Fall 2004, CIS, Temple University

CIS527: Data Warehousing, Filtering, and Mining

Lecture 4

- Tutorial: Connecting SQL Server to Matlab using Database Matlab Toolbox
- Association Rule MIning

Lecture slides taken/modified from:

- Jiawei Han (http://www-sal.cs.uiuc.edu/~hanj/DM_Book.html)
- Vipin Kumar (http://www-users.cs.umn.edu/~kumar/csci5980/index.html)

Motivation: Association Rule Mining

- Cho trước một tập các giao dịch, tìm các luật có thể tiên đoán sự xuất hiện của một mặt hàng này dựa trên sự xuất hiện của các mặt hàng khác trong giao dịch đó

Market-Basket transactions

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Association Rules

{Diaper} $\rightarrow$ {Beer},
{Milk, Bread} $\rightarrow$ {Eggs,Coke},
{Beer, Bread} $\rightarrow$ {Milk},

Implication means co-occurrence, not causality!

Applications: Association Rule Mining

- * $\Rightarrow$ Maintenance Agreement
  - What the store should do to boost Maintenance Agreement sales
- Home Electronics $\Rightarrow$ *
  - What other products should the store stocks up?
- Attached mailing in direct marketing
- Detecting “ping-ponging” of patients
- Marketing and Sales Promotion
- Supermarket shelf management

Definition: Frequent Itemset (Tập mặt hàng thường xuyên)

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
- **Support count (σ) (Độ đếm hỗ trợ)**
  - Tần số xuất hiện của 1 tập mặt hàng
  - E.g. σ({Milk, Bread,Diaper}) = 2
- **Support (Độ hỗ trợ)**
  - Tỷ lệ các giao dịch có chứa 1 tập mặt hàng nào đó
  - E.g. s({Milk, Bread, Diaper}) = 2/5
- **Frequent Itemset**
  - một tập mặt hàng có độ hỗ trợ >= một ngưỡng minsup (độ hỗ trợ nhỏ nhất)

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Definition: Association Rule (Luật kết hợp)

- **Association Rule**
  - An implication expression of the form X → Y, ở đó X,Y là tập các mặt hàng
  - Example:
    {Milk, Diaper} → {Beer}

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

- **Rule Evaluation Metrics**
  - Độ hỗ trợ (s)
    - Tỷ lệ các giao dịch chứa cả X và Y
  - Độ tin cậy (c)
    - Tần suất các mặt hàng của Y xuất hiện trong các giao dịch có chứa X
    - Ví dụ như trên thì là số lần Beer xuất hiện trong các giao dịch có chứa Milk và Diaper.

Example:

$$\{\text{Milk}, \text{Diaper}\} \Rightarrow \text{Beer}$$

$$s = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{|\text{T}|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

Nhiệm vụ khai phá tìm luật kết hợp

- Given a set of transactions T, the goal of association rule mining is to find all rules having
  - support ≥ *minsup* threshold
  - confidence ≥ *minconf* threshold

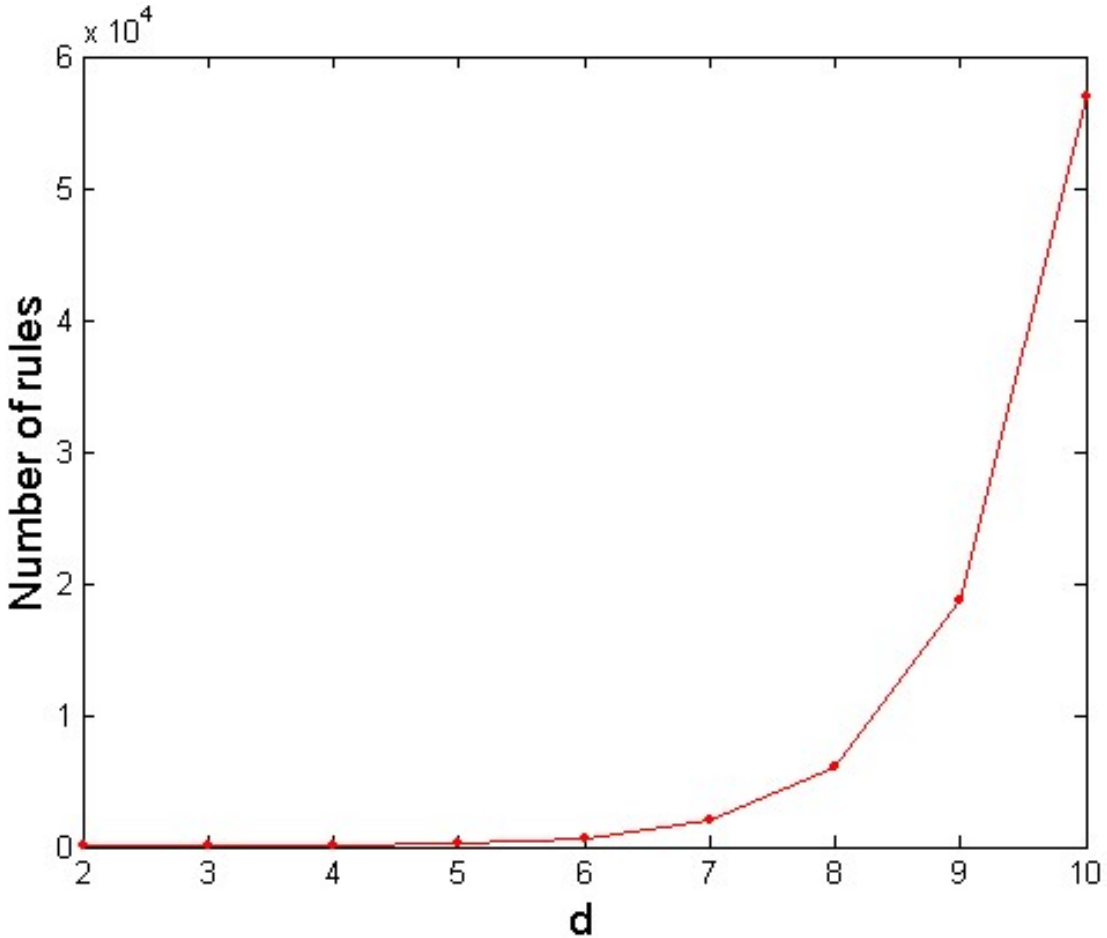
Các luật dưới ngưỡng minconf có thể được coi là không tin cậy và được bỏ đi.

- **Brute-force approach (Cách tiếp cận vét cạn)**:
  - List all possible association rules
  - Compute the support and confidence for each rule
  - Prune rules that fail the *minsup* and *minconf* thresholds

  ⇒ Computationally prohibitive!

Computational Complexity

- Given d unique items:
  - Total number of itemsets = 2^d
  - Total number of possible association rules:



$$R = \sum_{k=1}^{d-1}\left[\binom{d}{k} \times \sum_{j=1}^{d-k}\binom{d-k}{j}\right]$$

$$= 3^d - 2^{d+1} + 1$$

If d=6, R = 602 rules

Mining Association Rules: Decoupling

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Rules:

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observations:

- All the above rules are binary partitions of the same itemset:
  {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may **decouple** the support and confidence requirements

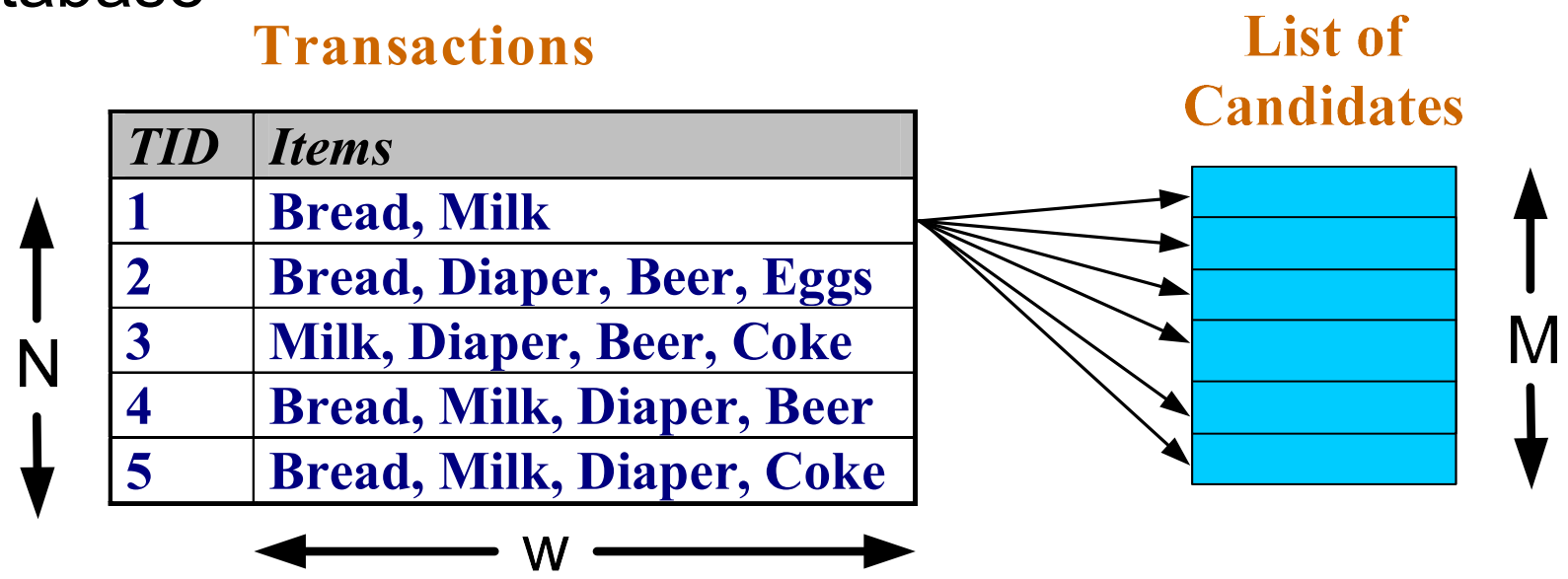
Mining Association Rules

- **Two-step approach (cách tiếp cận 2 bước)**:
  1. Frequent Itemset Generation
     - Sinh tất cả các tập mặt hàng có độ hỗ trợ (s) ≥ *minsup*
  2. Rule Generation
     - Sinh các luật bằng cách sinh ra các luật từ tập các mặt hàng thường xuyên có độ tin cậy lớn hơn trong đó mỗi luật chính là một sự phân chia nhị phân (làm hai phần) của một tập mặt hàng thường xuyên

- Frequent itemset generation is still computationally **expensive**

Frequent Itemset Generation

- Brute-force approach:
  - Each itemset in the lattice is a candidate frequent itemset
  - Count the support of each candidate by scanning the database



  - Match each transaction against every candidate
  - Complexity ~ O(NMw) => Expensive since $M = 2^d$!!!

Frequent Itemset Generation Strategies

- Reduce the number of candidates (M)
  - Complete search: $M=2^d$
  - Use pruning techniques to reduce M
- Reduce the number of transactions (N)
  - Reduce size of N as the size of itemset increases
  - Use a subsample of N transactions
- Reduce the number of comparisons (NM)
  - Use efficient data structures to store the candidates or transactions
  - No need to match every candidate against every transaction

Reducing Number of Candidates: Apriori

- Nguyên tắc Apriori:
  - Nếu một tập các mặt hàng là thường xuyên thì tất cả các tập con của nó cũng là thường xuyên

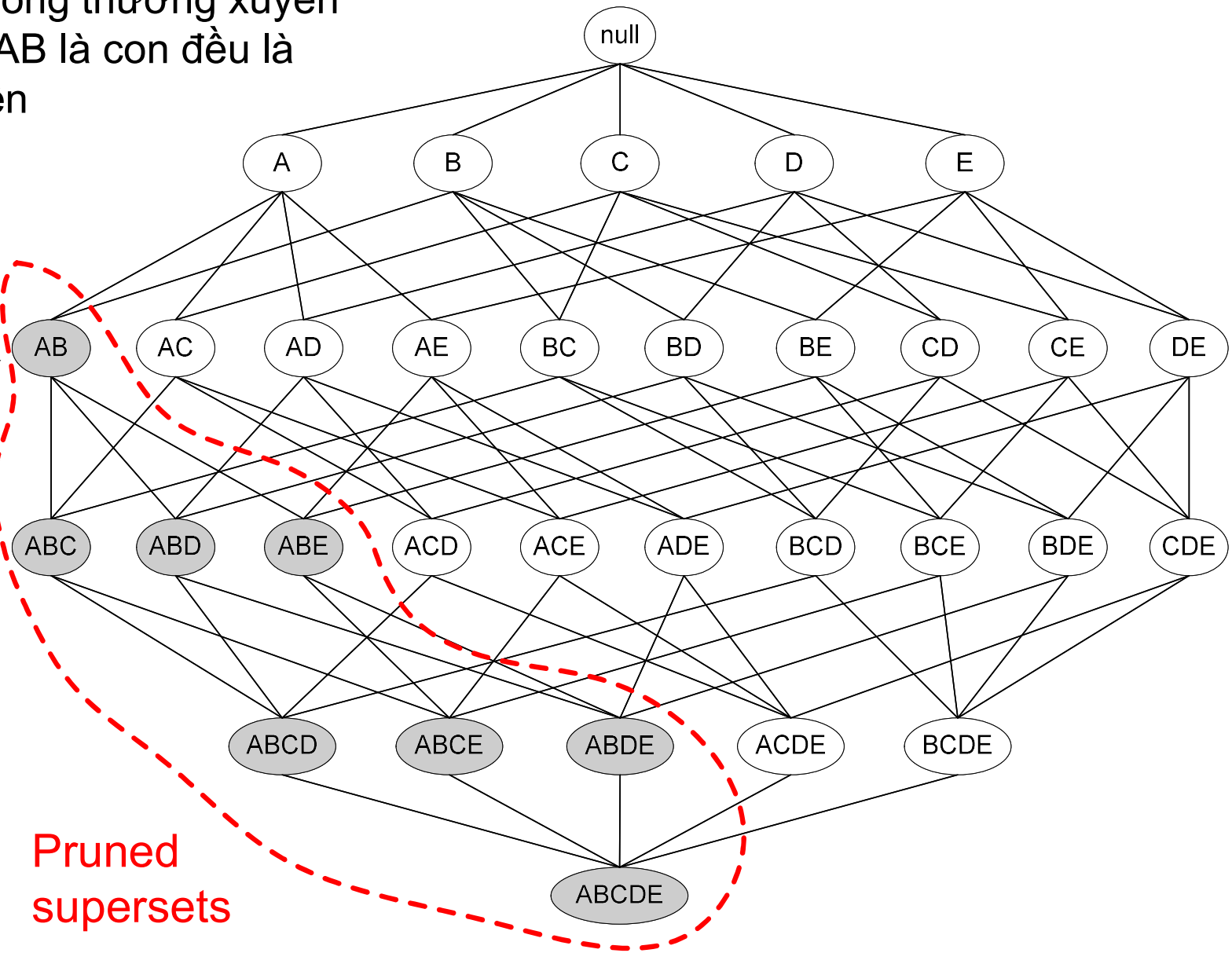
- Nguyên lý Apriori đúng đắn bởi thuộc tính sau đây của độ hỗ trợ là đúng:

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

  - Support of an itemset never exceeds the support of its subsets
  - This is known as the anti-monotone property of support

Illustrating Apriori Principle

AB được thấy là không thường xuyên
=> tất cả tập chứa AB là con đều là không thường xuyên
=> cắt bỏ toàn bộ



Illustrating Apriori Principle

Item	Count
Bread	**4**
Coke	**2**
Milk	**4**
Beer	**3**
Diaper	**4**
Eggs	**1**

Items (1-itemsets)

Itemset	Count
{Bread,Milk}	**3**
{Bread,Beer}	**2**
{Bread,Diaper}	**3**
{Milk,Beer}	**2**
{Milk,Diaper}	**3**
{Beer,Diaper}	**3**

Pairs (2-itemsets)

Không cần phải sinh các tập liên quan tới Coke hoặc Eggs

Minimum Support = 3

Triplets (3-itemsets)

Itemset	Count
{Bread,Milk,Diaper}	**3**

If every subset is considered,

$${}^6C_1 + {}^6C_2 + {}^6C_3 = 41$$

With support-based pruning,

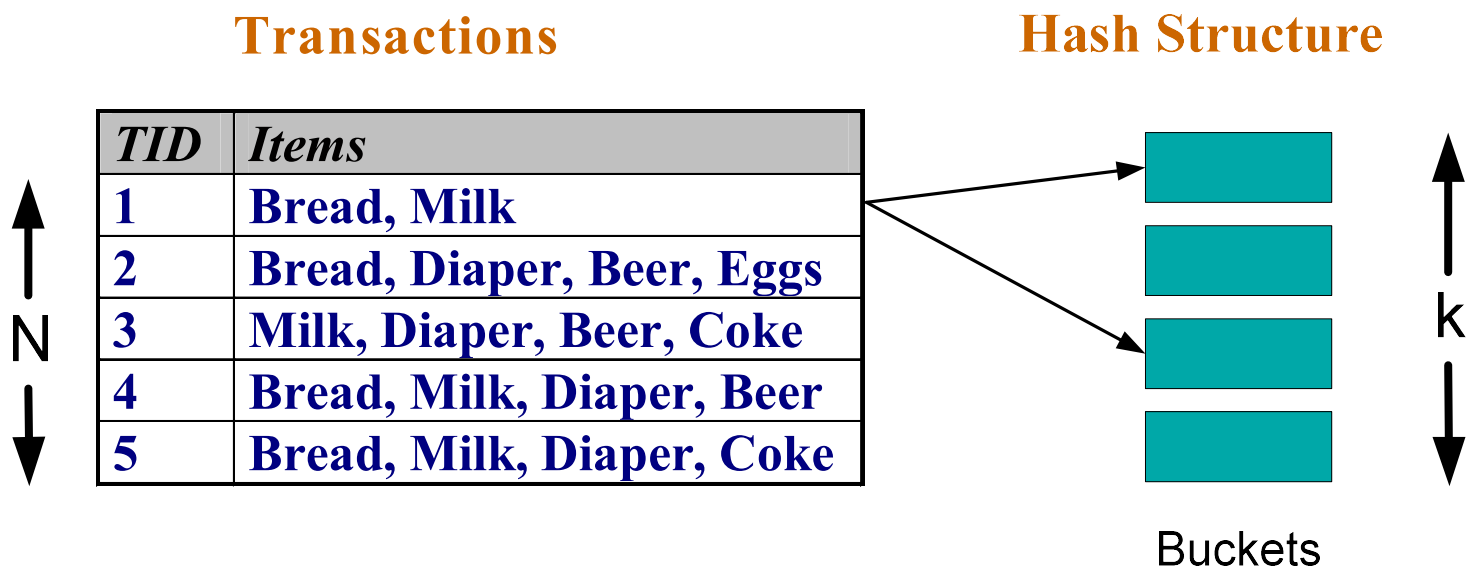
$$6 + 6 + 1 = 13$$

Apriori Algorithm

- Method:
  - Gán k=1
  - Sinh ra một tập mặt hàng với độ dài là 1
  - Lặp cho tới khi không còn tập mặt hàng mới nào được xác định
    - Sinh ra các tập mặt hàng với (k+1) phần tử từ các tập mặt hàng với k phần tử
    - Cắt bỏ những tập mặt hàng chứa tập con có độ dài k mà không phải là tập thường xuyên
    - Đếm độ hỗ trợ của mỗi ứng viên bằng cách quét toàn bộ cơ sở dữ liệu
    - Loại bỏ những ứng viên không phải thường xuyên, chỉ để lại những tập mặt hàng thường xuyên.

Apriori: Reducing Number of Comparisons (Reduce NM)

- Candidate counting:
  - Scan the database of transactions to determine the support of each candidate itemset
  - To reduce the number of comparisons, store the candidates in a hash structure
    - Instead of matching each transaction against every candidate, match it against candidates contained in the hashed buckets



Apriori: Implementation Using Hash Tree

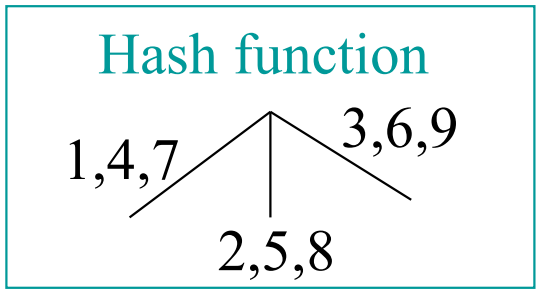
Suppose you have 15 candidate itemsets of length 3:

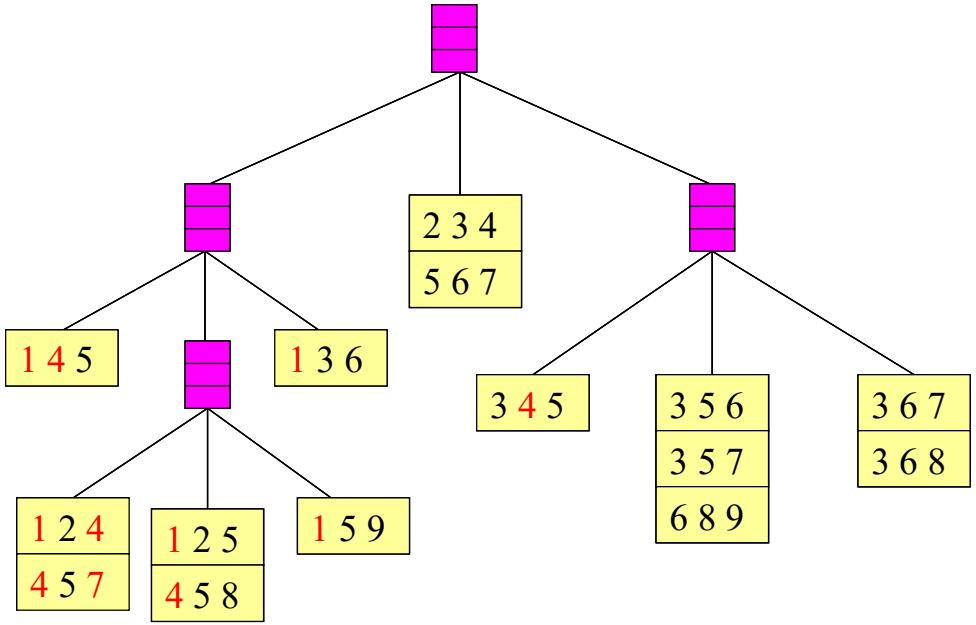
{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

You need:

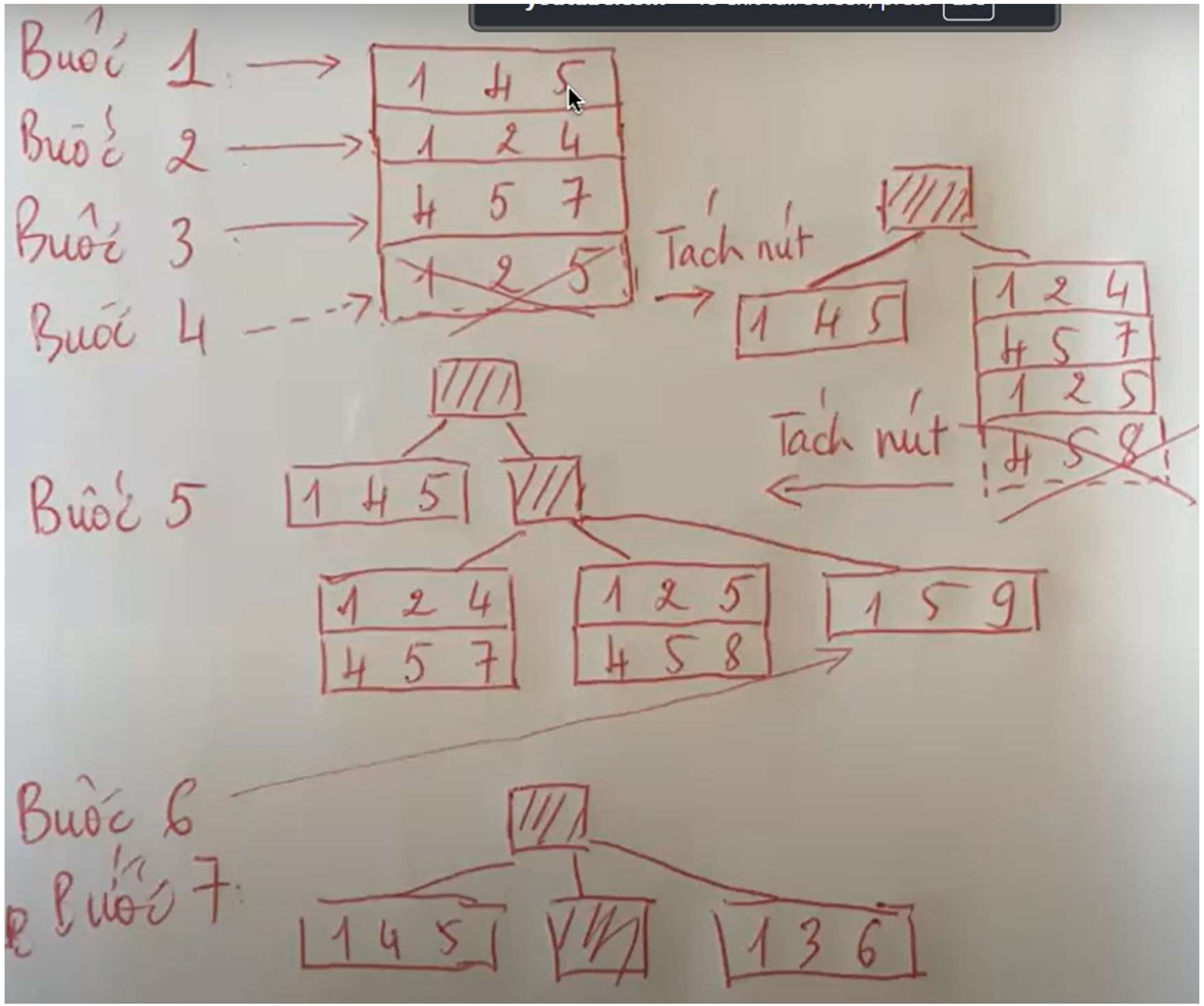
- Hash function
- Max leaf size: max number of itemsets stored in a leaf node

  (if number of candidate itemsets exceeds max leaf size, split the node)

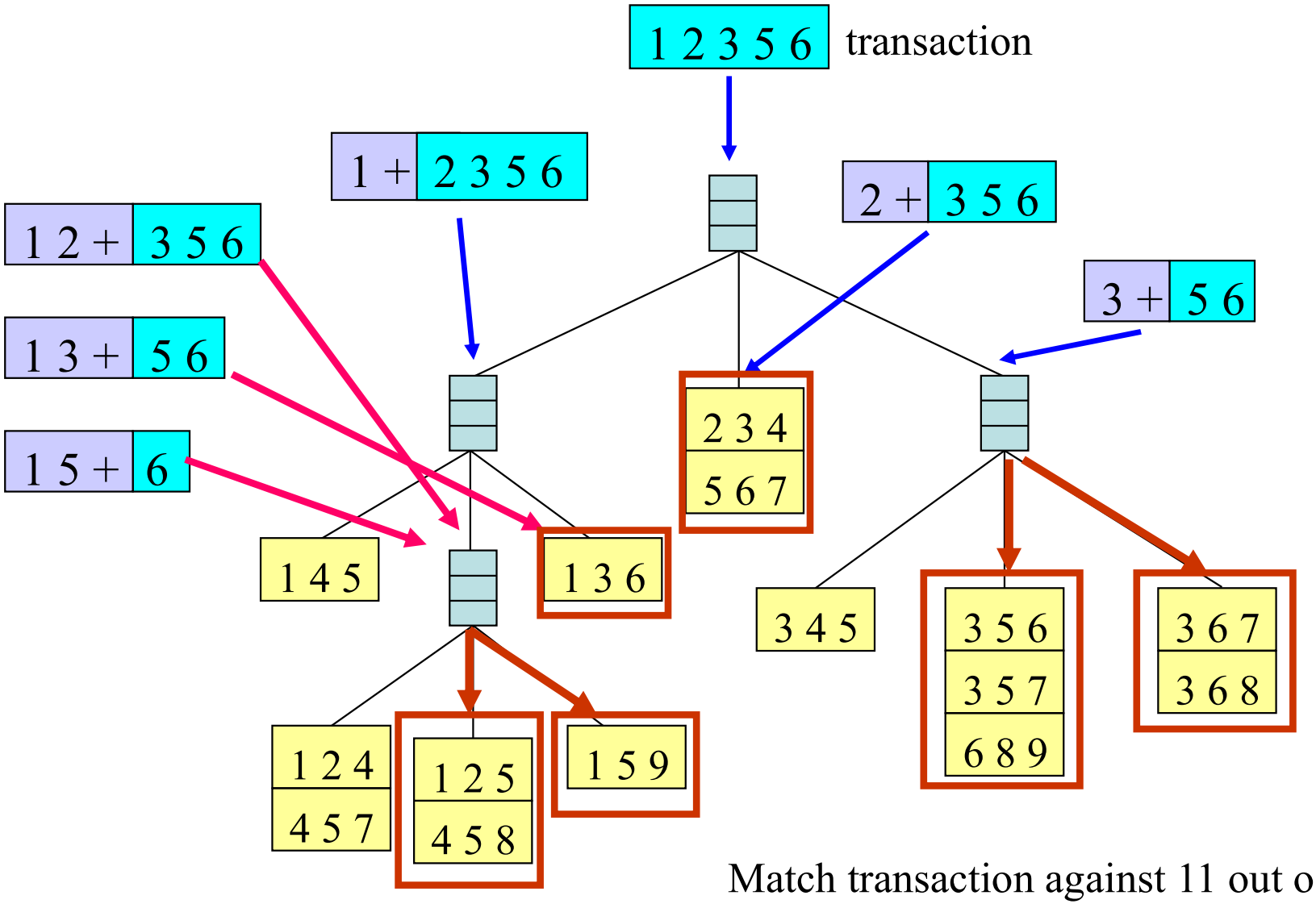




Apriori: Implementation Using Hash Tree



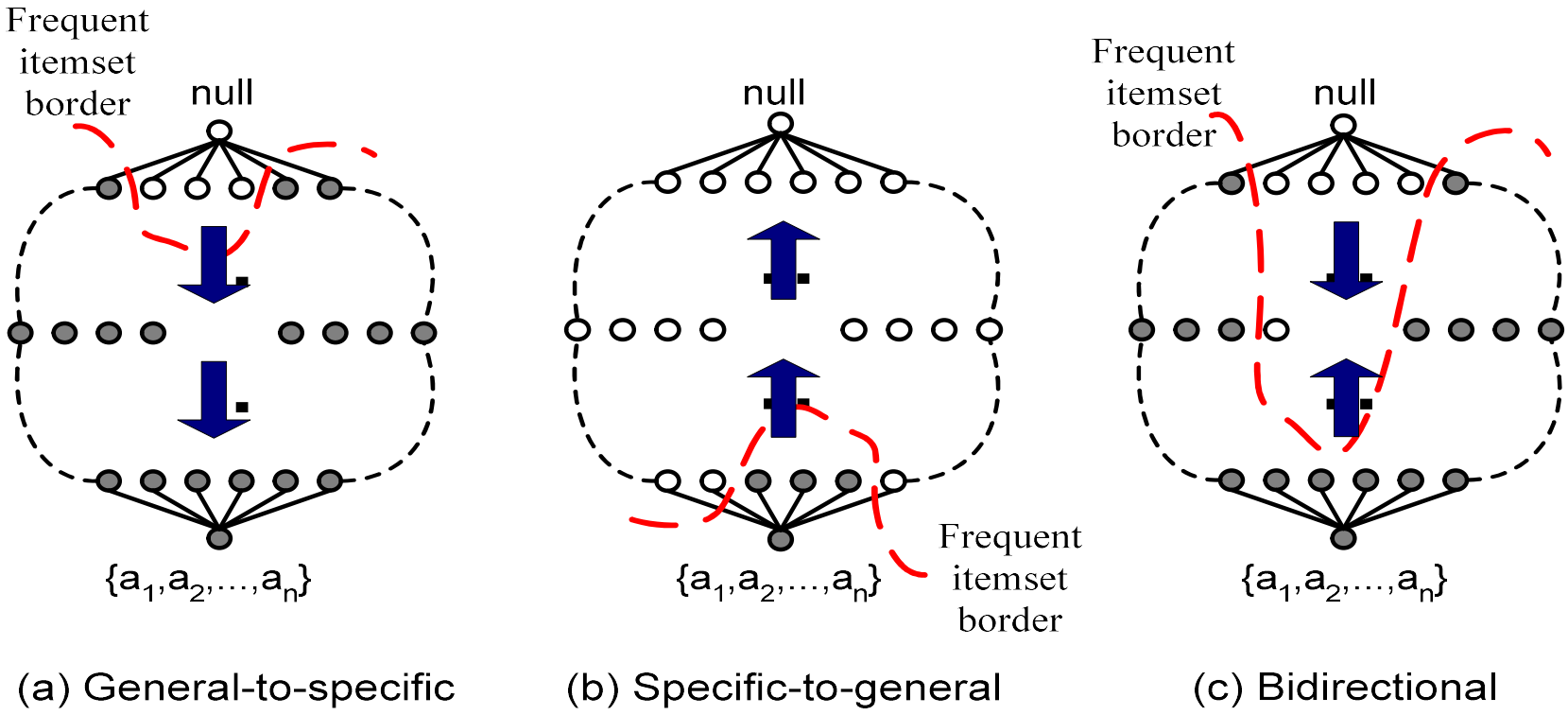
Apriori: Implementation Using Hash Tree



Match transaction against 11 out of 15 candidates

Apriori: Alternative Search Methods

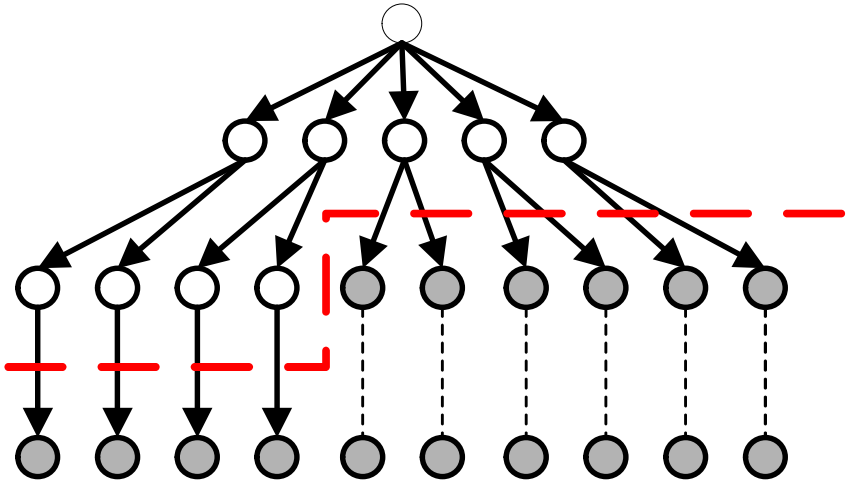
- Traversal of Itemset Lattice
  - General-to-specific vs Specific-to-general



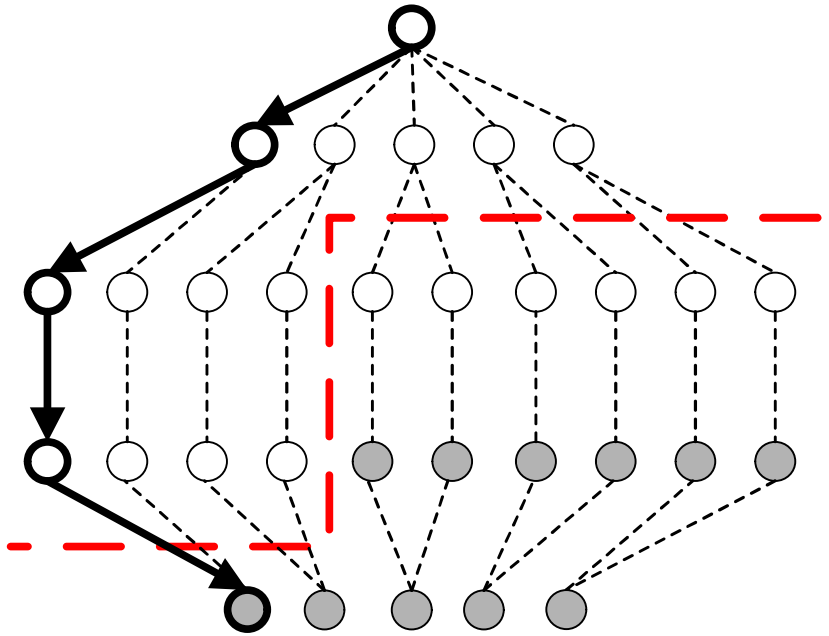
(a) General-to-specific (b) Specific-to-general (c) Bidirectional

Apriori: Alternative Search Methods

- Traversal of Itemset Lattice
  - Breadth-first vs Depth-first



(a) Breadth first



(b) Depth first

Bottlenecks of Apriori

- Candidate generation can result in huge candidate sets:
  - 10^4 frequent 1-itemset will generate 10^7 candidate 2-itemsets
  - To discover a frequent pattern of size 100, e.g., $\{a_1, a_2, \ldots, a_{100}\}$, one needs to generate $2^{100} \sim 10^{30}$ candidates.
- Multiple scans of database:
  - Needs (n +1) scans, n is the length of the longest pattern

ECLAT: Another Method for Frequent Itemset Generation

- ECLAT: for each item, store a list of transaction ids (tids); vertical data layout

Horizontal Data Layout

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

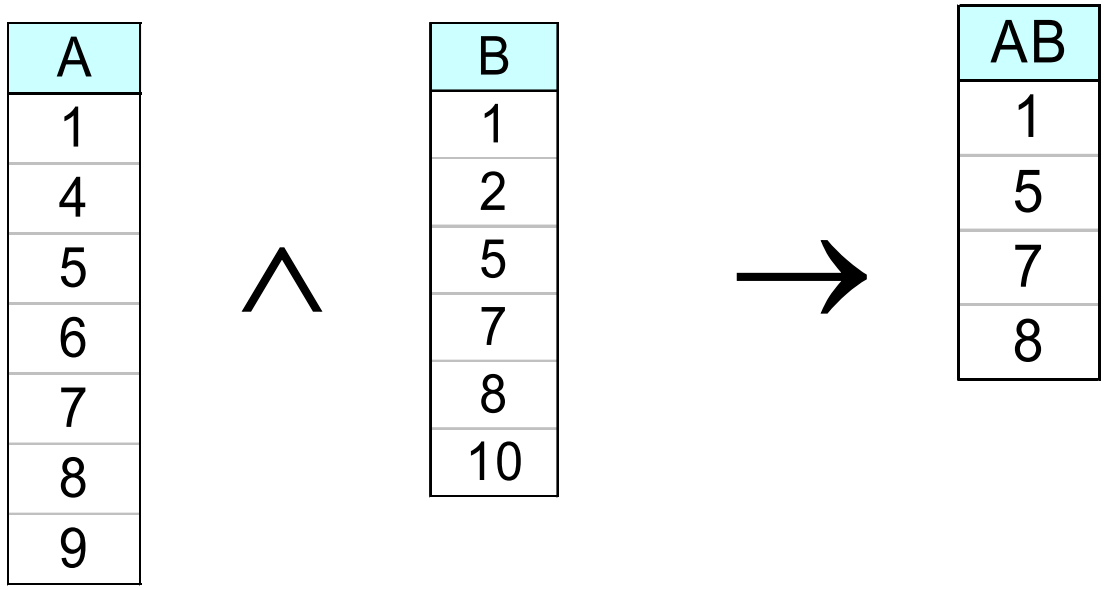
Vertical Data Layout

A	B	C	D	E
1	1	2	2	1
4	2	3	4	3
5	5	4	5	6
6	7	8	9	
7	8	9		
8	10			
9				

TID-list

ECLAT: Another Method for Frequent Itemset Generation

- Tính độ hỗ trợ của mỗi tập k mặt hàng bằng cách lấy giao của danh sách mã giao dịch của hai tập mặt hàng con có k-1 phần tử



- 3 traversal approaches:
  - top-down, bottom-up and hybrid
- Advantage: tính độ hỗ trợ khá nhanh
- Disadvantage: tốn bộ nhớ để lưu trữ danh sách các mã giao dịch trung gian

FP-growth: Another Method for Frequent Itemset Generation

- Use a compressed representation of the database using an FP-tree

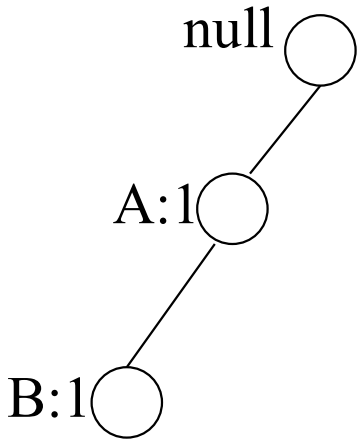
- Xây dựng 1 cây FP, sau đó nó sử dụng một cách tiếp cận chia để trị đệ quy để khai phá ra các tập mặt hàng thường xuyên

FP-Tree Construction

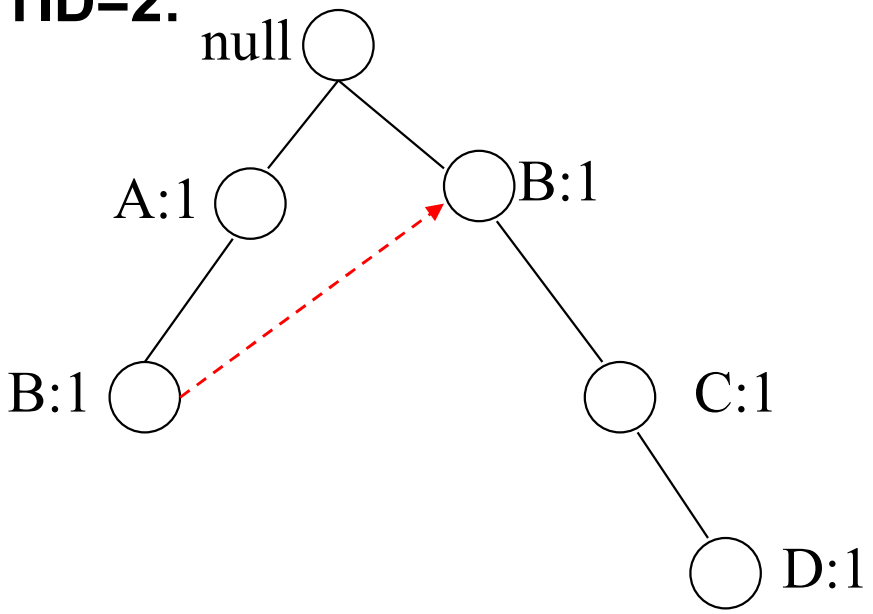
Nhớ phải sắp xếp các item trong từng TID trước khi xây cây.

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

After reading TID=1:



After reading TID=2:



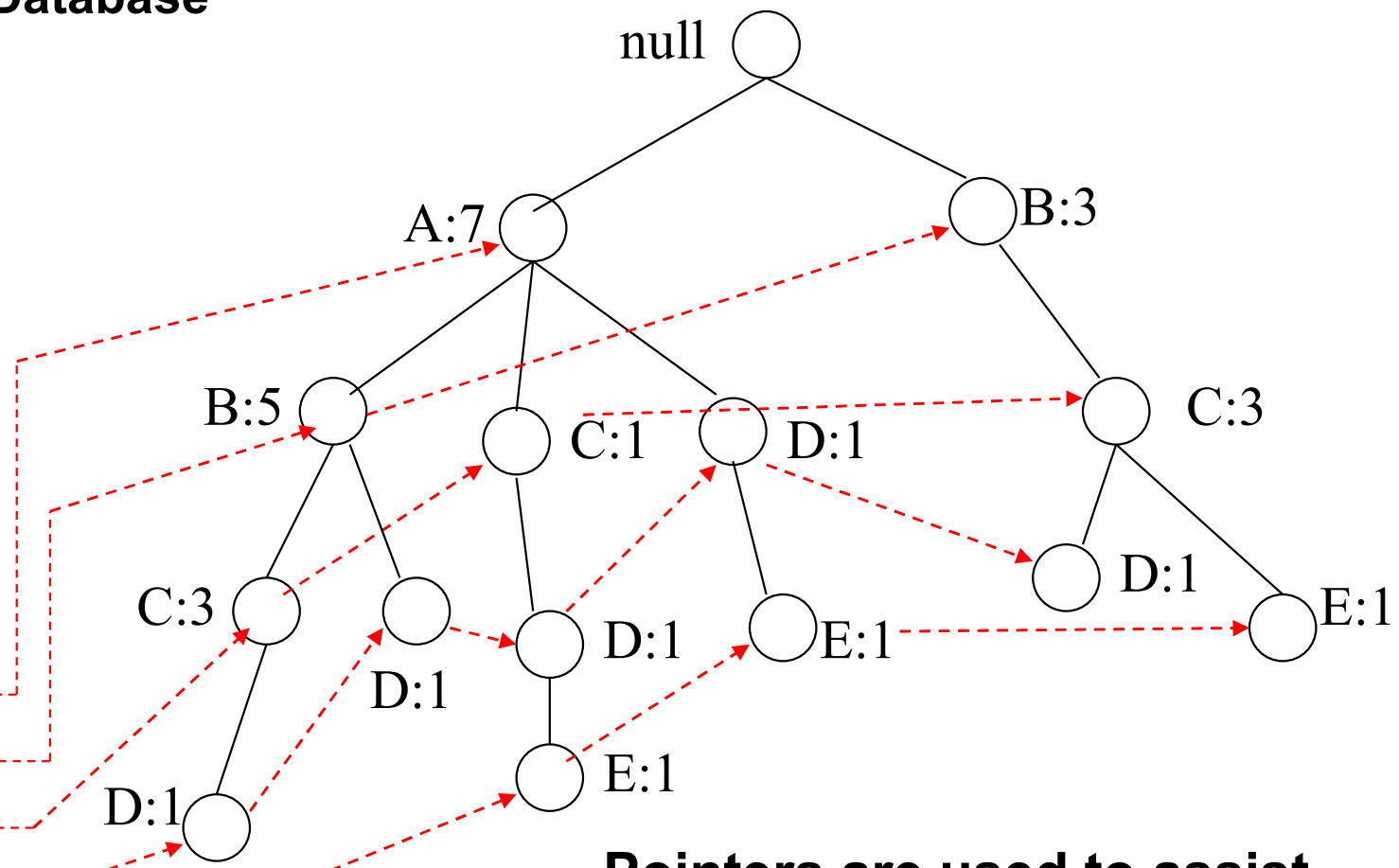
FP-Tree Construction

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

Transaction Database

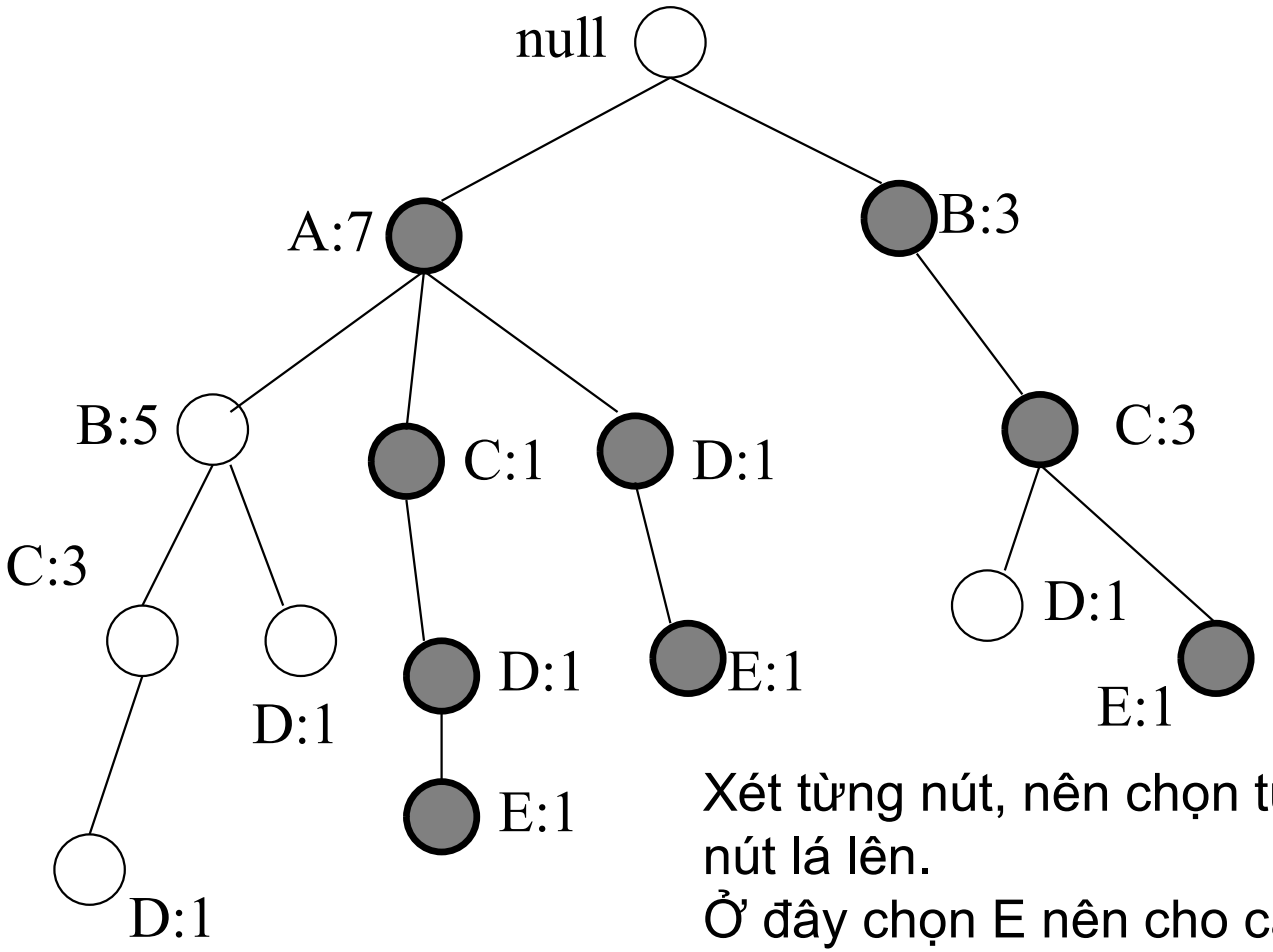
Header table

Item	Pointer
A	
B	
C	
D	
E	



Pointers are used to assist frequent itemset generation

FP-growth



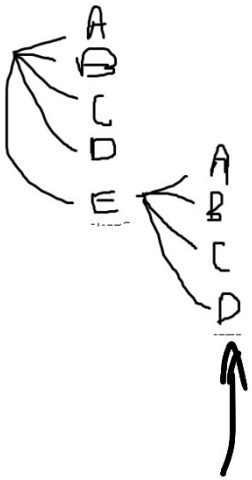
Xây dựng cây cơ sở điều kiện cho E:

P = {(A:1,C:1,D:1),
(A:1,D:1),
(B:1,C:1)}

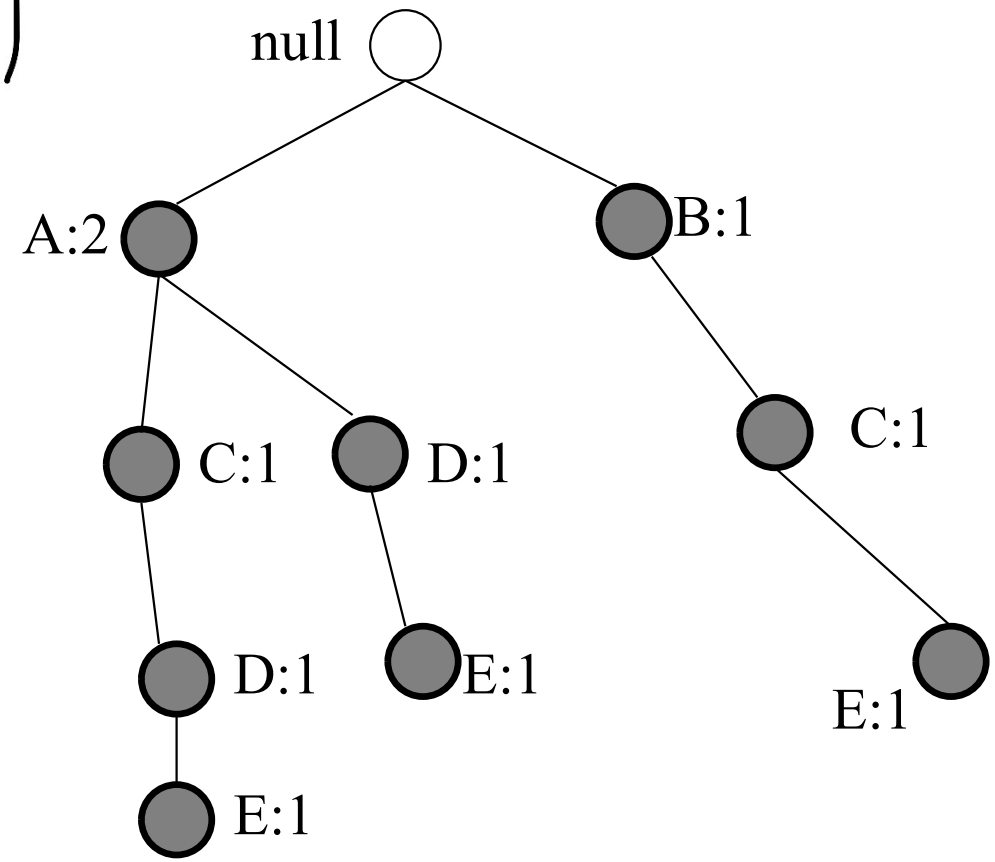
Recursively apply FP-growth on P

Xét từng nút, nên chọn từ nút lá lên.
Ở đây chọn E nên cho các nút còn lại vào stack

FP-growth



Cây điều kiện cho E:



Cây cơ sở điều kiện cho E:
P = {(A:1,C:1,D:1,E:1),
(A:1,D:1,E:1),
(B:1,C:1,E:1)}

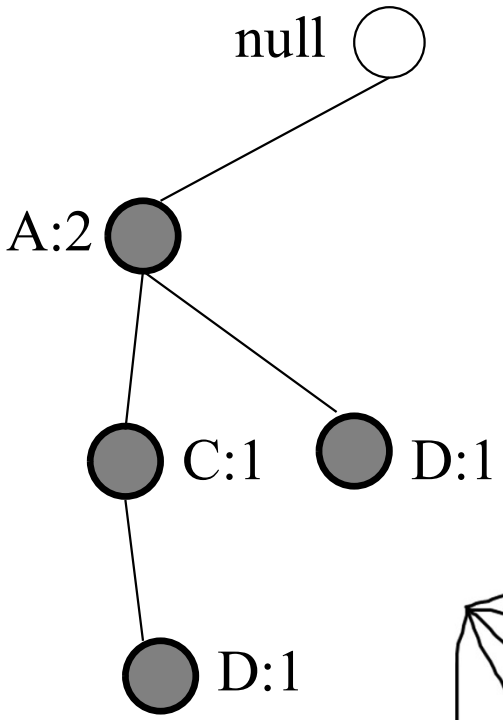
Count for E is 3: {E} is frequent itemset

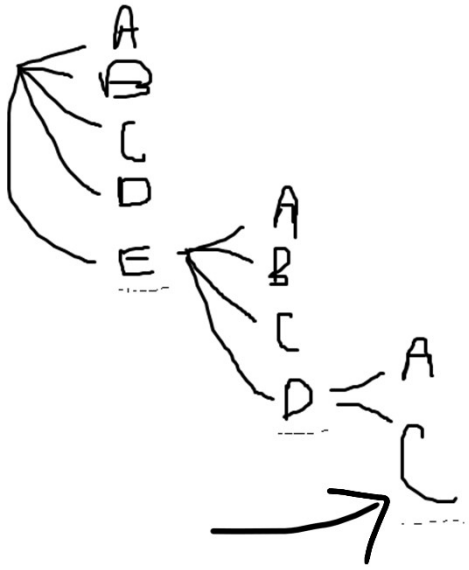
Recursively apply FP-growth on P

E xử lý xong, cắt E khỏi cây và đệ quy các nút khác trên cây này, có A, B, C và D có thể chọn, tiếp theo chọn nút D, bỏ còn lại vào stack

FP-growth

Cây điều kiện cho D nằm trong cây điều kiện cho E:





Conditional pattern base for D within conditional base for E:

P = {(A:1,C:1,D:1), (A:1,D:1)}

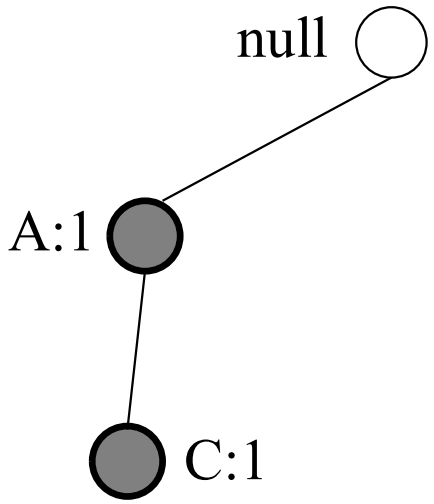
Count for D is 2: {D,E} is frequent itemset

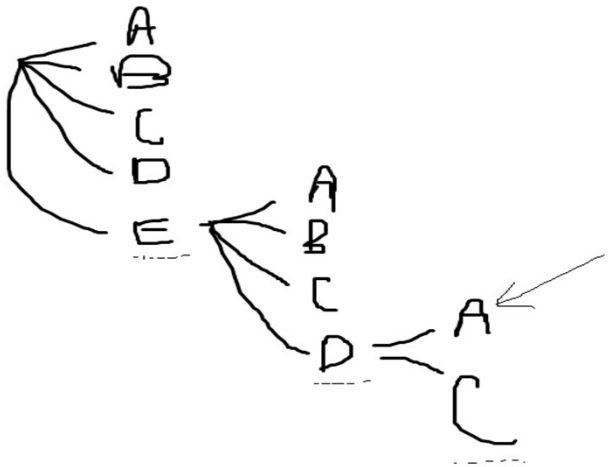
Recursively apply FP-growth on P

D xử lý xong, cắt D khỏi cây và đệ quy các nút khác trên cây này, có A và C có thể chọn, tiếp theo chọn nút C, bỏ A vào stack

FP-growth

Cây điều kiện cho C nằm trong cây điều kiện cho D nằm trong cây điều kiện cho E:





Conditional pattern base for C within D within E:

P = {(A:1,C:1)}

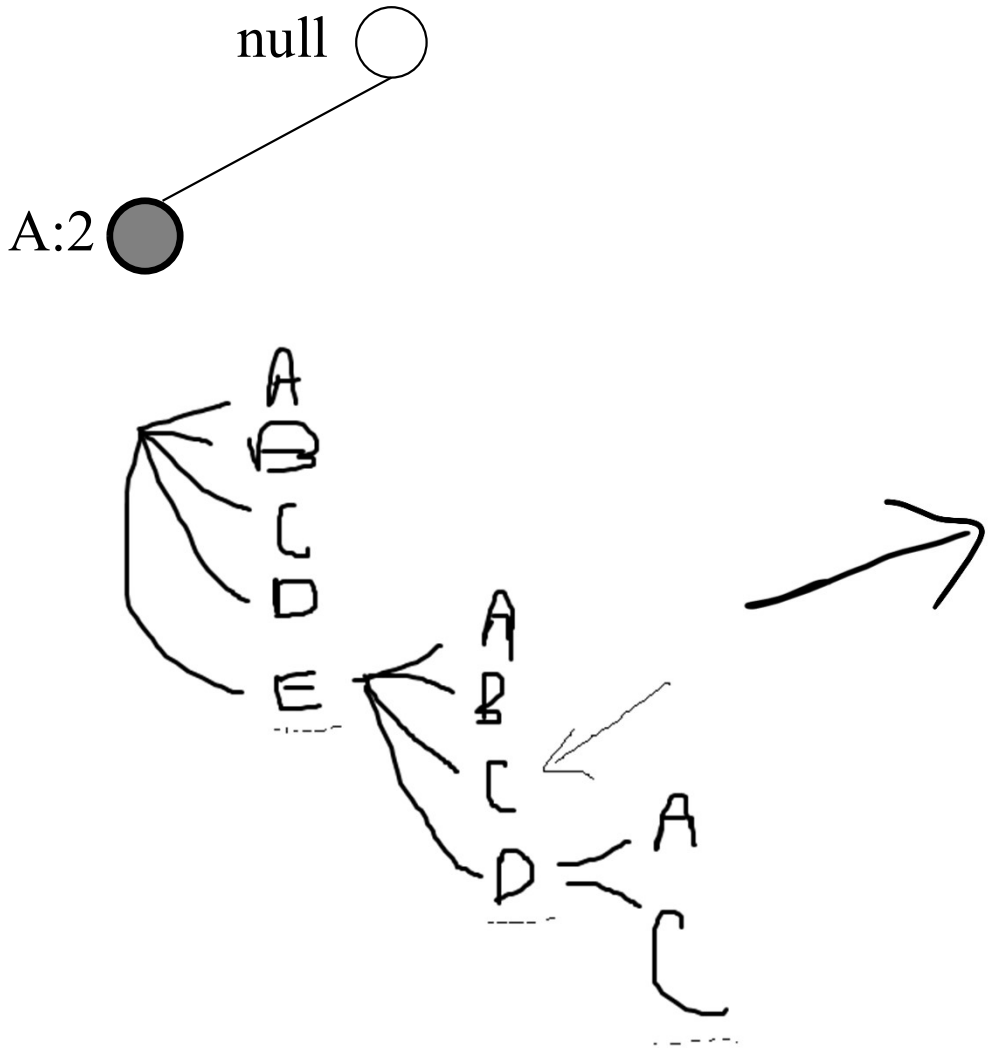
Count for C is 1: {C,D,E} is NOT frequent itemset

Dừng đệ quy bỏ nút này

Lấy A khỏi stack để tạo cây tiếp nằm trong D nằm trong E.

FP-growth

<u>Cây điều kiện cho A nằm trong cây điều kiện cho D nằm trong cây điều kiện cho E :</u>



Count for A is 2: {A,D,E} is frequent itemset

Hết các nút để đệ quy tiếp, bắt đầu nhả stack.

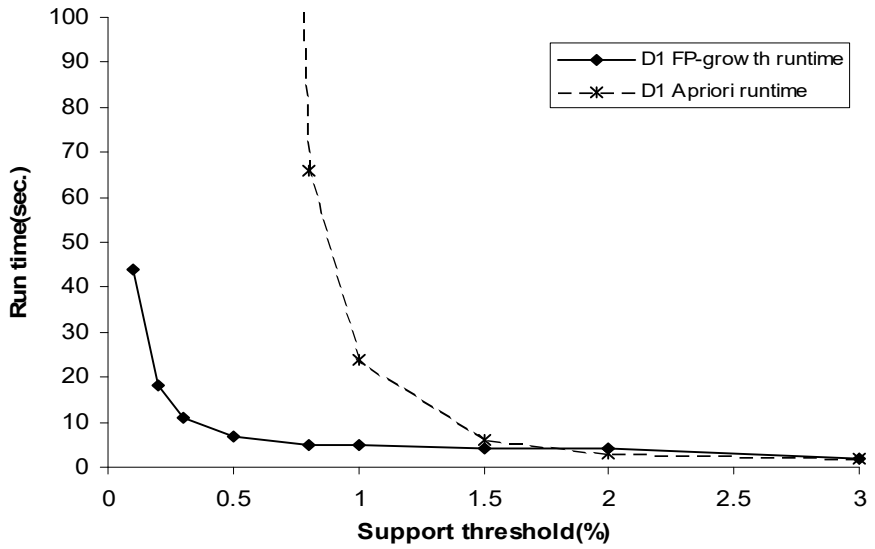
Next step:

Construct conditional tree C within conditional tree E

Continue until exploring conditional tree for A (which has only node A)

Benefits of the FP-tree Structure

- Performance study shows
  - **FP-growth is an order of magnitude faster than Apriori**, and is also faster than tree-projection
- Reasoning
  - Không phải tạo ứng cử viên, không phải kiểm tra ứng viên đó có phù hợp không
  - Sử dụng cấu trúc dữ liệu nén
  - Loại bỏ được việc duyệt CSDL nhiều lần
  - Phép toán cơ bản là đếm và xây dựng cây FP



Complexity of Association Mining

- Choice of minimum support threshold
  - lowering support threshold results in more frequent itemsets
  - this may increase number of candidates and max length of frequent itemsets
- Dimensionality (number of items) of the data set
  - more space is needed to store support count of each item
  - if number of frequent items also increases, both computation and I/O costs may also increase
- Size of database
  - since Apriori makes multiple passes, run time of algorithm may increase with number of transactions
- Average transaction width
  - transaction width increases with denser data sets
  - This may increase max length of frequent itemsets and traversals of hash tree (number of subsets in a transaction increases with its width)

Sinh luật kết hợp

- Cho một tập các mặt hàng thường xuyên L, hãy tìm tất cả các tập con f không rỗng của L sao cho f $\rightarrow$ L – f thỏa mãn yêu cầu về độ tin cậy nhỏ nhất
  - If {A,B,C,D} is a frequent itemset, candidate rules:

ABC $\rightarrow$D,	ABD $\rightarrow$C,	ACD $\rightarrow$B,	BCD $\rightarrow$A,
A $\rightarrow$BCD,	B $\rightarrow$ACD,	C $\rightarrow$ABD,	D $\rightarrow$ABC
AB $\rightarrow$CD,	AC $\rightarrow$ BD,	AD $\rightarrow$ BC,	BC $\rightarrow$AD,
BD $\rightarrow$AC,	CD $\rightarrow$AB,		

- If |L| = k, then there are 2^k – 2 candidate association rules (ignoring L $\rightarrow \varnothing$ and $\varnothing \rightarrow$ L)

Rule Generation

- How to efficiently generate rules from frequent itemsets?
  - In general, confidence does not have an anti-monotone property

    c(ABC $\rightarrow$D) can be larger or smaller than c(AB $\rightarrow$D)

  - But confidence of rules generated from the same itemset has an anti-monotone property
  - e.g., L = {A,B,C,D}:

$$c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$$

c(ABC->D) = s(ABCD) / s(ABC) >= c(AB->CD) = s(ABCD) / s(AB)

- Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

Rule Generation

Lattice of rules