Association analysis

Naive algorithm

Apriori algorithm

Multilevel association rules discovery

Naive algorithm (1)

- Given a set of items I and a database D
- Generate all possible subsets of the set I and, then, for each subset (candidate itemset) calculate support of this itemset in the database D
- For each itemset, those support is greater/equal minsup, generate an association rule – for each generated rule calculate its confidence
- The number of all possible subsets of a set I is:

  $2^{|I|} - 1$ (size of I ≈ 200 000 items)
- The number of all possible binary association rules for a set of items I is: $3^{|I|} - 2^{|I|+1} + 1$

Naive algorithm (2)

- Consider the dataset D from the previous example:
  - A set of items I = 4
  - The number of all possible binary association rules for a set of items I is: $3^{|I|} - 2^{|I|+1} + 1 = 50$
  - The number of strong binary association rules for I is 14, i.e. 28% of all possible binary association rules that can be generated for the set I
  - Application of the naive algorithm leads to the waste of time that we have to spend calculating support and confidence measures of rejected rules

Naive algorithm (3)

- How can we restrict a number of generated association rules to avoid the necessity of calculating support and confidence of rejected rules?
- Answer: it is necessary to consider separately minimum support and minimum confidence thresholds while generating association rules
- Notice that the support of a rule $X \rightarrow Y$ is equal to the support of the set (X, Y)

Naive algorithm (4)

- If the support of the set (X, Y) is less than *minsup*, then we may skip the calculation of the confidence of rules $X \rightarrow Y$

  and $Y \rightarrow X$
- If the support of the set (X, Y, Z) is less than *minsup* then we may skip the calculation of the confidence of rules:

  $X \rightarrow Y, Z$ $Y \rightarrow X, Z$ $Z \rightarrow X, Y$

  $X, Y \rightarrow Z$ $X, Z \rightarrow Y$ $Y, Z \rightarrow X$
- In general, if the support of a set (X1, X2, …, Xk) is less than *minsup*, sup(X1, X2, …, Xk) < *minsup*, we may skip the calculation of the confidence of 2^k - 2 association rules

General algorithm of association rule discovery

Algorithm 1.1: General algorithm of association rule discovery

- Find all sets of items $L_i=\{I_{i1}, I_{i2}, \ldots, I_{im}\}$, $L_i \subseteq I$, that have ***sup***$(L_i) \geq$ *minsup*. Sets L_i are called ***frequent itemsets***.
- Use the frequent itemsets to generate the association rules using the algorithm 1.2.

Rule Generation Algorithm

Algorithm 1.2: Rule generation.

```
for each frequent itemset $L_i$ do
    for each subset $subL_i$ of $L_i$ do
        if $support(L_i)/support(subL_i) \geq minconf$ then
            output the rule $subL_i \Rightarrow (L_i - subL_i)$
            with confidence = $support(L_i)/support(subL_i) \geq$
    and support = $support(L_i)$
```

Algorithm 1.3: Apriori

Notation:

- Assume that all transactions are internally ordered
- L_k denotes a set of frequent itemsets of size k (those with minimum support) – frequent k-itemsets
- C_k denotes a set of candidate itemsets of size k (potentially frequent itemsets) – candidate k- itemsets

Algorithm 1.3: Apriori

Algoritm 1.3: Frequent itemsets discovery algorithm (Apriori)

In: dataset *D*, minimum support threshold minsup

Out: all frequent itemsets contained in D

```
1.     L1 $\leftarrow$ frequent 1-itemsets;
2.     for ($k = 2$; $L_{k-1} \neq \varnothing$; $k$++) do
3.         Ck $\leftarrow$ apriori_gen($L_{k-1}$);
4.         for all transakcji $t \in D$ do
5.             $C_t \leftarrow$ subset($C_k$, t);
6.             for all candidate itemsets $c \in C_t$ do
7.                 c.count++;
8.             end for
9.         end for
10.        Lk $\leftarrow \{c \in C_k \mid c{:}count \geq minsup\}$;
11.    end for
12.    return $\bigcup_k L_k$
```

Function: Apriori_Gen(C_k) (1)

Algoritm 1.3: Frequent itemsets discovery algorithm (Apriori)

In: frequent (k-1)-itemsets L_{k-1}

Out: candidate k-itemsets C_k

```
1. insert into $C_k$
2. select p.$item_1$, p.$item_2$, ..., p.$item_{k-1}$, q.$item_{k-1}$
3. from $L_{k-1}$ p, $L_{k-1}$ q
4. where p.$item_1$ = q.$item_1$
5.     and p.$item_2$ = q.$item_2$
6.     ...
7.     p.$item_{k-2}$ = q.$item_{k-2}$,
8.     p.$item_{k-1}$ < q.$item_{k-1}$;
```

Function: Apriori_Gen(C_k) (2)

Algorytm 1.3: Frequent itemsets discovery algorithm (Apriori)

In: frequent (k-1)-itemsets L_{k-1}

Out: candidate k-itemsets C_k

```
9. forall candidate itemsets $c \in C_k$ do
10.     forall (k-1)-subsets s of c do
11.      if ( $s \notin L_{k-1}$ ) then
12.       delete c from $C_k$;
13.         end if
14.     end for
15. end for
```

Example 2

- Assume that minsup = 50% (2 transactions)

TID	Items
100	134
200	235
300	1235
400	25

C1

Itemset	Sup
1	2
2	3
3	3
4	1
5	3

$\longrightarrow$

L1

Itemset	Sup
1	2
2	3
3	3
5	3

Example 2 (cont.)

C2

itemset	Sup
12	1
13	2
15	1
23	2
25	3
35	2

$\rightarrow$

L2

itemset	Sup
13	2
23	2
25	3
35	2

C3

itemset	Sup
235	2

$\rightarrow$

L3

itemset	Sup
235	2

$C4 = \varnothing \longrightarrow L4 = \varnothing$

Apriori Candidate Generation (1)

- Given Lk, generate Ck+1 in two steps:

L2

itemset	sup
13	2
23	2
25	3
35	2

1. Join step: Join Lk^1 with Lk^2, with the join condition that the first k-1 items should be the same and $Lk^1[k] < Lk^2[k]$
2. Prune step: delete all candidates, which have a non-frequent subset

Apriori Candidate Generation (2)

- Given L_2

L_2

set	sup
1 3	2
2 3	2
2 5	3
3 5	2
3 7	2

join

C_3 – after join

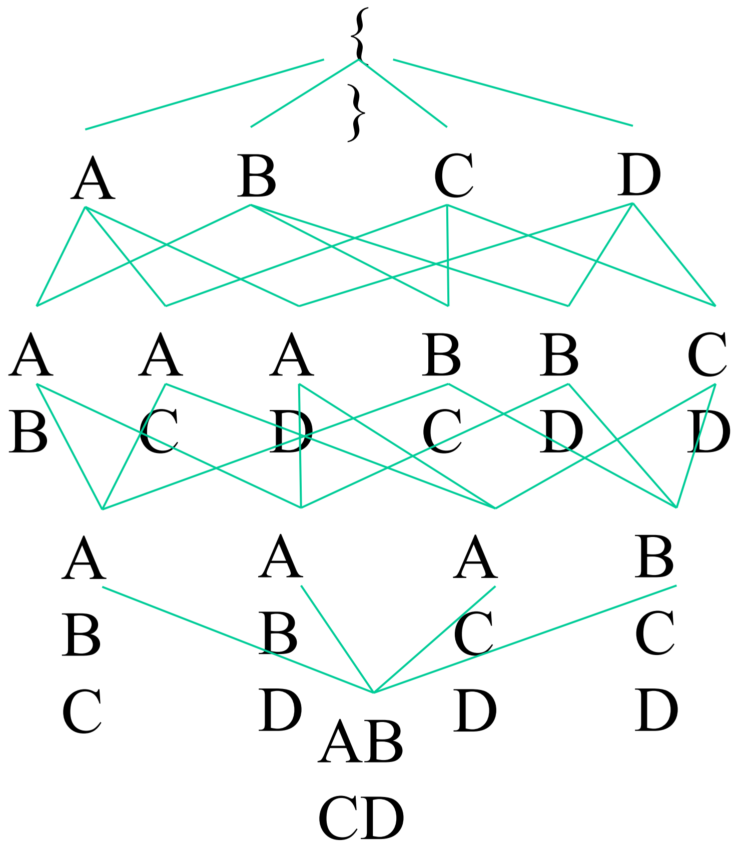
set	sup
2 3 5	
3 5 7	

prune

C_3 – final form

set	sup
2 3 5	

Discovery of frequent itemsets



The lattice of subsets of the set I represents the space of solutions (search space)

The aim of each algorithm of frequent itemsets discovery is to restrict the number of analyzed itemsets of the lattice

Properties of measures

Monotonicity property

Let I be a set of items, and $J = 2^{|I|}$ be the power set of I. A measure f is monotone on the set J if:

$$\forall X; Y \in J : (X \subseteq Y) \rightarrow f(X) \leq f(Y)$$

Monotone property of the measure *f* means that if X is a subset of *Y*, then f (X) must not exceed f (Y)

Anti-monotonicity property

Let I be a set of items, and $J = 2^{|I|}$ be the power set of I. A measure f is anti-monotone on the set J if:

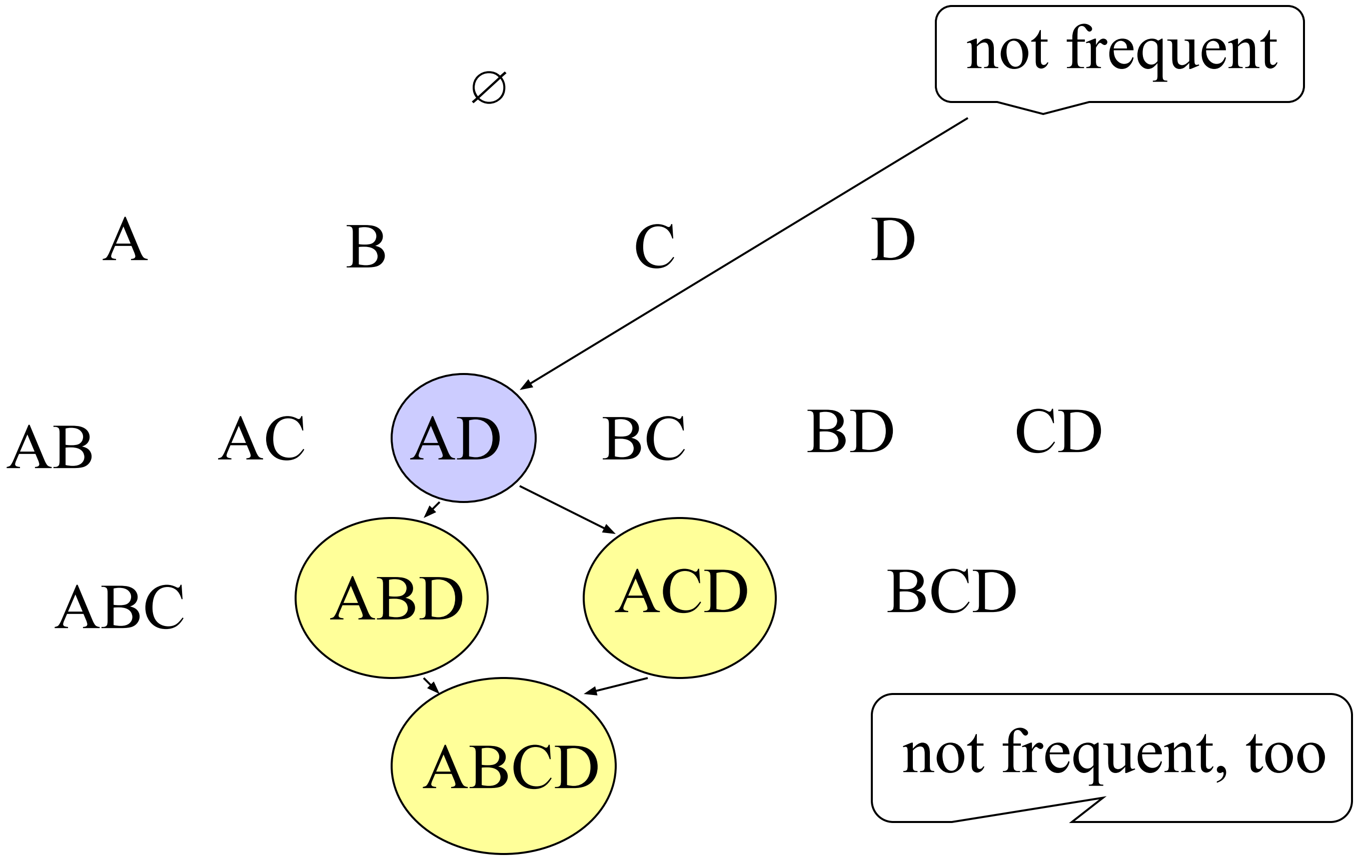
$$\forall X; Y \in J : (X \subseteq Y) \rightarrow f(Y) \leq f(X)$$

Anti-monotone property of the measure *f* means that if X is a subset of *Y*, then f (Y) must not exceed f (X)

Why does it work? (1)

- Anti-monotone property of the support measure: all subsets of a frequent itemset are frequent, in other words, if B is frequent and $A \subseteq B$, then A is also frequent
- Consequence: if A is not frequent, then it is not necessary to generate the itemsets which include A

Apriori Property



Why does it work? (2)

- The join step is equivalent to extending each itemset in Lk with every item in the database and then deleting those itemsets Ck+1 whose subset (Ck+1 –C[k]) is not frequent.

Discovering Rules

L3

itemset	sup
235	2

$23 \rightarrow 5$	support = 2	confidence = 100%
$25 \rightarrow 3$	support = 2	confidence = 66%
$35 \rightarrow 2$	support = 2	confidence = 100%
$2 \rightarrow 35$	support = 2	confidence = 66%
$3 \rightarrow 25$	support = 2	confidence = 66%
$5 \rightarrow 23$	support = 2	confidence = 66%

Rule generation (1)

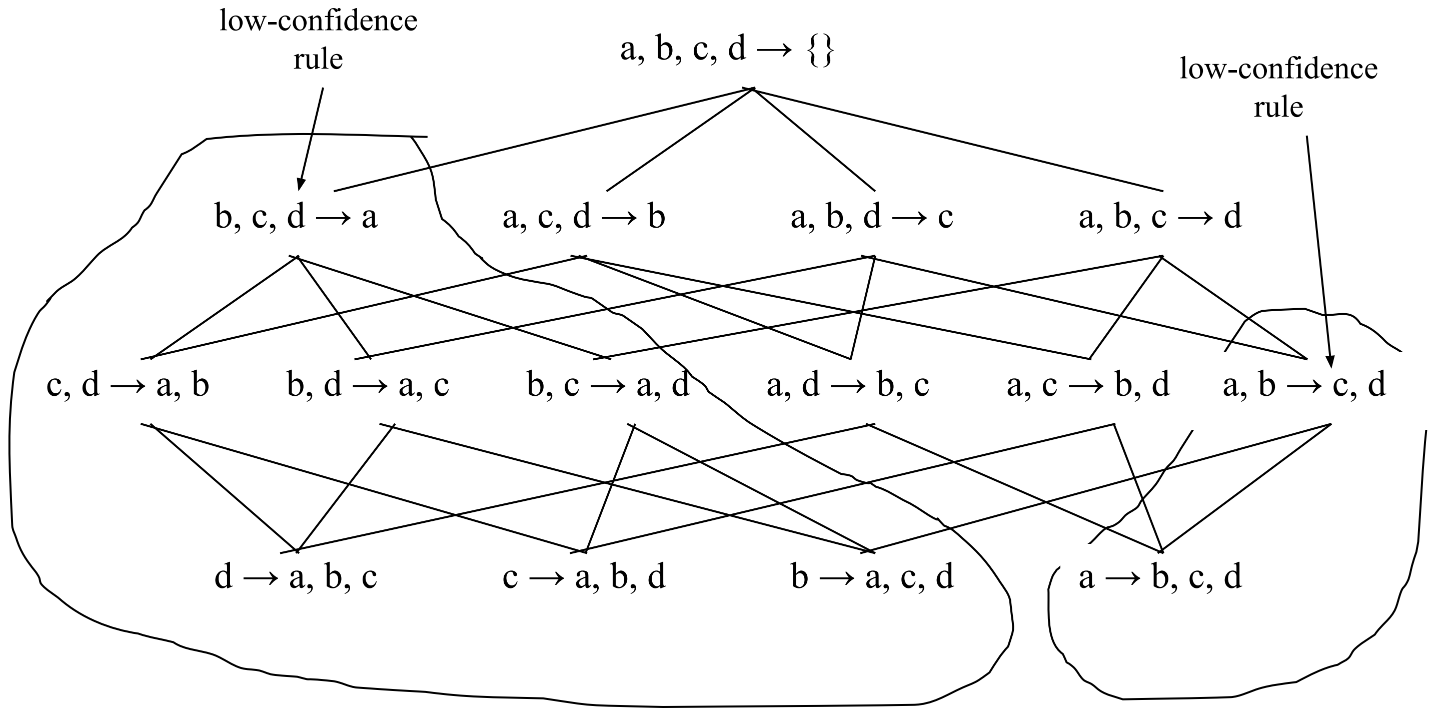
- For each k-itemset X we can produce up to $2^k - 2$ association rules
- Confidence does not have any monotone property
- conf($X \rightarrow Y$) ??? conf($X' \rightarrow Y'$), where $X' \subseteq X$ and $Y' \subseteq Y$
- Is it possibble to prune association rules using the confidence measure?

Rule generation (2)

- Given a frequent iteset Y
- **Theorem:**

  if a rule X → Y – X does not satisfy the minconf threshold, then any rule X’ → Y – X’, where X’ ⊆ X, must not satisfy the minconf threshold as well
- Prove the theorem

Rule generation (3)



Example 2

Given the following database:

TrID	product
1	bread, milk
2	beer, milk, sugar
3	bread
4	bread, beer, milk
5	beer, milk, sugar

Assume the following values for minsup and minconf:

minsup = 30%
minconf = 70%

Example 2 (cont.)

C1

candidate 1-itemset	id	sup (%)
bread	1	60
milk	2	80
beer	3	60
sugar	4	40

L1

frequent 1-itemset	id	sup (%)
bread	1	60
milk	2	80
beer	3	60
sugar	4	40

C2

candidate 2-itemset	sup (%)
12	40
13	20
14	0
23	60
24	40
34	40

L2

frequent 2-itemset	sup (%)
12	40
23	60
24	40
34	40

Example 2 (cont.)

C3

candidate 3-itemset	sup (%)
234	40

L3

frequent 3-itemset	sup (%)
234	40

$C4 = \varnothing \rightarrow L4 = \varnothing$

This is the end of the first step - generation of frequent itemsets

Example 2 (cont.)

rule generation

fi	sup	rule	conf
1	0.40	beer → sugar	0.67
1	0.40	sugar → beer	1.00
2	0.60	beer → milk	1.00
2	0.60	milk → beer	0.75
3	0.40	sugar → milk	1.00
3	0.40	milk → sugar	0.50
4	0.40	milk → bread	0.50
4	0.40	bread → milk	0.67
5	0.40	beer ∧ sugar → milk	1.00
5	0.40	beer ∧ milk → sugar	0.67
5	0.40	sugar ∧ milk → beer	1.00
5	0.40	beer → sugar ∧ milk	0.67
5	0.40	sugar → beer ∧ milk	1.00
5	0.40	milk → beer ∧ sugar	0.50

Example 2 (cont.)
rule generation

Only few rules fulfil the confidence requirements. So, the final result of ***Apriori*** algorithm is the following:

fi	**sup**	**rule**	**conf**
1	0.40	sugar $\rightarrow$ beer	1.00
2	0.60	beer $\rightarrow$ milk	1.00
2	0.60	milk $\rightarrow$ beer	0.75
3	0.40	sugar $\rightarrow$ milk	1.00
5	0.40	beer $\wedge$ sugar $\rightarrow$ milk	1.00
5	0.40	sugar $\wedge$ milk $\rightarrow$ beer	1.00
5	0.40	sugar $\rightarrow$ beer $\wedge$ milk	1.00

Closed frequent itemsets (1)

- In any large dataset we can discover millions of frequent itemsets which has usually to be preserved for the future mining and rule generation
- It is useful to identify a small representative set of itemsets from which all other frequent itemsets can be derived
- Two such representations from which all other frequent itemsets can be derived are closed frequent itemsets and maximal frequent itemsets

Closed frequent itemsets (2)

- An itemset X is a closed in the dataset D if none of its immediate supersets has exactly the same support count as X (there is no immediate superset Y, $X \subset Y$, for which $\sup(X) = \sup(Y)$
- An itemset Y is a superset of X if it contains all items of the set X plus one additional item which does not belong to X
- An itemset X is a **closed frequent itemset** in the dataset D if it is closed and frequent (its support is greater than or equal to minsup)

Closed frequent itemsets (3)

- From a set of closed frequent itemsets we can derive all frequent itemsets together with their support counts
- A set of closed frequent itemsets – minimal representation of frequent itemsets that preserves the support information
- The number of closed frequent itemsets is usually much smaller (an order of magnitude) then the number of frequent itemsets

Closed frequent itemsets (4)

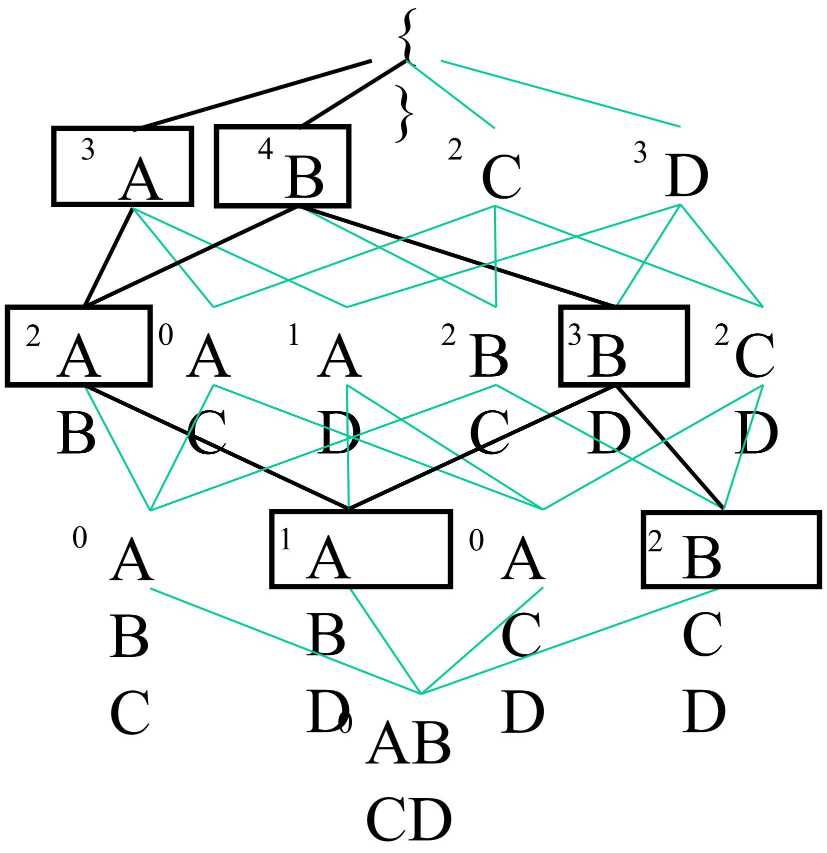
dataset

tid	items
1	A, B
2	B, C, D
3	A
4	A, B, D
5	B, C, D

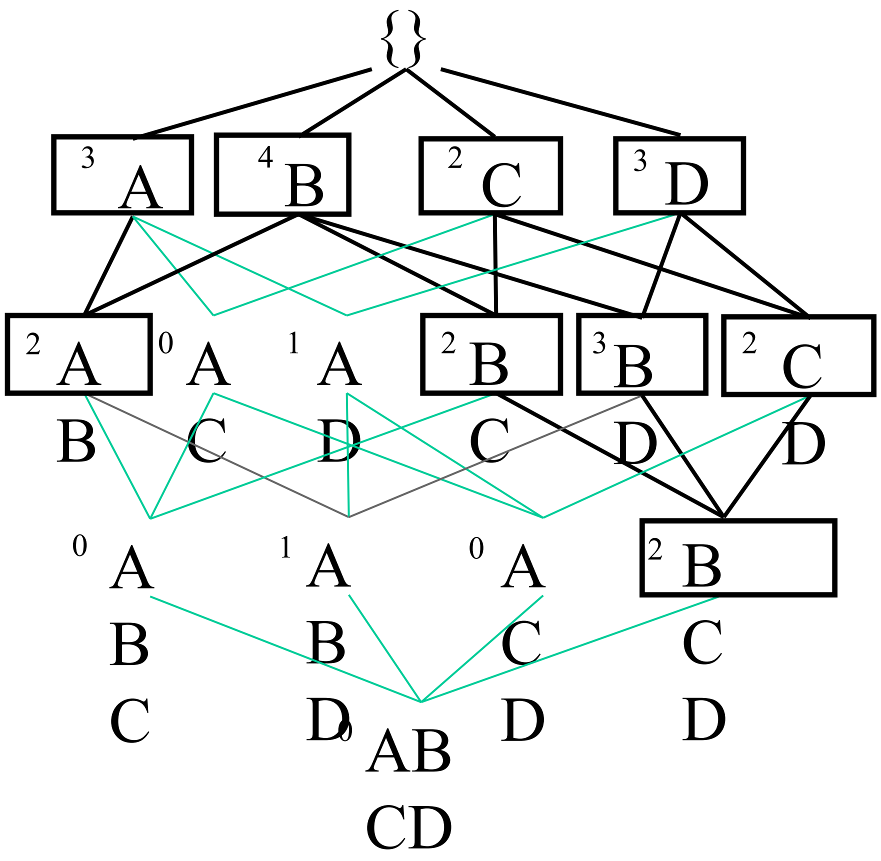
Assume that the minimum support threshold minsup = 30% (2 transactions)

set	support
A	3
B	4
C	2
D	3
A, B	2
A, C	0
A, D	1
B, C	2
B, D	3
C, D	2
A, B, C	0
A, B, D	1
A, C, D	0
B, C, D	2
A, B, C, D	0

Semi-lattice of closed itemsets

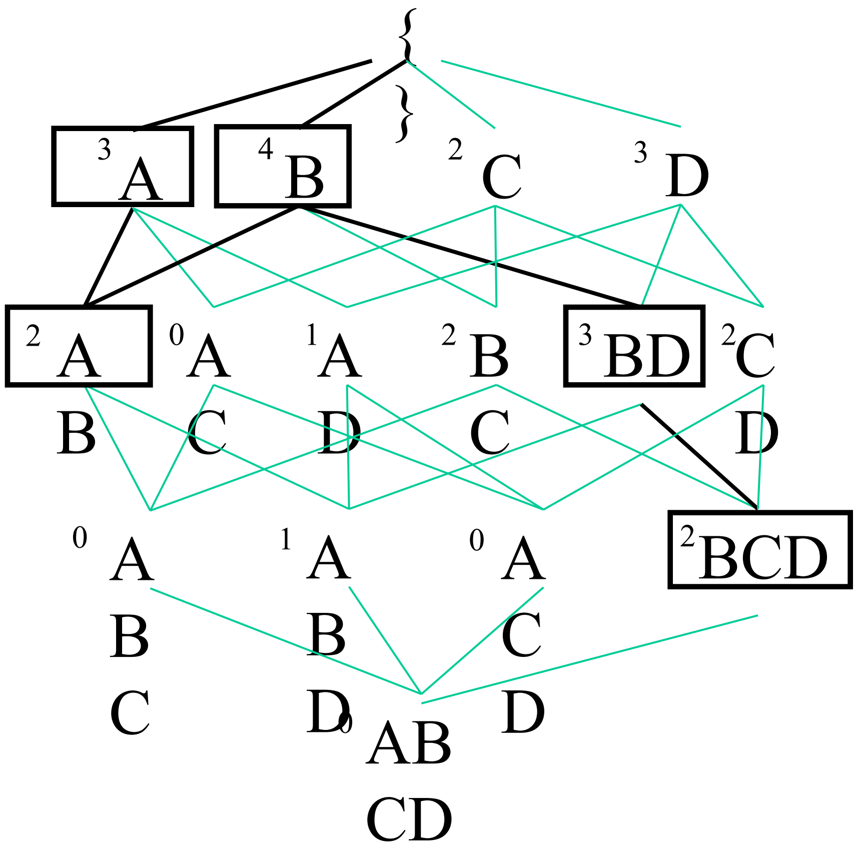


Semi-lattice of frequent itemsets



Semi-lattice of closed frequent itemsets

closed frequent
Itemsets (5 sets)



frequent itemsets
(9 sets)

Generation of frequent itemsets from a set of closed frequent itemsets (1)

```
1.  Let FI denotes a collection of frequent itemsets, and $Dom_k$ denotes a
    collection of closed frequent k-itemsets;
2.  FI = $\varnothing$;
3.  k = 1;
4.  FI $\leftarrow$ $Dom_k$;     \*add all closed frequent 1-itemsets to FI *\,
5.  k= k+1;
6.  while $Dom_k \neq \varnothing$
    – for each $X_k \in Dom_k$
        • generate all subsets $X_{ik}$, i=1,…, m, of the set $X_k$ ;
```

Generation of frequent itemsets from a set of closed frequent itemsets (2)

```
            • for i= 1 to m do
                – if $X_{ik} \notin FI$ then
                  $FI \leftarrow FI \cup \{X_{ik}\}$;
                  $sup(X_{ik}) = sup(X_k)$;
            • end for
        –   $k = k+1$;
8.      end while;
9.      return
```

Generation of frequent itemsets from a set of closed frequent itemsets (3)

- Example:
  - FI = $\varnothing$; k=1;
  - FI $\leftarrow$ Dom_1;   FI = {(A), (B)}, sup(A) = 3, sup(B) = 4;
  - k=2;
  - Dom_2 = {(A, B), (B, D)}
  - X_{12} = (A, B); subsets of the set X_{12} ={(A), (B), (A, B)}
  - subsets (A) i (B) are already in FI; * omit their analysis*\
  - subset (A, B) $\notin$ FI, so add (A, B) to FI, sup(A, B) = 2;
  - X_{22} = (C, D) subsets of the set X_{22} ={(B), (D), (B, D)}
  - subset (B) is already in FI; * omit his analysis *\
  - subset (D) $\notin$ FI, so add (D) to FI, sup(D) = sup(B, D) = 3;

Generation of frequent itemsets from a set of closed frequent itemsets (4)

- Example (cd.):
  - Subset (B, D) $\notin$ FI, so add (B, D) to FI, sup(B, D) = 3;
  - k= 3;
  - Dom_3 = {(B, C, D)}
  - X_{13} = (B, C, D); subsets of X_{13} ={(B), (C), (D), (B, C), (B,D), (C, D)}
  - subsets (B), (D) i (B, D) are already in FI; * omit their analysis*\
  - subset (C) $\notin$ FI, so add (C) to FI, sup(C) = sup(B, C, D) = 2;
  - subsets (B, C) and (C, D) $\notin$ FI, add (B, C) and (C, D) to FI, sup(B, C) = sup(B, C, D) = 2; sup(C, D) = sup(B, C, D) = 2;
  - Subset (B, C, D) $\notin$ FI, add (B, C, D) to FI, sup(B, C, D) = 2;
  - $Dom_4 = \varnothing$, end of the algorithm

Maximal frequent itemsets (1)

- An itemset X is a maximal frequent itemset in the dataset D if it is frequent and none of its immediate supersets Y is frequent
- Maximal frequent itemsets provide most compact representation of frequent itemsets, however they do contain the full support information of their subsets
- All frequent itemsets contained in a dataset D are subsets of maximal frequent itemsets of D

Maximal frequent itemsets (2)

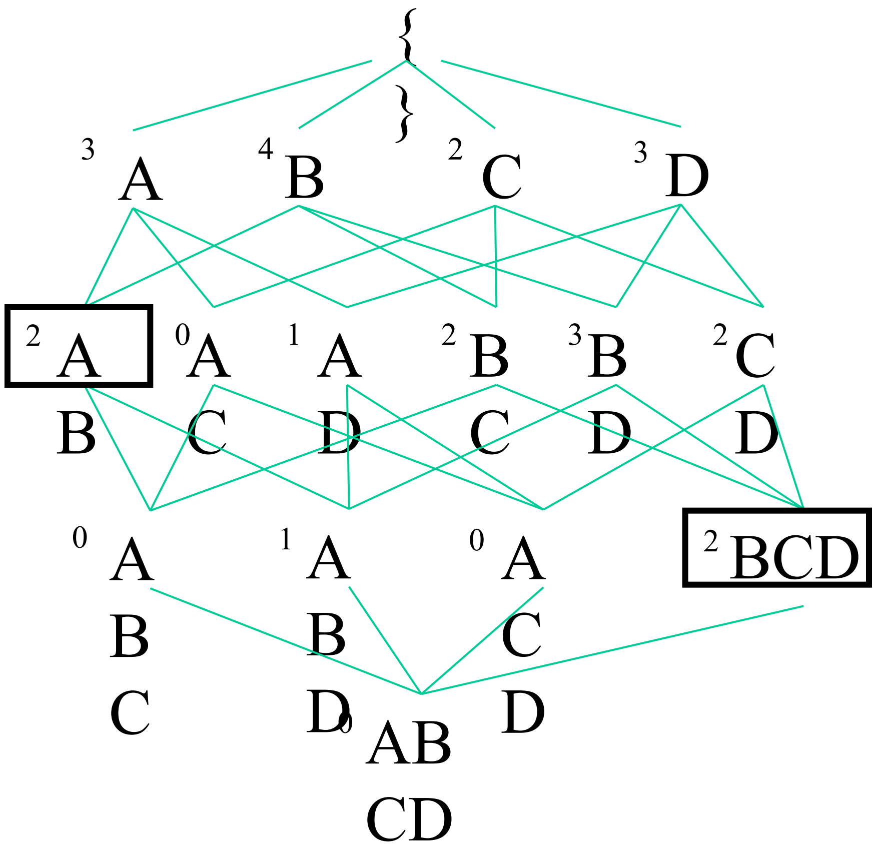
- Let us consider the dataset given below:

tid	items
1	A, B
2	B, C, D
3	A
4	A, B, D
5	B, C, D

set	support
A	3
B	4
C	2
D	3
A, B	2
A, C	0
A, D	1
B, C	2
B, D	3
C, D	2

set	support
A, B, C	0
A, B, D	1
A, C, D	0
B, C, D	2
A, B, C, D	0

Maximal frequent itemsets (3)



Maximal frequent itemsets (4)

- Maximal frequent itemsets:
  (A, B) and (B, C, D)
- Easy to notice that all other frequent itemsets can be derived from both sets
- From (A,B) the following 3 frequent itemsets can be derived: (A), (B) i (A, B)
- From (B, C, D) we derive 6 frequent itemsets: (C), (D), (B, C), (B, D), (C, D), (B, C, D)
- Maximal frequent itemsets provide most compact representation of frequent itemsets, however they do contain the full support information of their subsets

Generalized Association Rules
or
Multilevel Association Rules

Multilevel AR (1)

- It is difficult to find interesting, strong associations among data items at a too primitive level due to the **sparsity of data**
- Approach: reason at suitable level of abstraction
- Data mining system should provide capabilities to mine association rules at multiple levels of abstractions and traverse easily among different abstraction levels

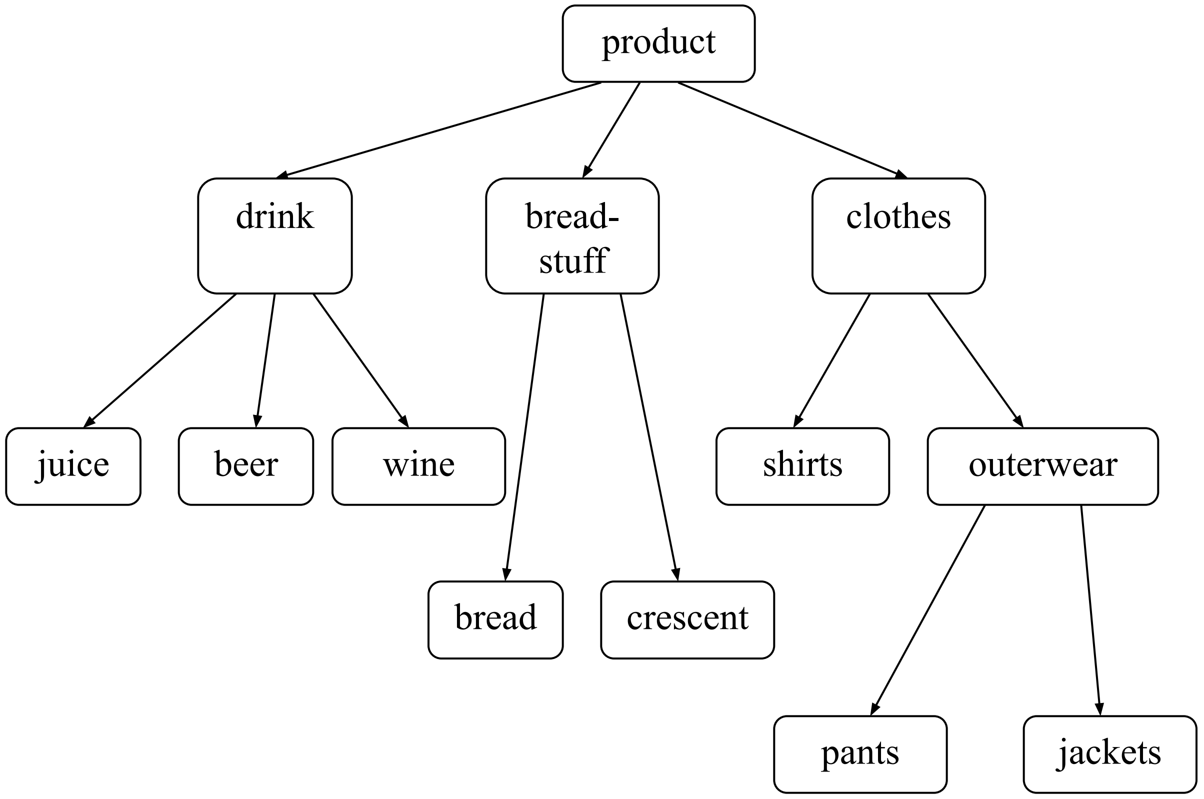
Multilevel AR (2)

- Multilevel association rule:

  „50% of clients who purchase bread-stuff (bread, rolls, croissants, etc.) purchase also diary products”
- **A multilevel (generalized) association rule** is an association rule which represents an association among named abstract groups of items (events, properties, services, etc.)
- Multilevel association rules represent associations at multiple levels of abstractions which are more understandable and represent more general knowledge
- Multilevel association rules can’t be derived from single-level association rules

Multilevel AR - item hierarchy

- An attribute (item) may be generalized or specialized according to a hierarchy of concepts (dimension hierarchy!)



Item hierarchy

- **Item hierarchy (dimension hierarchy)** – semantic classification of items
  - It describes generalization/specialization relationship among items and/or abstract groups of items
  - It is a rooted tree whose leaves represent items of the set I, and whose internal nodes represent abstract groups of items
  - A root of the hierarchy represents the set I (all items)
- dimensions and levels can be efficiently encoded in transactions
- multilevel (generalized) association rules: rules which combine associations with item hierarchy

Basic algorithm (1)

1. Extend each transaction $T_i \in D$ by adding all ancestors of each item in a transaction to the transaction (extended transaction) (omit the root of the taxonomy and, eventually, remove all repeating items)
2. Run any of algorithms for mining association rules over those “extended transactions” (e.g. Apriori)
3. Remove all trivial multilevel association rules

Basic algorithm (2)

- A trivial multilevel association rule is the rule of the form „node → ancestor (node)”, where node represents a single item or an abstract group of items
- Use taxonomy information to prune redundant or uninteresting rules
- Replace many specialized rules with one general rule: e.g. rules „bread → drinks” and „croissant → drinks” replace with the rule „breadstuff → drinks” (use taxonomy information to perform the replacement)

Drawbacks of the basic algorithm

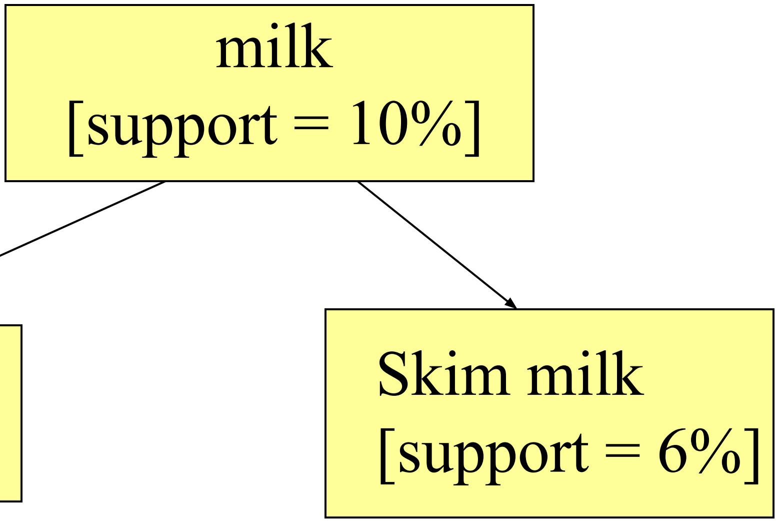
- Drawbacks of the approach:
  - The number of candidate itemsets is much larger,
  - The size of the average candidate itemset is much larger.
  - The number of database scans is larger

MAR: uniform support vs. reduced support

- Uniform support: the same minimum support for all levels
  - one minsup: no need to examine itemsets containing any item whose ancestors do not have minimum support
  - minsup value:
    - high: miss low level associations
    - low: generate too many high level associations

MAR: uniform support vs. reduced support

- Reduced Support: reduced minimum support at lower levels - different strategies possible



Basic MAR discovery algorithm with reduced minsup

Top-down greedy algorithm

- Step 1: find all frequent items (abstract groups of items) at the highest level of the taxonomy (most abstract level)
- Step 2: find all frequent items at consecutive lower levels of the taxonomy – till leaves of the taxonomy
- Step 3: find frequent itemsets containing frequent items belonging to different levels of the taxonomy