KHAI THÁC DỮ LIỆU
(Data Mining)

Lê Văn Hạnh
levanhanhvn@gmail.com

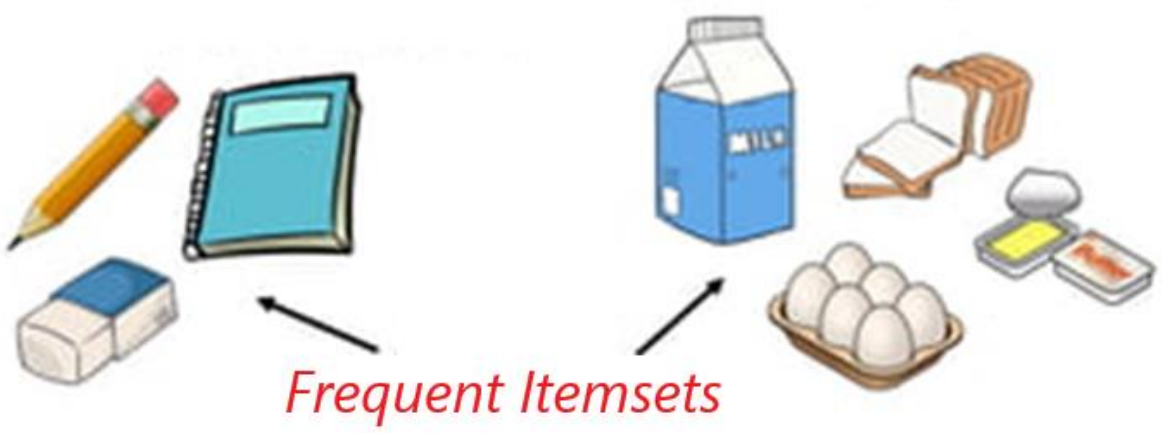
NỘI DUNG MÔN HỌC

1. Tổng quan về Data Science
2. Tìm hiểu dữ liệu
3. Tiền xử lý dữ liệu
4. Khai thác các mẫu phổ biến, mối kết hợp và mối tương quan
5. Phân loại (*Classification*)
6. Phân tích cụm (*Cluster analysis*)

Chương 4

CÁC PHƯƠNG PHÁP CƠ BẢN VỀ KHAI THÁC CÁC MẪU PHỔ BIẾN, MỐI KẾT HỢP VÀ MỐI TƯƠNG QUAN

(Mining Frequent Patterns, Associations, and Correlations: Basic Concepts and Methods)



Lê Văn Hạnh
levanhanhvn@gmail.com

NỘI DUNG CHƯƠNG 4

1. Một số khái niệm cơ bản
2. Các phương pháp khai thác tập mục phổ biến
3. Các phương pháp đánh giá mẫu
4. Bài tập

1. MỘT SỐ KHÁI NIỆM CƠ BẢN

- Các mẫu phổ biến (*Frequent patterns hay* tập mục - *itemsets*, hoặc chuỗi con - *subsequences* hoặc cấu trúc con - *substructures*) là các mẫu xuất hiện thường xuyên trong một tập dữ liệu.

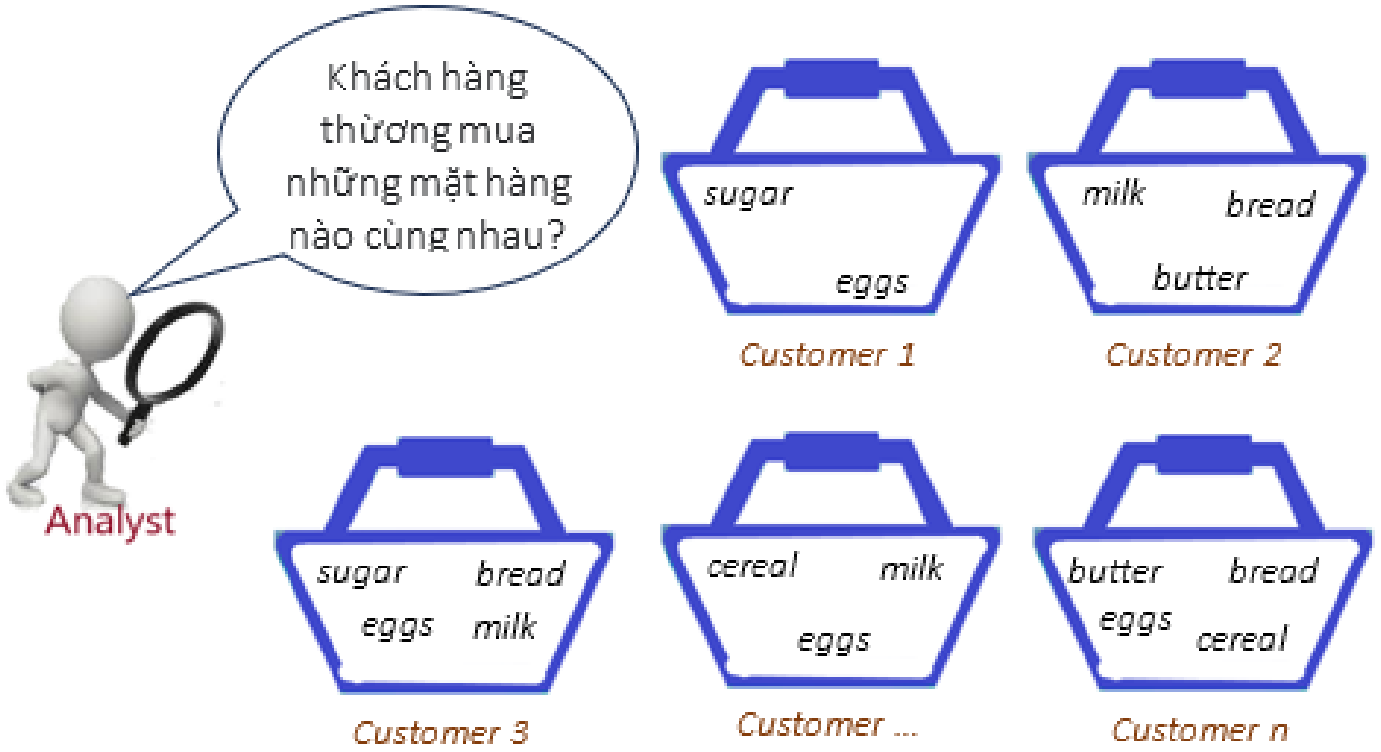
  Ví dụ, trong tập dữ liệu bán hàng, người ta nhận thấy các tập mục (*itemsets*) sau thường xuất hiện cùng nhau trong 1 hóa đơn hoặc của cùng 1 khách hàng như:

  - Sữa và bánh mì (khi mua sữa thương mua kèm với bánh mì).
  - Máy ảnh kỹ thuật số, thẻ nhớ, PC (khi mua máy chụp ảnh, thường mua kèm thẻ nhớ, có thể là ngay khi đó hoặc là sau đó khách hàng sẽ mua PC).
  - …

- Nếu các *itemsets* đó xuất hiện thường xuyên trong CSDL về lịch sử mua sắm, thì *itemsets* đó được gọi là các mẫu phổ biến (*frequent patterns*) hay mẫu tuần tự (*sequential pattern*).

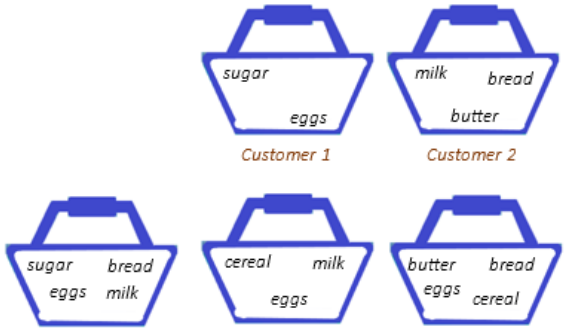
1.1. Phân tích giỏ hàng

- Việc khai thác tập mục phổ biến dẫn đến việc phát hiện ra các mối liên kết và mối tương quan giữa các mục trong các tập dữ liệu quan hệ hoặc giao dịch lớn.
- Với lượng dữ liệu khổng lồ liên tục được thu thập và lưu trữ, nhiều ngành công nghiệp đang quan tâm đến việc khai thác các mẫu như vậy từ CSDL của họ.
- Việc phát hiện ra mối quan hệ tương quan giữa số lượng lớn hồ sơ giao dịch kinh doanh có thể giúp ích trong nhiều quá trình ra quyết định kinh doanh như thiết kế danh mục, tiếp thị chéo và phân tích hành vi mua sắm của khách hàng.



1.1. Phân tích giỏ hàng

- Có thể gán cho mỗi mặt hàng một biến Boolean biểu thị sự hiện diện hay vắng mặt của mặt hàng đó.



Customer	Sugar	Eggs	Milk	Bread	Butter	Cereal
1	T	T	F	F	F	F
2	F	F	T	T	T	F
3	T	T	T	T	F	F
...	F	T	T	F	F	T
n	F	T	F	T	T	T

- Sau đó, mỗi giỏ có thể được biểu diễn bằng một vectơ Boolean gồm các giá trị được gán cho các biến này. Các vectơ Boolean có thể được phân tích để tìm ra các mẫu mua phản ánh các mặt hàng thường được liên kết hoặc mua cùng nhau. Những mẫu này có thể được biểu diễn dưới dạng luật kết hợp (*association rules*). Ví dụ:

Eggs ⇒ Sugar, Milk, Bread, Cereal [support = 40%, confidence = 60%] (rule 1)
Milk ⇒ Eggs, Bread [support = 45%, confidence = 55%] (rule 2)
Bread, Eggs ⇒ Milk [support = 37%, confidence = 42%] (rule 3)
…. (rule m)

1.1. Phân tích giỏ hàng

Eggs ⇒ Sugar, Milk, Bread, Cereal [support = 40%, confidence = 60%] (rule 1)
Milk ⇒ Eggs, Bread [support = 45%, confidence = 55%] (rule 2)
…

- Độ hỗ trợ (support) và độ tin cậy (confidence) là hai thước đo phản ánh tính hữu ích và tính chắc chắn của các luật kết hợp được khám phá.
- Mức hỗ trợ 40% cho Rule_1 có nghĩa là trong số tất cả các giao dịch được phân tích cho thấy 40% khách hàng khi mua Eggs sẽ mua thêm Sugar, Milk, Bread, Cereal.
- Độ tin cậy 60% (rule 1)có nghĩa là 60% khách hàng mua Eggs sẽ mua thêm Sugar, Milk, Bread, Cereal.
- Thông thường, các luật kết hợp được coi là hữu dụng nếu chúng thỏa mãn cả ngưỡng hỗ trợ tối thiểu (min_sup) và ngưỡng tin cậy tối thiểu (min_conf). **Các ngưỡng này** có thể do **người dùng hoặc chuyên gia về lĩnh vực đặt ra**. Phân tích bổ sung có thể được thực hiện để khám phá mối tương quan thống kê hữu ích giữa các mục liên quan.

1.2. Tập mục phổ biến *(Frequent Itemsets)*, tập mục đóng *(Closed Itemsets)* và luật kết hợp *(Association Rules)*

- Cho:
  - $I = \{I_1, I_2, ..., I_m\}$ là một tập mục.
  - Giả sử D là một tập hợp các giao dịch CSDL trong đó mỗi giao dịch T (*Transaction*) là một tập mục không trống sao cho $T \subseteq I$.
  - Mỗi giao dịch được liên kết với một mã định danh, được gọi là TID.
- Cho A là một tập hợp các mục. Giao dịch T được gọi là chứa A nếu $A \subseteq T$. Luật kết hợp có dạng $A \Rightarrow B$, trong đó $A \subset I$, $B \subset I$, $A \neq \emptyset$, $B \neq \emptyset$ và $A \cap B = \varphi$.
- Luật $A \Rightarrow B$ có độ hỗ trợ (*Support*) s, trong đó s là tỷ lệ phần trăm các giao dịch trong D chứa $A \cup B$ (tức là hợp của tập A và B, hoặc cả A và B).
- Luật $A \Rightarrow B$ có độ tin cậy (*Confidence*) C trong tập giao dịch D, trong đó C là tỷ lệ phần trăm các giao dịch trong D cùng chứa A và B.

1.2. Tập mục phổ biến *(Frequent Itemsets)*, tập mục đóng *(Closed Itemsets)* và luật kết hợp *(Association Rules)*

- Theo quy ước, giá trị hỗ trợ và độ tin cậy sẽ sử dụng giá trị phần trăm (trong khoảng từ 0% đến 100%), thay vì sử dụng giá trị số thực (từ 0 đến 1,0).
- Các luật thỏa mãn cả ngưỡng hỗ trợ tối thiểu (*minimum support –min_sup*) và ngưỡng tin cậy tối thiểu (*minimum confidence – min_conf*) được gọi là ***mạnh***.
- Một tập hợp các mục được gọi là một tập mục (itemset).
- Một tập mục chứa k mục là một tập mục k. Bộ {Bread, Milk} là bộ gồm 2 mục.
- Tần suất xuất hiện của một tập mục là số lượng giao dịch có chứa tập mục đó. Điều này còn được gọi đơn giản là tần suất (*frequency*), số lượng hỗ trợ (*support count*) hoặc số lượng (*count*) của tập mục.
- Lưu ý rằng itemset support được gọi là độ hỗ trợ tương đối, trong khi tần suất xuất hiện được gọi là độ hỗ trợ tuyệt đối.
- Nếu độ hỗ trợ tương đối của tập mục I thỏa mãn ngưỡng hỗ trợ tối thiểu được xác định trước (nghĩa là độ hỗ trợ tuyệt đối của I thỏa mãn ngưỡng số lượng hỗ trợ tối thiểu tương ứng), thì I là tập phổ biến (a frequent itemset). Tập k-mục phổ biến thường được ký hiệu là L_k.

1.2. Tập mục phổ biến *(Frequent Itemsets)*, tập mục đóng *(Closed Itemsets)* và luật kết hợp *(Association Rules)*

- Công thức:

$$\text{confidence}(A \Rightarrow B) = P(B|A) = \frac{\text{Support}(A \cup B)}{\text{Support}(A)} = \frac{\text{Support\_count}(A \cup B)}{\text{Support\_count}(A)}$$

Qua công thức, cho thấy độ tin cậy của luật A ⇒ B có thể dễ dàng suy ra từ số lượng hỗ trợ (*support counts*) của A và A ∪ B. Nghĩa là, khi tìm thấy số lượng hỗ trợ của A, B và A ∪ B, có thể dễ dàng rút ra các luật kết hợp tương ứng A ⇒ B và B ⇒ A và kiểm tra xem chúng có mạnh hay không.

Vì vậy, vấn đề khai phá các luật kết hợp có thể được giảm xuống thành vấn đề khai phá các tập phổ biến (*frequent itemsets*).

1.2. Tập mục phổ biến *(Frequent Itemsets)*, tập mục đóng *(Closed Itemsets)* và luật kết hợp *(Association Rules)*

- Quá trình khai phá luật kết hợp gồm hai bước:
  - **i.** ***Tìm tất cả các tập phổ biến*** (*frequent itemsets*): Theo định nghĩa, tần suất xuất hiện của mỗi tập mục này sẽ lớn hơn hay bằng số lượng hỗ trợ tối thiểu (*min_sup*) được xác định trước.
  - **ii.** ***Tạo ra các luật kết hợp mạnh*** (*strong association rules*) từ các tập phổ biến (*frequent itemsets*): Theo định nghĩa, các luật này phải đáp ứng độ hỗ trợ tối thiểu (*min_sup*) và độ tin cậy tối thiểu (*min_conf*).

1.2. Tập mục phổ biến *(Frequent Itemsets)*, tập mục đóng *(Closed Itemsets)* và luật kết hợp *(Association Rules)*

- Một thách thức lớn trong việc khai thác các tập phổ biến từ một tập dữ liệu lớn là việc khai thác như vậy thường tạo ra một số lượng lớn các tập mục thỏa mãn ngưỡng hỗ trợ tối thiểu (*min_sup*), đặc biệt khi *min_sup* được đặt ở mức thấp. Điều này là do nếu một tập mục là phổ biến thì mỗi tập con của nó cũng phổ biến. Một tập mục dài sẽ chứa một số tổ hợp các tập mục con ngắn hơn, phổ biến hơn.

- Ví dụ: một tập phổ biến có độ dài 100, chẳng hạn như $\{a_1, a_2, \dots, a_{100}\}$, chứa (100/1=) 100 tập phổ biến 1 mục: $\{a_1\}$, $\{a_2\}$, ..., (100/2) tập phổ biến 2 mục: $\{a_1, a_2\}$, $\{a_1, a_3\}$,...,$\{a_{99}, a_{100}\}$; và tương tự cho các tập từ 3 mục trở lên. Do đó, tổng số tập mục phổ b $\frac{100}{1} + \frac{100}{2} + \dots + \frac{100}{100} = 2^{100}-1 \cong 1.27 \times 10^{30}$

$\Rightarrow$ Số lượng itemsets quá lớn để bất kỳ máy tính nào có thể tính toán hoặc lưu trữ.

1.2. Tập mục phổ biến *(Frequent Itemsets)*, tập mục đóng *(Closed Itemsets)* và luật kết hợp *(Association Rules)*

- Để khắc phục khó khăn này, cần nắm thêm các khái niệm về tập phổ biến đóng (closed frequent itemset) và tập phổ biến tối đại (maximal frequent itemset).

Ví dụ: Giả sử CSDL D chỉ có hai giao dịch: $\{<a_1, a_2, \dots, a_{100}>; <a_1, a_2, \dots, a_{50}>\}$.

Đặt ngưỡng số lượng hỗ trợ tối thiểu là min_sup = 1. Ta tìm thấy:

- Hai tập phổ biến đóng (*closed frequent itemsets*) và số lượng hỗ trợ (*support count*) của chúng, đó là C = $\{\{a_1, a_2, \dots, a_{100}\} : 1; \{a_1, a_2, \dots, a_{50}\} : 2\}$.
- Một tập phổ biến tối đại: M = $\{\{a_1, a_2, \dots, a_{100}\} : 1\}$.

Lưu ý: không thể bao gồm tập $\{a_1, a_2, \dots, a_{50}\}$ làm tập phổ biến tối đại vì nó có tập cha phổ biến là $\{a_1, a_2, \dots, a_{100}\}$.

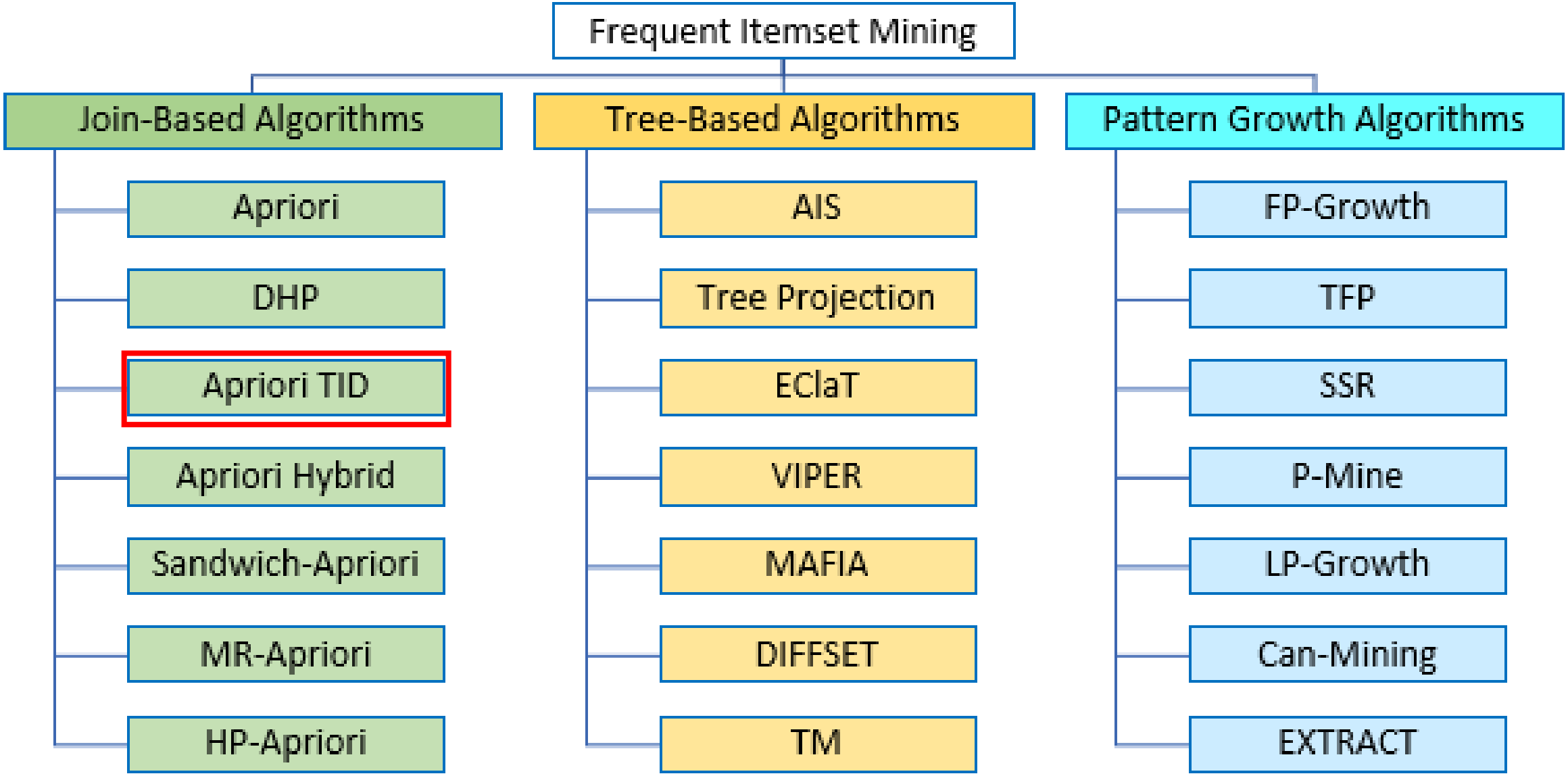
- Tập các tập phổ biến đóng (*set of closed frequent itemsets*) chứa thông tin đầy đủ về các tập phổ biến. Ví dụ, từ C, ta có thể suy ra, chẳng hạn:
  i. $\{a_2, a_{45} : 2\}$ vì $\{a_2, a_{45}\}$ là tập mục con của tập mục $\{a_1, a_2, \dots, a_{50} : 2\}$;
  ii. $\{a_8, a_{55} : 1\}$ vì $\{a_8, a_{55}\}$ không phải là tập mục con của tập mục $\{a_1, a_2, \dots, a_{50} : 2\}$ mà của tập mục $\{a_1, a_2, \dots, a_{100} : 1\}$.
- Tuy nhiên, từ tập mục phổ biến tối đại, ta chỉ có thể khẳng định rằng cả hai tập mục ($\{a_2, a_{45}\}$ và $\{a_8, a_{55}\}$) đều phổ biến, nhưng ta không thể khẳng định số lượng hỗ trợ thực tế của chúng.

NỘI DUNG CHƯƠNG 4

1. Một số khái niệm cơ bản
2. Các phương pháp khai thác tập mục phổ biến
3. Các phương pháp đánh giá mẫu
4. Bài tập

2. CÁC PHƯƠNG PHÁP KHAI THÁC TẬP MỤC PHỔ BIẾN

(Frequent Itemset Mining Methods)



2.1. Thuật toán Apriori: Tìm các tập mục phổ biến (Frequent Itemsets) bằng cách tạo ứng viên bị giới hạn (*Confined Candidate)*

- Apriori là một thuật toán cơ bản được đề xuất bởi R. Agrawal và R. Srikant vào năm 1994 để khai thác các tập phổ biến cho luật kết hợp Boolean (*Boolean association rules*).
- Apriori sử dụng một cách tiếp cận lặp đi lặp lại được gọi là tìm kiếm theo cấp độ, trong đó các tập mục k được sử dụng để khám phá các tập mục (k+1).
  i. Đầu tiên, tập hợp các tập gồm 1 mục phổ biến được tìm thấy bằng cách quét CSDL để tích lũy số lượng cho từng mục và thu thập các mục đó thỏa mãn độ hỗ trợ tối thiểu. Tập kết quả được ký hiệu là L_1.
  ii. Sử dụng L_1 để tìm L_2, tập hợp gồm 2 mục phổ biến,
  iii. Sử dụng L_2 để tìm L_3, tập hợp gồm 3 mục phổ biến,
  iv. Thực hiện lặp lại cho đến khi không thể tìm thấy tập k+1 mục phổ biến nào nữa.

  Lưu ý: Việc tìm ra mỗi L_k yêu cầu phải quét toàn bộ CSDL một lần.

2.1. Thuật toán Apriori: Tìm các tập mục phổ biến *(Frequent Itemsets)* bằng cách tạo ứng viên bị giới hạn *(Confined Candidate)*

- Để nâng cao hiệu quả của việc tạo các tập phổ biến theo cấp độ, một thuộc tính quan trọng được gọi là thuộc tính Apriori được sử dụng để giảm không gian tìm kiếm.
- ***Thuộc tính Apriori***: Tất cả các tập con khác rỗng của tập phổ biến cũng phải là tập phổ biến.

2.1. Thuật toán Apriori: Tìm các tập mục phổ biến *(Frequent Itemsets)* bằng cách tạo ứng viên bị giới hạn *(Confined Candidate)*

2.1.1. Thuật toán Apriori

1. **Bước nối (*join*)**: Để tìm L_k, một tập k-mục ứng viên được tạo ra bằng cách nối (join - $\triangleright\triangleleft$) L_{k-1} với chính nó. Tập ứng viên này (*candidates*) được ký hiệu là C_k. Cho l_1 và l_2 là các tập mục trong L_{k-1}. Ký hiệu $l_i[j]$ đề cập đến mục thứ j trong l_i (ví dụ: $l_1[k-2]$ đề cập đến mục thứ hai đến mục cuối cùng trong l_1). Đối với tập mục $(k-1)$, l_i, điều này có nghĩa là các mục được sắp xếp sao cho $l_i[1] < l_i[2] < \cdots < l_i[k-1]$. Phép nối, $L_{k-1} \triangleright\triangleleft Lk_{-1}$, được thực hiện, trong đó các thành viên của L_{k-1} có thể nối được nếu các mục đầu tiên $(k-2)$ của chúng. Nghĩa là, các thành viên l_1 và l_2 của L_{k-1} được nối nếu $(l_1[1] = l_2[1]) \wedge (l_1[2] = l_2[2]) \wedge \cdots \wedge (l_1[k-2] = l_2[k-2]) \wedge (l_1[k-1] < l_2[k-1])$. Điều kiện $l_1[k-1] < l_2[k-1]$ chỉ đơn giản đảm bảo rằng không có bản sao nào được tạo ra. Tập mục kết quả được hình thành bằng cách nối l_1 và l_2 là $\{l_1[1], l_1[2], \ldots, l_1[k-2], l_1[k-1], l_2[k-1]\}$.
2. **Bước cắt tỉa (*prune*)**: Ck là siêu tập (*superset*) của L_k, nghĩa là các thành viên của nó có thể phổ biến hoặc không phổ biến, nhưng tất cả *k-mục* (*k-itemset*) mục phổ biến đều có trong C_k. Việc quét CSDL để xác định số lượng mỗi ứng viên trong C_k sẽ dẫn đến việc xác định L_k. Tuy nhiên, C_k có thể rất lớn. Để giảm kích thước của C_k, tính chất Apriori được sử dụng như sau. Bất kỳ tập mục $(k-1)$ nào không phổ biến thì không thể là tập con của tập mục k phổ biến. Do đó, nếu bất kỳ tập con $(k-1)$ nào của tập k mục ứng viên không thuộc L_{k-1} thì ứng viên đó cũng không thể phổ biến và do đó có thể bị loại khỏi C_k.

2.1. Thuật toán Apriori: Tìm các tập mục phổ biến *(Frequent Itemsets)* bằng cách tạo ứng viên bị giới hạn *(Confined Candidate)*

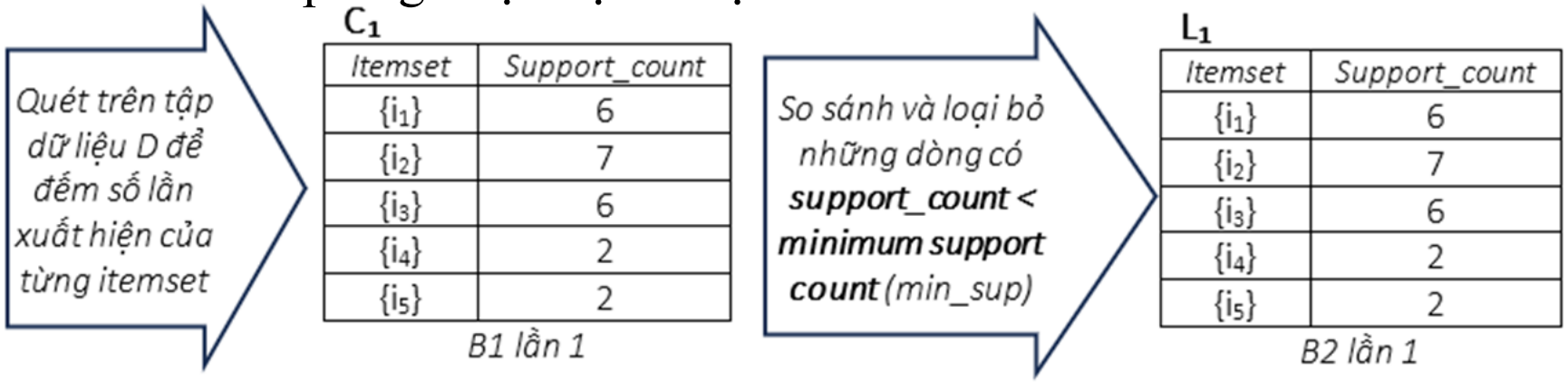
2.1.1. Thuật toán Apriori

- Ví dụ: Cho CSDL D gồm 9 giao dịch (|D|=9). Với số lượng hỗ trợ tối thiểu (min_sup) là 2 (độ hỗ trợ tương ứng là 2/9 = 22%). Hãy tìm các tập ứng viên?

TID	List of item IDs
T1	I1, I2, I5
T2	I2, I4
T3	I2, I3
T4	I1, I2, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I1, I2, I3, I5
T9	I1, I2, I3

i. Lặp lần 1:

- ***B1***: Quét trên tập dữ liệu D để phát sinh tập ứng viên (candidate) C_1, với mỗi itemset chỉ gồm 1 mục (1-item). Thuật toán chỉ cần quét tất cả các giao dịch để đếm số lần xuất hiện của 1-item.
- ***B2***: Với số lượng hỗ trợ tối thiểu (min_sup) = 2, do đó bất kỳ 1-item nào có số lần xuất hiện từ 2 lần trở lên sẽ thuộc về tập hợp L_1 Trong ví dụ, tất cả các ứng viên ở C1 đều đáp ứng được độ hỗ trợ tối thiểu.



Quét trên tập dữ liệu D để phát sinh ra C1.
Đưa những itemset có support_count < min_sup *vào L1*

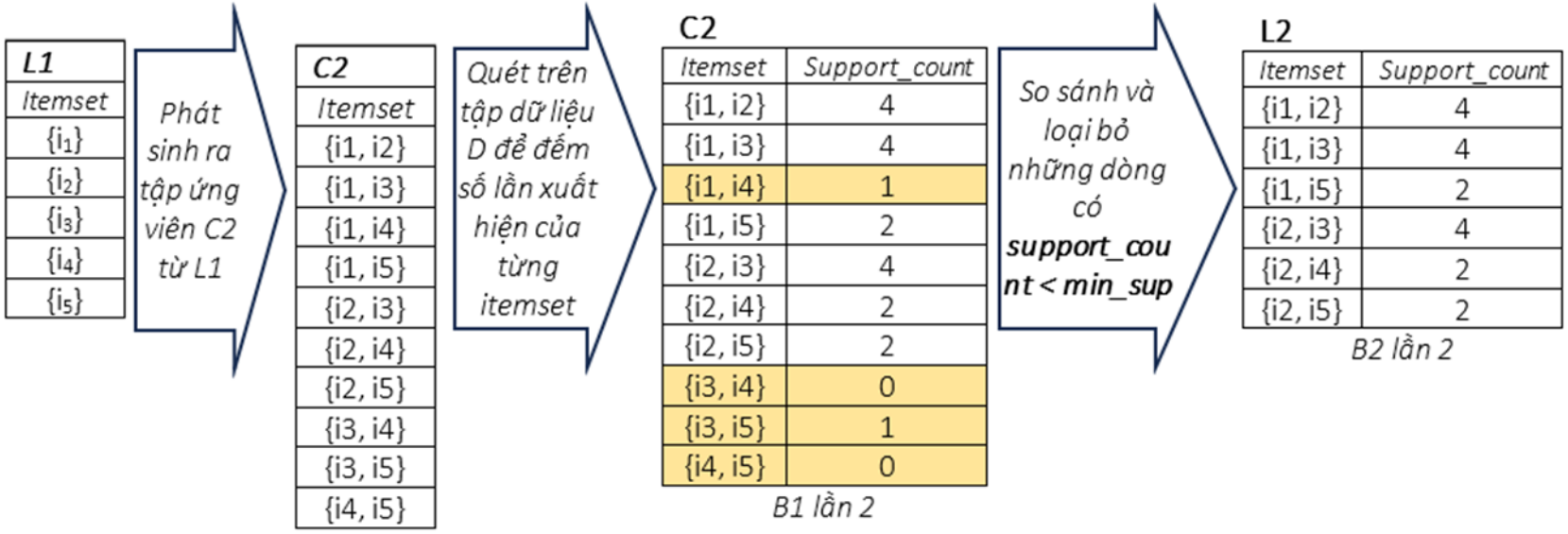
2.1.1. Thuật toán Apriori

TID	List of item IDs
T1	I1, I2, I5
T2	I2, I4
T3	I2, I3
T4	I1, I2, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I1, I2, I3, I5
T9	I1, I2, I3

- Ví dụ: Cho CSDL D và số lượng hỗ trợ tối thiểu (min_sup) là 2. Tìm các các tập ứng viên?

ii. Lặp lần 2:

- ***B1***:
  - Phát sinh ra tập ứng viên C_2 từ L_1 bằng cách kết hợp từng itemset trong L_1 với từng itemset có trong L_1 ($L_1 \triangleright\triangleleft L_1$):
  - Rà quét trên tập dữ liệu D để đếm số lần xuất hiện của từng itemset.
- ***B2***: So sánh và loại bỏ những dòng có *support_count < minimum support count (min_sup):* có 4 item set bị cắt tỉa do có *support_count < min_sup.*



Phát sinh C2 từ L1. Tìm support count cho các itemsets.
Đưa những itemset có support_count < min_sup *vào L2*

2.1.1. Thuật toán Apriori

- Ví dụ: Cho CSDL D và số lượng hỗ trợ tối thiểu (min_sup) là 2. Tìm các tập ứng viên?

iii. Lặp lần 3:

- ***B1***:
  - Phát sinh ra tập ứng viên C_3 từ L_2 bằng cách kết hợp từng itemset trong L_2 với từng itemset có trong L_2 ($L_2 \triangleright\triangleleft L_2$), tức là:

{{i1, i2}, {i1, i3}, {i1, i5}, {i2, i3}, {i2, i4}, {i2, i5}} ▷◁ {{i1, i2}, {i1, i3}, {i1, i5}, {i2, i3}, {i2, i4}, {i2, i5}}

Dựa trên thuộc tính Apriori rằng tất cả các tập con của một tập phổ biến cũng phải phổ biến, xét các itemset này, ta thu được C3 = { {i1, i2, i3}, {i1, i2, i5} }:

L2	*L2*	*C3= L2▷◁L2*	*Thứ*	*Giải thích*
Itemset	*Itemset*	*Itemset*	*tự*	
{i1, i2}	{i1, i2}	{i1, i2}	1	Chỉ gồm 2 item nên loại
	{i1, i3}	{i1, i2, i3}	2	Itemset hợp lệ
	{i1, i5}	{i1, i2, i5}	3	Itemset hợp lệ
	{i2, i3}	{i1, i2, i3}	4	Trùng 2
	{i2, i4}	{i1, i2, i4}	5	Không thể phát sinh vì {i1, i4} đã bị cắt tỉa ở bước trước
	{i2, i5}	{i1, i2, i5}	6	Trùng 3
{i1, i3}	{i1, i2}	{i1, i3, i2}	7	Trùng 2
	{i1, i3}	{i1, i3}	8	Chỉ gồm 2 item nên loại
	{i1, i5}	{i1, i3, i5}	9	Không thể phát sinh vì {i3, i5} đã bị cắt tỉa ở bước trước
	{i2, i3}	{i1, i3, i2}	10	Trùng 2
	{i2, i4}	{i1, i3, i2, i4}	11	Gồm 4 item nên chưa xét ở bước này=>loại
	{i2, i5}	{i1, i3, i2, i5}	12	Gồm 4 item nên chưa xét ở bước này=>loại

2.1.1. Thuật toán Apriori

- Ví dụ: Cho CSDL D và số lượng hỗ trợ tối thiểu (min_sup) là 2. Tìm các tập ứng viên?

iii. Lặp lần 3:

- ***B1***:
  - Phát sinh ra tập ứng viên C_3 từ L_2 :

L2 *Itemset*	*L2* *Itemset*	*C3= L2▷◁L2* *Itemset*	*Thứ tự*	*Giải thích*
{i1, i4}	{i1, i2}	{i1, i4, i2}	13	Không thể phát sinh vì {i1, i4} đã bị cắt tỉa ở bước trước
	{i1, i3}	{i1, i4, i3}	14	Không thể phát sinh vì {i3, i4} đã bị cắt tỉa ở bước trước
	{i1, i5}	{i1, i4, i5}	15	Không thể phát sinh vì {i4, i5} đã bị cắt tỉa ở bước trước
	{i2, i3}	{i1, i4, i2, i3}	16	Gồm 4 item nên chưa xét ở bước này=>loại
	{i2, i4}	{i1, i4, i2}	17	Không thể phát sinh vì {i1, i4} đã bị cắt tỉa ở bước trước
	{i2, i5}	{i1, i4, i2, i5}	18	Gồm 4 item nên chưa xét ở bước này=>loại
{i1, i5}	{i1, i2}	{i1, i5, i2}	19	Trùng 3
	{i1, i3}	{i1, i5, i3}	20	Không thể phát sinh vì {i3, i5} đã bị cắt tỉa ở bước trước
	{i1, i5}	{i1, i5 }	21	Chỉ gồm 2 item nên loại
	{i2, i3}	{i1, i5, i2, i3}	22	Gồm 4 item nên chưa xét ở bước này=>loại
	{i2, i4}	{i1, i5, i2, i4}	23	Gồm 4 item nên chưa xét ở bước này=>loại
	{i2, i5}	{i1, i5, i2}	24	Trùng 3
{i2, i3}	{i1, i2}	{i2, i3, i1}	25	Trùng 2
	{i1, i3}	{i2, i3, i1}	26	Không thể phát sinh vì {i1, i4} đã bị cắt tỉa ở bước trước
	{i1, i5}	{i2, i3, i1,i5}	27	Không thể phát sinh vì {i3, i5} đã bị cắt tỉa ở bước trước
	{i2, i3}	{i2, i3}	28	Chỉ gồm 2 item nên loại
	{i2, i4}	{i2, i3, i1}	29	Trùng 2
	{i2, i5}	{i2, i3, i5}	30	Không thể phát sinh vì {i3, i5} đã bị cắt tỉa ở bước trước
{i2, i4}	{i1, i2}	{i2, i4, i1}	31	Không thể phát sinh vì {i1, i4} đã bị cắt tỉa ở bước trước
	{i1, i3}	{i2, i4, i1, i3}	32	Không thể phát sinh vì {i3, i4} đã bị cắt tỉa ở bước trước
	{i1, i5}	{i2, i4, i1, i5}	33	Không thể phát sinh vì {i4, i5} đã bị cắt tỉa ở bước trước
	{i2, i3}	{i2, i4, i3}	34	Không thể phát sinh vì {i3, i4} đã bị cắt tỉa ở bước trước
	{i2, i4}	{i2, i4}	35	Chỉ gồm 2 item nên loại
	{i2, i5}	{i2, i4, i5}	36	Không thể phát sinh vì {i4, i5} đã bị cắt tỉa ở bước trước
{i2, i5}	{i1, i2}	{i2, i5, i1}	37	Trùng 3
	{i1, i3}	{i2, i5, i1, i3}	38	Không thể phát sinh vì {i3, i5} đã bị cắt tỉa ở bước trước
	{i1, i5}	{i2, i5, i1}	39	Không thể phát sinh vì {i4, i5} đã bị cắt tỉa ở bước trước
	{i2, i3}	{i2, i5, i3}	40	Không thể phát sinh vì {i3, i5} đã bị cắt tỉa ở bước trước
	{i2, i4}	{i2, i5, i4}	41	Không thể phát sinh vì {i4, i5} đã bị cắt tỉa ở bước trước
	{i2, i5}	{i2, i5}	42	Chỉ gồm 2 item nên loại

2.1.1. Thuật toán Apriori

TID	List of item IDs
T1	I1, I2, I5
T2	I2, I4
T3	I2, I3
T4	I1, I2, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I1, I2, I3, I5
T9	I1, I2, I3

- Ví dụ: Cho CSDL D và số lượng hỗ trợ tối thiểu (min_sup) là 2. Tìm các tập ứng viên?

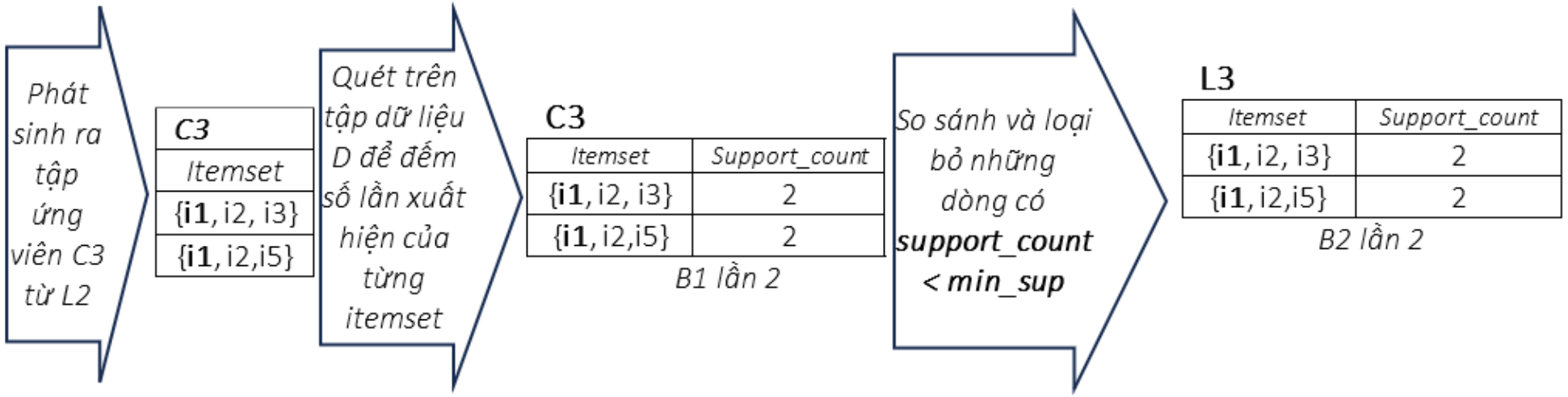
iii. Lặp lần 3:

- ***B1***:

  Nhờ việc xác định các item set cho C_3= { {i1, i2, i3}, {i1, i2, i5} }, giúp tiết kiệm công sức lấy số lượng của chúng một cách không cần thiết trong quá trình quét D tiếp theo để xác định L_3.

  - Rà quét trên tập dữ liệu D để đếm số lần xuất hiện của từng itemset.

- ***B2***: So sánh và loại bỏ những dòng có *support_count < minimum support count* *(min_sup)*: có 4 item set bị cắt tỉa do có *support_count < min_sup.*



Phát sinh C2 từ L1. Tìm support count cho các itemsets.
Đưa những itemset có support_count < min_sup *vào L2*

- Ví dụ: Cho CSDL D và số lượng hỗ trợ tối thiểu (min_sup) là 2. Tìm các tập ứng viên?

iv. Lặp lần 4:

- ***B1***:
  - Phát sinh ra tập ứng viên C_4 từ L_3 bằng cách kết hợp từng itemset trong L_3 với từng itemset có trong L_3 ($L_3 \triangleright\triangleleft L_3$): tập C4 rỗng do các tập itemset con của những itemset vừa phát sinh đều có cá tập itemset con đã bị cắt tỉa trước đó. Thuật toán Apriori kết thúc.

$\Rightarrow$ Kết quả thu được: $L_3 = \{ \{i1, i2, i3\}, \{i1, i2, i5\} \}$

2.1.2. Phát sinh luật kết hợp từ các tập phổ biến

- Các luật kết hợp mạnh cần thỏa mãn cả độ hỗ trợ tối thiểu và độ tin cậy tối thiểu bằng cách sử dụng công thức sau:

$$confidence(A \Rightarrow B) = P(B|A) = \frac{Support\_count(A \cup B)}{Support\_count(A)}$$

- Dựa trên điều này phương trình, luật kết hợp có thể được tạo ra như sau:
  - Với mỗi tập phổ biến Li, sinh ra tất cả các tập con khác rỗng (s_j) của L_i.
  - Với mọi tập con s_j khác rỗng của L_i, sẽ xuất ra luật: "s ⇒ (l −s)" nếu

$$\frac{\text{supportcount (l)}}{\text{supportcount (s)}} >= \text{min\_conf}$$

Trong đó, *min_conf* là ngưỡng tin cậy tối thiểu.

2.1.2. Phát sinh luật kết hợp từ các tập phổ biến

- Ví dụ: trở lại ví dụ trước với L_3= { {i1, i2, i3}, {i1, i2, i5} }. Lúc này giả sử min_conf được xác định là 70%:
  - FI_1= {I1, I2, I3}. Các tập con khác rỗng của X là {I1, I2}, {I1, I3}, {I2, I3}, {I1}, {I2} và {I3}.
  - FI_2= {I1, I2, I5}. Các tập con khác rỗng của X là {I1, I2}, {I1, I5}, {I2, I5}, {I1}, {I2} và {I5}.

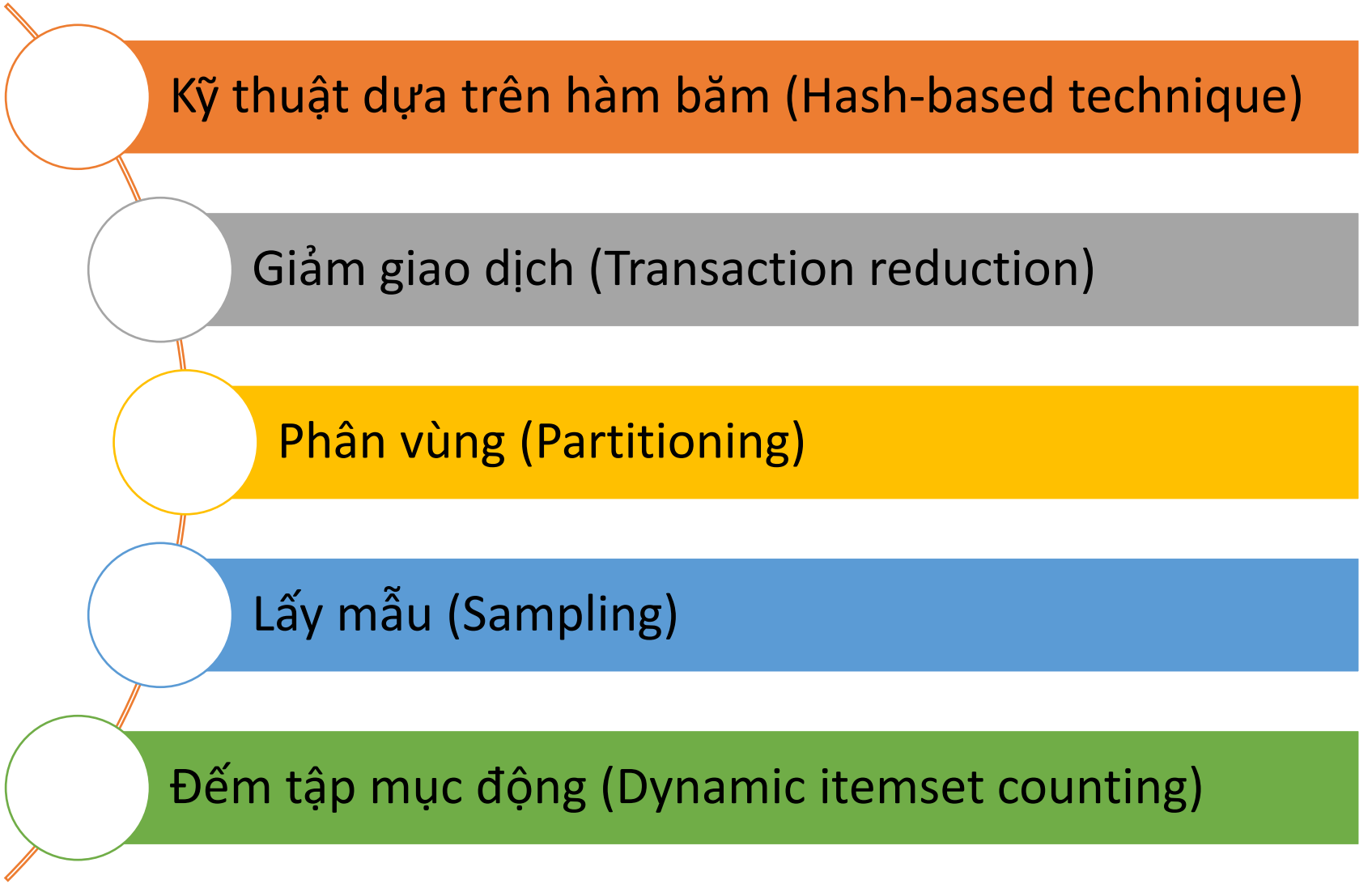
Các luật kết hợp thu được như được hiển thị trong bảng sau:

Frequent Itemset number	*Frequent itemset*	*Non Empty subsets of FI_i*	*Candidate of Association Rules*	*Confidence*	*Compare with min_conf*	*Association Rules*
FI_1	{I1, I2, I3}	{I1, I2}	{I1, I2}⇒{I3}	2/4= 50%	<70%	
		{I1, I3}	{I1, I3}⇒{I2}	2/4= 50%	<70%	
		{I2, I3}	{I2, I3}⇒{I1}	2/4= 50%	<70%	
		{I1}	{I1}⇒{I2, I3}	2/6= 33%	<70%	
		{I2}	{I2}⇒{I1, I3}	2/7= 43%	<70%	
		{I3}	{I3}⇒{I1, I2}	2/6= 33%	<70%	
FI_2	{I1, I2, I5}	{I1, I2}	{I1, I2}⇒{I5}	2/4 = 50%	<70%	
		{I1, i5}	{I1, i5}⇒{I2}	2/2 = 100%	>70%	AR1
		{I2, i5}	{I2, i5}⇒{I1}	2/2 = 100%	>70%	AR2
		{I1}	{I1}⇒{I2, I5}	2/6 = 33%	<70%	
		{I2}	{I2}⇒{I1, I5}	2/7 = 29%	<70%	
		{I5}	{I5}⇒{I1, I2}	2/2 = 100%	>70%	AR3

⇦

TID	*List of item IDs*
T1	I1, I2, I5
T2	I2, I4
T3	I2, I3
T4	I1, I2, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I1, I2, I3, I5
T9	I1, I2, I3

2.1.3. Cải thiện hiệu quả của Apriori



2.1.3. Cải thiện hiệu quả của Apriori

2.1.3.1. Kỹ thuật dựa trên hàm băm *(Hash-based technique)*

- Thực hiện băm (tức là ánh xạ) các tập mục vào các nhóm tương ứng.
- Kỹ thuật dựa trên hàm băm có thể được sử dụng để giảm kích thước của k-itemset ứng viên, C*k*, với k > 1. Ví dụ: khi quét từng giao dịch trong CSDL để tạo tập hợp 1 mục phổ biến, L_1, ta có thể tạo tất cả tập hợp gồm 2 item cho mỗi giao dịch, băm chúng vào các nhóm khác nhau của cấu trúc bảng băm và tăng số lượng nhóm tương ứng. Tập gồm 2 mục có số nhóm tương ứng trong bảng băm nằm dưới ngưỡng hỗ trợ phổ biến sẽ được loại bỏ khỏi tập ứng viên.
- Kỹ thuật dựa trên hàm băm như vậy có thể làm giảm đáng kể số lượng tập mục k ứng viên được kiểm tra (đặc biệt khi k = 2).

2.1.3.1. Kỹ thuật dựa trên hàm băm *(Hash-based technique)*

- Ví dụ: với tập dữ liệu như trong ví dụ trước, tạo bảng băm (hash table) H gồm 7 phần tử. Sử dụng hàm băm sau để đưa các tập 2-itemsets vào bảng băm H:

$$h(x, y) = ((order\ of\ x) * 10 + (order\ of\ y)) \bmod 7$$

- H(I1, I2) = ((1 * 10) + 2) mod 7 = 12 mod 7 = 5
- H(I1, I4) = ((1 * 10) + 4) mod 7 = 14 mod 7 = 0
- H(I3, I5) = ((3 * 10) + 5) mod 7 = 35 mod 7 = 0
- …

TID	List of item IDs
T1	I1, I2, I5
T2	I2, I4
T3	I2, I3
T4	I1, I2, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I1, I2, I3, I5
T9	I1, I2, I3

CSDL D

TID	I1	I2	I3	I4	I5
T1	1	1			1
T2		1		1	
T3		1	1		
T4	1	1		1	
T5	1		1		
T6		1	1		
T7	1		1		
T8	1	1	1		1
T9	1	1	1		

hash table H

bucket address	0	1	2	3	4	5	6
bucket count (=support count)	2	2	4	2	2	4	4
bucket contents	{I1, I4} {I3, I5}	{I1, I5} {I1, I5}	{I2, I3} {I2, I3} {I2, I3} {I2, I3}	{I2, I4} {I2, I4}	{I2, I5} {I2, I5}	{I1, I2} {I1, I2} {I1, I2} {I1, I2}	{I1, I3} {I1, I3} {I1, I3} {I1, I3}

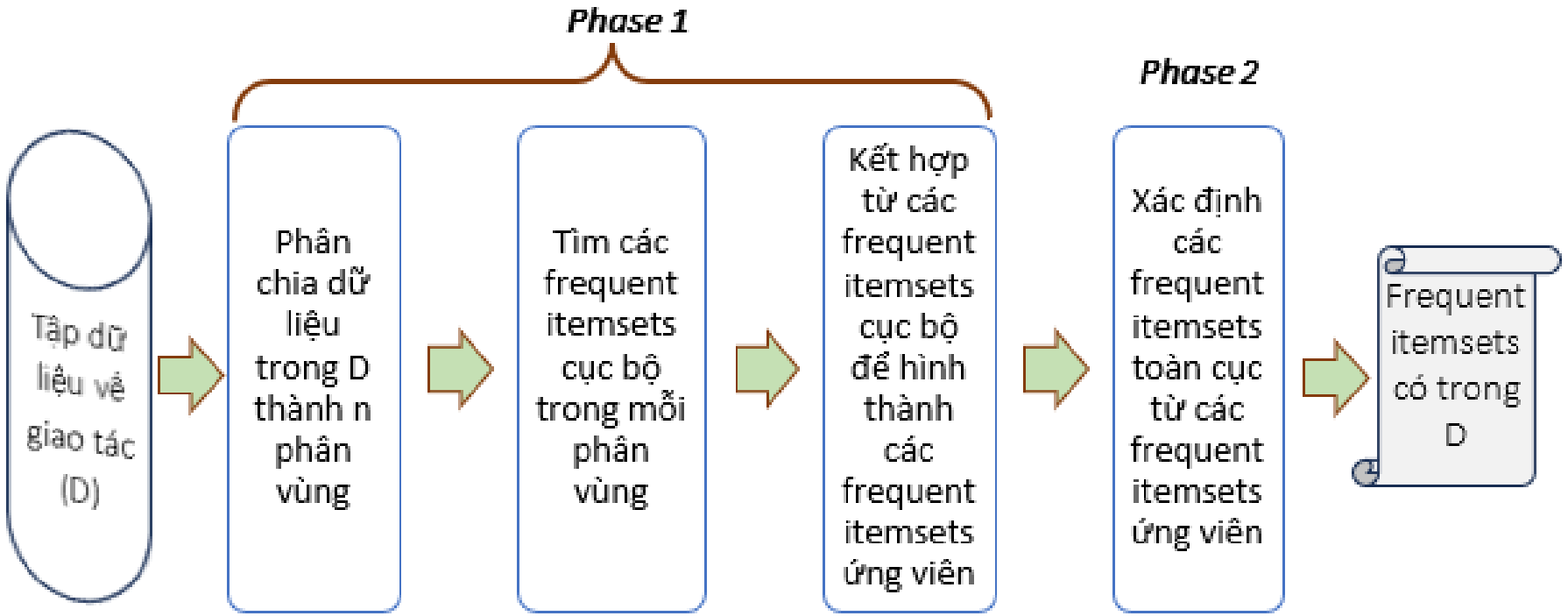
Bảng băm này được tạo bằng cách quét giao dịch trong khi xác định L_1. Nếu số lượng hỗ trợ tối thiểu (min_sup) là 3, thì các itemsets tại các vị trí **H[0]**, **H[1]**, **H[3]** và **H[4]** (có **support_count=2 < minsup=3**) không thể là tập phổ biến, do đó không được đưa vào C_2.

2.1.3.2. Giảm giao dịch *(Transaction reduction)*

- Là giảm số lượng giao dịch được quét trong các lần lặp sau. Một giao dịch không chứa bất kỳ tập mục k phổ biến (*frequent k-itemsets*) nào thì không thể chứa bất kỳ tập mục phổ biến (k +1) - (*frequent* (*k+1)-itemsets*) - nào khác . Do đó, một giao dịch như vậy có thể được đánh dấu hoặc loại bỏ khỏi việc xem xét thêm vì các lần quét CSDL tiếp theo để tìm các tập mục j, trong đó j > k, sẽ không cần phải xem xét giao dịch đó.

2.1.3.3. Phân vùng *(Partitioning)*

- Là phân vùng dữ liệu để tìm các tập mục ứng viên (*candidate itemsets*).
- Kích thước phân vùng và số lượng phân vùng được đặt sao cho mỗi phân vùng có thể vừa với bộ nhớ chính và do đó chỉ được đọc một lần trong mỗi giai đoạn.
- Nhờ vậy, chỉ cần hai lần quét CSDL để khai thác các tập mục phổ biến.



Khai thác bằng cách phân vùng dữ liệu

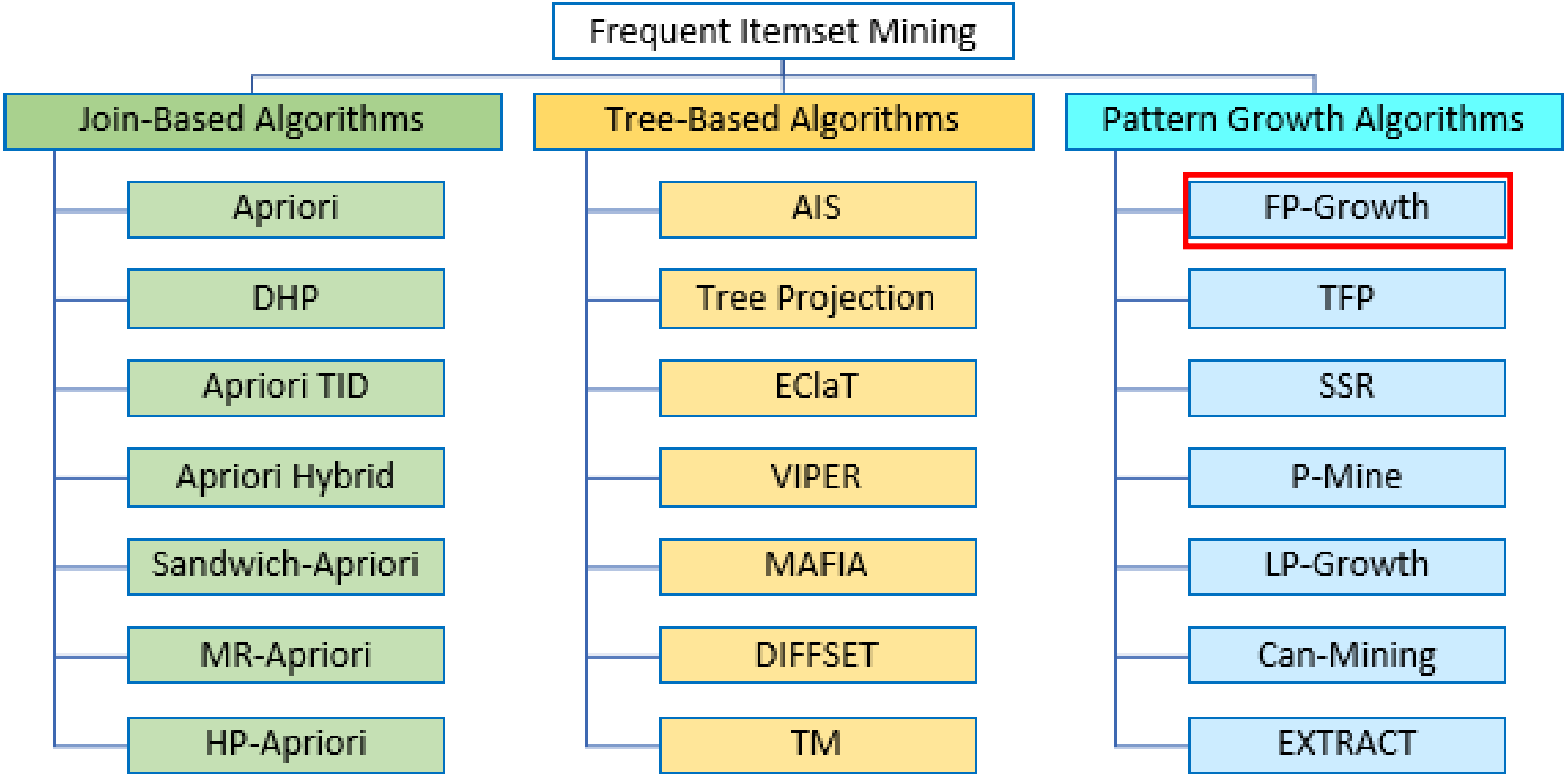
2.1.3.4. Lấy mẫu *(Sampling)*

- Là khai thác trên một tập hợp con của dữ liệu đã cho.
- Ý tưởng cơ bản của phương pháp lấy mẫu là chọn một mẫu S ngẫu nhiên của dữ liệu D đã cho, sau đó tìm kiếm các tập phổ biến trong S thay vì trong cả D.
- Ưu nhược điểm của phương pháp lấy mẫu:
  - Về tổng thể, cách này việc tìm kiếm các tập phổ biến trong S có thể được thực hiện trong bộ nhớ chính và do đó **chỉ cần quét một lần các giao dịch** trong S.
  - Do việc tìm kiếm các tập phổ biến thực hiện trong S chứ không phải trong D nên **có thể sẽ bỏ sót** một số **tập mục phổ biến toàn cục**.
    - Để giảm khả năng này, **cần sử dụng ngưỡng hỗ trợ thấp hơn mức hỗ trợ tối thiểu** để tìm các tập phổ biến cục bộ trong S (ký hiệu là L^S). Phần còn lại của CSDL sau đó được sử dụng để tính toán tần suất thực tế của từng tập mục trong L^S. Một cơ chế được sử dụng để xác định xem tất cả các tập phổ biến toàn cục có được đưa vào L^S hay không. Nếu L^S thực sự chứa tất cả các tập phổ biến trong D thì chỉ cần quét D một lần. Ngược lại, bước thứ hai có thể được thực hiện để tìm các tập phổ biến bị bỏ sót trong lần đầu tiên.
    - Theo cách này, độ chính xác của kết quả sẽ giảm nhưng hiệu quả sẽ tăng lên như trong các ứng dụng tính toán chuyên sâu phải chạy thường xuyên.

2.1.3.5. Đếm tập mục động *(Dynamic itemset counting)*

- Là thêm các tập mục ứng viên tại các điểm khác nhau trong quá trình quét.
- Phương pháp này được đề xuất trong đó CSDL được phân chia thành các khối được đánh dấu bằng điểm bắt đầu. Trong biến thể này, các tập mục ứng viên mới có thể được thêm vào bất kỳ điểm bắt đầu nào, không giống như trong Apriori, Apriori chỉ xác định các tập mục ứng viên mới ngay trước mỗi lần quét CSDL hoàn chỉnh. Kỹ thuật này sử dụng *count-so-far* làm giới hạn dưới của số lượng thực tế. Nếu *count-so-far* vượt qua mức hỗ trợ tối thiểu, tập mục này sẽ được thêm vào tập phổ biến và có thể được sử dụng để tạo ra các ứng viên dài hơn. Điều này dẫn đến việc quét CSDL ít hơn so với Apriori để tìm tất cả các tập phổ biến.

2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến *(A Pattern-Growth Approach for Mining Frequent Itemsets)*



2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến *(A Pattern-Growth Approach for Mining Frequent Itemsets)*

- Trong nhiều trường hợp, phương pháp tạo và kiểm tra ứng viên Apriori làm giảm đáng kể kích thước của các tập ứng viên, dẫn đến đạt được hiệu suất tốt. Tuy nhiên, nó có thể phải chịu hai chi phí không hề nhỏ:
  - Có thể vẫn cần tạo ra một số lượng lớn các tập ứng viên. Ví dụ: nếu có 10^4 tập 1-itemsets, thuật toán Apriori sẽ cần tạo ra hơn 10^7 tập 2-itemsets.
  - Có thể cần phải quét liên tục toàn bộ CSDL và kiểm tra một tập lớn các ứng viên bằng cách so khớp mẫu. Sẽ rất tốn kém khi xem xét từng giao dịch trong CSDL để xác định độ hỗ trợ của các tập mục ứng viên.
- Có thể thiết kế một phương pháp khai thác toàn bộ các tập phổ biến mà không cần quá trình tạo ứng viên tốn kém như vậy không? Trong nỗ lực này, phương pháp tăng trưởng mẫu phổ biến (hay tăng trưởng FP-growth) đã được đề xuất.

2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến *(A Pattern-Growth Approach for Mining Frequent Itemsets)*

- Phương pháp FP-growth được thực hiện như sau:

i. Nén CSDL biểu diễn các mục phổ biến (*frequent itemset*) thành cây mẫu phổ biến (*Frequent Pattern tree*) hoặc FP-tree, cây này giữ lại thông tin liên kết tập mục.

ii. Chia CSDL nén thành một tập hợp CSDL có điều kiện (một loại CSDL dự kiến đặc biệt), mỗi CSDL được liên kết với một mục phổ biến hoặc "đoạn mẫu" (*pattern fragment*) và khai thác từng CSDL riêng biệt. Đối với mỗi "đoạn mẫu", chỉ cần kiểm tra các tập dữ liệu liên quan của nó. Do đó, cách tiếp cận này có thể làm giảm đáng kể kích thước của các tập dữ liệu cần tìm kiếm, cùng với sự "tăng trưởng" của các mẫu đang được kiểm tra.

2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến *(A Pattern-Growth Approach for Mining Frequent Itemsets)*

- Ví dụ: sử dụng lại CSDL của ví dụ trước
  - ***B1**: Tìm các 1-itemsets > min_sup*

    Lần quét CSDL đầu tiên giống như Apriori, lấy ra tập hợp các mục phổ biến (1-itemset) và số lượng hỗ trợ của chúng (frequencies). Đặt số lượng hỗ trợ tối thiểu là 2. Tập hợp các mục phổ biến được sắp xếp theo thứ tự số lượng hỗ trợ giảm dần. Tập hợp hoặc danh sách kết quả này được ký hiệu là L.
  - ***B2**: Xây dựng FP-tree*
    - Đầu tiên, tạo gốc của cây, được gắn nhãn "null". Quét CSDL D lần thứ hai. Các mục trong mỗi giao dịch được xử lý theo thứ tự L (tức là được sắp xếp theo số lượng hỗ trợ giảm dần) và một nhánh được tạo cho mỗi giao dịch.

2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- Ví dụ: sử dụng lại CSDL của ví dụ trước

TID	List of item IDs
T1	I1, I2, I5
T2	I2, I4
T3	I2, I3
T4	I1, I2, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I1, I2, I3, I5
T9	I1, I2, I3

Sort Frequent itemsets by itemsets name

Itemset	Support_ count
$\{i_1\}$	6
$\{i_2\}$	7
$\{i_3\}$	6
$\{i_4\}$	2
$\{i_5\}$	2

Sort Frequent itemsets by support_count (L)

Itemset	Support_ count
$\{i_2\}$	7
$\{i_1\}$	6
$\{i_3\}$	6
$\{i_4\}$	2
$\{i_5\}$	2

TID	List of item IDs
T1	I2, I1, I5
T2	I2, I4
T3	I2, I3
T4	I2, I1, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I2, I1, I3, I5
T9	I2, I1, I3

- ***B2**: Xây dựng FP-tree*
  - Đầu tiên, tạo gốc của cây, được gắn nhãn "null".

2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

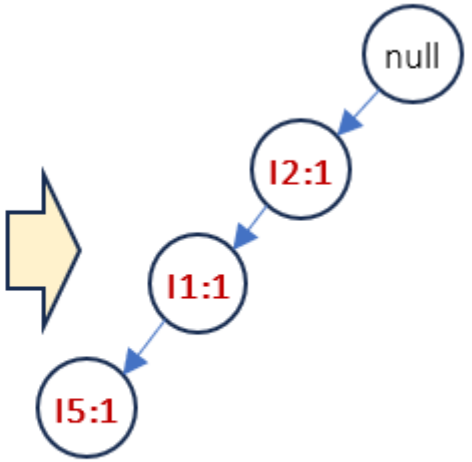
- Ví dụ: sử dụng lại CSDL của ví dụ trước
  - ***B2**: Xây dựng FP-tree*

    Quét CSDL D lần thứ hai. Các mục trong mỗi giao dịch được xử lý theo thứ tự L (tức là được sắp xếp theo số lượng hỗ trợ giảm dần) và một nhánh được tạo cho mỗi giao dịch.
    - Quét giao dịch đầu tiên, "T1: I_1, I_2, I_5" chứa ba mục (I_2, I_1, I_5 theo thứ tự L), dẫn đến việc xây dựng nhánh đầu tiên của cây có ba nút, hI_2: 1, hI_1: 1 và hI_5: 1, trong đó I_2 được liên kết với gốc, I_1 được liên kết với I_2 và I_5 được liên kết với I_1.

TID	List of item IDs
T1	I2, I1, I5
T2	I2, I4
T3	I2, I3
T4	I2, I1, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I2, I1, I3, I5
T9	I2, I1, I3

Sort Frequent itemsets by support_count

Itemset	Support_ count
{i_2}	7
{i_1}	6
{i_3}	6
{i_4}	2
{i_5}	2



FP-tree sau khi quét T1

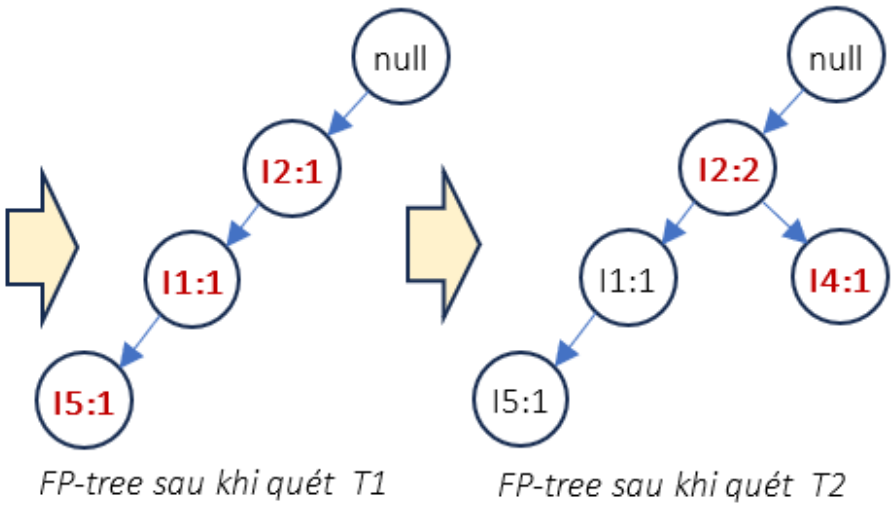
2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- Ví dụ: sử dụng lại CSDL của ví dụ trước
  - ***B2**: Xây dựng FP-tree*
    - Giao dịch thứ hai, T2, chứa các mục I_2 và I_4 theo thứ tự L: do T1 và T2 cùng bắt đầu bằng I_2, nên 2 giao tác này sẽ sử dụng chung tiền tố từ gốc đến I_2. Từ I_2 sẽ tạo ra thêm một nhánh để liên kết I_2 với I_4 sẽ được thêm vào (do đó I_4 sẽ có số lượng là 1). Do 2 giao tác có dùng chung tiền tố I_2, do đó tăng số lượng tại nút I_2 lên thêm 1 để trở thành **I2:2**.

TID	*List of item IDs*
T1	I2, I1, I5
T2	I2, I4
T3	I2, I3
T4	I2, I1, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I2, I1, I3, I5
T9	I2, I1, I3

Sort Frequent itemsets by support_count

Itemset	*Support_ count*
$\{i_2\}$	7
$\{i_1\}$	6
$\{i_3\}$	6
$\{i_4\}$	2
$\{i_5\}$	2



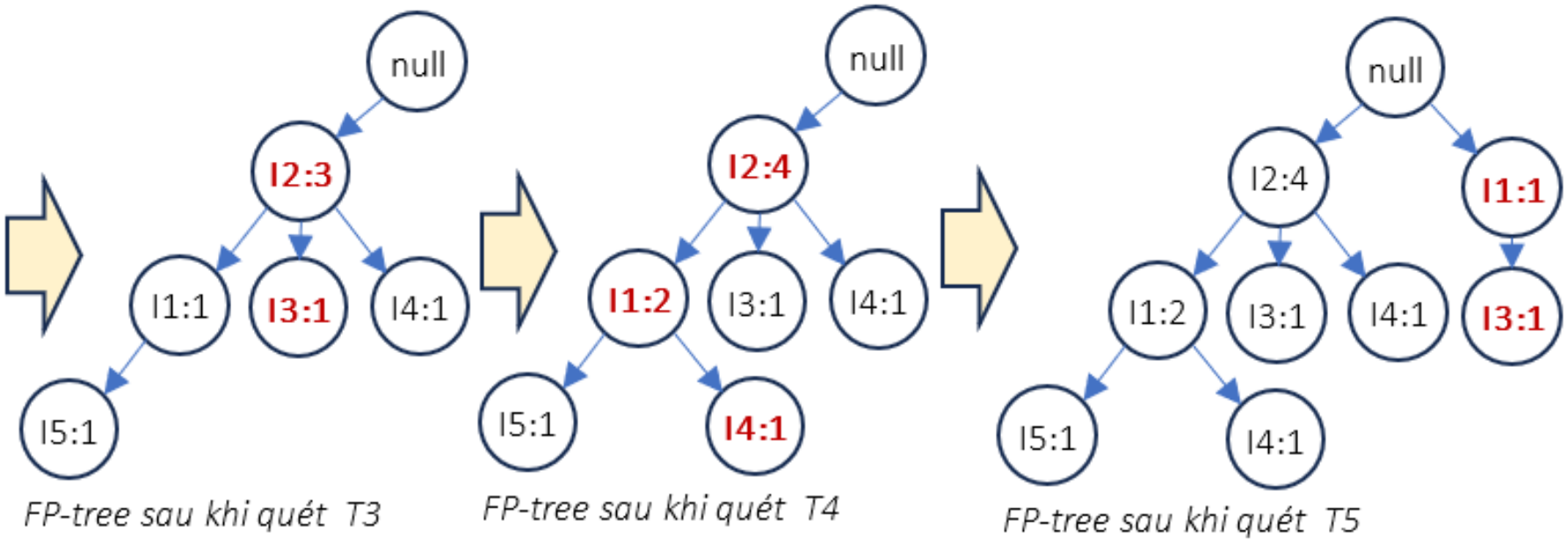
FP-tree sau khi quét T1 *FP-tree sau khi quét T2*

2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- Ví dụ: sử dụng lại CSDL của ví dụ trước
  - ***B2**: Xây dựng FP-tree*

Nói chung, khi xem xét nhánh sẽ được thêm vào cho một giao dịch, số lượng mỗi nút dọc theo một tiền tố chung sẽ tăng thêm 1 và các nút cho các mục theo sau tiền tố đó sẽ được tạo và liên kết tương ứng.

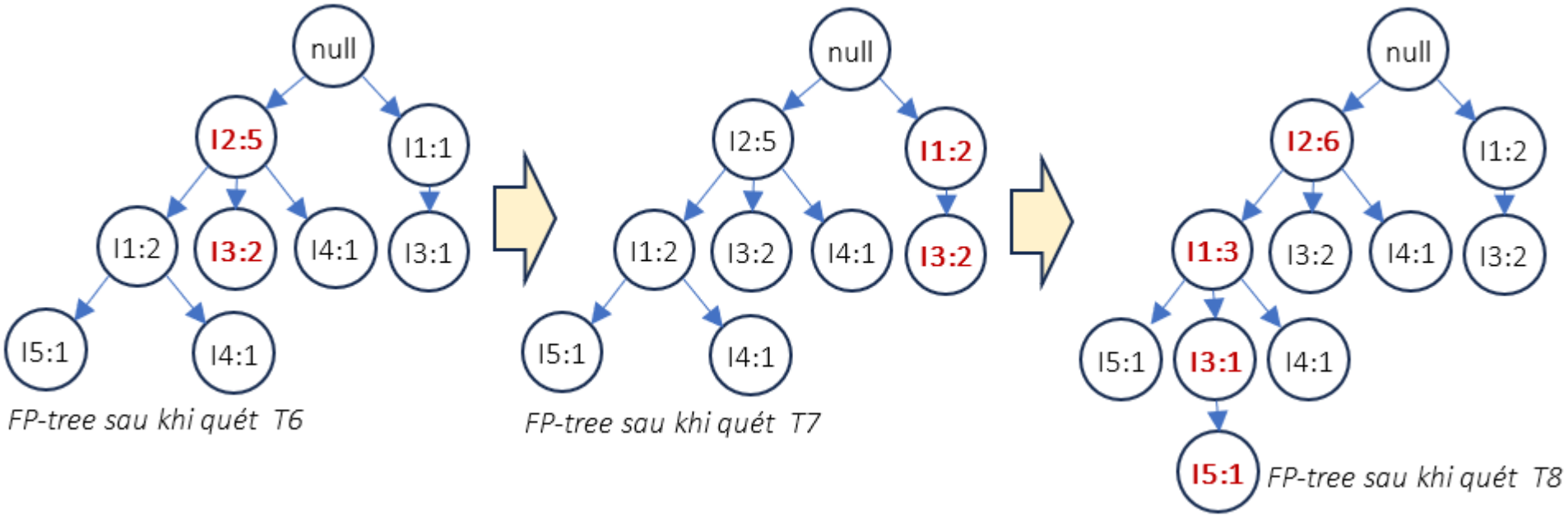
TID	*List of frequent item*
T1	I2, I1, I5
T2	I2, I4
T3	I2, I3
T4	I2, I1, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I2, I1, I3, I5
T9	I2, I1, I3



2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- Ví dụ: sử dụng lại CSDL của ví dụ trước
  - ***B2**: Xây dựng FP-tree*

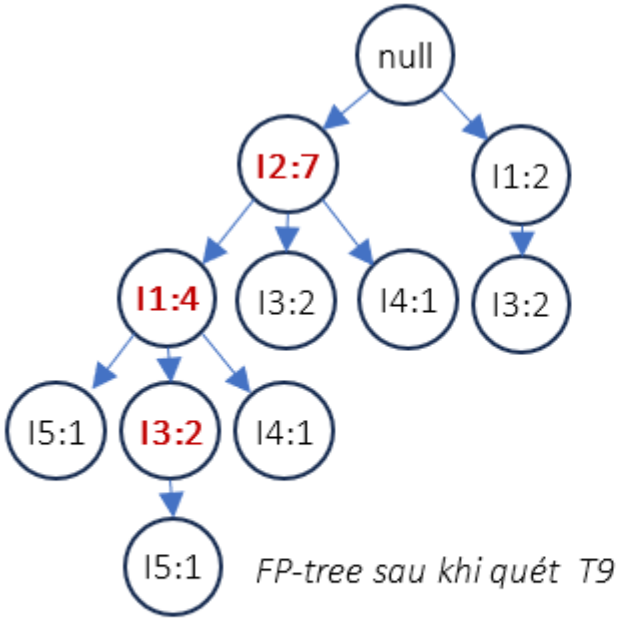
TID	*List of frequent item*
T1	I2, I1, I5
T2	I2, I4
T3	I2, I3
T4	I2, I1, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I2, I1, I3, I5
T9	I2, I1, I3



2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- Ví dụ: sử dụng lại CSDL của ví dụ trước
  - ***B2**: Xây dựng FP-tree*

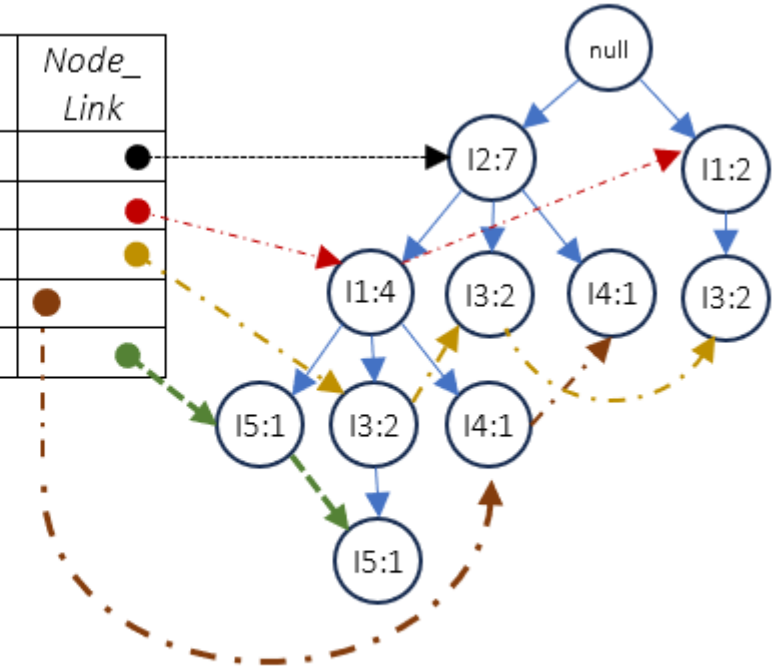
TID	*List of frequent item*
T1	I2, I1, I5
T2	I2, I4
T3	I2, I3
T4	I2, I1, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I2, I1, I3, I5
T9	I2, I1, I3



FP-tree sau khi quét T9

Itemset	*Support_ count*	*Node_ Link*
$\{i_2\}$	7	
$\{i_1\}$	6	
$\{i_3\}$	6	
$\{i_4\}$	2	
$\{i_5\}$	2	

Item header table



Một cây FP đăng ký thông tin mẫu thường xuyên, nén

- Để tạo thuận lợi cho việc duyệt cây, một bảng tiêu đề các items (*item header table*) được xây dựng sao cho mỗi item trỏ đến các lần xuất hiện của nó trong cây thông qua một chuỗi các nút liên kết.
- Bằng cách này, vấn đề khai thác các mẫu phổ biến trong CSDL được chuyển thành vấn đề khai thác cây FP.

- ***B3**: Khai thác FP-tree*

  Bắt đầu từ mỗi frequent itemsets có frequent itemset=1 (*1-itemsets*), duyệt các item phổ biến mức 1 theo thứ tự tăng dần của độ hỗ trợ là I5, I4, I3, I1, I2. Với mỗi item, xây dựng các **cơ sở mẫu điều kiện** *(conditional pattern-base)* và sau đó là các **FP-Tree điều kiện** *(conditional FP-Tree)* của nó.

  - *Xét* I_5: cơ sở mẫu điều kiện của nó là tất cả các đường dẫn tiền tố của cây FP-Tree khi duyệt từ gốc root = null đến nút I_5, chính là $\{I_2, I_1, I_5: 1\}$ và $\{I_2, I_1, I_3, I_5: 1\}$. Với số theo sau là số lần xuất hiện của nút I_5 tương ứng với mỗi tiền tố đó). Các đường dẫn được hình thành bởi các nhánh này là $\{I_2, I_1, I_5: 1\}$ và $\{I_2, I_1, I_3, I_5: 1\}$. Do đó, do coi I_5 là hậu tố thì hai đường dẫn tiền tố tương ứng của nó là $\{I_2, I_1: 1\}$ và $\{I_2, I_1, I_3: 1\}$, tạo thành cơ sở mẫu điều kiện của nó.

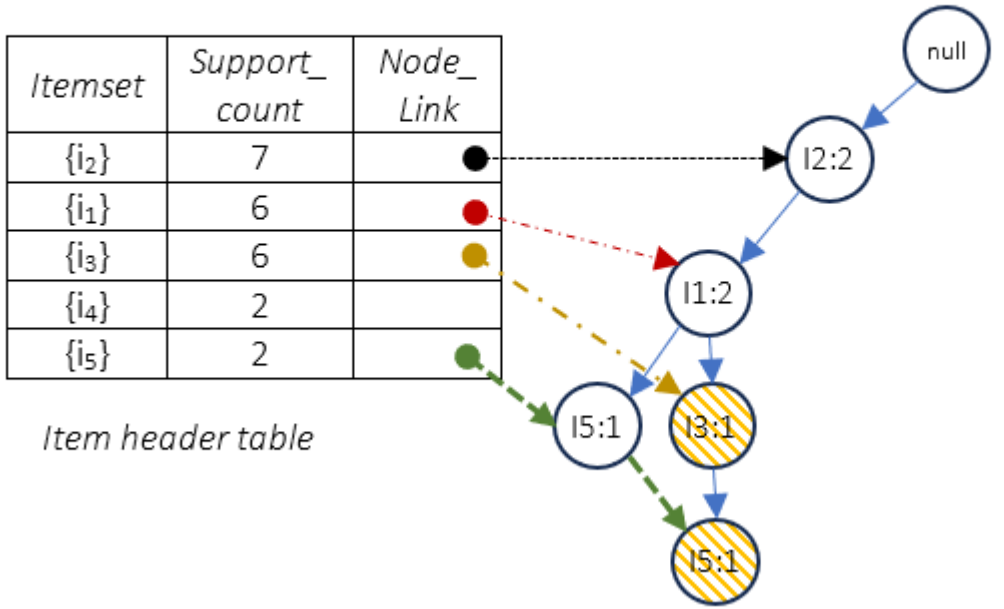
2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- ***B3**: Khai thác FP-tree*
  - *Xét I_5*:

Trộn (đếm số lần xuất hiện của mỗi nút trong tất cả các đường dẫn), ta được: I_2=2; I_1=2; I_3=1. Do số lượng hỗ trợ của I_3 là 1 nhỏ hơn số lượng hỗ trợ tối thiểu (=2) nên loại I_3.

Như vậy FP-tree có điều kiện của I_5 chỉ gồm 1 đường dẫn duy nhất là {I_2: 2, I_1: 2}. Từ đó ta được đường dẫn duy nhất tạo ra tất cả các kết hợp của các mẫu phổ biến: {I_2, I_5: 2}; {I_1, I_5: 2}; {I_2, I_1, I_5: 2}

Item	*Conditional Pattern Base*	*Conditional FP-tree*	*Frequent Patterns Generated*
I_5	{{I_2, I_1: 1}; {I_2, I_1, I_3: 1}}	I_2: 2; I_1: 2	{I_2, I_5: 2}; {I_1, I_5: 2}; {I_2, I_1, I_5: 2}

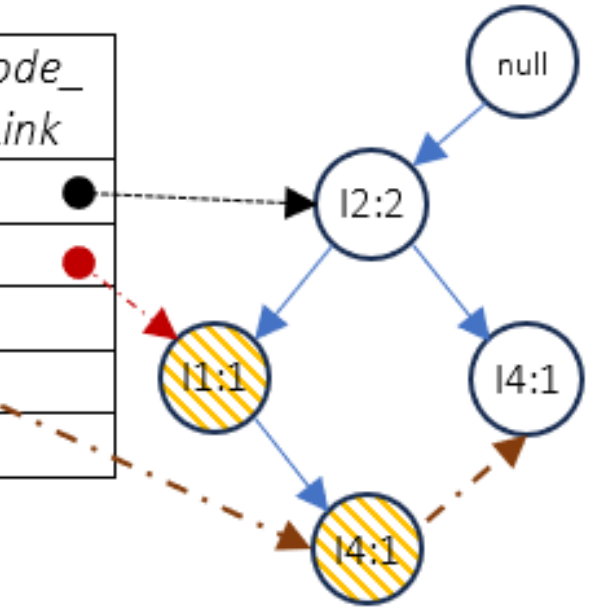


2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- ***B3**: Khai thác FP-tree*
  - *Xét I_4*: hai đường dẫn tiền tố của nó tạo thành cơ sở mẫu có điều kiện, {{I_2, I_1: 1}; {I_2: 1}}, tạo ra một cây FP có điều kiện một nút, {I_2: 2} và dẫn xuất một mẫu phổ biến, {I_2, I_4: 2}.

Itemset	*Support_ count*	*Node_ Link*
{i_2}	7	
{i_1}	6	
{i_3}	6	
{i_4}	2	
{i_5}	2	

Item header table



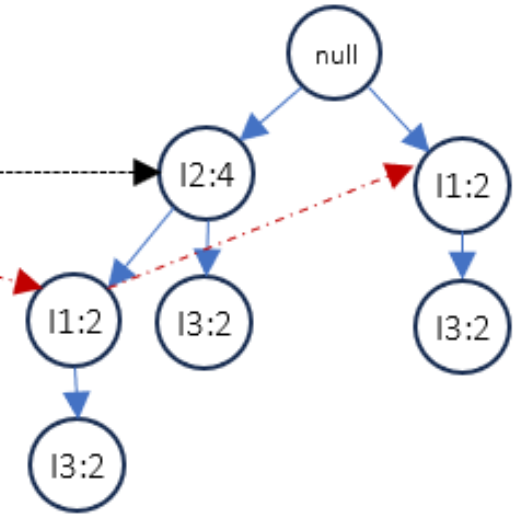
Item	*Conditional Pattern Base*	*Conditional FP-tree*	*Frequent Patterns Generated*
I5	{{I2, I1: 1}; {I2, I1, I3: 1}}	I2: 2; I1: 2	{I2, I5: 2}; {I1, I5: 2}; {I2, I1, I5: 2}
I4	{{I2, I1: 1}; {I2: 1}}	I2: 2	{I2, I4: 2}

2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- ***B3**: Khai thác FP-tree*
  - *Xét I_3*: ba đường dẫn tiền tố của I3 tạo thành cơ sở mẫu có điều kiện: {{I_1, I_3: 2}; {I_2, I_3: 2}; {I_2, I_1, I_3: 2}}. Như vậy cây FP có điều kiện của I_3 gồm các nhánh: {{I_1:4};{I_2:4};{I_2,I1:2}} và dẫn xuất ra các mẫu phổ biến là: { {I_2, I_3: 4}, {I_1, I_3: 4}, {I_2, I_1, I_3: 2}}.

Itemset	*Support_ count*	*Node_ Link*
{i_2}	7	●
{i_1}	6	●

Item header table



Cây FP có điều kiện được liên kết với nút có điều kiện I3

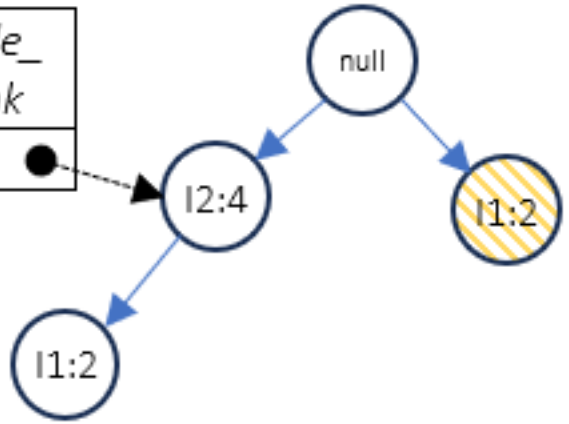
Item	*Conditional Pattern Base*	*Conditional FP-tree*	*Frequent Patterns Generated*
I5	{{I2, I1: 1}; {I2, I1, I3: 1}}	I2: 2; I1: 2	{I2, I5: 2}; {I1, I5: 2}; {I2, I1, I5: 2}
I4	{{I2, I1: 1}; {I2: 1}}	I2: 2	{I2, I4: 2}
I3	{{I2, I1: 2}; {I2: 2}; {I1: 2}}	I2: 4; I1: 4; I2, I1: 2	{I2, I3: 4}, {I1, I3: 4}, {I2, I1, I3: 2}

2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- ***B3**: Khai thác FP-tree*
  - *Xét I_1*: hai đường dẫn tiền tố của I_1 tạo thành cơ sở mẫu có điều kiện: {{I_2, I_1: 4}; {I_1:∅}}. Như vậy cây FP có điều kiện của I_1 chỉ gồm 1 nhánh: {I_2 : 4} và dẫn xuất ra các mẫu phổ biến là: { {I_2, I_1: 4}}.

Itemset	*Support_ count*	*Node_ Link*
{I_2}	7	

Item header table



Item	*Conditional Pattern Base*	*Conditional FP-tree*	*Frequent Patterns Generated*
I5	{{I2, I1: 1}; {I2, I1, I3: 1}}	I2: 2; I1: 2	{I2, I5: 2}; {I1, I5: 2}; {I2, I1, I5: 2}
I4	{{I2, I1: 1}; {I2: 1}}	I2: 2	{I2, I4: 2}
I3	{{I2, I1: 2}; {I2: 2}; {I1: 2}}	I2: 4; I1: 4; I2, I1: 2	{I2, I3: 4}; {I1, I3: 4}; {I2, I1, I3: 2}
I1	{{I2: 4}}	hI2: 4	{I2, I1: 4}

2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- ***B3**: Khai thác FP-tree*
  - *Xét I_2*: không có đường dẫn tiền tố của I2, do đó cơ sở mẫu có điều kiện của I2 là: {I2:∅}. Như vậy cây FP có điều kiện của I2 là rỗng nên cũng không có dẫn xuất ra các mẫu phổ biến.

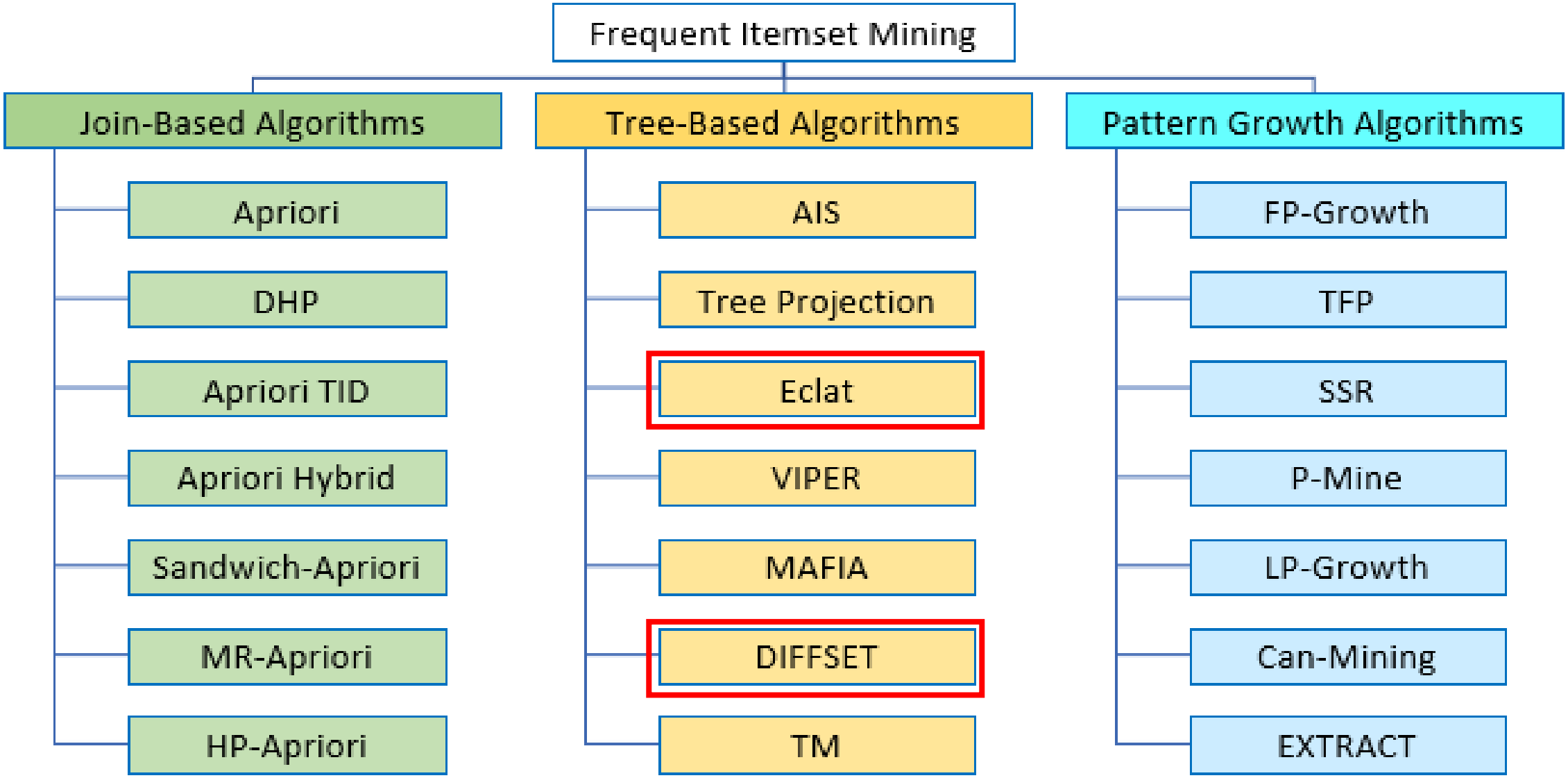
Tóm lại, việc khai thác cây FP được tóm tắt trong bảng sau:

Item	*Conditional Pattern Base*	*Conditional FP-tree*	*Frequent Patterns Generated*
I5	{{I2, I1: 1}; {I2, I1, I3: 1}}	I2: 2; I1: 2	{I2, I5: 2}; {I1, I5: 2}; {I2, I1, I5: 2}
I4	{{I2, I1: 1}; {I2: 1}}	I2: 2	{I2, I4: 2}
I3	{{I2, I1: 2}; {I2: 2}; {I1: 2}}	I2: 4; I1: 4; I2, I1: 2	{I2, I3: 4}; {I1, I3: 4}; {I2, I1, I3: 2}
I1	{{I2: 4}}	hI2: 4	{I2, I1: 4}

2.2. Phương pháp tiếp cận tăng trưởng theo mẫu để khai thác các tập mục phổ biến

- ***Đặc điểm của phương pháp FP-growth***:
  - Phương pháp FP-growth biến đổi vấn đề tìm kiếm các mẫu phổ biến dài thành tìm kiếm các mẫu ngắn hơn trong CSDL có điều kiện nhỏ hơn nhiều theo cách đệ quy và sau đó ghép nối hậu tố.
  - FP-growth sử dụng các mục ít phổ biến nhất làm hậu tố, mang lại khả năng chọn lọc tốt. Phương pháp này làm giảm đáng kể chi phí tìm kiếm.
  - Hiệu suất của phương pháp tăng trưởng FP cho thấy rằng nó hiệu quả và có thể mở rộng để khai thác cả các mẫu phổ biến dài và ngắn và nhanh hơn thuật toán Apriori khoảng một bậc.
  - Khi CSDL lớn, việc xây dựng cây FP dựa trên bộ nhớ chính đôi khi là không thực tế. Một giải pháp thay thế khác là trước tiên hãy phân vùng CSDL thành một tập hợp các CSDL dự kiến, sau đó xây dựng cây FP và khai thác nó trong mỗi CSDL dự kiến. Quá trình này có thể được áp dụng đệ quy cho bất kỳ CSDL dự kiến nào nếu cây FP của nó vẫn không thể vừa với bộ nhớ chính.

2.3. Khai thác các tập phổ biến bằng cách sử dụng định dạng dữ liệu dọc *(Vertical Data Format)*



2.3. Khai thác các tập phổ biến bằng cách sử dụng định dạng dữ liệu dọc *(Vertical Data Format)*

- Cả hai phương pháp tăng trưởng *Apriori* và *FP* đều khai thác các mẫu phổ biến từ một tập hợp các giao dịch ở định dạng dữ liệu ngang (*horizontal data format* - *TID-itemset*):

  { TID : itemset}

  trong đó
  - TID là ID giao dịch
  - itemset là tập hợp các mục được mua trong TID giao dịch.

- Ngoài ra, dữ liệu có thể được trình bày ở định dạng dữ liệu dọc (vốn là bản chất của thuật toán Eclat - Chuyển đổi lớp tương đương) :

  { item : TID_set }

  trong đó:
  - item là tên mặt hàng.
  - TID_set: là tập hợp các mã định danh giao dịch có chứa mặt hàng đó.

Định dạng dữ liệu theo chiều ngang của CSDL D

TID	List of item IDs
T1	I1, I2, I5
T2	I2, I4
T3	I2, I3
T4	I1, I2, I4
T5	I1, I3
T6	I2, I3
T7	I1, I3
T8	I1, I2, I3, I5
T9	I1, I2, I3

Định dạng dữ liệu theo chiều dọc của CSDL D

Itemset	TID_set
I1	{T1, T4, T5, T7, T8, T9}
I2	{T1, T2, T3, T4, T6, T8, T9}
I3	{T3, T5, T6, T7, T8, T9}
I4	{T2, T4}
I5	{T1, T8}

2.3. Khai thác các tập phổ biến bằng cách sử dụng định dạng dữ liệu dọc *(Vertical Data Format)*

- Ví dụ: vẫn lấy dữ liệu từ ví dụ trước đây.
  - Kết hợp 5 item set trong bảng, thu được 10 tập hợp, trong đó 2 tổ hợp $\{I_3, I_4\}$ và $\{I_4, I_5\}$ là rỗng (do không có chung giao tác). Do đó còn 8 tập hợp 2-itemset không rỗng.

Định dạng dữ liệu theo chiều dọc của CSDL D

Itemset	TID_set
I1	{T1, T4, T5, T7, T8, T9}
I2	{T1, T2, T3, T4, T6, T8, T9}
I3	{T3, T5, T6, T7, T8, T9}
I4	{T2, T4}
I5	{T1, T8}

2-itemsets theo định dạng dữ liệu theo chiều dọc

Itemset	TID_set	
{I1, I2}	{T1, T4, T8, T9}	
{I1, I3}	{T5, T7, T8, T9}	
{I1, I4}	{T4}	*Loại do <min_sup*
{I1, I5}	{T1, T8}	
{I2, I3}	{T3, T6, T8, T9}	
{I2, I4}	{T2, T4}	
{I2, I5}	{T1, T8}	
{I3, I5}	{T8}	*Loại do <min_sup*

- Dựa trên thuộc tính Apriori, một tập *3-itemsets* được xem là ứng viên nếu mỗi tập con trong số *2-itemsets* là phổ biến. Bằng cách thực hiện phép giao giữa các TID_set của hai tập *2-itemsets* tương ứng bất kỳ trong các tập 3 mục ứng viên này. Quá trình tạo ứng viên ở đây sẽ chỉ tạo ra được hai tập *3-itemsets*: $\{I_1, I_2, I_3\}$ và $\{I_1, I_2, I_5\}$ do có độ hỗ trợ >= *min_sup* (>=2).

3-itemsets theo định dạng dữ liệu theo chiều dọc

itemset	TID set
{I1, I2, I3}	{T8, T9}
{I1, I2, I5}	{T1, T8}

2.3. Khai thác các tập phổ biến bằng cách sử dụng định dạng dữ liệu dọc *(Vertical Data Format)*

- Ví dụ: vẫn lấy dữ liệu từ ví dụ trước đây.
  - Kết hợp 5 item set trong bảng, thu được 10 tập hợp, trong đó 2 tổ hợp $\{I_3, I_4\}$ và $\{I_4, I_5\}$ là rỗng (do không có chung giao tác). Do đó còn 8 tập hợp 2-itemset không rỗng.

Định dạng dữ liệu theo chiều dọc của CSDL D

Itemset	TID_set
I1	{T1, T4, T5, T7, T8, T9}
I2	{T1, T2, T3, T4, T6, T8, T9}
I3	{T3, T5, T6, T7, T8, T9}
I4	{T2, T4}
I5	{T1, T8}

2-itemsets theo định dạng dữ liệu theo chiều dọc

Itemset	TID_set	
{I1, I2}	{T1, T4, T8, T9}	
{I1, I3}	{T5, T7, T8, T9}	
{I1, I4}	{T4}	*Loại do <min_sup*
{I1, I5}	{T1, T8}	
{I2, I3}	{T3, T6, T8, T9}	
{I2, I4}	{T2, T4}	
{I2, I5}	{T1, T8}	
{I3, I5}	{T8}	*Loại do <min_sup*

- Dựa trên thuộc tính Apriori, một tập *3-itemsets* được xem là ứng viên nếu mỗi tập con trong số *2-itemsets* là phổ biến. Bằng cách thực hiện phép giao giữa các TID_set của hai tập *2-itemsets* tương ứng bất kỳ trong các tập 3 mục ứng viên này. Quá trình tạo ứng viên ở đây sẽ chỉ tạo ra được hai tập *3-itemsets*: $\{I_1, I_2, I_3\}$ và $\{I_1, I_2, I_5\}$ do có độ hỗ trợ >= *min_sup* (>=2).

3-itemsets theo định dạng dữ liệu theo chiều dọc

itemset	TID set
{I1, I2, I3}	{T8, T9}
{I1, I2, I5}	{T1, T8}

2.3. Khai thác các tập phổ biến bằng cách sử dụng định dạng dữ liệu dọc *(Vertical Data Format)*

- Đặc điểm của phương pháp:
  - **Tận dụng được tính chất Apriori** trong việc tạo tập mục ứng cử viên (k+1) từ tập k mục phổ biến.
  - **Không cần quét CSDL để tìm độ hỗ trợ của (k +1)-itemset** (với $k \geq 1$).
  - Tuy nhiên, các ***TID_set* có thể khá dài, chiếm nhiều dung lượng bộ nhớ** cũng như thời gian tính toán để thực hiện phép giao giữa các *TID_set* dài này.

    Để giảm hơn nữa chi phí đăng ký các *TID_set* dài, cũng như chi phí tiếp theo của việc thực hiện phép giao giữa các tập này, có thể sử dụng một kỹ thuật gọi là diffset, kỹ thuật này chỉ theo dõi sự khác biệt của các *TID_set* của tập mục (k+1) và một k-itemsets tương ứng. Chẳng hạn, trong ví dụ, có $\{I_1\} = \{T_1, T_4, T_5, T_7, T_8, T_9\}$ và $\{I_1, I_2\} = \{T_1, T_4, T_8, T_9\}$. Độ chênh lệch giữa hai giá trị này là diffset($\{I_1, I_2\}, \{I_1\}$) = $\{T_5, T_7\}$. Do đó, thay vì ghi lại bốn *TID* được tạo thành từ việc giao giữa$\{I_1\}$ và $\{I_2\}$, có thể sử dụng diffset để chỉ ghi lại hai *TID*, biểu thị sự khác biệt giữa $\{I_1\}$ và $\{I_1, I_2\}$.

2.4. Khai thác các mẫu đóng và mẫu tối đại *(Mining Closed and Max Patterns)*

- Trong các phần trước cho thấy việc khai thác tập mục phổ biến:
  - Có thể tạo ra một số lượng lớn các tập hợp, đặc biệt khi ngưỡng tối thiểu được đặt ở mức thấp hoặc khi tồn tại các mẫu dài trong tập dữ liệu.
  - Việc khai thác các tập mục phổ biến đóng (*closed frequent itemsets*) có thể giảm đáng kể số lượng mẫu được tạo ra trong quá trình khai thác tập mục phổ biến trong khi vẫn bảo toàn thông tin đầy đủ về tập các tập mục phổ biến. Nghĩa là, từ tập các tập phổ biến đóng, có thể dễ dàng suy ra tập các tập phổ biến và độ hỗ trợ của chúng.

  $\Rightarrow$ Vì vậy, trong thực tế, việc khai thác tập các tập phổ biến đóng tốt hơn là tập hợp tất cả các tập phổ biến trong hầu hết các trường hợp.
- Một phương pháp được đề xuất là tìm kiếm các tập phổ biến đóng trực tiếp trong quá trình khai thác. Điều này đòi hỏi phải tinh gọn (cắt tỉa) không gian tìm kiếm ngay khi có thể xác định được trường hợp tập mục đóng trong quá trình khai thác.

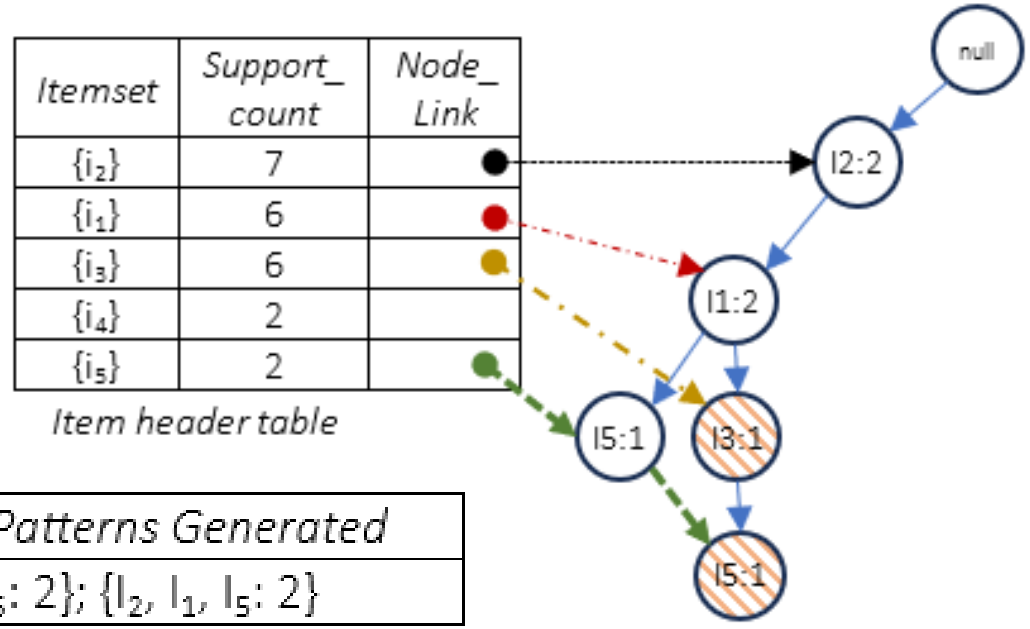
2.4. Khai thác các mẫu đóng và mẫu tối đại *(Mining Closed and Max Patterns)*

2.4.1. Chiến lược cắt tỉa: bao gồm

i. Sáp nhập mục(*Itemmerging*): Nếu mọi giao dịch chứa tập mục phổ biến X cũng chứa tập mục Y nhưng không có tập siêu thực sự nào của Y thì X∪Y tạo thành tập mục phổ biến đóng và không cần tìm kiếm bất kỳ tập mục nào chứa X nhưng không có Y.

Ví dụ: lấy lại ví dụ trong phần FP-tree, CSDL có điều kiện dự kiến cho tập mục tiền tố $\{I_5{:}2\}$ là $\{\{I_2, I_1\}, \{I_2, I_1, I_3\}\}$, từ đó ta có thể thấy rằng mỗi giao dịch của nó chứa tập mục $\{I_2, I_1\}$ nhưng không có tập cha nào phù hợp của $\{I_2, I_1\}$.

Tập mục $\{I_2, I_1\}$ có thể được hợp nhất với $\{I_5\}$ để tạo thành tập mục đóng, $\{I_5, I_2, I_1{:}\ 2\}$ và ta không cần khai thác các tập mục đóng có chứa I_5 nhưng không chứa $\{I_2, I_1\}$.



Item	Conditional Pattern Base	Conditional FP-tree	Frequent Patterns Generated
I_5	$\{\{I_2, I_1{:}\ 1\}; \{I_2, I_1, I_3{:}\ 1\}\}$	$I_2{:}\ 2; I_1{:}\ 2$	$\{I_2, I_5{:}\ 2\}; \{I_1, I_5{:}\ 2\}; \{I_2, I_1, I_5{:}\ 2\}$

2.4. Khai thác các mẫu đóng và mẫu tối đại *(Mining Closed and Max Patterns)*

2.4.1. Chiến lược cắt tỉa: bao gồm

ii. ***Cắt bớt tập mục con*** (*Sub-itemset pruning*): Nếu tập mục phổ biến X là tập con đúng của tập mục đóng phổ biến Y đã được tìm thấy và số hỗ trợ(X) = số hỗ trợ(Y), thì X và tất cả các con cháu của X trong cây liệt kê tập hợp không thể có trong các tập mục đóng, do đó có thể được cắt bớt.

Giả sử CSDL giao dịch chỉ có hai giao dịch: $\{a_1, a_2, ..., a_{100}\}$ và $\{a_1, a_2, ..., a_{50}\}$ và số lượng hỗ trợ tối thiểu là *min_sup = 2*. Phép chiếu trên mục đầu tiên, a_1, dẫn xuất tập mục phổ biến, $\{a_1, a_2, ..., a_{50} : 2\}$, dựa trên tối ưu hóa việc hợp nhất tập mục. Vì $\text{support}(\{a_2\}) = \text{support}(\{a_1, a_2, ..., a_{50}\}) = 2$ và $\{a_2\}$ là tập con đúng của $\{a_1, a_2, ..., a_{50}\}$ nên không cần thiết để kiểm tra a_2 và CSDL dự kiến của nó. Việc cắt tỉa tương tự cũng có thể được thực hiện cho $a_3, ..., a_{50}$. Do đó, việc khai thác các tập phổ biến đóng trong tập dữ liệu này sẽ kết thúc sau khi khai thác CSDL dự kiến của a_1.

2.4. Khai thác các mẫu đóng và mẫu tối đại *(Mining Closed and Max Patterns)*

2.4.1. Chiến lược cắt tỉa: bao gồm

iii. Bỏ qua mục (*Item skipping*): Trong khai thác sâu các tập mục đóng, ở mỗi cấp độ sẽ có một tập mục tiền tố X được liên kết với bảng tiêu đề và cơ sở dữ liệu dự kiến. Nếu một mục phổ biến cục bộ p có cùng độ hỗ trợ trong một số bảng tiêu đề ở các cấp độ khác nhau, ta có thể cắt bớt p khỏi các bảng tiêu đề ở các cấp cao hơn một cách an toàn.

Ví dụ: hãy xem xét cơ sở dữ liệu giao dịch trước đó chỉ có hai giao dịch: $\{a_1, a_2, ..., a_{100}\}$ và $\{a_1, a_2, ..., a_{50}\}$, trong đó min_sup = 2. Bởi vì a_2 trong cơ sở dữ liệu dự kiến của a_1 có hỗ trợ tương tự như a_2 trong bảng tiêu đề chung, a_2 có thể được cắt bớt khỏi bảng tiêu đề chung. Việc cắt tỉa tương tự có thể được thực hiện cho $a_3, ..., a_{50}$. Không cần phải khai thác thêm bất cứ thứ gì sau khi khai thác cơ sở dữ liệu dự kiến của a_1.

2.4.2. Kiểm tra hiệu quả từng tập mục phổ biến mới được dẫn xuất

Bên cạnh việc cắt bớt không gian tìm kiếm trong quá trình khai thác tập mục đóng, một tối ưu hóa quan trọng khác là thực hiện kiểm tra hiệu quả từng tập mục phổ biến mới được dẫn xuất để xem liệu nó có bị đóng hay không? (Do quá trình khai thác không thể đảm bảo rằng mọi tập mục phổ biến được tạo đều được đóng).

Khi một tập phổ biến mới được tạo ra, cần thực hiện kiểm tra hai loại:

i. ***Kiểm tra siêu tập hợp*** (*superset checking*): kiểm tra xem tập phổ biến mới này phải là siêu tập hợp của một số tập đóng (*closed itemsets*) đã tìm thấy có cùng độ hỗ trợ hay không?

ii. ***Kiểm tra tập con*** (*subset checking*): kiểm tra xem tập mới được tìm thấy có phải là tập con của một tập đóng (*closed itemsets*) đã tìm thấy có cùng độ hỗ trợ hay không?

2.4.2. Kiểm tra hiệu quả từng tập mục phổ biến mới được dẫn xuất

- Nếu **đã áp dụng phương pháp cắt tỉa *Sáp nhập mục*** (*Itemmerging*) dựa trên chia để trị (*divide-and-conquer*), thì việc kiểm tra superset thực sự đã được tích hợp sẵn và **không cần phải thực hiện kiểm tra superset** một cách rõ ràng. Điều này là do nếu một tập mục phổ biến XUY được tìm thấy muộn hơn tập mục X và có cùng mức hỗ trợ như X, thì nó phải nằm trong cơ sở dữ liệu dự kiến của X và phải được tạo ra trong quá trình sáp nhập tập mục.

2.4.2. Kiểm tra hiệu quả từng tập mục phổ biến mới được dẫn xuất

- Để hỗ trợ việc kiểm tra tập hợp con, một cây mẫu nén (*compressed pattern-tree*) có thể được xây dựng để duy trì tập hợp các tập mục đóng được khai thác cho đến thời điểm đang xét. Cây mẫu có cấu trúc tương tự như FP-tree ngoại trừ tất cả các tập mục đóng được tìm thấy đều được lưu trữ rõ ràng trong các nhánh cây tương ứng. Để kiểm tra tập con hiệu quả, có thể sử dụng thuộc tính sau:
  - Nếu tập mục hiện tại Sc có thể được gộp bởi tập mục đóng Sa đã tìm thấy khác thì (1) Sc và Sa có cùng độ hỗ trợ, (2) độ dài của Sc nhỏ hơn của Sa, và (3) tất cả các mục trong Sc đều chứa trong Sa.
  - Dựa trên thuộc tính này, cấu trúc chỉ mục băm hai cấp có thể được xây dựng để truy cập nhanh vào cây mẫu (*pattern-tree*): Cấp đầu tiên sử dụng mã định danh của mục cuối cùng trong Sc làm khóa băm (*hask-key*, vì mã định danh này phải nằm trong nhánh của Sc) và cấp độ thứ hai sử dụng sự hỗ trợ của Sc làm khóa băm (vì Sc và Sa có cùng độ hỗ trợ). Điều này sẽ tăng tốc đáng kể quá trình kiểm tra tập hợp con.

NỘI DUNG CHƯƠNG 4

1. Một số khái niệm cơ bản
2. Các phương pháp khai thác tập mục phổ biến
3. Các phương pháp đánh giá mẫu
4. Bài tập

3. CÁC PHƯƠNG PHÁP ĐÁNH GIÁ MẪU
(Which Patterns Are Interesting?—Pattern Evaluation Methods)

3.1. Luật kết hợp mạnh mẽ chưa chắc đã hữu ích *(Strong Rules Are Not Necessarily Interesting)*

- Việc đánh giá một luật kết hợp hữu ích hay không có thể được đánh giá một cách chủ quan hoặc khách quan.
- Trong thực tế, chỉ người dùng mới có thể đánh giá xem một luật nhất định có hữu ích hay không và đánh giá này, mang tính khách quan, có thể khác nhau ở mỗi người dùng.
- Các thước đo về mức độ hữu ích của luật kết hợp cần dựa thêm vào số liệu thống kê "đằng sau" dữ liệu, có thể được sử dụng như một bước hướng tới mục tiêu loại bỏ các luật không hữu ích mà lẽ ra sẽ được đưa ra cho người dùng.

3.1. Luật kết hợp mạnh mẽ chưa chắc đã hữu ích *(Strong Rules Are Not Necessarily Interesting)*

- Ví dụ: Giả sử ta quan tâm đến việc phân tích các giao dịch liên quan đến việc mua trò chơi máy tính và video. Trong số 10.000 giao dịch được phân tích, dữ liệu cho thấy 6000 giao dịch của khách hàng bao gồm trò chơi máy tính, trong khi 7500 giao dịch bao gồm video và 4000 giao dịch bao gồm cả trò chơi máy tính và video. Giả sử rằng một chương trình khai thác dữ liệu để khám phá các luật kết hợp được chạy trên dữ liệu trên, sử dụng độ hỗ trợ tối thiểu là 30% và độ tin cậy tối thiểu là 60%. Luật kết hợp sau đây được phát hiện:

$$buys(X, \text{“computer games”}) \Rightarrow buys(X, \text{“videos”}) \; [support = 40\%, confidence = 66\%]$$

Đây là một luật kết hợp mạnh mẽ vì luật này có

- Độ hỗ trợ là 40% (=4.000/10.000) đáp ứng mức hỗ trợ tối thiểu (min_sup=30%)
- Độ tin cậy là 66% (=4.000/6.000) đáp ứng độ tin cậy tối thiểu (min_conf=60%).

Tuy nhiên, luật gây nhầm lẫn vì trên thực tế, trò chơi và video trên máy tính có mối quan hệ liên hệ nghịch với nhau vì **việc mua một trong những mặt hàng này thực sự làm giảm khả năng mua mặt hàng kia**. Nếu không hiểu rõ sự đối nghịch này, có thể dễ dàng đưa ra những quyết định kinh doanh thiếu chính xác dựa trên luật vừa có.

3.2. Từ phân tích kết hợp đến phân tích tương quan *(From Association Analysis to Correlation Analysis)*

- Như đã thấy, độ hỗ trợ và độ tin cậy không đủ để lọc ra các luật kết hợp không hữu ích.
- Để khắc phục điểm yếu này, một thước đo tương quan có thể được sử dụng để tăng cường khung hỗ trợ - độ tin cậy (*support–confidence*) cho các luật kết hợp.
- Điều này dẫn đến các luật tương quan (*correlation*) có dạng:

$$A \Rightarrow B \text{ [support, confidence, correlation]}$$

- Nghĩa là, một luật tương quan được đo lường không chỉ bởi độ hỗ trợ và độ tin cậy của nó mà còn bởi mối tương quan giữa các tập mục A và B.
- Có nhiều thước đo tương quan khác nhau để lựa chọn. Một đặc tính chung khác là mỗi thước đo **nằm trong khoảng từ 0 đến 1** và **giá trị càng cao thì mối quan hệ giữa A và B càng chặt chẽ**.

3.2. Từ phân tích kết hợp đến phân tích tương quan *(From Association Analysis to Correlation Analysis)*

3.2.1. Thang đo tương quan Lift

- *Lift* là một thước đo tương quan đơn giản được đưa ra như sau. Sự xuất hiện của tập mục A độc lập với sự xuất hiện của tập mục B nếu P(A ∪B) = P(A)P(B); mặt khác, các tập mục A và B phụ thuộc và tương quan như các sự kiện. Định nghĩa này có thể dễ dàng được mở rộng cho nhiều hơn hai tập mục.
- Thang đo *Lift* giữa sự xuất hiện của A và B (còn được gọi là độ *lift* của luật kết hợp (hoặc tương quan) A ⇒ B) có thể được đo bằng công thức:

$$lift(A,B) = \frac{P(A \cup B)}{P(A)\,P(B)} \quad \text{tương đương với} \quad lift(A,B) = \frac{P(B|A)}{P(B)} = \frac{conf(A \Rightarrow B)}{sup(B)}$$

Nếu giá trị của *lift(A,B)*:

<1 ⇒ sự xuất hiện của A có **tương quan nghịch** với sự xuất hiện của B, nghĩa là sự xuất hiện của cái này có thể dẫn đến sự vắng mặt của cái kia.

>1 ⇒ A và B có mối **tương quan thuận**, nghĩa là sự xuất hiện của cái này hàm ý sự xuất hiện của cái kia.

=1 ⇒ A và B độc lập và **không có mối tương quan** giữa chúng.

3.2. Từ phân tích kết hợp đến phân tích tương quan *(From Association Analysis to Correlation Analysis)*

3.2.1. Thang đo tương quan Lift

- Ví dụ: Gọi $\overline{\text{Game}}$ là các giao dịch không chứa trò chơi máy tính và $\overline{\text{Video}}$ là những giao dịch không chứa video. Các giao dịch có thể được tóm tắt trong bảng sau:

	Game	$\overline{\text{Game}}$	Σrow
Video	**4000**	3500	**7500**
$\overline{\text{Video}}$	2000	500	2500
Σcol	**6000**	4000	10000

Từ bảng, ta có:

- Xác suất mua một trò chơi máy tính là P({game}) = 6.000/10.000 = 0,60
- Xác suất mua một video là P({video}) = 7.500/10.000 = 0,75
- Xác suất mua cả hai là P({trò chơi,video}) = 4.000/10.000 = 0,40.

Giá trị tương quan Lift là:

$$lift(A,B) = \frac{P(A \cup B)}{P(A) * P(B)} = \frac{P(Game \cup Video)}{P(Game) * P(Video)} = \frac{0,4}{0,6 * 0,75} = 0,89$$

Vì giá trị tương quan Lift nhỏ hơn 1 nên có mối tương quan nghịch giữa sự xuất hiện của {game} và {video}.

Mối tương quan tiêu cực như vậy không thể được xác định bằng khung hỗ trợ - độ tin cậy.

3.2.2. Thang đo χ^2

- Thước đo tương quan χ^2 đã được giới thiệu trong chương "Tiền xử lý dữ liệu". Để tính giá trị χ^2, ta lấy chênh lệch bình phương giữa giá trị được quan sát và giá trị mong đợi (kỳ vọng) cho một vị trí (cặp A và B) trong bảng thống kê, chia cho giá trị dự kiến. Số tiền này được tính tổng cho tất cả các vị trí của bảng thống kê.

- Trở lại số liệu của ví dụ trước (trong đó các số trong ngoặc là giá trị kỳ vọng):

	Game	$\overline{\text{Game}}$	Σrow
Video	4000 (4500)	3500 (3000)	7500
$\overline{\text{Video}}$	2000 (1500)	500 (1000)	2500
Σcol	6000	4000	10000

$$\chi^2 = \sum \frac{(observed - expected)^2}{expected}$$

$$= \frac{(4000-4500)^2}{4500} + \frac{(3500-3000)^2}{3000} + \frac{(3500-3000)^2}{1500} + \frac{(3500-3000)^2}{1000} = 555.6$$

Bởi vì giá trị χ^2 lớn hơn 1 và giá trị quan sát được của slot (game, video) = 4000, nhỏ hơn giá trị mong đợi là 4500, nên việc mua trò chơi và mua video có mối tương quan nghịch.

3.2.3. Độ đo tin cậy toàn phần *(all_confidence)*

Cho hai tập mục A và B, độ đo tin cậy toàn phần (*all_confidence*) của A và B được định nghĩa là:

$$all\_conf(A, B) = \frac{sup(A \cup B)}{max\{ sup(A), sup(B) \}} = min\{P(A|B), P(B|A)\}$$

trong đó, *max{sup(A), sup(B)}* là độ hỗ trợ tối đa của tập mục *A* và *B*. Do đó, mọi *conf(A,B)* cũng là độ tin cậy tối thiểu của hai luật kết hợp liên quan đến *A* và *B*, cụ thể là , *"A ⇒ B"* và *"B ⇒ A"*

3.2.4. Độ đo tin cậy tối đa *(max_confidence)*

Cho hai tập mục A và B, độ tin cậy tối đa (max_confidence) của A và B được xác định là:

$$max\_conf(A, B) = max\{ P(A|B), P(B|A) \}$$

Độ đo tin cậy tối đa (*max_confidence*) là độ tin cậy tối đa của hai luật kết hợp, "$A \Rightarrow B$" và "$B \Rightarrow A$"

3.2.5. Độ đo Kulczynski *(Kulczynski measure)*

Độ đo này được đề xuất vào năm 1927 bởi nhà toán học người Ba Lan S. Kulczynski.

Cho hai tập mục A và B, độ đo Kulczynski của A và B (viết tắt là *Kulc*) được định nghĩa là:

$$Kulc(A,B) = \frac{P(A|B) + P(B|A)}{2}$$

Có thể xem độ đo này như là mức trung bình của hai thước đo độ tin cậy. Nghĩa là, nó là trung bình của hai xác suất có điều kiện: xác suất của tập mục B cho tập mục A và xác suất của tập mục A cho tập mục B.

3.2.6. Độ đo cosin *(cosine measure)*

Cho hai tập mục A và B, độ đo cosin của A và B được định nghĩa là:

$$\text{cosine}(A, B) = \frac{P(A \cup B)}{\sqrt{P(A) \times P(B)}} = \frac{\sup(A \cup B)}{\sqrt{\sup(A) \times \sup(B)}}$$

Độ đo cosine có thể được xem như một thước đo hài hòa (*harmonized lift measure*): Hai công thức tương tự nhau ngoại trừ đối với cosine, căn bậc hai được lấy trên tích các xác suất của A và B.

Tuy nhiên, đây là một sự khác biệt quan trọng bởi vì bằng cách lấy căn bậc hai, giá trị cosin chỉ bị ảnh hưởng bởi độ hỗ trợ của A, B và A ∪B chứ không phải bởi tổng số giao dịch.

3.3. So sánh các biện pháp đánh giá mẫu *(A Comparison of Pattern Evaluation Measures)*

- Thước đo nào là tốt nhất trong 6 thang đo đã biết?
- Ví dụ: So sánh mối quan hệ giữa việc mua hai mặt hàng, sữa và cà phê, có thể được kiểm tra bằng cách tóm tắt lịch sử mua hàng của chúng trong bảng sau:

	milk	$\overline{milk}$	Σrow
coffee	mc	$\overline{m}c$	c
$\overline{coffee}$	$m\overline{c}$	$\overline{mc}$	$\overline{c}$
Σcol	m	$\overline{m}$	Σ

- Bảng sau trình bày một tập hợp các bộ dữ liệu giao dịch (từ D1 đến D6) và các giá trị liên quan cho từng thước đo trong số sáu thước đo đánh giá.

Dataset	mc	$\overline{m}c$	$m\overline{c}$	$\overline{mc}$	χ^2	lift	all_conf	max_conf	Kulc	cosine
D1	10,000	1000	1000	100,000	90557	9.26	0.91	0.91	0.91	0.91
D2	10,000	1000	1000	100	0	1	0.91	0.91	0.91	0.91
D3	100	1000	1000	100,000	670	8.44	0.09	0.09	0.09	0.09
D4	1000	1000	1000	100,000	24740	25.75	0.5	0.5	0.5	0.5

3.3. So sánh các biện pháp đánh giá mẫu *(A Comparison of Pattern Evaluation Measures)*

- **Nhận xét trên 4 bộ dữ liệu từ D1 đến D4 với các thước đo** *all_conf, max_conf, Kulc, cosine*:

Dataset	mc	$\overline{m}c$	$m\overline{c}$	$\overline{mc}$	χ^2	lift	all_conf	max_conf	Kulc	cosine
D1	10,000	1000	1000	100,000	90557	9.26	0.91	0.91	0.91	0.91
D2	10,000	1000	1000	100	0	1	0.91	0.91	0.91	0.91
D3	100	1000	1000	100,000	670	8.44	0.09	0.09	0.09	0.09
D4	1000	1000	1000	100,000	24740	25.75	0.5	0.5	0.5	0.5
D5	1000	100	10,000	100,000	8173	9.18	0.09	0.91	0.5	0.29
D6	1000	10	100,000	100,000	965	1.97	0.01	0.99	0.5	0.10

- m và c có mối tương quan **thuận** ở D1 và D2 Đối với D1 và D2, m và c có mối tương quan thuận vì mc (10.000) lớn hơn đáng kể so với $\overline{mc}$ và $m\overline{c}$ (cùng =1000). Theo trực giác, đối với những người đã mua sữa (m = 10.000 + 1000 = 11.000), rất có thể họ cũng đã mua cà phê (mc/m = 10/11 = 91%), và ngược lại.
- m và c có mối tương quan **nghịch** ở D3
- m và c có mối tương quan **trung tính** ở D4.

3.3. So sánh các biện pháp đánh giá mẫu *(A Comparison of Pattern Evaluation Measures)*

- **Nhận xét trên 4 bộ dữ liệu từ D1 đến D4 với các thước đo χ^2, Lift:**

Dataset	mc	$\overline{m}c$	$m\overline{c}$	$\overline{mc}$	χ^2	lift	all_conf	max_conf	Kulc	cosine
D1	10,000	1000	1000	100,000	90557	9.26	0.91	0.91	0.91	0.91
D2	10,000	1000	1000	100	0	1	0.91	0.91	0.91	0.91
D3	100	1000	1000	100,000	670	8.44	0.09	0.09	0.09	0.09
D4	1000	1000	1000	100,000	24740	25.75	0.5	0.5	0.5	0.5

- D1 và D2: tạo ra các giá trị đo khác nhau đáng kể cho D1 và D2 do độ nhạy của chúng với $\overline{mc}$. Trên thực tế, giá trị này thường lớn và không ổn định.
- D3: đo χ^2, *Lift* mâu thuẫn theo cách không chính xác: giá trị của chúng đối với D2 nằm giữa giá trị của D1 và D3.
- D4, cả ***lift*** và χ^2 đều biểu thị mối liên hệ tích cực cao giữa m và c, trong khi các dữ liệu khác biểu thị mối liên hệ "*trung tính*" vì tỷ lệ giữa mc và $\overline{mc}$ bằng tỷ lệ giữa mc và $m\overline{c}$, tức là 1. Điều này có nghĩa là nếu một khách hàng mua cà phê (hoặc sữa) thì xác suất người đó cũng sẽ mua sữa (hoặc cà phê) là chính xác 50%.

3.3. So sánh các biện pháp đánh giá mẫu *(A Comparison of Pattern Evaluation Measures)*

- **Trong 4 thước đo *all_conf, max_conf, Kulc, cosine*, thước đo nào tốt nhất trong việc chỉ ra các mối quan hệ trong mẫu hữu ích?**
  - Công thức về tỷ lệ mất cân bằng (*imbalance ratio* - IR)

$$IR(A,B) = \frac{|sup(A) - sup(B)|}{sup(A) + sup(B) - sup(A \cup B)}$$

    - Nếu IR(A, B) = 0 $\Rightarrow$ sự tương quan giữa A và B là như nhau thì
    - Nếu IR(A, B) càng lớn $\Rightarrow$ tỷ lệ mất cân bằng càng lớn.
    - Tỷ lệ này **không phụ thuộc** vào số lượng **giao dịch null** và không phụ thuộc vào **tổng số giao dịch**.

3.3. So sánh các biện pháp đánh giá mẫu *(A Comparison of Pattern Evaluation Measures)*

- **Trong 4 thước đo *all_conf, max_conf, Kulc, cosine*, thước đo nào tốt nhất trong việc chỉ ra các mối quan hệ trong mẫu hữu ích?**
  - Ví dụ: cho bảng dữ liệu sau:

Dataset	*mc*	$\overline{m}c$	$m\overline{c}$	$\overline{m}\overline{c}$	χ2	lift	all_conf	max_conf	Kulc	cosine
D5	1000	100	10,000	100,000	8173	9.18	0.09	0.91	0.5	0.29
D6	1000	10	100,000	100,000	965	1.97	0.01	0.99	0.5	0.10

Nhận xét: Các phương pháp đo cho kết quả rất đa dạng!

- Hai sự kiện m và c có xác suất có điều kiện không cân bằng. Nghĩa là tỉ số giữa mc và c lớn hơn 0,9 (D5 = 1000/1100 = 0.909; D6 = 1000/1010 = 0.99). Điều này có nghĩa là việc c xảy ra chắc chắn gợi ý rằng m cũng xảy ra. Tỷ lệ giữa [illegible] m [illegible] 000/1100 = 0.09; D6 = 1000/101000 = 0.009), cho thấy [illegible]
- Độ đo *all_confidence* và *cosine* xem cả hai trườ[illegible]
- Độ đo Kulc xem cả hai trường hợp là cân b[illegible]
- Thước đo *max_confidence* khẳng định mối [illegible] đối với những trường hợp này.

Theo kinh nghiệm, bạn nên sử dụng Kulc kết hợp với tỷ lệ mất cân bằng

NỘI DUNG CHƯƠNG 4

1. Một số khái niệm cơ bản
2. Các phương pháp khai thác tập mục phổ biến
3. Các phương pháp đánh giá mẫu
4. Bài tập

4. BÀI TẬP

i. Tìm tất cả các tập phổ biến và luật kết hợp bằng cách sử dụng Apriori và FP-growth tương ứng. So sánh 2 kết quả.

a. CSDL của trường hợp 1: Dữ liệu về bán hàng. Với min sup = 60% và min conf = 80%. Sử dụng thang đo *Lift* và χ^2 để đánh giá; so sánh kết quả của 2 độ đo này.

TID	*items bought*
T1	{M, O, N, K, E, Y}
T2	{D, O, N, K, E, Y }
T3	{M, A, K, E}
T4	{M, U, C, K, Y}
T5	{C, O, O, K, I, E}

b. CSDL của trường hợp 2: Dữ liệu về bán hàng. Với min sup = 60% và min conf = 80%. Sử dụng *Độ tin cậy toàn phần* và *Độ tin cậy tối đa* để đánh giá; so sánh kết quả của 2 độ đo này.

cust ID	*TID*	*Items*
01	T1	{King's-Crab, Sunset-Milk, Dairyland-Cheese, Best-Bread}
02	T2	{Best-Cheese, Dairyland-Milk, Goldenfarm-Apple, Tasty-Pie, Wonder-Bread}
01	T3	{Westcoast-Apple, Dairyland-Milk, Wonder-Bread, Tasty-Pie}
03	T4	{Wonder-Bread, Sunset-Milk, Dairyland-Cheese}

4. BÀI TẬP

i. Đặt min sup = 60% và min conf = 80%. Tìm tất cả các tập phổ biến và luật kết hợp bằng cách sử dụng Apriori và FP-growth tương ứng. So sánh 2 kết quả.

c. CSDL của trường hợp 3: Thống kê tiện nghi sử dụng của các hộ. Với min sup = 50% và min conf = 50%. Sử dụng *Độ đo Kulc* và *Độ đo cosine* để đánh giá; so sánh kết quả của 2 độ đo này.

Hộ	***Tiện nghi sử dụng***
1	Tivi, Máy vi tính
2	Tủ lạnh, máy lạnh
3	Tivi, Máy giặt, Máy lạnh
4	Tivi, Tủ lạnh, Máy lạnh
5	Tivi, Tủ lạnh, Máy giặt
6	Tivi, Tủ lạnh, Máy vi tính
7	Tivi, Tủ lạnh, Máy giặt, Máy lạnh, Máy vi tính
8	Tivi, Máy giặt, Máy vi tính

4. BÀI TẬP

ii. Bảng sau tóm tắt dữ liệu giao dịch trong 1 siêu thị, trong đó *hot dogs* đề cập đến các giao dịch có mua *hot dogs*, $\overline{hot\ dogs}$ đề cập đến các giao dịch không mua *hot dogs*, hamburger đề cập đến các giao dịch có mua *hamburger* và $\overline{hambugers}$ đề cập đến giao dịch không mua *hamburger*.

	hot dogs	$\overline{hot\ dogs}$	Σrow
hamburgers	2000	500	2500
$\overline{hambugers}$	1000	1500	2500
Σcol	3000	2000	5000

a. Giả sử đã khai thác được luật kết hợp "hot dog ⇒ hamburger". Với ngưỡng hỗ trợ tối thiểu là 25% và ngưỡng tin cậy tối thiểu là 50%, luật kết hợp này có mạnh không?

b. Dựa trên dữ liệu đã cho, việc mua hot dog có độc lập với việc mua hamburger không? Nếu không, mối quan hệ tương quan nào tồn tại giữa hai sản phẩm này?

c. So sánh việc sử dụng các độ đo độ tin cậy toàn phần (*all_confidence*), độ tin cậy tối đa (*max_confidence*), Kulczynski và cosine với độ đo lift và độ đo tương quan trên dữ liệu đã cho.


Q&A
How
Why
When
Where
What
Who