

BỘ GIÁO DỤC VÀ ĐÀO TẠO  
ĐẠI HỌC KINH TẾ TP HỒ CHÍ MINH  
TRƯỜNG CÔNG NGHỆ VÀ THIẾT KẾ



NHÓM 7

ĐỀ TÀI BÁO CÁO:

KHAI PHÁ LUẬT KẾT HỢP BẰNG THUẬT TOÁN ECLAT  
TRÊN BỘ DỮ LIỆU MARKET BASKET ANALYSIS 5

**Học phần:** Khai Phá Dữ Liệu

**Chuyên Ngành:** KHOA HỌC DỮ LIỆU

**Khóa:** K47

**Giảng Viên:** TS. Nguyễn An Té

TP. Hồ Chí Minh, Ngày 15 tháng 12 năm 2023

## LỜI CẢM ƠN

Kính gửi Thầy Nguyễn An Té,

Chúng em xin dành những lời tri ân sâu sắc nhất đến thầy với lòng biết ơn không ngừng, với sự tận tâm giảng dạy của thầy. Thầy đã là nguồn động viên vô cùng quan trọng trong hành trình học tập và nghiên cứu của chúng em tại Đại học Kinh tế TPHCM. Sự hiểu biết sâu rộng, sự tận tâm và sự cống hiến không ngừng nghỉ của thầy đã giúp đỡ chúng em không chỉ trong việc tiếp cận kiến thức mà còn trong việc phát triển kỹ năng và tư duy. Thầy không chỉ là một người thầy giáo mẫu mực, mà còn là một người hướng dẫn tận tâm, luôn sẵn lòng chia sẻ kiến thức, kinh nghiệm và sự định hình cho hành trình nghiên cứu của chúng em. Trong quá trình làm đồ án vẫn còn các hạn chế, sai sót và chưa tối ưu về mặt kiến thức và kỹ năng. Nhóm chúng em mong nhận được sự phản hồi, nhận xét của thầy để có thể cải thiện được báo cáo tốt hơn nữa. Một lần nữa, chúng em xin bày tỏ lòng biết ơn đến thầy vì đã dành thời gian và kiến thức quý báu để hỗ trợ và hướng dẫn chúng em hoàn thành được báo cáo kết thúc môn.

## MỤC LỤC

<b>CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI.....</b>	<b>4</b>
1.1 Giới thiệu đề tài .....	4
1.2 Mục tiêu đề tài .....	4
1.3 Phương pháp nghiên cứu .....	4
1.4 Tài nguyên sử dụng .....	4
1.5 Ngôn ngữ sử dụng .....	4
<b>CHƯƠNG 2: TỔNG QUAN VỀ LUẬT KẾT HỢP.....</b>	<b>5</b>
2.1 Tổng quan về luật kết hợp .....	5
2.1.1 Giới thiệu khai phá luật kết hợp .....	5
2.1.2 Giới thiệu luật kết hợp .....	5
2.2 Tổng quan về thuật toán Apriori .....	7
2.2.1 Giới thiệu thuật toán Apriori .....	7
2.2.2 Tính chất của Apriori.....	8
2.2.3 Ưu điểm và nhược điểm: .....	8
2.3 Tổng quan về thuật toán Fp-growth .....	9
2.3.1 Giới thiệu thuật toán FP-Growth .....	9
2.3.2 Ưu điểm và nhược điểm của thuật toán .....	9
<b>CHƯƠNG 3: TỔNG QUAN VỀ THUẬT TOÁN ECLAT .....</b>	<b>11</b>
3.1 Giải thuật tìm kiếm trong ECLAT .....	11
3.2 Tổng quan về thuật toán ECLAT .....	13
3.3 Ưu điểm và nhược điểm của thuật toán ECLAT .....	14
3.4 Sơ đồ giải thuật.....	14
3.5 Ví dụ minh họa.....	17
3.6 So sánh thuật toán Apriori, ECLAT và FP - Growth .....	19
<b>CHƯƠNG 4: XÂY DỰNG MÔ HÌNH.....</b>	<b>24</b>
4.1 Tổng quan về bộ dữ liệu.....	24
4.2 Tiền xử lý dữ liệu .....	24
4.3 Trực quan dữ liệu .....	27
4.4 Cài đặt thuật toán.....	28
<b>CHƯƠNG 5: KẾT QUẢ VÀ ĐÁNH GIÁ .....</b>	<b>31</b>
5.1 So sánh thời gian chạy của các thuật toán .....	31
5.2 So sánh tập luật của các thuật toán:.....	34
5.3 Chọn và trực quan tập luật đáng quan tâm.....	36

5.4 Đưa ra khuyến nghị dựa trên tập luật .....	39
<b>CHƯƠNG 6: KẾT LUẬN .....</b>	<b>40</b>
<b>TÀI LIỆU THAM KHẢO .....</b>	<b>41</b>
<b>DANH SÁCH THÀNH VIÊN .....</b>	<b>43</b>

## CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

### **1.1 Giới thiệu đề tài**

Luật kết hợp đóng một vai trò quan trọng trong việc hiểu và phân tích các mối quan hệ, mô hình trong dữ liệu. Trong đời sống hàng ngày, áp dụng các nguyên lý của luật kết hợp giúp chúng ta hiểu rõ hơn về sự tương tác và liên kết giữa các yếu tố khác nhau, từ đó đưa ra những quyết định thông minh và tối ưu hóa kết quả.

Trong lĩnh vực kinh doanh và bán lẻ, luật kết hợp được sử dụng để phân tích hành vi mua sắm của khách hàng. Việc hiểu rõ những sản phẩm nào thường được mua cùng nhau giúp doanh nghiệp đưa ra chiến lược đặt hàng, quảng cáo, và giảm giá hiệu quả hơn. Điều này giúp tối ưu hóa lợi nhuận và cải thiện trải nghiệm mua sắm của khách hàng.

ECLAT, một trong những thuật toán khai thác luật kết hợp, đặc biệt phổ biến trong lĩnh vực kinh doanh. Trong nghiên cứu này, nhóm chúng em đã dành thời gian để hiểu sâu hơn về luật kết hợp, bao gồm tổng quan, khái niệm và các thuật toán liên quan. Nhóm tập trung nghiên cứu chi tiết về thuật toán ECLAT và cách nó được ứng dụng trong việc phân tích giỏ hàng. Hy vọng rằng thông qua nghiên cứu này, nhóm sẽ có cái nhìn chi tiết và sâu sắc hơn về lý thuyết và ứng dụng của thuật toán trên bộ dữ liệu thực tế.

### **1.2 Mục tiêu đề tài**

Qua đề tài này, nhóm mong muốn hiểu rõ hơn về thuật toán khai thác luật kết hợp ECLAT, đi từ các khái niệm cơ bản, điểm mạnh, điểm yếu, giải thuật và ví dụ minh họa. Ngoài ra nhóm còn thực hiện việc ứng dụng thuật toán ECLAT vào bộ dữ liệu Market Basket Analysis 5 để có thể tìm ra các tập phổ biến, song song đó nhóm tiến hành đánh giá hiệu suất của thuật toán Eclat và các thuật toán khai thác luật kết hợp khác để có cái nhìn tổng quan hơn về các thuật toán này.

### **1.3 Phương pháp nghiên cứu**

- Tìm hiểu lý thuyết về luật kết hợp, thuật toán ECLAT.
- Sử dụng thuật toán ECLAT để tìm ra các luật kết hợp mạnh.

### **1.4 Tài nguyên sử dụng**

- Bộ dữ liệu Market Basket Analysis 5.

### **1.5 Ngôn ngữ sử dụng**

- Ngôn ngữ lập trình phân tích dữ liệu Python.

## CHƯƠNG 2: TỔNG QUAN VỀ LUẬT KẾT HỢP

### 2.1 Tổng quan về luật kết hợp

#### 2.1.1 Giới thiệu khai phá luật kết hợp

Khai phá luật kết hợp là một khái niệm quan trọng trong khai phá dữ liệu và thuật toán máy học. Mục tiêu của nó là khám phá các mối quan hệ giữa các mục trong một tập dữ liệu. Khái niệm khai phá luật kết hợp có nguồn gốc từ những năm 1960. Đây là kết quả của công trình nghiên cứu được công bố trong bài báo "*The GUHA method and its meaning for data mining*" vào năm 1966, liên quan đến phương pháp GUHA do Petr Hájek cùng với nhóm nghiên cứu phát triển.

Một ứng dụng tiêu biểu của việc khai phá luật kết hợp là trong việc phân tích giỏ hàng của ngành bán lẻ. Mục đích chính là tìm hiểu về các mối quan hệ giữa các sản phẩm mà khách hàng thường mua cùng nhau. Những quy tắc như *nếu một khách hàng mua sản phẩm X thì họ sẽ cũng thường mua thêm sản phẩm Y* có thể phần nào giúp cho các doanh nghiệp trong việc xây dựng các chiến lược quảng cáo, quản lý đặt hàng cũng như tối ưu hóa việc sắp xếp các sản phẩm trên kệ hàng.

#### 2.1.2 Giới thiệu luật kết hợp

Luật kết hợp (Association rule) là mối quan hệ kết hợp giữa các tập thuộc tính trong cơ sở dữ liệu và là công cụ hữu ích để khám phá các mối liên kết trong dữ liệu. Luật kết hợp là một mệnh đề kéo theo có dạng  $X \Rightarrow Y$ , trong đó  $X \subset I \neq \emptyset$ ,  $Y \subset I \neq \emptyset$ , và

$$X \cap Y = \emptyset.$$

Khai thác quy tắc kết hợp có 2 bước:

- Tìm tất cả các frequent itemsets: Mỗi itemset này sẽ xuất hiện ít nhất bằng một số lượng hỗ trợ tối thiểu ( $minSup$ ) được xác định trước.
- Tạo ra các quy tắc liên kết mạnh từ các frequent itemsets: Các quy tắc này phải thỏa mãn độ hỗ trợ tối thiểu ( $minSup$ ) và độ tin cậy tối thiểu ( $minConf$ ).

Các khái niệm liên quan: (Nguyễn, 2023)

Database:  $D = \{T_i\}$  là một bộ gồm hai thành phần ( $T$ ,  $I$ ), trong đó:

- Transaction (giao dịch): tập hợp các items xuất hiện cùng nhau.  
 $\emptyset \neq T \subseteq I$ .

(items trong  $T$  được sắp xếp tăng dần)

- Items:  $I = \{I_1, I_2, \dots, I_n\}$ . Các đơn vị cơ bản trong một tập dữ liệu được gọi là các item. Trong phân tích giỏ hàng, mỗi item là một loại sản phẩm được mua cùng nhau trong một giao dịch.
- Itemset: các tập hợp của các items.

$$X \subseteq I.$$

- k-itemset: được sử dụng để xác định các tập hợp mục xuất hiện thường xuyên trong dữ liệu.

$$X_k = \{I_1, I_2, \dots, I_k\} \quad k \leq n$$

- Tid\_set:  $tidset(X) = \{T_m / T_m \in T \wedge X \subseteq T_m\}$  là một tập hợp các giao dịch chứa  $X$ .
- Tần số (frequency) của itemset  $X$  = số lần xuất hiện của  $X$  trong các giao dịch của  $D$ .

$$freq(X) = | \{T \in D / X \subseteq T\} |$$

Tid	Items
1	kem, sữa, bơ
2	pizza, kem
3	bánh, pizza, kem, sữa
4	pizza, bánh, bơ, kem
5	sữa, bơ

$freq(\{\text{sữa, bơ}\}) = 2$   
 $freq(\{\text{pizza, kem}\}) = 3$

- Độ hỗ trợ (support) của  $X \Rightarrow Y$ : độ phổ biến của luật (trong  $D$ ).

$$sup(X \Rightarrow Y) = P(X \cup Y) = \frac{freq(X \cup Y)}{|D|}$$

Tid	Items
1	kem, sữa, bơ
2	pizza, kem
3	bánh, pizza, kem, sữa
4	pizza, bánh, bơ, kem
5	sữa, bơ

$sup(\text{sữa} \Rightarrow \text{bơ}) = (2/5) = 40\%$   
 $sup(\text{pizza} \Rightarrow \text{kem}) = (3/5) = 60\%$

- Độ tin cậy (confidence) của  $X \Rightarrow Y$ : độ chính xác của luật.

$$conf(X \Rightarrow Y) = P(Y / X) = \frac{freq(X \cup Y)}{freq(X)}$$

Tid	Items
1	kem, sữa, bơ
2	pizza, kem
3	bánh, pizza, kem, sữa
4	pizza, bánh, bơ, kem
5	sữa, bơ

$conf(\text{sữa} \Rightarrow \text{bơ}) = (2/3) = 66.6\%$   
 $conf(\text{pizza} \Rightarrow \text{kem}) = (3/3) = 100\%$

- Luật kết hợp mạnh (strong association rule): Các luật mà thỏa mãn cả ngưỡng hỗ trợ tối thiểu ( $minSup$ ) và ngưỡng độ tin cậy tối thiểu ( $minConf$ ) được gọi là mạnh.

$$sup(X \Rightarrow Y) \geq minSup$$

$$conf(X \Rightarrow Y) \geq minConf$$

- Tập phổ biến (frequent itemset):

$$freq(X) \geq minSup$$

$$L_k = \{frequent\ k-itemsets\}$$

- Độ đo tương quan (Lift): được sử dụng để đánh giá mẫu phổ biến trong khai phá luật kết hợp.

$$lift(X \Rightarrow Y) = \frac{sup(X \Rightarrow Y)}{sup(X).sup(Y)} = \frac{conf(X \Rightarrow Y)}{sup(Y)}$$

$$lift(X \Rightarrow Y) = \frac{P(X \cap Y)}{P(X).P(Y)} = \frac{P(Y|X)}{P(Y)}$$

- ❖ Nếu  $lift(X \Rightarrow Y) = 1$ :  $P(Y|X) = P(Y)$ : X và Y độc lập, không có tương quan.
- ❖ Nếu  $lift(X \Rightarrow Y) > 1$ : X tương quan thuận với Y, nghĩa là khi X xuất hiện thì khả năng Y cũng xuất hiện sẽ cao.
- ❖ Nếu  $lift(X \Rightarrow Y) < 1$ : X tương quan nghịch với Y, nghĩa là khi X xuất hiện thì khả năng không xuất hiện của Y sẽ cao.

Ngày nay, khai phá luật kết hợp được áp dụng rộng rãi trong nhiều lĩnh vực khác nhau, từ thương mại điện tử đến y tế và tài chính, giúp khám phá các mối quan hệ giữa các sản phẩm mà khách hàng thường mua cùng nhau, từ đó tạo ra các chiến lược quảng cáo và đặt hàng hiệu quả.

## 2.2 Tổng quan về thuật toán Apriori

### 2.2.1 Giới thiệu thuật toán Apriori

- Thuật toán Apriori là thuật toán được đề xuất bởi R.Agrawal và R.Srikant vào năm 1993 thông qua bài báo có tựa đề là "*Mining Association Rules between Sets of Items in Large Databases*", đã tạo ra một ảnh hưởng lớn đối với các thuật toán khai phá dữ liệu sau này. Thuật toán này hoạt động dựa trên hai ngưỡng  $minSup$  và  $minConf$ .
- Thuật toán Apriori bao gồm 2 giai đoạn :
  - Khai phá những tập phổ biến: tìm tập L gồm tất cả các itemset phổ biến (có độ hỗ trợ không nhỏ hơn  $minSup$ ).
  - Xác định những luật kết hợp (mạnh) từ các tập phổ biến.
- Apriori tìm tất cả X phổ biến bằng Breadth-First Search: dùng k-itemset để tìm (k+1)-itemsets. (*Nguyễn, 2023*)
  - Tạo  $L_1 = \{ frequent\ 1-itemset \}$
  - Mở rộng  $L_1$  để tìm  $L_2 = \{ frequent\ 2-itemset \}$
  - Mở rộng  $L_2$  để tìm  $L_3 = \{ frequent\ 3-itemset \}$
  - Tiếp tục cho đến khi không tìm được k-itemset nào.

### 2.2.2 Tính chất của Apriori

(Jiawei Han et al., 2012), (Nguyễn, 2023)

- “Mọi tập con khác rỗng của một tập phô biến X cũng là tập phô biến.”
- “Mọi tập chứa X không phô biến cũng là tập không phô biến.”

- Ví dụ: Nếu X phô biến thì  $\forall A \subset X, A \neq \emptyset$ , A cũng phải là tập phô biến.
- Cách  $L_{k-1}$  được sử dụng để tìm  $L_k$  ( $k \geq 2$ ). Một quy trình hai bước được thực hiện, bao gồm join step và prune step.
  - Công đoạn nối kết (join step): tạo  $C_k$  chứa các tập ứng viên  
Phép kết:  $C_k = L_{k-1} \bowtie L_{k-1}$

Điều kiện kết hai tập ứng viên (dãy hay mảng) x, y  $L_{k-1}$ :

$$x[i] = y[i] \quad \forall i \leq (k-2)$$

$$x[k-1] < y[k-1]$$

Kết quả  $x \bowtie y = \{x[1], \dots, x[k-1], y[k-1]\}$

- Công đoạn cắt tia (prune step):

$$L_k: \{C \in C_k | freq(C) \geq minSup\}$$

### 2.2.3 Ưu điểm và nhược điểm:

#### - Ưu điểm:

- **Tính toán các tập phô biến lớn:** Thuật toán Apriori có khả năng xử lý và tìm ra những mẫu phô biến trong tập dữ liệu lớn. Nói cách khác, nó giúp tìm các itemsets xuất hiện cùng nhau nhiều lần.
- **Dễ hiểu và áp dụng:** Thuật toán khá đơn giản và dễ hiểu. Nó hoạt động bằng cách lặp đi lặp lại qua các cơ sở dữ liệu, mỗi lần tăng kích thước của itemsets và chỉ giữ lại những itemsets phô biến. Điều này giúp thuật toán dễ dàng được áp dụng trong nhiều ngữ cảnh khai thác dữ liệu khác nhau.

#### - Nhược điểm:

- **Tốn kém về mặt tính toán:** Thuật toán cần quét toàn bộ cơ sở dữ liệu để tìm các mục hỗ trợ, tức là nó phải xem xét mỗi giao dịch trong cơ sở dữ liệu. Do đó, có thể gây tốn kém về mặt tính toán, đặc biệt là khi cơ sở dữ liệu lớn.
- **Tốn kém về mặt bộ nhớ và thời gian xử lý:** Trong mỗi vòng lặp của thuật toán, nó tạo ra một tập ứng viên  $C_k$  mới, trong đó,  $k$  là số lượng item. Số lượng các tập ứng viên này có thể tăng lên rất nhanh khi  $k$  tăng, vậy nên dẫn đến việc tốn kém bộ nhớ và thời gian xử lý lâu.

## 2.3 Tổng quan về thuật toán Fp-growth

### 2.3.1 Giới thiệu thuật toán FP-Growth

- Thuật toán FP-growth (Frequent Pattern growth) được giới thiệu bởi các tác giả J.Han, J.Pei và Y.Yin trong bài báo “*Mining Frequent Patterns without Candidate Generation*” vào năm 2000. Đây là một thuật toán quan trọng trong lĩnh vực khai phá dữ liệu, được tạo ra để khám phá các mẫu phổ biến trong dữ liệu một cách hiệu quả. Nó được giới thiệu như một giải pháp để giải quyết một số hạn chế của thuật toán Apriori, đặc biệt là về khả năng sinh ra một lượng lớn các tập ứng viên.

Trái với Apriori, FP-growth sử dụng cấu trúc cây FP-Tree để nén cơ sở dữ liệu, giữ lại thông tin về các item phổ biến. Cấu trúc này giúp giảm đáng kể không gian lưu trữ và thời gian xử lý so với Apriori.

Sau khi xây dựng FP-Tree, thuật toán FP-growth sử dụng phương pháp chia để trị để khai thác các itemset phổ biến. Thay vì tạo ra tất cả các tập ứng viên, FP-growth chỉ tập trung vào một tập con của cơ sở dữ liệu tại mỗi bước, giúp tăng tốc quá trình khai thác.

- Các bước thực hiện: (*Nguyễn, 2023*)
  - B1. Xây dựng F-list từ  $L_1$ .
  - B2. Xây dựng FP-Tree và Item Header Table.
  - B3. Duyệt “ngược” các item  $I_k \in F\text{-list}$  để tạo các tập phổ biến:
    - a) Tạo cơ sở mẫu điều kiện CPB từ prefix của các paths đến  $I_k$ .
    - b) Dùng CPB như CSDL để tạo FP-Tree điều kiện.

Tạo Item Header Table chỉ bao gồm các items phổ biến (cộng dồn freq(item) trong các paths).

c) Phát sinh tập phổ biến từ mỗi path trong FP-Tree điều kiện.

### 2.3.2 Ưu điểm và nhược điểm của thuật toán

#### - Ưu điểm:

- **Giảm số lượng tập ứng viên:** FP-Growth không cần phải tạo ra các tập ứng viên, mà chỉ sử dụng cấu trúc cây FP-Tree để lưu trữ thông tin về các mục phổ biến. Điều này giúp giảm đáng kể không gian tìm kiếm và chi phí tính toán.
- **Giảm số lần quét cơ sở dữ liệu:** Đối với các tập dữ liệu lớn, FP-growth thường hiệu quả hơn Apriori, bởi vì không cần phải quét lại toàn bộ cơ sở dữ liệu nhiều lần mà chỉ cần quét cơ sở dữ liệu một lần để xây dựng FP-Tree. Điều này giúp làm giảm đáng kể thời gian thực hiện.
- **Dễ triển khai và cài đặt:** FP-growth thường dễ triển khai và cài đặt, đặc biệt là khi sử dụng các cấu trúc dữ liệu như cây FP-Tree. Cấu trúc cây FP-Tree cũng là một cấu trúc dữ liệu phổ biến và hiệu quả.

#### - Nhược điểm:

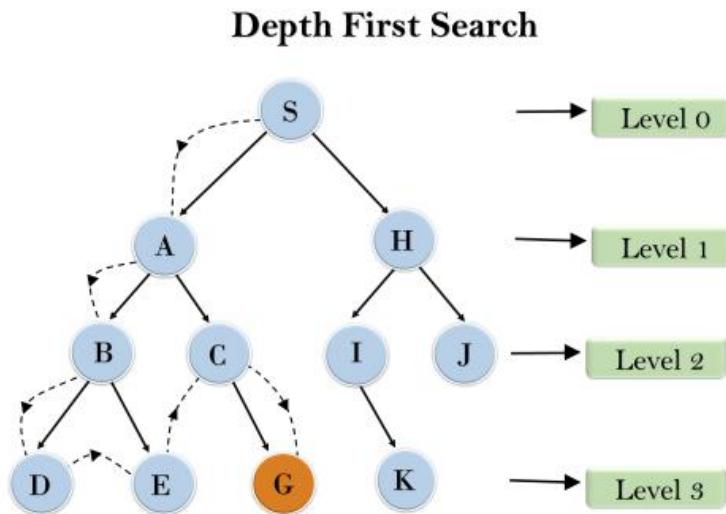
- **Cần phải xây dựng FP-Tree từ đầu khi thêm dữ liệu mới:** Khi có dữ liệu mới được thêm vào, cấu trúc cây FP-Tree có thể bị thay đổi, do đó cần phải xây dựng lại cây FP-Tree từ đầu. Điều này gây tốn thời gian và bộ nhớ.

- **Không thích hợp cho việc xử lý dữ liệu có tần suất thay đổi:** Nếu dữ liệu thay đổi thường xuyên, ví dụ như trong các tập dữ liệu về thời tiết hay là giao thông, thuật toán FP-Growth có thể không phản ánh được các mối quan hệ thực tế giữa các item. Việc xây dựng lại FP-Tree có thể trở nên tốn kém và không hiệu quả.
- **Chiếm nhiều bộ nhớ:** Thuật toán FP-Growth mặc dù có thể tiết kiệm bộ nhớ hơn Apriori do không cần tạo ra các tập ứng viên. Tuy nhiên, trong một số trường hợp, FP-Tree cũng có thể chiếm nhiều bộ nhớ, đặc biệt là khi xử lý các tập dữ liệu lớn.

## CHƯƠNG 3: TỔNG QUAN VỀ THUẬT TOÁN ECLAT

### 3.1 Giải thuật tìm kiếm trong ECLAT

Depth-First Search (DFS) là một thuật toán tìm kiếm trong đồ thị được áp dụng vào ECLAT, nó được sử dụng để duyệt qua và truy cập tất cả các đỉnh của đồ thị theo một hướng sâu vào bên trong trước khi quay lại và di chuyển sang các nhánh khác. DFS thường được triển khai bằng cách sử dụng ngăn xếp (stack) để theo dõi các đỉnh cần duyệt.

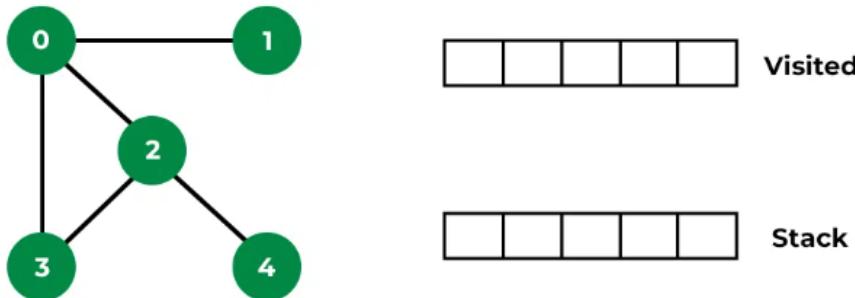


Các bước thực hiện thuật toán Depth - First Search:

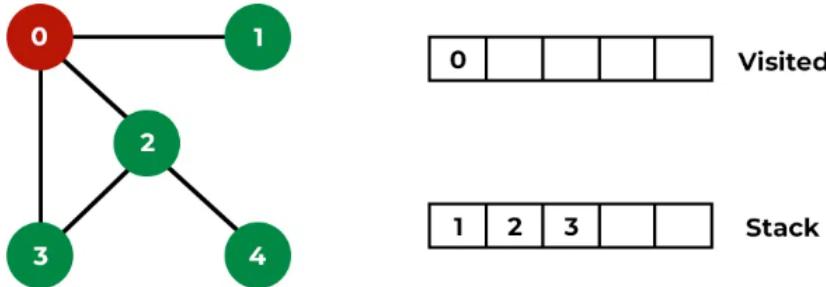
Tìm kiếm theo chiều sâu sử dụng một ngăn xếp để lưu trữ các đỉnh đã được duyệt. Thuật toán bắt đầu bằng cách đưa đỉnh gốc vào ngăn xếp. Sau đó, thuật toán lấy một đỉnh từ ngăn xếp và duyệt tất cả các đỉnh liền kề của đỉnh đó. Nếu một đỉnh liền kề chưa được duyệt, thì thuật toán sẽ đưa đỉnh đó vào ngăn xếp. Quá trình này tiếp tục cho đến khi ngăn xếp trống.

Ví dụ minh họa:

- Bước 1: Tạo một ngăn xếp với kích thước bằng tổng số đỉnh của đồ thị.



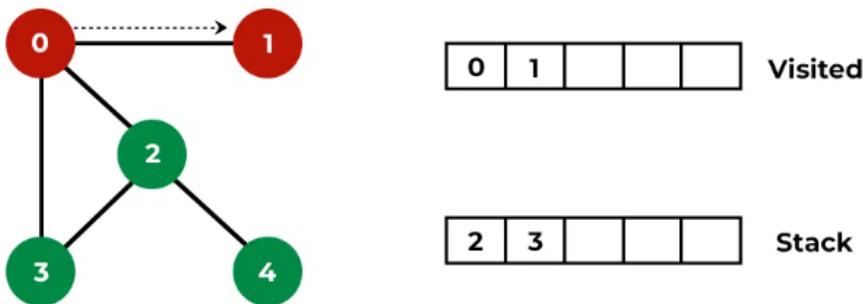
- Bước 2: Chọn một đỉnh bất kỳ (ở đây là đỉnh 0) làm điểm khởi đầu của quá trình duyệt. Đánh dấu đỉnh 0 đã truy cập. Sau đó đưa các đỉnh liền kề chưa được duyệt của đỉnh 0 (là đỉnh 1, 2, 3) vào ngăn xếp:



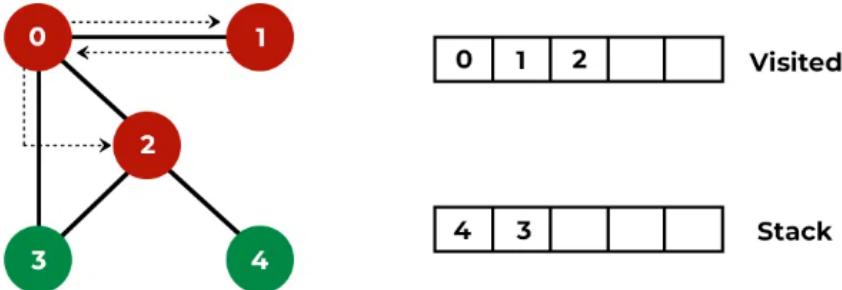
- Bước 3: Lấy đỉnh trên cùng của ngăn xếp là đỉnh 1.

Nếu đỉnh này chưa được truy cập, đánh dấu là đã truy cập và thực hiện các thao tác cần thiết.

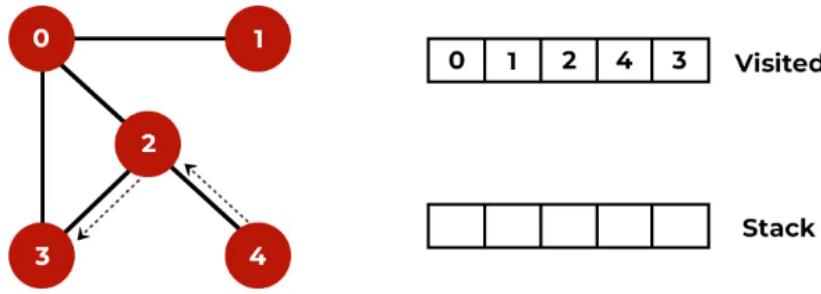
Kiểm tra tất cả các đỉnh liền kề (chưa được truy cập trước đó) của đỉnh 1 và đặt tất cả các nút đó vào ngăn xếp



- Bước 4: Đỉnh 2 đang ở trên cùng của ngăn xếp, do đó truy cập đỉnh 2 và lấy nó ra khỏi ngăn xếp, đưa các đỉnh liền kề chưa được truy cập của đỉnh 2 (là 3, 4) vào ngăn xếp.



- Bước 5: Lặp lại các bước tương tự như trên, cho đến khi ngăn xếp trống. Ta thu được kết quả là thứ tự truy cập đến các đỉnh như sau:  $0 \Rightarrow 1 \Rightarrow 2 \Rightarrow 4 \Rightarrow 3$ .

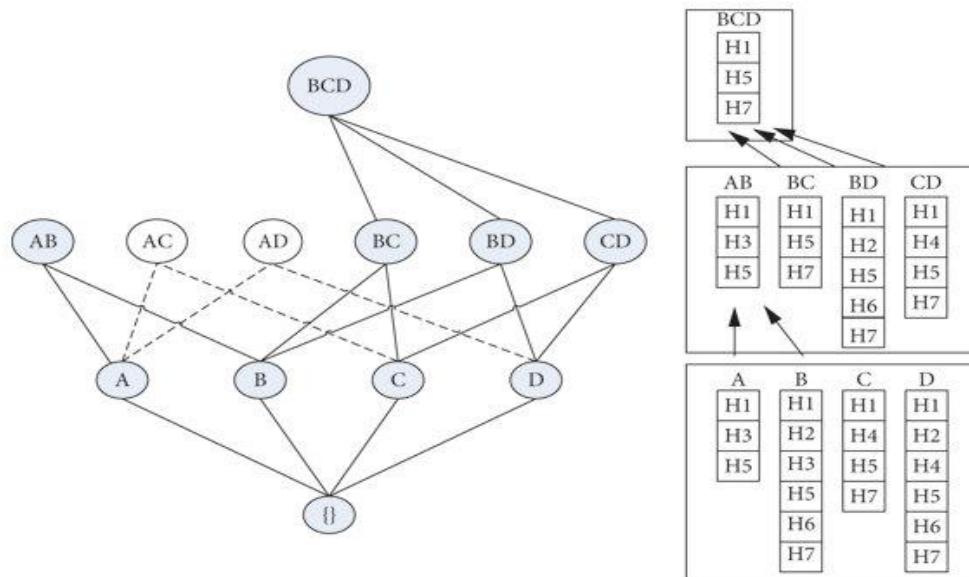


### 3.2 Tổng quan về thuật toán ECLAT

Thuật toán ECLAT (Equivalence Class Clustering and bottom-up Lattice Traversal) là một thuật toán khai thác luật kết hợp, sử dụng phân cụm lớp tương đương kết hợp duyệt mảng lướt từ dưới lên để tìm các tập phổ biến. Thuật toán này được giới thiệu lần đầu vào năm 1977 ở bài báo khoa học có tên là “New Algorithms for Fast Discovery of Association Rules” của nhóm tác giả: Srinivasan Parthasarathy, Mitsunori Ogihara, Wei li.

Sự ra đời của ECLAT nhằm khắc phục những điểm yếu của một số thuật toán khác trong lĩnh vực khai thác dữ liệu ví dụ như Apriori. Những sự thay đổi dẫn đến sự tối ưu hơn của thuật toán ECLAT bao gồm việc nó sử dụng thuật toán Depth-First Search để khám phá các tập dữ liệu, sử dụng cơ sở dữ liệu dọc, sử dụng lớp tương đương.

Vertical format được hiểu là cách tổ chức dữ liệu theo chiều dọc, mỗi mặt hàng có một dòng riêng biệt, liệt kê các mã giao dịch (TID) chứa mặt hàng đó (trái ngược với horizontal format được hiển thị theo chiều ngang, mỗi giao dịch được lưu trên một dòng riêng biệt với các mặt hàng được mua liệt kê). (A. Subashini & M. Karthikeyan, 2019)



Nguồn ảnh: (Xinhao Wang et al., 2021)

### 3.3 Ưu điểm và nhược điểm của thuật toán ECLAT

(Rana Ishita & Amit Rathod, 2016)

#### a. Ưu điểm

**Hiệu quả về bộ nhớ:** Việc sử dụng thuật toán Depth-First Search trong ECLAT giúp giảm thiểu yêu cầu về bộ nhớ bằng cách chỉ giữ thông tin cần thiết để thực hiện tìm kiếm hiệu quả, thay vì lưu trữ toàn bộ không gian tìm kiếm.

**Không cần quét lại cơ sở dữ liệu:** ECLAT quét tập dữ liệu được tạo ra tại thời điểm hiện tại, loại bỏ sự cần thiết phải quét lại toàn bộ cơ sở dữ liệu cho mỗi lần tạo itemset mới.

#### b. Nhược điểm:

**Bộ T-id dài:** ECLAT lưu trữ các T-id (transaction IDs) cho từng itemset, dẫn đến các bộ T-id có thể trở nên rất dài và tốn kém để thao tác, ảnh hưởng đến hiệu suất.

#### c. Hạn chế:

**Thuật toán ECLAT không tận dụng toàn bộ thuộc tính của Apriori:** Một tính chất quan trọng trong việc giảm số lượng itemset ứng viên cần khám phá (thuộc tính của Apriori cho biết con của phổ biến là phổ biến, cha của không phổ biến thì cũng không phổ biến), dẫn đến việc khám phá nhiều hơn các itemset ứng viên không cần thiết, làm giảm hiệu quả.

**Thuật toán ECLAT thực hiện tính toán bằng cách giao các bộ T-id (transaction IDs) của các itemset tương ứng:** Bộ T-id được hiểu là một tập hợp các ID giao dịch chứa itemset tương ứng. Đối với các itemset nhỏ, bộ T-id có thể tương đối nhỏ và việc tính toán giao của các bộ T-id có thể không tốn kém về mặt tính toán. Tuy nhiên, đối với các itemset lớn, bộ T-id có thể rất lớn và việc tính toán giao của các bộ T-id có thể tốn nhiều thời gian và tài nguyên.

**Thứ tự giải quyết lớp của ECLAT là từ dưới lên trên:** Điều này có nghĩa là các lớp itemset nhỏ hơn sẽ được giải quyết trước các lớp itemset lớn hơn. Thứ tự này có thể hiệu quả trong một số trường hợp, chẳng hạn như khi các tập phổ biến có kích thước nhỏ. Tuy nhiên, thứ tự này có thể không tối ưu trong những trường hợp khác, chẳng hạn như khi các tập phổ biến có kích thước lớn hoặc khi có nhiều itemset cần so sánh.

**Nhận xét:** ECLAT là một thuật toán hiệu quả để tìm tập phổ biến. Nó có ưu điểm là không cần quét lại cơ sở dữ liệu và hiệu quả về bộ nhớ. Tuy nhiên, nó cũng có một số nhược điểm như tính toán phức tạp và thứ tự giải quyết lớp không tối ưu.

### 3.4 Sơ đồ giải thuật

**Các khái niệm được sử dụng trong thuật toán ECLAT:** (Philippe Fournier-Viger, 2000)

- $I = \{I_1, I_2, \dots, I_m\}$  là tập hợp các mục (sản phẩm) được bán trong một cửa hàng.  
VD:  $I = \{\text{kem}, \text{trứng}, \text{táo}, \text{sữa}, \text{kẹo}\}$
- Itemset X là tập hợp các mục ( $X \subseteq I$ ). Một itemset nếu chứa k mục thì được gọi là có kích thước k.

Ví dụ:

$\{\text{kem}\}, \{\text{trứng}\}, \{\text{táo}\}, \{\text{sữa}\}, \{\text{kẹo}\}$  với kích thước  $k = 1$

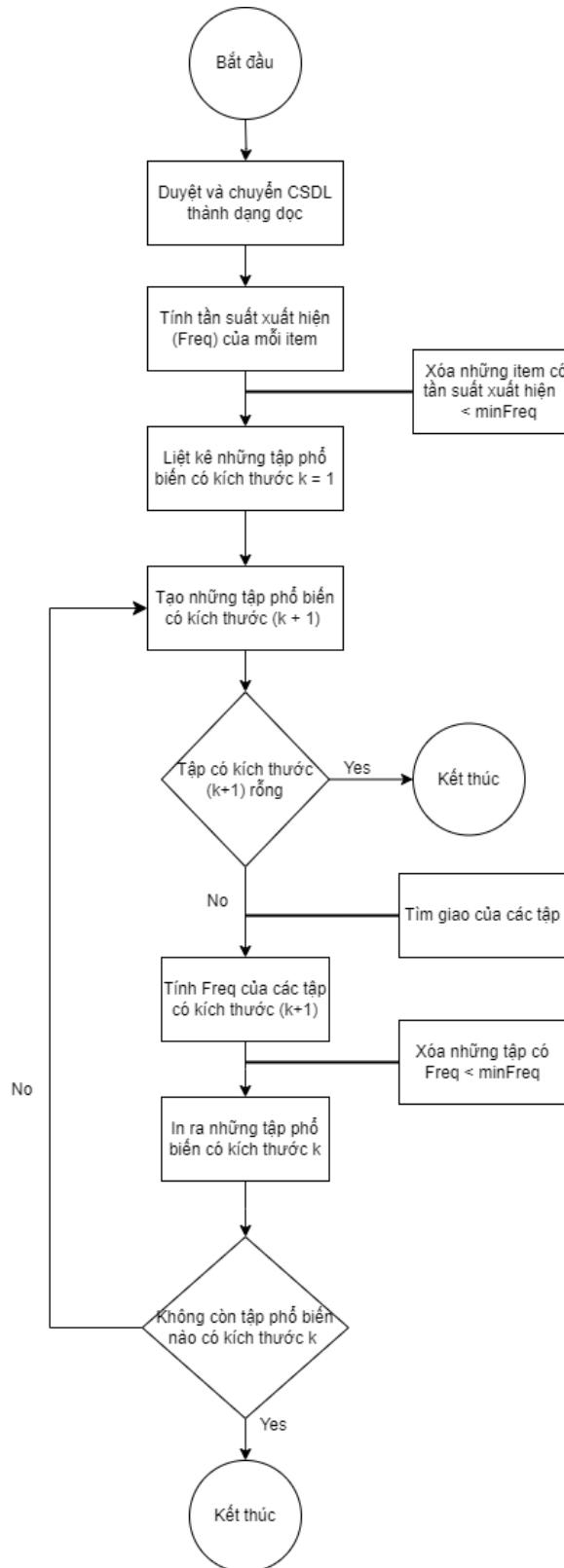
{kem, trứng}, {kem, táo} {kem, sữa},... với kích thước k = 2

- Total order: Là thứ tự sắp xếp của các itemset và các giao dịch, giúp đơn giản hóa quá trình xử lý và giúp đảm bảo tính ổn định của thuật toán, mặc dù không ảnh hưởng đến kết quả cuối cùng. Ví dụ, thứ tự sắp xếp có thể theo bảng chữ cái, hoặc sắp xếp theo thứ tự của các giao dịch.
- Equivalence class: Lớp tương đương. Giả sử có 2 itemsets là X và Y.  
Khi đó X và Y cùng thuộc một lớp tương đương nếu ( $k - 1$ ) phần tử đầu tiên của X và Y giống nhau (lưu ý sắp xếp theo total order).  
Ví dụ: kem và trứng cùng thuộc một lớp tương đương:

{kem, trứng, táo},

{kem, trứng, sữa}

**Sơ đồ thuật toán ECLAT:** (He Lan et al., 2022)



- Bước 1: Duyệt toàn bộ CSDL và tạo ra bảng dữ liệu mới theo dạng dọc (vertical database).
- Bước 2: Tính tần suất xuất hiện Freq của mỗi item. Xóa những item có Freq < minFreq.

- Bước 3: Liệt kê những tập phô biến có kích thước  $k = 1$ .  
Kết hợp các lớp tương đương từ các itemsets (giao của các tập) để tạo ra các lớp tương đương mới có kích thước  $k + 1$ .
- Bước 4: Tiếp tục vòng lặp đến khi không tìm thấy tập phô biến nào có kích thước  $(k + 1)$  thỏa mãn  $\text{Freq} < \text{minFreq}$  nữa.
- Bước 5: Kết thúc và in ra những tập phô biến thỏa điều kiện.

### 3.5 Ví dụ minh họa

Để mô tả rõ hơn về các bước trong sơ đồ thuật toán ECLAT, ta sẽ tìm hiểu thông qua ví dụ cụ thể như dưới đây:

Cho CSDL của các giao dịch trong một cửa hàng như sau:

Giao dịch	Các mặt hàng có trong giao dịch
T1	{kem, trứng, táo, sữa}
T2	{kem, trứng, táo, }
T3	{kem, sữa}
T4	{kem, trứng, sữa, kẹo}

- Bước 1: Duyệt toàn bộ CSDL và tạo ra bảng dữ liệu mới theo dạng dọc.



Giao dịch	Các mặt hàng có trong giao dịch
T1	{kem, trứng, táo, sữa}
T2	{kem, trứng, táo, }
T3	{kem, sữa}
T4	{kem, trứng, sữa, kẹo}

Mặt hàng	Giao dịch chứa các mặt hàng
kem	T1, T2, T3, T4
trứng	T1, T2, T4
táo	T1, T2
sữa	T1, T3, T4
kẹo	T4

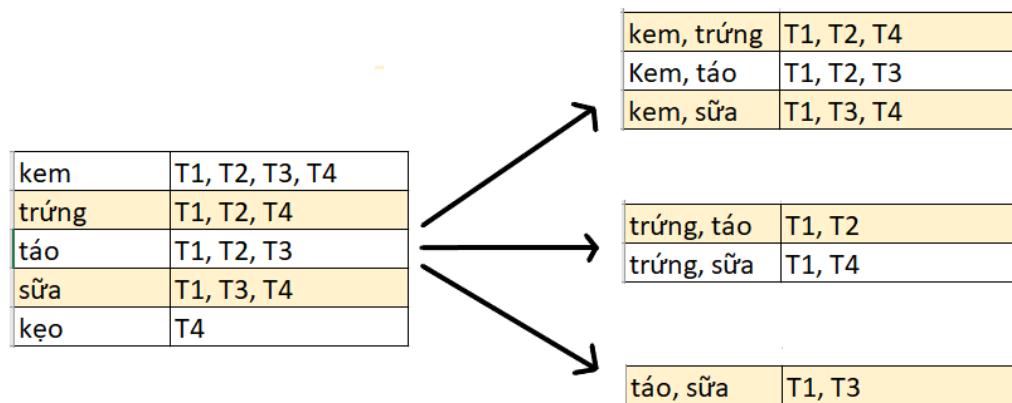
Sau khi chuyển về dạng dọc, mỗi hàng được gọi là TID-list (Transaction ID List).

- Bước 2: Thuật toán ECLAT sẽ loại bỏ các itemsets có số lần xuất hiện ít hơn giá trị  $\text{minFreq}$ .

Giả sử trong trường hợp này, với  $\text{minSup} = 30\% \Rightarrow \text{minFreq} = 30\% * 5 = 2$  lần. Như vậy, ECLAT sẽ loại bỏ những itemset có tần số xuất hiện < 2.

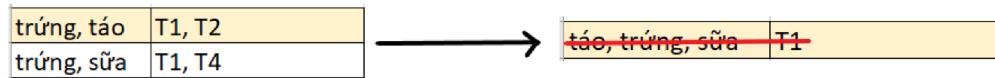
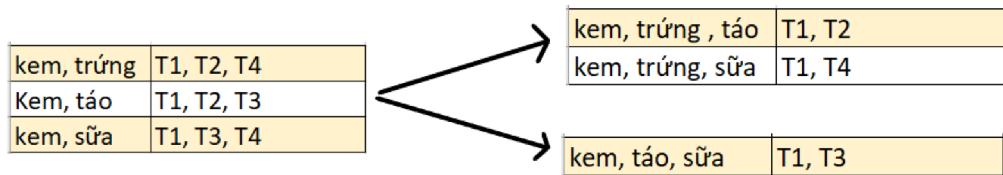
Mặt hàng	Giao dịch chứa các mặt hàng
kem	T1, T2, T3, T4
trứng	T1, T2, T4
táo	T1, T2
sữa	T1, T3, T4
kẹo	T4

- Bước 3: Kết hợp các lớp tương đương từ các itemsets để tạo ra các lớp tương đương mới có kích thước  $k + 1$ .



Với  $\text{minFreq} = 2$ , có 6 tập phổ biến được tạo ra: {kem, trứng}, {kem, táo}, {kem, sữa}, {trứng, táo}, {trứng, sữa}, {táo, sữa}

- Bước 4: Tiếp tục xử lý đệ quy các lớp tương đương như ở bước 3. Quá trình này lặp lại với  $k$  tăng thêm 1 mỗi vòng lặp, cho đến khi không còn tập phổ biến nào được tìm thấy.



Với minFreq =2, chỉ có 3 tập phô biến mới được tạo ra là {kem, trứng, táo}, {kem, trứng, sữa} và {kem, táo, sữa}

Như vậy, thuật toán ECLAT đã tìm được các tập phô biến thỏa minFreq = 2 như dưới đây:

{kem}	Freq = 4
{trứng}	Freq = 3
{táo}	Freq = 3
{sữa}	Freq = 3
{trứng, táo}	Freq = 2
{trứng, sữa}	Freq = 2
{táo, sữa}	Freq = 2
{kem, trứng}	Freq = 3
{kem, táo}	Freq = 3
{kem, sữa}	Freq = 3
{kem, trứng, táo}	Freq = 2
{kem, trứng sữa}	Freq = 2
{kem, táo, sữa}	Freq = 2

### 3.6 So sánh thuật toán Apriori, ECLAT và FP - Growth

Đặc điểm	Apriori	ECLAT	FP-Growth
<b>Phương pháp</b>	Tìm các tập phô biến bằng Breadth-First search và các thuộc tính của Apriori như: <ul style="list-style-type: none"> <li>- Mọi tập con khác <math>\emptyset</math> của tập phô biến X cũng là tập phô biến.</li> <li>- Mọi tập chứa X không phô biến cũng là tập không phô biến.</li> </ul>	Tìm các tập phô biến bằng Depth-First Search và giao của các T-id để tạo itemset ứng viên.	Sử dụng phương pháp chia để trị để khai thác các itemset phô biến.
<b>Số lần duyệt CSDL</b>	Mỗi lần tạo itemset, nếu có k tập phô biến ở cấp độ trước, Apriori sẽ thực hiện tối đa k lần duyệt.	<b>Một lần.</b>  (Trong trường hợp dữ liệu ở dạng ngang thì sẽ cần thêm một lần quét để chuyển dữ liệu từ dạng ngang sang dạng dọc).	<b>Hai lần.</b>  Lần 1 dùng phép chiếu để tìm tất cả các item phô biến và sắp xếp thứ tự giảm dần theo tần suất. Sau đó tiến hành duyệt CSDL lần 2 để xây dựng FP-tree.
<b>Thời gian thực thi</b>	Thời gian thực thi của thuật toán Apriori đáng kể vì nó phải quét lại cơ sở dữ liệu mỗi lần tạo ra một bộ ứng viên itemset.	<b>Ngắn hơn so với thuật toán Apriori.</b> Điều này là do ECLAT tính toán tần suất xuất hiện bằng cách sử dụng giao điểm giữa các tập mà không cần phải duyệt lại từ đầu, do đó tránh được việc duyệt CSDL nhiều lần dẫn đến tốn nhiều thời gian.	Thời gian thực thi của thuật toán FP-growth ngắn hơn so với thuật toán Apriori vì không cần phải quét lại toàn bộ cơ sở dữ liệu nhiều lần mà chỉ cần quét cơ sở dữ liệu một lần để xây dựng FP-Tree. Điều này giúp làm giảm đáng kể thời gian thực hiện.
<b>Định dạng dữ liệu</b>	Ngang	Dọc	Ngang

Cấu trúc dữ liệu	Mảng	Mảng	Cây (FP Tree)
<b>Ưu điểm</b>	<p><b>Tính toán các tập phổ biến lớn:</b> Thuật toán Apriori có khả năng xử lý và tìm ra những mẫu phổ biến trong tập dữ liệu lớn. Nói cách khác, nó giúp tìm các itemsets xuất hiện cùng nhau nhiều lần.</p> <p><b>Dễ hiểu và áp dụng:</b> Thuật toán khá đơn giản và dễ hiểu. Nó hoạt động bằng cách lặp đi lặp lại qua các cơ sở dữ liệu, mỗi lần tăng kích thước của itemsets và chỉ giữ lại những itemsets phổ biến. Điều này giúp thuật toán dễ dàng được áp dụng trong nhiều ngữ cảnh khai thác dữ liệu khác nhau.</p>	<p><b>Hiệu quả về bộ nhớ:</b> Việc sử dụng thuật toán Depth-First Search trong ECLAT giúp giảm thiểu yêu cầu về bộ nhớ bằng cách chỉ giữ thông tin cần thiết để thực hiện tìm kiếm hiệu quả, thay vì lưu trữ toàn bộ không gian tìm kiếm.</p> <p><b>Không cần quét lại cơ sở dữ liệu:</b> Eclat quét tập dữ liệu được tạo ra tại thời điểm hiện tại, loại bỏ sự cần thiết phải quét lại toàn bộ cơ sở dữ liệu cho mỗi lần tạo itemset mới.</p>	<p><b>Giảm số lượng tập ứng viên:</b> FP-Growth không cần phải tạo ra các tập ứng viên, mà chỉ sử dụng cấu trúc cây FP-Tree để lưu trữ thông tin về các mục phổ biến. Điều này giúp giảm đáng kể không gian tìm kiếm và chi phí tính toán.</p> <p><b>Giảm số lần quét cơ sở dữ liệu:</b> Đối với các tập dữ liệu lớn, FP-growth thường hiệu quả hơn Apriori, bởi vì không cần phải quét lại toàn bộ cơ sở dữ liệu nhiều lần mà chỉ cần quét cơ sở dữ liệu một lần để xây dựng FP-Tree. Điều này giúp làm giảm đáng kể thời gian thực hiện.</p> <p><b>Dễ triển khai và cài đặt:</b> FP-growth thường dễ triển khai và cài đặt, đặc biệt là khi sử dụng các cấu trúc dữ liệu như cây FP-Tree. Cấu trúc cây FP-Tree cũng là một cấu trúc dữ liệu</p>

			phổ biến và hiệu quả.
Nhược điểm	<p><b>Tốn kém về mặt tính toán:</b> Thuật toán cần quét toàn bộ cơ sở dữ liệu để tìm các mục hỗ trợ, tức là nó phải xem xét mỗi giao dịch trong cơ sở dữ liệu. Do đó, có thể gây tốn kém về mặt tính toán, đặc biệt là khi cơ sở dữ liệu lớn.</p> <p><b>Tốn kém về mặt bộ nhớ và thời gian xử lý:</b> Trong mỗi vòng lặp của thuật toán, nó tạo ra một tập ứng viên <math>C_k</math> mới, trong đó, <math>k</math> là số lượng item. Số lượng các tập ứng viên này có thể tăng lên rất nhanh khi <math>k</math> tăng, vậy nên dẫn đến việc tốn kém bộ nhớ và thời gian xử lý lâu.</p>	<p><b>Bộ T-id dài:</b> Eclat lưu trữ các T-id (transaction IDs) cho từng itemset, dẫn đến các bộ T-id có thể trở nên rất dài và tốn kém để thao tác, ảnh hưởng đến hiệu suất.</p> <p><b>Thuật toán Eclat không tận dụng toàn bộ thuộc tính của Apriori:</b> Một tính chất quan trọng trong việc giảm số lượng itemset ứng viên cần khám phá (Con của phổ biến là phổ biến, cha của không phổ biến thì cũng không phổ biến), dẫn đến việc khám phá nhiều hơn các itemset ứng viên không cần thiết, làm giảm hiệu quả.</p> <p><b>Thứ tự giải quyết lớp của Eclat là từ dưới lên trên:</b> Điều này có nghĩa là các lớp itemset nhỏ hơn sẽ được giải quyết trước các lớp itemset lớn hơn. Thứ tự này có thể hiệu quả trong một số trường hợp, chẳng hạn như khi các tập phổ biến có kích thước nhỏ. Tuy nhiên, thứ tự này có thể không tối</p>	<p><b>Cần phải xây dựng FP-Tree từ đầu khi thêm dữ liệu mới:</b> Khi có dữ liệu mới được thêm vào, cấu trúc cây FP-Tree có thể bị thay đổi, do đó cần phải xây dựng lại cây FP-Tree từ đầu. Điều này gây tốn thời gian và bộ nhớ.</p> <p><b>Không thích hợp cho việc xử lý dữ liệu có tần suất thay đổi:</b> Nếu dữ liệu thay đổi thường xuyên, ví dụ như trong các tập dữ liệu về thời tiết hay là giao thông, thuật toán FP-Growth có thể không phản ánh được các mối quan hệ thực tế giữa các item. Việc xây dựng lại FP-Tree có thể trở nên tốn kém và không hiệu quả.</p> <p><b>Chiếm nhiều bộ nhớ:</b> Thuật toán FP-Growth mặc dù có thể tiết kiệm bộ nhớ hơn Apriori do không cần tạo ra các tập ứng viên. Tuy nhiên, trong một số trường hợp, FP-Tree cũng có thể trở nên phức tạp, đặc biệt là khi xử lý các tập dữ</p>

		ưu trong những trường hợp khác, chẳng hạn như khi các tập phỏ biến có kích thước lớn hoặc khi có nhiều itemset cần so sánh.	liệu lớn.
--	--	---	-----------

Nguồn: (Shamila Nasreen et al., 2014), (He Lan et al., 2022), (Yi Zeng et al., 2015)

## **CHƯƠNG 4: XÂY DỰNG MÔ HÌNH**

#### 4.1 Tổng quan về bộ dữ liệu

Nhóm sử dụng bộ dữ liệu “Market Basket Analysis 5”, đây là một bộ dữ liệu lưu trữ những thông tin giao dịch của một cửa hàng từ 01/01/2000 đến ngày 26/02/2002 với tổng cộng 1499 giao dịch (giả định rằng các dòng riêng biệt trong file đại diện cho các giao dịch riêng biệt trong cửa hàng cho dù có nhiều giao dịch xảy ra trong cùng 1 ngày). Hình bên dưới thể hiện cách dữ liệu được lưu trữ trong file.

1/1/2000yogurt, pork, sandwich bags, lunch meat, all-purpose, flour, soda, butter, vegetables, beef, aluminum foil, all-purpose, dinner rolls, shampoo, all-purpose, mixes, soap, laundry detergent, ice cream, dinner rolls, 1/1/2001paper, paper, shampoo, hand soap, waffles, vegetables, cheeses, mixes, milk, sandwich bags, laundry detergent, dishwashing liquid/detergent, waffles, individual meals, hand soap, vegetables, individual meals, yogurt, cereals, shampoo, vegetables, a 1/2/2000soda, por, soap, ice cream, toilet paper, dinner rolls, hand soap, spaghetti sauce, milk, ketchup, sandwich loaves, poultry, toilet paper, ice cream, ketchup, vegetables, laundry detergent, spaghetti sauce, bagels, soap, ice cream, shampoo, lunch me 2/1/2000cereals, juice, lunch meat, soda, toilet paper, all-purpose, 2/1/2000sandwich loaves, pasta, tortillas, mixes, hand soap, toilet paper, vegetables, vegetables, paper towels, vegetables, flour, vegetables, pork, poultry, eggs, vegetables, pork, spaghetti sauce, vegetables, milk, waffles, individual meals, vegetables, dinner 2/1/2000laundry detergent, toilet paper, eggs, toilet paper, vegetables, bagels, dishwashing liquid/detergent, detergents, cereals, paper towels, laundry detergent, butter, cereals, bagels, paper towels, shampoo, toilet paper, soap, soap, pasta, coffee/tea, poultry, bagels 3/1/2000individual meals, paper towels, tortillas, vegetables, milk, ice cream, juice, dishwashing liquid/detergent, soap, sandwich bags, pasta, ketchup, all-purpose, yogurt, mixes, mixes, toilet paper, vegetables, beef, sandwich bags, eggs, spaghetti sauce, fr 4/1/2000cream, juice, paper towels, waffles, soda, cheeses, poultry, toilet paper, vegetables,

## 4.2 Tiết xử lý dữ liệu

Có thể thấy rằng dữ liệu nằm trong file đang có định dạng chưa được ẩn (như là có chuỗi thời gian nằm cạnh sản phẩm đầu tiên, có dấu ‘,’ ở cuối mỗi transaction) để có thể cho vào thuật toán. Do đó mà ta cần phải tiến hành tiền xử lý dữ liệu để có thể đưa dữ liệu vào thuật toán ECLAT. Hình dưới đây là kiểu dataframe mà ta nhắm tới sau khi tiền xử lý:

Cụ thể các bước tiền xử lý sẽ như sau:

- Nhóm đọc dữ liệu từ file csv vào trong dataframe của pandas:

```
1 # Đọc DataFrame
2 df = pd.read_csv('Market Basket Analysis 5.csv', sep = 'delimiter', header = None)
3 df
```

		0	
0	1/1/2000yogurt, pork, sandwich bags, lunch meat, all...		
1	1/1/2000toilet paper, shampoo, hand soap, waffles, veg...		
2	2/1/2000soda, pork, soap, ice cream, toilet paper, din...		
3	2/1/2000cereals, juice, lunch meat, soda, toiletries, ...		
4	2/1/2000sandwich loaves, pasta, tortillas, mixes, hand...		
...	...		
1494	22/2/2002sugar, beef, sandwich bags, hand soap, paper t...		
1495	23/2/2002coffee/tea, dinner rolls, lunch meat, spaghetti...		
1496	24/2/2002beef, lunch meat, eggs, poultry, vegetables, t...		
1497	25/2/2002sandwich bags, ketchup, milk, poultry, cheeses...		
1498	26/2/2002soda, laundry detergent, vegetables, shampoo, ...		
1499 rows × 1 columns			

- Nhóm xử lý chuỗi thời gian ở đầu mỗi row:

```
1 # Xử lý chuỗi thời gian ở đầu mỗi row
2 df[df.columns[0]] = df[df.columns[0]].str.replace(r'\d{1,2}/\d{1,2}/\d{4}', '', regex=True)
```

Và thu được kết quả như bảng sau:

		0	
0	yogurt, pork, sandwich bags, lunch meat, all...		
1	toilet paper, shampoo, hand soap, waffles, vegetables, t...		
2	soda, pork, soap, ice cream, toilet paper, dinner rolls, ...		
3	cereals, juice, lunch meat, soda, toilet paper, ...		
4	sandwich loaves, pasta, tortillas, mixes, hand...		
...	...		
1494	sugar, beef, sandwich bags, hand soap, paper t...		
1495	coffee/tea, dinner rolls, lunch meat, spaghetti, ...		
1496	beef, lunch meat, eggs, poultry, vegetables, t...		
1497	sandwich bags, ketchup, milk, poultry, cheeses...		
1498	soda, laundry detergent, vegetables, shampoo, ...		
1499 rows × 1 columns			

3. Nhóm xử lý các khoảng trắng và tách chuỗi trong danh sách để ra được dataframe như hình:

Xử lý các khoảng trắng																																
df[df.columns[0]] = df[df.columns[0]].str.strip(' ,')																																
Tách chuỗi sang danh sách																																
df = df[df.columns[0]].str.split(' , ', expand=True)																																
df																																
0	yogurt	pork	sandwich bags	lunch meat	all-purpose	flour	soda	butter	vegetables	beef	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
1	toilet paper	shampoo	hand soap	waffles	vegetables	cheeses	mixes	milk	sandwich bags	laundry detergent	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
2	soda	pork	soap	ice cream	toilet paper	dinner rolls	hand soap	spaghetti sauce	milk	ketchup	...	spaghetti sauce	pork	vegetables	cheeses	eggs	vegetables	vegetables	None													
3	cereals	juice	lunch meat	soda	toilet paper	all-purpose	None	None	None	None	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
4	sandwich loaves	pasta	tortillas	mixes	hand soap	toilet paper	vegetables	vegetables	paper towels	vegetables	...	all-purpose	soda	yogurt	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1494	sugar	beer	sandwich bags	hand soap	paper towels	paper towels	all-purpose	beef	fruits	coffee/tea	...	beef	cereals	juice	poultry	sugar	soap	None	None	None	None	None	None	None	None	None	None	None	None	None	None	
1495	coffee/tea	dinner rolls	lunch meat	spaghetti sauce	pasta	vegetables	cereals	dinner rolls	soap	milk	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
1496	beef	lunch meat	eggs	poultry	vegetables	tortillas	beef	beef	individual meals	dishwashing liquid/detergent	...	vegetables	pork	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
1497	sandwich bags	ketchup	milk	poultry	cheeses	soap	toilet paper	yogurt	beef	waffles	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
1498	soda	laundry detergent	vegetables	shampoo	vegetables	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	
1499 rows × 34 columns																																

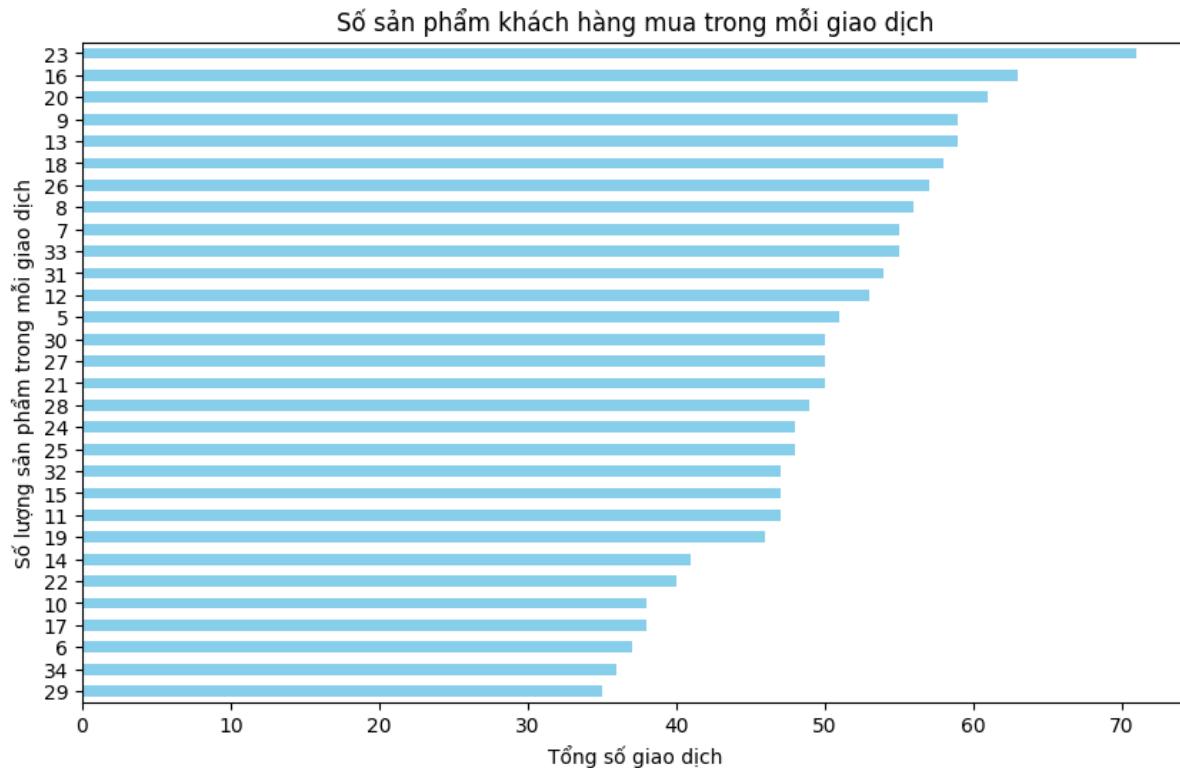
4. Tuy nhiên nhóm thấy rằng trong dataframe như trên vẫn còn xuất hiện việc 1 sản phẩm xuất hiện nhiều lần trong cùng 1 giao dịch, do đó nhóm cần phải loại bỏ bớt các item xuất hiện nhiều lần trong cùng 1 giao dịch đi (làm việc này cũng để thuận tiện hơn khi so sánh với 2 thuật toán khác là Apriori và FP-Growth). Ở phần này ta sẽ làm 2 bước chính đó là chuyển dataframe các giao dịch này thành một danh sách lớn chứa các danh sách nhỏ hơn với các danh sách nhỏ hơn đại diện cho mỗi giao dịch, và các sản phẩm xuất hiện nhiều hơn 1 lần sẽ được xử lí để xuất hiện chỉ 1 lần trong các danh sách nhỏ này (danh sách lớn này sẽ được dùng để áp dụng với 2 thuật toán Apriori và FP-Growth như đề cập ở trên), bước thứ 2 (để phục vụ cho Eclat) đó là chuyển danh sách lớn này về ngược lại dạng dataframe. Lúc này ta sẽ có dataframe như hình sau:

Xử lý các khoảng trắng																																	
df[df.columns[0]] = df[df.columns[0]].str.strip(' ,')																																	
Tách chuỗi sang danh sách																																	
df = df[df.columns[0]].str.split(' , ', expand=True)																																	
0	all-purpose	dinner rolls	soap	lunch meat	sandwich bags	yogurt	mixes	shampoo	laundry detergent	ice cream	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
1	milk	cheeses	mixes	sandwich bags	hand soap	yogurt	dishwashing liquid/detergent	shampoo	laundry detergent	cereals	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
2	soda	dinner rolls	bagels	lunch meat	sandwich loaves	hand soap	ketchup	vegetables	milk	spaghetti sauce	...	soap	pork	poultry	None																		
3	all-purpose	lunch meat	cereals	soda	juice	toilet paper	None	None	None	None	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
4	all-purpose	paper towels	soda	individual meals	dinner rolls	pasta	sandwich loaves	hand soap	yogurt	tortillas	...	flour	pork	poultry	None																		
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1494	all-purpose	paper towels	sandwich bags	dishwashing liquid/detergent	juice	dinner rolls	hand soap	cheeses	sugar	shampoo	...	coffee/tea	fruits	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
1495	milk	spaghetti sauce	dinner rolls	soap	eggs	lunch meat	pasta	cheeses	mixes	hand soap	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
1496	dinner rolls	bagels	eggs	lunch meat	sugar	hand soap	paper towels	soap	yogurt	dishwashing liquid/detergent	...	vegetables	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
1497	milk	cheeses	spaghetti sauce	soap	all-purpose	sandwich bags	sugar	yogurt	paper towels	soda	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None		
1498	vegetables	soda	laundry detergent	shampoo	None	None	None	None	None	None	...	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	
1499 rows × 27 columns																																	

Và như vậy là ta đã có một dataframe đã tiền xử lí và có thể áp dụng vào thuật toán ECLAT.

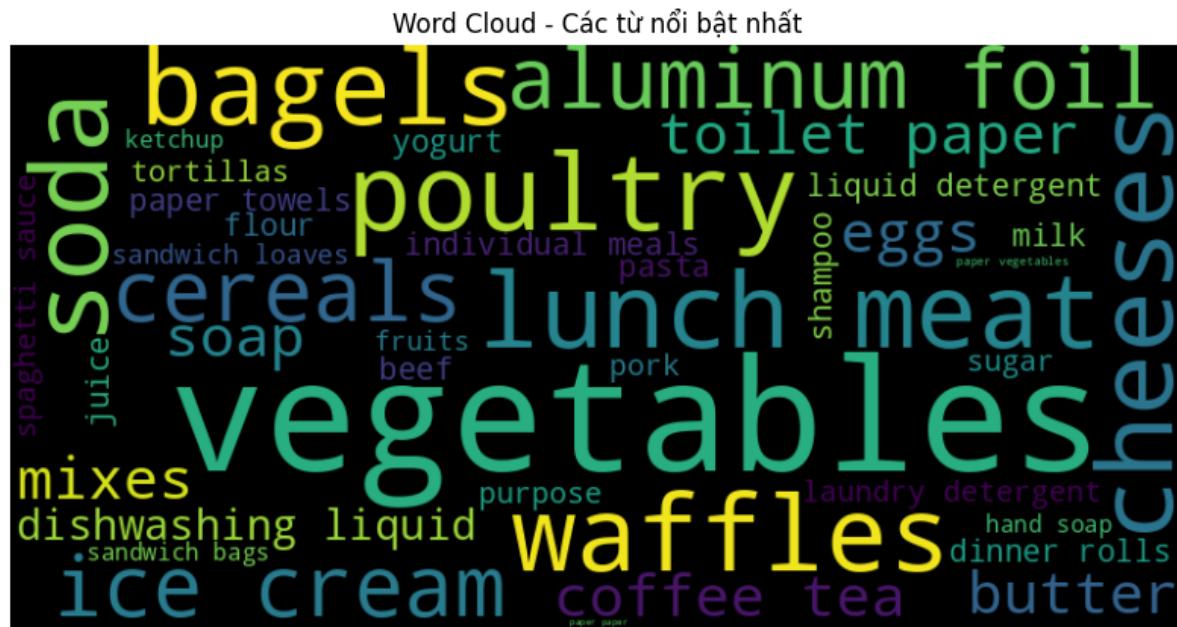
### 4.3 Trực quan dữ liệu

Ở đây, ta sẽ trực quan hóa dữ liệu để tìm kiếm những thông tin hữu ích:



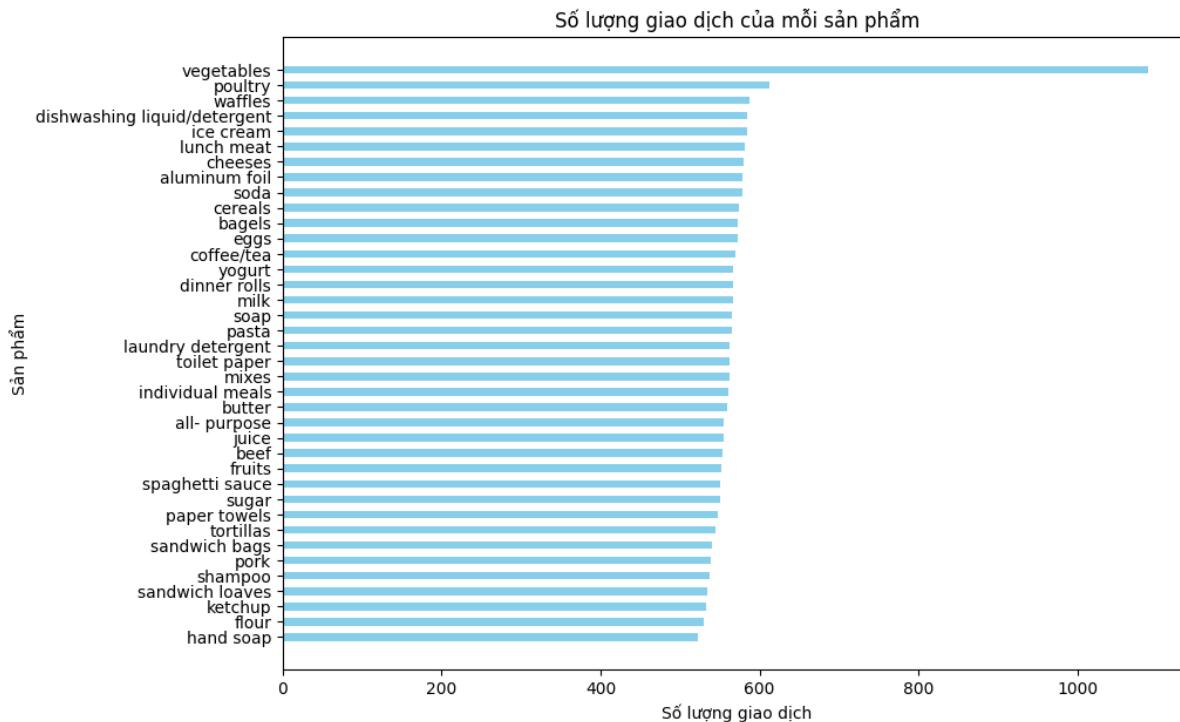
Các giao dịch có số lượng sản phẩm trong mỗi giao dịch dao động từ 5 tới nhiều nhất là 34 sản phẩm, và trong 1 giao dịch sẽ có trường hợp có vài sản phẩm được mua đi mua lại nhiều lần.

Ngoài ra ta cũng có thể biết sản phẩm nào đang chiếm nhiều trong các giao dịch của khách hàng bằng hình vẽ wordcloud như dưới đây:



Có thể thấy rằng sản phẩm nổi bật nhất là các loại rau, sau đó là gia cầm, bánh vòng... Và từ đây ta cũng có thể có những suy đoán là các luật kết hợp của chúng ta có khả năng là sẽ xoay quanh các sản phẩm này, tuy nhiên để biết có thực sự như vậy không thì ta cần phải tìm ra luật kết hợp của các sản phẩm bằng cách thuật toán và đánh giá xem luật kết hợp đó là có đáng tin cậy hay không, từ đó ta mới có những chiến lược phù hợp khi ta đưa ra những chiến lược giá...

Và để tìm hiểu sâu hơn về các sản phẩm đang được bán trên thị trường, ta cũng có thể biểu hiện nó bằng hình vẽ sau:



Và như những gì đã đề cập ở hình vẽ wordcloud, ta thấy được rằng sản phẩm rau xanh là mặt hàng rất phổ biến và hầu như xuất hiện tới khoảng 70% trong tổng số các giao dịch được ghi nhận. Còn những sản phẩm khác ngoài rau xanh thì lại có sự chênh lệch không đáng kể với nhau. Và do mặt hàng rau xanh có tần suất xuất hiện rất cao trong bộ dữ liệu nên chúng ta cần phải có những thước đo phù hợp để đánh giá tập luật.

#### 4.4 Cài đặt thuật toán

Việc cài đặt thuật toán khá đơn giản khi ta có những thư viện để làm việc đó. Và với thuật toán Eclat, nhóm sử dụng thư viện pyECLAT để đưa ra các tập kết hợp và sử dụng hàm association\_rules của mlxtend để in ra các thước đo đánh giá của tập luật. Việc cài đặt thuật toán diễn ra như sau:

Đầu tiên ta cần import những hàm cần thiết:

```
9 from pyECLAT import ECLAT
10 from mlxtend.preprocessing import TransactionEncoder
11 from mlxtend.frequent_patterns import apriori
12 from mlxtend.frequent_patterns import fpgrowth
13 from mlxtend.frequent_patterns import association_rules
```

Sau đó cho khởi tạo và cho dữ liệu vào thuật toán:

```
# Tạo instance eclat
eclat = ECLAT(data=data, verbose=True)

# Chạy thuật toán eclat
start = time.time()

ECLAT_indexes, ECLAT_supports = eclat.fit(min_support=0.2,
                                             min_combination=1,
                                             max_combination=3,
                                             separator=' & ',
                                             verbose=True)

end = time.time()
time_eclat = end - start

100%|██████████| 39/39 [00:00<00:00, 69.83it/s]
100%|██████████| 39/39 [00:00<00:00, 3575.63it/s]
100%|██████████| 39/39 [00:00<00:00, 1543.70it/s]
Combination 1 by 1
38it [00:00, 117.47it/s]
Combination 2 by 2
703it [00:03, 199.54it/s]
Combination 3 by 3
8436it [00:46, 179.73it/s]
```

```
frequent_itemsets_eclat = pd.DataFrame(ECLAT_supports.items(), columns=['itemsets', 'support'])
```

Nhóm sử dụng minSup = 20% và quan tâm tới các tổ hợp chứa từ 1 đến 3 phần tử trong 1 tập. Có thể thấy rằng với một giá trị minSup là 20% thì ta sẽ có 38 tổ hợp đạt yêu cầu tập chứa 1 sản phẩm, 703 tổ hợp đạt yêu cầu tập chứa 2 sản phẩm và 8436 tổ hợp đạt yêu cầu tập chứa 3 sản phẩm. Có thể thấy rằng nếu số lượng tổ hợp chứa 1 phần tử đạt yêu cầu tăng, sẽ dẫn đến số tổ hợp chứa 2 và 3 sản phẩm cũng tăng lên, do đó làm tăng thời gian chạy của thuật toán.

Sau đó thực hiện biến đổi thêm dataframe df một chút để có thể đưa vào hàm association\_rules để đưa tính toán các thước đo đánh giá các luật kết hợp. Cuối cùng ta được bảng như sau:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
57	(vegetables)	(yogurt)	0.726484	0.378252	0.308205	0.424242	1.121586	0.033411	1.079878	0.396341
56	(yogurt)	(vegetables)	0.378252	0.726484	0.308205	0.814815	1.121586	0.033411	1.476985	0.174356
53	(vegetables)	(eggs)	0.726484	0.381588	0.310874	0.427916	1.121408	0.033656	1.080981	0.395823
52	(eggs)	(vegetables)	0.381588	0.726484	0.310874	0.814685	1.121408	0.033656	1.475953	0.175067
21	(vegetables)	(laundry detergent)	0.726484	0.375584	0.304203	0.418733	1.114885	0.031347	1.074233	0.376749
...	...	...	...	...	...	...	...	...	...	...
61	(tortillas)	(vegetables)	0.363576	0.726484	0.275517	0.757798	1.043103	0.011385	1.129288	0.064929
9	(vegetables)	(pork)	0.726484	0.359573	0.267512	0.368228	1.024069	0.006287	1.013699	0.085932
8	(pork)	(vegetables)	0.359573	0.726484	0.267512	0.743970	1.024069	0.006287	1.068297	0.036700
35	(vegetables)	(coffee/tea)	0.726484	0.379586	0.282188	0.388430	1.023297	0.006425	1.014460	0.083238
34	(coffee/tea)	(vegetables)	0.379586	0.726484	0.282188	0.743409	1.023297	0.006425	1.065962	0.036696

Và như đã đề cập ở trên về các thước đo sẽ sử dụng để đánh giá tập luật, ở đây ta sẽ chỉ tập trung vào 3 thước đo chính là ‘support’, ‘confidence’, ‘lift’. Và cụ thể hơn thì ta sẽ tập trung vào những tập luật mà có chỉ số lift cao bởi vì ta thấy rằng sản phẩm rau xanh đang có quá nhiều, như vậy thì tập luật từ các sản phẩm bát kì dẫu tới rau xanh liệu có còn đáng quan tâm không? Như được đề cập ở trên, lift được sử dụng để đánh giá mẫu phổ biến trong khai phá luật kết hợp, cụ thể hơn thì lift là một công cụ tính hữu dụng để cho chúng ta biết liệu một luật kết hợp có đáng quan tâm.

Ở trong dataframe kết quả, ta có thể thấy rằng giá trị lift của các tập luật đều lớn hơn 1, thể hiện sự tương quan thuận giữa các vật phẩm đang xét. Do đó, ta tạm thời yên tâm về việc những tập luật mà ta phát triển là có ý nghĩa. Vì thế mà ở chương sau ta sẽ xem xét nên chọn tập luật nào để phục vụ cho bài toán thực tế giúp cửa hàng bán được nhiều sản phẩm hơn.

## CHƯƠNG 5: KẾT QUẢ VÀ ĐÁNH GIÁ

### 5.1 So sánh thời gian chạy của các thuật toán

*So sánh với Apriori và FP-Growth có cùng minSup:*

Để có thể áp dụng được cả hai thuật toán Apriori và FP-Growth thì dataframe phải chuyển đổi các dữ liệu từ dạng danh sách các giao dịch thành một ma trận nhị phân (DataFrame) mà trong đó mỗi hàng tương ứng với một giao dịch và mỗi cột tương ứng với một mặt hàng. Trong ma trận này, giá trị 1 (hoặc True) biểu thị rằng mặt hàng đó xuất hiện trong giao dịch, và giá trị 0 (hoặc False) biểu thị ngược lại.

- Khởi tạo Apriori và FP-Growth với cùng minSup và so sánh thời gian thực thi:

```
start = time.time()

frequent_itemsets_apriori = apriori(df_exp, min_support=0.2, use_colnames = True)

end = time.time()
time_apriori = end - start
print("Thời gian chạy Apriori: ", time_apriori)
```

Thời gian chạy Apriori: 0.014746665954589844

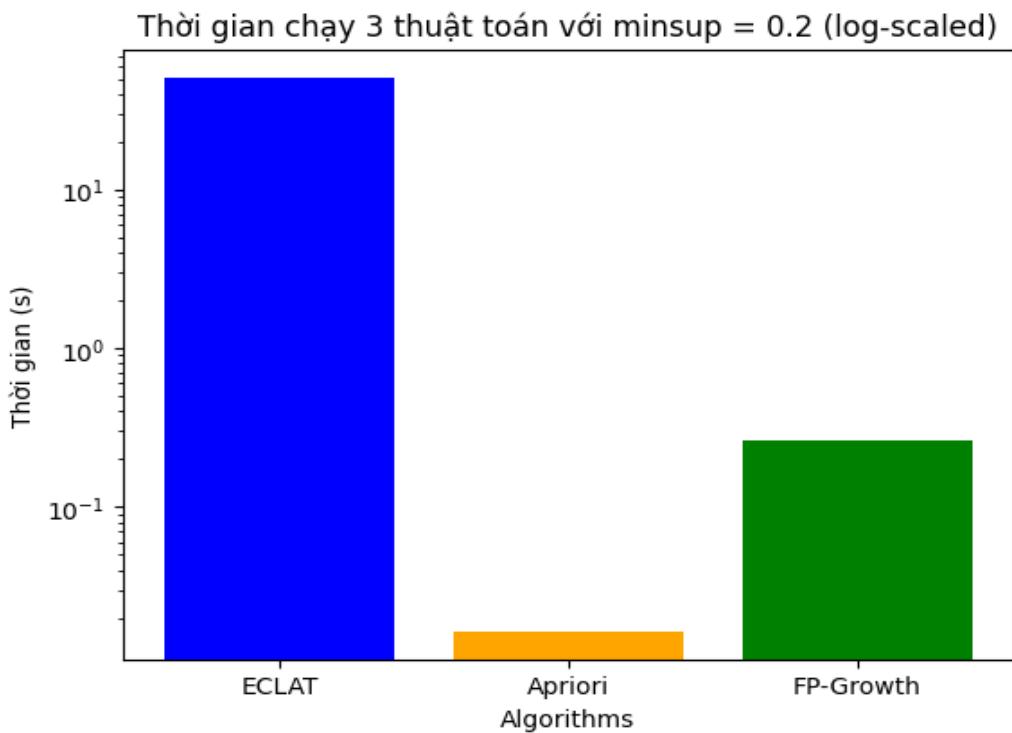
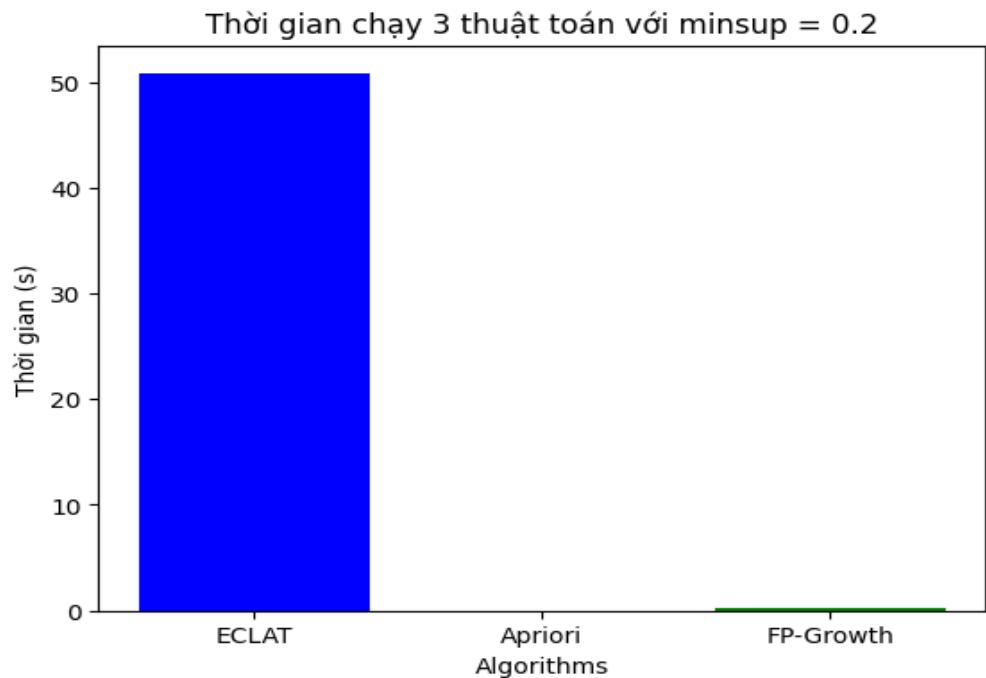
```
start = time.time()

frequent_itemsets_fpg = fpgrowth(df_exp, min_support=0.2, use_colnames=True)

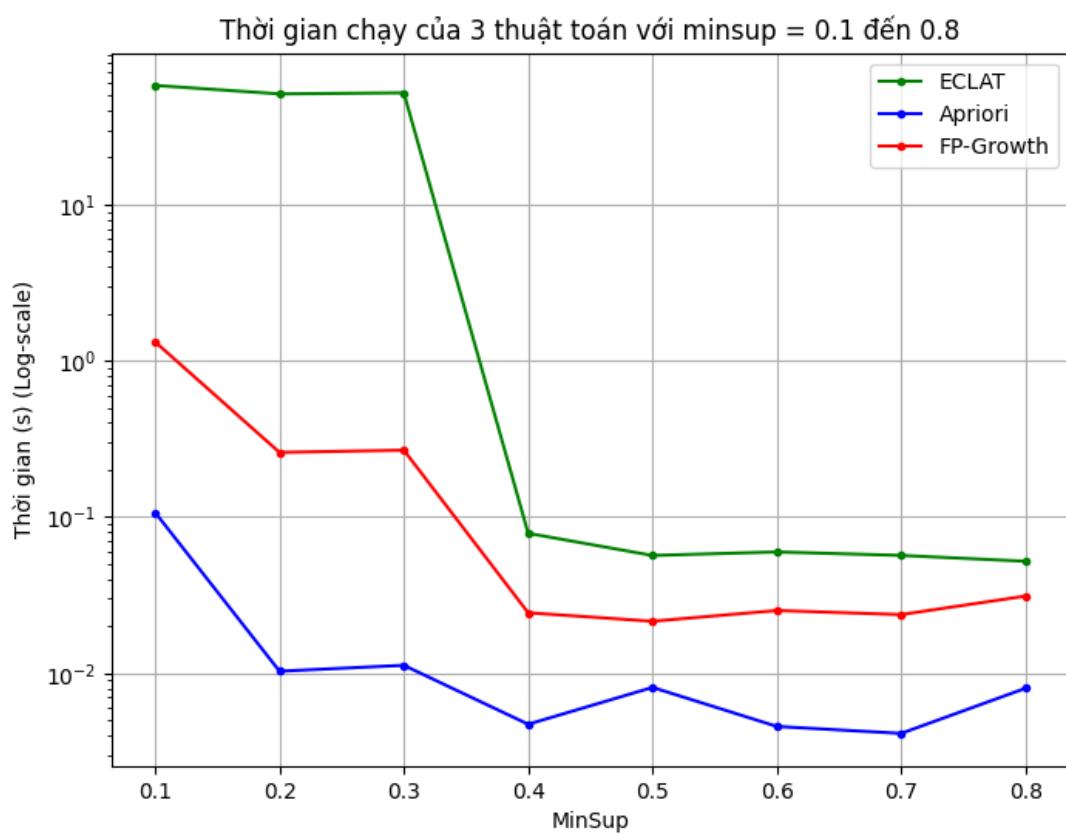
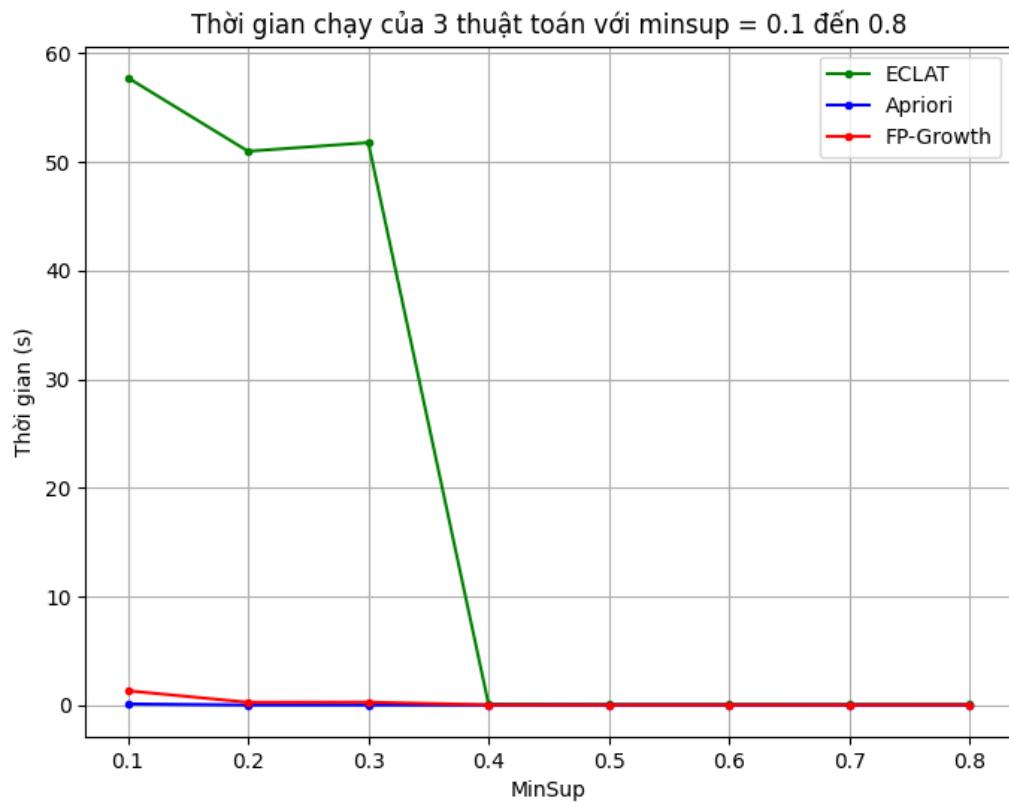
end = time.time()
time_fpg = end - start
print("Thời gian chạy FP-growth: ", time_fpg)
```

Thời gian chạy FP-growth: 0.3475456237792969

- Biểu diễn trực quan thời gian chạy với minsup = 0.2:



- Biểu diễn trực quan thời gian chạy với từ minsup = 0.1 đến 0.8:



## Nhận xét:

Ở biểu đồ thời gian chạy thuật toán thì ECLAT có thời gian chạy cao nhất và cao hơn đáng kể so với 2 thuật toán còn lại. Hai thuật toán Apriori và FP-Growth còn lại có sự chênh lệch thời gian chạy rất thấp cho thấy kết quả gần như tương đồng nhau và có thời gian tìm ra những luật kết hợp rất nhanh.

Ở biểu đồ log-scaled ta thấy minsup từ 0.1 đến 0.2 đã có sự chênh lệch khoảng 1s giữ FP-Growth và Apriori. Điều này có thể do khi định nghĩa minsup sẽ có thể gây ảnh hưởng thời gian tìm các tập phổ biến ở hai thuật toán này, mặc dù không nhiều.

Đối với ECLAT, đây là thuật toán được xây dựng bởi thư viện pyECLAT và khác với hai thuật toán kia có thư viện mlxtend nên có thể sẽ khác ở cấu trúc xây dựng vì Apriori và FP-Growth cần chuyển đổi dữ liệu từ dạng danh sách các giao dịch thành một ma trận nhị phân (DataFrame) mà trong đó mỗi hàng tương ứng với một giao dịch và mỗi cột tương ứng với một mặt hàng như đã nói bên trên. Với ECLAT thì thuật toán sẽ duyệt qua dạng TransactionID set và với minsup càng thấp thì sẽ có nhiều luật được duyệt và có thể làm tăng bộ nhớ, cụ thể là các list transaction làm ảnh hưởng đến thời gian chạy của ECLAT.

## 5.2 So sánh tập luật của các thuật toán:

```
print("Tổng các tập luật giữa apriori và ECLAT là bằng nhau: ",
      bool(len(frequent_itemsets_apriori) == len(frequent_itemsets_eclat)))
print("Tổng các tập luật giữa FP-Growth và ECLAT là bằng nhau: ",
      bool(len(frequent_itemsets_fpg) == len(frequent_itemsets_eclat)))
```

Tổng các tập luật giữa apriori và ECLAT là bằng nhau: True  
 Tổng các tập luật giữa FP-Growth và ECLAT là bằng nhau: True

### ECLAT:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
37	(vegetables)	(yogurt)	0.726484	0.378252	0.308205	0.424242	1.121586	0.033411	1.079878	0.396341
36	(yogurt)	(vegetables)	0.378252	0.726484	0.308205	0.814815	1.121586	0.033411	1.476985	0.174356
1	(vegetables)	(eggs)	0.726484	0.381588	0.310874	0.427916	1.121408	0.033656	1.080981	0.395823
0	(eggs)	(vegetables)	0.381588	0.726484	0.310874	0.814685	1.121408	0.033656	1.475953	0.175067
65	(vegetables)	(laundry detergent)	0.726484	0.375584	0.304203	0.418733	1.114885	0.031347	1.074233	0.376749
...	...	...	...	...	...	...	...	...	...	...
69	(vegetables)	(tortillas)	0.726484	0.363576	0.275517	0.379247	1.043103	0.011385	1.025246	0.151078
49	(vegetables)	(pork)	0.726484	0.359573	0.267512	0.368228	1.024069	0.006287	1.013699	0.085932
48	(pork)	(vegetables)	0.359573	0.726484	0.267512	0.743970	1.024069	0.006287	1.068297	0.036700
43	(vegetables)	(coffee/tea)	0.726484	0.379586	0.282188	0.388430	1.023297	0.006425	1.014460	0.083238
42	(coffee/tea)	(vegetables)	0.379586	0.726484	0.282188	0.743409	1.023297	0.006425	1.065962	0.036696

74 rows × 10 columns

### *Apriori:*

rules_apriori = association_rules(frequent_itemsets_apriori, metric="support", min_threshold=0.2).sort_values(by = 'lift', ascending=False)										
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
73	(vegetables)	(yogurt)	0.726484	0.378252	0.308205	0.424242	1.121586	0.033411	1.079878	0.396341
72	(yogurt)	(vegetables)	0.378252	0.726484	0.308205	0.814815	1.121586	0.033411	1.476985	0.174356
20	(eggs)	(vegetables)	0.381588	0.726484	0.310874	0.814685	1.121408	0.033656	1.475953	0.175067
21	(vegetables)	(eggs)	0.726484	0.381588	0.310874	0.427916	1.121408	0.033656	1.080981	0.395823
36	(vegetables)	(laundry detergent)	0.726484	0.375584	0.304203	0.418733	1.114885	0.031347	1.074233	0.376749
...	...	...	...	...	...	...	...	...	...	...
69	(vegetables)	(tortillas)	0.726484	0.363576	0.275517	0.379247	1.043103	0.011385	1.025246	0.151078
49	(vegetables)	(pork)	0.726484	0.359573	0.267512	0.368228	1.024069	0.006287	1.013699	0.085932
48	(pork)	(vegetables)	0.359573	0.726484	0.267512	0.743970	1.024069	0.006287	1.068297	0.036700
15	(vegetables)	(coffee/tea)	0.726484	0.379586	0.282188	0.388430	1.023297	0.006425	1.014460	0.083238
14	(coffee/tea)	(vegetables)	0.379586	0.726484	0.282188	0.743409	1.023297	0.006425	1.065962	0.036696

74 rows × 10 columns

### *FP-Growth:*

rules_fpg = association_rules(frequent_itemsets_fpg, metric="support", min_threshold=0.2).sort_values(by = 'lift', ascending=False)										
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
8	(yogurt)	(vegetables)	0.378252	0.726484	0.308205	0.814815	1.121586	0.033411	1.476985	0.174356
9	(vegetables)	(yogurt)	0.726484	0.378252	0.308205	0.424242	1.121586	0.033411	1.079878	0.396341
55	(vegetables)	(eggs)	0.726484	0.381588	0.310874	0.427916	1.121408	0.033656	1.080981	0.395823
54	(eggs)	(vegetables)	0.381588	0.726484	0.310874	0.814685	1.121408	0.033656	1.475953	0.175067
15	(laundry detergent)	(vegetables)	0.375584	0.726484	0.304203	0.809947	1.114885	0.031347	1.439153	0.165029
...	...	...	...	...	...	...	...	...	...	...
47	(vegetables)	(tortillas)	0.726484	0.363576	0.275517	0.379247	1.043103	0.011385	1.025246	0.151078
26	(pork)	(vegetables)	0.359573	0.726484	0.267512	0.743970	1.024069	0.006287	1.068297	0.036700
27	(vegetables)	(pork)	0.726484	0.359573	0.267512	0.368228	1.024069	0.006287	1.013699	0.085932
69	(vegetables)	(coffee/tea)	0.726484	0.379586	0.282188	0.388430	1.023297	0.006425	1.014460	0.083238
68	(coffee/tea)	(vegetables)	0.379586	0.726484	0.282188	0.743409	1.023297	0.006425	1.065962	0.036696

74 rows × 10 columns

Tuy sử dụng 3 thuật toán khác nhau nhưng ta lại thu về cùng một danh sách kết quả về các luật có thể tạo thành với minSup bằng 0.2. Tuy nhiên ta thấy rằng có sự chênh lệch về thời gian khá nhiều giữa thuật toán ECLAT so với Apriori và FP-Growth nên ta cũng nên cân nhắc khi đưa thuật toán nào vào sử dụng trong thực tế. Và đồng thời, với danh sách các luật là như nhau, ta có thể tin rằng thuật toán của ta đang chạy đúng.

### 5.3 Chọn và trực quan tập luật đáng quan tâm

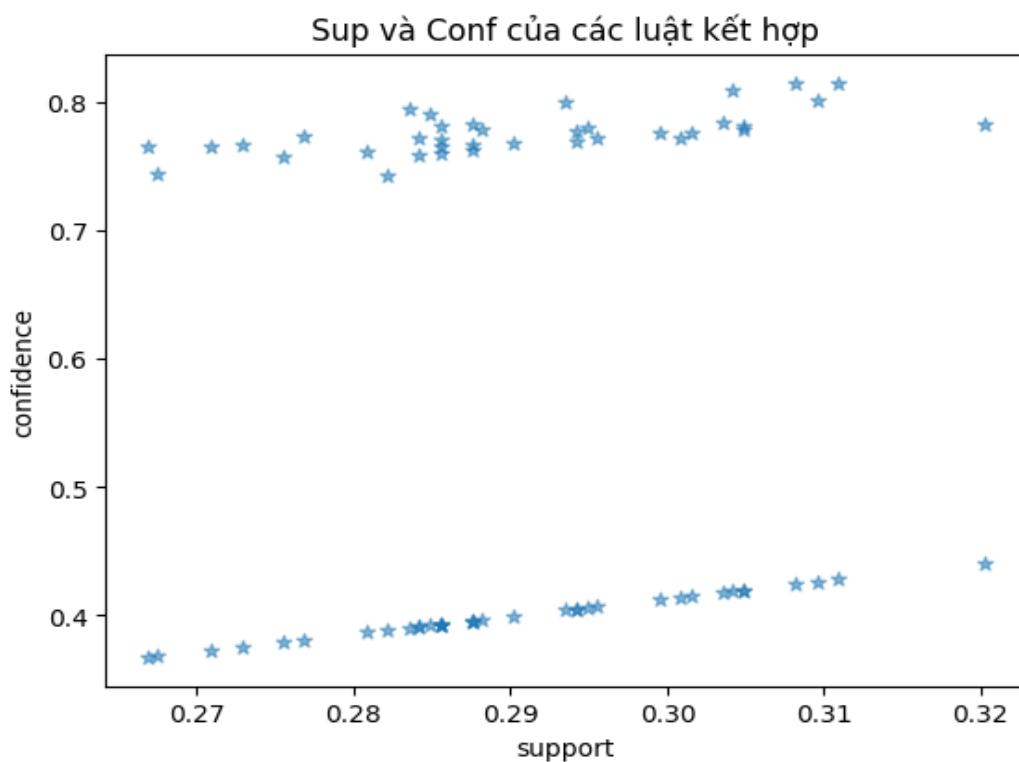
Như đã đề cập ở trên về việc chỉ số lift đều lớn hơn 1, và do đó ta tin rằng những tập luật mà ta tạo ra là có ý nghĩa, song như vậy vẫn chưa đủ do có quá nhiều tập luật, cho nên điều ta cần làm lúc này là tập trung vào vài luật kết hợp đáng quan tâm thỏa mãn các điều kiện của ta. Vì vậy ta cần xem phân bố của các tập luật là như thế nào, từ đó vẽ biểu đồ phân tán của các luật kết hợp theo chỉ số confidence và support của từng luật.

```

for i in range(len(support)):
    support[i] = support[i]
    confidence[i] = confidence[i]

plt.scatter(support, confidence, alpha=0.5, marker="*")
plt.xlabel('support')
plt.ylabel('confidence')
plt.show()

```



Ta thấy rằng các luật của chúng ta có độ tin cậy tập trung chủ yếu ở hai khoảng: Dưới 0.5 hoặc trên 0.7, và do mục tiêu là để tối đa nguồn lực kinh tế, điều ta cần làm là chọn ra các tập luật có độ tin cậy cao hơn. Bên cạnh đó, ta cũng cần chọn ra các tập luật có độ hỗ trợ cao hơn các tập luật còn lại. Và với những lập luận như vừa rồi, ta sẽ chọn những luật kết hợp có độ tin cậy lớn hơn 70% và độ hỗ trợ là hơn 31%. Ta được hình kết quả như dataframe dưới đây:

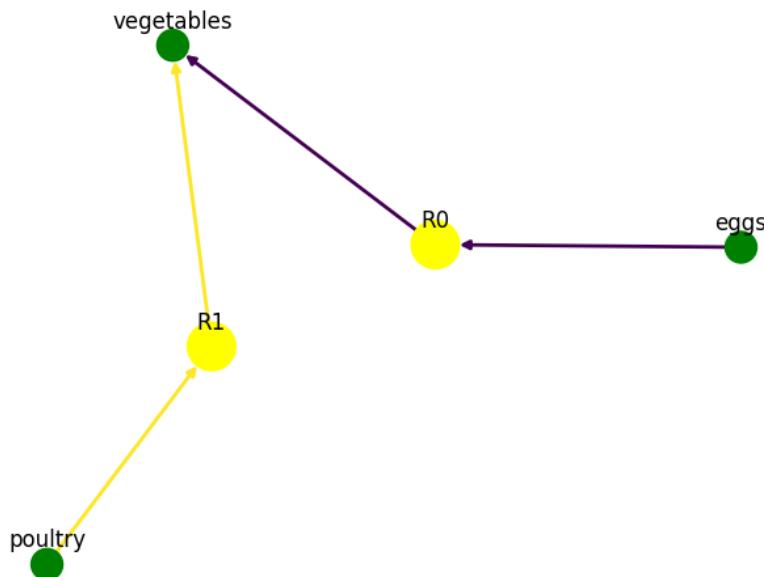
```
selected_rules = rules_eclat[(rules_eclat['confidence'] > 0.7) & (rules_eclat['support'] > 0.31)]
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
50	(eggs)	(vegetables)	0.381588	0.726484	0.310874	0.814685	1.121408	0.033656	1.475953	0.175067
6	(poultry)	(vegetables)	0.408939	0.726484	0.320213	0.783034	1.077841	0.023125	1.260640	0.122185

Ta cũng có thể trực quan luật kết hợp này bằng cách dùng một hàm nhóm tạo tên là draw\_graph. Hàm này sẽ lấy một dataframe chứa kết quả của các luật kết hợp, trực quan k luật kết hợp (k trong trường hợp này là 2 như trong hình), còn tham số cuối cùng là random\_seed để cố định lại hình vẽ, trong trường hợp này nhóm chọn random\_seed là 42. Ta gọi hàm:

```
# Vẽ luật kết hợp tập luật đã chọn:  
draw_graph(selected_rules, 2, 42)
```

Dưới đây là hình vẽ của luật kết hợp:



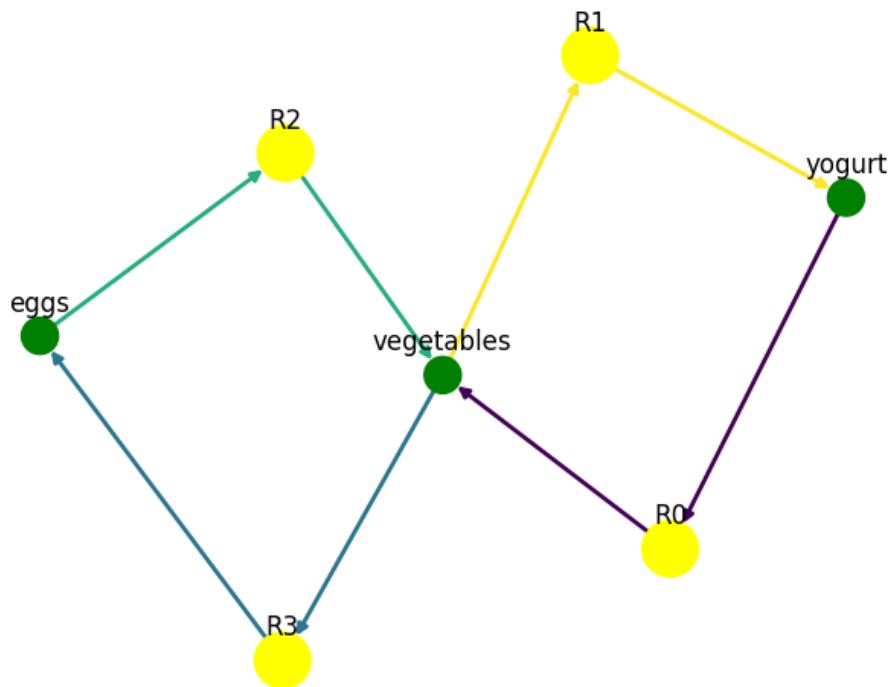
Luật này nói rằng nếu như một người đã mua các sản phẩm gia cầm thì theo như luật có index 1 trong bảng kết quả, người đó sẽ mua rau. Cách lập luận tương tự cho mặt hàng trứng, nếu một người đã mua trứng, thì theo như luật kết hợp có index 0 trong bảng kết quả, người đó sẽ mua rau.

Dựa vào đồ thị trên, có thể thấy rằng rau là một mặt hàng có ảnh hưởng tới hầu hết các luật kết hợp. Tuy nhiên, khi độ hỗ trợ lớn hơn độ tin cậy của các luật như “nếu người dùng mua sản phẩm X thì họ sẽ mua rau”, luật kết hợp sẽ trở nên vô nghĩa. Nhưng lift thấp nhất trong trường hợp này là một con số lớn hơn giá trị 1, do đó mà ta có thể kết luận rằng luật có nghĩa. Bên cạnh đó, ta cũng có thể vẽ để xác định các tập luật có lift cao nhất, để xem có tồn tại mối liên hệ gì giữa các luật kết hợp này không. Nhóm thực hiện điều này bằng cách gọi lại hàm vừa tạo, lần

này là với dataframe kết quả ban đầu, không phải cái đã được chọn như lần trước, và lần này nhóm sẽ vẽ đồ thị của 4 tập luật có lift cao nhất như sau:

```
# Vẽ các tập luật có lift cao nhất:  
draw_graph(rules_eclat, 4, 42)
```

Dưới đây là đồ thị thể hiện mối quan hệ giữa các luật kết hợp:



Nhớ lại với bảng kết quả về các luật kết hợp đầu tiên được tạo ra bởi thuật toán ECLAT, ta thấy rằng 4 tập luật đầu tiên đều có độ support và confidence tương đối cao, tuy nhiên các luật này lại không đạt yêu cầu về minSup và minConf mà nhóm đặt ra như lúc trước là minSup = 31% (minConf thì luật yogurt => vegetables đã đạt). Việc đặt ra mức minSup và minConf như vậy quá khắt khe cho nên chỉ thu được 2 luật kết hợp thỏa mãn. Tuy nhiên, có thể tạm chấp nhận luật kết hợp yogurt => vegetables do lift của nhóm này cao, và độ support của luật này gần như đạt ngưỡng của minSup.

Như vậy, từ bộ dữ liệu, ta sẽ chọn ra 3 luật kết hợp chính đó là:

**poultry => vegetables**

**eggs => vegetables**

**yogurt => vegetables**

#### **5.4 Đưa ra khuyến nghị dựa trên tập luật**

Sau khi chạy thuật toán ECLAT và nhận được bảng kết quả về các luật kết hợp có thể chọn, nhóm đã lọc ra những luật đáng tin cậy, kết quả cho thấy có 3 luật kết hợp như sau: Khi mua sản phẩm "poultry," họ cũng mua "vegetables"; khi mua "eggs," họ cũng mua "vegetables"; và khi mua "yogurt," họ cũng mua "vegetables."

Từ những luật này, chúng ta có thể suy đoán rằng những người mua này đang duy trì một lối sống khá lành mạnh, với sự ưu tiên cho các sản phẩm thuộc nhóm thực phẩm tươi như rau củ, từ đó phản ánh được sự quan tâm đến sức khỏe và chất lượng cuộc sống. Ngoài ra, sự liên kết giữa những sản phẩm như gà, trứng và sữa chua với rau củ cũng có thể chỉ ra rằng đây là lối tiêu dùng chung của hầu hết mọi phân khúc khách hàng.

Để tích hợp các sản phẩm như gà, trứng và sữa chua với rau củ trong chiến lược marketing và quảng bá, cửa hàng có thể thực hiện các ý tưởng sau:

- Tạo các gói sản phẩm kết hợp dựa trên giá trị dinh dưỡng, ví dụ như "Bữa ăn dinh dưỡng cho gia đình" bao gồm gà tươi, trứng hữu cơ, sữa chua thạch và rau củ tươi ngon.
- Tổ chức chương trình ẩm thực gia đình, tập trung vào việc nấu ăn với các sản phẩm trên và cung cấp ưu đãi khi mua sắm đồng loạt cho bữa ăn gia đình.
- Tạo thực đơn hoặc bảng gợi ý mua sắm trên các website, đề xuất cách kết hợp các sản phẩm để tạo nên các bữa ăn ngon miệng và dinh dưỡng.
- Tổ chức các buổi hướng dẫn nấu ăn trực tuyến hoặc tại cửa hàng, tập trung vào việc sử dụng đồng loạt các sản phẩm trên, và chia sẻ công thức nấu ăn.
- Tập trung vào lợi ích sức khỏe của việc ăn uống đa dạng và cân đối với sự kết hợp của gà, trứng, sữa chua và rau củ, tạo nội dung truyền thông với thông điệp về sự quan trọng của việc tích hợp những sản phẩm này vào chế độ ăn hàng ngày.

## CHƯƠNG 6: KẾT LUẬN

Tổng kết, việc áp dụng thuật toán ECLAT đã mang lại những kết quả quan trọng trong việc tìm kiếm và xác định các luật kết hợp đáng tin cậy từ dữ liệu. Các luật này không chỉ giúp hiểu rõ hơn về mối quan hệ giữa các sản phẩm, mà còn mở ra cơ hội để phát triển chiến lược tiếp thị và quảng bá sáng tạo.

Tính đến thời điểm hiện tại, nhóm đã gặp một số khó khăn trong việc xử lý thời gian chạy của thuật toán ECLAT. Thời gian chạy không như mong đợi, và điều này có thể phản ánh sự ảnh hưởng của khối lượng dữ liệu lớn hoặc một số vấn đề kỹ thuật khác trong quá trình triển khai.

Mặc dù vẫn đề về thời gian chạy, nhóm nhận thức rằng thuật toán ECLAT vẫn đưa ra bảng kết quả chính xác về các luật kết hợp. Điều này cho thấy sức mạnh và hiệu suất của thuật toán trong việc phân tích mối quan hệ giữa các sản phẩm, thậm chí khi đối mặt với thách thức về thời gian.

Một số nguyên nhân có thể xuất phát từ đặc điểm của dữ liệu hoặc từ thư viện mà nhóm sử dụng là chưa hiệu quả. Để giải quyết vấn đề này, cần tiếp tục nghiên cứu và thử nghiệm để tối ưu hóa thuật toán cho nhu cầu cụ thể của nhóm. Đồng thời, việc đánh giá và tối ưu hóa hiệu suất của thư viện sử dụng là một phần quan trọng để giải quyết vấn đề thời gian chạy.

Tóm lại, mặc dù còn một số thách thức, nhưng việc sử dụng thuật toán ECLAT đã đặt ra nền tảng cho việc tìm hiểu sâu hơn về hành vi mua sắm của khách hàng và cung cấp cơ sở cho quyết định kinh doanh chiến lược hơn. Điều này là một bước quan trọng trong hành trình nghiên cứu và phát triển của nhóm.

## TÀI LIỆU THAM KHẢO

A. Subashini, & M. Karthikeyan. (2019, 06 03). Itemset Mining using Horizontal and Vertical Data Format. *International Journal for Research in Engineering Application & Management (IJREAM)*, 05. <https://ijream.org/papers/IJREAMV05I0351188.pdf>

He Lan, Xiaoxue Ma, Laihao Ma, & Weiliang Qiao. (2022, 10 8). Reliability Engineering and System Safety 229 (2023) 108893 Available online 8 October 2022 0951-8320/© 2022 Published by Elsevier Ltd. Pattern investigation of total loss maritime accidents based on association rule mining. *ELSEVIER*.

[https://www.researchgate.net/publication/364262124\\_Pattern\\_investigation\\_of\\_total\\_loss\\_maritime\\_accidents\\_based\\_on\\_association\\_rule\\_mining](https://www.researchgate.net/publication/364262124_Pattern_investigation_of_total_loss_maritime_accidents_based_on_association_rule_mining)

Jiawei Han, Micheline Kamber, & Jian Pei. (2012). *Data Mining Concepts and Techniques*. <https://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-Jian-Pei-Data-Mining.-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf>

Nguyễn, T. A. (2023). *Chương 4. Luật kết hợp*.

[https://drive.google.com/drive/folders/1b6Ra7KaL7xb8R9NcIWC9SLxwISt\\_a-qj](https://drive.google.com/drive/folders/1b6Ra7KaL7xb8R9NcIWC9SLxwISt_a-qj)

Philippe Fournier-Viger. (2000). The Eclat Algorithm. [https://www.philippe-fournier-viger.com/COURSES/Pattern\\_mining/Eclat.pdf?fbclid=IwAR3Ajz3sYTvkEYY7GhoNt-PbOdOCcxK7UNCXAs1VecSwFHfiQ9kc2CdR1xA](https://www.philippe-fournier-viger.com/COURSES/Pattern_mining/Eclat.pdf?fbclid=IwAR3Ajz3sYTvkEYY7GhoNt-PbOdOCcxK7UNCXAs1VecSwFHfiQ9kc2CdR1xA)

Rana Ishita, & Amit Rathod. (2016, 06 13). Eclat with Large Data base Parallel Algorithm and Improve its Efficiency. *International Journal of Computer Applications*. <https://www.ijcaonline.org/archives/volume143/number13/ishita-2016-ijca-910462.pdf>

Shamila Nasreen, Muhammad Awais Azam, Khurram Shehzad, Usman Naeem, & Mustansar Ali Ghazanfar. (2014). Frequent Pattern Mining Algorithms for Finding Associated Frequent Patterns for Data Streams: A Survey. *ELSEVIER*.  
[https://www.researchgate.net/publication/265788733\\_Frequent\\_Pattern\\_Mining\\_Algorithms\\_for\\_Finding\\_Associated\\_Frequent\\_Patterns\\_for\\_Data\\_Streams\\_A\\_Survey](https://www.researchgate.net/publication/265788733_Frequent_Pattern_Mining_Algorithms_for_Finding_Associated_Frequent_Patterns_for_Data_Streams_A_Survey)

Xinhao Wang, Xifei Huang, Yingju Zhang, Xuhai Pan, & Kai Sheng. (2021). A Data-Driven Approach Based on Historical Hazard Records for Supporting Risk Analysis in Complex Workplaces. *Hindawi*. [https://www.researchgate.net/publication/355875276\\_A\\_Data-Driven\\_Approach\\_Based\\_on\\_Historical\\_Hazard\\_Records\\_for\\_Supporting\\_Risk\\_Analysis\\_in\\_Complex\\_Workplaces](https://www.researchgate.net/publication/355875276_A_Data-Driven_Approach_Based_on_Historical_Hazard_Records_for_Supporting_Risk_Analysis_in_Complex_Workplaces)

Yi Zeng, Shiqun Yi, Jiangyue Liu, & Miao Zhang. (2015). Research of Improved FP-Growth Algorithm in Association Rules Mining. *Scientific Programming*.  
<https://www.hindawi.com/journals/sp/2015/910281/>

**DANH SÁCH THÀNH VIÊN**

MSSV	Họ và tên	Mức độ đóng góp
31211027664	Nguyễn Nhật Quang	100%
31211027667	Đào Thị Phương Quỳnh	100%
31211027669	Văn Ngọc Như Quỳnh	100%
31211027671	Nguyễn Thị Phương Thảo	100%
31211026754	Lý Gia Thuận	100%