# Chapter 5
# Mining Frequent Pattern and Association Rule

**Based on CSC603.5:** Students should be able to apply various association rule mining techniques to real world systems.

**By, Safa Hamdare**

# Outline:

- Market Basket Analysis,
- Frequent Item sets, Closed Item sets, and Association Rule, Frequent Pattern Mining,
- Efficient and Scalable Frequent Item set Mining Methods:
  - **Apriori Algorithm:** Association Rule Generation, Improving the Efficiency of Apriori,
  - **Introduction to Mining Multilevel Association Rules and Multidimensional Association Rules**
  - **FP growth:** Mining frequent Item sets using Vertical Data Format,

# Introduction



- **Data mining** is the discovery of knowledge and useful information from the large amounts of data stored in databases.

- **Association Rules:** describing Association relationships among the attributes in the set of relevant data.

# Applications of Association

1. **Market Basket Analysis:** given a database of customer transactions, where each transaction is a set of items the goal is to find groups of items which are frequently purchased together.
2. **Telecommunication**: Each customer is a transaction containing the set of phone calls.
3. **Credit Cards/ Banking Services:** Each card/account is a transaction containing the set of customer's payments.
4. **Medical Treatments:** Each patient is represented as a transaction containing the ordered set of diseases.
5. **Basketball-Game Analysis:** Each game is represented as a transaction containing the ordered set of ball passes).

# Association rule mining

- Proposed by Agrawal et al in 1993.
- It is an important data mining model studied extensively by the database and data mining community.
- Assume all **data are categorical.**
- Initially used for Market Basket Analysis to find how items purchased by customers are related.

# Elucidate Market Basket Analysis with an example. (5marks) Dec 2019

# Market Basket Analysis (May 2012)

- **Market basket analysis** is a modeling technique based upon the theory that "*if you buy a certain group of items, you are more (or less) likely to buy another group of items*".
- **Example:** Market basket transaction data for Supermarket:

  t1: {bread, cheese, milk} t2: {apple, eggs, salt, yogurt} tn: {biscuit, eggs, milk}
- The information that the customer who purchase bread also tend to buy milk at the same time is represented in association rule below:

  Bread → Milk [sup = 5%, conf = 100%]
- The algorithm for performing market basket analysis are fairly straightforward.
- A major difficulty is that a large number of the rules found may be trivial for anyone familiar with the business.

# The model: Data

- **Concepts:**
  - An *item*: an item/article in a basket
  - *I*: the set of all items sold in the store
  - A *transaction*: items purchased in a basket; it may have TID (transaction ID)
  - A *transactional dataset*: A set of transactions

- A transaction *t* contains *X*, a set of items (item set) in *I*, if $X \subseteq t$.
- An association rule is an implication of the form:
  $X \rightarrow Y$, where $X, Y \subset I$, *and* $X \cap Y = \emptyset$

- An item set is a set of items.
  - E.g., X = {milk, bread, cereal} is an item set.

# Rule strength measures

1. **Support:** The support of an item set expresses *how often the item set appears in a single transaction* in the database, i.e. the support of an item (or set of item) is the percentage of transactions in which that item(or items) occurs.
   - The rule holds with support *sup* in *T* (the transaction data set) if sup% of transactions contain $X \cup Y$.
     - $sup = Pr(X \cup Y)$.
   - If A=> B , i.e. then the support for an association rule A=>B is the percentage of transactions in the dB that contains XUB, it is given as:

$$\text{support}(A => B) = \frac{\text{No.of tuples containing both A and B}}{\text{Total no.of tuples}}$$

   - An item set is considered to be a large item set if its support is above some threshold.

# Example: To Find Support

- **Database with transactions ( customer_# : item_a1, item_a2, ..) If min support threshold= 50%**

  **1: 1, 3, 5.**

  **2: 1, 8, 14, 17, 12.**

  **3: 4, 6, 8, 12, 9, 104.**

  **4: 2, 1, 8.**

  - **Support:** *how often the item set appears in a single transaction*

    **support {8,12} = 2 (,or 50% ~ 2 of 4 customers)**

    **support {1, 5} = 1 (,or 25% ~ 1 of 4 customers )**

    **support {1} = 3 (,or 75% ~ 3 of 4 customers)**

- An item set is called **frequent** if its **support** is **equal or greater** than an agreed upon minimal value – **the support threshold.**
  - then item sets {8,12} and {1} called **frequent with high Support**

# Rule strength measures

2. **Confidence:** *It is defined as the measure of certainty or trustworthiness associated with each discovered pattern.*
   - The confidence or strength for an association rule A=>B is the ratio of the number of transactions that contain A U B to the number of transactions that contain A.

$$\text{Confidence}(A => B) = \frac{\text{No.of tuples containing both A and B}}{\text{Total no.of tuples containing A}}$$

   - The rule holds in *T* with confidence *conf* if *conf*% of transactions 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.

# Example: To Find Confidence

- **Database with transactions ( customer_# : item_a1, item_a2, ..) If min Confidence threshold = 50%**

  **1: 1, 3, 5**

  **2: 1, 8, 14, 17, 12**

  **3: 4, 6, 8, 12, 9, 104**

  **4: 2, 1, 8**

  - **Confidence:** *for an association rule X=>Y is the ratio of the number of transactions that contain X U Y to the number of transactions that contain A* **(X => Y: if someone buys X, he also buys Y)**

    **Confidence {8=>12} = 2/3 (,or 66% ~ 2 of 3 customers)**

    **Confidence {1=>5} = 1/3 (,or 33% ~ 1 of 3 customers )**

- An item set is called **Strong** if its **Confidence** is **equal or greater** than an agreed upon minimal value **– the confidence threshold.**
  - then item sets {8,12} called **Strong, with high Confidence**

# Example 1: To Calculate Support & Confidence

Example: Database with transactions ( customer_# : item_a1, item_a2, … )

**If Min Confidence Threshold = 50% and Min Support Threshold = 30%**

1: 3, 5, 8.
2: 2, 6, 8.
3: 1, 4, 7, 10.
4: 3, 8, 10.
5: 2, 5, 8.
6: 1, 5, 6.
7: 4, 5, 6, 8.
8: 2, 3, 4.
9: 1, 5, 7, 8.
10: 3, 8, 9, 10.

**Find out Support & confidence for transaction {5,8} ?**

Support({5}) = 5 , Support ({8}) = 7 , Support({5,8}) = 4 **(or 40% ~ 4 of 10)**

**Confidence( {5} => {8} ) = 4/5 = 0.8 or 80%**

**Minimum Confidence 50% and Minimum support is 30% (given), then the rule are output**

# Example 2: To Calculate Support & Confidence

**Example:** Database with transactions ( customer_# : item_a1, item_a2, … )

**If Min Confidence Threshold = 50% and Min Support Threshold = 50%**

| | |
|---|---|
| **1:** | 3, 5, 8. |
| **2:** | 2, 6, 8. |
| **3:** | 1, 4, 7, 10. |
| **4:** | 3, 8, 10. |
| **5:** | 2, 5, 8. |
| **6:** | 1, 5, 6. |
| **7:** | 4, 5, 6, 8. |
| **8:** | 2, 3, 4. |
| **9:** | 1, 5, 7, 8. |
| **10:** | 3, 8, 9, 10. |

**Find out Support & confidence for transaction ( {9} => {3} ) ?**

Support({9}) = 1 , Support ({3}) = 4 , Support({9,3}) = 1 **(or 10% ~ 1 of 10)**

**Confidence( {9} => {3} ) = 1/1 = 1 or 100**

**Minimum Confidence 50% and Minimum support is 50% (given), then the rule are output :High Confidence, Low Support.**

**-> Rule ( {9} => {3} ) not meaningful**

# APRIORI ALGORITHM

- **Mining single-dimensional Boolean association rules from transactional databases**

# APRIORI ALGORTHM

- **APRIOIRI** *is an efficient algorithm to find association rules (or, actually, frequent item sets).*
- The Apriori technique is used for "**generating large item sets**." Out of all candidate (k)-item sets, generate all candidate (k+1)-item sets.
- **Method Used: (Candidate Generation-and-Test Approach)**
  - Initially, scan DB once to get frequent 1-itemset
  - Generate length (k+1) candidate item sets from length k frequent item sets
  - Test the candidates against DB for Minimum Support
  - Terminate when no frequent or candidate set can be generated

# Apriori Candidate Generation

- **Input:** Database D of Transactions; minimum support threshold (min_Sup), minimum Confidence threshold (min_Conf)
- **Output:** L, Frequent item sets in D
- **It has two steps:**
  - **Join Step**: To Find $L_k$, a set of candidate K-item sets is generated by Joining $L_{k-1}$ with itself, Generate all possible candidate item sets $C_k$ of length $k$
  - **Prune Step**: Remove those candidates in $C_k$ that cannot be frequent.

# The Apriori Algorithm



- Pseudo-code:

  $C_k$: Candidate item set of size k

  $L_k$ : frequent item set 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 $t$

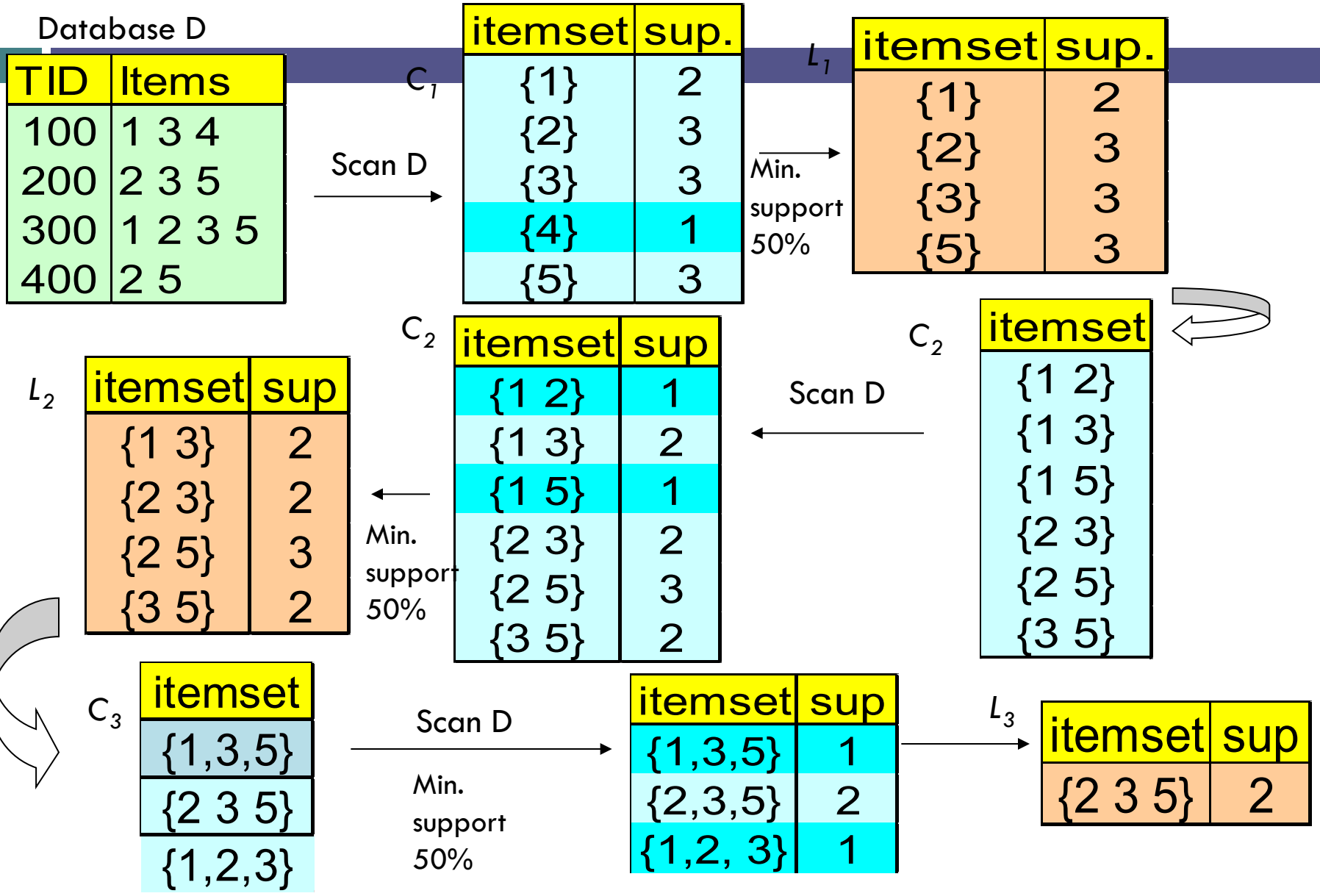
  $L_{k+1}$ = candidates in $C_{k+1}$ with min_support

  **end**

  **return** $\cup_k L_k$;

# The Apriori Algorithm - 1.Example

Min. support 50%
Min. confidence 50%



# The Apriori Algorithm-1.Example

Min. support 50%
Min. confidence 50%

- So data contain the frequent itemset1 {2,3,5}
- Therefore the association rule that can be generated from L3 are as shown below with the support and confidence.
- **Final Resulted Rules are:**

| Association Rule | Support | Confidence | Confidence % |
|---|---|---|---|
| {2,3}=>5 | 2 | 2/2=1 | 100% |
| {2,5}=>3 | 2 | 2/3=0.66 | 66% |
| {3,5}=>2 | 2 | 2/2=1 | 100% |
| {2}=>{3,5} | 2 | 2/3=0.66 | 66% |
| {3}=>{2,5} | 2 | 2/3=0.66 | 66% |
| {5}=>{2,3} | 2 | 2/3=0.66 | 66% |

*In this case all will be generated*

- **Note:** *If the minimum confidence threshold is 70% (given), then only the first and third rules above are output, since these are the only ones generated that are strong.*

# The Apriori Algorithm - 2.Example

Min. support 30%
Min. confidence 75%

Database D

| TID | Items |
|---|---|
| 1 | A, B, D |
| 2 | B,C,D |
| 3 | A,B |
| 4 | B,D |
| 5 | A,B,C |

$C_1$

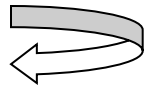
Scan D →

$L_1$

→ Min. support 50%

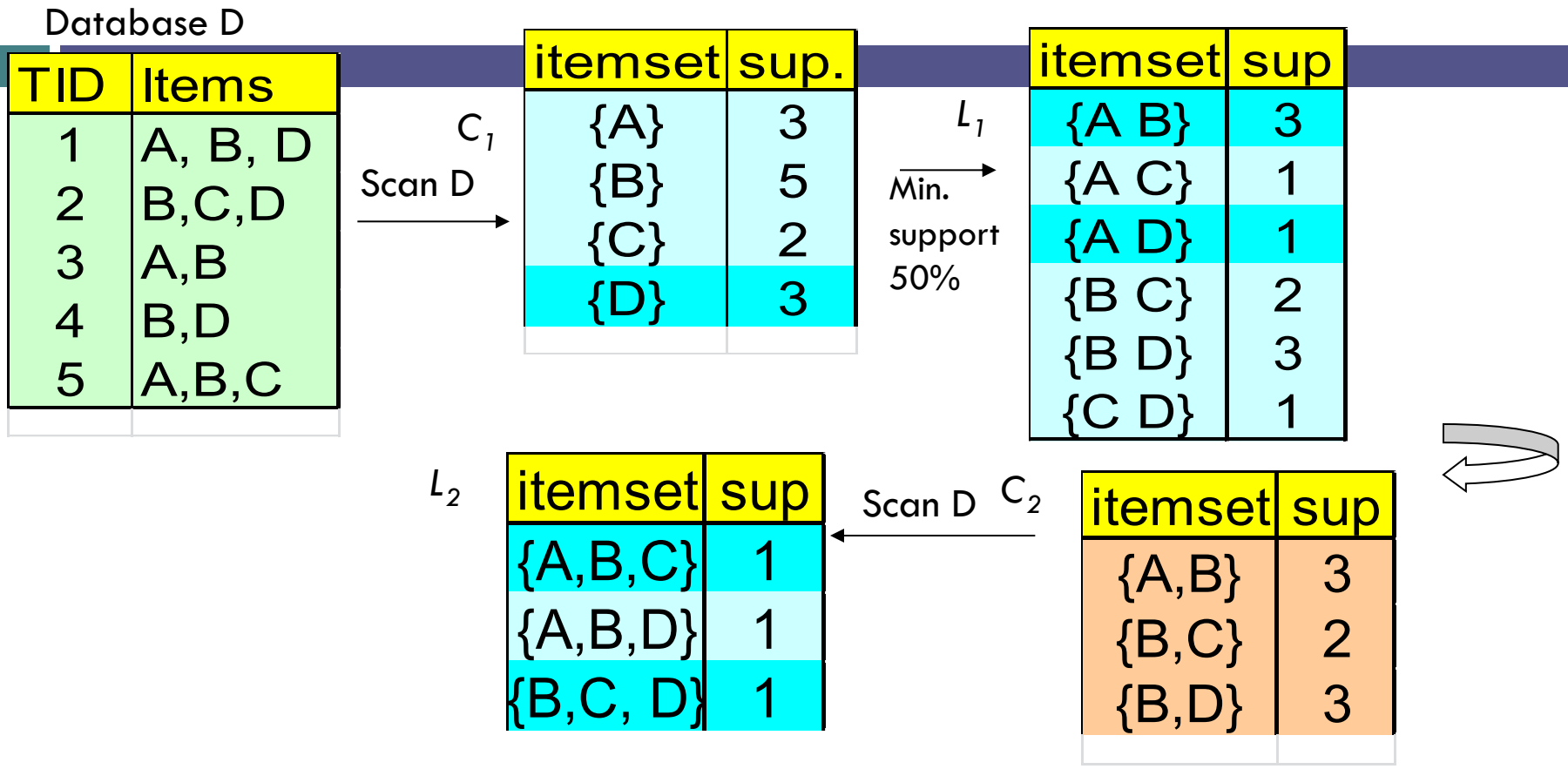
$L_2$

← Scan D $C_2$



# The Apriori Algorithm - 2.Example

Min. support 30%
Min. confidence 75%




Since all the combinations are below minimum support of 30%, so no rules for association can be generated.

# Apriori Advantages/Disadvantages

- **Advantages:**
  - Uses large item set property
  - Easily parallelized
  - Easy to implement
- **Disadvantages:**
  - Assumes transaction database is memory resident
  - Requires many database scans

# University Question on Apriori Algorithm

1. Explain how Apriori Algorithm is useful in identifying frequent item set? **(Dec 2011)**
2. What is Association Rule Mining? Give the Apriori Algorithm. Apply AR mining to find all frequent item sets from the following table: **(May 2011, May 2016)**

| Transcation – ID | Items |
|---|---|
| 100 | 1, 2, 5 |
| 200 | 2, 4 |
| 300 | 2, 3 |
| 400 | 1, 2, 4 |
| 500 | 1, 3 |
| 600 | 1, 3 |
| 700 | 1, 3, 2, 5 |
| 800 | 1, 3 |
| 900 | 1, 2, 3 |

Minimum support count=2

Minimum- confidence=70%

**(Here calculate the % i.e. 2/9=22%)**

# University Question on Apriori Algorithm

3. Explain what is meant by association rule mining. For the table given below perform Apriori algorithm. Also **(June 2010, May 2012, May 2018)**
   - Determine the K-item sets (frequent) obtained.
   - Justify the strong association rule that has been determined i.e. specify which is the strongest rule obtained.

- The table is as follows:

| TID | Items |
|---|---|
| 01 | 1, 3, 4, 6 |
| 02 | 2, 3, 5, 7 |
| 03 | 1, 2, 3, 5, 8 |
| 04 | 2, 5, 9, 10 |
| 05 | 1, 4 |

- Assume Minimum Support of 30% and Minimum confidence of 70%.

# University Question on Apriori Algorithm

4. Consider the five transactions given below. If minimum support is 30% and minimum confidence is 80%, determine the frequent item sets and association rules using a priori algorithm. **(Dec 2012)**

| Transaction | Items |
|---|---|
| T1 | Bread, Jelly, Butter |
| T2 | Bread, Butter |
| T3 | Bread, Milk, Butter |
| T4 | Coke, Bread |
| T5 | Coke, Milk |

5. Apply the Apriori algorithm with minimum support of 30% and minimum confidence of 70%, and find all the association rules in the data set. **(May 2013, May 2017)**

Consider the following transaction database:

| TID | Items |
|---|---|
| 01 | A, B, C, D |
| 02 | A, B, C, D, E, G |
| 03 | A, C, G, H, K |
| 04 | B, C, D, E, K |
| 05 | D, E, F, H, L |
| 06 | A, B, C, D, L |
| 07 | B, I, E, K, L |
| 08 | A, B, D, E, K |
| 09 | A, E, F, H, L |
| 10 | B, C, D, F |

| Transaction | Items |
|---|---|
| T1 | Bread, Jelly, Butter |
| T2 | Bread, Butter |
| T3 | Bread, Milk, Butter |
| T4 | Coke, Bread |
| T5 | Coke, Milk |

minimum support is 30% and minimum confidence is 80%

| | |
|---|---|
| Bread | 4/5=80% |
| Jelly | 1/5=20% |
| Butter | 3/5=60% |
| Milk | 2/5=40% |
| Coke | 2/5=40% |

→

| | |
|---|---|
| Bread, Butter | 3= 60% |
| Bread, Milk | 1=20% |
| Bread, Coke | 1=20% |
| Butter, Milk | 1=20% |
| Butter, Coke | 0% |
| Coke, Milk | 1=20% |

↓

| | |
|---|---|
| Bread, Butter | 3= 60% |

Frequent item set are { Bread, Butter}

Association Rule:

Bread-> Butter = ¾ =75%

Butter-> Bread =3/ 3=100%

Final Association rule is:

Butter-> Bread =3/ 3=100%

| T_ID | Items bought |
|---|---|
| T-1000 | M, O, N , K, E, Y |
| T-1001 | D, O, N, K, E, Y |
| T-1002 | M, A, K, E |
| T-1003 | M, U, C, K, Y |
| T-1004 | C, O, O, K, E |

min-support=60% and min-confidence = 80%.

| | |
|---|---|
| M | 3 |
| O | 3 |
| N | 2 |
| K | 5 |
| E | 4 |
| Y | 3 |
| D | 1 |
| A | 1 |
| U | 1 |
| C | 2 |

| | |
|---|---|
| M,O-1 | O,E-3 |
| M,K-3 | O,Y-2 |
| M,E- 2 | K,E-4 |
| M,Y-2 | K,Y-3 |
| O,K--3 | E,Y--2 |

M,K,O- 1
M,K,E- 2
O,K,E- 3
O,K,Y- 2
M,K,Y- 2
M,E,Y- 1
K,E,Y- 2
O,E,Y-2
M,O,Y- 1

Association Rule:

O-> K,E 3/3
K-> O,E 3/5
E->K,O 3/4
O,K->E 3/3
O,E->K 3/3
K,E->O 3/4

Frequent Item Set: O,K,E- 3

So final Association rules are:

O-> K,E 3/3
O,K->E 3/3
O,E->K 3/3

# University Question on Apriori Algorithm

i) Discuss Association Rule Mining and Apriori Algorithm.

ii) A database has four transactions. Let minimum support = 50% and minimum confidence = 50%

**Dec 2017**

| TID | Items-bought |
|---|---|
| T100 | A,B,C |
| T200 | A,C |
| T300 | A,D |
| T400 | B,E,F |

Find all frequent item sets using apriori algorithm. List strong association rules.

# University Question on Apriori Algorithm

☐**What is meant by market-basket analysis? Explain with an example. State and explain with formula the meaning of the terms: (May 2012)**

**1.Support**

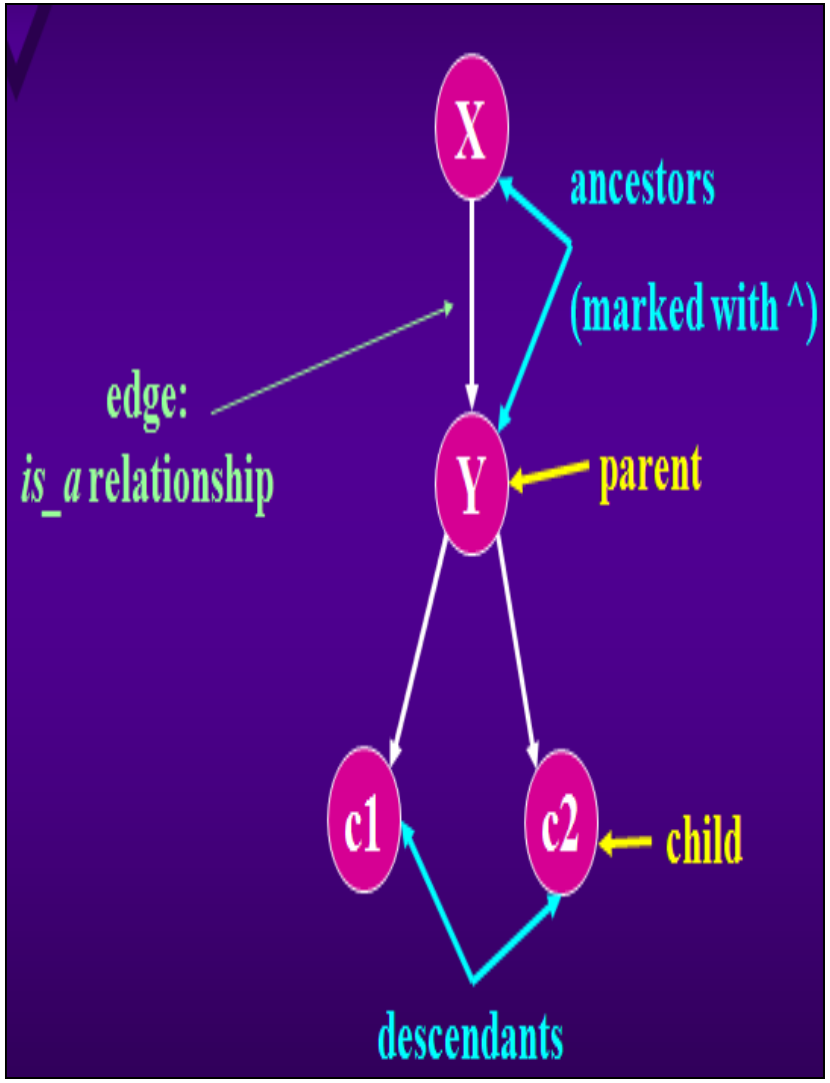
**2.Confidence**

**Hence explain how to mine multi-level association rules from transaction databases, with example for each.**

# Mining multilevel association rules

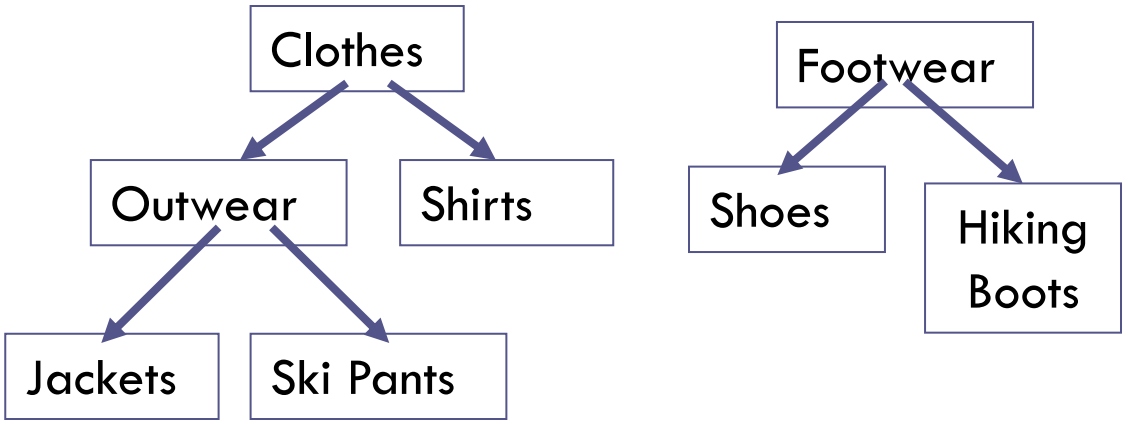
# 1. Generalized Association Rules (May 2011)

- Here the rules are allowed at different levels.
- **A generalized association rule:** $X \rightarrow Y$ if $X \subset I$ , $Y \subset I$ , $X \cap Y = \varnothing$ , and no item in Y is an ancestor(i.e. above ) any item in X.
- The rule $X \rightarrow Y$ has confidence c in D if c% of transactions in D that support X also support Y.
- The rule $X \rightarrow Y$ has support s in D if s% of transactions in D supports $X \cup Y$.
- When generating generalized association rules, all possible rules are generated using one or more given hierarchies.



# 1. Generalized Association Rules (contd..)

- Hierarchy shows that Jacket & Ski-pants is-a Outwear, Outwear is-a clothes.
- **Users are interested in generating rules that span different levels of the hierarchy.**
- **For e.g. We may Infer that people who buy outwear tend to buy Hiking boots from the facts that people bought Jackets with Hiking boots and ski-pants with Hiking boots.**



# Example



**Database D**

| Transaction | Items Bought |
|---|---|
| 100 | Shirt |
| 200 | Jacket, Hiking Boots |
| 300 | Ski Pants, Hiking Boots |
| 400 | Shoes |
| 500 | Shoes |
| 600 | Jacket |

**Frequent Itemsets**

| Itemset | Support |
|---|---|
| {Jacket} | 2 |
| {Outwear} | 3 |
| {Clothes} | 4 |
| {Shoes} | 2 |
| {Hiking Boots} | 2 |
| {Footwear} | 4 |
| {Outwear, Hiking Boots} | 2 |
| {Clothes,Hiking Boots} | 2 |
| {Outwear, Footwear} | 2 |
| {Clothes, Footwear} | 2 |

**Rules**

| Rule | Support | Confidence |
|---|---|---|
| Outwear ➔ Hiking Boots | 33% | 66.6% |
| Outwear ➔ Footwear | 33% | 66.6% |
| Hiking Boots ➔ Outwear | 33% | 100% |
| Hiking Boots ➔ Clothes | 33% | 100% |

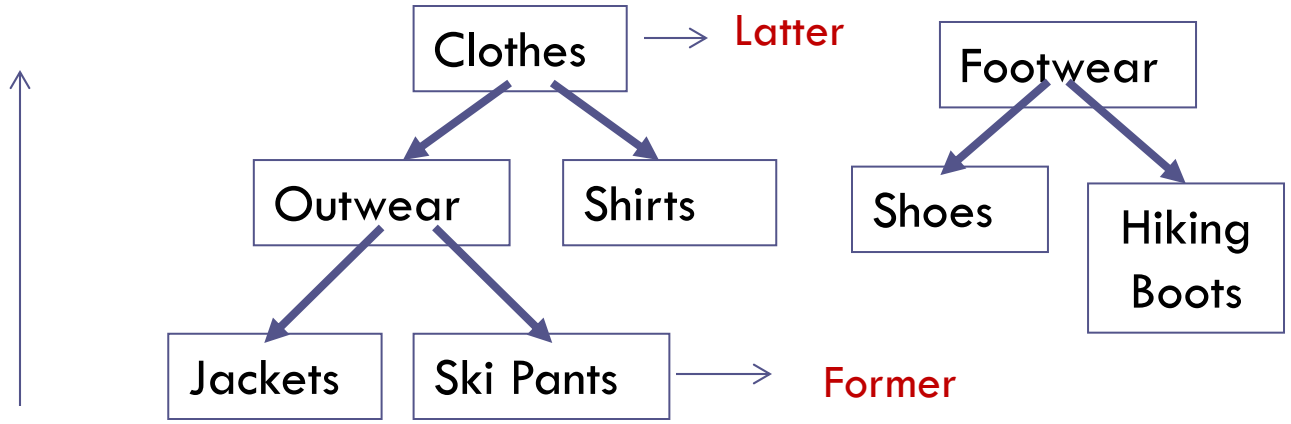
***minsup* = 30%**

***minconf* = 60%**

# Taxonomy - Example



- Let say we found the rule: Outwear ➔ Hiking Boots with minimum support and confidence.
- The rule Jackets ➔ Hiking Boots may not have minimum support
- The rule Clothes ➔ Hiking Boots may not have minimum confidence.
  - Rules of lower levels may not have minimum support and Rules of Higher levels may not have minimum Confidence
  - Because the former may not have minimum Support, and the later may not have minimum Confidence.

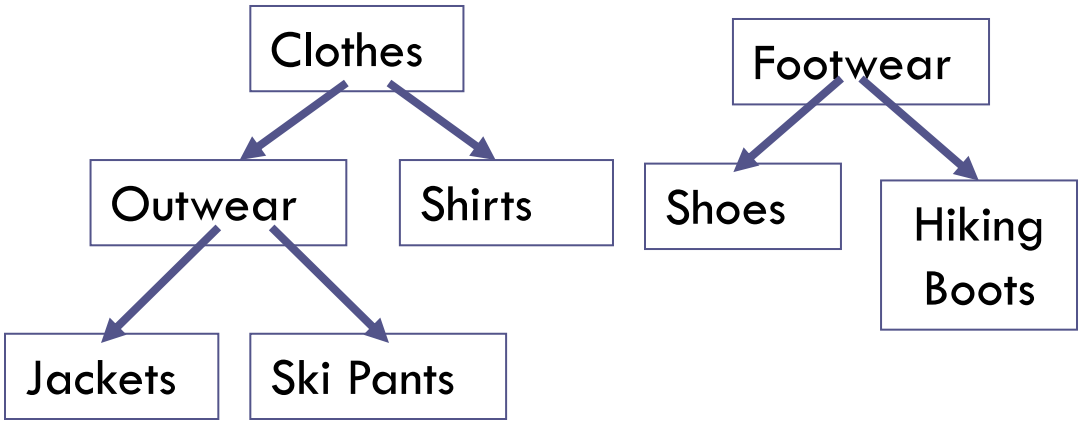


# Observation 1



- If the set{x,y} has minimum support,
  - so do {x^, y} {x,y^} and {x^,y^}
- **For example:**

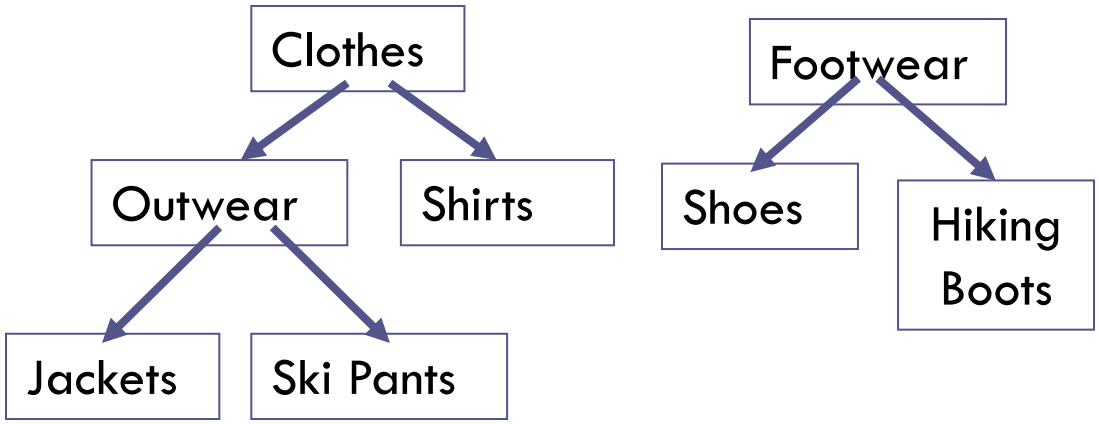
  If {Jacket, Shoes} has minsup,
  - so will {Outwear, Shoes}, {Jacket, Footwear}, and {Outwear, Footwear}



# Observation 2



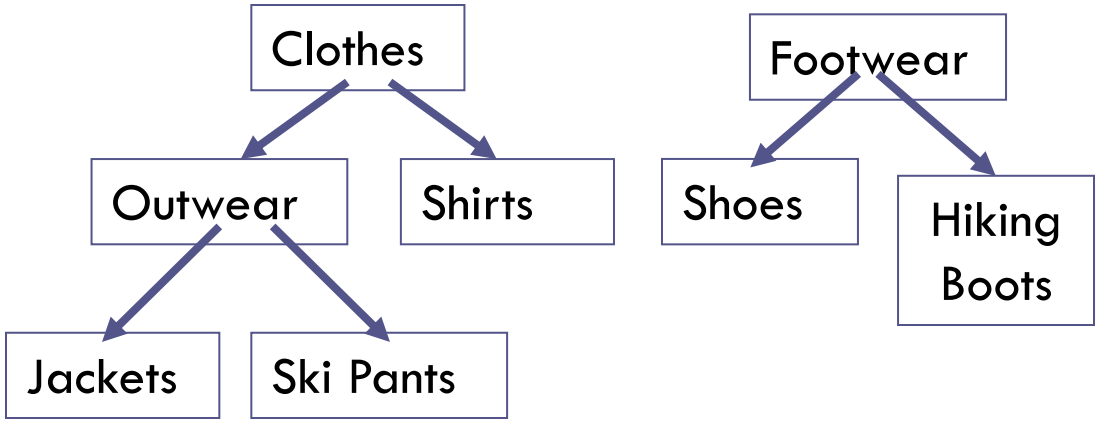
- If the rule x→y has minimum support and confidence, only x→y^ is guaranteed to have both minsup and minconf.
- **For e.g.** The rule Outwear→Hiking Boots has minsup and minconf.
  - The rule Outwear→Footwear has both minsup and minconf.



# Observation 2 – cont.

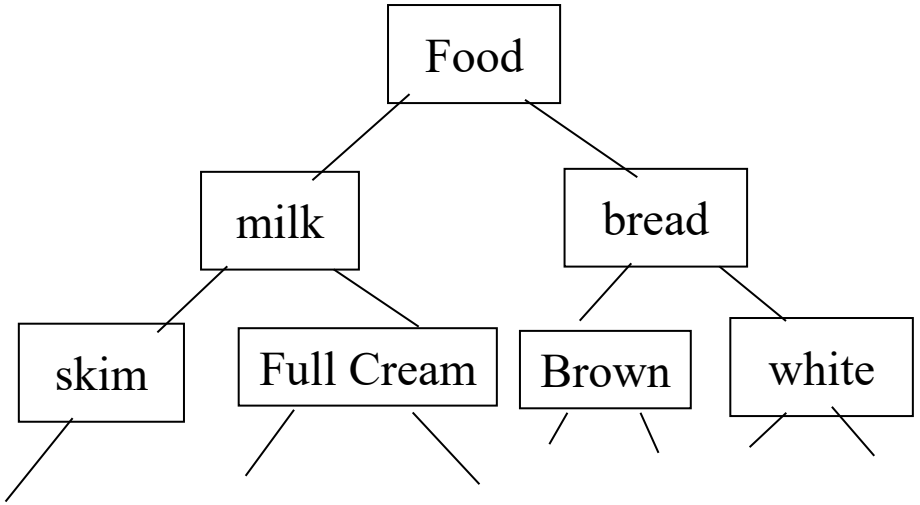


- However, the rules x^➔y and x^➔y^ will have minsup, they may not have minconf.
- For example:
  The rules Clothes➔Hiking Boots and Clothes➔Footwear have minsup, but not minconf.



# Multiple-Level Association Rules (May 2012)

- Items often form hierarchy.
- Items at the lower level are expected to have lower support.
- Rules regarding item sets at appropriate levels could be quite useful.
- We can explore shared multi-level mining.
- Rules which combine association with hierarchy of concepts are called Multilevel Association Rules.

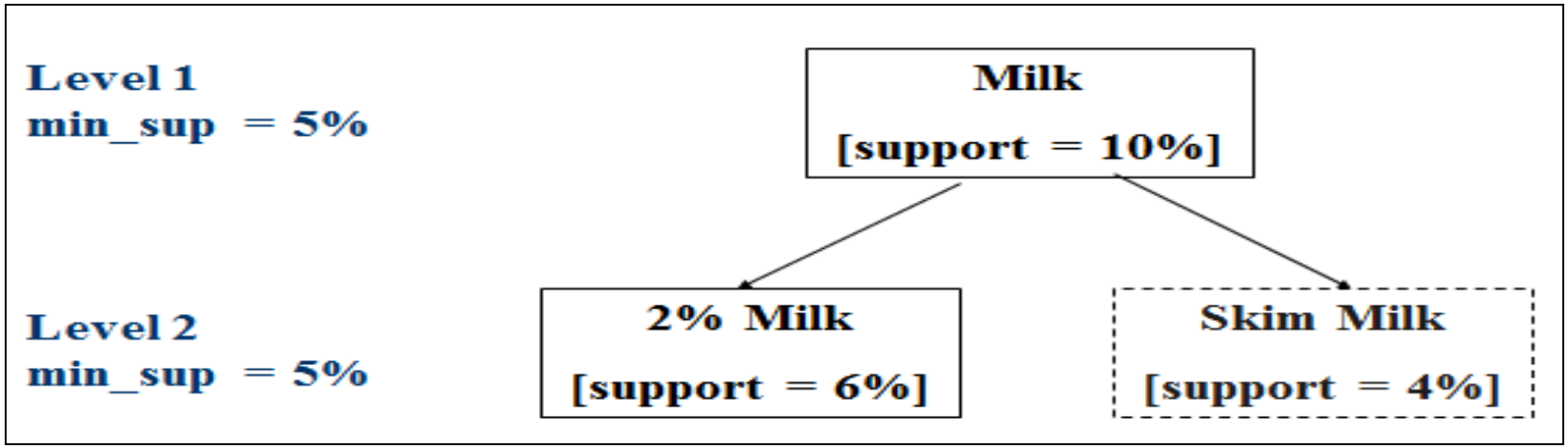



| Hierarchy | Items |
|---|---|
| Department | Food |
| Type of Food | Milk, Bread |
| Variety of food | Milk (Skim, Full Cream)<br>Bread (White, Brown) |

# Two Approaches of Multi level Association Rule

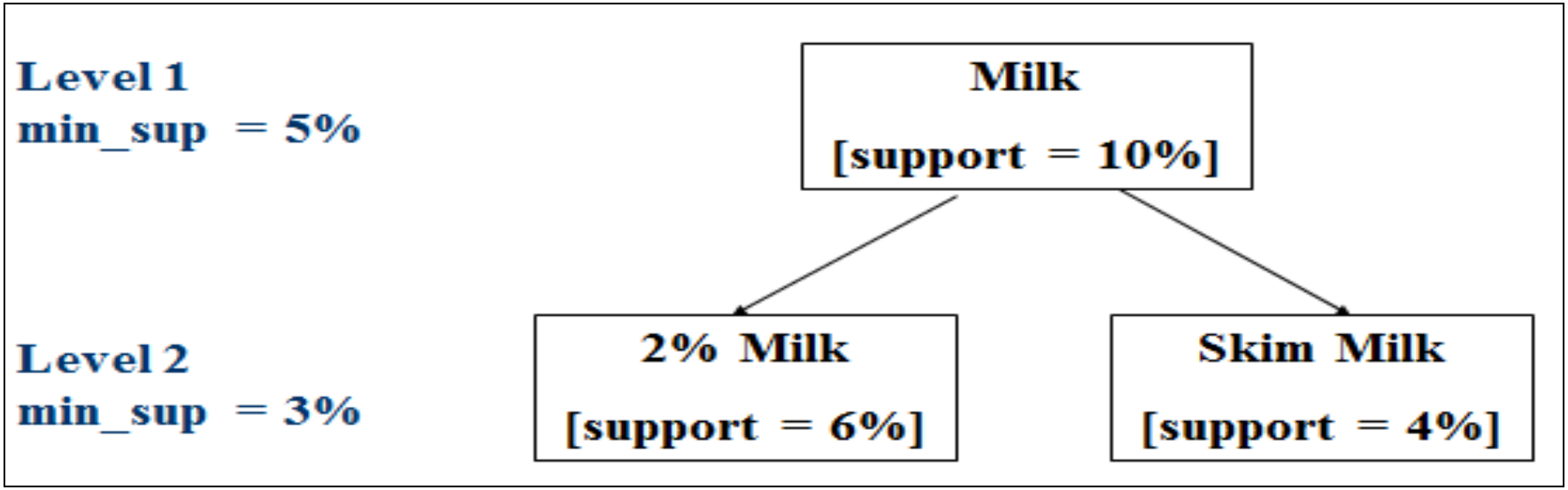
1. **Using Uniform Minimum Support for all Levels:**

- The same minimum support for all levels.
- There is only one minimum support threshold so no need to examine item sets containing any item whose ancestors do not have minimum support.
  - **If support threshold is too high-** miss low level association
  - **If support threshold is too low-** generate too many high level association



# Two Approaches of Multi level Association Rule

2. **Reduced Support: reduced minimum support at lower levels:**
   - At every level of abstraction, there is its own minimum support threshold
   - So minimum support at lower levels reduces.



# Multi level Association Rule :

## * Reduced Support different approaches

- **There are 4 search strategies:**

1. **Level-by-level independent:** In this case each node is examined though its parent node is frequent or not frequent. This is a *full-breadth search method.* *(Everything to be Scanned)*
2. **Level-cross filtering by k-item set:** If the node is frequent then only its children will be examined. *(Food is frequent then both Milk and bread is examined)*
3. **Level-cross filtering by single item:** In this case k item set at the ith level is examined if k item set at the (i-1) i.e. parent level is frequent. *(Examine Skim Milk, full cream; only if Milk is frequent)*
4. **Controlled level-cross filtering by single item:** Modified version of type 3. There is a "Level Passage Threshold" for relatively frequent items to lower level. So the items which do not satisfy minimum support threshold are examined for "Level Passage Threshold". *(Examine Skim milk , full cream; even if milk is not satisfying the minimum support threshold)*

# Multi Dimensional association rules

# Multi-dimensional Association



- **Single-dimensional (or Intra-dimension)** association rules: single distinct predicate (buys)

  **buys(X, "milk") ⇒ buys(X, "bread")**

- **Multi-dimensional rules**: multiple predicates
  - **Inter-dimension** association rules (*no repeated predicates*)

    **age(X,"20-29") ∧ occupation(X,"student") ⇒ buys(X,"laptop")**
  - **Hybrid-dimension** association rules (*repeated predicates*)

    **age(X,"20-29") ∧ buys(X, "laptop") ⇒ buys(X, "printer")**

- Rules like these are called *multidimensional association rules*. The dimensions represent attributes of records of a file or, in terms of relations, columns of rows of a relation, and can be categorical or quantitative.

# Multi-dimensional Association



- **Database attributes can be *categorical* or *quantitative***
- **Categorical (nominal)** Attributes
  - finite number of possible values, no ordering among the values
  - e.g., occupation, brand, color
- **Quantitative** Attributes
  - numeric, implicit ordering among values
  - e.g., age, income, price
- ***Techniques for mining multidimensional association rules can be categorized according to three basic approaches regarding the treatment of quantitative attributes.***

# Techniques for Mining MD Associations



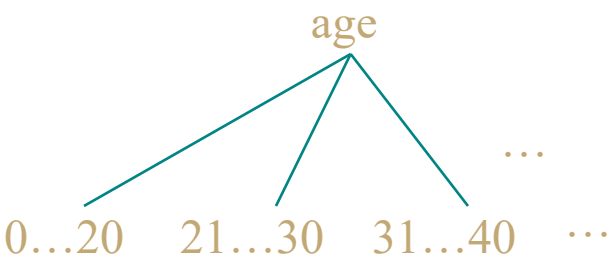
1. **Quantitative attributes are statically discretized using predefined concept hierarchies**
   - treat the numeric attributes as categorical attributes
   - a concept hierarchy for income: "0…20K", "21K…30K",…
2. **Quantitative attributes are dynamically discretized into "bins" based on the distribution of the data**
   - treat the numeric attribute values as quantities
   - quantitative association rules
3. **Quantitative attributes are dynamically discretized so as to capture the semantic meaning of such interval data**
   - consider the distance between data points
   - distance-based association rules

## Techniques for Mining MD Associations

### 1. Quantitative attributes are statically discretized using predefined concept hierarchies



- Search for frequent *k*-predicate set:
  - Example: {age, occupation, buys} is a 3-predicate set
  - Techniques can be categorized by how age are treated



## Techniques for Mining MD Associations

### 1. Quantitative attributes are statically discretized using predefined concept hierarchies



- Discretized prior to mining using concept hierarchy.
- Numeric values are replaced by ranges (categories).
- Numeric attributes are dynamically discretized and may later be further combined during the mining process
- 2-D quantitative association rules: $A_{quan1} \wedge A_{quan2} \Rightarrow A_{cat}$
- Example
  - age(X,"30-39") $\wedge$ income(X,"42K - 48K") $\Rightarrow$ buys(X," high resolution TV")

## Techniques for Mining MD Associations

**2. Quantitative attributes are dynamically discretized into "bins" based on the distribution of the data**



- Numeric attributes are dynamically discretized and may later be further combined during the mining process
- 2-D quantitative association rules: $A_{quan1} \wedge A_{quan2} \Rightarrow A_{cat}$
- Example
  - age(X,"30-39") $\wedge$ income(X,"42K - 48K") $\Rightarrow$ buys(X,"high resolution TV")

# Techniques for Mining MD Associations

## 2. Quantitative attributes are dynamically discretized into "bins" based on the distribution of the data



- The ranges of quantitative attributes are partitioned into intervals
- The partition process is referred to as binning
- Binning strategies
  - Equiwidth binning, the interval size of each bin is the same
  - Equidepth binning, each bin has approximately the same number of tuples assigned to it
  - Homogeneity-based binning, bin size is determined so that the tuples in each bin are uniformly distributed

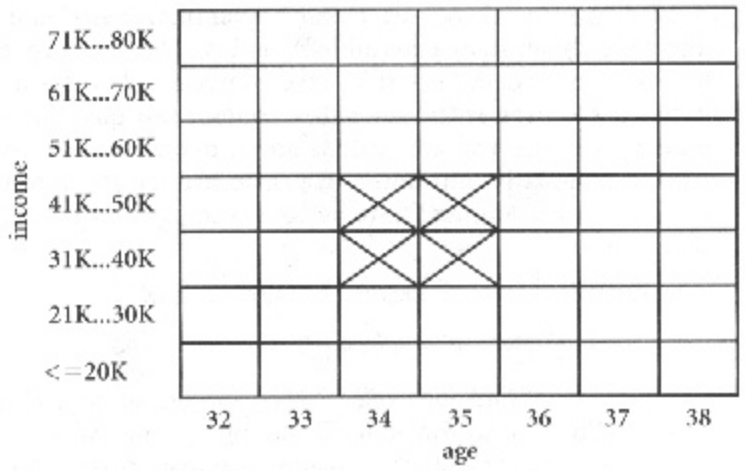
| Price($) | Equi-width (width $10) | Equi-depth (depth 2) |
|---|---|---|
| 7 | [0,10] | [7,20] |
| 20 | [11,20] | [22,50] |
| 22 | [21,30] | [51,53] |
| 50 | [31,40] | |
| 51 | [41,50] | |
| 53 | [51,60] | |

# Techniques for Mining MD Associations

## 2. Quantitative attributes are dynamically discretized into "bins" based on the distribution of the data



- Cluster "adjacent" association rules to form general rules using a 2-D grid
  - age(X,34) $\wedge$ income(X,"31K - 40K") $\Rightarrow$ buys(X,"high resolution TV")
  - age(X,35) $\wedge$ income(X,"31K - 40K") $\Rightarrow$ buys(X,"high resolution TV")
  - age(X,34) $\wedge$ income(X,"41K - 50K") $\Rightarrow$ buys(X,"high resolution TV")
  - age(X,35) $\wedge$ income(X,"41K - 50K") $\Rightarrow$ buys(X,"high resolution TV")
- Clustered to form
  - age(X,"34…35") $\wedge$ income(X,"31K - 50K") $\Rightarrow$ buys(X,"high resolution TV")



# Techniques for Mining MD Associations

## 2. Quantitative attributes are dynamically discretized into "bins" based on the distribution of the data



- Binning methods do not capture the semantics of interval data

| Price($) | Equi-width (width $10) | Equi-depth (depth 2) | Distance-based |
|---|---|---|---|
| 7 | [0,10] | [7,20] | [7,7] |
| 20 | [11,20] | [22,50] | [20,22] |
| 22 | [21,30] | [51,53] | [50,53] |
| 50 | [31,40] | | |
| 51 | [41,50] | | |
| 53 | [51,60] | | |

- Distance-based partitioning, more meaningful discretization considering:
  - density/number of points in an interval
  - "closeness" of points in an interval

# Techniques for Mining MD Associations

## 3. Quantitative attributes are dynamically discretized so as to capture the semantic meaning of such interval data



- Intervals for each quantitative attribute can be established by *clustering* the values for the attribute
- The support and confidence measures do not consider the closeness of values for a given attribute
  - Item_type(X,"electronic") $\wedge$ manufacturer(X,"foreign") $\Rightarrow$ price(X,$200)
- Distance-based association rules capture the semantics of interval data while allowing for approximation in data values
  - The prices of foreign electronic items are close to or approximately $200 rather than exactly $200

# University Asked Question

1. Write short note on Advanced Association Rules. (Dec 2011)
   - Generalized Association Rule
   - Multiple level Association Rule
   - Multi-dimensional Association Rule
2. Write short note on Generalized Association Rules. (May 2011)
3. Write short note on multidimensional and multi level association mining. May 2018
4. Demonstrate Multidimensional and Multilevel Association Rule Mining with suitable examples. Dec 2019

# Goal and key features (Association Rule Mining)

- **Goal:** Find all rules that satisfy the user-specified *minimum support* (minsup) and *minimum confidence* (minconf).
- **Key Features**
  - Completeness: find all rules.
  - No target item(s) on the right-hand-side

# Final Association rules

1. We should only consider **rules** derived from item sets with **high support,** and that also **have high confidence.**
2. "A rule with low confidence is not meaningful."
3. Rules don't explain anything, they just point out hard facts in data volumes.

# Frequent Pattern Tree

Write short note on FP Tree.

(10 marks, Dec 2017, May 2016, May 2017)

# Mining Frequent Item sets Concept

- If we use candidate generation
  - Need to generate a huge number of candidate sets.
  - Need to repeatedly scan the database and check a large set of candidates by pattern matching.
- Can we avoid that?
  - **FP-Trees (Frequent Pattern Trees)**

# FP (Frequent Pattern)-Tree

- **General Idea**
  - Divide and Conquer
- **FP-Tree Construction Algorithm**
  - **Input:** a transaction dB and a minimum support threshold
  - **Output:** its frequent Pattern Tree i.e. FP Tree
  - **Method:**
    - For each item create its conditional pattern base **(Contains transactions in which an element e.g. p occurs)**
    - Then expand it and create its conditional FP-Tree **(Finding all frequent patterns containing 'p' for this find all frequent patterns in CPB)**
    - Repeat the process recursively on each constructed FP-Tree
    - Until FP-Tree is either empty or contains only one path
    - Every branch represents a frequent pattern.

# FP (Frequent Pattern)-Tree Algorithm

- **Algorithm:**
  1. **Scan DB Once:**
     - Collect the set of frequent items F and their Supports
     - Sort F in support descending order as L, the list of Frequent items
  2. **Create a root of an FP-Tree, T, and Label it as "Null"**
     - Select and sort the frequent items in Transaction dB according to the order of L
     - Let the sorted frequent item list in Trans be [p|P], where p is the first element and P is the remaining List.
  3. **Insert_tree ([p |P],T) is performed as follows:**
     - If T has a child N such that N.item_name=p.Item_name, then increment N's Count by 1
     - Else create a new node N, and let its count be 1, its parent link be linked to T, and its node-link be linked to the nodes with the same item_name
     - If P is nonempty, call insert_tree(P,N) recursively.

# Example 1: Finding all the patterns with 'p' in the FP Tree given below

| Transaction | Items |
|---|---|
| T1 | a, c, d, f, g, i, m, p |
| T2 | a, b, c, f, l, m, o |
| T3 | b, f, h, j, o |
| T4 | b, c, k, n, p |
| T5 | a, c, e, f, l, m, n, p |

Min support = 60%

# Step 1

Find support of every item.

| Item | Support | Item | Support |
|---|---|---|---|
| a | 3 | i | 1 |
| b | 3 | j | 1 |
| c | 4 | k | 1 |
| d | 1 | l | 2 |
| e | 1 | m | 3 |
| f | 4 | n | 2 |
| g | 1 | o | 2 |
| h | 1 | p | 3 |

# Step 2

- Choose items that are above min support and arrange them in descending order (min support=3).

| item | support |
|---|---|
| f | 4 |
| c | 4 |
| a | 3 |
| b | 3 |
| m | 3 |
| p | 3 |

- (f:4,c:4,a:3,b:3,m:3,p:3)

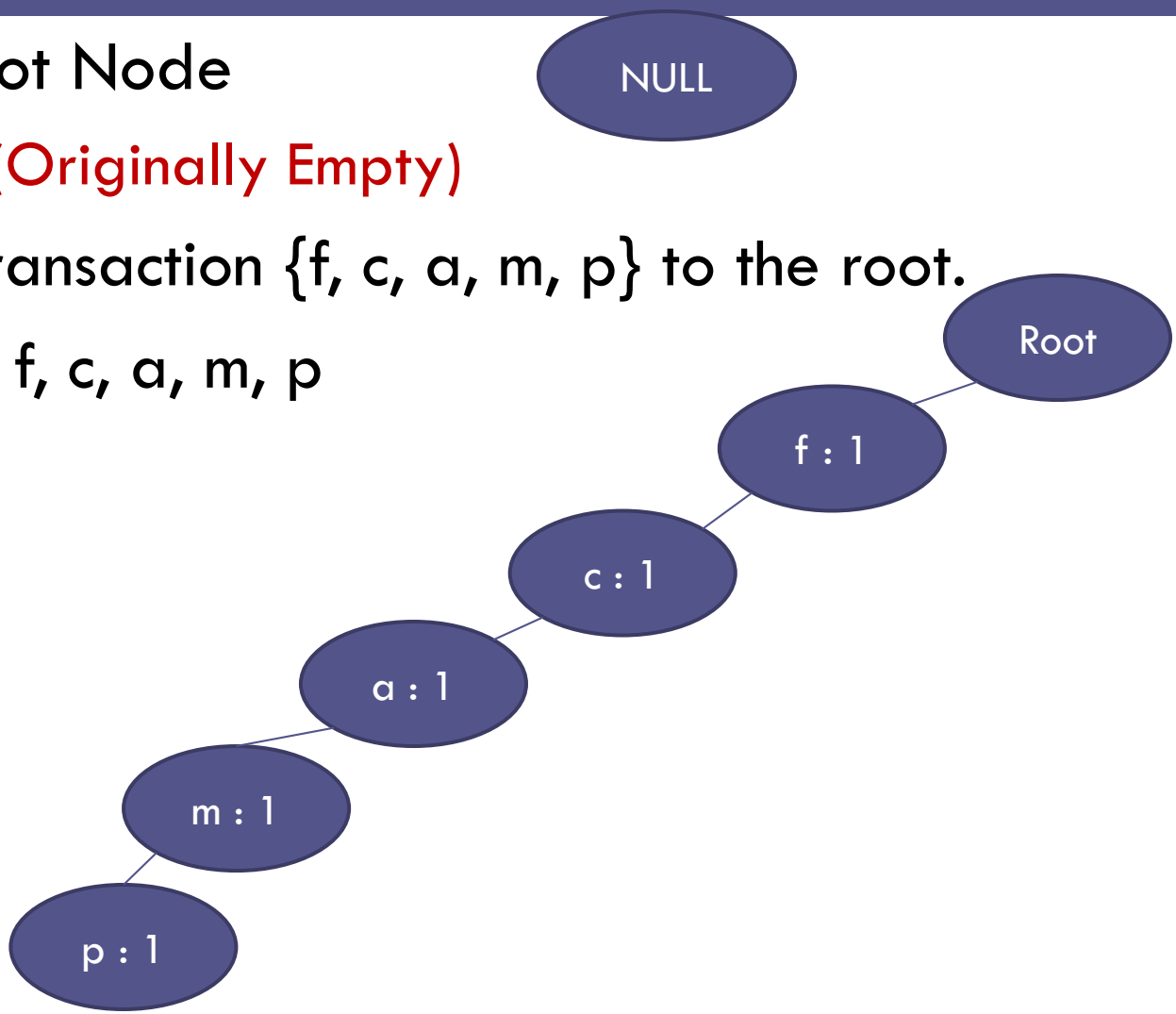
# Step 3

- Rewrite transactions with only those items that have more than min support, in descending order *( rest of the items are discarded)*

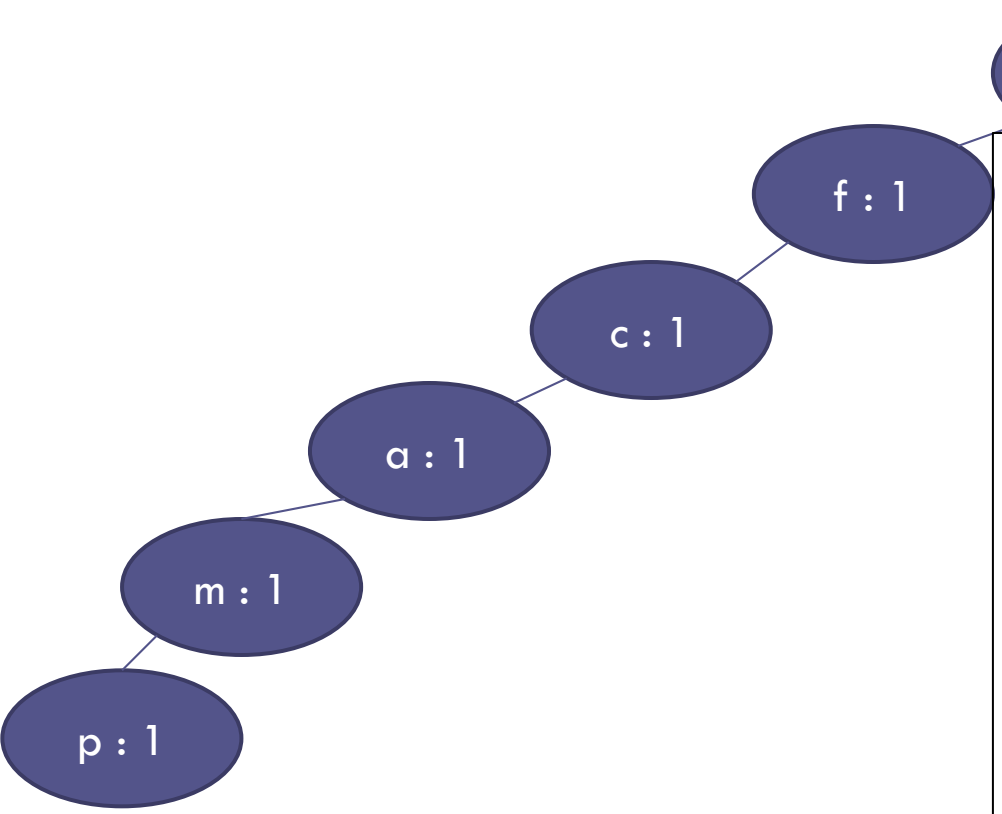
| Transaction | Items | Frequent items |
|---|---|---|
| T1 | a, c, d, f, g, i, m, p | f, c, a, m, p |
| T2 | a, b, c, f, l, m, o | f, c, a, b, m |
| T3 | b, f, h, j, o | f, b |
| T4 | b, c, k, n, p | c, b, p |
| T5 | a, c, e, f, l, m, n, p | f, c, a, m, p |

# Step 4: FP tree Construction-Insert T1

- Introduce a Root Node
  - Root = NULL (Originally Empty)
- Give the first transaction {f, c, a, m, p} to the root.
- Transaction T1: f, c, a, m, p



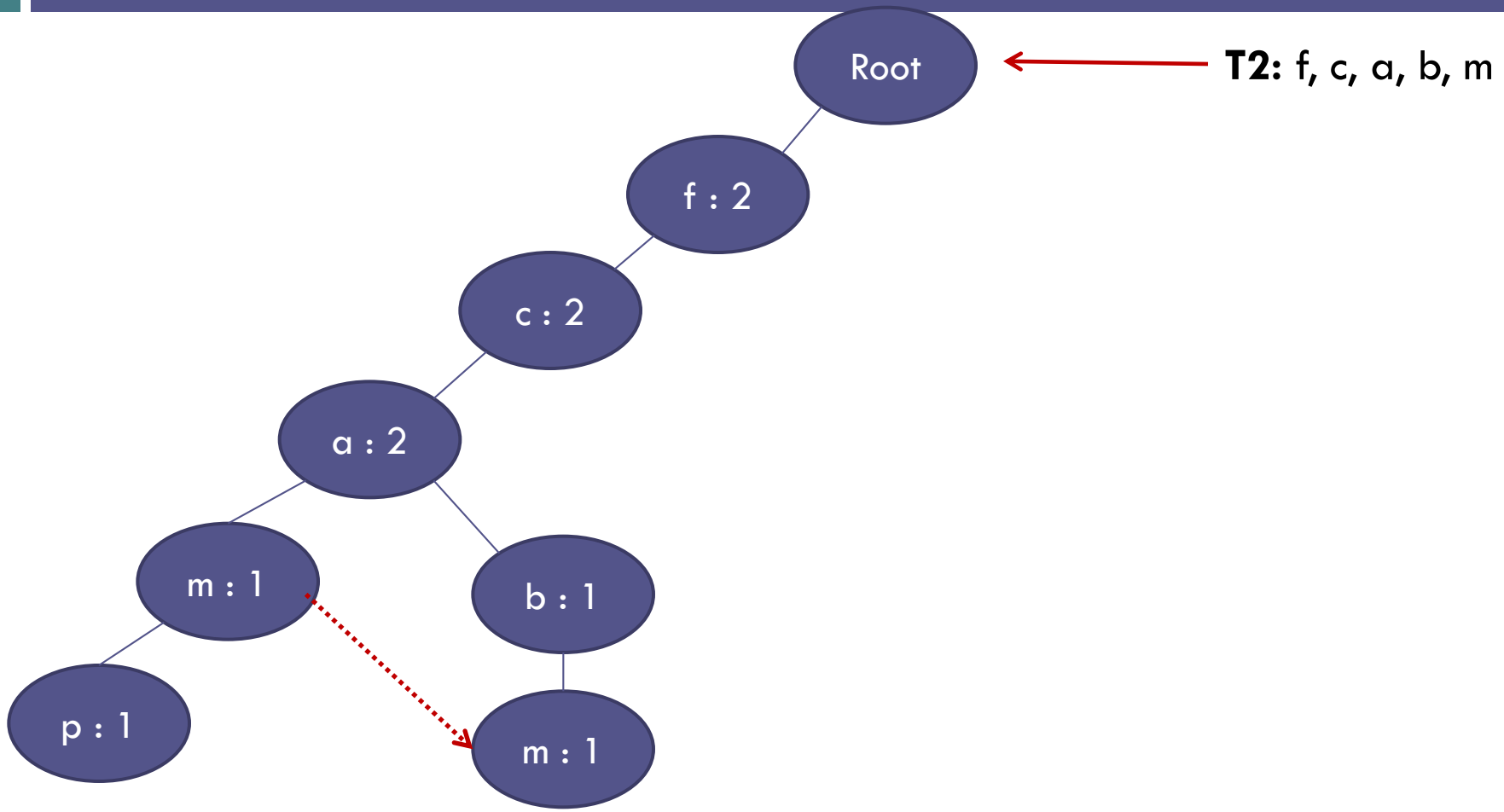
# Step 5: Insert Transaction T2: f, c, a, b, m



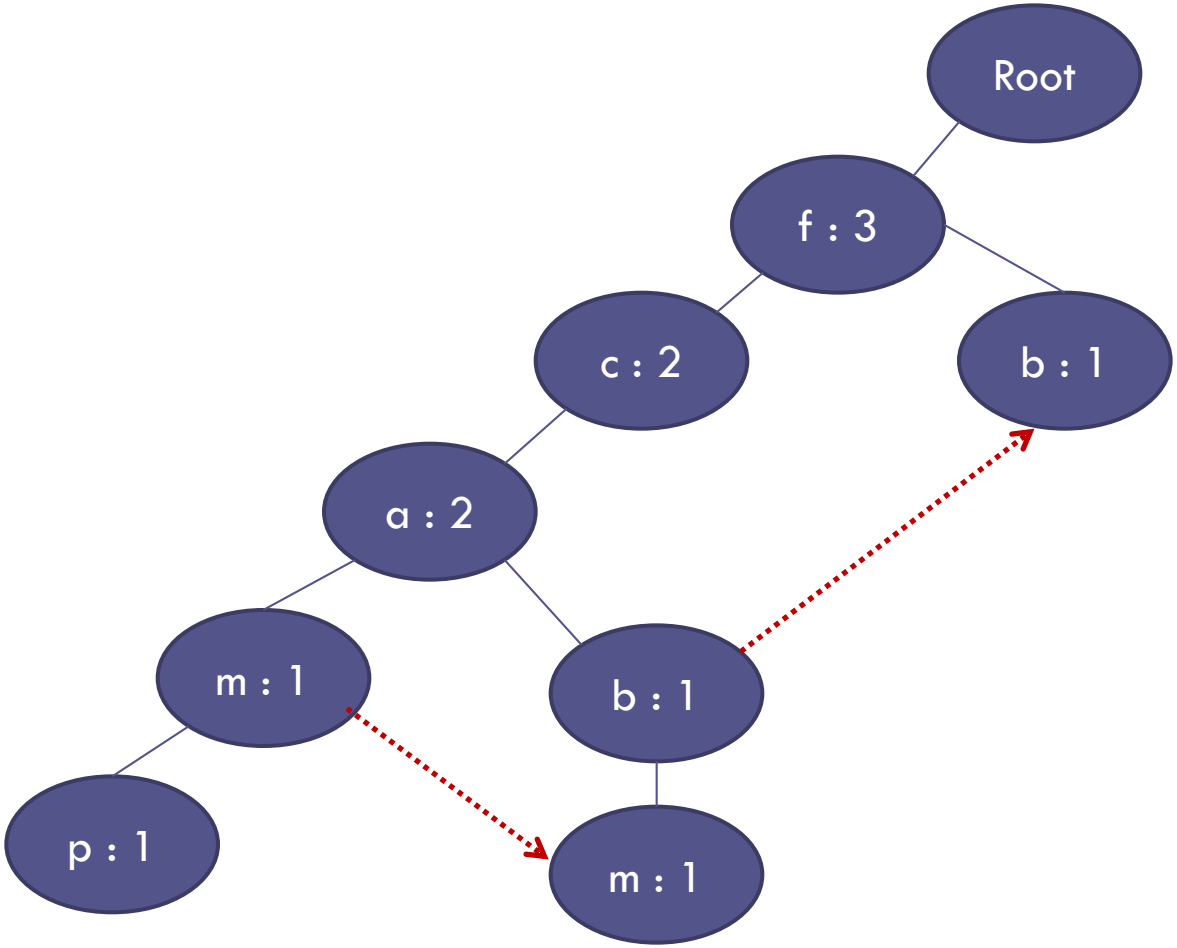

**T2:** f, c, a, b, m

- Consider the first item in T2 i.e. **f** and add it in the tree **(Algo.3.1)**
- Consider the Second item in T2 i.e. **c** and add it in the tree **(Algo.3.1)**
- Consider the third item in T2 i.e. **a** and add it in the tree **(Algo.3.1)**
- Since we do not have a **node b,** we create one node for b below the node a (to maintain the path) **(Algo.3.2)**
- Now only m of second transaction is left. Though a node m is already exists still we cannot increase its count of the existing node m as we need to represent the second transaction in FP tree, so add new node m below node b and link it with existing node m.

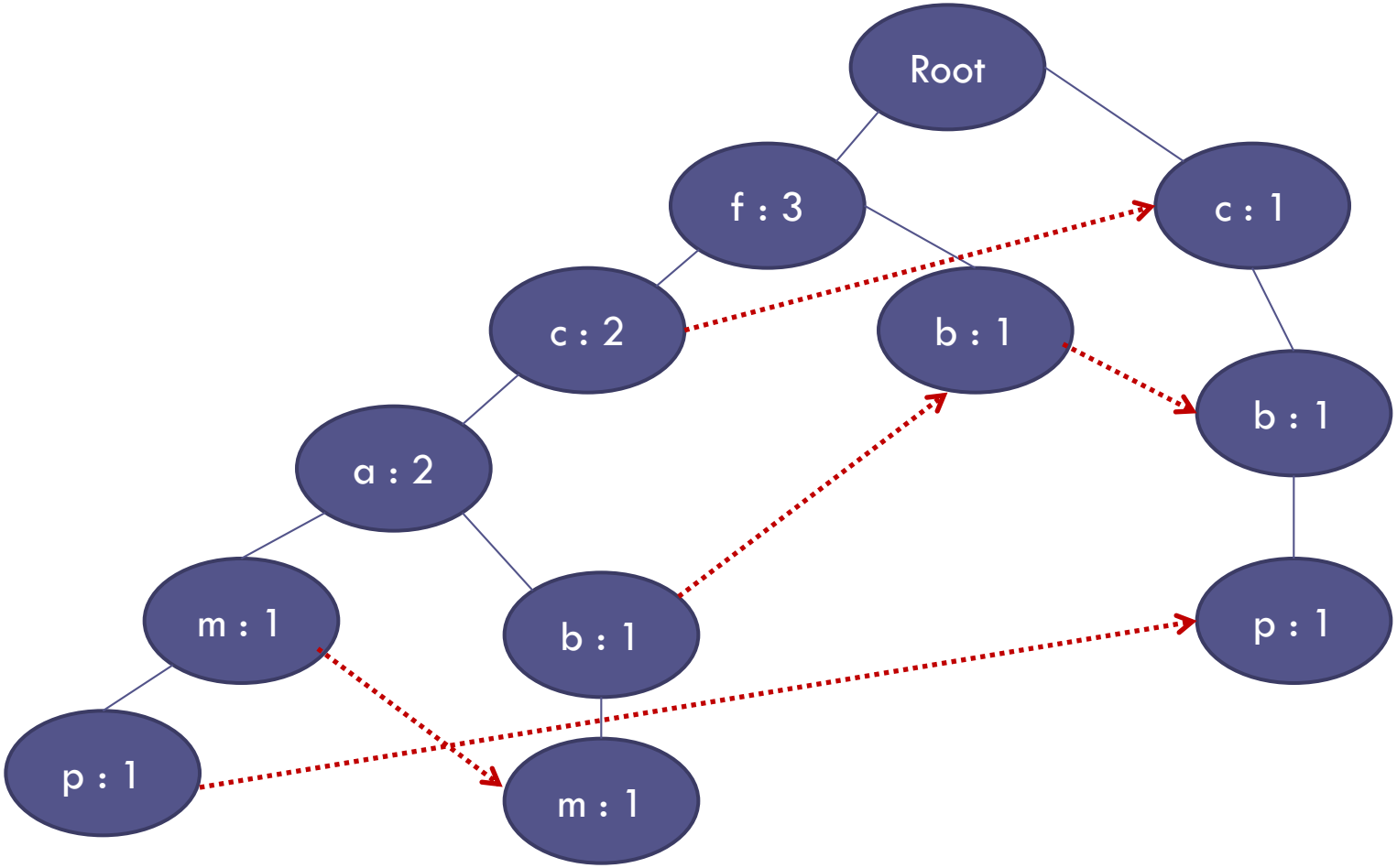
# After Inserting Transaction T2: f, c, a, b, m



# Transaction T3 : f, b

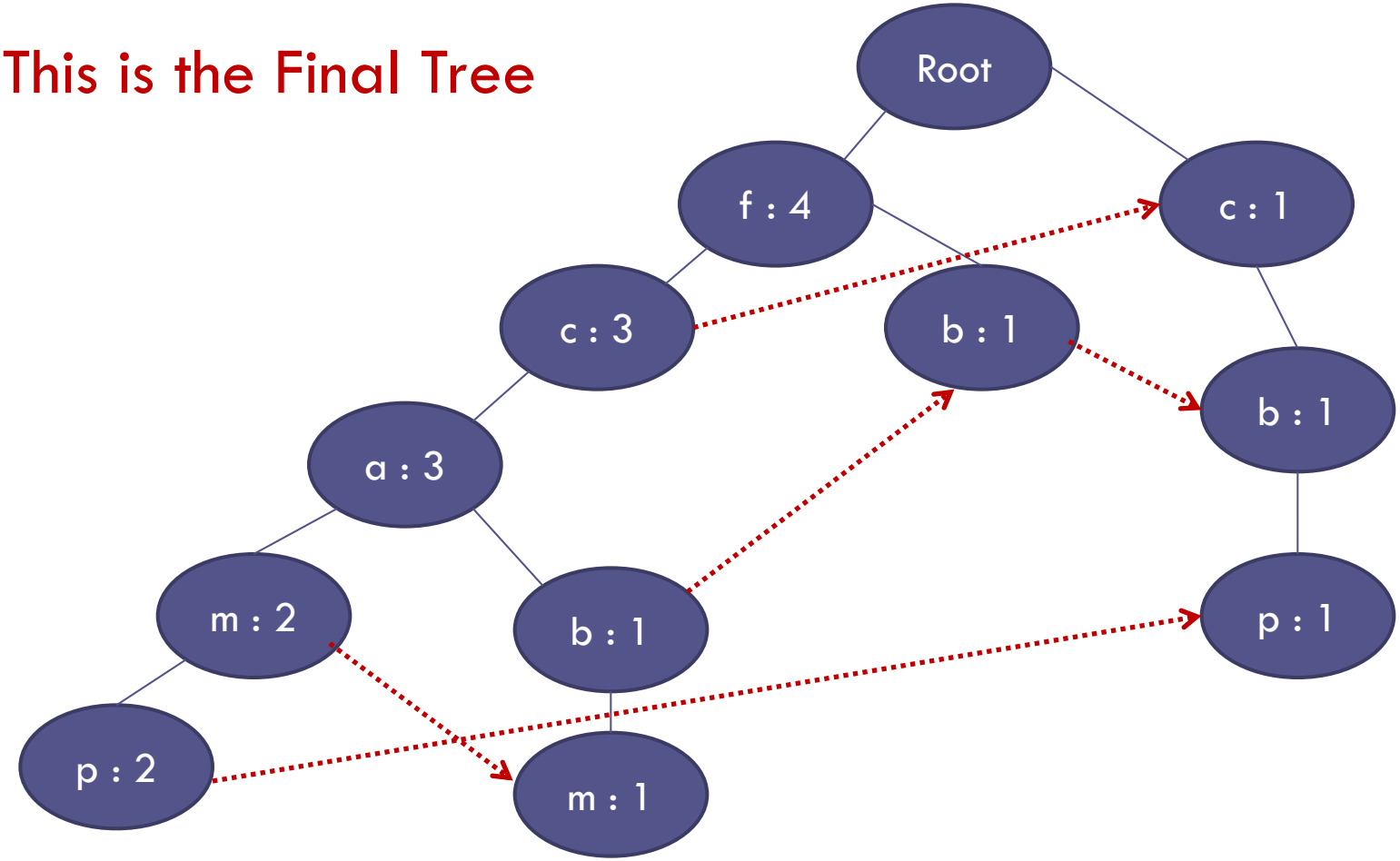


# Transaction T4: c, b, p

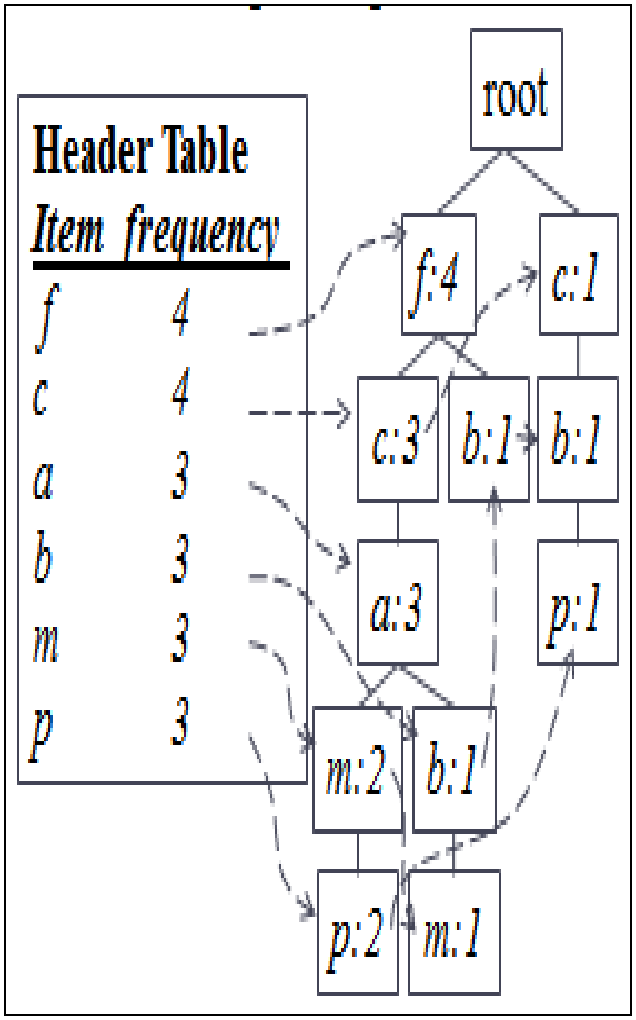


# Transaction T5: f, c, a, m, p

This is the Final Tree



# Mining the FP Tree to get Frequent Pattern




| Frequent items | Condition Pattern Base | Conditional FP Tree | Frequent Patterns Generated |
|---|---|---|---|
| p | fcam:2, cb: 1 | c:3 | Cp:3 (all FP concerning P) |
| m | fca: 2, fcab: 1 | f:3, c:3, a:3 | Fm:3, cm:3,am:3,fcam:3, fcm:3, cam:3, fam:3 |
| b | fca:1, f:1, c:1 | empty | - |
| a | fc:3 | f:3, c:3 | Fa:3,ac:3,Fca:3 |
| c | f:3 | f:3 | Fc:3 |
| f | empty | empty | - |

# Advantages of FP-Tree

- **Completeness:**
  - Never breaks a long pattern of any transaction
  - Preserves complete information for frequent pattern mining
- **Compactness:**
  - Reduce irrelevant information—infrequent items are gone
  - Frequency descending ordering: more frequent items are more likely to be shared
  - Never be larger than the original database (if not count node-links)

# **Example 2:** Finding all the patterns with 'p' in the FP Tree given below

| Transaction Id | Items Purchased |
|---|---|
| t1 | b, e |
| t2 | a, b, c, e |
| t3 | b, c, e |
| t4 | a, c, d |
| t5 | a |

Given: min support = 40% (at least count=2)

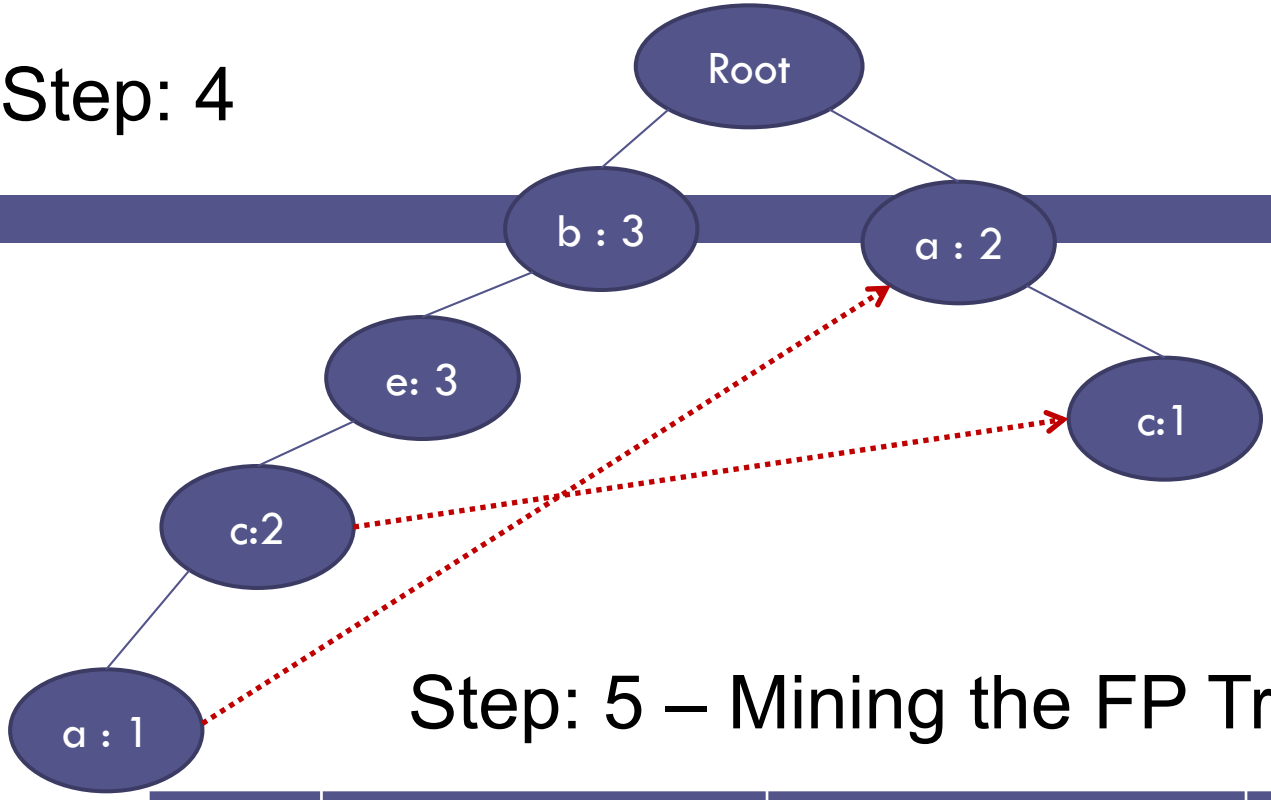
Step 1/ 2:

| Items | Support |
|---|---|
| a | 3 |
| b | 3 |
| c | 3 |
| d | 1 |
| e | 3 |

Step: 3

| Transactions | Items in descending order |
|---|---|
| t1 | b, e |
| t2 | b, e, c, a |
| t3 | b, e, c |
| t4 | a, c |
| t5 | a |

# Step: 4




## Step: 5 – Mining the FP Tree

| Item | Conditional pattern | FP Conditional tree | Frequent Patterns Generated |
|---|---|---|---|
| a | bec: 1 | empty | - |
| b | empty | empty | - |
| c | be: 2, a: 1 | b:2, e:2 | bc:2, ec:2, bce:2 |
| e | b:3 | b:3 | be:3 |

# **Example 3:** Finding all the patterns with 'p' in the FP Tree given below

| Transcation – ID | Items |
|---|---|
| 100 | 1, 2, 5 |
| 200 | 2, 4 |
| 300 | 2, 3 |
| 400 | 1, 2, 4 |
| 500 | 1, 3 |
| 600 | 1, 3 |
| 700 | 1, 3, 2, 5 |
| 800 | 1, 3 |
| 900 | 1, 2, 3 |

Given: min support = 40% (at least count=4)

# University Question on FP Algorithm

Frequent pattern mining algorithms considers only distinct items in a transaction. However, multiple occurrences of an item in the same shopping basket, such as four cakes and three jugs of milk, can be important in transactional data analysis. How can one mine frequent itemsets efficiently considering multiple occurrences of items? Generate Frequent Pattern Tree for the following transaction with 30% minimum support: **[10]**

| Transaction ID | Items |
|---|---|
| T1 | E, A, D, B |
| T2 | D, A, C, E, B |
| T3 | C, A, B, E |
| T4 | B, A, D |
| T5 | D |
| T6 | D, B |
| T7 | A, D, E |
| T8 | B, C |

May 2019

# Thank You