Association rule mining

- Association rule induction: Originally designed for **market basket analysis**.
- Aims at finding patterns in the shopping behavior of customers of supermarkets, mail-order companies, on-line shops etc.
- More specifically:
  Find sets of products that are frequently bought together.
- Example of an association rule:

  If a customer buys bread and wine,
  then she/he will probably also buy cheese.

Association rule mining

- Possible applications of found association rules:
  - Improve arrangement of products in shelves, on a catalog's pages.
  - Support of cross-selling (suggestion of other products), product bundling.
  - Fraud detection, technical dependence analysis.
  - Finding business rules and detection of data quality problems.
  - ...

Association rules

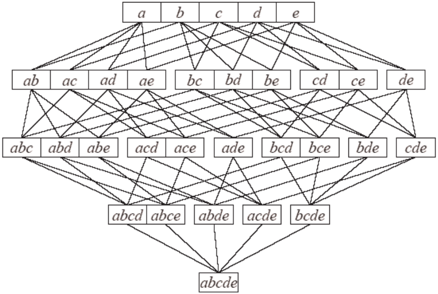
- Assessing the quality of association rules:
  - **Support of an item set**:
    Fraction of transactions (shopping baskets/carts) that contain the item set.
  - **Support of an association rule** $X \rightarrow Y$:
    Either: Support of $X \cup Y$
    (more common: rule is correct)
    Or: Support of X
    (more plausible: rule is applicable)
  - **Confidence of an association rule** $X \rightarrow Y$:
    Support of $X \cup Y$ divided by support of X (estimate of $P(Y \mid X)$).

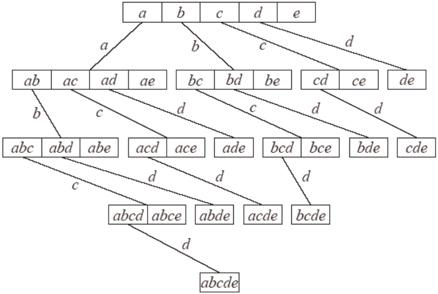
Association rules

- Two step implementation of the search for association rules:
  - Find the **frequent item sets** (also called large item sets), i.e., the item sets that have at least a user-defined **minimum support**.
  - Form rules using the frequent item sets found and select those that have at least a user-defined **minimum confidence**.

Finding frequent item sets

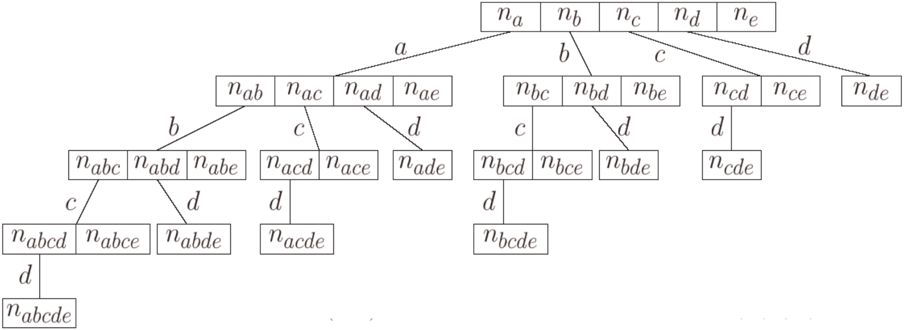
Subset lattice and a prefix tree for five items:





- It is not possible to determine the support of all possible item sets, because their number grows exponentially with the number of items.
- Efficient methods to search the subset lattice are needed.

Item set trees



A (full) item set tree for the five items $a, b, c, d,$ and e.

- Based on a global order of the items.
- The item sets counted in a node consist of
  - all items labeling the edges to the node (common prefix) and
  - one item following the last edge label.

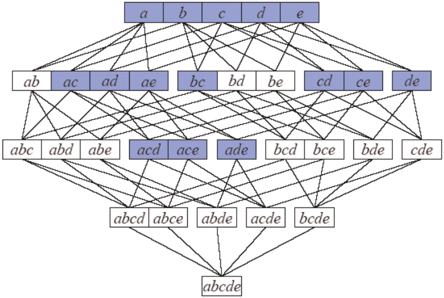
Item set tree pruning

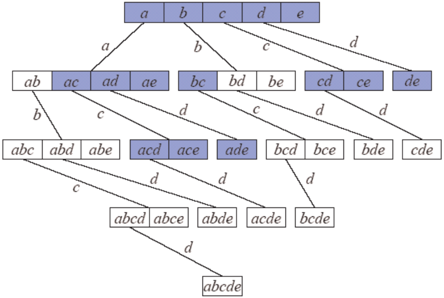
In applications item set trees tend to get very large, so pruning is needed.

- **Structural Pruning:**
  - Make sure that there is only one counter for each possible item set.
  - Explains the unbalanced structure of the full item set tree.
- **Size Based Pruning:**
  - Prune the tree if a certain depth (a certain size of the item sets) is reached.
  - Idea: Rules with too many items are difficult to interpret.
- **Support Based Pruning:**
  - **No superset of an infrequent item set can be frequent.**
  - No counters for item sets having an infrequent subset are needed.

Searching the subset lattice

Boundary between frequent (blue) and infrequent (white) item sets:





- **Apriori**: Breadth-first search (item sets of same size).
- **Eclat**: Depth-first search (item sets with same prefix).

Apriori: Breadth first search

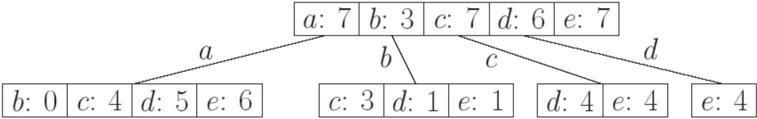
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$

a: 7	b: 3	c: 7	d: 6	e: 7

- Example transaction database with 5 items and 10 transactions.
- Minimum support: 30%, i.e., at least 3 transactions must contain the item set.
- All one item sets are frequent $\rightarrow$ full second level is needed.

Apriori: Breadth first search

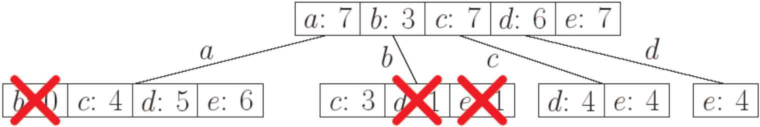
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Determining the support of item sets: For each item set traverse the database and count the transactions that contain it (highly inefficient).
- Better: Traverse the tree for each transaction and find the item sets it contains (efficient: can be implemented as a simple double recursive procedure).

Apriori: Breadth first search

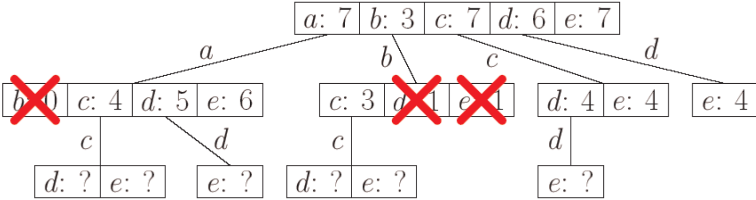
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Minimum support: 30%, i.e., at least 3 transactions must contain the item set.
- Infrequent item sets: $\{a, b\}$, $\{b, d\}$, $\{b, e\}$.
- The subtrees starting at these item sets can be pruned.

Apriori: Breadth first search

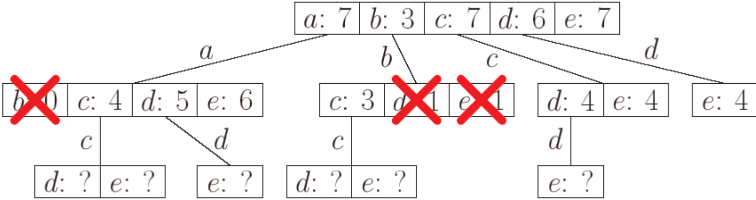
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Generate candidate item sets with 3 items (parents must be frequent).

Apriori: Breadth first search

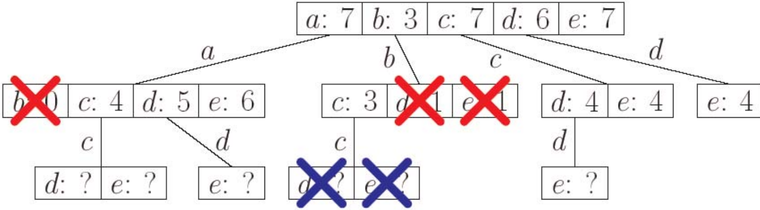
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Before counting, check whether the candidates contain an infrequent item set.
    - An item set with k items has k subsets of size $k-1$.
    - The parent is only one of these subsets.

Apriori: Breadth first search

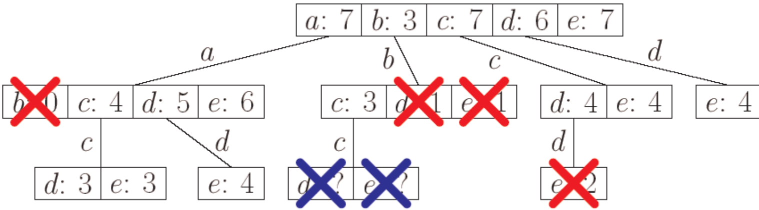
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- The item sets $\{b, c, d\}$ and $\{b, c, e\}$ can be pruned, because
  - $\{b, c, d\}$ contains the infrequent item set $\{b, d\}$ and
  - $\{b, c, e\}$ contains the infrequent item set $\{b, e\}$.
- Only the remaining four item sets of size 3 are evaluated.

Apriori: Breadth first search

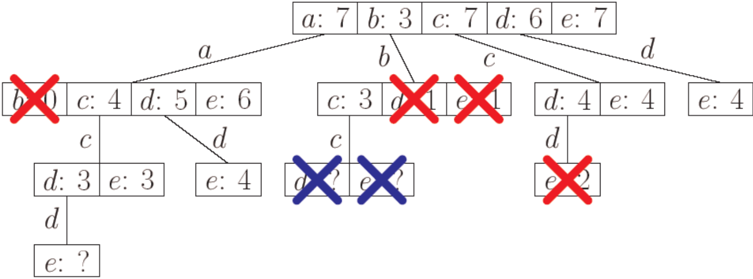
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Minimum support: 30%, i.e., at least 3 transactions must contain the item set.
- Infrequent item set: $\{c, d, e\}$.

Apriori: Breadth first search

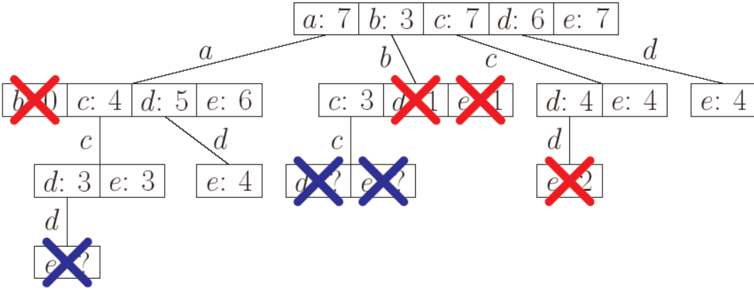
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Generate candidate item sets with 4 items (parents must be frequent).
- Before counting, check whether the candidates contain an infrequent item set.

Apriori: Breadth first search

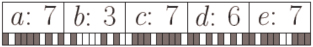
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- The item set $\{a, c, d, e\}$ can be pruned, because it contains the infrequent item set $\{c, d, e\}$.
- Consequence: No candidate item sets with four items.
- Fourth access to the transaction database is not necessary.

Eclat: Depth first search

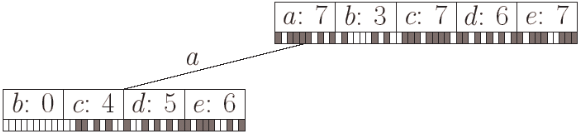
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Form a transaction list for each item. Here: bit vector representation.
  - grey: item is contained in transaction
  - white: item is not contained in transaction
- Transaction database is needed only once (for the single item transaction lists).

Eclat: Depth first search

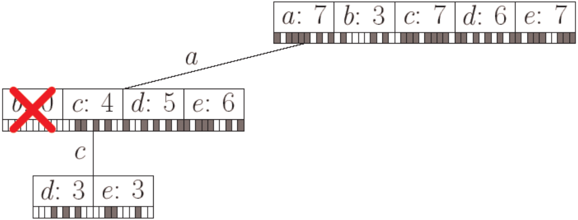
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Intersect the transaction list for item a with the transaction lists of all other items.
- Count the number of set bits (containing transactions).
- The item set $\{a, b\}$ is infrequent and can be pruned.

Eclat: Depth first search

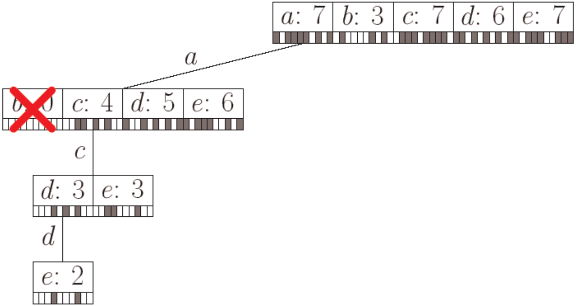
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Intersect the transaction list for $\{a, c\}$ with the transaction lists of $\{a, x\}$, $x \in \{d, e\}$.
- Result: Transaction lists for the item sets $\{a, c, d\}$ and $\{a, c, e\}$.

Eclat: Depth first search

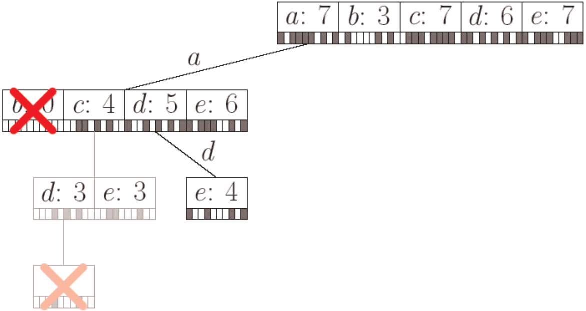
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Intersect the transaction list for $\{a, c, d\}$ and $\{a, c, e\}$.
- Result: Transaction list for the item set $\{a, c, d, e\}$.
- With Apriori this item set could be pruned before counting, because it was known that $\{c, d, e\}$ is infrequent.

Eclat: Depth first search

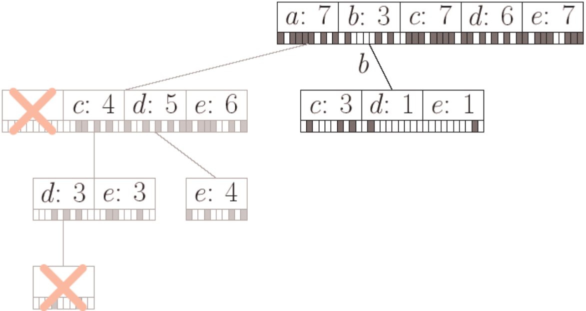
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Backtrack to the second level of the search tree and intersect the transaction list for $\{a, d\}$ and $\{a, e\}$.
- Result: Transaction list for $\{a, d, e\}$.

Eclat: Depth first search

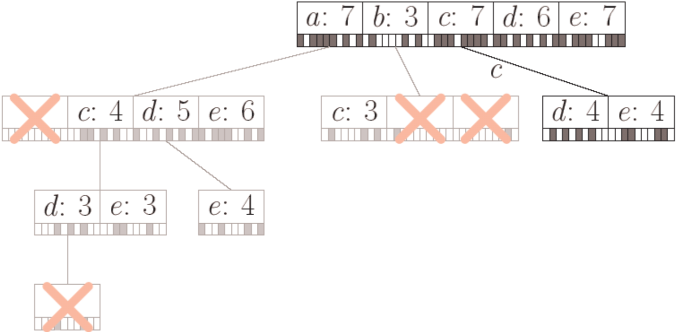
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Backtrack to the first level of the search tree and intersect the transaction list for b with the transaction lists for c, d, and e.
- Result: Transaction lists for the item sets $\{b, c\}$, $\{b, d\}$, and $\{b, e\}$.
- Only one item set with sufficient support $\rightarrow$ prune all subtrees.

Eclat: Depth first search

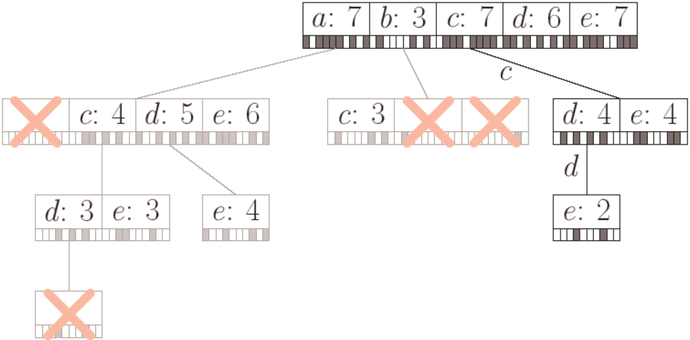
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Backtrack to the first level of the search tree and intersect the transaction list for c with the transaction lists for d and e.
- Result: Transaction lists for the item sets $\{c, d\}$ and $\{c, e\}$.

Eclat: Depth first search

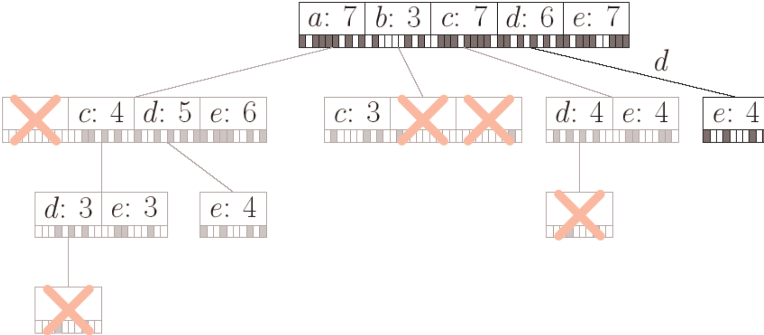
1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Intersect the transaction list for $\{c, d\}$ and $\{c, e\}$.
- Result: Transaction list for $\{c, d, e\}$.
- Infrequent item set: $\{c, d, e\}$.

Eclat: Depth first search

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{c, b, e\}$
10: $\{a, d, e\}$



- Backtrack to the first level of the search tree and intersect the transaction list for d with the transaction list for e.
- Result: Transaction list for the item set $\{d, e\}$.
- With this step the search is finished.

1 item	2 items		3 items
$\{a\}^+$: 70%	$\{a,c\}^+$: 40%	$\{c,e\}^+$: 40%	$\{a,c,d\}^{+*}$: 30%
$\{b\}$: 30%	$\{a,d\}^+$: 50%	$\{d,e\}$: 40%	$\{a,c,e\}^{+*}$: 30%
$\{c\}^+$: 70%	$\{a,e\}^+$: 60%		$\{a,d,e\}^{+*}$: 40%
$\{d\}^+$: 60%	$\{b,c\}^{+*}$: 30%		
$\{e\}^+$: 70%	$\{c,d\}^+$: 40%		

Types of frequent item sets

- **Free Item Set**: Any frequent item set (support is higher than the minimal support).
- **Closed Item Set** (marked with $^+$): A frequent item set is called <u>closed</u> if no superset has the same support.
- **Maximal Item Set** (marked with *): A frequent item set is called <u>maximal</u> if no superset is frequent.

Generating association rules

For each frequent item set S:

- Consider all pairs of sets $X, Y \in S$ with $X \cup Y = S$ and $X \cap Y = \emptyset$. Common restriction: $|Y| = 1$, i.e. only one item in consequent (then-part).
- Form the association rule $X \rightarrow Y$ and compute its confidence.

$$\text{conf}(X \rightarrow Y) = \frac{\text{supp}(X \cup Y)}{\text{supp}(X)} = \frac{\text{supp}(S)}{\text{supp}(X)}$$

- Report rules with a confidence higher than the minimum confidence.

Further rule filtering can rely on:

- Require a minimum difference between rule confidence and consequent support.
- Compute information gain or χ^2 for antecedent (if-part) and consequent.

Generating association rules

Example: $S = \{a, c, e\}$, $X = \{c, e\}$, $Y = \{a\}$.

$$\text{conf}(c, e \to a) = \frac{\text{supp}(\{a, c, e\})}{\text{supp}(\{c, e\})} = \frac{30\%}{40\%} = 75\%$$

Minimum confidence: 80%

association rule	support of all items	support of antecedent	confidence
$b \to c$:	30%	30%	100%
$d \to a$:	50%	60%	83.3%
$e \to a$:	60%	70%	85.7%
$a \to e$:	60%	70%	85.7%
$d, e \to a$:	40%	40%	100%
$a, d \to e$:	40%	50%	80%

Summary association rules

- **Association Rule Induction is a Two Step Process**
  - Find the frequent item sets (minimum support).
  - Form the relevant association rules (minimum confidence).
- **Finding the Frequent Item Sets**
  - Top-down search in the subset lattice / item set tree.
  - Apriori: Breadth first search; Eclat: Depth first search.
  - Other algorithms: FP-growth, H-Mine, LCM, Mafia, Relim etc.
  - Search Tree Pruning:
    No superset of an infrequent item set can be frequent.
    (other possible
- **Generating the Association Rules**
  - Form all possible association rules from the frequent item sets.
  - Filter "interesting" association rules.

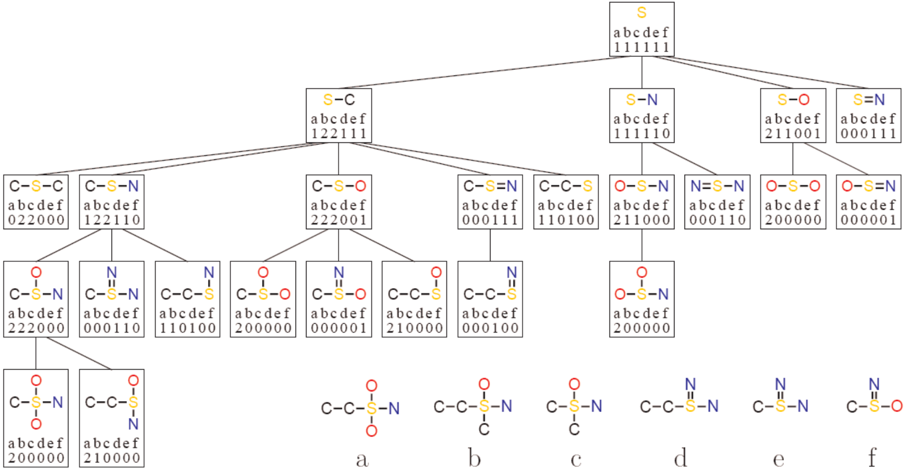
Structured itemsets

Sometimes, an additional structure is imposed on the "item sets".

- The "item sets" are sequences of events.
  - For instance: Customer contact (buying, complaint, questionnaire,...)
  - Association rules have the form: If a and then b happens, then probably c happens next.
- Items sets are molecules: Find frequent substructures.

The additional structure leads to different tree structure, but the principal algorithm remains the same.

Finding frequent molecule structures



Other applications

- Finding business rules and detection of data quality problems.
  - Association rules with confidence close to 100% could be business rules.
  - Exceptions might be caused by data quality problems.
- Construction of partial classifiers.
  - Search for association rules with a given conclusion part.
  - If . . ., then the customer probably buys the product.