Association Model

Agenda

- **Introduction to Association Rules**
- Apriori algorithm
- Frequent pattern growth
- Model Evaluation & Selection
- Association Cases
- Software Demo

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 Transaction

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 (σ)**
  - Frequency of occurrence of an itemset
  - E.g. σ({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}

- 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

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

{Milk,Diaper} $\rightarrow$ Beer

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

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

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

⇒ 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 ≥ minsup

  2. **Rule Generation**

     Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset

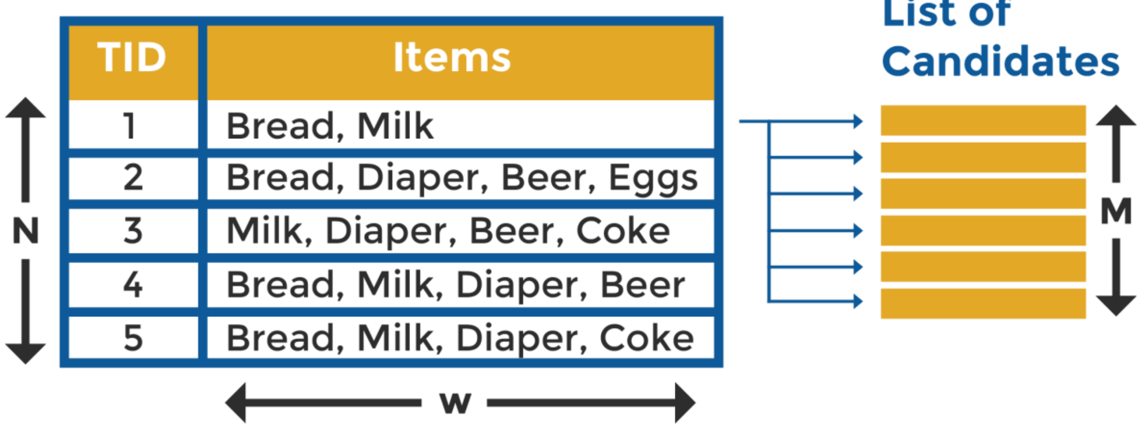
- Frequent itemset generation is still computationally expensive

Frequent Itemset Generation

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

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



  - 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

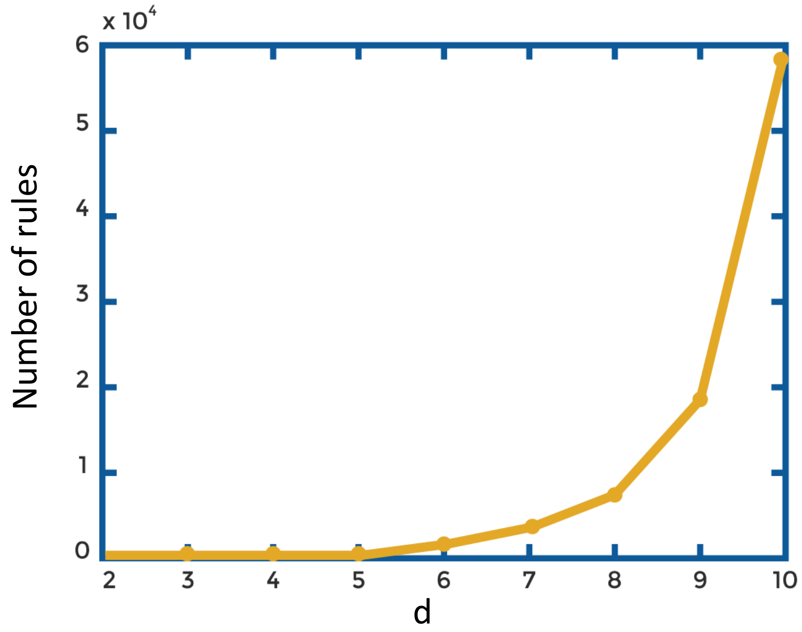
- Reduce the number of transactions (N)
  - Reduce size of N as the size of itemset increases
  - Used by DHP and vertical-based mining algorithms

- 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

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

Agenda

- Introduction to Association Rules
- **Apriori algorithm**
- Frequent pattern growth
- Model Evaluation & Selection
- Association Cases
- Software Demo

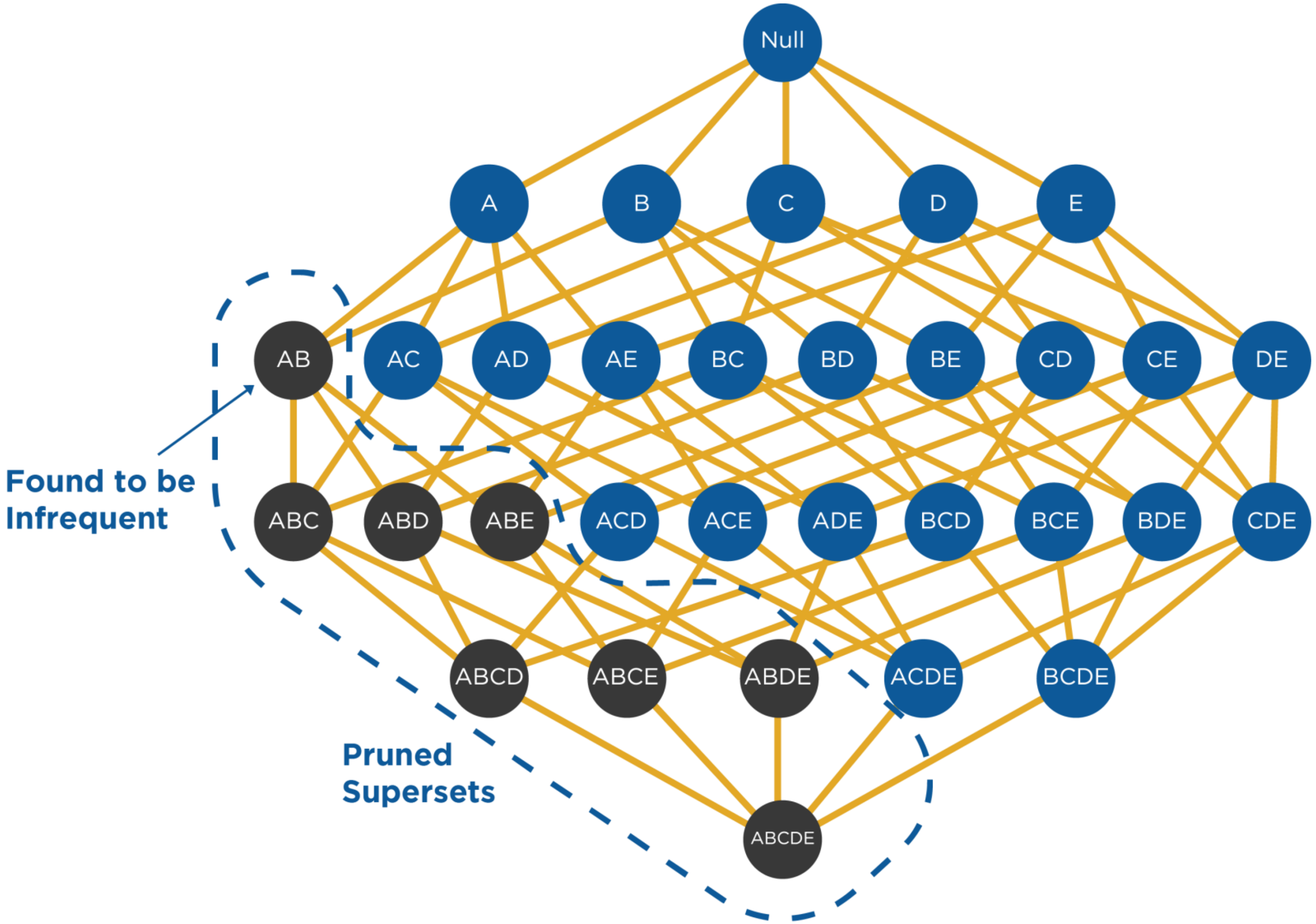
Reducing
Number of Candidates

- 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-Itemset)

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)

Itemset	Count
{Bread,Milk,Diaper}	3

Triplets (3-itemsets)

Minimum Support = 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

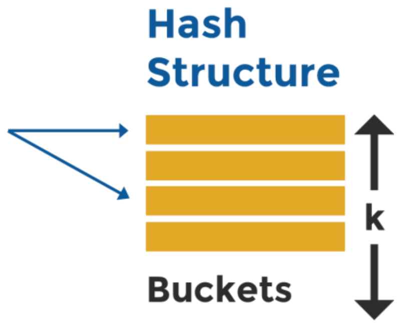
Reducing Number of Comparisons

- Candidate counting:
  - Scan the database of transactions to determine the support of each candidate itemset
  - To reduce the number of comparisons, store the candidates in a hash structure
    - Instead of matching each transaction against every candidate, match it against candidates contained in the hashed buckets

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



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)

Agenda

- Introduction to Association Rules
- Apriori algorithm
- **Frequent pattern growth**
- Model Evaluation & Selection
- Association Cases
- Software Demo

Pattern-Growth Approach:

Mining Frequent Patterns Without Candidate Generation

- Bottlenecks of the Apriori approach
  - Breadth-first (i.e., level-wise) search
  - Candidate generation and test
    - Often generates a huge number of candidates
- The FPGrowth Approach *(J. Han, J. Pei, and Y. Yin, SIGMOD' 00)*
  - Depth-first search
  - Avoid explicit candidate generation
- Major philosophy: Grow long patterns from short ones using local frequent items only
  - "abc" is a frequent pattern
  - Get all transactions having "abc", i.e., project DB on abc: DB|abc
  - "d" is a local frequent item in DB|abc → abcd is a frequent pattern

Alternative Methods
for Frequent Itemset Generation

- Representation of Database
  - horizontal vs vertical data layout

Horizontal Data Layout

TID	Items
1	A,B,E
2	B,C,D
3	C,E
4	A,C,D
5	A,B,C,D
6	A,E
7	A,B
8	A,B,C
9	A,C,D
10	B

Vertical Data Layout

A	B	C	D	E
1	1	2	2	1
4	2	3	4	3
5	5	4	5	6
6	7	8	9	
7	8	9		
8	10			
9				

FP-growth
Algorithm

- Use a compressed representation of the database using an FP-tree

- Once an FP-tree has been constructed, it uses a recursive divide-and-conquer approach to mine the frequent itemsets

Steps in FP-Growth

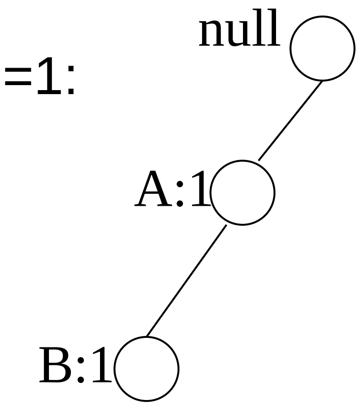
1. Calculate minimum support
2. Find frequency of occurrence
3. Prioritize the items
4. Order the items according to priority
5. Draw FP-Tree
6. Validation
7. Identify frequent item-sets

(an example is provided in Appendix section of this presentation slides)

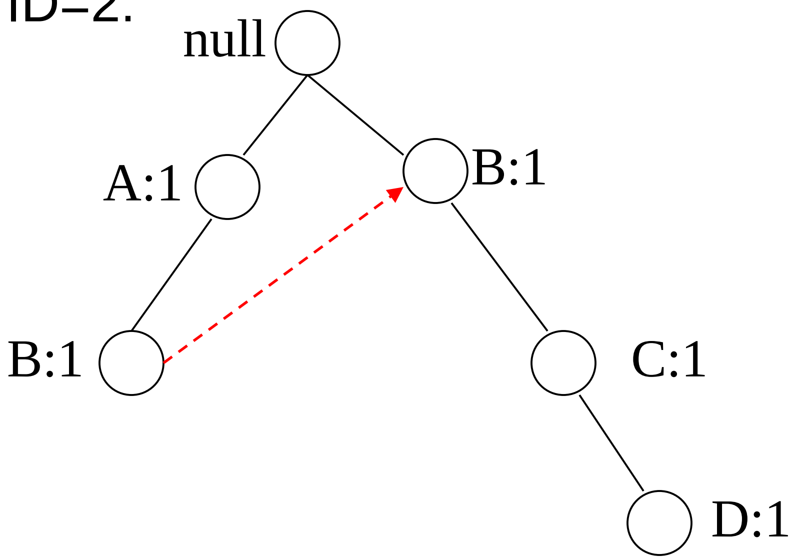
FP-tree Construction

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

After reading TID=1:



After reading TID=2:



FP-tree Construction

1. Scan DB once, find frequent 1-itemset (single item pattern)
2. Sort frequent items in frequency descending order, f-list
3. Scan DB again, construct FP-tree

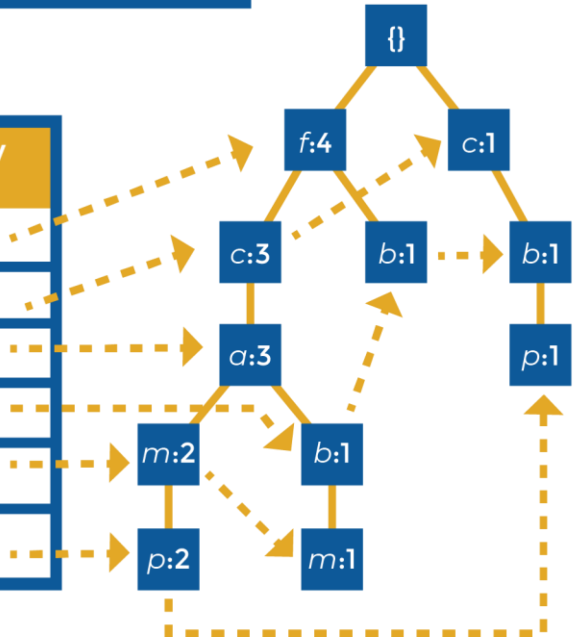
TID	Item Bought	(Ordered) Frequent Items
100	{f,a,c,d,g,i,m,p}	{f,c,a,m,p}
200	{a,b,c,f,l,m,o}	{f,c,a,b,m}
300	{b,f,h,j,o,w}	{f,b}
400	{b,c,k,s,p}	{c,b,p}
500	{a,f,c,e,l,p,m,n}	{f,c,a,m,p}

min_support = 3

F-list = f-c-a-b-m-p

Header Table

Item	Frequency Head
f	4
c	4
a	3
b	3
m	3
p	3



FP-growth

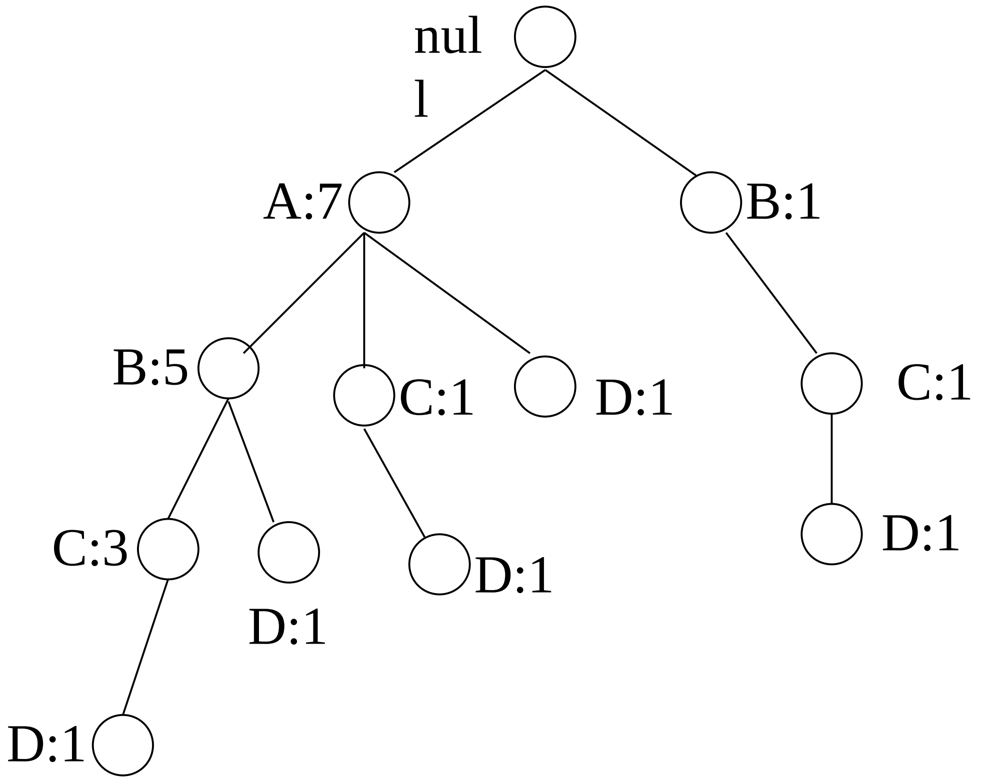
- Conditional Pattern base for D:

  P = {(A:1,B:1,C:1),
  (A:1,B:1),
  (A:1,C:1),
  (A:1),
  (B:1,C:1)}

- Recursively apply FP-growth on P

- Frequent Itemsets found (with sup > 1):

  AD, BD, CD, ACD, BCD



Benefits of the FP-tree Structure

- Completeness
  - Preserve complete information for frequent pattern mining
  - Never break a long pattern of any transaction
- Compactness
  - Reduce irrelevant info—infrequent items are gone
  - Items in frequency descending order: the more frequently occurring, the more likely to be shared
  - Never be larger than the original database (not count node-links and the count field)

The Frequent Pattern Growth Mining Method

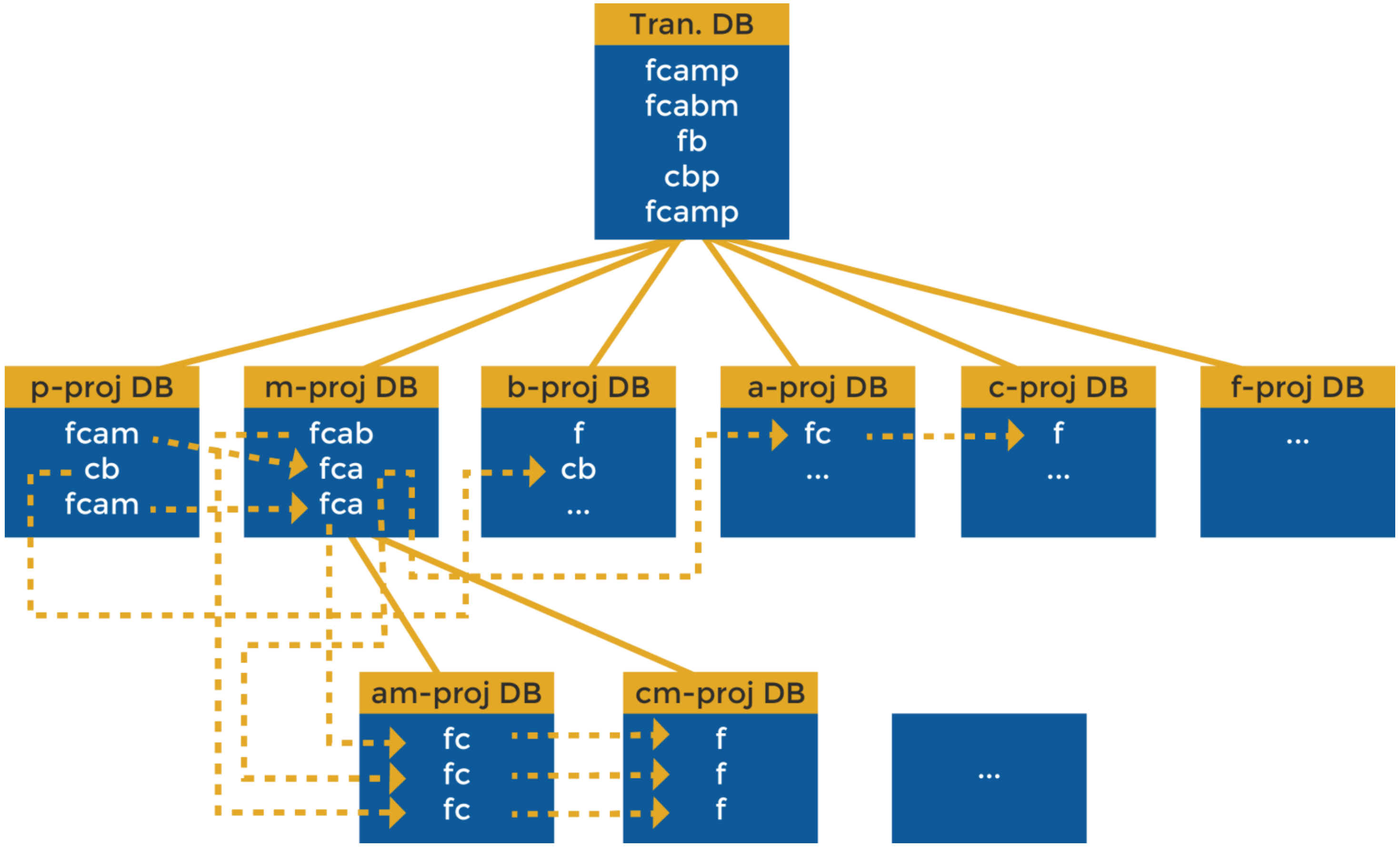
- Idea: Frequent pattern growth
  - Recursively grow frequent patterns by pattern and database partition
- Method
  - For each frequent item, construct its conditional pattern-base, and then its conditional FP-tree
  - Repeat the process on each newly created conditional FP-tree
  - Until the resulting FP-tree is empty, or it contains only one path—single path will generate all the combinations of its sub-paths, each of which is a frequent pattern

Scaling FP-growth
by Database Projection

- What about if FP-tree cannot fit in memory? DB projection
- First partition a database into a set of projected DBs
- Then construct and mine FP-tree for each projected DB
- Parallel projection vs. partition projection techniques
  - Parallel projection
    - Project the DB in parallel for each frequent item
    - Parallel projection is space costly
    - All the partitions can be processed in parallel
  - Partition projection
    - Partition the DB based on the ordered frequent items
    - Passing the unprocessed parts to the subsequent partitions

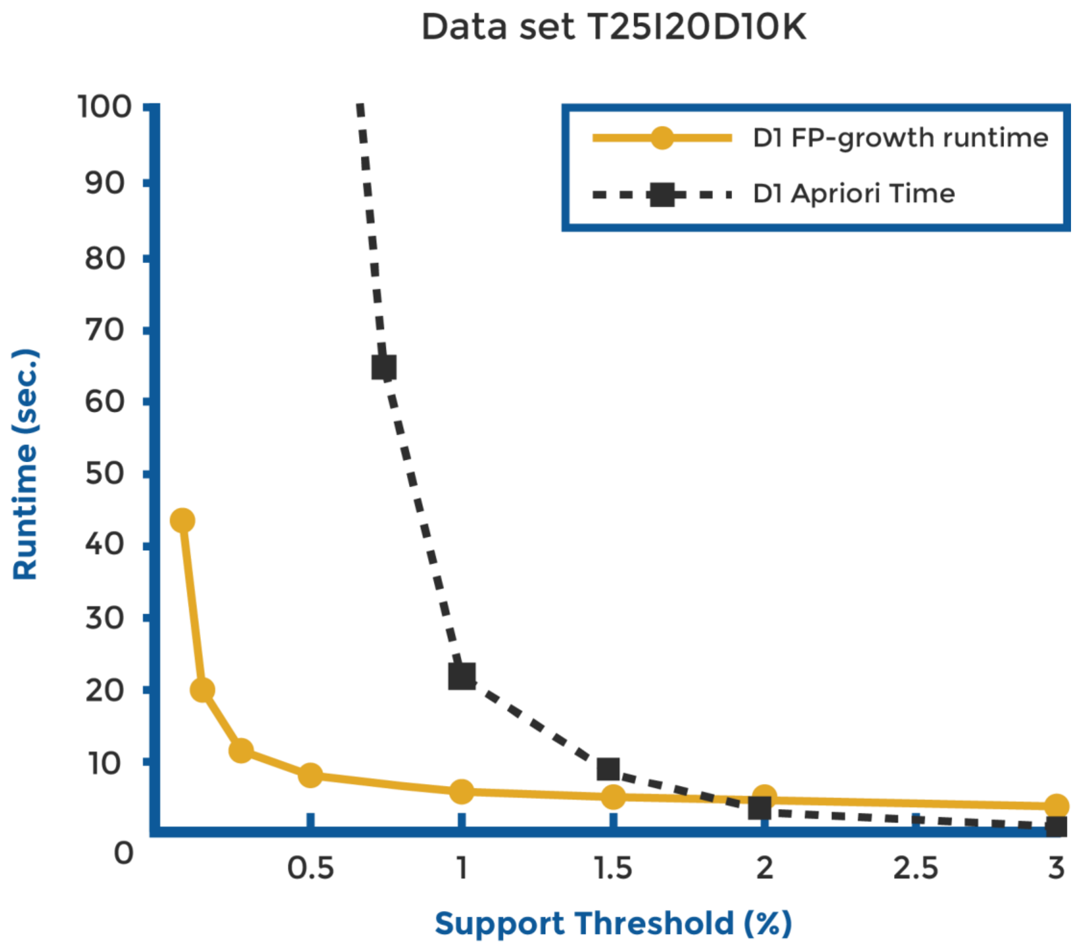
Partition-Based Projection

- **Parallel projection** needs a lot of disk space
- **Partition projection** saves it



Association Model

FP-Growth vs. Apriori:
Scalability With the Support Threshold



Advantages of the Pattern Growth Approach

- Divide-and-conquer
  - Decompose both the mining task and DB according to the frequent patterns obtained so far
  - Lead to focused search of smaller databases
- Other factors
  - No candidate generation, no candidate test
  - Compressed database: FP-tree structure
  - No repeated scan of entire database
  - Basic ops: counting local freq items and building sub FP-tree, no pattern search and matching
- A good open-source implementation and refinement of FPGrowth
  - FPGrowth+ (Grahne and J. Zhu, FIMI'03)

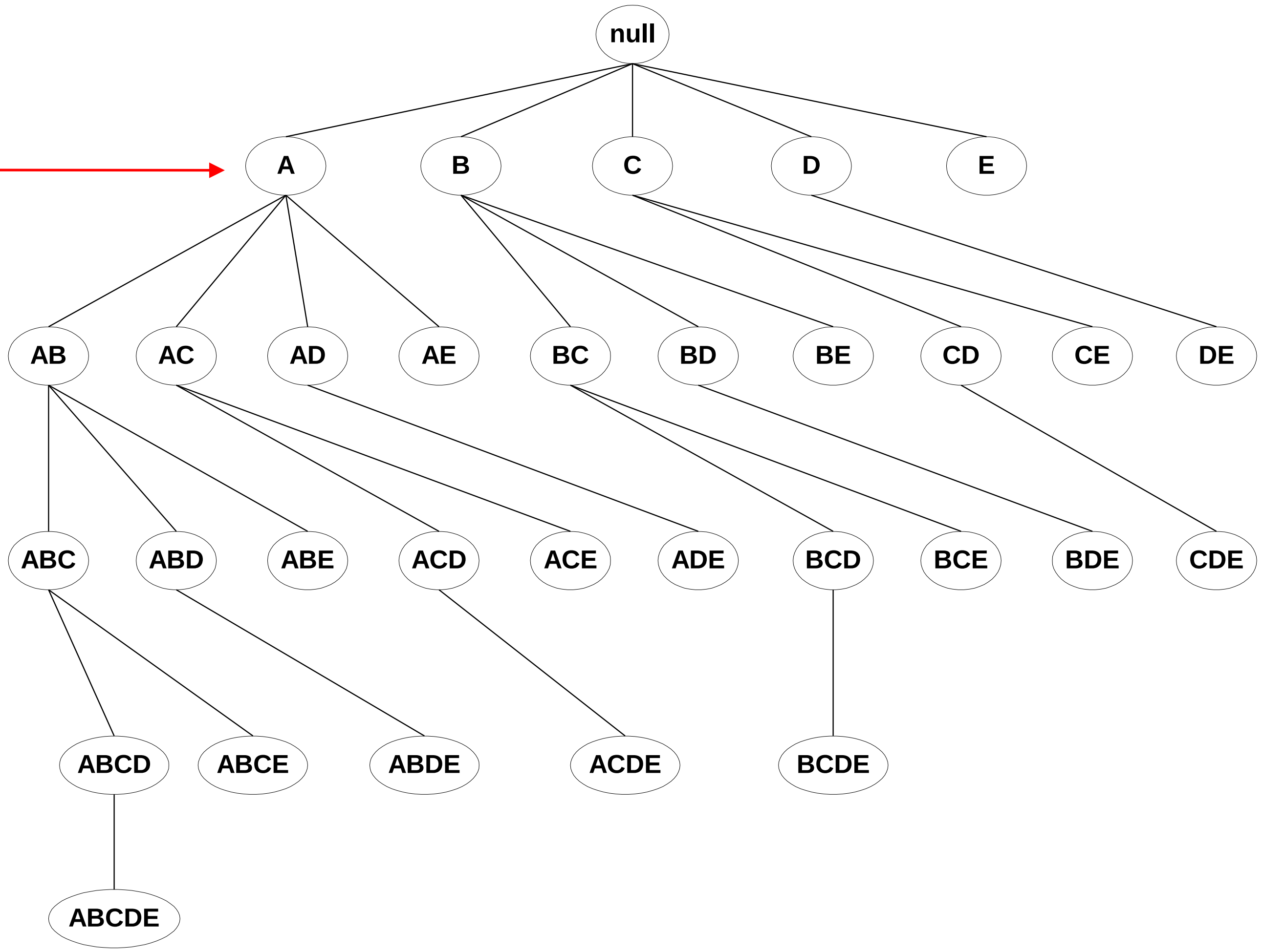
Tree Projection

- Items are listed in lexicographic order
- Each node P stores the following information:
  - Itemset for node P
  - List of possible lexicographic extensions of P: E(P)
  - Pointer to projected database of its ancestor node
  - Bitvector containing information about which transactions in the projected database contain the itemset

Tree Projection

Set enumeration tree:

Possible Extension: E(A) = {B,C,D,E}



Possible Extension: E(ABC) = {D,E}

Projected Database

Original Database:

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,C,D,E}
4	{A,D,E}
5	{A,B,C}
6	{A,B,C,D}
7	{B,C}
8	{A,B,C}
9	{A,B,D}
10	{B,C,E}

Projected Database for node A:

TID	Items
1	{B}
2	{}
3	{C,D,E}
4	{D,E}
5	{B,C}
6	{B,C,D}
7	{}
8	{B,C}
9	{B,D}
10	{}

For each transaction T, projected transaction at node A is $T \cap E(A)$

ECLAT

Equivalence Class Transformation

- For each item, store a list of transaction ids (tids)

Horizontal
Data Layout

TID	Items
1	A,B,E
2	B,C,D
3	C,E
4	A,C,D
5	A,B,C,D
6	A,E
7	A,B
8	A,B,C
9	A,C,D
10	B

Vertical Data Layout

A	B	C	D	E
1	1	2	2	1
4	2	3	4	3
5	5	4	5	6
6	7	8	9	
7	8	9		
8	10			
9				

TID-list

ECLAT

- Determine support of any k-itemset by intersecting tid-lists of two of its (k-1) subsets.

A
1
4
5
6
7
8
9

$\wedge$

B
1
2
5
7
8
10

$\rightarrow$

AB
1
5
7
8

- 3 traversal approaches:
  - top-down, bottom-up and hybrid
- Advantage: very fast support counting
- Disadvantage: intermediate tid-lists may become too large for memory

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 $L \to \emptyset$ and $\emptyset \to L$)

Rule Generation

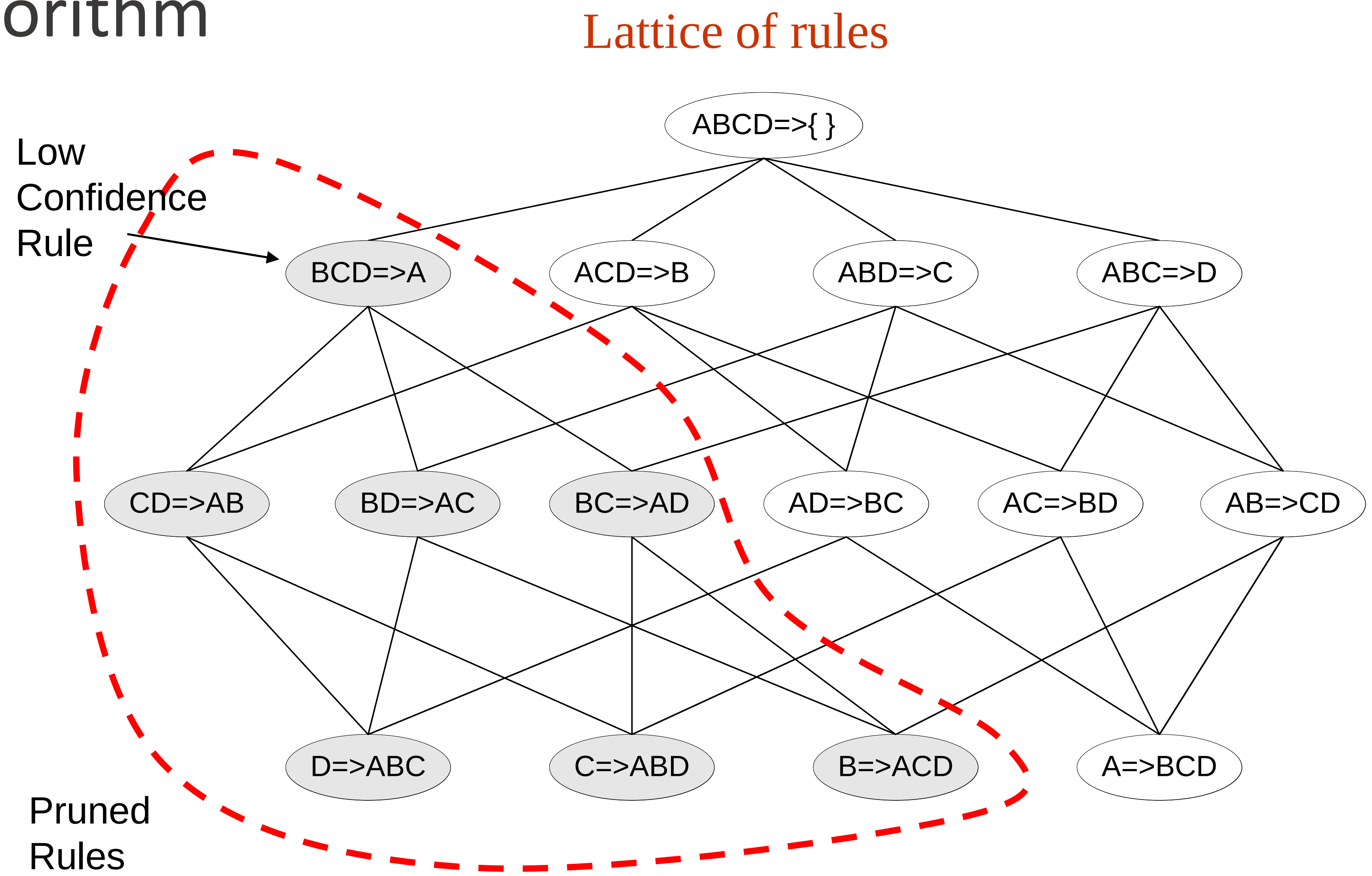
How to efficiently generate rules from frequent itemsets?

- 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}:

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

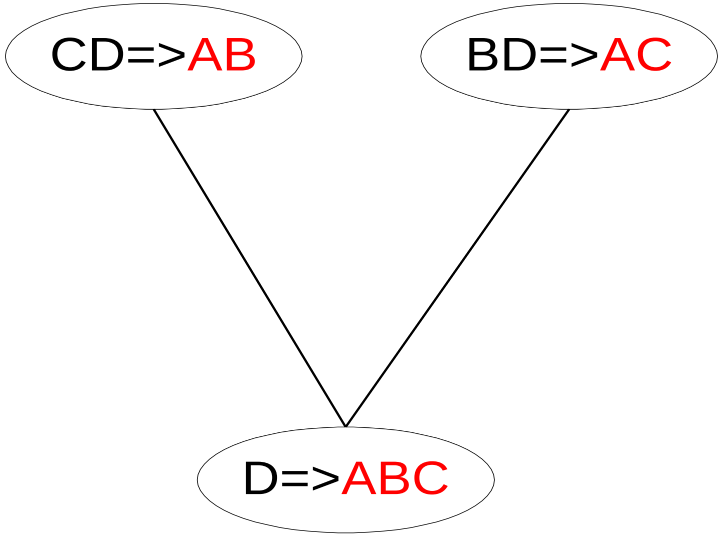
  - Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

Rule Generation
for Apriori Algorithm



Rule Generation
for Apriori Algorithm

- Candidate rule is generated by merging two rules that share the same prefix in the rule consequent
- join(CD=>AB,BD=>AC) would produce the candidate rule D => ABC
- Prune rule D=>ABC if its subset AD=>BC does not have high confidence



Agenda

- Introduction to Association Rules
- Apriori algorithm
- Frequent pattern growth
- **Model Evaluation & Selection**
- Association Cases
- Software Demo

Model Evaluation and Selection

- Support & Confidence

$$\textbf{SUPPORT} = \frac{\text{number of transactions containing X and Y}}{\text{total number of transactions}}$$

$$\textbf{CONFIDENCE} = \frac{\text{number of transactions containing X and Y}}{\text{number of transactions containing X}}$$

SUPPORT: *P(A,B)*

CONFIDENCE: *max(P(B|A), P(A|B))*

Effect of
Support Distribution

- How to set the appropriate minsup threshold?
  - 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

- Using a single minimum support threshold may not be effective

Multiple
Minimum Support

How to apply multiple minimum supports?

- MS(i): minimum support for item i
- e.g.: MS(Milk)=5%, MS(Coke) = 3%,
  MS(Broccoli)=0.1% MS(Salmon)=0.5%
- MS({Milk, Broccoli}) = min (MS(Milk), MS(Broccoli))
  = 0.1%
- Challenge: Support is no longer anti-monotone
  - Suppose: Support(Milk, Coke) = 1.5% and
    Support(Milk, Coke, Broccoli) = 0.5%
  - {Milk,Coke} is infrequent but {Milk,Coke,Broccoli} is frequent

Multiple
Minimum Support (Liu 1999)

How to apply multiple minimum supports?

- MS(i): minimum support for item i
- e.g.:
  - MS(Milk) =5%
  - MS(Coke) = 3%
  - MS(Broccoli) =0.1%
  - MS(Salmon) =0.5%
- MS({Milk, Broccoli}) = min (MS(Milk), MS(Broccoli))
  = 0.1%

Challenge

Support is no longer anti-monotone, Suppose:

- Support(Milk, Coke) = 1.5%
- Support(Milk, Coke, Broccoli) = 0.5%

{Milk,Coke} is infrequent but
{Milk,Coke,Broccoli} is frequent

Multiple
Minimum Support (Liu 1999)

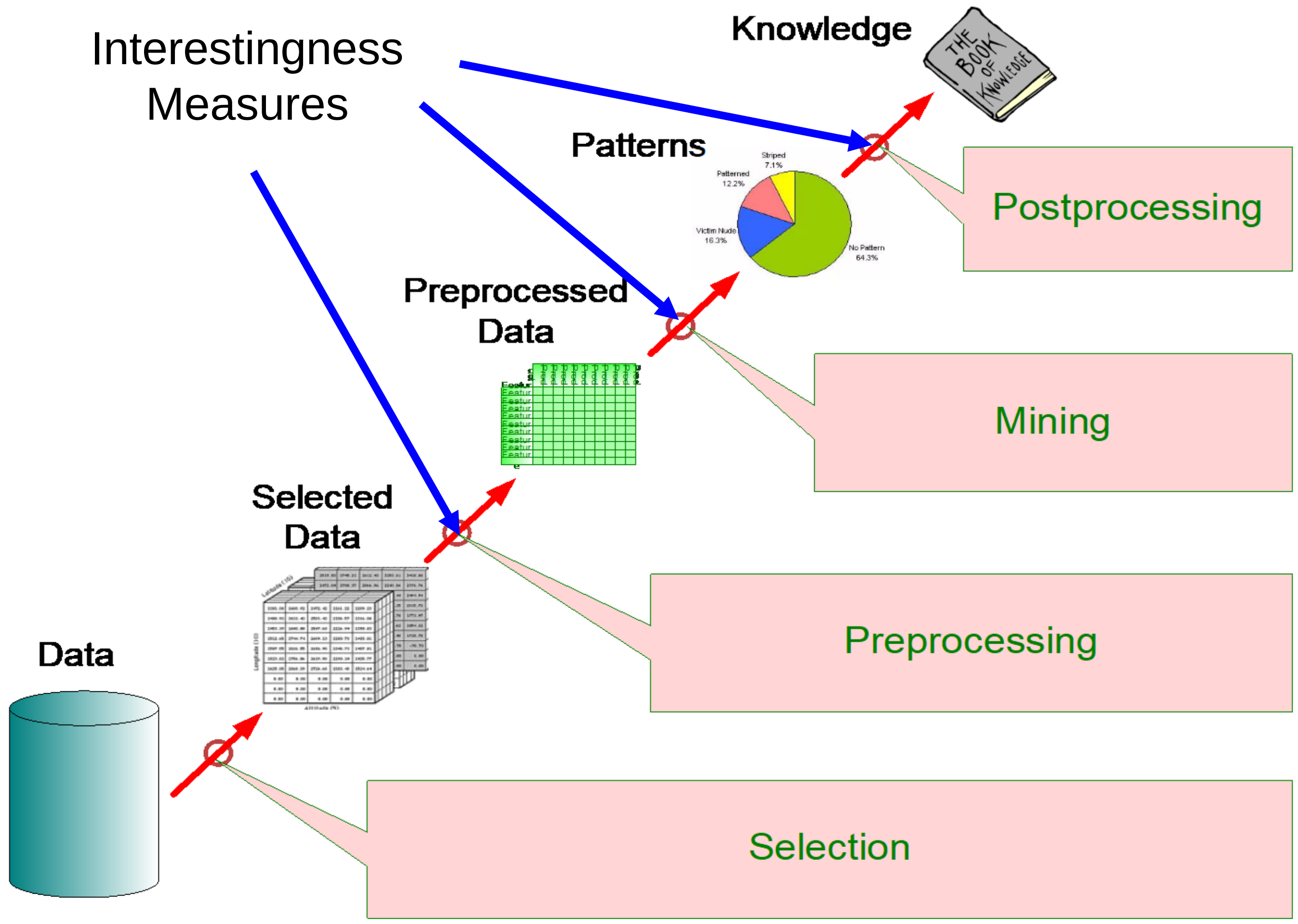
Modifications to Apriori:

- In traditional Apriori,
  - A candidate (k+1)-itemset is generated by merging two frequent item-sets of size k
  - The candidate is pruned if it contains any infrequent subsets of size k
- Pruning step has to be modified:
  - Prune only if subset contains the first item
  - e.g.: Candidate={Broccoli, Coke, Milk} (ordered according to minimum support)
  - {Broccoli, Coke} and {Broccoli, Milk} are frequent but {Coke, Milk} is infrequent
    - Candidate is not pruned because {Coke,Milk} does not contain the first item, i.e., Broccoli.

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

Application of Interestingness Measure



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

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

Used to define various measures

- support, confidence, lift, Gini, J-measure, etc.

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

Confidence= P (Coffee|Tea) = 0.75

but P(Coffee) = 0.9

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

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

Statistical-based Measures

Measures that take into account statistical dependence

$$Lift = \frac{P(Y \mid X)}{P(Y)} or \frac{s(X \cup Y)}{s(X) \times s(Y)} or \frac{\sup(X,Y)}{\sup(X)\sup(Y)}$$

$$Interest = \frac{P(X,Y)}{P(X)P(Y)} or \,|(c(X \cup Y) - s(X)|$$

$$PS = P(X,Y) - P(X)P(Y) = \sup(X,Y) - \sup(X)\sup(Y)$$

$$\phi - coefficient = \frac{P(X,Y) - P(X)P(Y)}{\sqrt{P(X)[1-P(X)]P(Y)[1-P(Y)]}}$$

Definition:

- Lift: the ratio of the observed support to that expected if X and Y were independent. X and Y negatively correlated, if the value is less than 1; otherwise A and B positively correlated.
- Interest: the absolute value of the amount by which the confidence differs from what you would expect, were items selected independently of one another.
- PS (Piatetsky-Shapiro): proportion of additional elements covered by both the premise (X) and consequence (Y) above the expected if independent.
- $\Phi\text{−}coefficient$: Degree of association between X and Y

Example:
Lift / Interest

	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)

There are lots of measures proposed in the literature

Some measures are good for certain applications, but not for others

What criteria should we use to determine whether a measure is good or bad?

What about Apriori-style support based pruning? How does it affect these measures?

#	Measure	Formula
1	ϕ-coefficient	$\frac{P(A,B)-P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
2	Goodman-Kruskal's (λ)	$\frac{\sum_j \max_k P(A_j,B_k)+\sum_k \max_j P(A_j,B_k)-\max_j P(A_j)-\max_k P(B_k)}{2-\max_j P(A_j)-\max_k P(B_k)}$
3	Odds ratio (α)	$\frac{P(A,B)P(\overline{A},\overline{B})}{P(A,\overline{B})P(\overline{A},B)}$
4	Yule's Q	$\frac{P(A,B)P(\overline{AB})-P(A,\overline{B})P(\overline{A},B)}{P(A,B)P(\overline{AB})+P(A,\overline{B})P(\overline{A},B)} = \frac{\alpha-1}{\alpha+1}$
5	Yule's Y	$\frac{\sqrt{P(A,B)P(\overline{AB})}-\sqrt{P(A,\overline{B})P(\overline{A},B)}}{\sqrt{P(A,B)P(\overline{AB})}+\sqrt{P(A,\overline{B})P(\overline{A},B)}} = \frac{\sqrt{\alpha}-1}{\sqrt{\alpha}+1}$
6	Kappa (κ)	$\frac{P(A,B)+P(\overline{A},\overline{B})-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A)P(B)-P(\overline{A})P(\overline{B})}$
7	Mutual Information (M)	$\frac{\sum_i \sum_j P(A_i,B_j)\log\frac{P(A_i,B_j)}{P(A_i)P(B_j)}}{\min(-\sum_i P(A_i)\log P(A_i), -\sum_j P(B_j)\log P(B_j))}$
8	J-Measure (J)	$\max\Big(P(A,B)\log(\frac{P(B\mid A)}{P(B)})+P(A\overline{B})\log(\frac{P(\overline{B}\mid A)}{P(\overline{B})}),$ $P(A,B)\log(\frac{P(A\mid B)}{P(A)})+P(\overline{A}B)\log(\frac{P(\overline{A}\mid B)}{P(\overline{A})})\Big)$
9	Gini index (G)	$\max\Big(P(A)[P(B\mid A)^2+P(\overline{B}\mid A)^2]+P(\overline{A})[P(B\mid\overline{A})^2+P(\overline{B}\mid\overline{A})^2]$ $-P(B)^2-P(\overline{B})^2,$ $P(B)[P(A\mid B)^2+P(\overline{A}\mid B)^2]+P(\overline{B})[P(A\mid\overline{B})^2+P(\overline{A}\mid\overline{B})^2]$ $-P(A)^2-P(\overline{A})^2\Big)$
10	Support (s)	$P(A,B)$
11	Confidence (c)	$\max(P(B\mid A),P(A\mid B))$
12	Laplace (L)	$\max\Big(\frac{NP(A,B)+1}{NP(A)+2},\frac{NP(A,B)+1}{NP(B)+2}\Big)$
13	Conviction (V)	$\max\Big(\frac{P(A)P(\overline{B})}{P(A\overline{B})},\frac{P(B)P(\overline{A})}{P(B\overline{A})}\Big)$
14	Interest (I)	$\frac{P(A,B)}{P(A)P(B)}$
15	cosine (IS)	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
16	Piatetsky-Shapiro's (PS)	$P(A,B)-P(A)P(B)$
17	Certainty factor (F)	$\max\Big(\frac{P(B\mid A)-P(B)}{1-P(B)},\frac{P(A\mid B)-P(A)}{1-P(A)}\Big)$
18	Added Value (AV)	$\max(P(B\mid A)-P(B),P(A\mid B)-P(A))$
19	Collective strength (S)	$\frac{P(A,B)+P(\overline{AB})}{P(A)P(B)+P(\overline{A})P(\overline{B})}\times\frac{1-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A,B)-P(\overline{AB})}$
20	Jaccard (ζ)	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$
21	Klosgen (K)	$\sqrt{P(A,B)}\max(P(B\mid A)-P(B),P(A\mid B)-P(A))$

Subjective
Interestingness Measurement

Objective Measure

- Rank patterns based on statistics computed from data
- e.g., 21 measures of association (support, confidence, Laplace, Gini, mutual information, Jaccard, etc).

Subjective Measure

- Rank patterns according to user's interpretation
  - A pattern is subjectively interesting if it contradicts the expectation of a user (Silberschatz & Tuzhilin)
  - A pattern is subjectively interesting if it is actionable (Silberschatz & Tuzhilin)

Agenda

- Introduction to Association Rules
- Apriori algorithm
- Frequent pattern growth
- Model Evaluation & Selection
- **Association Cases**
- Software Demo

Agenda

- Introduction to Association Rules
- Apriori algorithm
- Frequent pattern growth
- Model Evaluation & Selection
- Association Cases
- **Software Demo**