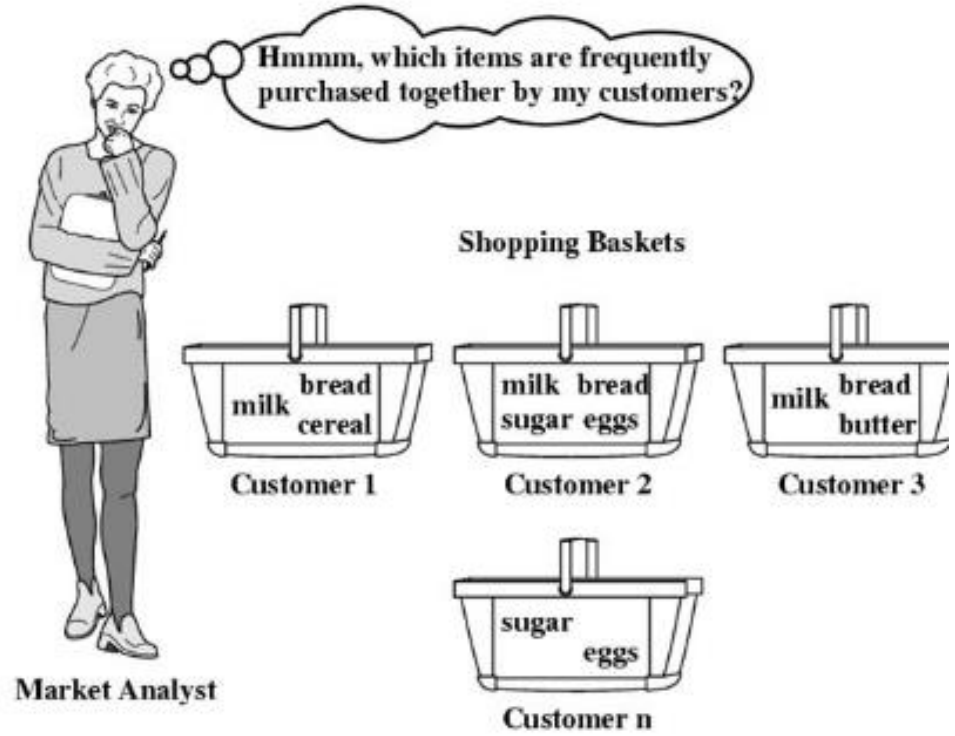
ASSOCIATION RULES
LUẬT KẾT HỢP

TS. Nguyễn Thị Ngọc Anh
Email: ngocanhnt@ued.udn.vn

ASSOCIATION RULES: LUẬT KẾT HỢP



- Phân tích việc mua hàng của khách hàng bằng cách tìm ra những "mối kết hợp" giữa những mặt hàng mà khách đã mua.
- Bài toán được R. Agrawal thuộc nhóm nghiên cứu của IBM đưa ra vào năm 1993.

Nội dung

1. Khái niệm và định nghĩa
   - Tập mục, giao dịch, CSDL giao dịch
   - Tập phổ biến (TPB) và luật kết hợp (LKH)
2. **Các phương pháp khai phá TPB và LKH**
   - Phương pháp Apriori
   - Phương pháp FP-Growth
   - Các phương pháp khác
3. Đánh giá luật kết hợp
4. Các ứng dụng thực tiễn

KHÁI NIỆM LUẬT KẾT HỢP

- Luật kết hợp: *Mối quan hệ kết hợp giữa các tập thuộc tính trong cơ sở dữ liệu.*
- Ví dụ:
  - $\{$*bánh mỳ, bơ, mứt dâu*$\} \rightarrow \{$*sữa tươi*$\}$ (phổ biến: 3%, tin cậy: 80%)
  - $\{$*tuổi* > 45, *gia đình có lịch sử tiểu đường, huyết áp cao*$\} \rightarrow \{$*mắc bệnh tiểu đường*$\}$ (phổ biến: 1.5%, tin cậy: 76%)

KHÁI NIỆM LUẬT KẾT HỢP

Khai phá luật kết hợp: là tìm ra các mẫu có tần suất cao, các mẫu kết hợp, liên quan hoặc các cấu trúc tồn tại giữa các tập hợp đối tượng trong cơ sở dữ liệu các giao dịch, cơ sở dữ liệu quan hệ hoặc các kho chứa thông tin khác.

$\Rightarrow$ Đi tìm tất cả các **tập phổ biến** từ trong dữ liệu

$\Rightarrow$ Nhiệm vụ tìm ra các luật mà dự đoán sự xuất hiện của một đối tượng dựa vào sự xuất hiện của các đối tượng khác trong giao dịch.

Mẫu phổ biến (frequent patterns/itemsets) là các mẫu mà xuất hiện một cách thường xuyên trong một tập dữ liệu.

TẬP MỤC, GIAO DỊCH, VÀ CƠ SỞ DỮ LIỆU GIAO DỊCH (Itemset, Transaction, and Transactional Database)

item (hạng mục/phần tử)

itemset (Tập các hạng mục – Tập mục): danh sách các item trong giỏ hàng

Transaction (Giao dịch): là tập các Itemset được mua trong một giỏ hàng, lưu kèm với mã giao dịch (TID).

Frequent itemset (**Tập mục phổ biến**): các mẫu mà xuất hiện một cách thường xuyên trong một tập dữ liệu (xuất hiện khá nhiều trong các giao dịch).

k-itemset: danh sách sản phẩm:

+ 1-itemset: {A,B,C}

+ 2-itemset: {{A,B}, {A,C}}

+ 3-itemset: {{A,B,C}, {B,C, E}}

TẬP MỤC, GIAO DỊCH, VÀ CƠ SỞ DỮ LIỆU GIAO DỊCH (Itemset, Transaction, and Transactional Database)

- Ký hiệu $\mathbb{I} = \{x_1, x_2, \ldots, x_n\}$ là tập n mục (item). Ví dụ:
  - Tập tất cả các mặt hàng thực phẩm trong siêu thị: $\mathbb{I} = \{$*sữa*, *trứng*, *đường*, *bánh mỳ*, *mật ong*, *mứt*, *bơ*, *thịt bò*, *giá*, $\ldots\}$.
  - Tập tất cả các bộ phim: $\mathbb{I} = \{$*pearl harbor*, *fast and furious 7*, *fifty shades of grey*, *spectre*, $\ldots\}$.
- Một tập $X \subseteq \mathbb{I}$ được gọi là một tập mục (itemset).
- Nếu X có k mục (tức $|X| = k$) thì X được gọi là k–itemset.
- Ký hiệu $\mathbb{D} = \{T_1, T_2, \ldots, T_m\}$ là cơ sở dữ liệu gồm m giao dịch (transaction). Mỗi giao dịch $T_i \in \mathbb{D}$ là một tập mục, tức $T_i \subseteq \mathbb{I}$.

TẬP MỤC, GIAO DỊCH, VÀ CƠ SỞ DỮ LIỆU GIAO DỊCH (Itemset, Transaction, and Transactional Database)

Ví dụ

Tập tất cả các mục $\mathbb{I}$:

$\mathbb{I} = \{A, B, C, D, E\}$

Cơ sở dữ liệu giao dịch $\mathbb{D}$:

$\mathbb{D} = \{T_1, T_2, T_3, T_4, T_5, T_6\}$, cụ thể:

- $T_1 = \{A, B, D, E\}$
- $T_2 = \{B, C, E\}$
- $T_3 = \{A, B, D, E\}$
- $T_4 = \{A, B, C, E\}$
- $T_5 = \{A, B, C, D, E\}$
- $T_6 = \{B, C, D\}$

TẬP/MẪU PHỔ BIẾN
(Frequent itemset/pattern)

- Cho tập mục X ($\subseteq \mathbb{I}$).
- Độ hỗ trợ (*support*) của X, ký hiệu là $sup(X, \mathbb{D})$, là số lượng giao dịch trong $\mathbb{D}$ chứa X:

$$sup(X, \mathbb{D}) = |\{T | T \in \mathbb{D} \text{ và } X \subseteq T\}| \quad (1)$$

- Độ hỗ trợ tương đối (*relative support*) của X, ký hiệu là $rsup(X, \mathbb{D})$, là số phần trăm các giao dịch trong $\mathbb{D}$ chứa X:

$$rsup(X, \mathbb{D}) = \frac{sup(X, \mathbb{D})}{|\mathbb{D}|} \quad (2)$$

- Tập mục X được gọi là tập (mục) phổ biến (frequent itemset) trong $\mathbb{D}$ nếu $sup(X, \mathbb{D}) \geq minsup$, với $minsup$ là một ngưỡng độ hỗ trợ tối thiểu (minimum support threshold) do người dùng định nghĩa.
- Ký hiệu $\mathbb{F}$ là tập tất cả các tập phổ biến.
- Ký hiệu $\mathbb{F}^{(k)}$ là tập tất cả các tập phổ biến có độ dài k (frequent k–itemsets).

Các tập phổ biến (với $minsup = 3$) từ cơ sở dữ liệu $\mathbb{D}$

$\mathbb{D} = \{T_1, T_2, T_3, T_4, T_5, T_6\}$:

- $T_1 = \{A, B, D, E\}$
- $T_2 = \{B, C, E\}$
- $T_3 = \{A, B, D, E\}$
- $T_4 = \{A, B, C, E\}$
- $T_5 = \{A, B, C, D, E\}$
- $T_6 = \{B, C, D\}$

Tập tất cả các tập phổ biến $\mathbb{F}$ và các $\mathbb{F}^{(k)}$:

- $\mathbb{F} = \{A, B, C, D, E, AB, AD, AE, BC, BD, BE, CE, DE, ABD, ABE, ADE, BCE, BDE, ABDE\}$
- $\mathbb{F}^{(1)} = \{A, B, C, D, E\}$
- $\mathbb{F}^{(2)} = \{AB, AD, AE, BC, BD, BE, CE, DE\}$
- $\mathbb{F}^{(3)} = \{ABD, ABE, ADE, BCE, BDE\}$
- $\mathbb{F}^{(4)} = \{ABDE\}$

LUẬT KẾT HỢP (Association Rule)

- Luật kết hợp có dạng:

$$X \rightarrow Y \tag{3}$$

với X và Y là hai tập mục ($X, Y \subseteq \mathbb{I}$) và $X \cap Y = \emptyset$.

- Độ hỗ trợ (*support*) của luật $X \rightarrow Y$ trong cơ sở dữ liệu $\mathbb{D}$, ký hiệu là $sup(X \rightarrow Y, \mathbb{D})$, là số giao dịch chứa cả X và Y:

$$sup(X \rightarrow Y, \mathbb{D}) = sup(X \cup Y, \mathbb{D}) \tag{4}$$

- Độ hỗ trợ tương đối (*relative support*) của luật $X \rightarrow Y$ trong cơ sở dữ liệu $\mathbb{D}$, ký hiệu $rsup(X \rightarrow Y, \mathbb{D})$, là số phần trăm các giao dịch trong $\mathbb{D}$ chứa cả X và Y:

$$rsup(X \rightarrow Y, \mathbb{D}) = \frac{sup(X \cup Y, \mathbb{D})}{|\mathbb{D}|} \tag{5}$$

- Luật $X \rightarrow Y$ được gọi là phổ biến (frequent) nếu:

$$sup(X \rightarrow Y, \mathbb{D}) \geq minsup \tag{6}$$

LUẬT KẾT HỢP (Association Rule)

- Độ tin cậy (*confidence*) của luật $X \rightarrow Y$ trong $\mathbb{D}$, ký hiệu $conf(X \rightarrow Y, \mathbb{D})$, là tỉ lệ giữa số giao dịch chứa cả X và Y trên số giao dịch chỉ chứa X:

$$conf(X \rightarrow Y, \mathbb{D}) = \frac{sup(X \cup Y, \mathbb{D})}{sup(X, \mathbb{D})} \tag{7}$$

- Một cách diễn giải khác: $conf(X \rightarrow Y, \mathbb{D})$ là xác suất có điều kiện mà một giao dịch trong $\mathbb{D}$ chứa Y khi nó đã chứa X: $conf(X \rightarrow Y, \mathbb{D}) = P(Y|X)$. Tuy nhiên bản chất vẫn là mức độ tin cậy của luật.
- Luật $X \rightarrow Y$ được gọi là mạnh (*strong*) nếu độ tin cậy của nó lớn hơn hoặc bằng một ngưỡng *minconf* nào đó do người dùng định nghĩa:

$$conf(X \rightarrow Y, \mathbb{D}) \geq minconf \tag{8}$$

LUẬT KẾT HỢP (Association Rule) Ví dụ minh họa:

$\mathbb{D} = \{T_1, T_2, T_3, T_4, T_5, T_6\}$:

- $T_1 = \{A, B, D, E\}$
- $T_2 = \{B, C, E\}$
- $T_3 = \{A, B, D, E\}$
- $T_4 = \{A, B, C, E\}$
- $T_5 = \{A, B, C, D, E\}$
- $T_6 = \{B, C, D\}$

- Xét luật $\{B, C\} \rightarrow \{E\}$ (ngắn gọn là $BC \rightarrow E$):
  - $sup(BC \rightarrow E, \mathbb{D}) = sup(BCE, \mathbb{D}) = 3$
  - $conf(BC \rightarrow E, \mathbb{D}) = \frac{sup(BCE, \mathbb{D})}{sup(BC, \mathbb{D})} = \frac{3}{4} = 0.75$ (tức 75%)
- Xét luật $\{A, D\} \rightarrow \{B, E\}$ (ngắn gọn là $AD \rightarrow BE$):
  - $sup(AD \rightarrow BE, \mathbb{D}) = sup(ABDE, \mathbb{D}) = 3$
  - $conf(AD \rightarrow BE, \mathbb{D}) = \frac{sup(ABDE, \mathbb{D})}{sup(AD, \mathbb{D})} = \frac{3}{3} = 1.0$ (tức 100%)

Nội dung

1. Khái niệm và định nghĩa
   - Tập mục, giao dịch, CSDL giao dịch
   - Tập phổ biến (TPB) và luật kết hợp (LKH)
2. **Các phương pháp khai phá TPB và LKH**
   - Phương pháp Apriori
   - Phương pháp FP-Growth
   - Các phương pháp khác
3. Đánh giá luật kết hợp
4. Các ứng dụng thực tiễn

CÁC BƯỚC KHAI PHÁ LUẬT KẾT HỢP

Hai bước khai phá luật kết hợp từ CSDL giao dịch $\mathbb{D}$:

- **Mining frequent itemsets/patterns**: Khai phá tất cả các tập phổ biến từ cơ sở dữ liệu $\mathbb{D}$ với ngưỡng hỗ trợ tối thiểu *minsup*.
- **Generating strong rules from mined frequent itemsets/patterns**: Sinh tất cả các luật mạnh từ các tập phổ biến được khai phá ở bước trước với ngưỡng tin cậy tối thiểu *minconf*.

- Bước một có độ phức tạp tính toán cao hơn và thường chiếm phần lớn thời gian khai phá luật kết hợp.
- Số lượng các tập mục (itemsets) là rất lớn. Ví dụ với $\mathbb{I} = \{x_1, x_2, \ldots, x_{100}\}$ chúng ta có $2^{100} - 1 \approx 1.27 \times 10^{30}$ tập con (không tính tập $\emptyset$).

KHAI PHÁ TẬP MỤC THƯỜNG XUYÊN

Bài toán khai phá tập mục thường xuyên có thể chia thành hai bài toán nhỏ:

1. Tìm các tập mục ứng viên. Tập các ứng viên là tập mục mà có thể hy vọng nó là tập thường xuyên.
2. Tìm các tập mục thường xuyên. Tập mục thường xuyên là tập mục có độ hỗ trợ lớn hơn hoặc bằng ngưỡng hỗ trợ tối thiểu cho trước.

PHƯƠNG PHÁP APIORI

Apriori là một giải thuật được A. Agrawal và các cộng sự đề xuất lần đầu vào năm 1993 nhằm khai phá tập mục phổ biến nhị phân. Thuật toán này thực hiện lặp lại việc tìm kiếm theo mức, sử dụng thông tin ở mức k để duyệt mức k +1.

Phương pháp Apriori tìm các tập mục thường xuyên bằng cách sinh ứng viên.

MỘT SỐ TÍNH CHẤT SỬ DỤNG TRONG PHƯƠNG PHÁP APIORI

- Cho hai tập mục $X, Y \subseteq \mathbb{I}$ và cơ sở dữ liệu $\mathbb{D}$.
- Nếu $X \subseteq Y$ thì $sup(X, \mathbb{D}) \geq \sup(Y, \mathbb{D})$.

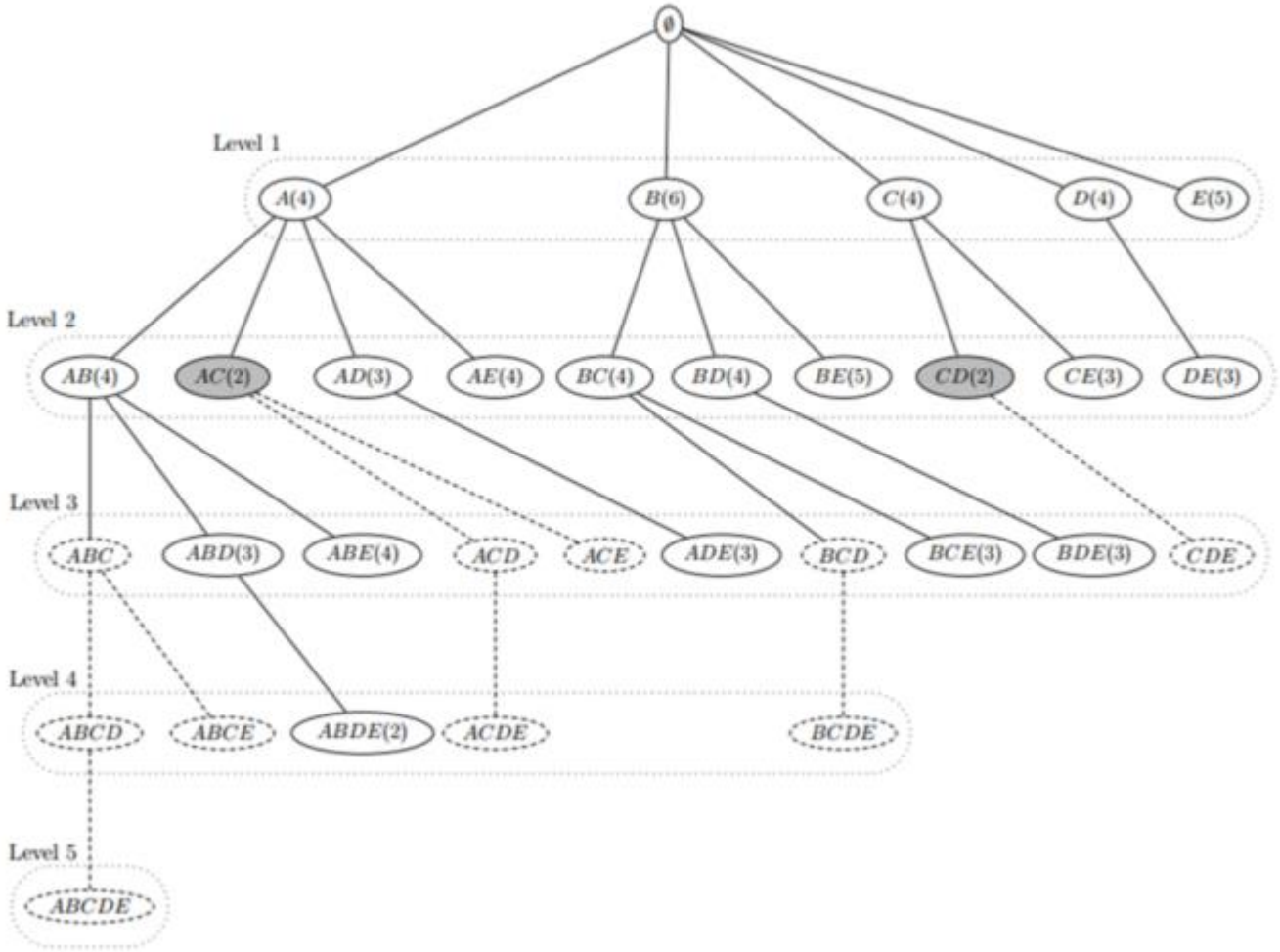
Hai tính chất Apriori:

- Nếu Y là tập phổ biến (frequent) thì mọi tập con X ($\subseteq Y$) của Y đều phổ biến.
- Nếu X là tập không phổ biến (infrequent) thì mọi tập cha Y ($\supseteq X$) của X đều không phổ biến.

- Phương pháp Apriori dựa vào hai tính chất trên để cải tiến phương pháp vét cạn bằng cách cắt tỉa các nhánh không cần thiết trên giàn tập mục.
- Cụ thể, khi duyệt theo bề rộng (BFS) trên giàn tập mục, thuật toán Apriori cắt tỉa hết tất cả các tập cha của tập không phổ biến.

PHƯƠNG PHÁP APIORI

Cắt tỉa trên giàn tập mục trong Apriori (*minsup* = 3)



[Nguồn: Data Mining and Analysis: Fundamental Concepts and Algorithms by Zaki and Jr]

PHƯƠNG PHÁP APIORI

Cắt tỉa trên giàn tập mục trong Apriori ($minsup = 3$) - tiếp

- Ở hình trước, các nút màu sậm là các tập mục không phổ biến.
- Tất cả các tập cha của chúng trên giàn (các nút vạch đứt) đều bị cắt tỉa, dẫn đến toàn bộ các nhánh vạch đứt được cắt tỉa.
- Ví dụ: tập AC có $sup(AC, \mathbb{D}) = 2 < minsup$ nên các tập cha của AC có tiền tố là AC sẽ bị cắt tỉa, dãn đến toàn bộ cây con dưới nút AC bị cắt tỉa.

APIORI ALGORITHM

1. **Duyệt toàn bộ CSDL** giao dịch để tính giá trị hỗ trợ là phần tử của tập phổ biến tiềm năng C^1 của 1-itemset, so sánh với minsup, để có được 1-itemset (F^1);
2. **F^1** nối (phép join) **F^1** để sinh ra 2-itemset là tập phổ biến tiềm năng. Loại bỏ các tập mục không phải là tập mục phổ biến thu được 2-itemset C^2;
3. **Duyệt toàn bộ CSDL** giao dịch để tính ra giá trị hỗ trợ của mỗi ứng viên 2-itemset, so sánh từng phần tử với minsup để thu được tập mục thường xuyên 2-itemset (**F^2**)
4. **Lặp lại từ bước 2** cho đến khi tập ứng cử tiềm năng C (Không tìm thấy tập mục phổ biến)
5. **Với mỗi mục phổ biến I,** sinh ra tất cả các tập con s không rỗng của I
6. **Với mỗi tập con s không rỗng của I,** sinh ra các luật s=>(I-s) nếu độ tin cậy của nó lớn hơn hoặc bằng minconf.

PHƯƠNG PHÁP APIORI: VÍ DỤ MINH HỌA

Tập tất cả các mục $\mathbb{I}$:

$\mathbb{I} = \{A, B, C, D, E\}$

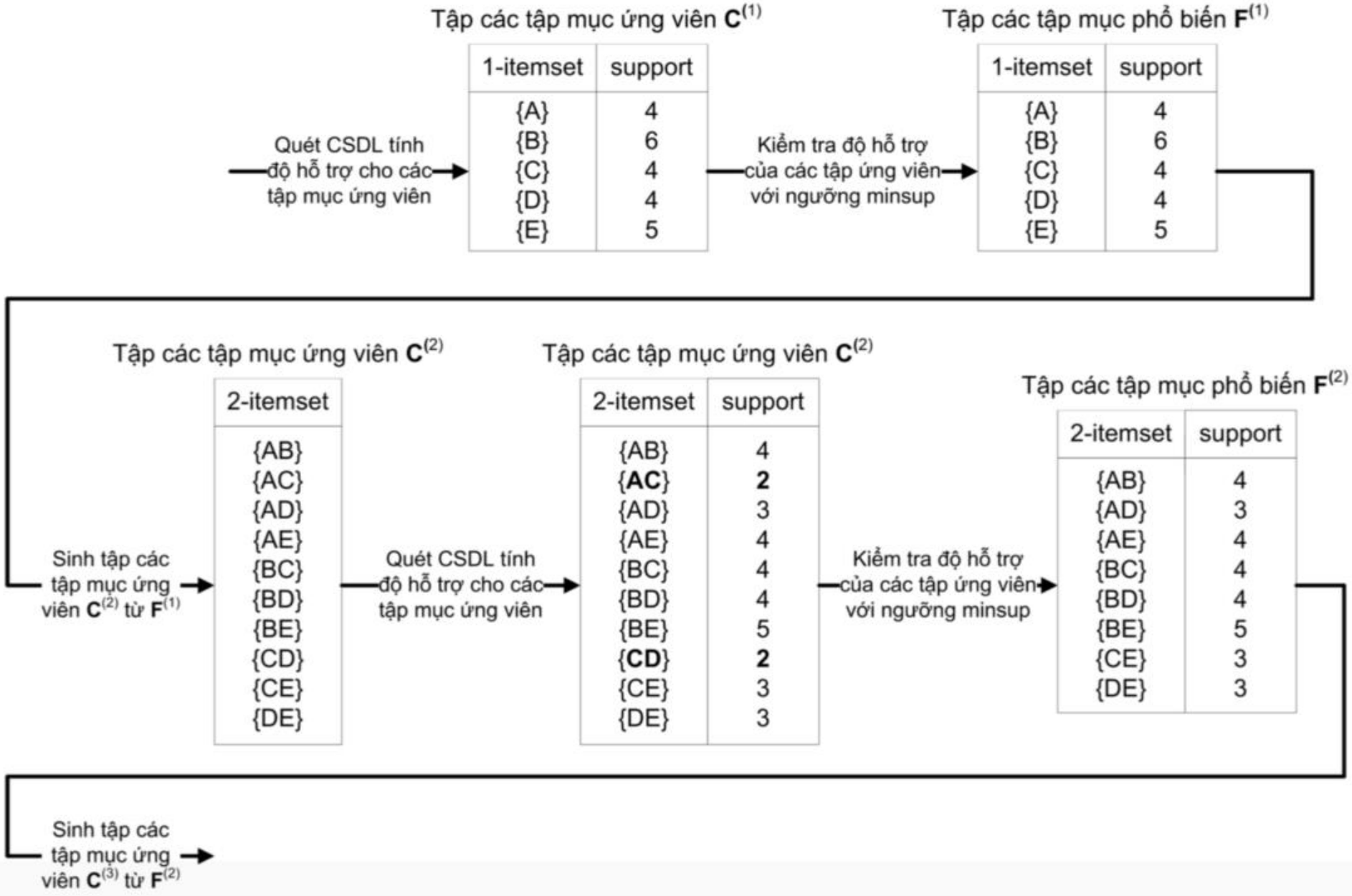
Cơ sở dữ liệu giao dịch $\mathbb{D}$:

$\mathbb{D} = \{T_1, T_2, T_3, T_4, T_5, T_6\}$, cụ thể:

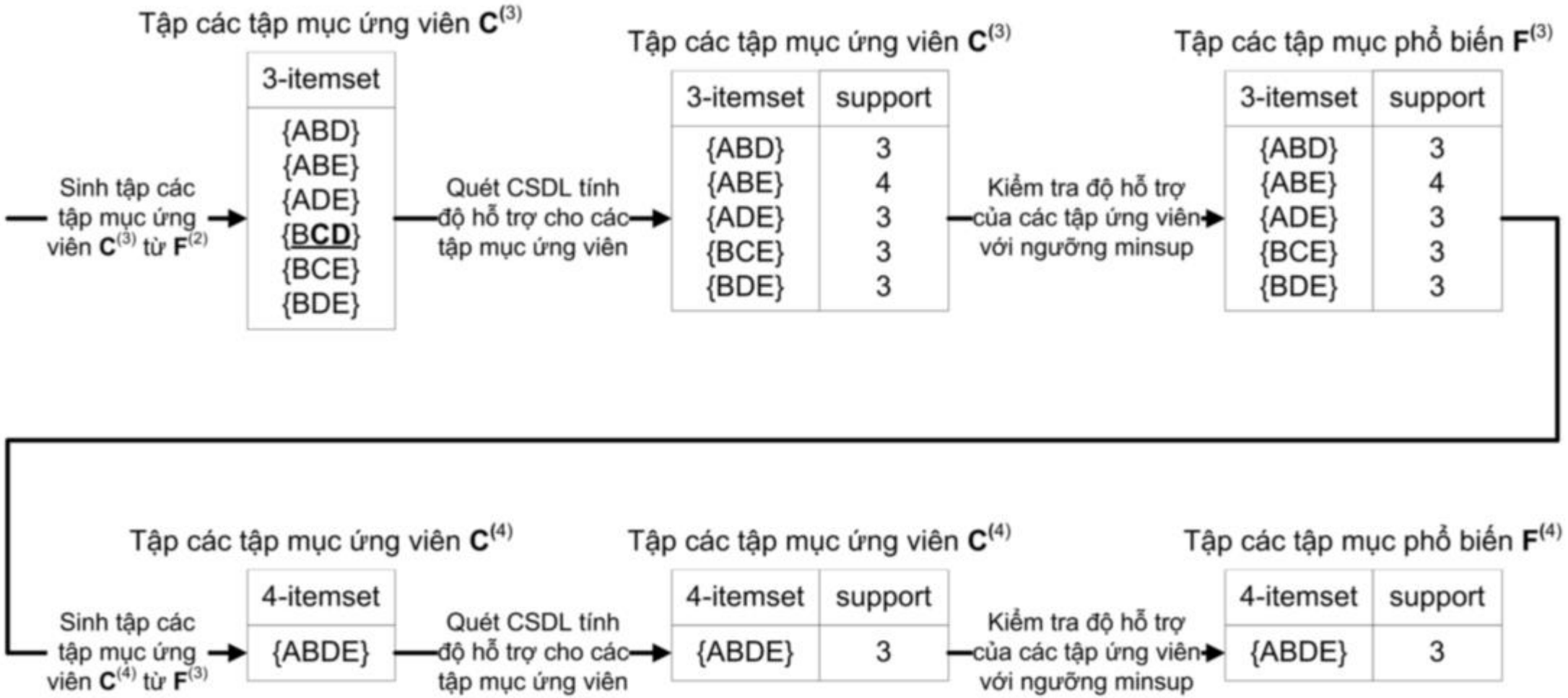
- $T_1 = \{A, B, D, E\}$
- $T_2 = \{B, C, E\}$
- $T_3 = \{A, B, D, E\}$
- $T_4 = \{A, B, C, E\}$
- $T_5 = \{A, B, C, D, E\}$
- $T_6 = \{B, C, D\}$

- Với $minsup = 3$.

PHƯƠNG PHÁP APIORI: VÍ DỤ MINH HỌA



PHƯƠNG PHÁP APIORI: VÍ DỤ MINH HỌA



PHƯƠNG PHÁP APIORI

Thuật toán Apriori

1: **procedure** APRIORI($\mathbb{D} = \{T_1, T_2, \ldots, T_m\}$, $\mathbb{I} = \{x_1, x_2, \ldots, x_n\}$, *minsup*)
2: Khởi tạo tập các tập phổ biến: $\mathbb{F} \leftarrow \emptyset$;
3: $\mathbb{F}^{(1)} \leftarrow$ FindFrequent1Itemsets($\mathbb{D}$, $\mathbb{I}$, *minsup*);
4: **for** ($k = 2$; $\mathbb{F}^{(k-1)} \neq \emptyset$; k++) **do**
5: $\mathbb{C}^{(k)} \leftarrow$ AprioriGen($\mathbb{F}^{(k-1)}$);
6: **for** (each transaction $T \in \mathbb{D}$) **do**
7: $\mathbb{C}_T \leftarrow$ SubsetsOfT($\mathbb{C}^{(k)}$, T);
8: **for** (each $C \in \mathbb{C}_T$) **do**
9: $C.count$++;
10: **end for**
11: **end for**
12: $\mathbb{F}^{(k)} \leftarrow \{C \in \mathbb{C}^{(k)} | C.count \geq minsup\}$;
13: **end for**
14: $\mathbb{F} \leftarrow \mathbb{F}^{(1)} \cup \mathbb{F}^{(2)} \cup \cdots \cup \mathbb{F}^{(k)}$;
15: **return** $\mathbb{F}$;
16: **end procedure**

PHƯƠNG PHÁP APIORI

Thuật toán Apriori (2)

```
1: procedure AprioriGen(𝔽^(k−1))
2:     Khởi tạo tập các tập mục ứng viên: ℂ^(k) ← ∅;
3:     for (each itemset F1 ∈ 𝔽^(k−1)) do
4:         for (each itemset F2 ∈ 𝔽^(k−1)) do
5:             if ((F1[1] = F2[1]) ∧ ... ∧ (F1[k − 2] = F2[k − 2]) ∧ (F1[k − 1] < F2[k − 1])) then
6:                 C ← F1 ⋈ F2;
7:                 if (HasInfrequentSubset(C, 𝔽^(k−1))) then
8:                     remove C;
9:                 else
10:                    ℂ^(k) ← ℂ^(k) ∪ {C};
11:                end if
12:            end if
13:        end for
14:    end for
15:    return ℂ^(k);
16: end procedure
```

PHƯƠNG PHÁP APIORI

Thuật toán Apriori (3)

```
1: procedure HasInfrequentSubset(C, F^(k−1))
2:     for (each (k − 1)–subset S of C) do
3:         if (S ∉ F^(k−1)) then
4:             return TRUE;
5:         end if
6:     end for
7:     return FALSE;
8: end procedure
```

SINH LUẬT KẾT HỢP PHỔ BIẾN VÀ MẠNH TỪ CÁC TẬP PHỔ BIẾN

- **Input**: Tập tất cả các tập phô biên $\mathbb{F}$.
- **Output**:Tập tất cả các luật phổ biến (frequent) và mạnh (strong): $\mathbb{R}$.

```
1:  procedure GenFrequentStrongRules(𝔽, minconf)
2:      Khởi tạo ℝ ← ∅;
3:      for (với mỗi tập mục phổ biến F ∈ 𝔽 và |F| ≥ 2) do
4:          𝕏 ← {X|X ⊂ F, X ≠ ∅};
5:          while (𝕏 ≠ ∅) do
6:              Y ← maximal element in 𝕏;
7:              𝕏 ← 𝕏 \ Y;
8:              if (conf(Y → F \ Y) ≥ minconf) then
9:                  ℝ ← ℝ ∪ {Y → F \ Y};
10:             else
11:                 𝕏 ← 𝕏 \ {Z|Z ⊂ Y}
12:             end if
13:         end while
14:     end for
15:     return ℝ;
16: end procedure
```

MINH HỌA THUẬT TOÁN SINH LUẬT

Sinh luật cho tập phổ biến $ABDE$ có độ hỗ trợ bằng 3 với độ tin cậy tối thiểu $minconf = 0.8$:

- $\mathbb{X} = \{ABD(3), ABE(4), ADE(3), BDE(3), AB(4), AD(4), AE(4), BD(4), BE(5), DE(3), A(4), B(6), D(4), E(5)\}$.
- $Y = ABD$: $conf(ABD \to E) = \frac{3}{3} = 1.0 \geq 0.8$ nên $ABD \to E$ là luật mạnh.
- $Y = ABE$: $conf(ABE \to D) = \frac{3}{4} = 0.75 < 0.8$ nên $ABE \to D$ không tin cậy. Khi đó có thể loại bỏ khỏi $\mathbb{X}$ tất cả các tập con của ABE. Do đó, $\mathbb{X} = \{ADE(3), BDE(3), AD(4), BD(4), DE(3), D(4)\}$.
- $Y = ADE$: $conf(ADE \to B) = \frac{3}{3} = 1.0 \geq 0.8$ nên $ADE \to B$ là luật mạnh.
- $Y = BDE$: $conf(BDE \to A) = \frac{3}{3} = 1.0 \geq 0.8$ nên $BDE \to A$ là luật mạnh.
- $Y = AD$: $conf(AD \to BE) = \frac{3}{4} = 0.75 < 0.8$ nên $AD \to BE$ không tin cậy. Khi đó có thể loại bỏ khỏi $\mathbb{X}$ tất cả các tập con của AD. Do đó, $\mathbb{X} = \{BD(4), DE(3)\}$.
- $Y = BD$: $conf(BD \to AE) = \frac{3}{4} = 0.75 < 0.8$ nên $BD \to AE$ không tin cậy. Khi đó có thể loại bỏ khỏi $\mathbb{X}$ tất cả các tập con của BD. Do đó, $\mathbb{X} = \{DE(3)\}$.
- $Y = DE$: $conf(DE \to AB) = \frac{3}{3} = 1.0 \geq 0.8$ nên $DE \to AB$ là luật mạnh.

PHƯƠNG PHÁP APIORI: ƯU VÀ NHƯỢC ĐIỂM

- Ưu điểm:
  - Nhờ các tính chất Apriori để cắt tỉa được nhiều nhánh trên giàn (lattice), giảm bớt đáng kể việc sinh các tập mục ứng viên và kiểm tra tính phổ biến của các tập ứng viên đó.
- Nhược điểm:
  - Vẫn cần sinh ra một lượng lớn các tập ứng viên. Ví dụ, nếu có 10^4 tập mục phổ biến gồm một mục (1–itemsets), thuật toán Apriori cần sinh ra hơn 10^7 tập mục ứng viên có hai mục (2–itemsets).
  - Cần quét cơ sở dữ liệu nhiều lần để đếm độ hỗ trợ của các tập ứng viên trong quá trình thực hiện thuật toán.

PHƯƠNG PHÁP FP-GROWTH

- Cấu trúc dữ liệu FP–Tree (Frequent Pattern Tree)
- Sinh cây FP–Tree từ cơ sở dữ liệu
- Sinh tập phổ biến từ FP-Tree
- Ưu và nhược điểm của phương pháp FP–Growth

PHƯƠNG PHÁP FP-GROWTH

Cấu trúc dữ liệu FP–Tree

- Mỗi nốt trên cây được gắn nhãn là một mục (item).
- Các nốt con của một nốt đại diện cho các mục khác nhau.
- Mỗi nốt cũng lưu thông tin về độ hỗ trợ (support) của tập mục (itemset) bao gồm tất cả các mục trên đường đi từ nốt gốc đến nó.
- Có một bảng lưu tất cả các mục và con trỏ (node–link) để liên kết tất cả các vị trí xuất hiện của mỗi mục trong cây.

Thuật toán sinh cây FP-Tree $\mathbb{T}$ từ CSDL giao dịch $\mathbb{D}$

1: **procedure** BUILDFPTREE($\mathbb{D} = \{T_1, T_2, \ldots, T_m\}$)
2: Khởi tạo cây FP–Tree $\mathbb{T}$ chỉ chứa nốt gốc $\emptyset$ và $\emptyset.support \leftarrow 0$;
3: **for** (với mỗi giao dịch $T \in \mathbb{D}$) **do**
4: $T' = \{x^1, \ldots, x^h\} \leftarrow$ sắp xếp các mục phổ biến $\in T$ giảm dần theo *support*;
5: $pNode \leftarrow \emptyset$;
6: **for** ($i = 1; i \leq h; i{+}{+}$) **do**
7: **if** ($cNode \in$ Children($pNode$) and $cNode.label = x^i$) **then**
8: $cNode.support{+}{+}$;
9: $pNode \leftarrow cNode$;
10: **else**
11: Tạo nốt $cNode$ là một nốt con mới của $pNode$;
12: $cNode.label \leftarrow x^i$;
13: $cNode.support \leftarrow 1$;
14: $pNode \leftarrow cNode$;
15: **end if**
16: **end for**
17: $\emptyset.support{+}{+}$;
18: **end for**
19: **return** cây FP–Tree $\mathbb{T}$;
20: **end procedure**

PHƯƠNG PHÁP FP-GROWTH

CSDL giao dịch $\mathbb{D}$ minh họa phương pháp FP–Growth

Tập tất cả các mục $\mathbb{I}$:

$\mathbb{I} = \{A, B, C, D, E\}$

Cơ sở dữ liệu giao dịch $\mathbb{D}$:

$\mathbb{D} = \{T_1, T_2, T_3, T_4, T_5, T_6\}$, cụ thể:

- $T_1 = \{A, B, D, E\}$
- $T_2 = \{B, C, E\}$
- $T_3 = \{A, B, D, E\}$
- $T_4 = \{A, B, C, E\}$
- $T_5 = \{A, B, C, D, E\}$
- $T_6 = \{B, C, D\}$

- Với $minsup = 3$.

PHƯƠNG PHÁP FP-GROWTH

Sắp xếp lại các mục (items) để xây dựng cây FP–Tree

Tập tất cả các mục $\mathbb{I}$:

$\mathbb{I} = \{B(6), E(5), A(4), C(4), D(4)\}$

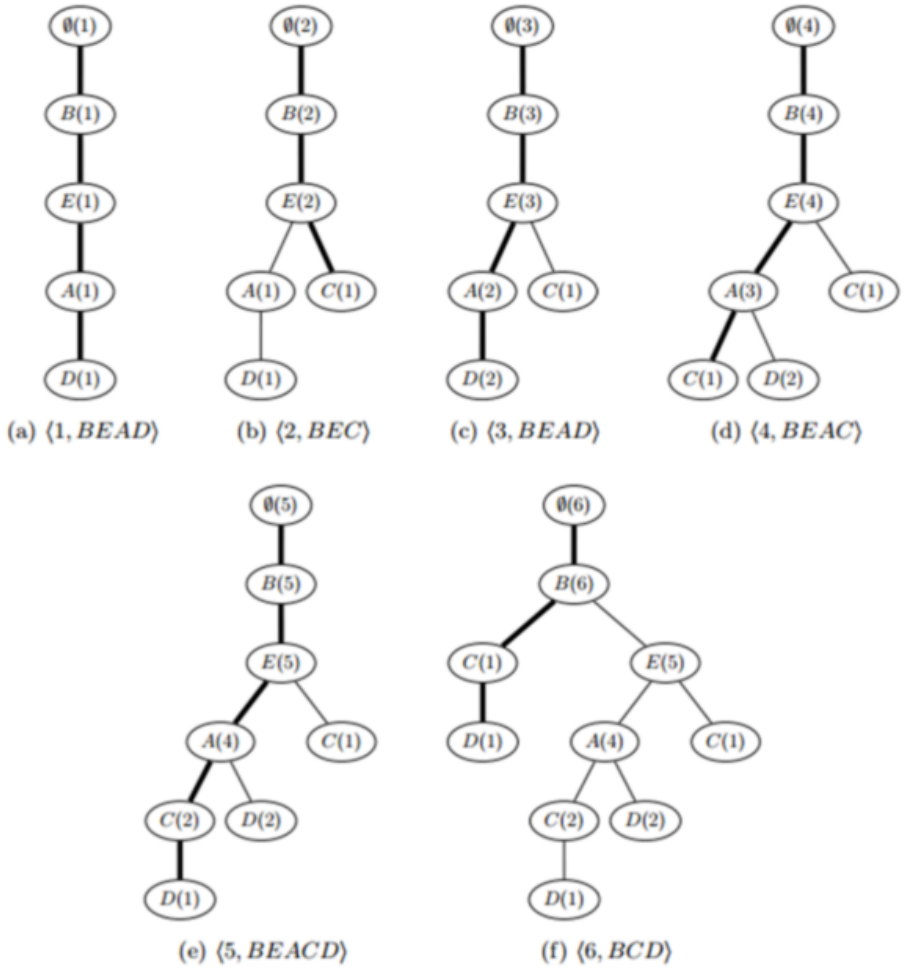
Cơ sở dữ liệu giao dịch $\mathbb{D}$:

$\mathbb{D} = \{T_1, T_2, T_3, T_4, T_5, T_6\}$, cụ thể:

- $T_1 = \{B, E, A, D\}$
- $T_2 = \{B, E, C\}$
- $T_3 = \{B, E, A, D\}$
- $T_4 = \{B, E, A, C\}$
- $T_5 = \{B, E, A, C, D\}$
- $T_6 = \{B, C, D\}$

PHƯƠNG PHÁP FP-GROWTH

Minh họa thuật toán sinh cây FP–Tree $\mathbb{T}$ từ CSDL $\mathbb{D}$



[Nguồn: Data Mining and Analysis: Fundamental Concepts and Algorithms by Zaki and Jr]

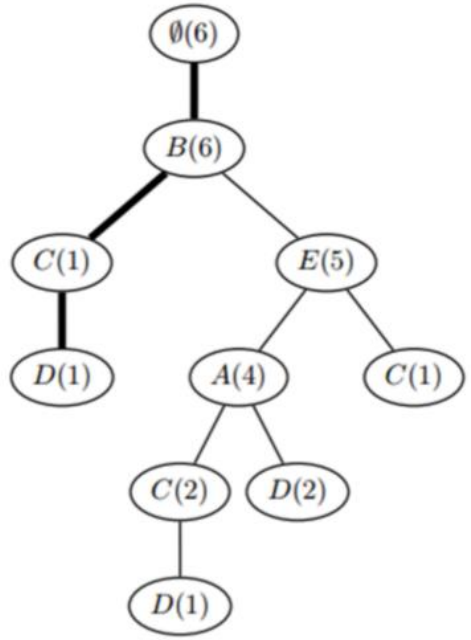
PHƯƠNG PHÁP FP-GROWTH

Một vài đặc điểm của cây FP-Tree

- Chỉ cần quét toàn bộ cơ sở dữ liệu $\mathbb{D}$ **2** lần để xây dựng cây FP–Tree $\mathbb{T}$.
- Cây FP–Tree là một dạng biểu diễn cô đọng (compressed) của cơ sở dữ liệu giao dịch $\mathbb{D}$.
- Cây FP–Tree càng nhỏ gọn càng tốt.
- Các mục (items) càng phổ biến (có độ hỗ trợ cao) càng nằm phía gần gốc cây.
- Tất cả các tập phổ biến (frequent itemsets) có thể được khai phá trực tiếp từ cây FP–Tree $\mathbb{T}$ thay vì từ CSDL $\mathbb{D}$.

PHƯƠNG PHÁP FP-GROWTH

Cây FP–Tree $\mathbb{T}$ được xây dựng từ CSDL $\mathbb{D}$



[Nguồn: Data Mining and Analysis: Fundamental Concepts and Algorithms by Zaki and Jr]

Thuật toán đệ quy sinh các tập phổ biến từ cây FP-Tree $\mathbb{T}$

```
1: procedure FPGrowth(𝕋, P, 𝔽, minsup)
2:     Loại bỏ các mục không phổ biến (infrequent items) trong 𝕋;
3:     if (IsPath(𝕋)) then
4:         for (với mỗi tập con Y ⊆ 𝕋) do
5:             X ← P ∪ Y;
6:             X.support ← min_{x∈Y}{cnt(x)};
7:             𝔽 ← 𝔽 ∪ {X};
8:         end for
9:     else
10:        for (mỗi mục y ∈ 𝕋 với thứ tự đã sắp xếp tăng dần theo sup(y)) do
11:            X ← P ∪ {y};
12:            X.support ← sup(y);      ▷ sup(y) là tổng cnt(y) tại mọi nốt có nhãn y trong 𝕋
13:            𝔽 ← 𝔽 ∪ {X};
14:            Khởi tạo FP–Tree 𝕋_X ← ∅;
15:            for (với mỗi đường đi path từ gốc xuống nốt có nhãn y trong cây 𝕋) do
16:                cnt(y) ← đếm tần suất của y trong path;
17:                Chèn path (ngoại trừ nốt y) vào cây FP-Tree 𝕋_X với cnt(y);
18:            end for
19:            if (𝕋_X ≠ ∅) then
20:                FPGrowth(𝕋_X, X, 𝔽, minsup);
21:            end if
22:        end for
23:    end if
24: end procedure
```

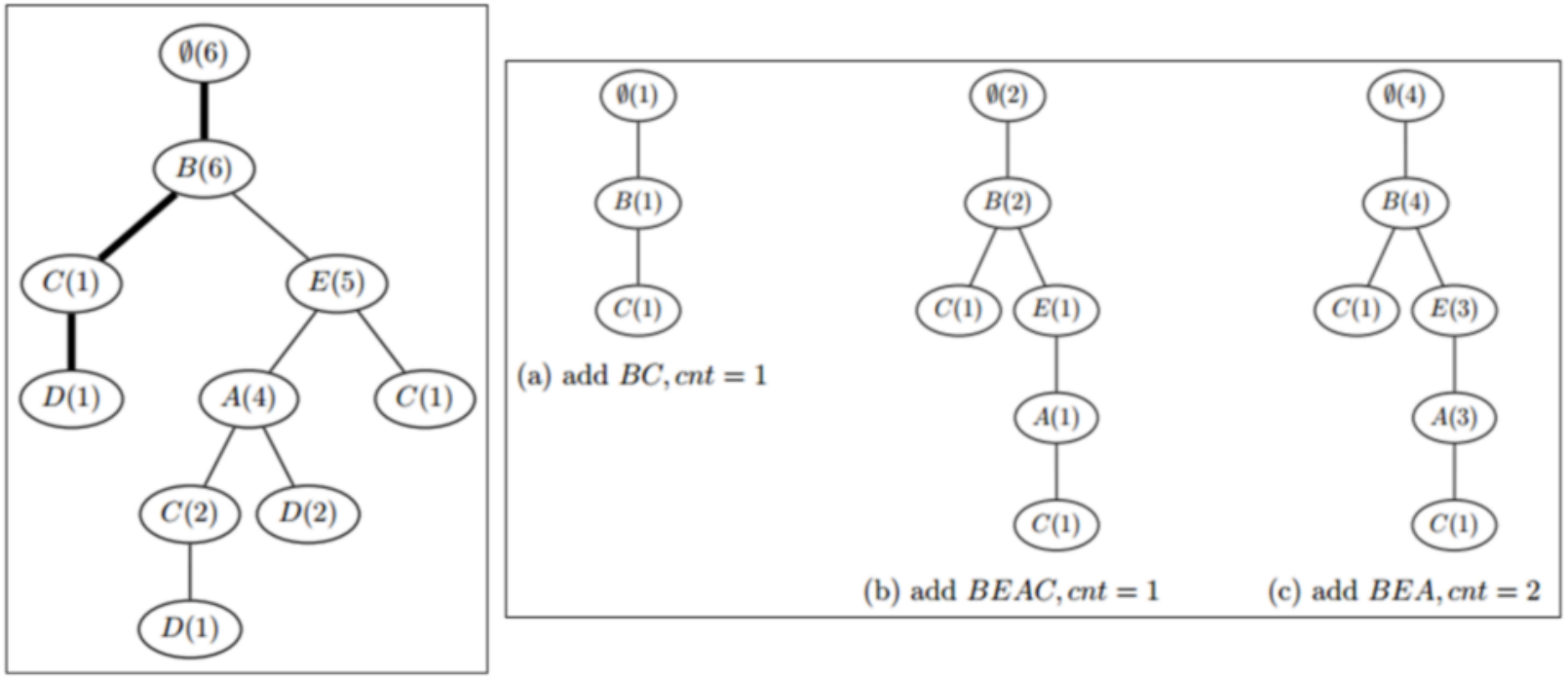
PHƯƠNG PHÁP FP-GROWTH

Sinh tập phổ biến từ FP-Tree: một số khái niệm

- Lời gọi hàm đầu tiên FPGrowth($\mathbb{T}$, $P \leftarrow \emptyset$, $\mathbb{F} \leftarrow \emptyset$, *minsup*).
- Phép chiếu chọn (projection) cây FP–Tree $\mathbb{T}$ theo một mục (item) nào đó.
- Cây FP–Tree $\mathbb{T}$ có thể là một đường tuyến tính (*path*).
- Loại bỏ các mục không phổ biến (infrequent items) trong một cây FP–Tree.

PHƯƠNG PHÁP FP-GROWTH

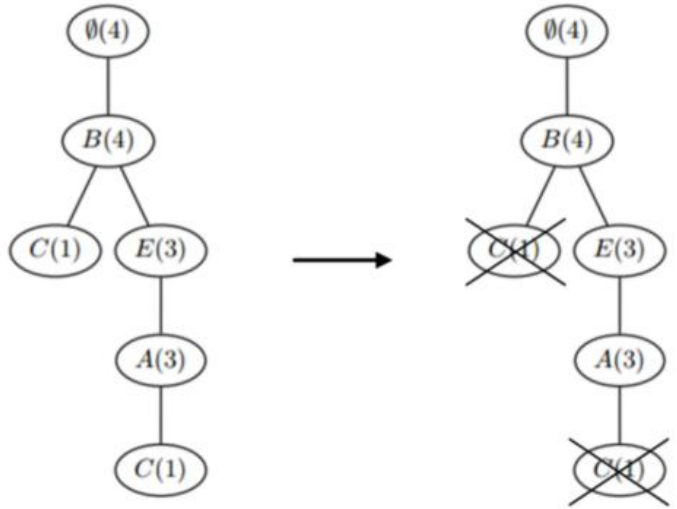
Cây FP–Tree chiếu chọn (projected) theo mục (item) *D*



[Nguồn: Data Mining and Analysis: Fundamental Concepts and Algorithms by Zaki and Jr]

PHƯƠNG PHÁP FP-GROWTH

Loại bỏ các mục không phổ biến (infrequent items) trong FP–Tree



- Bên trái: cây FP–Tree $\mathbb{T}_D$ chiếu theo mục D từ cây FP–Tree $\mathbb{T}$.
- Bên phải: Cây FP–Tree $\mathbb{T}_D$ sau khi đã loại bỏ mục C không phổ biến do $cnt(C) = 1 + 1 = 2 < minsup = 3$.

Minh họa thuật toán FP–Growth

- Với lời gọi đầu tiên: FPGrowth($\mathbb{T}$, $P \leftarrow \emptyset$, $\mathbb{F} \leftarrow \emptyset$, $minsup = 3$).
  - Không xóa bỏ được mục không phổ biến nào (tất cả đều phổ biến).
  - $\mathbb{T}$ không phải dạng đường tuyến tính *path*.
  - Tiền tố (prefix) $P = \emptyset$.
  - y sẽ lần lượt nhận $D(4), C(4), A(4), E(5), B(6)$.
  - Trước tiên y nhận D:
    - $X \leftarrow P \cup \{y\} = \emptyset \cup \{D\} = \{D\}$.
    - $\mathbb{F} \leftarrow \mathbb{F} \cup \{X\} = \emptyset \cup \{\{D(4)\}\} = \{\{D(4)\}\}$.
    - Có 3 đường đi tuyến tính (*path*) từ gốc của $\mathbb{T}$ đến nốt D: BCD, $cnt(D) = 1$; $BEACD$, $cnt(D) = 1$; và $BEAD$, $cnt(D) = 2$.
    - Tạo cây FP–Tree $\mathbb{T}_{\{D\}}$ từ 3 paths nói trên.
    - Gọi đệ quy hàm FPGrowth($\mathbb{T}_{\{D\}}$, $\{D\}$, $\{\{D(4)\}\}$, $minsup = 3$).
  - y nhận C:
    - ...
  - y nhận A:
    - ...
  - y nhận E:
    - ...
  - y nhận B:
    - ...

PHƯƠNG PHÁP FP-GROWTH

Minh họa thuật toán FP–Growth (2)

- Với lời gọi FPGrowth($\mathbb{T}_{\{D\}}$, $P = \{D\}$, $\mathbb{F} = \{\{D(4)\}\}$, $minsup = 3$):
  - Loại bỏ tất cả nốt C ra khỏi $\mathbb{T}_{\{D\}}$ vì $cnt(C) = 1 + 1 = 2 < minsup = 3$.
  - Cây FP–Tree $\mathbb{T}_{\{D\}}$ bây giờ trở thành một đường tuyến tính (*path*): $B(4)$ – $E(3)$ – $A(3)$:
    - ★ Liệt kê tất cả các tập con của đường tuyến tính: B, E, A, BE, BA, EA, BEA.
    - ★ Ghép với tiền tố $P = \{D\}$ tạo thành các tập phổ biến $DB(4)$, $DE(3)$, $DA(3)$, $DBE(3)$, $DBA(3)$, $DEA(3)$, $DBEA(3)$.
    - ★ Thêm các tập phổ biến vào trong $\mathbb{F}$ ta được $\mathbb{F} = \{D(4), DB(4), DE(3), DA(3), DBE(3), DBA(3), DEA(3), DBEA(3)\}$.
    - ★ Lời gọi FPGrowth($\mathbb{T}_{\{D\}}$, $P = \{D\}$, $\mathbb{F} = \{\{D(4)\}\}$, $minsup = 3$) kết thúc.

PHƯƠNG PHÁP FP-GROWTH

Minh họa thuật toán FP–Growth (3)

- Khi y nhận các mục khác:
  - y nhận C:
    - ★ $\mathbb{F} = \{D(4), DB(4), DE(3), DA(3), DBE(3), DBA(3), DEA(3), DBEA(3)\} \cup \underline{\{C(4), CB(4), CE(3), CBE(3)\}}$.
  - y nhận A:
    - ★ $\mathbb{F} = \{D(4), DB(4), DE(3), DA(3), DBE(3), DBA(3), DEA(3), DBEA(3)\} \cup \{C(4), CB(4), CE(3), CBE(3)\} \cup \underline{\{A(4), AE(4), AB(4), AEB(4)\}}$.
  - y nhận E:
    - ★ $\mathbb{F} = \{D(4), DB(4), DE(3), DA(3), DBE(3), DBA(3), DEA(3), DBEA(3)\} \cup \{C(4), CB(4), CE(3), CBE(3)\} \cup \{A(4), AE(4), AB(4), AEB(4)\} \cup \underline{\{E(5), EB(5)\}}$.
  - y nhận B:
    - ★ $\mathbb{F} = \{D(4), DB(4), DE(3), DA(3), DBE(3), DBA(3), DEA(3), DBEA(3)\} \cup \{C(4), CB(4), CE(3), CBE(3)\} \cup \{A(4), AE(4), AB(4), AEB(4)\} \cup \{E(5), EB(5)\} \cup \underline{\{B(6)\}}$.

PHƯƠNG PHÁP FP-GROWTH

Minh họa thuật toán FP–Growth (4)

- Vậy $\mathbb{F}$ bao gồm các tập phổ biến với các mức hỗ trợ khác nhau:
  - Support $= 6$: B
  - Support $= 5$: E, BE
  - Support $= 4$: D, C, A, DB, CB, AE, AB, ABE
  - Support $= 3$: DE, DA, CE, DBE, DBA, DAE, CBE, $DBEA$

PHƯƠNG PHÁP FP-GROWTH: ƯU và NHƯỢC ĐIỂM

- Ưu điểm:
  - Nén được cơ sở dữ liệu trong một cấu trúc cây gọn nhẹ FP–Tree.
  - Chỉ cần quét cơ sở dữ liệu 2 lần.
  - Hiệu quả kể cả khi ngưỡng *minsup* bé.
- Nhược điểm:
  - Thuật toán cài đặt phức tạp hơn so với Apriori.
  - Khi cơ sở dữ liệu lớn: FP–Tree lớn và khó lưu vừa trong bộ nhớ.
  - Sử dụng đệ quy (có thể khử đệ quy).

THE END