Subject : Data Warehouse and Data Mining

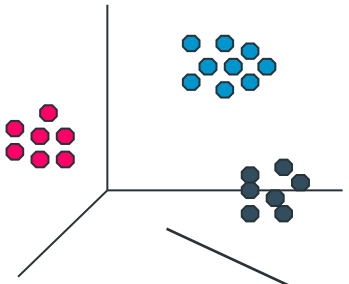
Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

(Computer Engineering and Technology)
(SYB.Tech-AI-DS)

Unit IV: Association Rule Mining

Market basket Analysis, Frequent item set, Closed item set, Association Rules, a-priori Algorithm, Generating Association Rules from Frequent Item sets, Improving the Efficiency of a-priori, Mining Frequent Item sets without Candidate Generation: FP Growth Algorithm; Mining ,Generating Rules.

Data Mining Task..

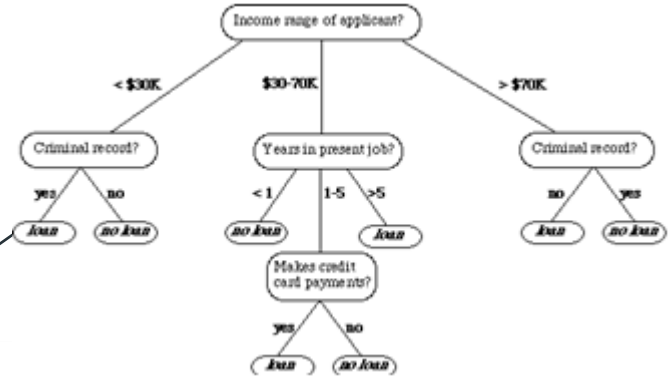


Clustering

Data

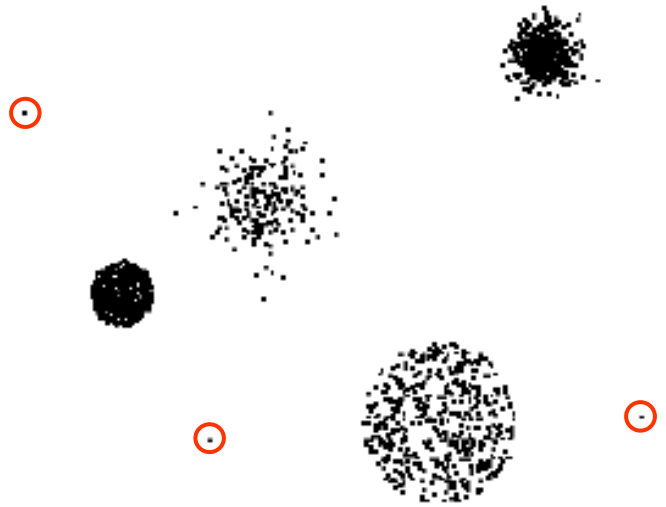
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes
11	No	Married	60K	No
12	Yes	Divorced	220K	No
13	No	Single	85K	Yes
14	No	Married	75K	No
15	No	Single	90K	Yes

Predictive Modeling



Association Rules

Anomaly Detection



Market Basket Analysis

Consider shopping cart filled with several items

Market basket analysis tries to answer the following questions:

- –Who makes purchases?
- –What do customers buy together?
- –In what order do customers purchase items?

•**Def:** Market Basket Analysis (Association Analysis) is a mathematical modeling technique based upon the theory that if you buy a certain group of items, you are likely to buy another group of items.

•It is **used to analyze the customer purchasing behavior** and **helps in increasing the sales and maintain inventory by focusing on the point of sale transaction data.**

•Given a dataset, **the Apriori Algorithm trains and identifies product baskets and product association rules**

Why Association Rule Analysis is Important

- Discloses an intrinsic and important property of data sets
- Forms the foundation for many essential data mining tasks
  - Association, correlation, and causality analysis
  - Sequential, structural (e.g., sub-graph) patterns
  - Pattern analysis in spatiotemporal, multimedia, time-series, and stream data
  - Classification: associative classification
  - Cluster analysis: frequent pattern-based clustering
  - Data warehousing: iceberg cube and cube-gradient
  - Semantic data compression: fascicles
  - Broad applications

Introduction : Frequent Patterns

- Frequent pattern: a pattern (a set of items, subsequences, substructures, etc.) that occurs frequently in a data set
- First proposed by Agrawal, Imielinski, and Swami [AIS93] in the context of **frequent item sets and association rule mining**
- Motivation: Finding inherent regularities in data

  What products were often **purchased together**?— Pen and Pencil?!

  What are the **subsequent purchases after buying a PC**?

  What kinds of **DNA are sensitive to this new drug**?
- **Applications**

  Basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis

Introduction : Association Rule Mining

- Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction

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} → {Beer},
{Milk, Bread} → {Eggs,Coke},
{Beer, Bread} → {Milk},

Implication means co-occurrence, not causality!

Definitions and Terminologies

- Transaction is a set of items (Itemset).
- Confidence : It is the measure of uncertainty or trust worthiness associated with each discovered pattern.
- Support : It is the measure of how often the collection of items in an association occur together as percentage of all transactions
- Frequent itemset : If an itemset satisfies minimum support, then it is a frequent itemset.
- Strong Association rules: Rules that satisfy both a minimum support threshold and a minimum confidence threshold
- In Association rule mining, **we first find all frequent itemsets and then generate strong association rules from the frequent itemsets**

Definition with example : Frequent Itemset

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
- **Support count (σ)**
  - Frequency of occurrence of an itemset
  - E.g. σ({Milk, Bread,Diaper}) = 2
- **Support**
  - Fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Diaper}) = 2/5
- **Frequent Itemset**
  - An itemset whose support is greater than or equal to a *minsup* threshold

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

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

- Association Rule
  - An implication expression of the form X → Y, where X and Y are itemsets
  - Example:
    {Milk, Diaper} → {Beer}

- Rule Evaluation Metrics
  - Support (s)
    - Fraction of transactions that contain both X and Y
  - Confidence (c)
    - Measures how often items in Y appear in transactions that contain X

$$support = \frac{(X \cup Y).count}{n}$$

$$confidence = \frac{(X \cup Y).count}{X.count}$$

Example:

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

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

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

Association Rule Mining Task

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

- Brute-force approach:
  List all possible association rules
  Compute the support and confidence for each rule
  Prune rules that fail the *minsup* and *minconf* thresholds

Revisited Terms :

Support: The rule holds with support *sup* in *T* (the transaction data set) if sup% of transactions contain $X \cup Y$.

–$sup = \Pr(X \cup Y)$.

•Confidence: The rule holds in *T* with confidence *conf* if *conf* % of tranactions that contain *X* also contain *Y.*

–$conf = \Pr(Y \mid X)$

•An association rule is a pattern that states when *X* occurs, *Y* occurs with certain probability.

Mining Association Rules

Two-step approach:

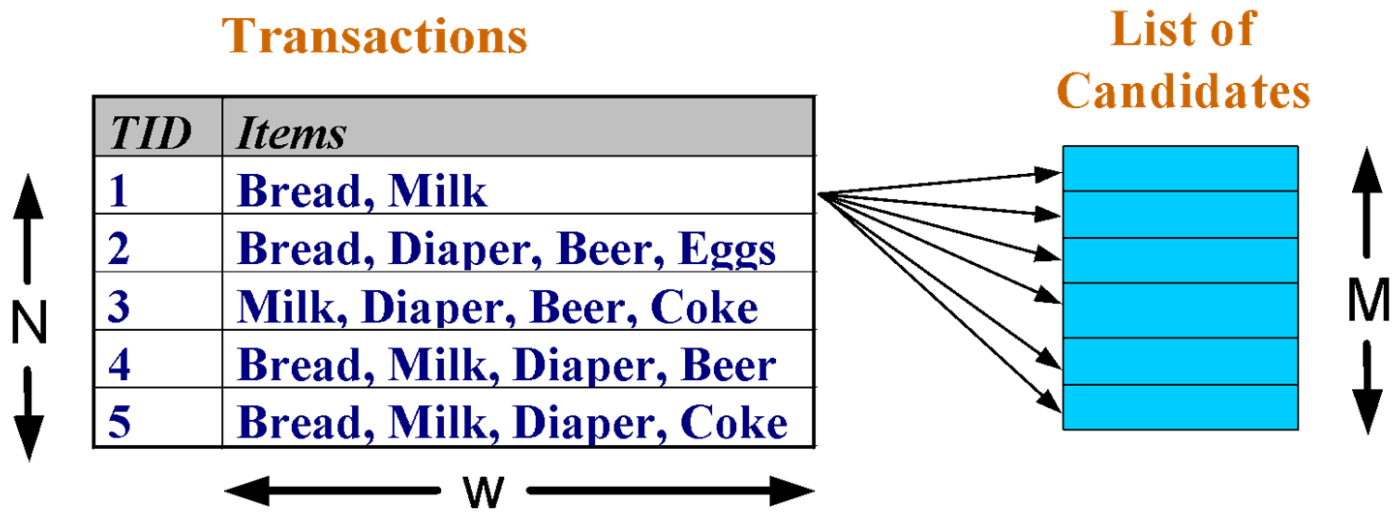
1. Frequent Itemset Generation
   - Generate all itemsets whose support ≥ minsup

2. Rule Generation
   - Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset

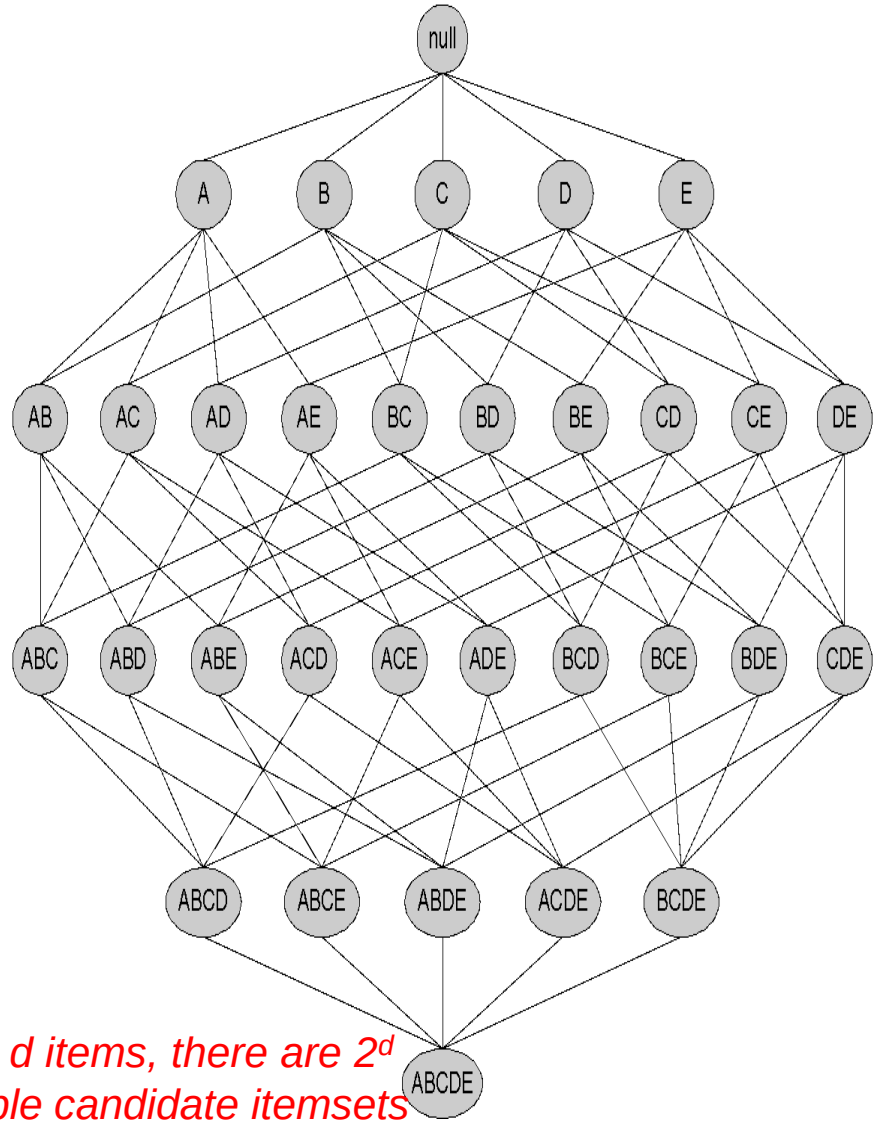
Frequent itemset generation is computationally expensive

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
  - Expensive !!!



Given d items, there are 2^d possible candidate itemsets

Frequent Itemset Generation Strategies

To Reduce the number of candidates (M) Complete search: $M=2^d$ use pruning techniques

Reducing Number of Candidates

- Apriori principle:
  - If an itemset is frequent, then all of its subsets must also be frequent

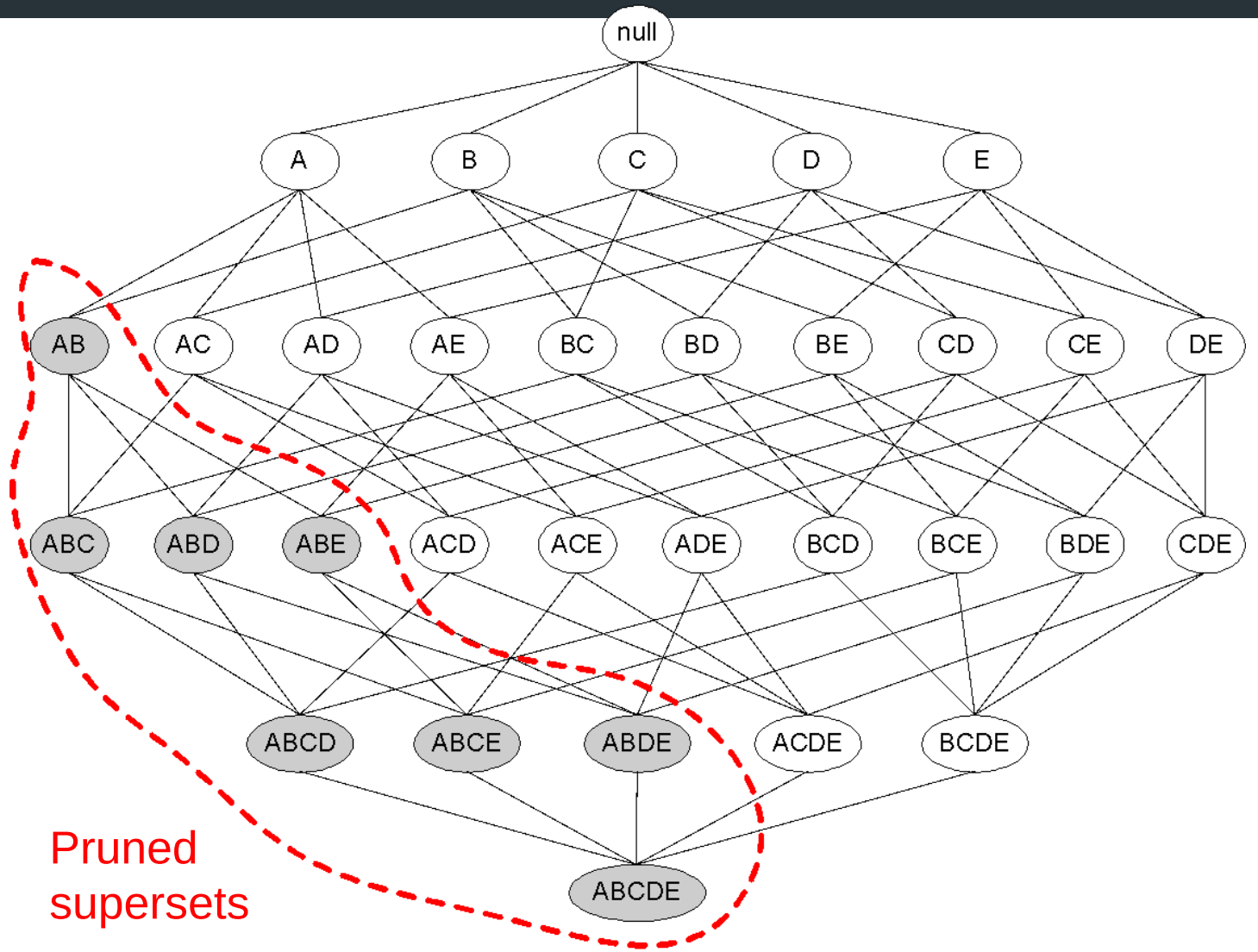
- Apriori principle holds due to the following property of the support measure:

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

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

Note : A constraint C is anti-monotone if the super pattern satisfies C, all of its sub-patterns do so too

Illustrating Apriori Principle



Apriori Algorithm

- F_k: frequent k-itemsets
- L_k: candidate k-itemsets

- Algorithm
  - Let k=1
  - Generate F_1 = {frequent 1-itemsets}
  - Repeat until F_k is empty
    - **Candidate Generation(Join Step)**: Generate L_{k+1} from F_k
    - **Support Counting**: Count the support of each candidate in L_{k+1} by scanning the DB
    - **Candidate Pruning(Prune Step): Prune candidate itemsets in Lk+1 containing subsets of length k that are infrequent .** Eliminate candidates in L_{k+1} that are infrequent, leaving only those that are frequent => F_{k+1}

The Apriori Algorithm (Pseudo-Code)

C_k: Candidate itemset of size k
L_k : frequent itemset of size k

L_1 = {frequent items};
for (k = 1; L_k !=∅; k++) **do begin**
    C_{k+1} = candidates generated from L_k;
    for each transaction t in database do
        increment the count of all candidates in C_{k+1} that are
        contained in t *%calculate Support)*
    L_{k+1} = candidates in C_{k+1} with min_support
    end
return $\cup_k L_k$;

Implementation of Apriori

- **How to generate candidates?**
  - **Step 1: self-joining L_k**
  - **Step 2: pruning**
- **Example of Candidate-generation**
  - **L_3={*abc, abd, acd, ace, bcd*}**
  - **Self-joining: L_3*L_3**
    - ***abcd* from *abc* and *abd***
    - ***acde* from *acd* and *ace***
  - **Pruning:**
    - ***acde* is removed because *ade* is not in L_3**
  - **C_4 = {*abcd*}**

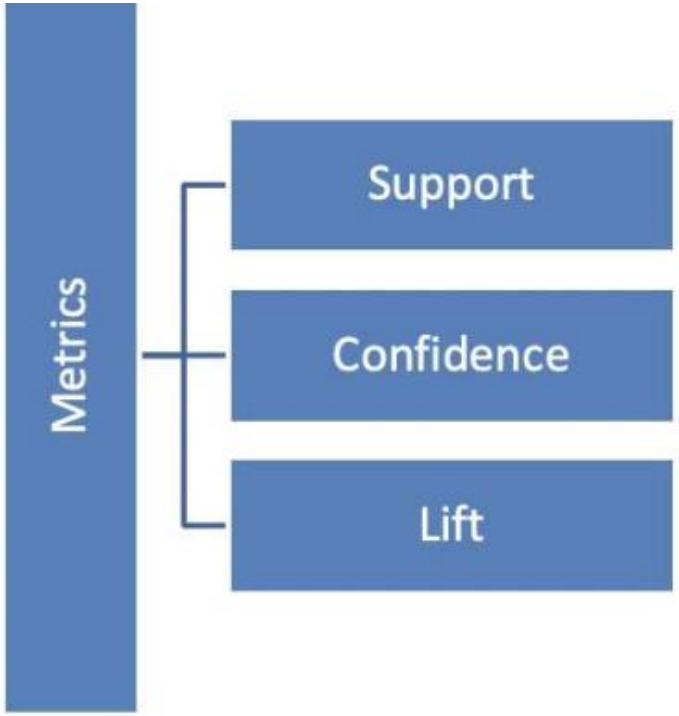
Frequent itemset Generation Method

Rule : Merge two frequent itemsets with size n if their first n-1 items are identical

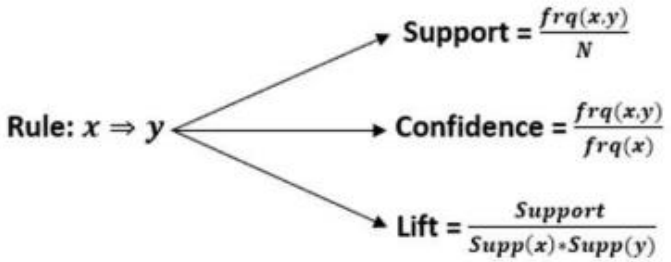
- F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
  - Merge(**AB**C, **AB**D) = **AB**CD
  - Merge(**AB**C, **AB**E) = **AB**CE
  - Merge(**AB**D, **AB**E) = **AB**DE

  - Do not merge(**A**BD,**A**CD) because they share only prefix of length 1 instead of length 2

Association Rule Measures with significance



-



Support:

This measure gives an idea of how frequent an itemset is in all the transactions.

Confidence:

It indicates how often a rule is found to be true. This measure defines the likeliness of occurrence of consequent on the cart given that the cart already has the antecedents. It is measured as the probability that transactions that contain the antecedent of the rule will also contain the consequent.

The confidence for an association rule having a very frequent consequent will always be high.

Lift: List summarises strength of association between antecedents and consequents. Larger the lift, higher the association .

Lift also detect misleading association rule. Generally lift to be used when support of Z>confidence of rule , For x,y→z

Lift more than 1 suggest that, presence of x,y increases probability of z may occur in transaction

Illustrating Apriori Principle : Candidate Generation for n=1

Minimum Support = 3

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

Items (1-itemsets)

Items
Bread
Coak
Milk
Beer
Diaper
Egg

Illustrating Apriori Principle : Support Counting

Minimum Support = 3

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

Items (1-itemsets)

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

Illustrating Apriori Principle : Prune infrequent Candidate

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

Minimum Support = 3

Items (1-itemsets)

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

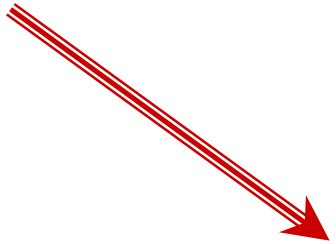
Candidate Generation(n=2)

Minimum Support = 3

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

Items (1-itemsets)

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



Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Itemset
{Bread,Milk}
{Bread, Beer }
{Bread,Diaper}
{Beer, Milk}
{Diaper, Milk}
{Beer,Diaper}

HINT : Merge two frequent itemsets with size n if their first n-1 items are identical

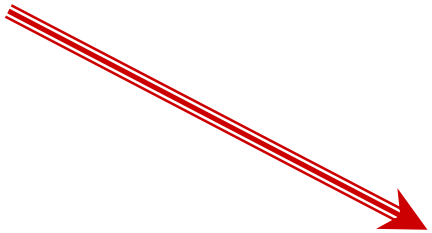
Support Counting

Minimum Support = 3

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

Items (1-itemsets)

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



Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

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

Prune infrequent Candidate and Candidate Generation for n=3

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

Items (1-itemsets)

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

Minimum Support = 3

Pairs (2-itemsets)

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

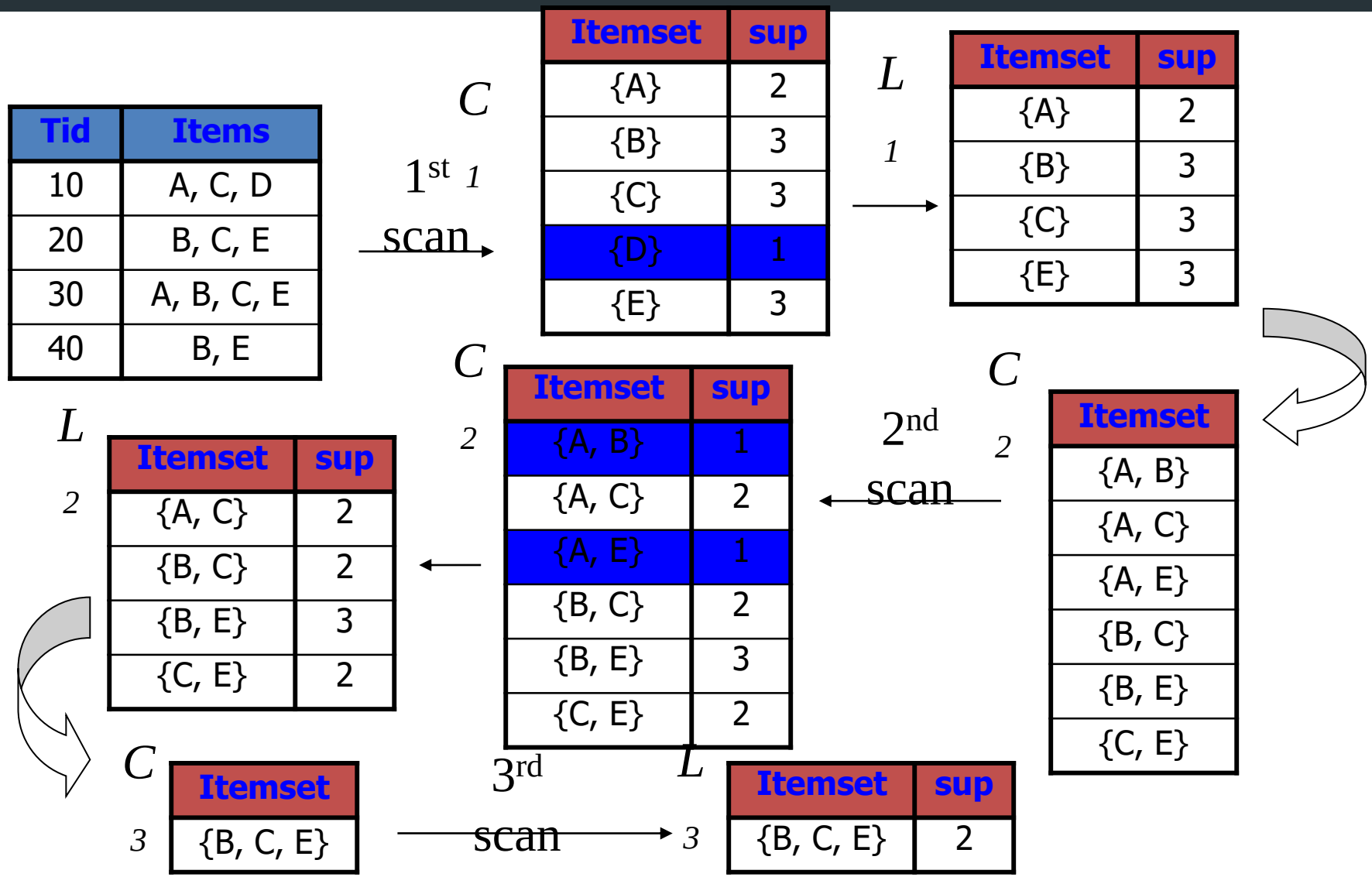
Triplets (3-itemsets)

Itemset	Count
{Bread, Diaper, Milk}	2

Note: Merge two frequent itemsets with size n if their first n-1 items are identical

Another Example

Sup_{min} = 2



Association Rule Generation

- Given a frequent itemset L, find all non-empty subsets f ⊂ L such that f → L – f satisfies the minimum confidence requirement

  - **If {A,B,C,D} is a frequent itemset, then candidate rules:**

    ABC →D,
    A →BCD,
    AB →CD,
    BD →AC,

    ABD →C,
    B →ACD,
    AC → BD,
    CD →AB

    ACD →B,
    C →ABD,
    AD → BC,

    BCD →A,
    D →ABC
    BC →AD,

- If |L| = k, then there are $2^k - 2$ candidate association rules (ignoring L → ∅ and ∅ → L)

What are the Association Rules for Below, calculate Confidence of Each Rule

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

Itemset	sup
{B, C, E}	2

- BC--> E
- B-->CE
- CE-->B
- E-->BC
- BE-->C
- C->BE

Example:Confidence calculation of association rule

- Suppose {2,3,4} is frequent, with sup=50%
  - Proper nonempty subsets: {2,3}, {2,4}, {3,4}, {2}, {3}, {4}, with sup=50%, 50%, 75%, 75%, 75%, 75% respectively
  - These generate these association rules:
    - 2,3 → 4, confidence=100%
    - 2,4 → 3, confidence=100%
    - 3,4 → 2, confidence=67%
    - 2 → 3,4, confidence=67%
    - 3 → 2,4, confidence=67%
    - 4 → 2,3, confidence=67%
    - All rules have support = 50%

Confidence of a rule =
the support for the combination / the support for the condition.

Performance Bottleneck of Apriori

Bottlenecks of *Apriori*: candidate generation

–Generate 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.

–Candidate Test incur multiple scans of database: each candidate

FP Growth Introduction

Introduction

Allows frequent itemset discovery without candidate itemset generation.

Two step process:

–Step-I: Build a compact data structure called FP-Tree(built using 2-passes over the data set)

–Step-II: extract frequent itemset directly from the FP-tree

Outline

Frequent Pattern Mining: Problem statement and an example

Overview

–FP-tree:•structure, construction and advantages

–FP-growth:

FP-tree -->conditional pattern bases -->conditional FP-tree -->frequent patterns

FP Growth

FP Growth Allows frequent itemset discovery without candidate generation.

- Two step Process

1.Build a compact data structure called the FP-tree : 2 passes over the database [Only !!!]
2.extracts frequent itemset directly from the FP-tree: Traverse through FP-tree

- Use a compressed representation of the database using an FP-tree
  Once an FP-tree has been constructed, it uses a recursive divide-and-conquer approach to mine the frequent itemsets.

- Building the FP-Tree

1. Scan data to determine the support count of each item.
   Infrequent items are discarded, while the frequent items are sorted in decreasing support counts.
2. Make a second pass over the data to construct the FPtree.
   As the transactions are read, before being processed, their items are sorted according to the above order.

FP Tree Definition

FP-tree is **a frequent pattern tree** . Formally, FP-tree is a tree structure defined below:

1. One root labeled as “null", a set of *item prefix sub-trees* as the children of the root, and a *frequent-item header table*.
2. Each node in *the item prefix sub-trees* has three fields:
   - –item-name : register which item this node represents,
   - –count, the number of transactions represented by the portion of the path reaching this node,
   - –node-link that links to the next node in the FP-tree carrying the same item-name, or null if there is none.
3. Each entry in the *frequent-item header table* has two fields,
   - –item-name, and
   - –head of node-link that points to the first node in the FP-tree carrying the item-name.

Construct FP-tree from a Transaction Database

TID	Items bought	(ordered) frequent items
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}
200	{a, b, c, f, l, m, o}	{f, c, a, b, m}
300	{b, f, h, j, o, w}	{f, b}
400	{b, c, k, s, p}	{c, b, p}
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}

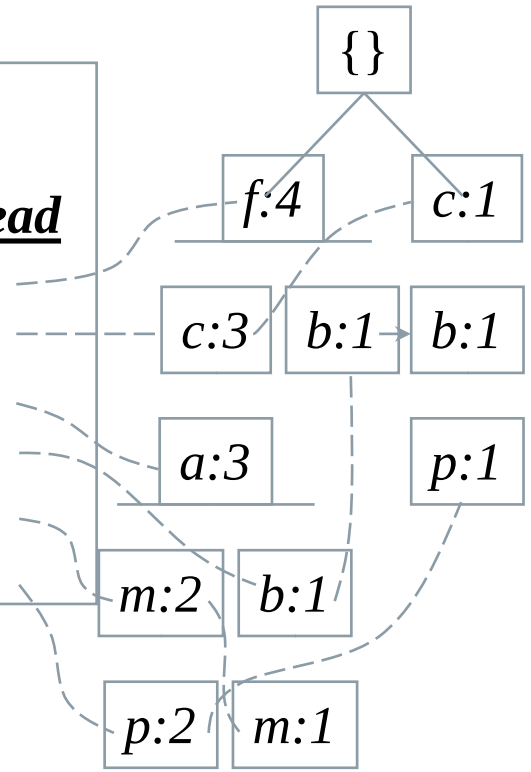
1. Scan DB once, find frequent 1-itemset (single item pattern)
2. Sort frequent items in frequency descending order, f-list (Why??)
3. Scan DB again, construct FP-tree

F-list=f-c-a-b-m-p

min_support = 3

Header Table

Item	frequency	head
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



Mining Frequent Patterns Using FP-tree

General idea (divide-and-conquer)

Recursively grow frequent patterns using the FP-tree:looking for shorter ones recursively and then concatenating the suffix:

–For each frequent item, construct its conditional pattern base, and then its conditional FP-tree;

–Repeat the process on each newly created conditional FP-tree until the resulting FP-tree is empty, or it contains only one path (single path will generate all the combinations of its sub-paths, each of which is a frequent pattern)

FP Growth : Mining Frequent Pattern:3 Major Steps

Starting the processing from the end of list L:

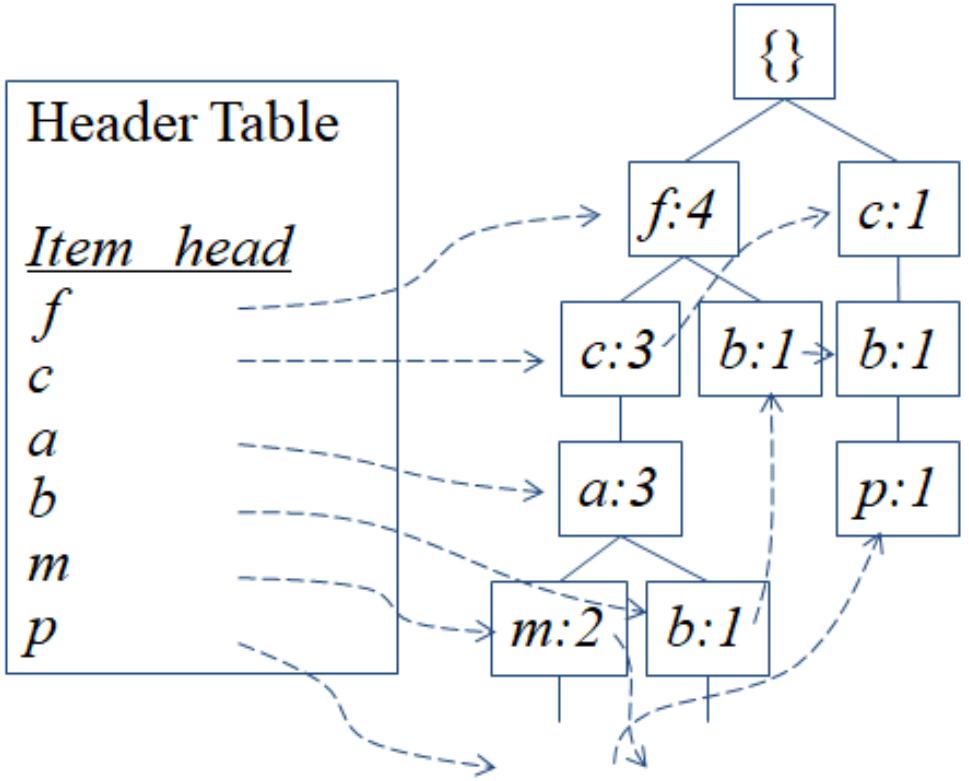
Step 1: Construct conditional pattern base for each item in the header table

Step 2: Construct conditional FP-tree from each conditional pattern base

Step 3: Recursively mine conditional FP-trees and grow frequent patterns obtained so far. If the conditional FP-tree contains a single path, simply enumerate all the patterns

Step 1:Construct Conditional Base

- Starting at the bottom of frequent-item header table in the FP-tree
- Traverse the FP-tree by following the link of each frequent item
- Accumulate all of **transformed prefix paths** of that item to form a **conditional pattern base**

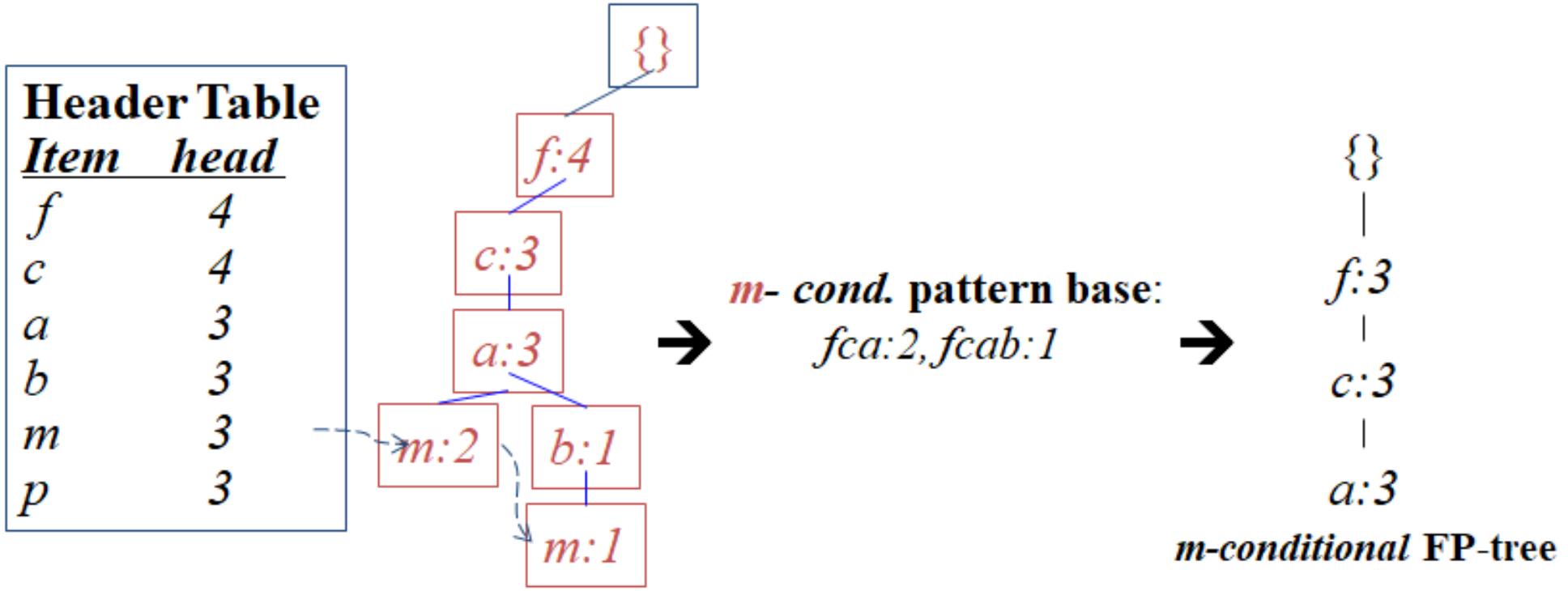


***Conditional* pattern bases**

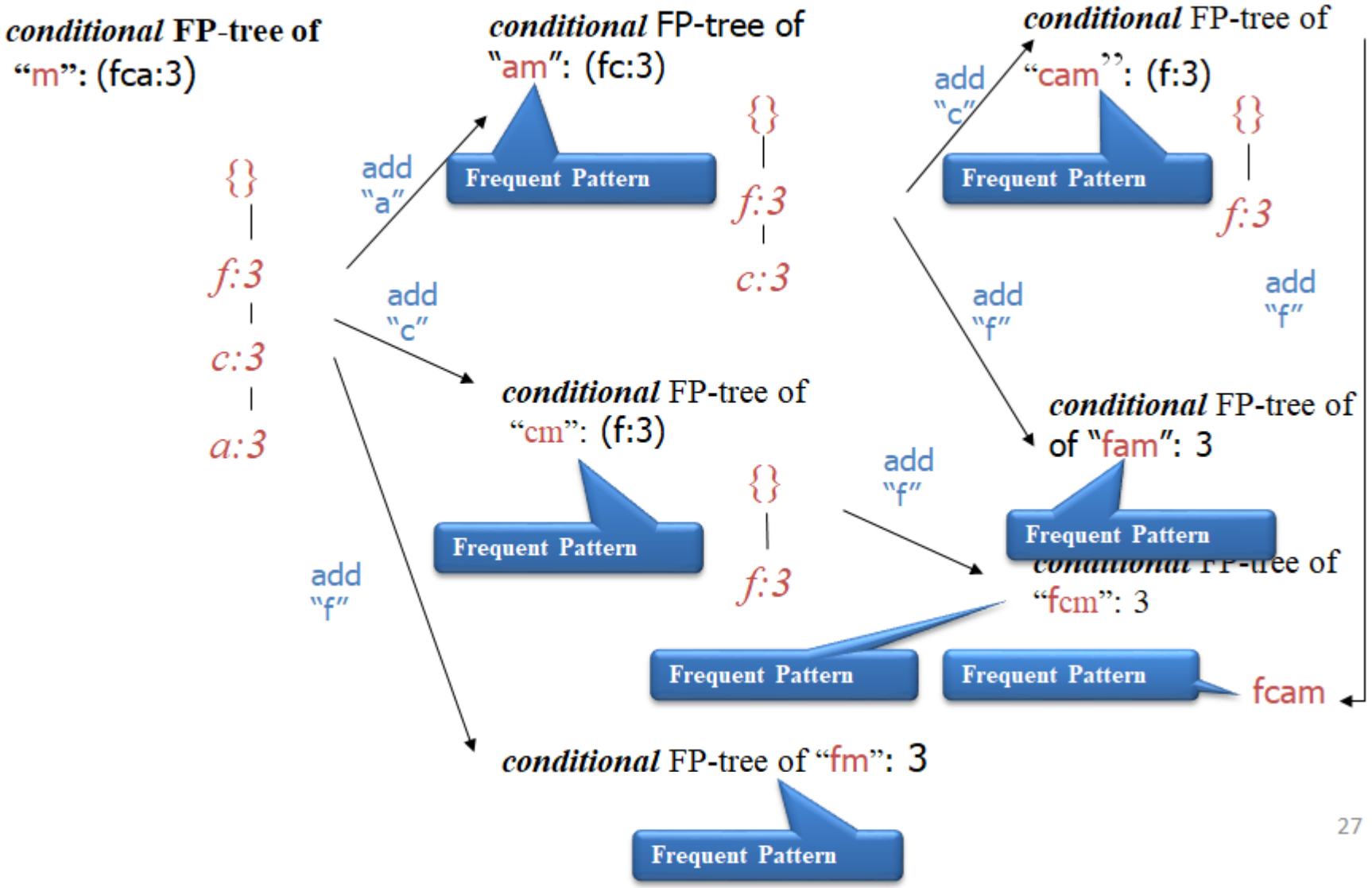
item	*cond. pattern base*
p	*fcam:2, cb:1*
m	*fca:2,fcab:1*
b	*fca:1,f:1, c:1*
a	*fc:3*
c	*f:3*
f	*{}*

Step 2: Construct Conditional FP Tree

- For each pattern base
  - Accumulate the count for each item in the base
  - Construct the conditional FP-tree for the frequent items of the pattern base



Step 3: Recursively mine the conditional FP-tree



27

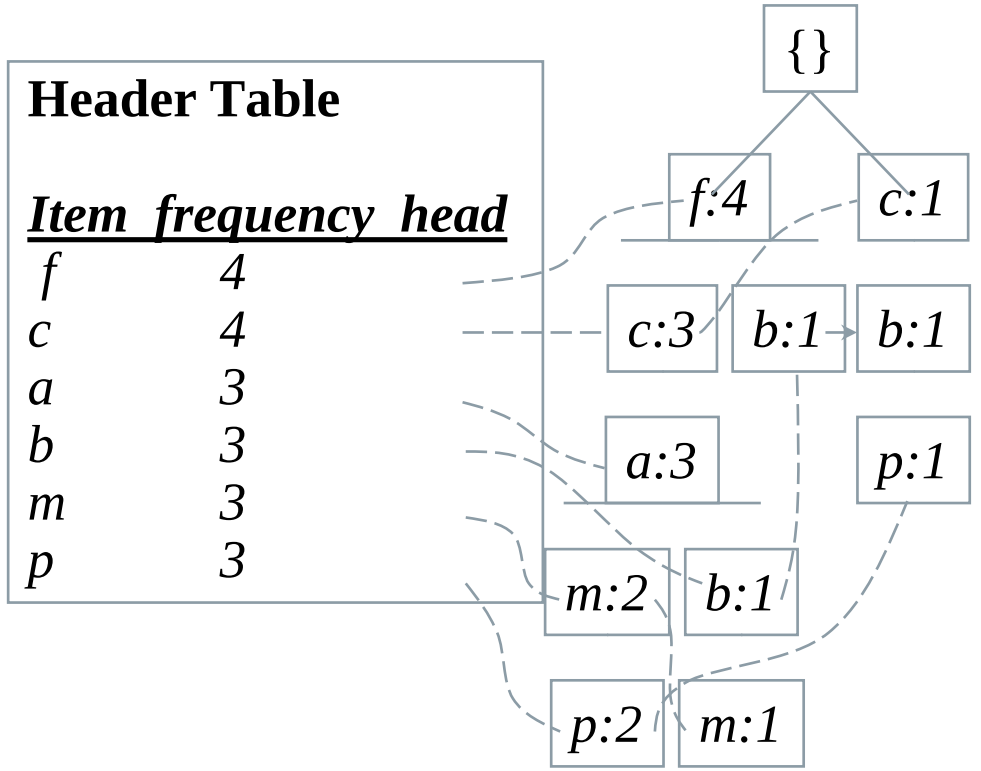
Principles of FP-Growth

- Pattern growth property
  - Let a be a frequent itemset in DB, B be a's conditional pattern base, and b be an itemset in B. Then a U b is a frequent itemset in DB iff b is frequent in B.
- Is "*fcabm* " a frequent pattern?
  - "fcab" is a branch of m's conditional pattern base
  - "b" is **NOT** frequent in transactions containing "fcab "
  - "bm" is **NOT** a frequent itemset.

Conditional Pattern Bases and Conditional FP-Tree

min_support = 3

Item	Conditional pattern base	Conditional FP-tree
p	{(fcam:2), (cb:1)}	{(c:3)}|p
m	{(fca:2), (fcab:1)}	{(f:3, c:3, a:3)}|m
b	{(fca:1), (f:1), (c:1)}	Empty
a	{(fc:3)}	{(f:3, c:3)}|a
c	{(f:3)}	{(f:3)}|c
f	Empty	Empty



Why Descending Order of Support?

This tree will become larger than FP-tree(in which descending order is considered) , because in FP-tree, more frequent items have a higher position, which makes branches less

FP Example

Transactions

A B C E F O

A C G

E I

A C D E G

A C E G L

E J

A B C E F P

A C D

A C E G M

A C E G N

Freq. 1-Itemsets.
Supp. Count ≥2

A:8	
C:8	
E:8	
G:5	
B:2	
D:2	
F:2	

Transactions with items sorted based on frequencies, and ignoring the infrequent items.

A C E B F

A C G

E

A C E G D

A C E G

E

A C E B F

A C D

A C E G

A C E G

FP-Tree after reading 1st transaction

A C E B F

A C G

E

A C E G D

A C E G

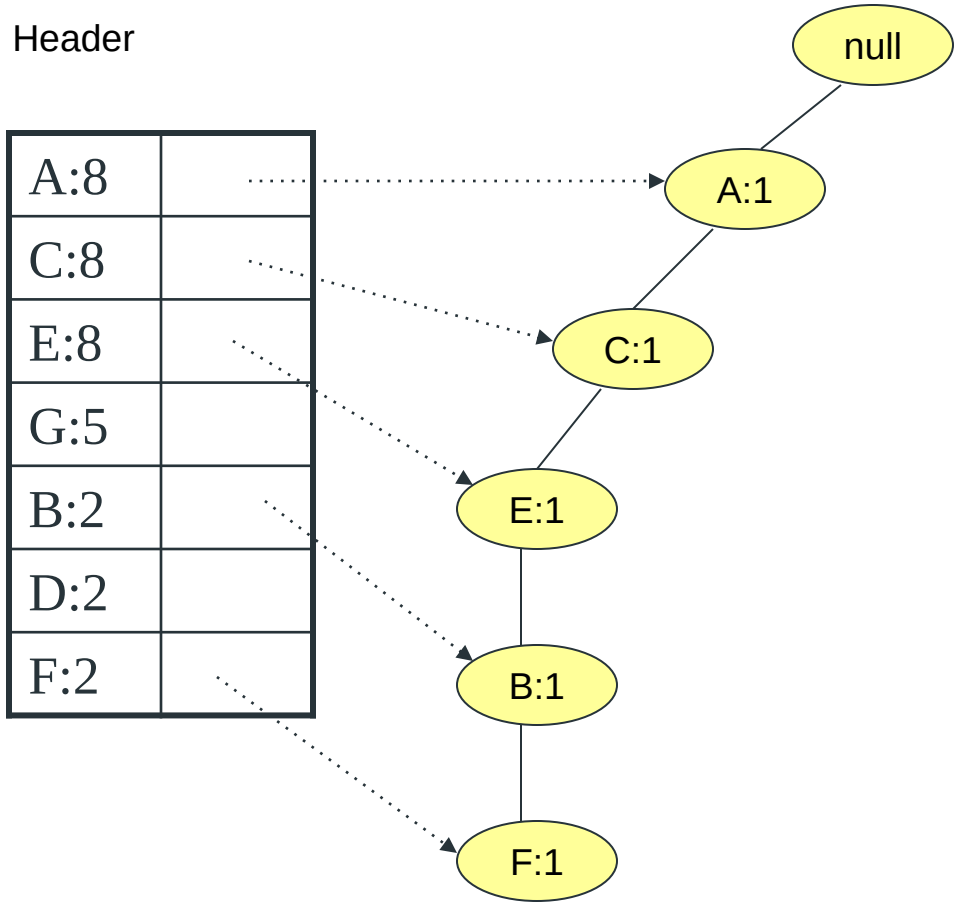
E

A C E B F

A C D

A C E G

A C E G



FP-Tree after reading 2nd transaction

A C E B F

A C G

E

A C E G D

A C E G

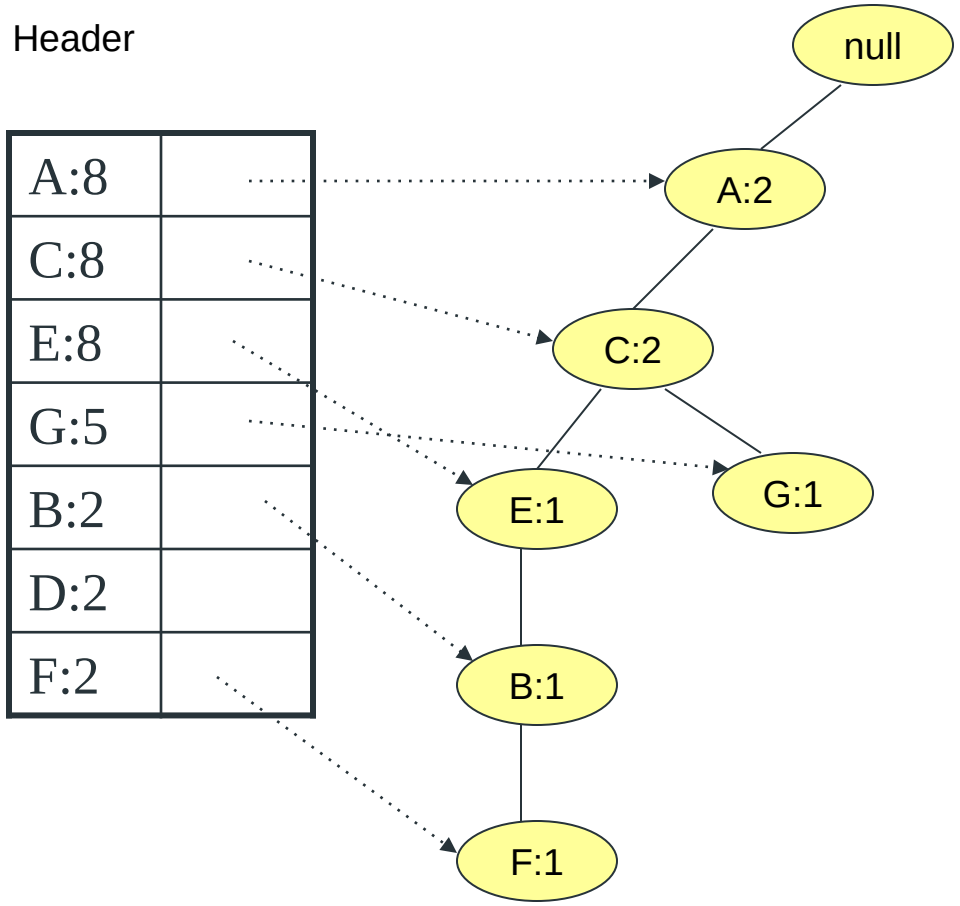
E

A C E B F

A C D

A C E G

A C E G



FP-Tree after reading 3rd transaction

A C E B F

A C G

E

A C E G D

A C E G

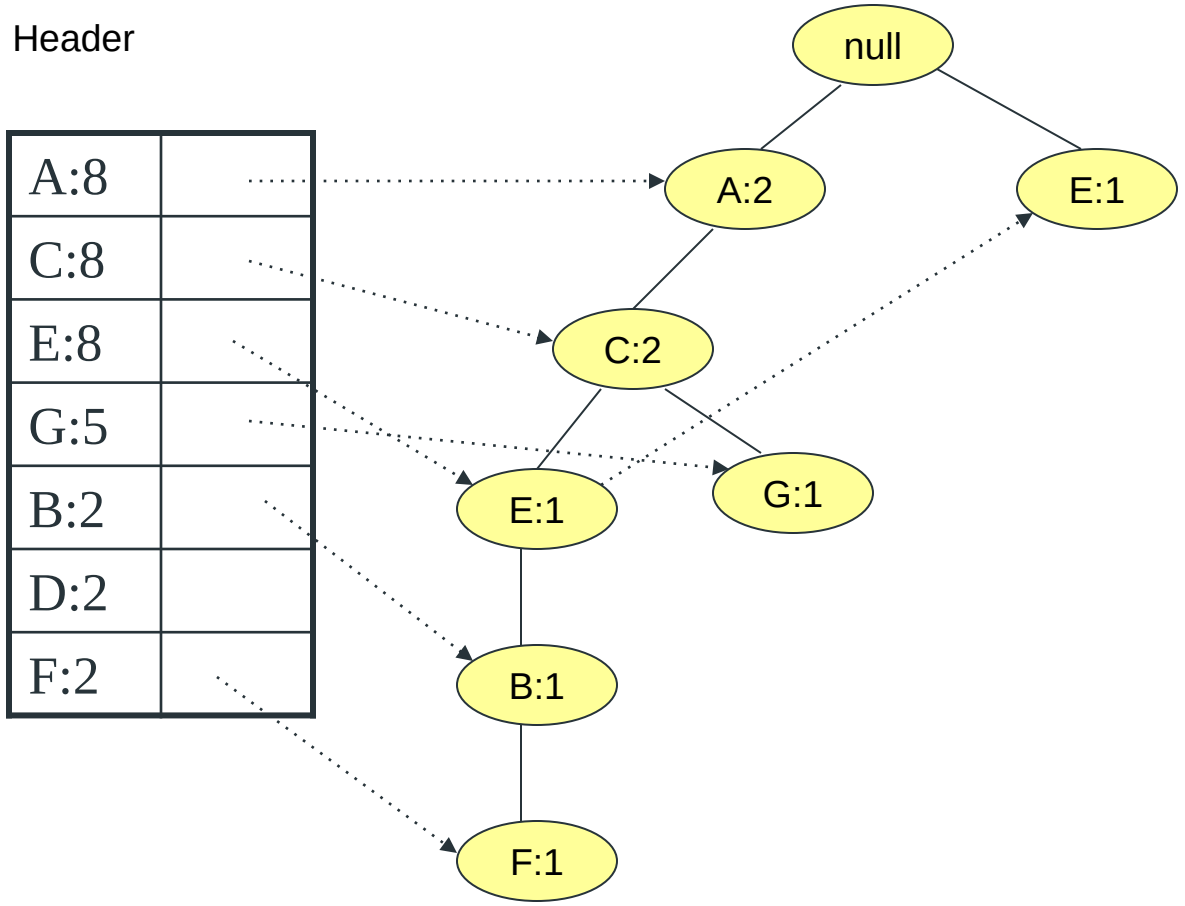
E

A C E B F

A C D

A C E G

A C E G



FP-Tree after reading 4th transaction

A C E B F

A C G

E

A C E G D

A C E G

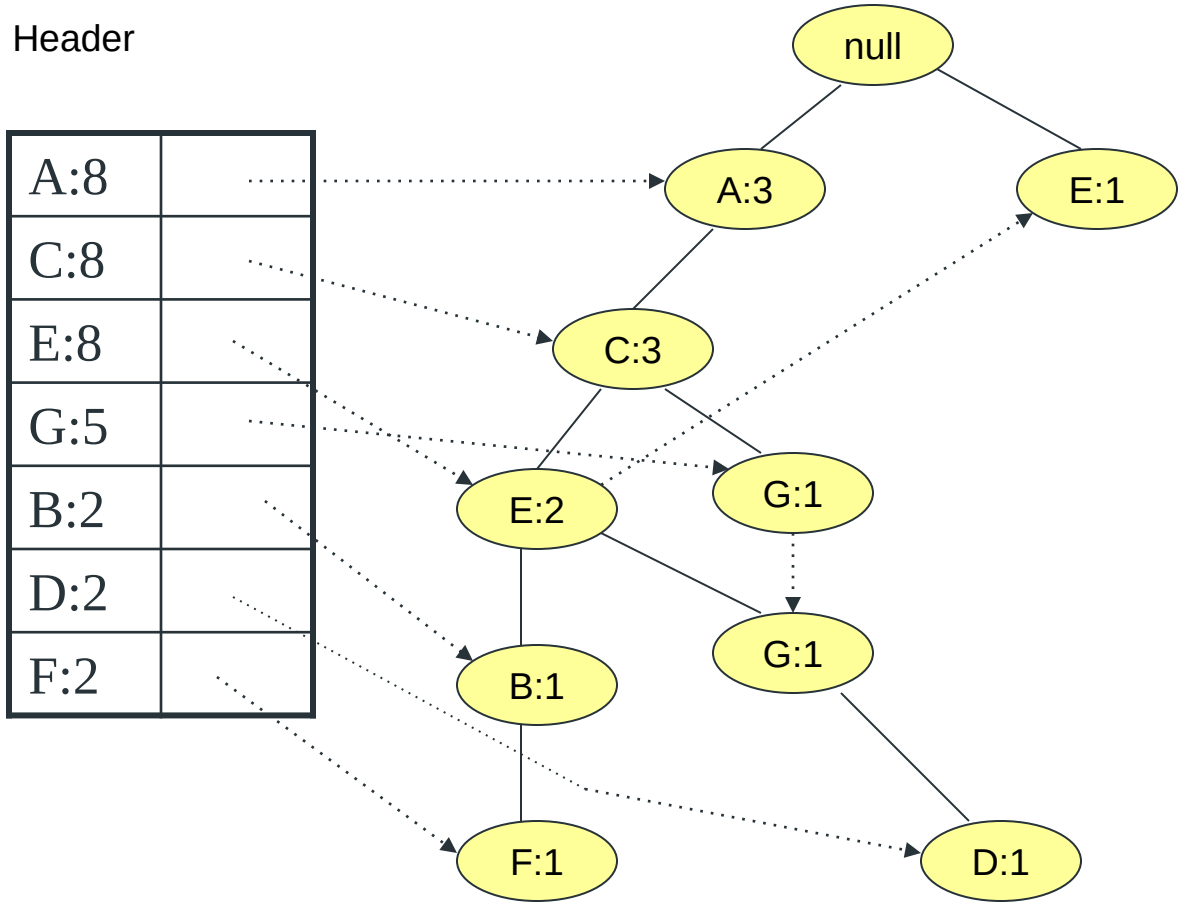
E

A C E B F

A C D

A C E G

A C E G



FP-Tree after reading 5th transaction

A C E B F

A C G

E

A C E G D

A C E G

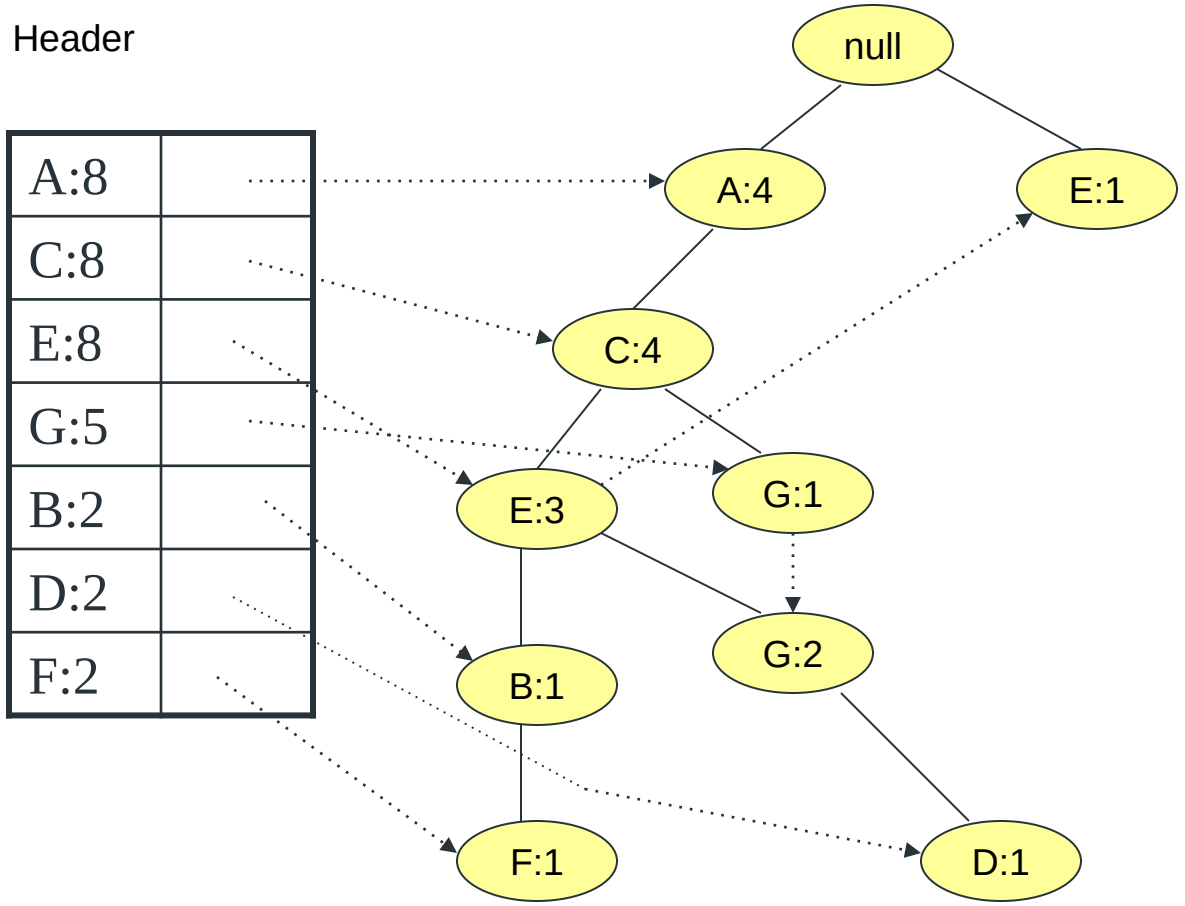
E

A C E B F

A C D

A C E G

A C E G



FP-Tree after reading 6th transaction

A C E B F

A C G

E

A C E G D

A C E G

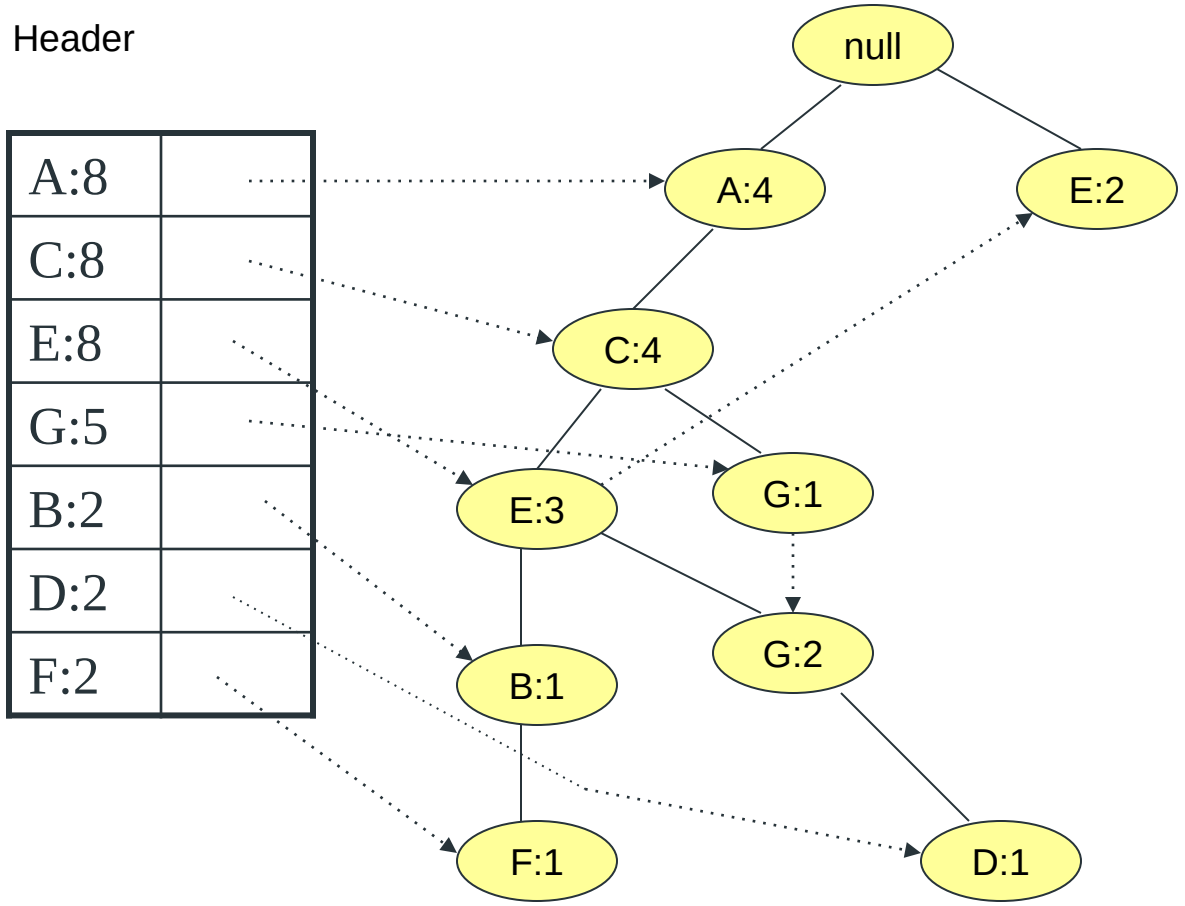
E

A C E B F

A C D

A C E G

A C E G



FP-Tree after reading 7th transaction

A C E B F

A C G

E

A C E G D

A C E G

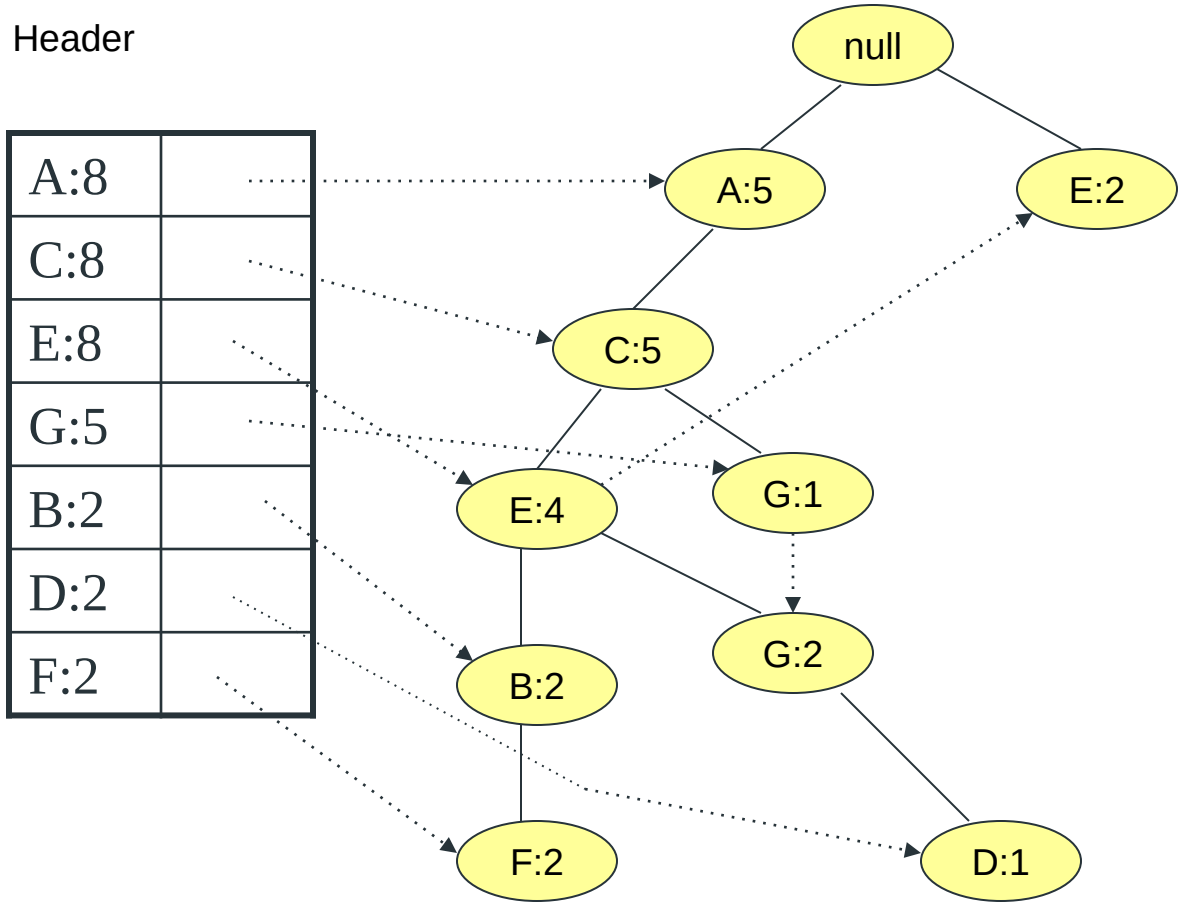
E

A C E B F

A C D

A C E G

A C E G



FP-Tree after reading 8th transaction

A C E B F

A C G

E

A C E G D

A C E G

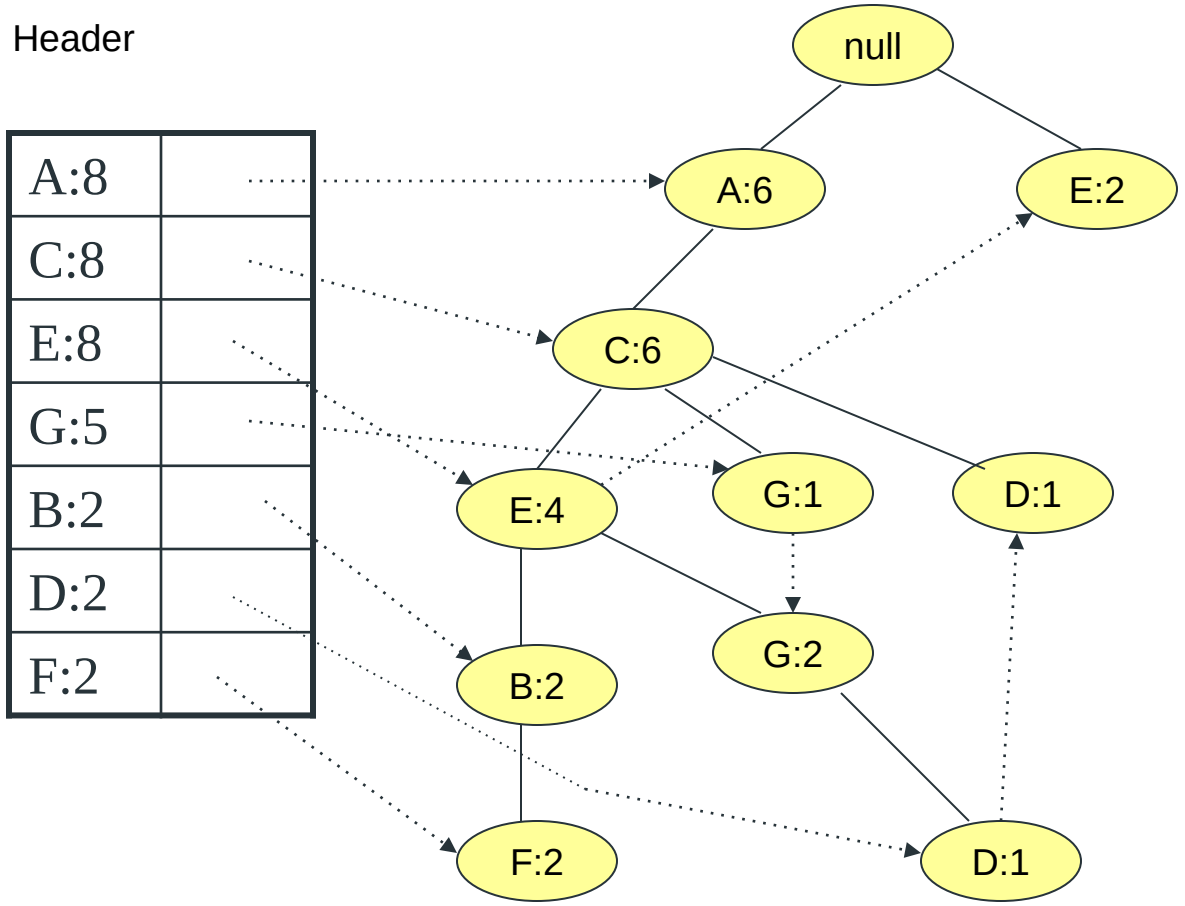
E

A C E B F

A C D

A C E G

A C E G



FP-Tree after reading 9th transaction

A C E B F

A C G

E

A C E G D

A C E G

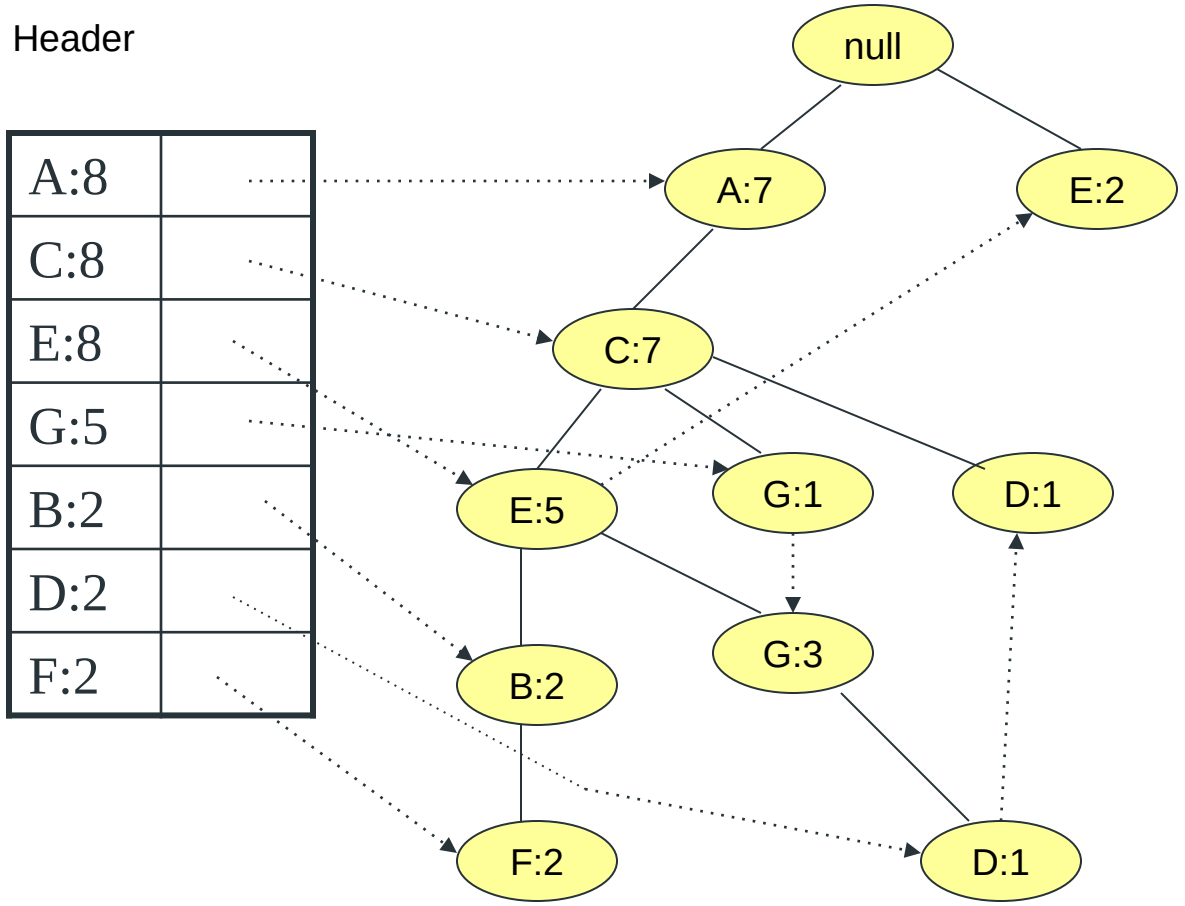
E

A C E B F

A C D

A C E G

A C E G



FP-Tree after reading 10th transaction

A C E B F

A C G

E

A C E G D

A C E G

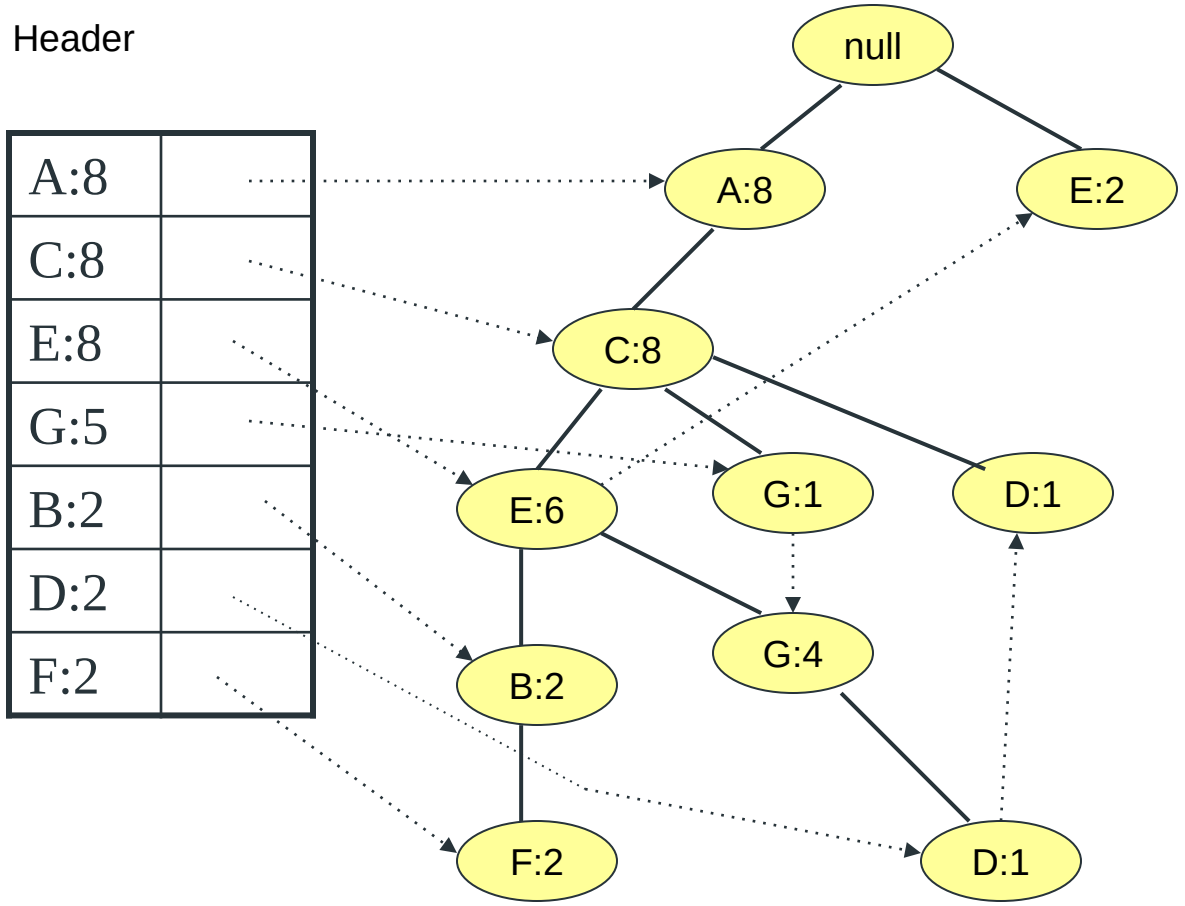
E

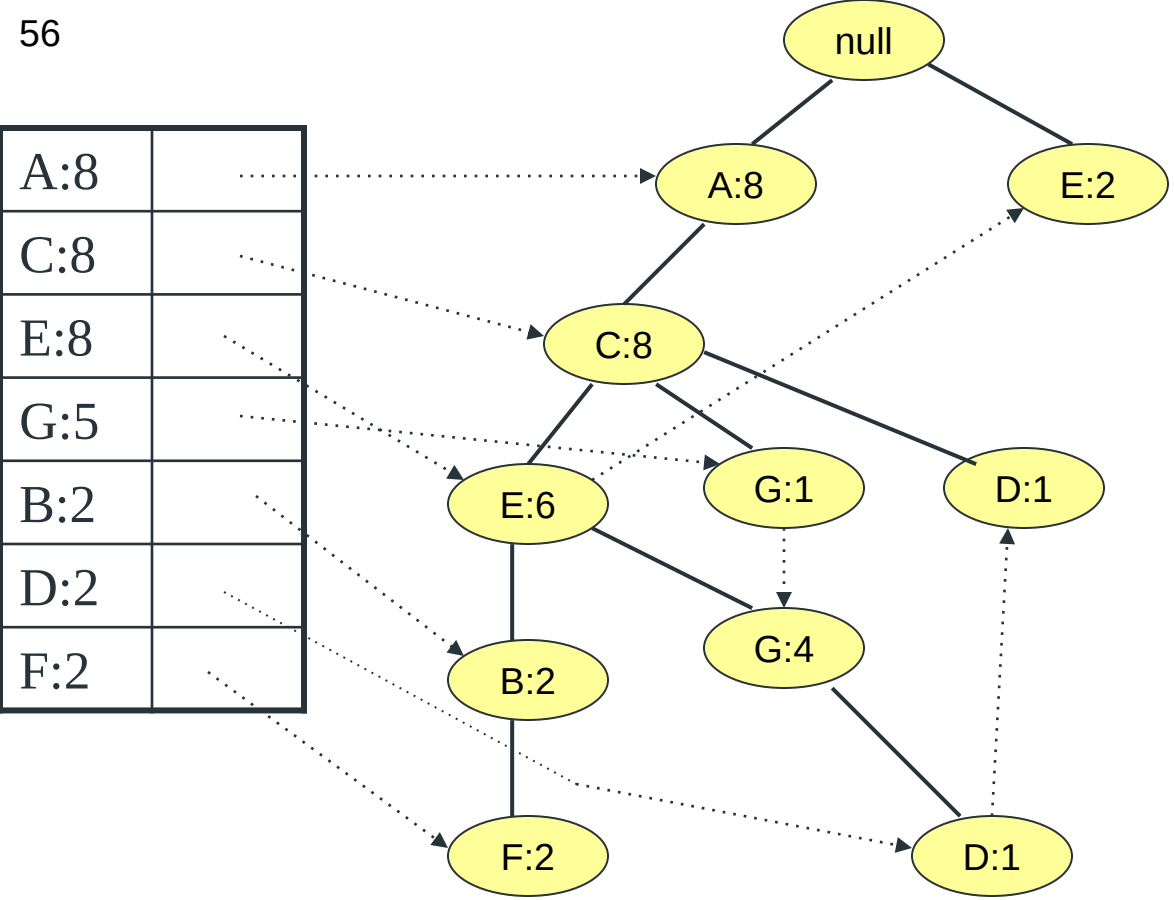
A C E B F

A C D

A C E G

A C E G





Items Arranged in ascending order of support	Conditional Pattern Base	Conditional Frequent Pattern Tree	Frequent Pattern Generated
F	{A,C,E,B :2}	{A:2,C:2,E:2,B:2}	{A,F:2} , {C,F:2} ,{E,F :2} ,{B,F:2} {A,C,E,B,F:2}
D	{A,C:1} {A,C,E,G:1}	{A:2 ,C:2}	{A,D:2} , {C,D:2} {A,C,D:2}
B	{A, C,E:2}	{A:2,C:2,E:2}	{A,B:2} {C,B:2} {E,B:2} {A,C,E,B:2}
G	{A,C:1} {A,C,E:4}	{A ,C:5} {E:4}	{A,G:5} {C,G:5} {A.C.G:5} {E,G:4}
E	{A,C:6}	{A,C:6}	{A,E:6} {C,E:6} {A,C.E:6}
C	{A:8}	{A:8}	{A,C:8}

Create FP Tree for Following , Identify Frequent Items

min_sup:3

Transaction ID	Items
T1	{E,K,M,N,O,Y}
T2	{D,E,K,N,O,Y}
T3	{A,E,K,M}
T4	{C,K,M,U,Y}
T5	{C,E,I,K,O,O}

Answer

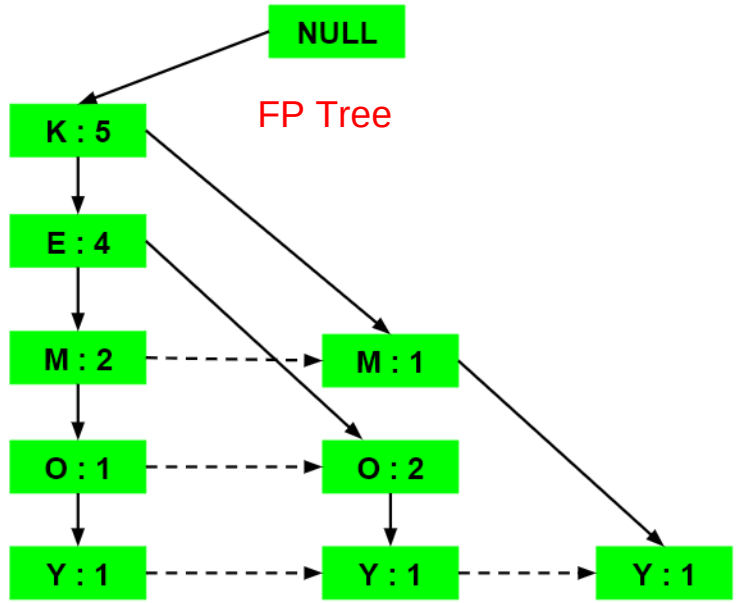
Original Set

Transaction ID	Items
T1	{E,K,M,N,O,Y}
T2	{D,E,K,N,O,Y}
T3	{A,E,K,M}
T4	{C,K,M,U,Y}
T5	{C,E,I,K,O,O}

Ordered Set as per Support Count

Transaction ID	Items	Ordered-Item Set
T1	{E,K,M,N,O,Y}	{K,E,M,O,Y}
T2	{D,E,K,N,O,Y}	{K,E,O,Y}
T3	{A,E,K,M}	{K,E,M}
T4	{C,K,M,U,Y}	{K,M,Y}
T5	{C,E,I,K,O,O}	{K,E,O}

Item Set with min_sup:3 = {K : 5, E : 4, M : 3, O : 3, Y : 3}



Items	Conditional Pattern Base
Y	{{K,E,M,O : 1}, {K,E,O : 1}, {K,M : 1}}
O	{{K,E,M : 1}, {K,E : 2}}
M	{{K,E : 2}, {K : 1}}
E	{K : 4}
K	

Items	Conditional Pattern Base	Conditional Frequent Pattern Tree
Y	{{K,E,M,O : 1}, {K,E,O : 1}, {K,M : 1}}	{K : 3}
O	{{K,E,M : 1}, {K,E : 2}}	{K,E : 3}
M	{{K,E : 2}, {K : 1}}	{K : 3}
E	{K : 4}	{K : 4}
K		

Items	Frequent Pattern Generated
Y	{<K,Y : 3>}
O	{<K,O : 3>, <E,O : 3>, <E,K,O : 3>}
M	{<K,M : 3>}
E	{E,K:4}
K	

Advantages Disadvantages of FP Growth

Advantages Of FP Growth Algorithm

1.This algorithm needs to scan the database only twice when compared to Apriori which scans the transactions for each iteration.
2.The pairing of items is not done in this algorithm and this makes it faster.
3.The database is stored in a compact version in memory.
4.It is efficient and scalable for mining both long and short frequent patterns.

Disadvantages Of FP-Growth Algorithm

1.FP Tree is more cumbersome and difficult to build than Apriori.
2.It may be expensive.
3.When the database is large, the algorithm may not fit in the shared memory.

Comparison of FP Growth with Apriori

FP Growth	Apriori
Pattern Generation	
FP growth generates pattern by constructing a FP tree	Apriori generates pattern by pairing the items into singletons, pairs and triplets.
Candidate Generation	
There is no candidate generation	Apriori uses candidate generation
Process	
The process is faster as compared to Apriori. The runtime of process increases linearly with increase in number of itemsets.	The process is comparatively slower than FP Growth, the runtime increases exponentially with increase in number of itemsets
Memory Usage	
A compact version of database is saved	The candidates combinations are saved in memory

Association Rule Mining Applications

- Market-basket analysis
  - Rules are used for sales promotion, shelf management, and inventory management
- Telecommunication alarm diagnosis
  - Rules are used to find combination of alarms that occur together frequently in the same time period
- Medical Informatics
  - Rules are used to find combination of patient symptoms and test results associated with certain diseases

Thank you