# Association Analysis: Basic Concepts and Algorithms

# Association Rule Mining

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

**Market-Basket transactions**

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

**Example of Association Rules**

{Diaper} $\rightarrow$ {Beer},
{Milk, Bread} $\rightarrow$ {Eggs,Coke},
{Beer, Bread} $\rightarrow$ {Milk},

Implication means co-occurrence, not causality!

# Definition: Frequent Itemset

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

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

# Definition: Association Rule

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

| 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 |

- **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

Example:

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

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

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

# Association Rule Mining Task

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

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

  $\Rightarrow$ Computationally prohibitive!

# Mining Association Rules

| *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 Rules:

{Milk,Diaper} $\rightarrow$ {Beer} (s=0.4, c=0.67)
{Milk,Beer} $\rightarrow$ {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} $\rightarrow$ {Milk} (s=0.4, c=0.67)
{Beer} $\rightarrow$ {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} $\rightarrow$ {Milk,Beer} (s=0.4, c=0.5)
{Milk} $\rightarrow$ {Diaper,Beer} (s=0.4, c=0.5)

## Observations:

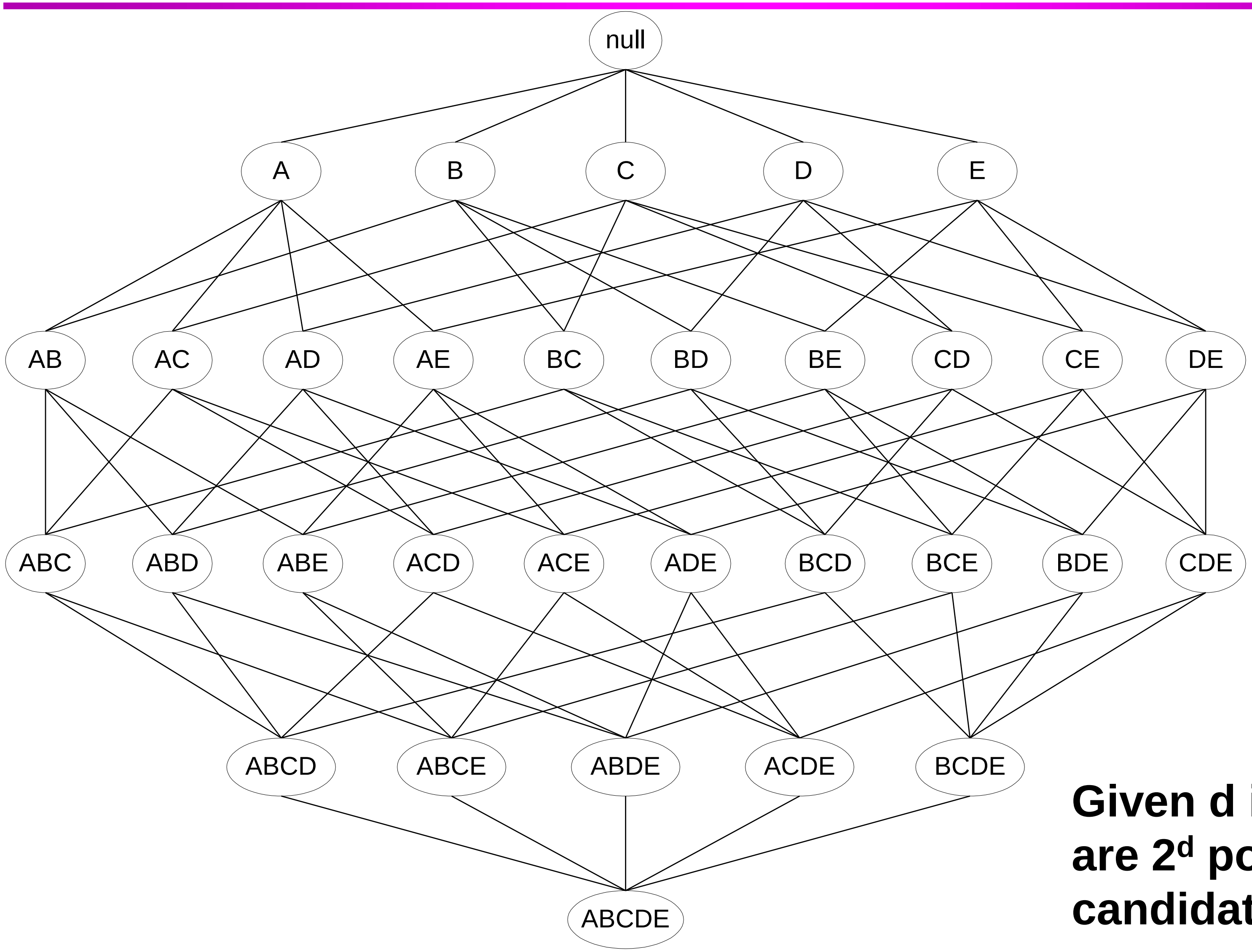
- All the above rules are binary partitions of the same itemset:
  {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may decouple the support and confidence requirements

# Mining Association Rules

- Two-step approach:
  1. Frequent Itemset Generation
     – Generate all itemsets whose support $\geq$ minsup
  2. Rule Generation
     – Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset

- Frequent itemset generation is still computationally expensive

# Frequent Itemset Generation




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

# Frequent Itemset Generation

- Brute-force approach:
  - Each itemset in the lattice is a candidate frequent itemset
  - Count the support of each candidate by scanning the database

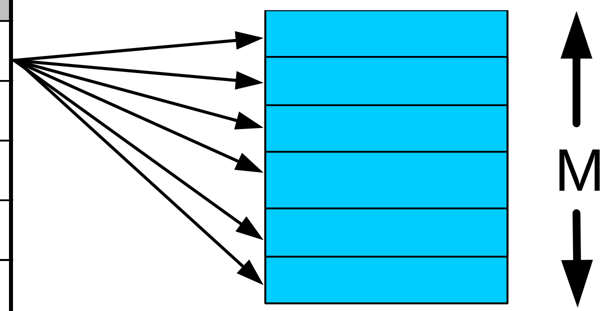
**Transactions**

N

| *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 |

w

**List of Candidates**




  - Match each transaction against every candidate
  - Complexity ~ O(NMw) => Expensive since $M = 2^d$ !!!

# Frequent Itemset Generation Strategies

- Reduce the number of candidates (M)
  - Complete search: $M=2^d$
  - Use pruning techniques to reduce M

- Reduce the number of transactions (N)
  - Reduce size of N as the size of itemset increases

- Reduce the number of comparisons (NM)
  - Use efficient data structures to store the candidates or transactions
  - No need to match every candidate against every transaction

# Reducing Number of Candidates

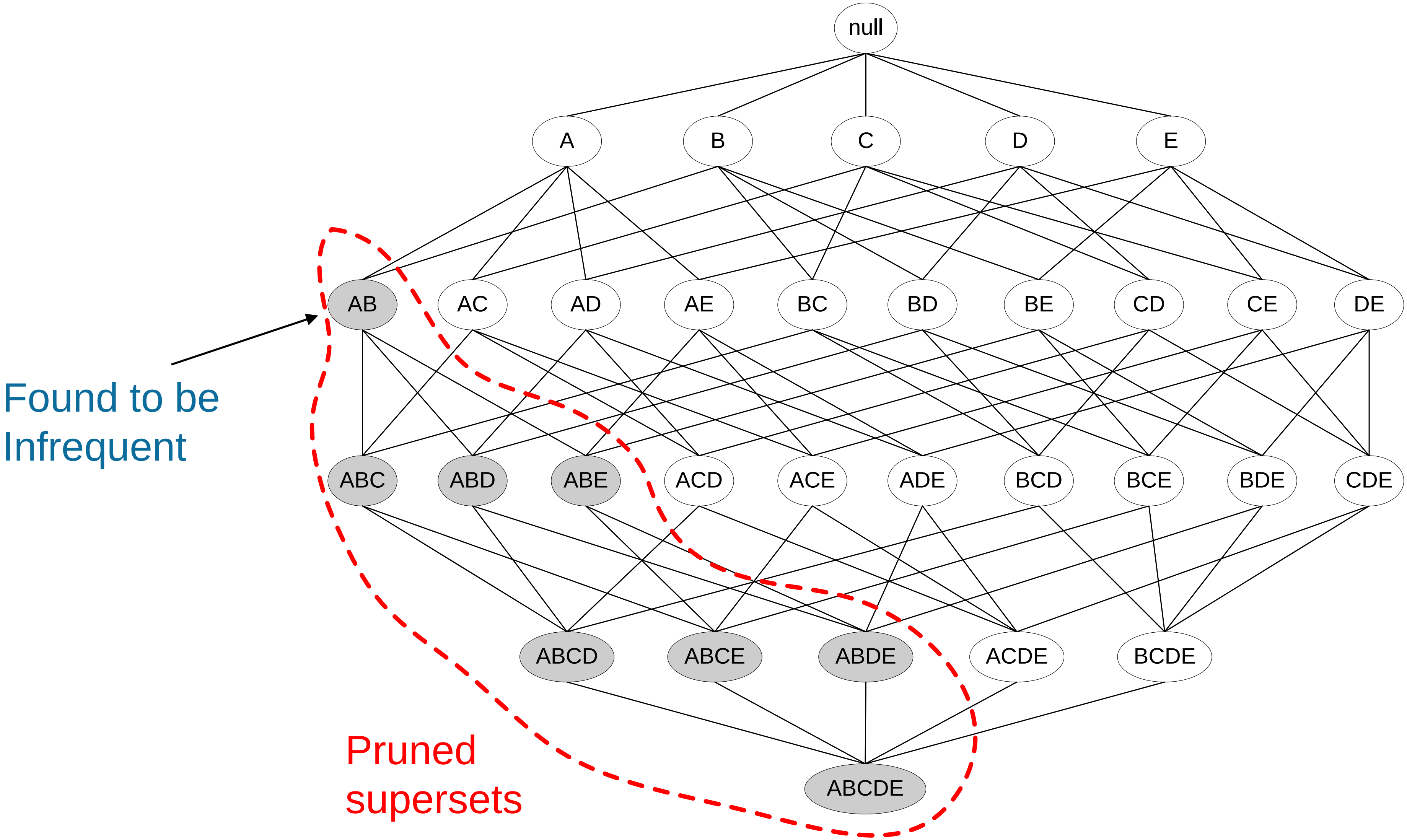
- Apriori principle:
  - If an itemset is frequent, then all of its subsets must also be frequent

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

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

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

# Illustrating Apriori Principle



# Illustrating Apriori Principle

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

Items (1-itemsets)

| Itemset | Count |
|---|---|
| **{Bread,Milk}** | **3** |
| **{Bread,Beer}** | **2** |
| **{Bread,Diaper}** | **3** |
| **{Milk,Beer}** | **2** |
| **{Milk,Diaper}** | **3** |
| **{Beer,Diaper}** | **3** |

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Minimum Support = 3

Triplets (3-itemsets)

| Itemset | Count |
|---|---|
| **{Bread,Milk,Diaper}** | **3** |

If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3 = 41$

With support-based pruning,

$6 + 6 + 1 = 13$

# Apriori Algorithm

- Method:
  - 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 candidates that are infrequent, leaving only those that are frequent

# Apriori Algorithm

**Algorithm 6.1** Frequent itemset generation of the *Apriori* algorithm.

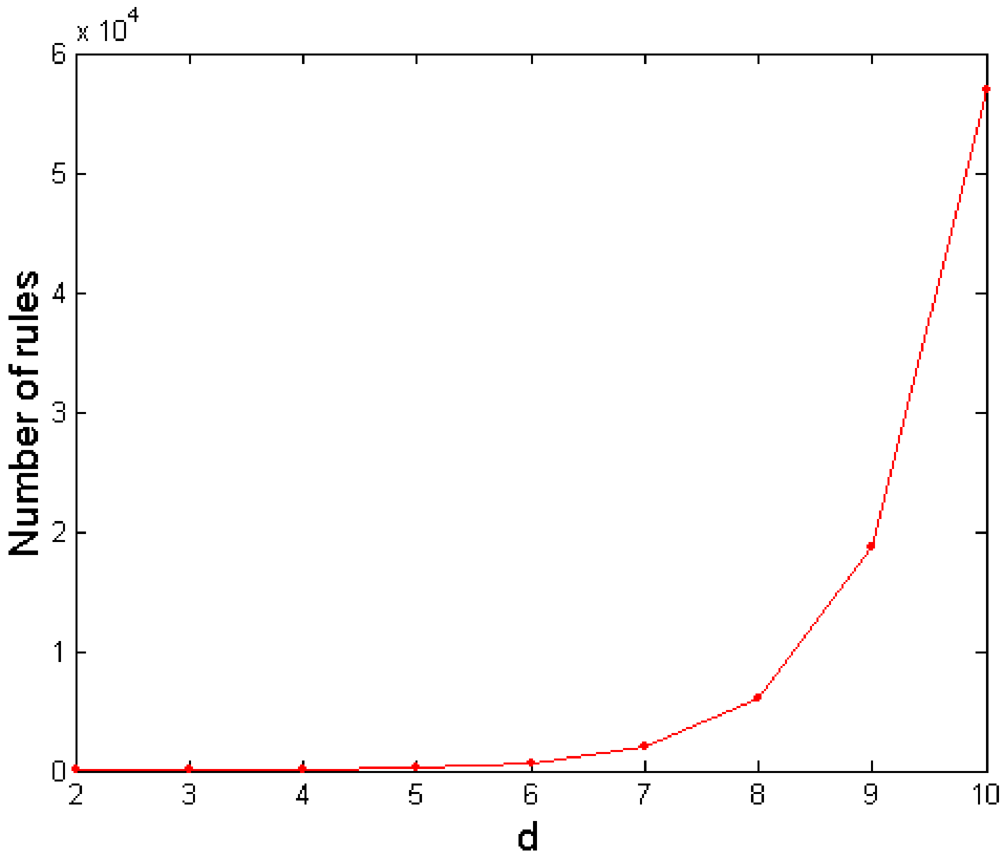
$k = 1$.
$F_k = \{\ i \mid i \in I \wedge \sigma(\{i\}) \geq N \times minsup\}$. {Find all frequent 1-itemsets}
**repeat**
  $k = k + 1$.
  $C_k = \text{apriori-gen}(F_{k-1})$. {Generate candidate itemsets}
  **for** each transaction $t \in T$ **do**
    $C_t = \text{subset}(C_k, t)$. {Identify all candidates that belong to $t$}
    **for** each candidate itemset $c \in C_t$ **do**
      $\sigma(c) = \sigma(c) + 1$. {Increment support count}
    **end for**
  **end for**
  $F_k = \{\ c \mid c \in C_k \wedge \sigma(c) \geq N \times minsup\}$. {Extract the frequent $k$-itemsets}
**until** $F_k = \emptyset$
Result $= \bigcup F_k$.

# Factors Affecting Complexity

- Choice of minimum support threshold
  - lowering support threshold results in more frequent itemsets
  - this may increase number of candidates and max length of frequent itemsets
- Dimensionality (number of items) of the data set
  - more space is needed to store support count of each item
  - if number of frequent items also increases, both computation and I/O costs may also increase
- Size of database
  - since Apriori makes multiple passes, run time of algorithm may increase with number of transactions
- Average transaction width
  - transaction width increases with denser data sets
  - This may increase max length of frequent itemsets and traversals of hash tree (number of subsets in a transaction increases with its width)

# Rule Generation: Computational Complexity

- Given d unique items:
  - Total number of itemsets = $2^d$
  - Total number of possible association rules:




$$R = \sum_{k=1}^{d-1}\left[\binom{d}{k} \times \sum_{j=1}^{d-k}\binom{d-k}{j}\right] = 3^d - 2^{d+1} + 1$$

**If d=6, R = 602 rules**

# Rule Generation

- Given a frequent itemset L, find all non-empty subsets $f \subset L$ such that $f \to L - f$ satisfies the minimum confidence requirement
  - If {A,B,C,D} is a frequent itemset, candidate rules:

    $ABC \to D$, $ABD \to C$, $ACD \to B$, $BCD \to A$,
    $A \to BCD$, $B \to ACD$, $C \to ABD$, $D \to ABC$
    $AB \to CD$, $AC \to BD$, $AD \to BC$, $BC \to AD$,
    $BD \to AC$, $CD \to AB$,

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

# Rule Generation

- How to efficiently generate rules from frequent itemsets?
  - In general, confidence does not have an anti-monotone property

    $c(ABC \rightarrow D)$ can be larger or smaller than $c(AB \rightarrow D)$

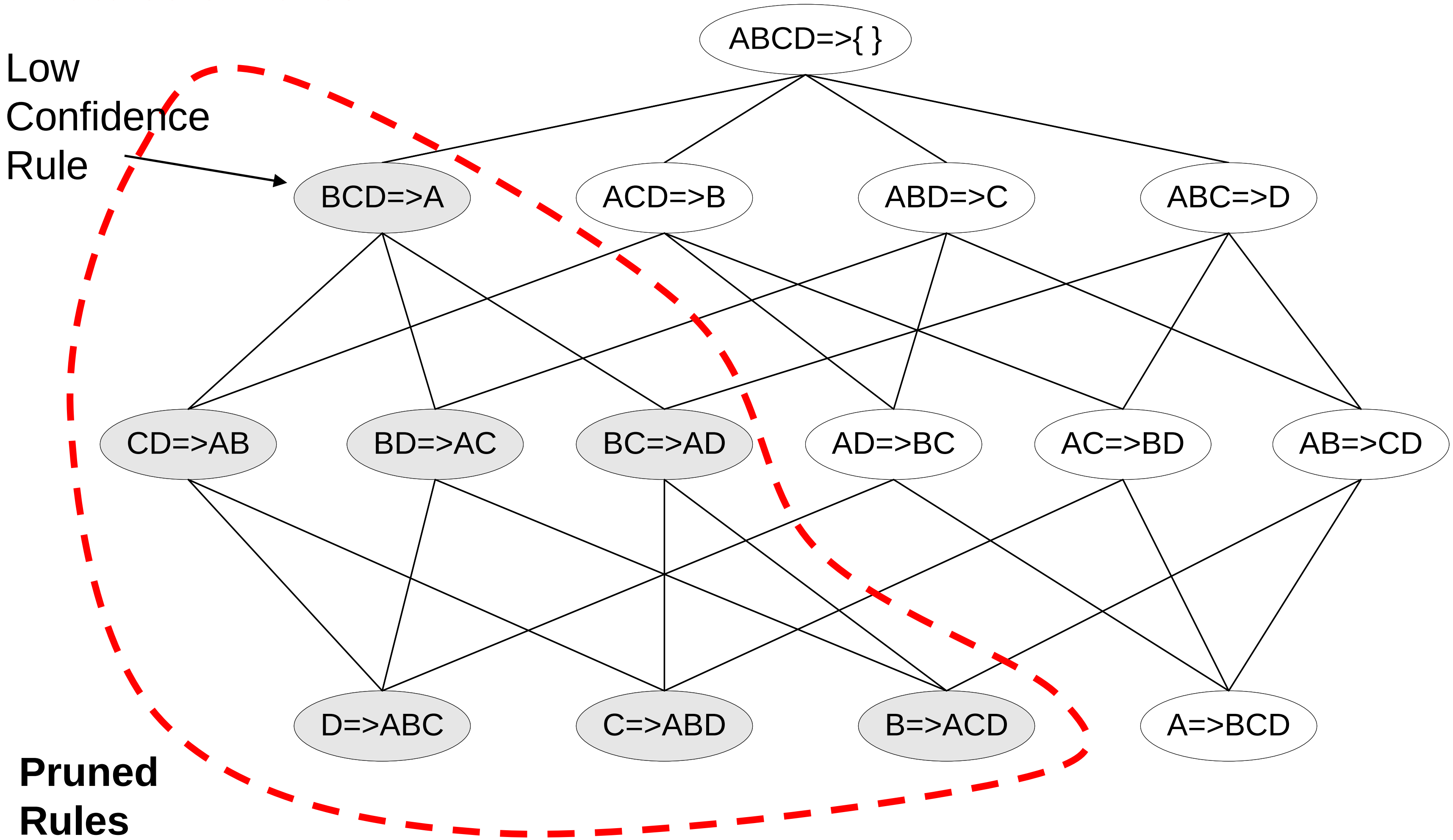
  - But confidence of rules generated from the same itemset has an anti-monotone property
  - e.g., L = {A,B,C,D}:

$$c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$$

    - Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

# Rule Generation for Apriori Algorithm

## Lattice of rules



# Pattern Evaluation

- Association rule algorithms tend to produce too many rules
  - many of them are uninteresting or redundant
  - Redundant if $\{A,B,C\} \rightarrow \{D\}$ and $\{A,B\} \rightarrow \{D\}$ have same support & confidence

- Interestingness measures can be used to prune/rank the derived patterns

- In the original formulation of association rules, support & confidence are the only measures used

# Computing Interestingness Measure

- Given a rule $X \rightarrow Y$, information needed to compute rule interestingness can be obtained from a contingency table

Contingency table for $X \rightarrow Y$

| | $Y$ | $\overline{Y}$ | |
|---|---|---|---|
| $X$ | $f_{11}$ | $f_{10}$ | $f_{1+}$ |
| $\overline{X}$ | $f_{01}$ | $f_{00}$ | $f_{o+}$ |
| | $f_{+1}$ | $f_{+0}$ | $\|T\|$ |

$f_{11}$: support of $X$ and $Y$
$f_{10}$: support of $X$ and $\overline{Y}$
$f_{01}$: support of $\overline{X}$ and $Y$
$f_{00}$: support of $\overline{X}$ and $\overline{Y}$

# Drawback of Confidence

| | Coffee | $\overline{\text{Coffee}}$ | |
|---|---|---|---|
| Tea | 15 | 5 | 20 |
| $\overline{\text{Tea}}$ | 75 | 5 | 80 |
| | 90 | 10 | 100 |

Association Rule: Tea $\rightarrow$ Coffee; support?

Confidence= P(Coffee|Tea) = 0.75

but P(Coffee) = 0.9

$\Rightarrow$ Although confidence is high, rule is misleading

$\Rightarrow$ P(Coffee|$\overline{\text{Tea}}$) = 0.9375

$\Rightarrow$ Knowing that a person drinks tea decreases his/her prob. of drinking coffee from 0.9 to 0.75!

$\Rightarrow$ The problem is confidence ignores the support of the rule in the rule consequent -> many people who drink tea also drinks coffee

# Statistical Independence

- Population of 1000 students
  - 600 students know how to swim (S)
  - 700 students know how to bike (B)
  - 420 students know how to swim and bike (S,B)

  - $P(S \wedge B) = 420/1000 = 0.42$
  - $P(S) \times P(B) = 0.6 \times 0.7 = 0.42$

  - $P(S \wedge B) = P(S) \times P(B)$ => Statistical independence
  - $P(S \wedge B) > P(S) \times P(B)$ => Positively correlated
  - $P(S \wedge B) < P(S) \times P(B)$ => Negatively correlated

# Statistical-based Measures

- Measures that take into account statistical dependence

$$Lift = \frac{P(Y \mid X)}{P(Y)} = \frac{P(X,Y)}{P(X)P(Y)}$$

- Lift is also called "*interest*"
- Lift measures how many times more often X and Y occur together than expected if they where statistically independent.

# Example: Lift

| | Coffee | $\overline{\text{Coffee}}$ | |
|---|---|---|---|
| Tea | 15 | 5 | 20 |
| $\overline{\text{Tea}}$ | 75 | 5 | 80 |
| | 90 | 10 | 100 |

Association Rule: Tea $\rightarrow$ Coffee

Confidence= P(Coffee|Tea) = 0.75

but P(Coffee) = 0.9

$\Rightarrow$ Lift = 0.75/0.9= 0.8333 (< 1, therefore is negatively associated)

# Drawback of Lift

| | $Y$ | $\overline{Y}$ | |
|---|---|---|---|
| $X$ | 10 | 0 | 10 |
| $\overline{X}$ | 0 | 90 | 90 |
| | 10 | 90 | 100 |

| | $Y$ | $\overline{Y}$ | |
|---|---|---|---|
| $X$ | 90 | 0 | 90 |
| $\overline{X}$ | 0 | 10 | 10 |
| | 90 | 10 | 100 |

$$Lift = \frac{0.1}{(0.1)(0.1)} = 10$$

$$Lift = \frac{0.9}{(0.9)(0.9)} = 1.11$$

**Remember: Statistical independence => If P(X,Y)=P(X)P(Y) => Lift = 1**

**Confidence might be better in this case! As confidence value for both cases is 1**

# Leverage & Conviction

- **Leverage (X -> Y) :P(X and Y) - (P(X)P(Y))**
  - measures the difference of X and Y appearing together in the data set and what would be expected if X and Y where statistically dependent
  - can be used in a sales setting in order to find out how many more units (items X and Y together) are sold than expected from the independent sells.
  - possible values lie between (-1,1)
- **Conviction (X -> Y) : P(X)P(not Y)/P(X and not Y)=(1-sup(Y))/(1-conf(X -> Y))**
  - compares the probability that X appears without Y if they were dependent with the actual frequency of the appearance of X without Y. In that respect it is similar to lift.
  - possible values lie between (0,+Inf)