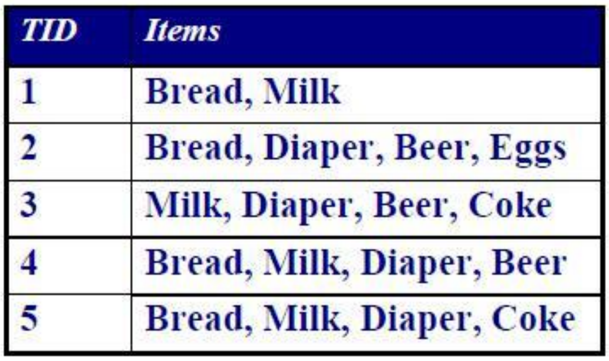
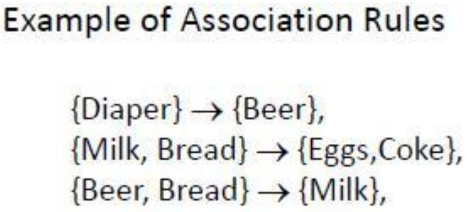
Unit 4: Association Analysis

Basic: - Many business enterprises accumulate large quantities of data from their day to day operations. For example, huge amounts of customer purchase data are collected daily at the checkout counters of grocery stores. Following table illustrates an example of such data, commonly known as **market basket transactions**. Each row in the table corresponds to a transaction, which contains a unique identifier labeled TID and set of items bought by a given customer. Retailers are interested in analyzing the data to learn about the purchasing behavior of their customers. Such valuable information can be used to support a variety of business related applications such as marketing promotions, inventory management and customer relationship management.



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},

Table: An example of market basket transactions.

Mining for associations among items in a large database of transactions is an important data mining function.

- Association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases.

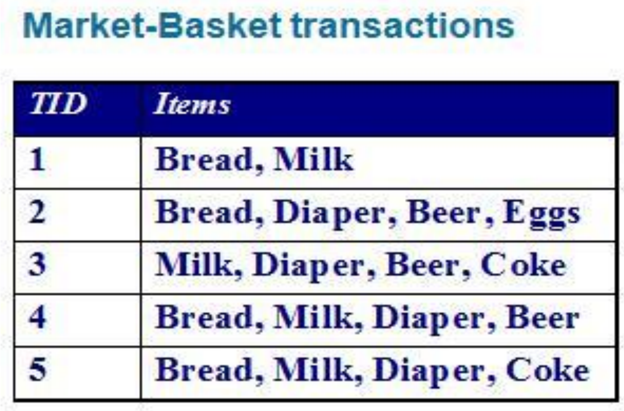
- Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction

- **Association rules are statements of the form {X1, X2, …, Xn} => Y, meaning that if we find all of X1, X2, ……… , Xn in the transaction then we have good chance of finding Y.**

Eg. The information that a customer who buys computer also tends to buy antivirus or pen drive.

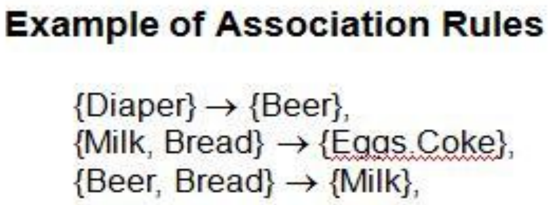
- Association analysis mostly applied in the field of market basket analysis, web-based mining, intruder detection etc.

Market Basket Analysis



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},

- Market basket analysis (also known as Affinity Analysis) is the study of items that are purchased or grouped together in a single transaction or multiple, sequential transactions.

- Understanding the relationships and the strength of those relationships is valuable information that can be used to make recommendations, cross-sell, up-sell, offer coupons, etc.

- A predictive market basket analysis can be used to identify sets of products/services purchased/events) that generally occur in sequence or something of interest to direct marketers.

- Advanced Market Basket Analysis provides an excellent way to get to know the customer and understand the different behaviors that can be leveraged to provide better assortment, design a better plan and devise more attractive promotions that can lead to more sales and profits.

- The analysis can be applied in various ways:
    - Develop combo offers based on products sold together.
    - Organize and place associated products/categories nearby inside a store.
    - Determine the layout of the catalog of an ecommerce site.
    - Control inventory based on product demands and what products sell together.
    - Support of a product or group of products indicates the popularity of the product or group of products in the transaction set. Higher the support, more popular is the product or product bundle. This measure can help in identifying selling strategy of the store. Eg: if Barbie dolls have a higher support then they can be attractively priced to attract traffic to a store.
    - Confidence can be used for product placement strategy and increasing profitability. Place high-margin items with associated high selling items. If Market Basket Analysis

indicates that customers who bought high selling Barbie dolls also bought high-margin candies, then candies should be placed near Barbie dolls.

- In order to gain better insights, Market Basket Analysis can based on

  □ Weekend vs weekday sales

  □ Month beginning vs month-end sales

  □ Different seasons of the year

  □ Different stores

  □ Different customer profiles

- Although Market Basket Analysis mostly applied for shopping carts and supermarket shoppers, there are many other areas in which it can be applied such as:

Few terminologies used in association analysis

Itemset:

– A collection of one or more items. Example: {Milk, Bread, Diaper}

– An itemset that contains k items is called k-itemset.

Support count:

- Frequency of occurrence of an itemset
- E.g. ({Milk, Bread, Diaper}) = 2

Support: □

- Fraction of transactions that contain an itemset
- E.g. s ({Milk, Bread, Diaper}) = 2/5

Support (A): Number of tuples containing A / Total number of tuples

Support (A = > B): Number of tuples containing A and B / Total number of tuples

- If minsup is set too high, we could miss itemsets involving interesting rare items (e.g., expensive products)

- If minsup is set too low, it is computationally expensive and the number of itemsets is very large

Frequent Itemset :

- □An itemset whose support is greater than or equal to a minsup threshold.

Association Rule:

– An implication expression of the form X -> Y, where X and Y are itemsets

Example: {Milk, Diaper} -> {Beer}

Rule Evaluation Metrics

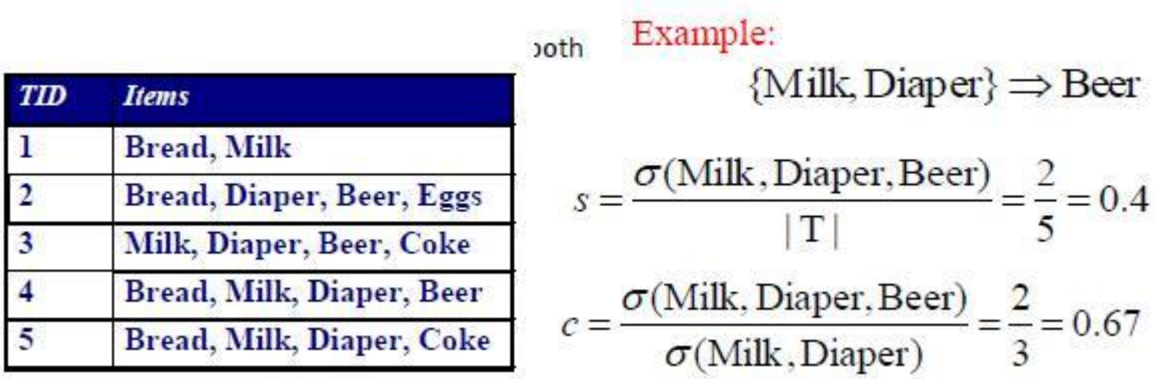
– Support (s)

- Fraction of transactions that contain both X and Y

– **Confidence (c)** :

- Measures how often items in Y appear in transactions that contain X

Confidence (A = > B): Number of tuples containing A and B / Total count of A



ɔoth

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:

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

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

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

Why Use Support and Confidence?

Support is an important measure because a rule that has very low support may occur simply by chance. A low support rule is also likely to be uninteresting from business perspective because it may not be profitable to promote items that customers seldom buy together. For this reasons, support is often used to eliminate uninteresting rules.

Confidence on the other hand, measures the reliability of the inference made by a rule. For a given rule X -> Y, the higher the confidence, the more likely it is for Y to be present in transactions that contain X.

Association Rules Mining

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

Some of approaches for association rules mining are:

Brute- Force Approach

- List all possible association rules.
- Compute the support and confidence for each rule.
- Prune rules that fail to minimum support and minimum confidence level. *This approach is computationally very expensive.

Mining Association Rules

Many association rule mining algorithms is to decompose the problem into two major subtasks:

1. Frequent ItemSet Generation, Whose objective is to find all the itemsets that satisfy the minsup threshold. These items are called frequent itemsets.

2. Rule Generation, whose objectives is to extract all the high-confidence rules from the frequent itemsets found in the previous step. These rules are called strong rules.

Frequent itemset generation is still computationally expensive

Frequent itemset generation

A lattice structure can be used to enumerate the list of all possible itemsets. Figure below shows an itemset lattice for I={a,b,c,d,e}. In general, a data set that contains k items can potentially generate up to 2^k-1 frequent itemsets, excluding the null sets. Because k can be very large in many practical applications, the search space of itemsets that need to be explored is exponentially large.

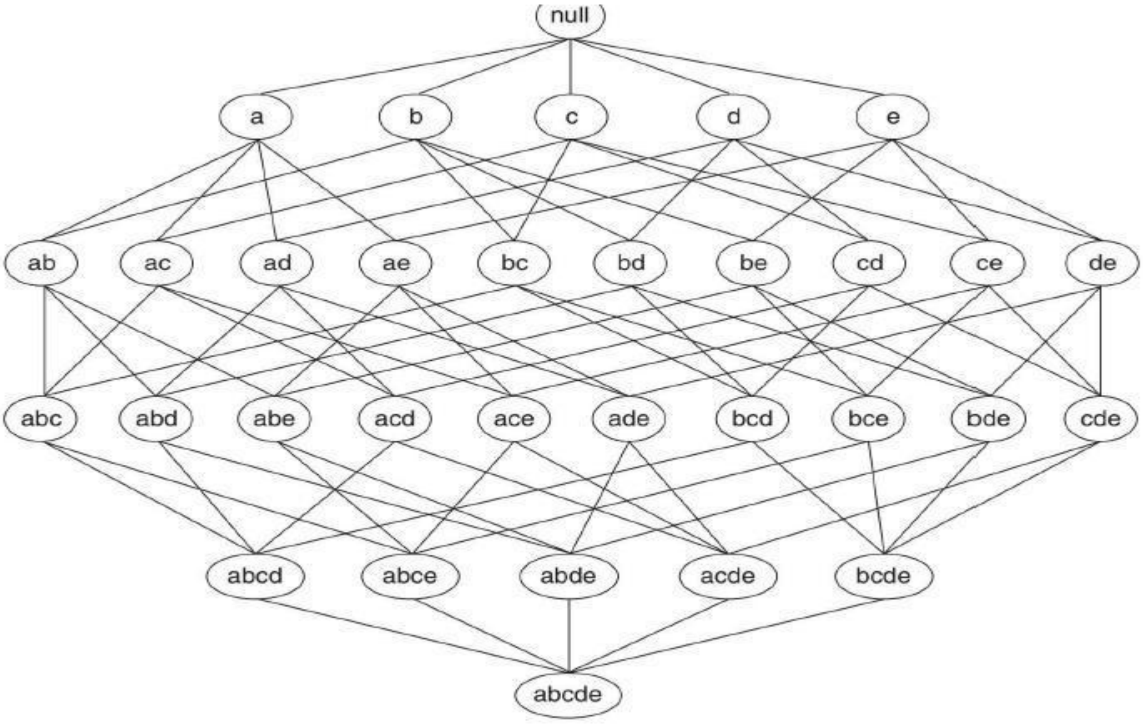


Fig: An Itemset lattice

There are several ways to reduce the computational complexity of frequent itemset generation.

1. Reduce the number of candidate itemsets (M).

2. Reduce the number of comparisons.

The Apriori Principle

“IF an itemset is frequent, then all of its subsets must also be frequent.”

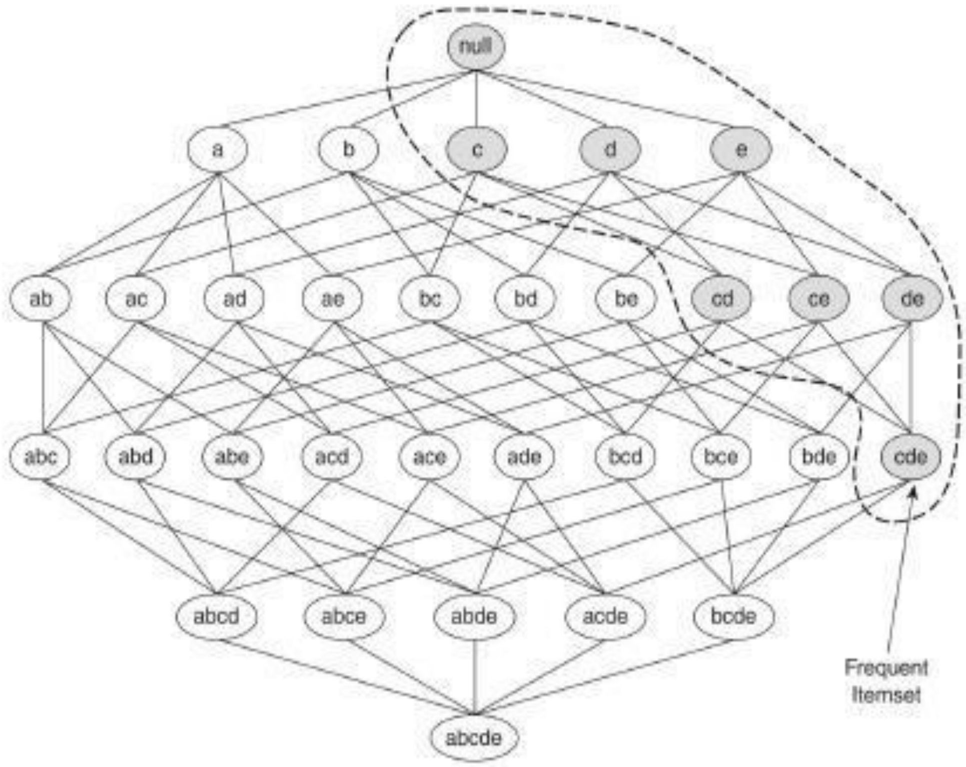


Fig: An illustration of the Apriori principle. If {c,d,e} is frequent, then all subsets of this itemsets are frequent.

Conversely, if an itemset such as {a,b} is infrequent, then all of its supersets must be infrequent too. An illustrated in figure below, the entire subgraph containing the supersets of {a,b} can be pruned immediately once {a,b} is found to be infrequent. This strategy of trimming the exponential search space based on the support measure is known as support based pruning. Such a pruning strategy is made possible by a key property of the support measure, namely, that the support for an itemset never exceeds the support of its subsets. This property is also known as the **anti-monotone property** of the support measure.

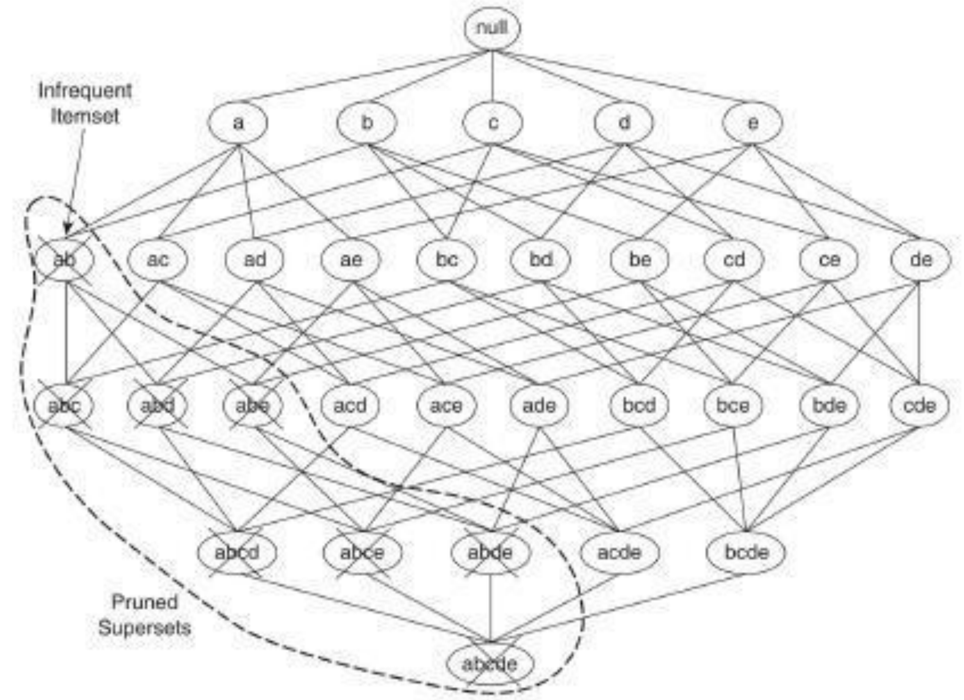


Fig: An illustration of support based pruning. If {a,b} is infrequent then all supersets of {a,b} are infrequent.

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

Which means that if X is subset of Y, then support of x is greater than or equal to support of y (Support of y never exceed then support of x)

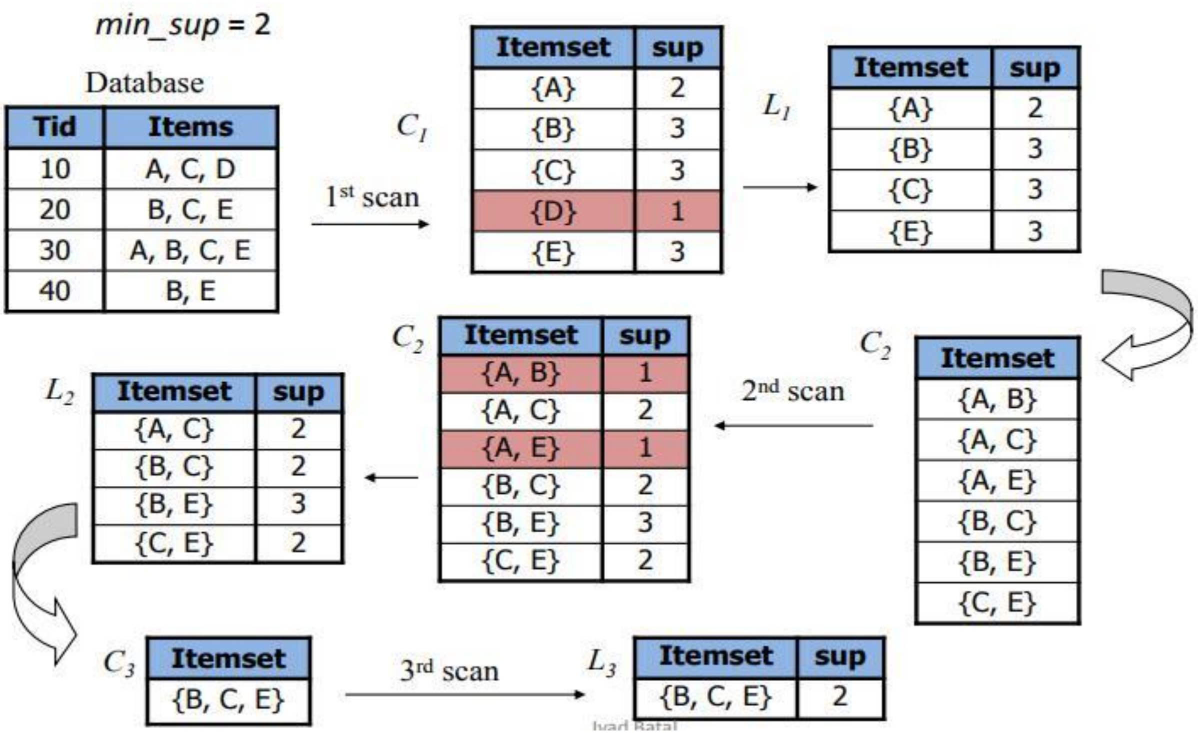
Apriori Algorithm

•Let k=1

•Generate frequent itemsets of length 1

•Repeat until no new frequent itemsets are identified

- Generate length (k+1) candidate itemsets from length k frequent itemsets
- Prune candidate itemsets containing subsets of length k that are infrequent
- Count the support of each candidate by scanning the DB
- Eliminate (prune) candidates that are infrequent, leaving only those that are frequent.



Database

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

C_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

L_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

C_2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

C_2

Itemset	sup
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

L_2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

C_3

Itemset
{B, C, E}

L_3

Itemset	sup
{B, C, E}	2

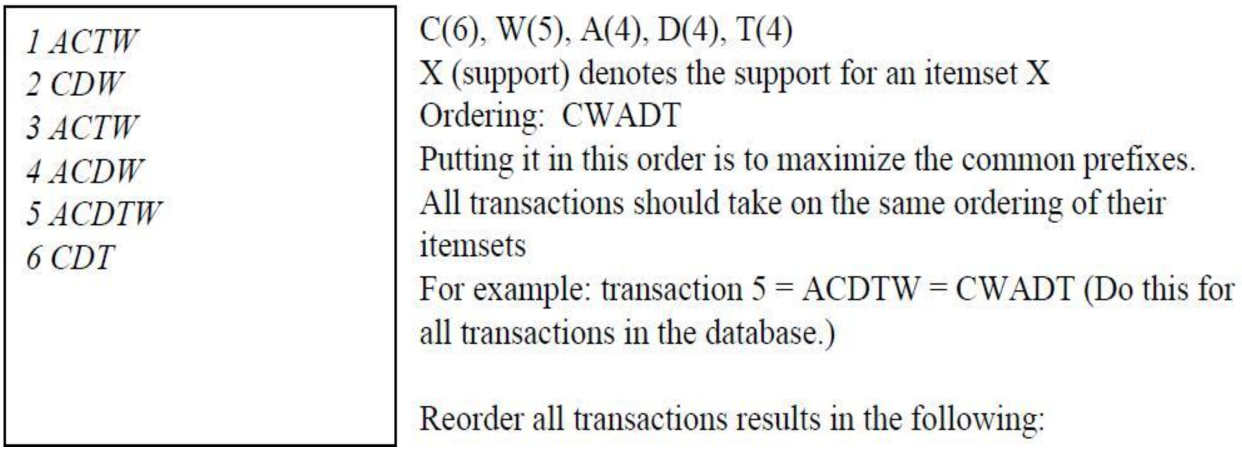
Note : Q.Why {1 2 3}, {1 2 5}, {1 3 5} are not listed in C3???

FP-Growth

FP-Growth is an improvement of apriori designed to eliminate some of the heavy bottlenecks in apriori. FP-Growth simplifies all the problems present in apriori by using a structure called an FP-Tree. In an FP-Tree each node represents an item and it's current count, and each branch represents a different association.

FP Tree (Frequent Pattern tree)

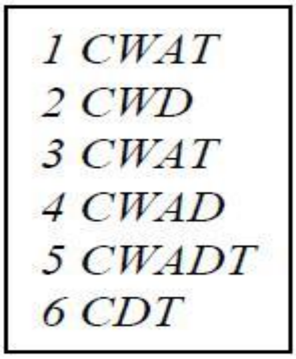
Using the database below, we will calculate the support for all single items. After calculating the support for each item in the database, we sort these items by support in decreasing order.



1 ACTW
2 CDW
3 ACTW
4 ACDW
5 ACDTW
6 CDT

C(6), W(5), A(4), D(4), T(4)
X (support) denotes the support for an itemset X
Ordering: CWADT
Putting it in this order is to maximize the common prefixes.
All transactions should take on the same ordering of their itemsets
For example: transaction 5 = ACDTW = CWADT (Do this for all transactions in the database.)

Reorder all transactions results in the following:

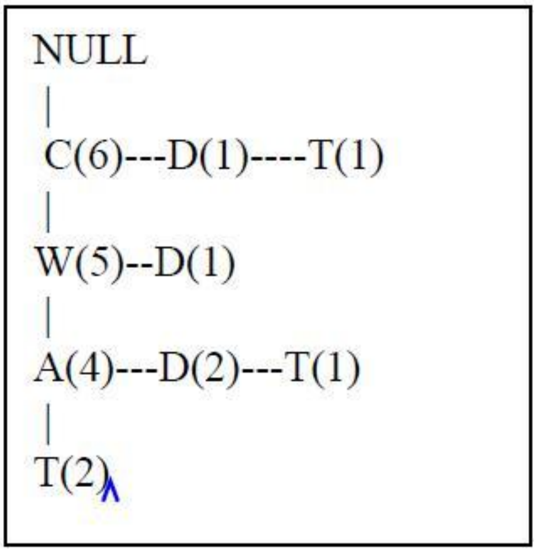


1 CWAT
2 CWD
3 CWAT
4 CWAD
5 CWADT
6 CDT

Assumption: minsup = 3/6

An FP –tree is a projection based approach

The FP-tree is constructed by using the new ordering of each transaction. We use the common prefixes to construct the tree and add branches wherever needed. Our result is the tree below:



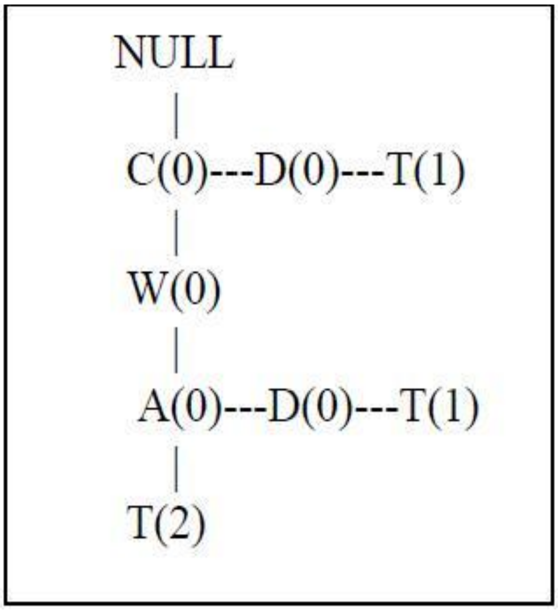
```
NULL
 |
 C(6)---D(1)----T(1)
 |
W(5)--D(1)
 |
A(4)---D(2)---T(1)
 |
T(2)
```

Now, we have to consider all items from T to C, in the reverse order of support, from smallest support to largest support. We have to generate all subsets that end in the particular item that we are processing before moving to the next item. All occurrences of each item are kept in a linked list.

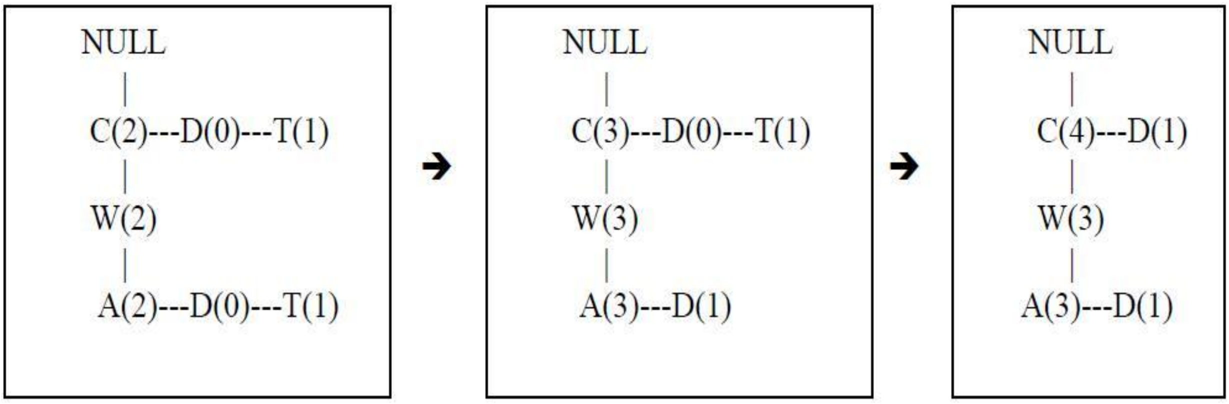
Example: a linked list for T that keeps track of T’s occurrence: T: T(2) — T(1)—T(1) . The same thing for D’s, D: D(2) — D(1) — D(1), and other items.

Conditional FP-tree for T

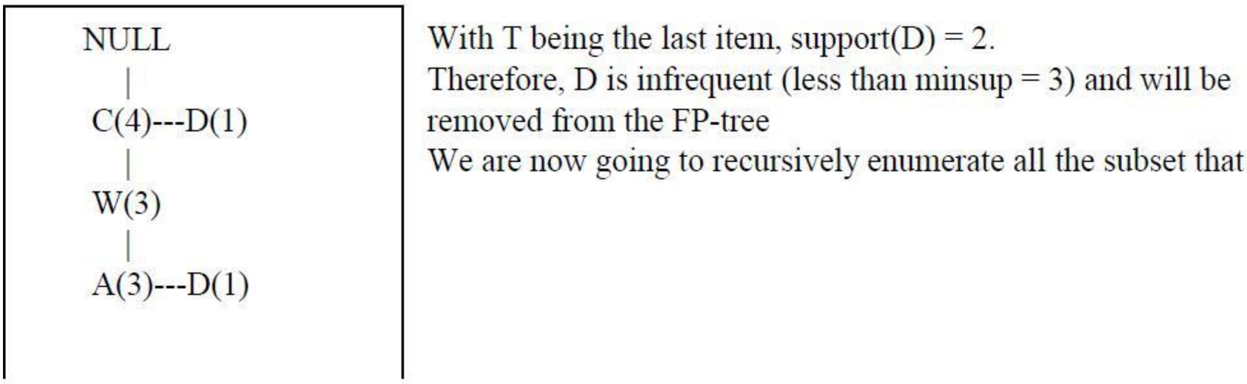
We are going to extract all paths that end with T. First, we count for T’s occurrence. To do this, we can trace the linked list and add up all of its support. T has a support of 4 because 4 is the sum of T’s linked list (T: T(2)—T(1)—T(1)). T is frequent because its support > 3. We start with drawing tree whose end-nodes are Ts, keeping only the support of Ts.



Then, we take out T one by one, and as we do so, we push its support to every node up the chain to the root that was a part of the same transaction in which T was.

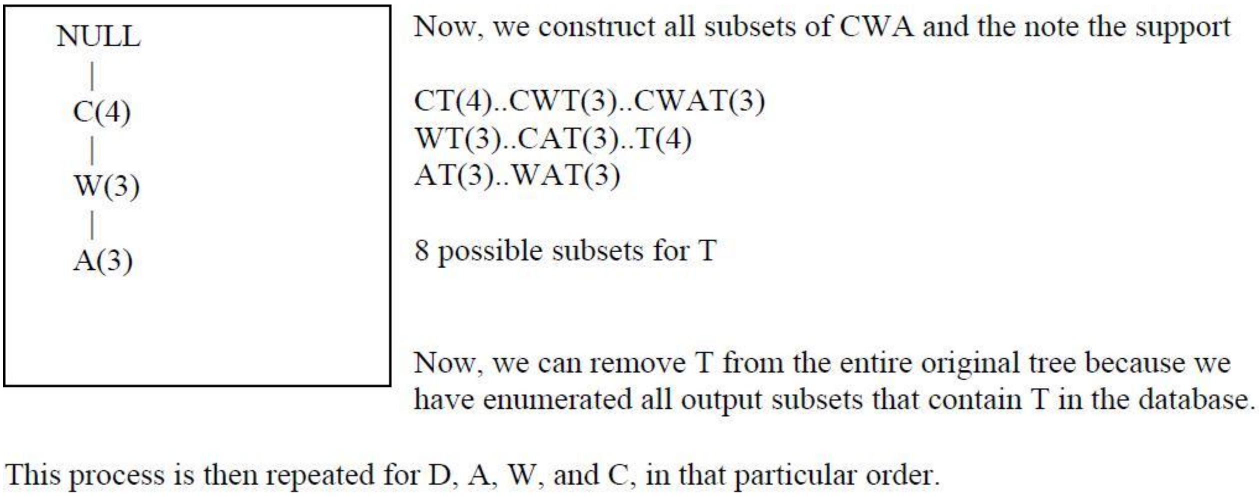


For example: C is part of 4 transactions in which T was. W is a part of 3 transactions in which was and so on.



With T being the last item, support(D) = 2.
Therefore, D is infrequent (less than minsup = 3) and will be removed from the FP-tree
We are now going to recursively enumerate all the subset that ends up with item T, so we need to apply the same principle of FP-tree. We consider each item that appears in the tree and create new FP-trees for C, W, A, and D (note we are now within the context of the new FP-Tree for T).

The below tree is obtained by removing D, which is infrequent. Since there is only one path, now we can enumerate all subsets of that path and add T at the end of each subset because T is the particular item that we are enumerating subsets for. The support of each subset is then set to be the supported of the last item before T. For example, CWT has a support of three because W (the last item before T in CWT) only has a support of three (from the subtree below, i.e. W(3)), even though C and T have a support of four.



Now, we construct all subsets of CWA and the note the support

CT(4)..CWT(3)..CWAT(3)
WT(3)..CAT(3)..T(4)
AT(3)..WAT(3)

8 possible subsets for T

Now, we can remove T from the entire original tree because we have enumerated all output subsets that contain T in the database.

This process is then repeated for D, A, W, and C, in that particular order.

Categorical data

- Categorical data is a statistical data type consisting of categorical variables, used for observed data whose value is one of a fixed number of nominal categories.

- More specifically, categorical data may derive from either or both of observations made of qualitative data, where the observations are summarized as counts or cross tabulations, or of quantitative data.

- Observations might be directly observed counts of events happening or they might counts of values that occur within given intervals.

- Often, purely categorical data are summarized in the form of a contingency table.

- However, particularly when considering data analysis, it is common to use the term "categorical data" to apply to data sets that, while containing some categorical variables, may also contain non-categorical variables.

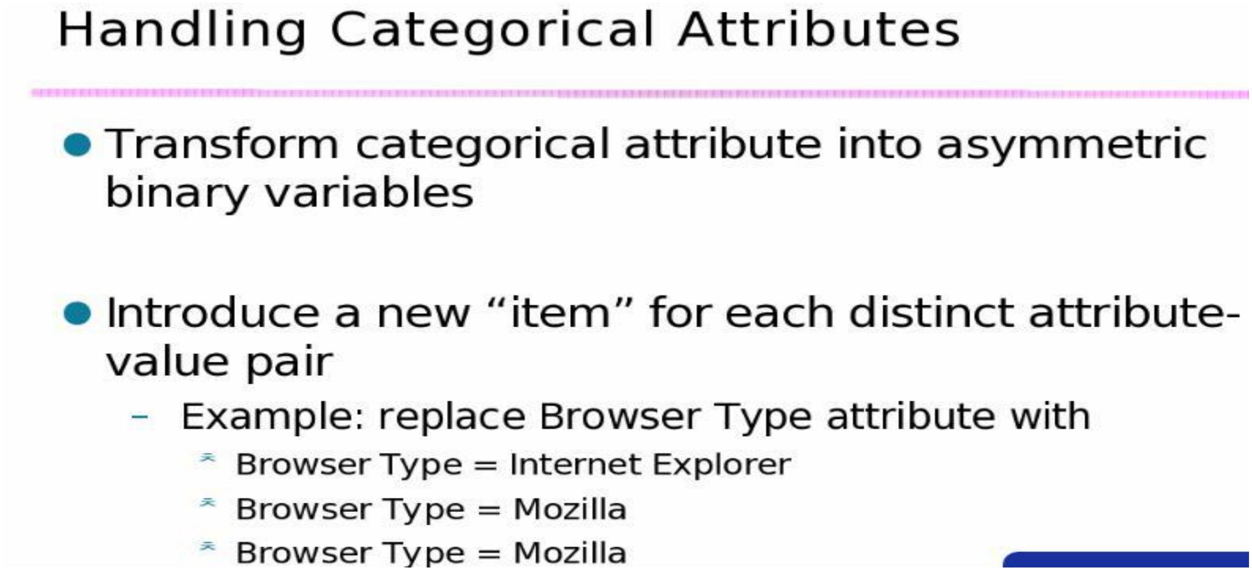
Potential Issues

- What if attribute has many possible values: Example: attribute country has more than 200 possible values. Many of the attribute values may have very low support.

Potential solution: Aggregate the low-support attributes values.

- What if distribution of attribute values is highly skewed: Example: 95% of the visitors have Buy = No. Most of the items will be associated with (Buy=No) item

Potential solution: drop the highly frequent items.



Handling Categorical Attributes

- Transform categorical attribute into asymmetric binary variables

- Introduce a new "item" for each distinct attribute-value pair
  - Example: replace Browser Type attribute with
    - Browser Type = Internet Explorer
    - Browser Type = Mozilla
    - Browser Type = Mozilla