoa CNTT, Trường ĐH Thủy Lợi

Hà Nội, 2022

# Nội dung

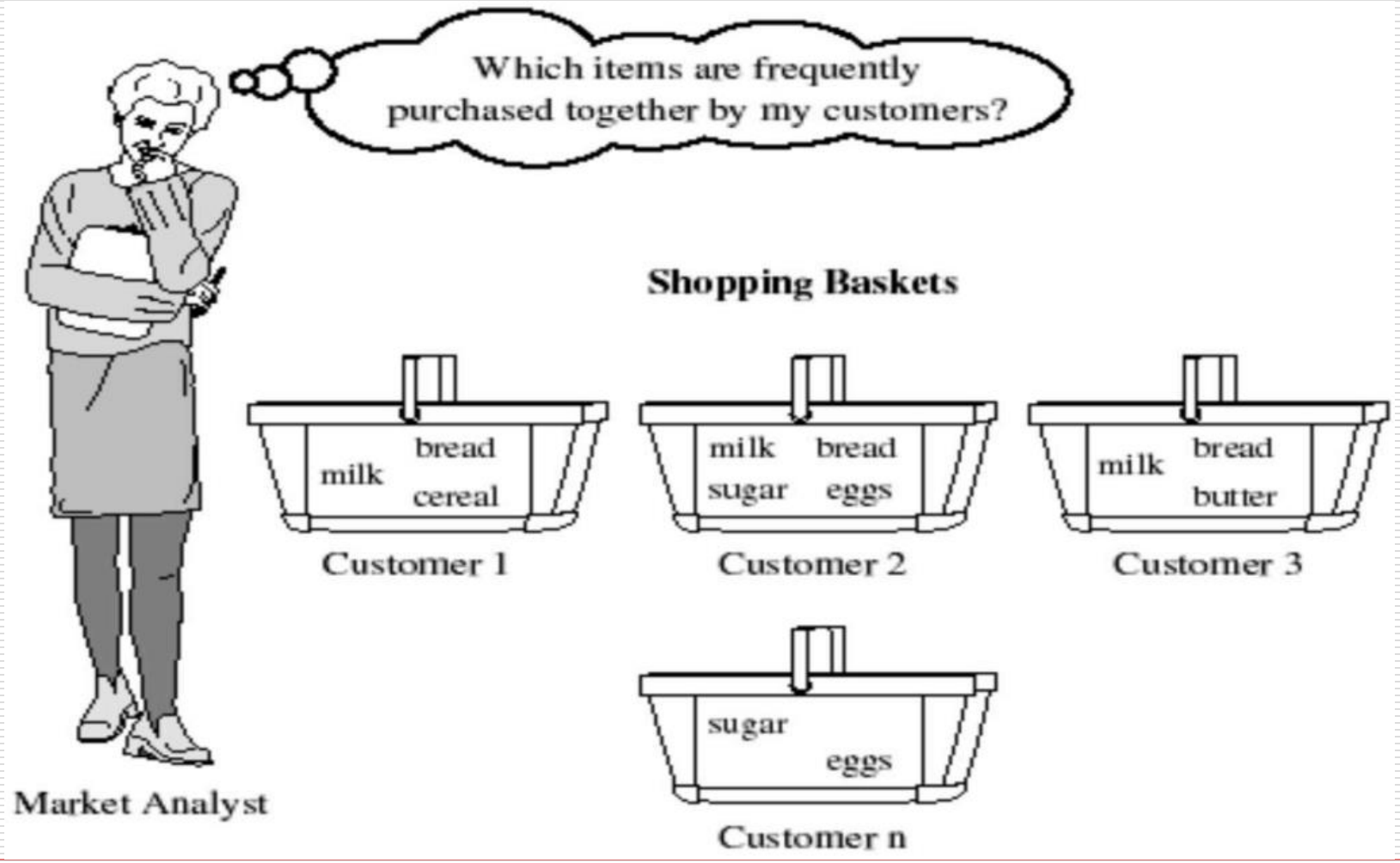
- Một số tình huống
- Khái niệm cơ bản về luật kết hợp
- Thuật toán tìm tập mục phổ biến
- Khai phá LKH từ tập mục phổ biến

# Một số tình huống

- Phân tích dữ liệu giỏ hàng (basket data analysis)
- Tiếp thị chéo (cross-marketing)

# Phân tích dữ liệu giỏ hàng

- Mặt hàng nào được được mua cùng nhau bởi khách hàng ?



# Tiếp thị chéo

- => Gợi ý mua sản phẩm khác

# Khái niệm cơ bản về luật kết hợp

- Tập mục và giao dịch
- Luật kết hợp
- Tính chất của luật kết hợp

# Tập mục và giao dịch

- Tập mục:
  - Cho một tập gồm n đối tượng I = {$I_1$, $I_2$,..., $I_n$}
  - Mỗi phần tử $I_i \in I$ gọi là một mục (item)
  - Một tập con bất kỳ X ⊆ I gọi là một tập mục (item set)
- Giao dịch
  - Cho một tập D = {$T_1$,$T_2$, ...,$T_m$}, mỗi phần tử Tj ∈ D gọi là một giao dịch và là một tập con nào đó của I (Tj ⊆ I)
  - Mỗi giao dịch định danh bởi $T_{ID}$ (Transaction Identification)
  - Gọi D là cơ sở dữ liệu giao dịch
  - Số giao dịch có trong D ký hiệu là |D|

# Tập mục và giao dịch

- Ví dụ:
  - I = {A, C, D, T, W}, X = {C, D, W} là một tập mục
  - Một cơ sở dữ liệu giao dịch D gồm các tập con Tj khác nhau của I:

| TID | Nội dung giao dịch |
|---|---|
| T1 | A, C, T, W |
| T2 | C, D, W |
| T3 | A, C, T, W |
| T4 | A, C, D, W |
| T5 | A, C, D, T, W |
| T6 | C, D, T |

# Luật kết hợp

- Cho X, Y ⊆ I, X ∩ Y = ∅
- Định nghĩa: Luật kết hợp ký hiệu là X => Y chỉ ra mối ràng buộc của tập mục Y theo tập mục X, nghĩa là khi X xuất hiện trong D thì sẽ kéo theo sự xuất hiện của Y với một tỉ lệ nào đấy
- ***Độ phổ biến*** (support) của một tập mục X trong CSDL D
  - Ký hiệu supp(X)
  - Là tỉ lệ phần trăm giữa số giao dịch chứa X trên tổng số các giao dịch trong D: $Supp(X) = \frac{count(X)}{|D|}$
  - Trong ví dụ trên, xét tập X = {C, W} thì số lần xuất hiện của C,W là 5 => supp(X) = 5/6 = 83%
- Độ phổ biến của một luật kết hợp X => Y, ký hiệu $Supp(X \Rightarrow Y)$, là độ phổ biến của tập mục $X \cup Y$: $Supp(X \Rightarrow Y) = \text{Supp}(X \cup Y)$

# Luật kết hợp

- ***Độ tin cậy*** (confidence): là tỉ lệ các giao dịch có chứa cả X và Y so với các giao dịch có chứa X:

$$Conf(X \Rightarrow Y) = \frac{\text{Supp}(X \cup Y)}{Supp(X)}$$

- Các luật có độ phổ biến lớn hơn ngưỡng minsupp và độ tin cậy lớn hơn ngưỡng minconf cho trước gọi là các luật mạnh (strong association rule) hay "luật có giá trị":
  - supp(X => Y) ≥ minsupp và conf(X => Y) ≥ minconf

# Tính chất của luật kết hợp

- *Tính chất 1:* Nếu X ⊆ Y với X,Y là các tập mục thì supp(X) >= supp(Y)
  - => hiển nhiên vì tất cả các giao dịch trong D hỗ trợ Y thì cũng hỗ trợ X
- *Tính chất 2:* Một tập chứa một tập không phổ biến thì cũng là tập không phổ biến
  - Nếu supp(X) < minsupp thì tập Y chứa X cũng không là tập phổ biến vì: supp(Y) <= supp(X) < minsup (theo tính chất 1)
- *Tính chất 3:* Các tập con của tập phổ biến cũng là tập phổ biến
  - Nếu Y là tập phổ biến trong D tức: supp(Y) >= minsup => mọi tập con A của Y cũng là phổ biến vì supp(A) >= supp(Y) >= minsup (theo tính chất 1)
  - Trường hợp đặc biệt, nếu tập A={i1,i2,..., ik} là tập phổ biến thì mọi tập con có (k-1) mục của nó cũng là phổ biến => ngược lại không đúng

# Thuật toán tìm tập mục phổ biến

- Input:
  - Cơ sở dữ liệu D (Tập giao dịch)
  - minsupp
- Output
  - Danh sách các tập phổ biến
- Các thuật toán:
  - Apriori
  - FP-Growth

# Thuật toán Apriori

- Ý tưởng thuật toán Apriori
- Ví dụ thuật toán Apriori
- Mã giả thuật toán Apriori
- Thủ tục apriori_gen
- Hạn chế của thuật toán Apriori

# Ý tưởng thuật toán Apriori

- Thuật toán tỉa bớt những tập ứng cử viên có tập con không phổ biến trước khi tính độ hỗ trợ
- Phát hiện các tập mục phổ biến qua nhiều lần duyệt dữ liệu
- Kiểm tra tập 1-mục:
  - B1: Tính độ phổ biến của các tập mục, mỗi tập chỉ gồm 1 mục (thuộc tính) riêng biệt
  - B2: Với mỗi tập, kiểm tra có thỏa mãn điều kiện tập phổ biến (thỏa mãn độ hỗ trợ cực tiểu)

# Ý tưởng thuật toán Apriori

- Kiểm tra tập k-mục (k > 1):
  - B1: Nếu k > tổng số mục => Kết thúc thuật toán: Các tập phổ biến có (k-1) mục
  - B2: Sinh các tập mục ứng viên k-mục khác nhau dựa trên các tập phổ biến (k-1)-mục – tìm thấy trong lần duyệt (k-1)
  - B3: Tính độ hỗ trợ thực sự của các tập mục ứng cử k-mục này và kiểm tra có thỏa mãn điều kiện tập phổ biến (thỏa mãn độ hỗ trợ cực tiểu)

# Ví dụ thuật toán Apriori

- Cho cơ sở dữ liệu D trong bảng bên
- Có 9 giao dịch => |D| = 9
- Giả sử minsup =2
- Quá trình thuật toán Apriori tìm tập mục phổ biến trong D thể hiện như sau:
- Lần 1:

| TID | Nội dung giao dịch |
|---|---|
| T1 | 1,2,5 |
| T2 | 2,4 |
| T3 | 2,3 |
| T4 | 1,2,4 |
| T5 | 1,3 |
| T6 | 2,3 |
| T7 | 1,3 |
| T8 | 1,2,3,5 |
| T9 | 1,2,3 |

Quét D để tính supp →

$C_1$

| Tập mục | Supp |
|---|---|
| {1} | 6 |
| {2} | 7 |
| {3} | 6 |
| {4} | 2 |
| {5} | 2 |

So sánh với minsup=2 →

$L_1$

| Tập mục | Supp |
|---|---|
| {1} | 6 |
| {2} | 7 |
| {3} | 6 |
| {4} | 2 |
| {5} | 2 |

# Ví dụ thuật toán Apriori

- Lần 2:

Sinh các ứng cử $C_2$ từ $L_1$ bằng cách kết nối $L_1$ với $L_1$

$C_2$

| Tập mục |
|---|
| {1,2} |
| {1,3} |
| {1,4} |
| {1,5} |
| {2,3} |
| {2,4} |
| {2,5} |
| {3,4} |
| {3,5} |
| {4,5} |

Quét D để tính độ hỗ trợ cho mỗi ứng cử trong $C_2$

$C_2$

| Tập mục | Supp |
|---|---|
| {1,2} | 4 |
| {1,3} | 4 |
| {1,4} | 1 |
| {1,5} | 2 |
| {2,3} | 4 |
| {2,4} | 2 |
| {2,5} | 2 |
| {3,4} | 0 |
| {3,5} | 1 |
| {4,5} | 0 |

So sánh với minsup=2

$L_2$

| Tập mục | Supp |
|---|---|
| {1,2} | 4 |
| {1,3} | 4 |
| {1,5} | 2 |
| {2,3} | 4 |
| {2,4} | 2 |
| {2,5} | 2 |

# Ví dụ thuật toán Apriori

- Lần 3:

Sinh các ứng cử $C_3$ từ $L_2$ bằng cách kết nối $L_2$ với $L_2$ và thực hiện tỉa dựa vào tính chất Apriori →

**$C_3$**

| Tập mục |
|---|
| {1,2,3} |
| {1,2,5} |

Quét D để tính độ hỗ trợ cho mỗi ứng cử trong $C_3$ →

**$C_3$**

| Tập mục | Supp |
|---|---|
| {1,2,3} | 2 |
| {1,2,5} | 2 |

So sánh với minsup=2 →

**$L_3$**

| Tập mục | Supp |
|---|---|
| {1,2,3} | 2 |
| {1,2,5} | 2 |

# Mã giả thuật toán Apriori

- (1) Quét cơ sở dữ liệu để tìm các tập 1 mục phổ biến $L_1$
- (2) For (k=2; $L_{k-1}$ # ∅; k++) {
- (3) $C_k$ = apriori_gen($L_{k-1}$, minsupp); // Sinh các ứng cử k-muc từ $L_{k-1}$
- (4) Foreach (t in D) { // Quét D để đếm
- (5) $C_t$=subnet($C_k$, t); // Lấy các ứng cử trong Ck mà là tập con của t
- (6) For (c in $C_t$)
- (7) c.count ++; }
- (8) $L_k$= {c ∈ $C_k$ sao cho: c.count >= minsupp} }
- (9) Return $U_{i=1}^{k} L_i$;

# Thủ tục apriori_gen

- Mục đích: Tìm $C_k$ – Tức sinh các ứng cử và tỉa các tập mục không phổ biến
- Đầu vào:
  - Tập phổ biến (k-1)-mục $L_{k-1}$
  - Ngưỡng độ hỗ trợ cực tiểu minsupp
- Đầu ra:
  - $C_k$
- Apriori giả định rằng tất cả các mục trong một giao dịch đều được sắp xếp theo thứ tự từ điển (có thể so sánh nhỏ hơn, lớn hơn)
- Phép nối $L_{k-1} \bowtie L_{k-1}$ được thực hiện khi các thành viên của $L_{k-1}$ là có thể nối được nếu các (k-2)-mục đầu tiên của chúng là giống nhau

# Thủ tục apriori_gen

- *Procedure* **apriori_gen**($L_{k-1}$: các tập phổ biến (k-1) mục; minsupp)

  (1) Foreach ($l_1$ in $L_{k-1}$)

  (2) Foreach ($l_2$ in $L_{k-1}$) {

  (3) If(l1[1]=l2[1]) ∧(l1[2]=l2[2]) ∧... ∧(l1[k-2]=l2[k-2]) ∧(l1[k-1]<l2[k-1])

  (4) then c=l1 ⋈ l2; // bước kết nối để sinh các ứng cử c

  (5) if has_infrequent_subset(c, $L_{k-1}$) then

  (6) Xóa c; // Bước tỉa

  (7) else thêm c vào $C_k$;

  (8) }

  (9) return $C_k$;

# Hạn chế của thuật toán Apriori

- Chi phí cho số lượng lớn các tập ứng cử
  - Nếu có $10^4$ tập 1-mục phổ biến thì thuật toán Apriori sẽ cần sinh ra hơn $10^7$ các ứng cử 2-mục và thực hiện kiểm tra sự xuất hiện của chúng
  - Để khám phá một mẫu phổ biến kích thước (độ dài) là l, thuật toán phải kiểm tra $(2^l – 2)$ các mẫu phổ biến tiềm năng:
    - Ví dụ nếu l=100 thì nó phải sinh ra tổng số $2^{100}$ tức khoảng $10^{30}$ các ứng cử
- Đòi hỏi lặp quá nhiều lần duyệt CSDL để kiểm tra các tập ứng cử:
  - Duyệt theo các item k, c, d

# Thuật toán FP-Growth NB

- Giới thiệu thuật toán FP-Growth
- Các bước chính thuật toán FP-Growth
- Giai đoạn 1: xây dựng cấu trúc cây
- Minh họa xây dựng cây FP-tree
- Mô tả bảng header và cấu trúc cây FP-tree
- Thủ tục cập nhật cây FP-tree với dãy mục
- Minh họa tìm tập mục phổ với FP-tree
- Giải thuật tìm các tập phổ biến với cây FP-tree

# Giới thiệu thuật toán FP-Growth

- Xây dựng cây mẫu phổ biến FP_tree (FP - frequent pattern) 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

# Các bước chính thuật toán FP-Growth

- Giai đoạn 1: Xây dựng cấu trúc cây
  - Duyệt cơ sở dữ liệu để tính độ phổ biến của từng mục (đếm số lần xuất hiện của từng mục)
  - Loại những mục không đủ độ phổ biến
  - Sắp xếp các mục còn lại theo thứ tự giảm dần của độ hỗ trợ -> thu được danh sách L các mục đã sắp
  - Duyệt CSDL lần thứ 2, với mỗi giao dịch t:
    - Loại các mục không đủ độ phổ biến
    - Các mục còn lại theo thứ tự giống như xuất hiện trong L và cất vào cây FP-tree
- Giai đoạn 2: Tìm các mẫu phổ biến trên cây FP-tree đã xây dựng (không cần duyệt cơ sở dữ liệu thêm một lần nào nữa)

# Giai đoạn 1: Xây dựng cấu trúc cây

- Mô tả xây dựng cấu trúc cây
- Minh họa xây dựng cây FP-tree
- Mô tả bảng header và cấu trúc cây FP-tree
- Thủ tục cập nhật cây FP-tree với dãy mục

# Mô tả xây dựng cấu trúc cây

- Xét CSDL sau đây, độ hỗ trợ cực tiểu minsupp = 3 (3/5 tức 60%)
- Gồm các bước:
  - B1: Duyệt CSDL, đếm số lần xuất hiện của từng mục, loại các mục không đủ độ phổ biến
  - B2: Sắp các mục đủ độ phổ biến theo thứ tự giảm dần của độ hỗ trợ => Nhận được danh sách L các mục đã được sắp xếp như bảng trên
  - B3: Duyệt CSDL lần hai và xây dựng FP-tree

| TID | Nội dung giao dịch |
|---|---|
| 1 | a,c,d,f,g,i,m,p |
| 2 | a,b,c,f,l,m,o |
| 3 | b,f,h,j,o |
| 4 | b,c,k,p,s |
| 5 | a,c,e,f,l,m,n,p |

| Mục | Tần số xuất hiện |
|---|---|
| f | 4 |
| c | 3 |
| a | 3 |
| m | 3 |
| p | 3 |

# Duyệt CSDL lần hai và xây dựng FP-tree

- Khởi tạo cây T, gốc của cây có nhãn null
- Duyệt cơ sở dữ liệu lần thứ 2, với mỗi giao dịch
  - Loại các mục không phổ biến
  - Sắp theo các mục còn lại thứ tự giảm dần về số lần xuất hiện

| TID | Nội dung giao dịch | Nội dung GD sau khi đã sắp xếp |
|---|---|---|
| 1 | a,c,d,f,g,i,m,p | f,c,a,m,p |
| 2 | a,b,c,f,l,m,o | f,c,a,b,m |
| 3 | b,f,h,j,o | F,b |
| 4 | b,c,k,p,s | C,b,p |
| 5 | a,c,e,f,l,m,n,p | F,c,a,m,p |

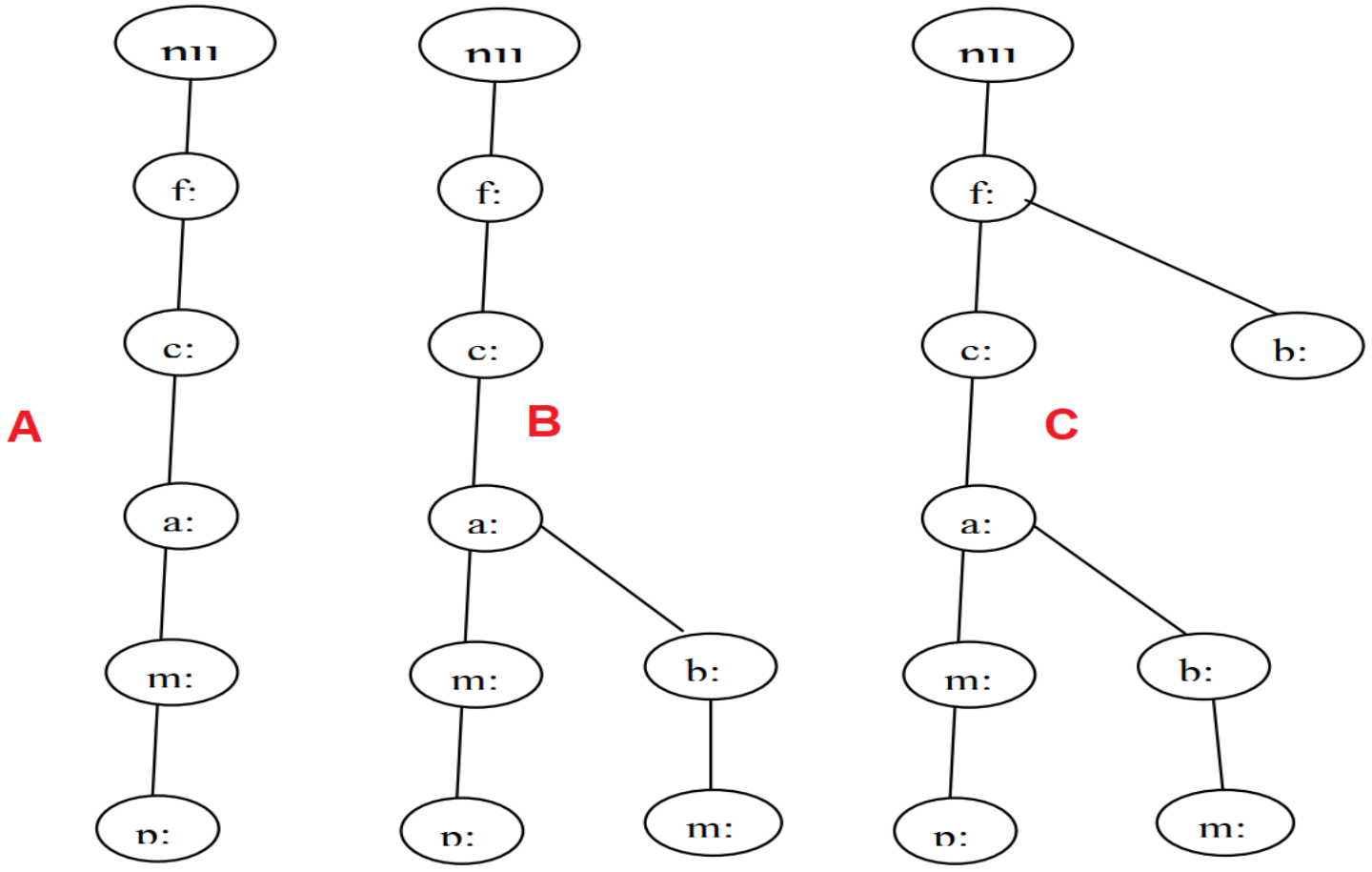
  - Thêm dãy các mục phổ biến đã sắp xếp vào cây cùng với thay đổi số đếm của các mục trên cây cho phù hợp

# Minh họa xây dựng cây FP-tree

- Cây FP-tree sau khi đọc giao dịch 1, 2, 3
- Cây FP-tree sau khi đọc tiếp giao dịch 4, 5
- Bảng header và cây FP-tree hoàn thiện

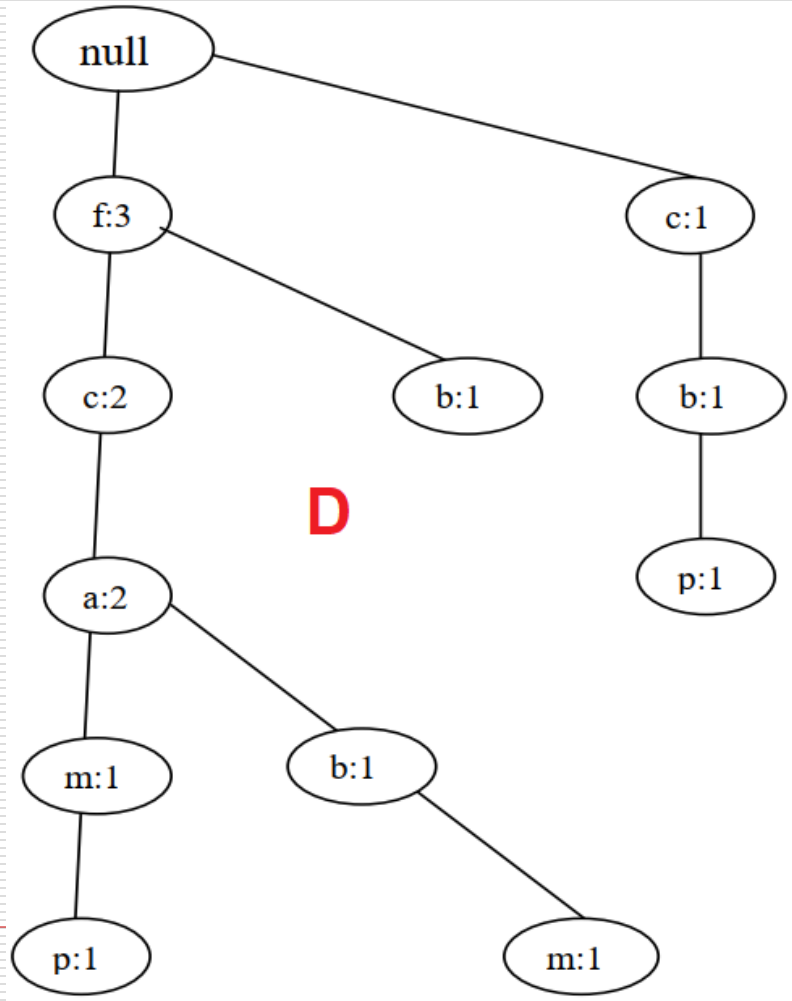
# Cây FP-tree sau khi đọc giao dịch 1, 2, 3

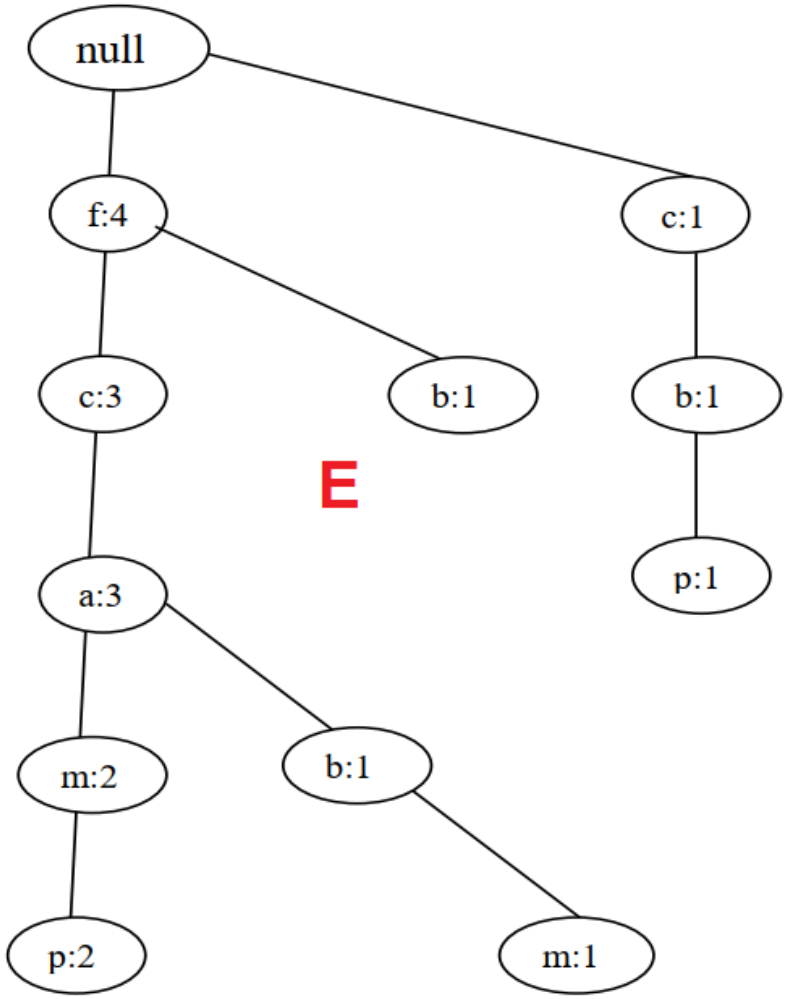
- A: Cây FP-tree xây dựng được sau khi đọc giao dịch 1
- B: Cây FP-tree xây dựng được sau khi đọc giao dịch 2
- C: Cây FP-tree xây dựng được sau khi đọc giao dịch 3



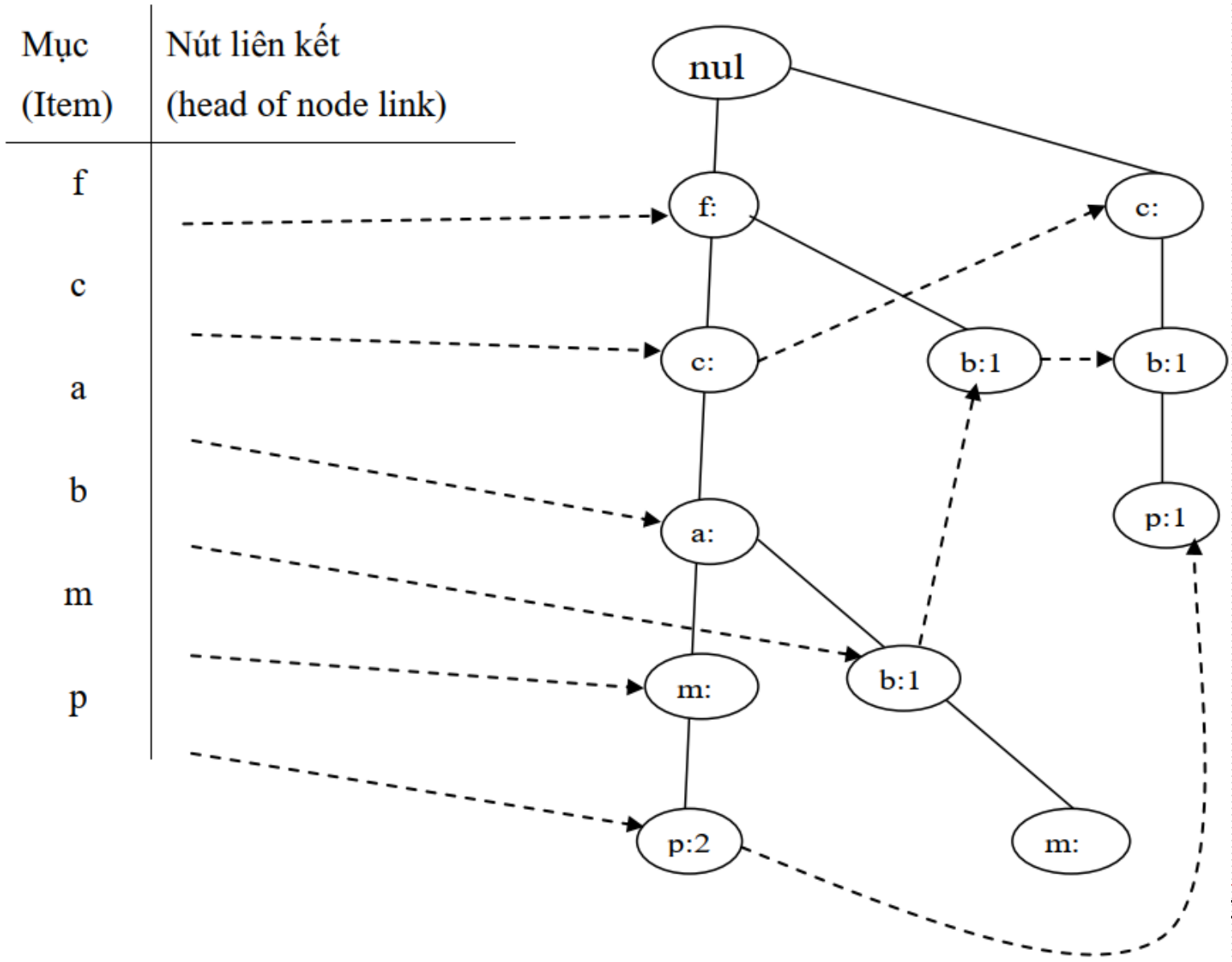
# Cây FP-tree sau khi đọc tiếp giao dịch 4, 5

- D: Cây FP-tree xây dựng được sau khi đọc giao dịch 4
- E: Cây FP-tree xây dựng được sau khi đọc giao dịch 5



# Bảng header và cây FP-tree hoàn thiện



# Mô tả bảng header và cấu trúc cây FP-tree

- Bảng các đầu mục phổ biến (header table)
  - Bắt đầu cho các liên kết
  - Mỗi đầu mục ứng với một dòng, có các liên kết tương ứng với nút cùng tên mục trên cây
- Gốc của cây có nhãn null
- Mỗi nút (trừ nút gốc) bao gồm:
  - Tên mục (item identifier)
  - Số đếm (Count)
  - Liên kết đến nút tiếp theo trên cây có cùng tên mục (Node link)
- Các liên kết trên cây: liên kết các nút cùng tên mục

# Thủ tục cập nhật cây FP-tree với dãy mục

- Dãy các mục (đã sắp giảm dần theo độ hỗ trợ) của các giao dịch vào cây thực hiện đệ quy như sau:
- Procedure insert_tree(string [p| P], tree có gốc T)
  - *1. Nếu T có nút con N mà N.itemname=p thì N.count ++*
  - *2. ngược lại*
  - *3. Tạo một nút mới N;*
  - *4. N.itemname:=p; N.count:=1;*
  - *5. Đặt liên kết của N bằng liên kết của mục p trong bảng*
  - *6. Thay đổi nút liên kết mục p trong bảng thành N;*
  - *7. Nếu P khác rỗng*
  - *8. Gọi thủ tục insert_tree(P,N);*
- p là mục thứ nhất của dãy các mục và P là phần còn lại
- Trong lần duyệt thứ 2, với mỗi giao dịch t, gọi thủ tục insert_tree(t‟,T):
  - t‟ là nội dung của t sau khi bỏ đi các mục không phổ biến và sắp theo thứ tự giảm dần của độ hỗ trợ; T là gốc của cây

# Minh họa tìm tập mục phổ với FP-tree

- Ý tưởng: Theo bảng header
  - Duyệt các item phổ biến mức 1 theo thứ tự tăng dần độ hỗ trợ là p, m, b, a, c, f
  - Với mỗi item, xây dựng các cơ sở mẫu điều kiện (conditional pattern-base) và các FP-Tree điều kiện (conditional FP-Tree)
- Bắt đầu với item p
  - Cơ sở mẫu điều kiện của nó là tất cả các đường dẫn tiền tố của cây FP-Tree khi duyệt từ gốc null đến nút p: là fcam:2 và cb:1 (theo sau là số lần xuất hiện của nút p tương ứng với mỗi nhánh đường dẫn)
  - Xây dựng FP-Tree điều kiện từ mẫu này bằng cách trộn tất cả các đường dẫn (theo thứ tự giảm dần các mục): fcam:2 và cb:1 trộn lại thành f:2, c:3, a:2, m:2, b:1
  - Giữ lại các nút có tổng các số đếm ≥ sup = 3: chỉ có c:3 là thoả mãn điều kiện => Các mẫu phổ biến chứa p là: p, cp

# Minh họa tìm tập mục phổ với với FP-tree

- Làm tương tự cho các item còn lại để tìm được các mẫu phổ biến chứa các item đó
- Cuối cùng có bảng sau đây:

| 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 |

# Giải thuật tìm các tập phổ biến với cây FP-tree

- Thủ tục chính tìm tập phổ biến
- Thủ tục: Tìm đường đi
- Thủ tục: Tìm tập phổ biến
- Thủ tục: Tìm phần tử chung

# Thủ tục chính tìm tập phổ biến

- 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 ∈L1
  - **3) TimDuongDi** (i, out MangDuongDi, out SoDD);
  - 4) **TimTapPhoBien** (i, MangDuongDi, SoDD)
- MangDuongDi: mảng các đường đi trong cây FP có chứa item i
- SoDD: số đường đi trong cây có chứa item i

# Thủ tục: Tìm đường đi

- Mục đích:
  - Tìm tất cả đường đi trong cây có chứa item i
- Input:
  - Item i
- Output:
  - MangDuongDi: là mảng các đường đi trong cây FP có chứa item i
  - SoDD: số đường đi trong cây có chứa item i

**Procedure TimDuongDi (Item i, string MangDuongDi)**

1) VT = i.HeadNodeLink; //Từ liên kết HeadNodeLink, xác định được ví 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 <> 0{
6) Support = N.Support
7) DuongDi = ∅
8) N1 = N
9) Duyệt ngược cây FP từ vị trí VT {
10) If N1.TenNutCha != <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

# Thủ tục: Tìm tập phổ biến

- 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 itm i
- Output: Tập các tập phổ biến

```
Procedure TimTapPhoBien(Item i, string MangDuongDi,
 int soDD)
1) j = 1
2) t = 1
3) while j <= SoDD{
4) Với mỗi kết hợp j phần tử trong MangDuongDi{
5) PhanTuChung = TimPhanTuChung (i, soPTChung)
6) If Support(PhanTuChung) >= Minsup then
7) t = 1
8) do while t<= SoPTChung{
9) Với mỗi kết hợp t phần tử k = {k1, k2, …k1}
10) Tạo ra tập phổ biến(k, i)}}
11) j = j+1}
```

# Thủ tục: Tìm phần tử chung

- Đặc điểm:
  - Tìm phần tử chung giữa j phần tử trong kết hợp, nếu có item giống nhau thì PhanTuChung = PhanTuChung + 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

# Khai phá LKH từ tập mục phổ biến

- Bài toán khai phá luật kết hợp
- Quy trình khai phá Luật kết hợp
- Ví dụ khai phá Luật kết hợp

# Bài toán khai phá luật kết hợp

- Input:
  - Một tập mục $I = \{I_1, I_2, \ldots, I_n\}$
  - CSDL giao dịch $D = \{T_1, T_2, \ldots, T_m\}$, $T_j \in I$
  - minsup, minconf
- Output:
  - Tất cả các luật dạng $X \Rightarrow Y$ ($X, Y \subseteq I$, $X \cap Y = \emptyset$) thỏa mãn độ phổ biến và độ tin cậy tối thiểu:
    - $supp(X \Rightarrow Y) \geq minsupp$
    - $conf(X \Rightarrow Y) \geq minconf$

# Quy trình khai phá Luật kết hợp

- Hầu hết các thuật toán đề xuất để khai phá dữ liệu nhờ luật kết hợp đều theo hướng chia bài toán thành hai pha như sau:
  - Pha 1:
    - Tìm tất cả các tập mục phổ biến từ cơ sở dữ liệu giao dịch, tức là tìm tất cả các tập mục X thỏa mãn Supp(X) >= minsupp
    - Tốn khá nhiều thời gian của CPU và thời gian vào ra ổ đĩa
  - Pha 2:
    - Sinh các luật kết hợp từ các tập phổ biến đã tìm thấy ở pha thứ nhất
    - Đơn giản và tốn kém ít thời gian so với pha 1
    - Nếu X là tập phổ biến thì luật kết hợp được sinh ra từ X có dạng: X ' => X \ X ‘ với độ tin cậy c >= minconf
    - X‟ là tập con khác rỗng của X, X \ X‟ là hiệu của 2 tập hợp

# Ví dụ khai phá Luật kết hợp

| Mã giao dịch | Tập mục |
|---|---|
| T1 | A, C, D |
| T2 | B, E |
| T3 | A, B, C, E |
| T4 | B, E |
| T5 | B, D, F |

- Cho CSDL D
- Số giao dịch: 5
- Số các mục: 6
- Xác định độ phổ biến các tập mục
  - Ví dụ: sup(A) = 2/5 = 40%

| Tập mục | Số lần xuất hiện | Supp |
|---|---|---|
| A | 2 | 40% |
| B | 4 | 80% |
| C | 2 | 40% |
| D | 2 | 40% |
| E | 3 | 60% |
| F | 1 | 20% |
| A,C | 2 | 40% |
| A,B | 1 | 20% |
| B,D | 1 | 20% |
| C,D | 1 | 20% |
| A,B,C | 1 | 20% |
| A,B,E | 1 | 20% |
| A,C,D | 1 | 20% |
| B,D,F | 1 | 20% |
| A,B,C,E | 1 | 20% |

- Từ các tập phổ biến => Xác định được luật kết hợp và độ tin cậy

# Ví dụ khai phá Luật kết hợp

- Từ các tập phổ biến => Xác định được luật kết hợp và độ tin cậy
- Ví dụ: với tập mục: A,C thì ta có luật kết hợp là: A => C

$$conf(A \Rightarrow C) = \frac{supp(A \cup C)}{supp(A)} = \frac{2}{2} = 100\%$$

- Tương tự với các luật khác:

| Luật kết hợp | Độ tin cậy |
|---|---|
| $A \rightarrow C$ | 100% |
| $A \rightarrow B$ | 50% |
| $B \rightarrow D$ | 25% |
| $C \rightarrow D$ | 50% |
| $A,B \rightarrow C$ | 100% |
| $A ,C \rightarrow B$ | 50% |
| $C,B \rightarrow A$ | 100% |


THE END