Association Analysis: Basic Concepts and Algorithms

Dr. Hui Xiong
Rutgers University

THE STATE UNIVERSITY OF NEW JERSEY
RUTGERS

Association Analysis: Basic Concepts and Algorithms

Basic Concepts

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 (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} → {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

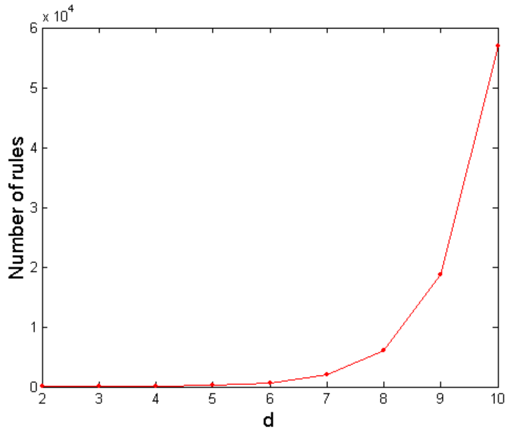
- Given a set of transactions T, the goal of association rule mining is to find all rules having
  - support ≥ *minsup* threshold
  - confidence ≥ *minconf* threshold

- 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

  ⇒ Computationally prohibitive!

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

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} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observations:

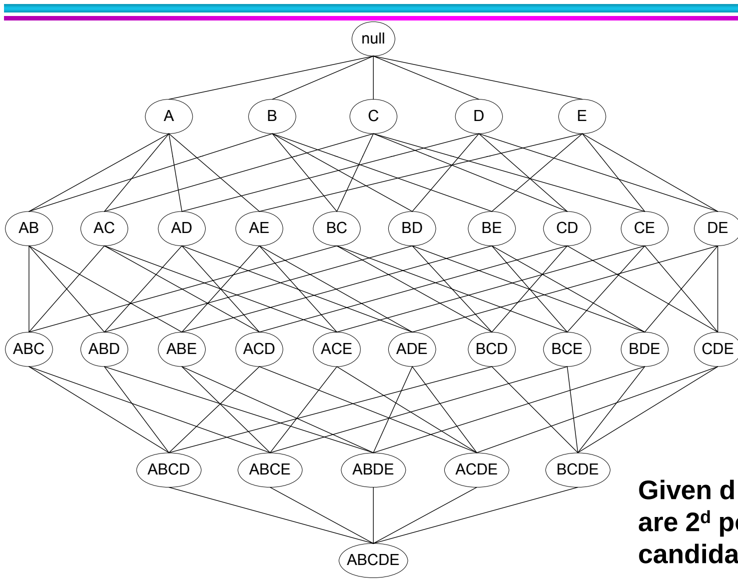
- All the above rules are binary partitions of the same itemset: {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence
- Thus, we may decouple the support and confidence requirements

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



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

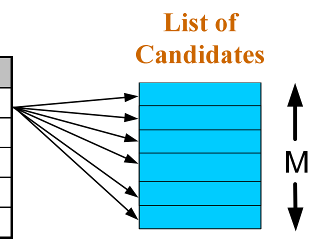
Frequent Itemset Generation

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

Transactions

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

N, w



  – 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

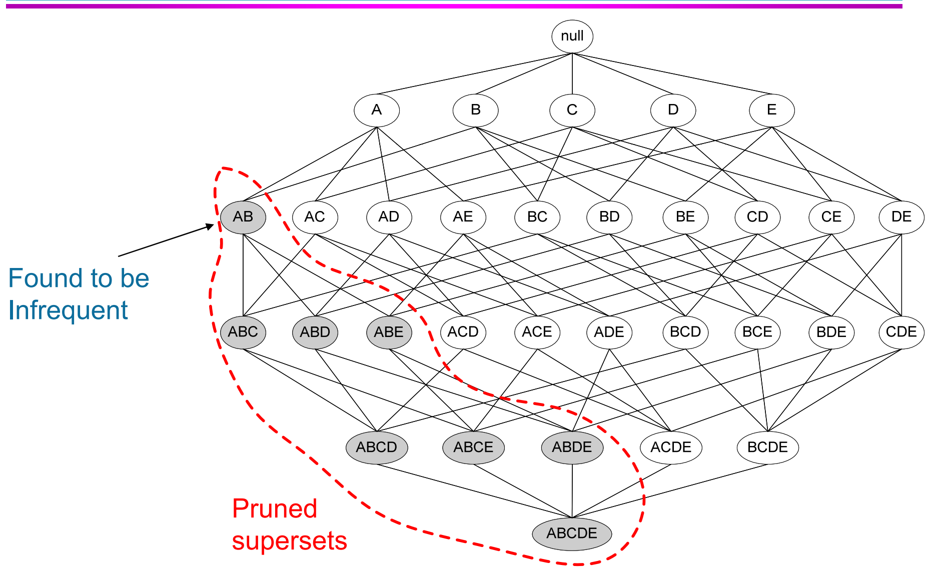
Reducing Number of Candidates

- 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

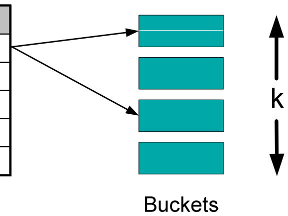
Reducing Number of Comparisons

- 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 / **Hash Structure**

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



Data Mining
Association Analysis: Basic Concepts and Algorithms

Algorithms and Complexity

Factors Affecting Complexity of Apriori

- 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)

Compact Representation of Frequent Itemsets

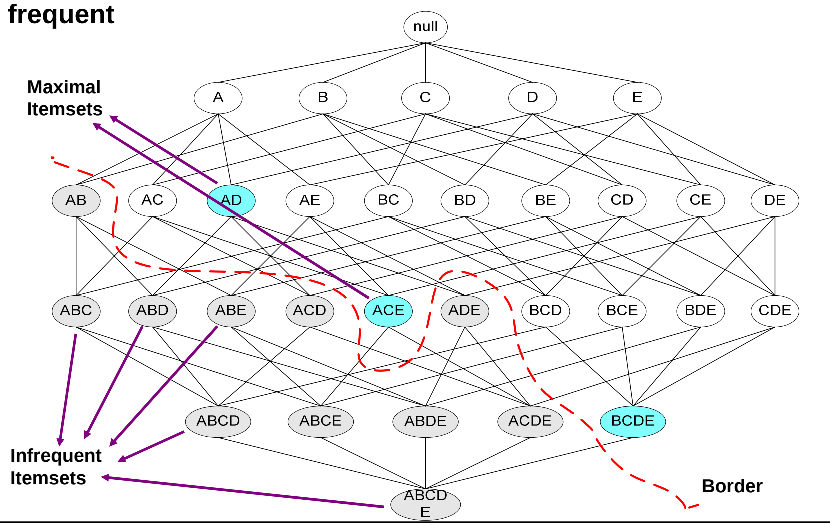
- Some itemsets are redundant because they have identical support as their supersets

TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

- Number of frequent itemsets $= 3 \times \sum_{k=1}^{10} \binom{10}{k}$
- Need a compact representation

Maximal Frequent Itemset

An itemset is maximal frequent if none of its immediate supersets is frequent



Closed Itemset

- An itemset is closed if none of its immediate supersets has the same support as the itemset

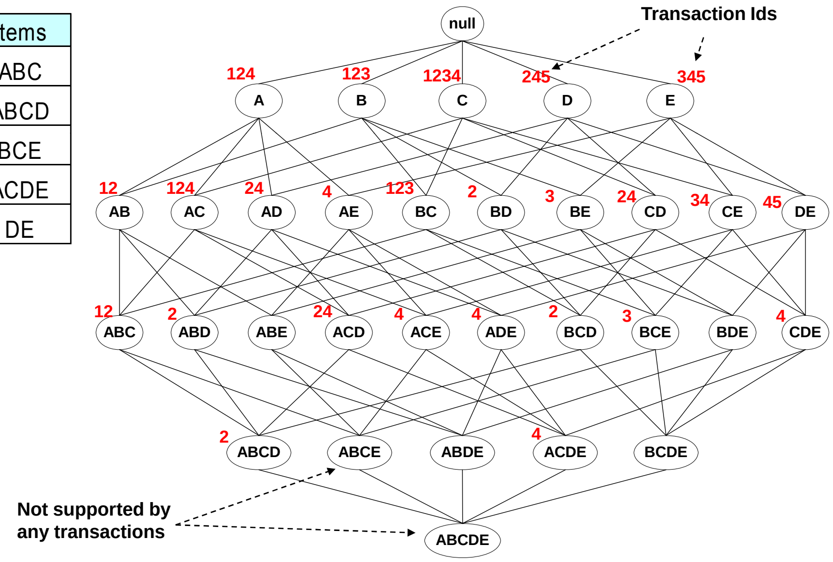
TID	Items
1	{A,B}
2	{B,C,D}
3	{A,B,C,D}
4	{A,B,D}
5	{A,B,C,D}

Itemset	Support
{A}	4
{B}	5
{C}	3
{D}	4
{A,B}	4
{A,C}	2
{A,D}	3
{B,C}	3
{B,D}	4
{C,D}	3

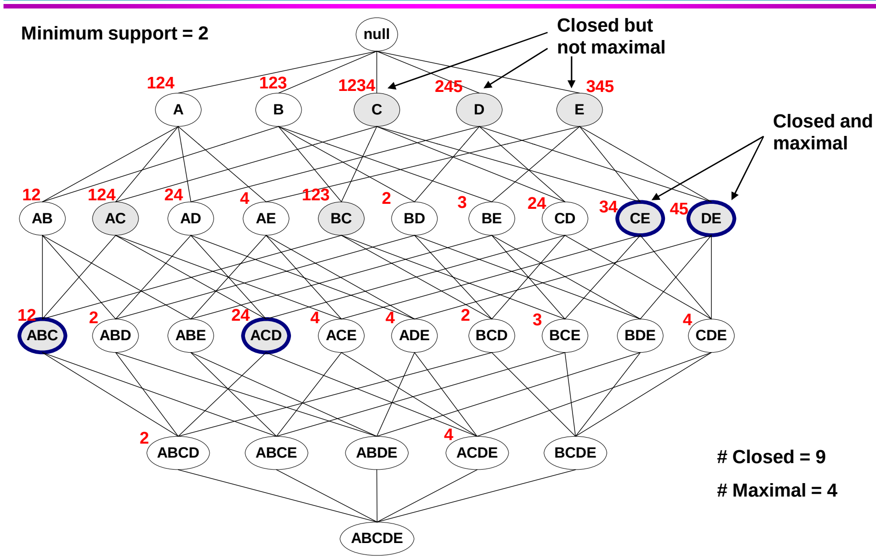
Itemset	Support
{A,B,C}	2
{A,B,D}	3
{A,C,D}	2
{B,C,D}	2
{A,B,C,D}	2

Maximal vs Closed Itemsets

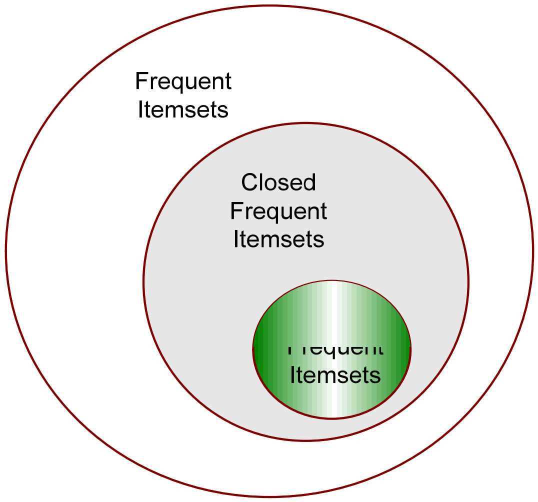
TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE



Maximal vs Closed Frequent Itemsets

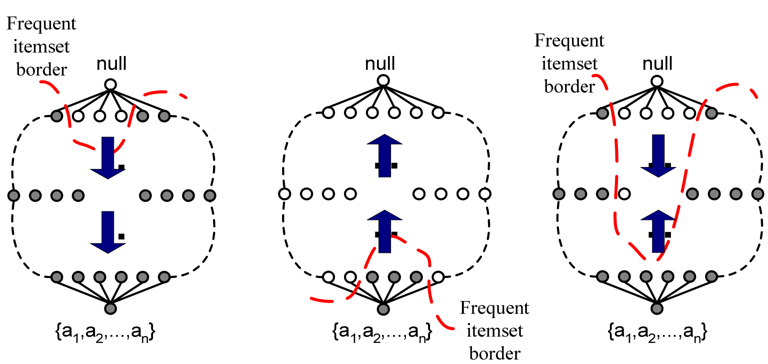


Maximal vs Closed Itemsets



Alternative Methods for Frequent Itemset Generation

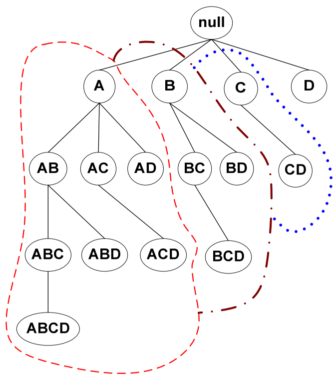
- Traversal of Itemset Lattice
  - General-to-specific vs Specific-to-general



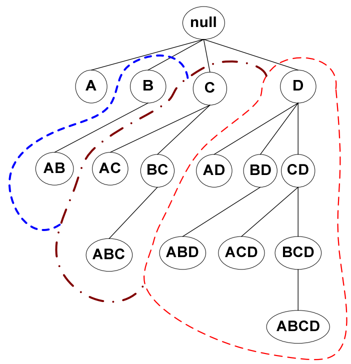
(a) General-to-specific (b) Specific-to-general (c) Bidirectional

Alternative Methods for Frequent Itemset Generation

- Traversal of Itemset Lattice
  - Equivalent Classes



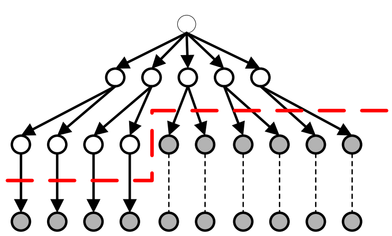
(a) Prefix tree



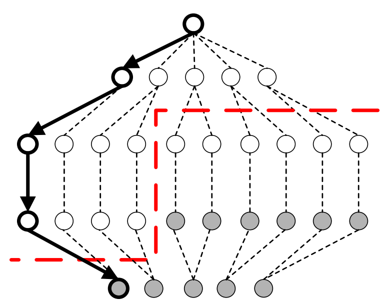
(b) Suffix tree

Alternative Methods for Frequent Itemset Generation

- Traversal of Itemset Lattice
  - Breadth-first vs Depth-first



(a) Breadth first



(b) Depth first

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				

Rule Generation

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

ABC →D,	ABD →C,	ACD →B,	BCD →A,
A →BCD,	B →ACD,	C →ABD,	D →ABC
AB →CD,	AC → BD,	AD → BC,	BC →AD,
BD →AC,	CD →AB,		

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

Rule Generation

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

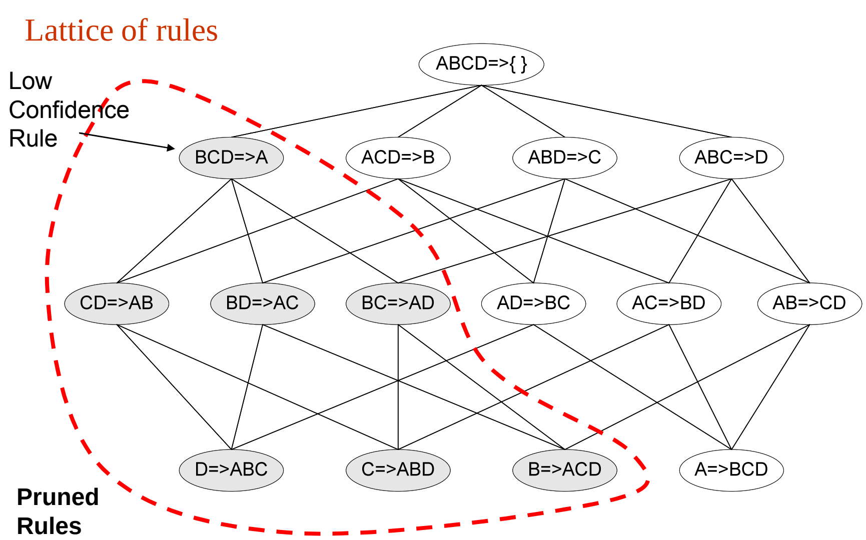
    c(ABC $\rightarrow$D) can be larger or smaller than c(AB $\rightarrow$D)

  - 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

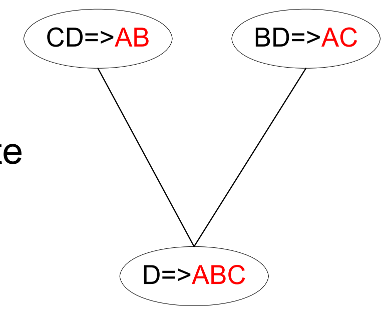


Rule Generation for Apriori Algorithm

- 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



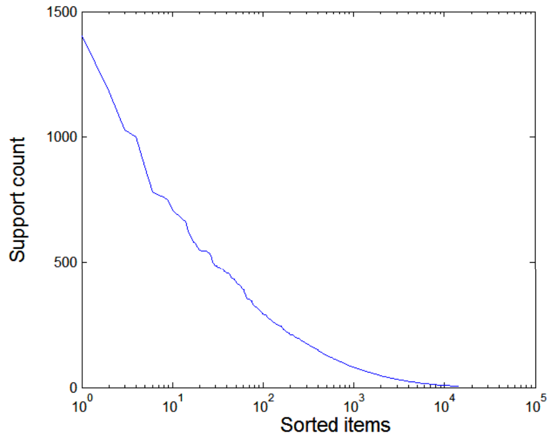
Association Analysis: Basic Concepts and Algorithms

Pattern Evaluation

Effect of Support Distribution

- Many real data sets have skewed support distribution

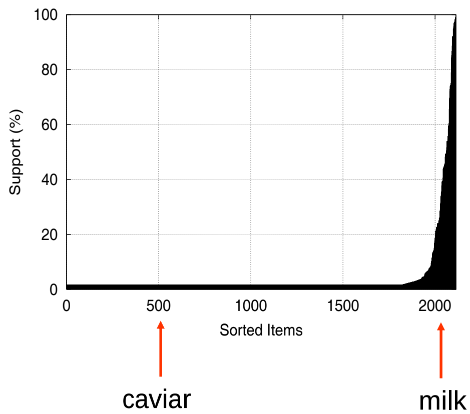
Support distribution of a retail data set



Effect of Support Distribution

- How to set the appropriate *minsup* threshold?
  - If *minsup* is too high, we could miss itemsets involving interesting rare items (e.g., expensive products)
  - If *minsup* is too low, it is computationally expensive and the number of itemsets is very large

Cross-Support Patterns

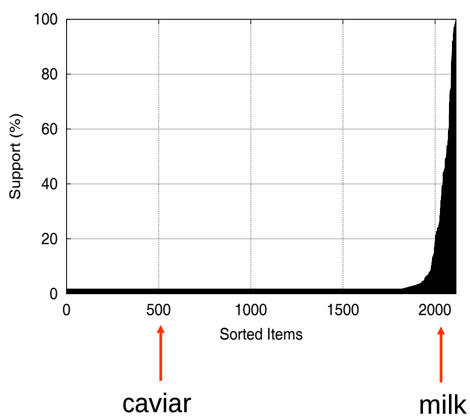


A cross-support pattern involves items with varying degree of support

- Example: {caviar,milk}

How to avoid such patterns?

Cross-Support Patterns



Observation:

Conf(caviar→milk) is very high

but

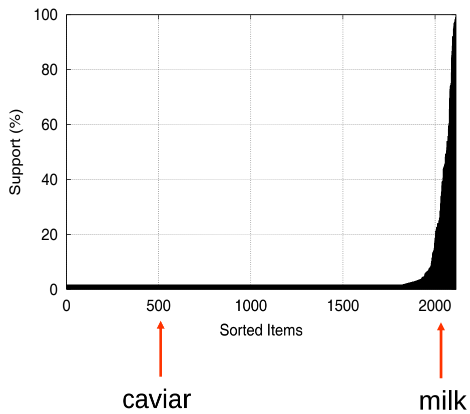
Conf(milk→caviar) is very low

Therefore

min(Conf(caviar→milk),
Conf(milk→caviar))
is also very low

h-Confidence

- h-confidence: $\frac{s(\{i_1, i_2, \cdots, i_k\})}{\max\left[s(i_1), s(i_2), \cdots, s(i_k)\right]}$.



Advantages of h-confidence:

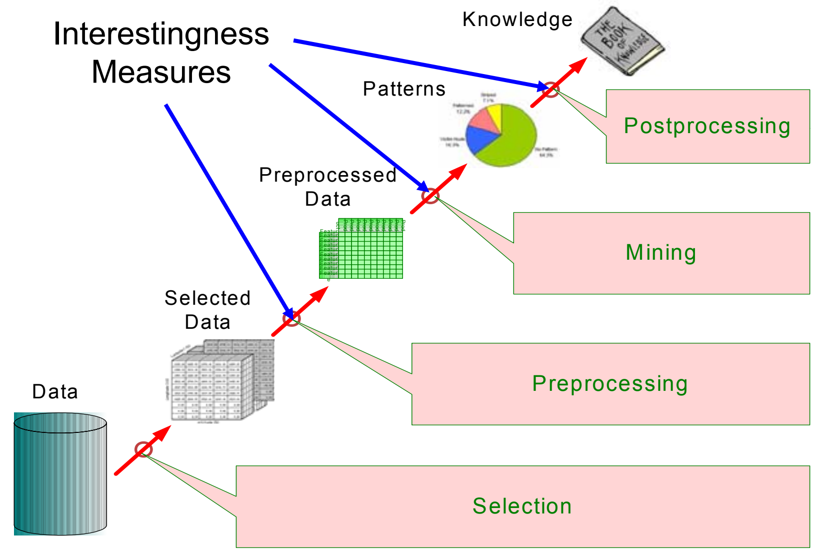
1. Eliminate cross-support patterns such as {caviar,milk}
2. Min function has anti-monotone property
   - Algorithm can be applied to efficiently discover low support, high confidence patterns

Pattern Evaluation

- Association rule algorithms can produce large number of 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 patterns
  - In the original formulation, 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	$\overline{Y}$	
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 $\overline{Y}$

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

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

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

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

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

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

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)

Drawback of Lift & Interest

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

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

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

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

Statistical independence:

If P(X,Y)=P(X)P(Y) => Lift = 1

There are lots of measures proposed in the literature

#	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)	$\max\Big(P(A,B)\log(\frac{P(B\|A)}{P(B)}) + P(A\overline{B})\log(\frac{P(\overline{B}\|A)}{P(\overline{B})}),$ $P(A,B)\log(\frac{P(A\|B)}{P(A)}) + P(\overline{A}B)\log(\frac{P(\overline{A}\|B)}{P(\overline{A})})\Big)$
9	Gini index (G)	$\max\Big(P(A)[P(B\|A)^2 + P(\overline{B}\|A)^2] + P(\overline{A})[P(B\|\overline{A})^2 + P(\overline{B}\|\overline{A})^2]$ $-P(B)^2 - P(\overline{B})^2,$ $P(B)[P(A\|B)^2 + P(\overline{A}\|B)^2] + P(\overline{B})[P(A\|\overline{B})^2 + P(\overline{A}\|\overline{B})^2]$ $-P(A)^2 - P(\overline{A})^2\Big)$
10	Support (s)	$P(A,B)$
11	Confidence (c)	$\max(P(B\|A), P(A\|B))$
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)	$\max\Big(\frac{P(B\|A)-P(B)}{1-P(B)}, \frac{P(A\|B)-P(A)}{1-P(A)}\Big)$
18	Added Value (AV)	$\max(P(B\|A) - P(B), P(A\|B) - P(A))$
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)	$\sqrt{P(A,B)}\max(P(B\|A) - P(B), P(A\|B) - P(A))$

Comparing Different Measures

10 examples of contingency tables:

Example	f_{11}	f_{10}	f_{01}	f_{00}
E1	8123	83	424	1370
E2	8330	2	622	1046
E3	9481	94	127	298
E4	3954	3080	5	2961
E5	2886	1363	1320	4431
E6	1500	2000	500	6000
E7	4000	2000	1000	3000
E8	4000	2000	2000	2000
E9	1720	7121	5	1154
E10	61	2483	4	7452

Rankings of contingency tables using various measures:

#	ϕ	λ	α	Q	Y	κ	M	J	G	s	c	L	V	I	IS	PS	F	AV	S	ζ	K
E1	1	1	3	3	3	1	2	2	1	3	5	5	4	6	2	2	4	6	1	2	5
E2	2	2	1	1	1	2	1	3	2	2	1	1	1	8	3	5	1	8	2	3	6
E3	3	3	4	4	4	3	3	8	7	1	4	4	6	10	1	8	6	10	3	1	10
E4	4	7	2	2	2	5	4	1	3	6	2	2	2	4	4	1	2	3	4	5	1
E5	5	4	8	8	8	4	7	5	4	7	9	9	9	3	6	3	9	4	5	6	3
E6	6	6	7	7	7	7	6	4	6	9	8	8	7	2	8	6	7	2	7	8	2
E7	7	5	9	9	9	6	8	6	5	4	7	7	8	5	5	4	8	5	6	4	4
E8	8	9	10	10	10	8	10	10	8	4	10	10	10	9	7	7	10	9	8	7	9
E9	9	9	5	5	5	9	9	7	9	8	3	3	3	7	9	9	3	7	9	9	8
E10	10	8	6	6	6	10	5	9	10	10	6	6	5	1	10	10	5	1	10	10	7

Property under Variable Permutation

	B	$\overline{B}$
A	p	q
$\overline{A}$	r	s

	A	$\overline{A}$
B	p	r
$\overline{B}$	q	s

Does M(A,B) = M(B,A)?

Symmetric measures:

- support, lift, collective strength, cosine, Jaccard, etc

Asymmetric measures:

- confidence, conviction, Laplace, J-measure, etc

Property under Row / Column Scaling

Grade-Gender Example (Mosteller, 1968):

	Male	Female	
High	2	3	5
Low	1	4	5
	3	7	10

	Male	Female	
High	4	30	34
Low	2	40	42
	6	70	76
	2x	10x	

Mosteller:

Underlying association should be independent of the relative number of male and female students in the samples

Property under Inversion Operation

	A	B	C	D	E	F
Transaction 1	1	0	0	1	0	0
■	0	0	1	1	1	0
■	0	0	1	1	1	0
■	0	0	1	1	1	0
■	0	1	1	0	1	1
■	0	0	1	1	1	0
	0	0	1	1	1	0
	0	0	1	1	1	0
	0	0	1	1	1	0
Transaction N	1	0	0	1	0	0
	(a)		(b)		(c)	

Example: ϕ-Coefficient

- ϕ-coefficient is analogous to correlation coefficient for continuous variables

	Y	$\overline{Y}$	
X	60	10	70
$\overline{X}$	10	20	30
	70	30	100

	Y	$\overline{Y}$	
X	20	10	30
$\overline{X}$	10	60	70
	30	70	100

$$\phi = \frac{0.6 - 0.7 \times 0.7}{\sqrt{0.7 \times 0.3 \times 0.7 \times 0.3}} = 0.5238$$

$$\phi = \frac{0.2 - 0.3 \times 0.3}{\sqrt{0.7 \times 0.3 \times 0.7 \times 0.3}} = 0.5238$$

ϕ Coefficient is the same for both tables

Property under Null Addition

	B	$\overline{\mathbf{B}}$
A	p	q
$\overline{\mathbf{A}}$	r	s

$\Rightarrow$

	B	$\overline{\mathbf{B}}$
A	p	q
$\overline{\mathbf{A}}$	r	s + k

Invariant measures:

- support, cosine, Jaccard, etc

Non-invariant measures:

- correlation, Gini, mutual information, odds ratio, etc

Different Measures have Different Properties

Symbol	Measure	Inversion	Null Addition	Scaling
ϕ	ϕ-coefficient	Yes	No	No
α	odds ratio	Yes	No	Yes
κ	Cohen's	Yes	No	No
I	Interest	No	No	No
IS	Cosine	No	Yes	No
PS	Piatetsky-Shapiro's	Yes	No	No
S	Collective strength	Yes	No	No
ζ	Jaccard	No	Yes	No
h	All-confidence	No	No	No
s	Support	No	No	No

Simpson's Paradox

Buy HDTV	Buy Exercise Machine		
	Yes	No	
Yes	99	81	180
No	54	66	120
	153	147	300

$$c(\{\text{HDTV} = \text{Yes}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 99/180 = 55\%$$
$$c(\{\text{HDTV} = \text{No}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 54/120 = 45\%$$

=> Customers who buy HDTV are more likely to buy exercise machines

Simpson's Paradox

Customer Group	Buy HDTV	Buy Exercise Machine Yes	Buy Exercise Machine No	Total
College Students	Yes	1	9	10
	No	4	30	34
Working Adult	Yes	98	72	170
	No	50	36	86

College students:

$$c(\{\text{HDTV} = \text{Yes}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 1/10 = 10\%$$

$$c(\{\text{HDTV} = \text{No}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 4/34 = 11.8\%$$

Working adults:

$$c(\{\text{HDTV} = \text{Yes}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 98/170 = 57.7\%$$

$$c(\{\text{HDTV} = \text{No}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 50/86 = 58.1\%$$

Simpson's Paradox

- Observed relationship in data may be influenced by the presence of other confounding factors (hidden variables)
  – Hidden variables may cause the observed relationship to disappear or reverse its direction!

- Proper stratification is needed to avoid generating spurious patterns

Extensions of Association Analysis

Dr. Hui Xiong
Rutgers University

THE STATE UNIVERSITY OF NEW JERSEY
RUTGERS

Association Analysis: Advanced Concepts

Extensions of Association Analysis to Continuous and Categorical Attributes and Multi-level Rules

Continuous and Categorical Attributes

How to apply association analysis to non-asymmetric binary variables?

Gender	...	Age	Annual Income	No of hours spent online per week	No of email accounts	Privacy Concern
Female	...	26	90K	20	4	Yes
Male	...	51	135K	10	2	No
Male	...	29	80K	10	3	Yes
Female	...	45	120K	15	3	Yes
Female	...	31	95K	20	5	Yes
Male	...	25	55K	25	5	Yes
Male	...	37	100K	10	1	No
Male	...	41	65K	8	2	No
Female	...	26	85K	12	1	No
...	...	...	...	...	...	...

Example of Association Rule:

{Gender=Male, Age $\in$ [21,30)} $\rightarrow$ {No of hours online $\geq$ 10}

Handling Categorical Attributes

- Example: Internet Usage Data

Gender	Level of Education	State	Computer at Home	Online Auction	Chat Online	Online Banking	Privacy Concerns
Female	Graduate	Illinois	Yes	Yes	Daily	Yes	Yes
Male	College	California	No	No	Never	No	No
Male	Graduate	Michigan	Yes	Yes	Monthly	Yes	Yes
Female	College	Virginia	No	Yes	Never	Yes	Yes
Female	Graduate	California	Yes	No	Never	No	Yes
Male	College	Minnesota	Yes	Yes	Weekly	Yes	Yes
Male	College	Alaska	Yes	Yes	Daily	Yes	No
Male	High School	Oregon	Yes	No	Never	No	No
Female	Graduate	Texas	No	No	Monthly	No	No
...	...	...	...	...	...	...	...

{Level of Education=Graduate, Online Banking=Yes}
$\rightarrow$ {Privacy Concerns = Yes}

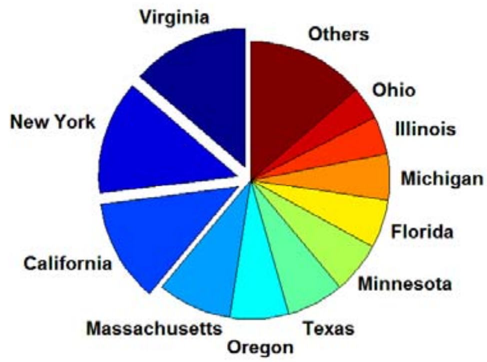
Handling Categorical Attributes

- Introduce a new "item" for each distinct attribute-value pair

Male	Female	Education = Graduate	Education = College	Education = High School	...	Privacy = Yes	Privacy = No
0	1	1	0	0	...	1	0
1	0	0	1	0	...	0	1
1	0	1	0	0	...	1	0
0	1	0	1	0	...	1	0
0	1	1	0	0	...	1	0
1	0	0	1	0	...	1	0
1	0	0	0	0	...	0	1
1	0	0	0	1	...	0	1
0	1	1	0	0	...	0	1
...	...	...	...	...	...	...	...

Handling Categorical Attributes

- Some attributes can have many possible values
  - Many of their attribute values have very low support
    - Potential solution: Aggregate the low-support attribute values



Handling Categorical Attributes

- Distribution of attribute values can be highly skewed
  - Example: 85% of survey participants own a computer at home
    - Most records have Computer at home = Yes
    - Computation becomes expensive; many frequent itemsets involving the binary item (Computer at home = Yes)
    - Potential solution:
      - discard the highly frequent items
      - Use alternative measures such as h-confidence
- Computational Complexity
  - Binarizing the data increases the number of items
  - But the width of the "transactions" remain the same as the number of original (non-binarized) attributes
  - Produce more frequent itemsets but maximum size of frequent itemset is limited to the number of original attributes

Handling Continuous Attributes

- Different methods:
  - Discretization-based
  - Statistics-based
  - Non-discretization based
    - minApriori
- Different kinds of rules can be produced:
  - {Age$\in$[21,30), No of hours online$\in$[10,20)} $\rightarrow$ {Chat Online =Yes}
  - {Age$\in$[21,30), Chat Online = Yes} $\rightarrow$ No of hours online: μ=14, σ=4

Discretization-based Methods

Gender	···	Age	Annual Income	No of hours spent online per week	No of email accounts	Privacy Concern
Female	···	26	90K	20	4	Yes
Male	···	51	135K	10	2	No
Male	···	29	80K	10	3	Yes
Female	···	45	120K	15	3	Yes
Female	···	31	95K	20	5	Yes
Male	···	25	55K	25	5	Yes
Male	···	37	100K	10	1	No
Male	···	41	65K	8	2	No
Female	···	26	85K	12	1	No
···	···	···	···	···	···	···

Male	Female	··· ···	Age < 13	Age $\in [13, 21)$	Age $\in [21, 30)$	··· ···	Privacy = Yes	Privacy = No
0	1	···	0	0	1	···	1	0
1	0	···	0	0	0	···	0	1
1	0	···	0	0	1	···	1	0
0	1	···	0	0	0	···	1	0
0	1	···	0	0	0	···	1	0
1	0	···	0	0	1	···	1	0
1	0	···	0	0	0	···	0	1
1	0	···	0	0	0	···	0	1
0	1	···	0	0	1	···	0	1
···	···	···	···	···	···	···	···	···

Discretization-based Methods

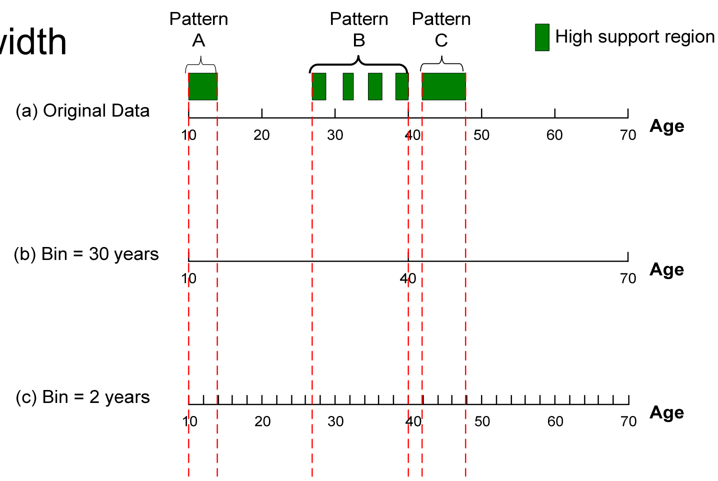
- Unsupervised:
  - Equal-width binning
  - Equal-depth binning
  - Cluster-based
- Supervised discretization

Continuous attribute, v

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
Chat Online = Yes	0	0	20	10	20	0	0	0	0
Chat Online = No	150	100	0	0	0	100	100	150	100
	bin_1		bin_2			bin_3			

Discretization Issues

- Interval width



Pattern A: Age $\in [10, 15) \longrightarrow$ Chat Online = Never
Pattern B: Age $\in [26, 41) \longrightarrow$ Chat Online = Never
Pattern C: Age $\in [42, 48) \longrightarrow$ Online Banking = Yes

Discretization Issues

- Interval too wide (e.g., Bin size= 30)
  - May merge several disparate patterns
    - Patterns A and B are merged together
  - May lose some of the interesting patterns
    - Pattern C may not have enough confidence
- Interval too narrow (e.g., Bin size = 2)
  - Pattern A is broken up into two smaller patterns
    - Can recover the pattern by merging adjacent subpatterns
  - Pattern B is broken up into smaller patterns
    - Cannot recover the pattern by merging adjacent subpatterns
- Potential solution: use all possible intervals
  - Start with narrow intervals
  - Consider all possible mergings of adjacent intervals

Discretization Issues

- Execution time
  - If the range is partitioned into k intervals, there are $O(k^2)$ new items
  - If an interval [a,b) is frequent, then all intervals that subsume [a,b) must also be frequent
    - E.g.: if {Age $\in$[21,25), Chat Online=Yes} is frequent,
      then {Age $\in$[10,50), Chat Online=Yes} is also frequent
  - Improve efficiency:
    - Use maximum support to avoid intervals that are too wide
    - Modify algorithm to exclude candidate itemsets containing more than one intervals of the same attribute

Discretization Issues

- Redundant rules

R1: {Age $\in$[18,20), Age $\in$[10,12)} $\rightarrow$ {Chat Online=Yes}

R2: {Age $\in$[18,23), Age $\in$[10,20)} $\rightarrow$ {Chat Online=Yes}

  - If both rules have the same support and confidence, prune the more specific rule (R1)

Statistics-based Methods

- Example:

  {Income > 100K, Online Banking=Yes} $\rightarrow$ Age: μ=34

- Rule consequent consists of a continuous variable, characterized by their statistics
  - mean, median, standard deviation, etc.
- Approach:
  - Withhold the target attribute from the rest of the data
  - Extract frequent itemsets from the rest of the attributes
    - Binarized the continuous attributes (except for the target attribute)
  - For each frequent itemset, compute the corresponding descriptive statistics of the target attribute
    - Frequent itemset becomes a rule by introducing the target variable as rule consequent
  - Apply statistical test to determine interestingness of the rule

Statistics-based Methods

Gender	...	Age	Annual Income	No of hours spent online per week	No of email accounts	Privacy Concern
Female	...	26	90K	20	4	Yes
Male	...	51	135K	10	2	No
Male	...	29	80K	10	3	Yes
Female	...	45	120K	15	3	Yes
Female	...	31	95K	20	5	Yes
Male	...	25	55K	25	5	Yes
Male	...	37	100K	10	1	No
Male	...	41	65K	8	2	No
Female	...	26	85K	12	1	No
...	...	...	...	...	...	...

Frequent Itemsets:

{Male, Income > 100K}

{Income < 30K, No hours $\in$[10,15)}

{Income > 100K, Online Banking = Yes}

....

Association Rules:

{Male, Income > 100K} $\rightarrow$ Age: μ = 30

{Income < 40K, No hours $\in$[10,15)} $\rightarrow$ Age: μ = 24

{Income > 100K,Online Banking = Yes} $\rightarrow$ Age: μ = 34

....

Statistics-based Methods

- How to determine whether an association rule interesting?
  - Compare the statistics for segment of population covered by the rule vs segment of population not covered by the rule:

    $A \Rightarrow B: \mu$ versus $\bar{A} \Rightarrow B: \mu'$

  - Statistical hypothesis testing:

    $$Z = \frac{\mu' - \mu - \Delta}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

    - Null hypothesis: H0: $\mu' = \mu + \Delta$
    - Alternative hypothesis: H1: $\mu' > \mu + \Delta$
    - Z has zero mean and variance 1 under null hypothesis

Statistics-based Methods

- Example:

  r: Browser=Mozilla $\wedge$ Buy=Yes $\rightarrow$ Age: μ=23
  - Rule is interesting if difference between μ and μ' is more than 5 years (i.e., Δ = 5)
  - For r, suppose n1 = 50, s1 = 3.5
  - For r' (complement): n2 = 250, s2 = 6.5

$$Z = \frac{\mu' - \mu - \Delta}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} = \frac{30 - 23 - 5}{\sqrt{\frac{3.5^2}{50} + \frac{6.5^2}{250}}} = 3.11$$

  - For 1-sided test at 95% confidence level, critical Z-value for rejecting null hypothesis is 1.64.
  - Since Z is greater than 1.64, r is an interesting rule

Min-Apriori

Document-term matrix:

TID	W1	W2	W3	W4	W5
D1	2	2	0	0	1
D2	0	0	1	2	2
D3	2	3	0	0	0
D4	0	0	1	0	1
D5	1	1	1	0	2

Example:

W1 and W2 tends to appear together in the same document

Min-Apriori

- Data contains only continuous attributes of the same "type"
  - e.g., frequency of words in a document

TID	W1	W2	W3	W4	W5
D1	2	2	0	0	1
D2	0	0	1	2	2
D3	2	3	0	0	0
D4	0	0	1	0	1
D5	1	1	1	0	2

- Potential solution:
  - Convert into 0/1 matrix and then apply existing algorithms
    - lose word frequency information
  - Discretization does not apply as users want association among words not ranges of words

Min-Apriori

- How to determine the support of a word?
  - If we simply sum up its frequency, support count will be greater than total number of documents!
    - Normalize the word vectors – e.g., using L_1 norms
    - Each word has a support equals to 1.0

TID	W1	W2	W3	W4	W5
D1	2	2	0	0	1
D2	0	0	1	2	2
D3	2	3	0	0	0
D4	0	0	1	0	1
D5	1	1	1	0	2

Normalize →

TID	W1	W2	W3	W4	W5
D1	0.40	0.33	0.00	0.00	0.17
D2	0.00	0.00	0.33	1.00	0.33
D3	0.40	0.50	0.00	0.00	0.00
D4	0.00	0.00	0.33	0.00	0.17
D5	0.20	0.17	0.33	0.00	0.33

Min-Apriori

- New definition of support:

$$\sup(C) = \sum_{i \in T} \min_{j \in C} D(i, j)$$

TID	W1	W2	W3	W4	W5
D1	0.40	0.33	0.00	0.00	0.17
D2	0.00	0.00	0.33	1.00	0.33
D3	0.40	0.50	0.00	0.00	0.00
D4	0.00	0.00	0.33	0.00	0.17
D5	0.20	0.17	0.33	0.00	0.33

Example:

Sup(W1,W2,W3)

= 0 + 0 + 0 + 0 + 0.17

= 0.17

Anti-monotone property of Support

TID	W1	W2	W3	W4	W5
D1	0.40	0.33	0.00	0.00	0.17
D2	0.00	0.00	0.33	1.00	0.33
D3	0.40	0.50	0.00	0.00	0.00
D4	0.00	0.00	0.33	0.00	0.17
D5	0.20	0.17	0.33	0.00	0.33

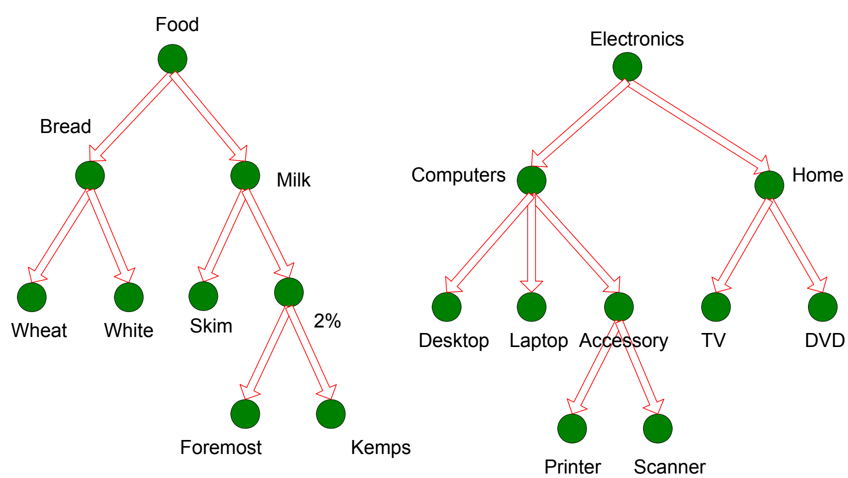
Example:

Sup(W1) = 0.4 + 0 + 0.4 + 0 + 0.2 = 1

Sup(W1, W2) = 0.33 + 0 + 0.4 + 0 + 0.17 = 0.9

Sup(W1, W2, W3) = 0 + 0 + 0 + 0 + 0.17 = 0.17

Concept Hierarchies



Multi-level Association Rules

- Why should we incorporate concept hierarchy?
  - Rules at lower levels may not have enough support to appear in any frequent itemsets
  - Rules at lower levels of the hierarchy are overly specific
    - e.g., skim milk → white bread, 2% milk → wheat bread, skim milk → wheat bread, etc.

    are indicative of association between milk and bread

Multi-level Association Rules

- How do support and confidence vary as we traverse the concept hierarchy?
  - If X is the parent item for both X1 and X2, then $\sigma(X) \leq \sigma(X1) + \sigma(X2)$
  - If $\sigma(X1 \cup Y1) \geq minsup$,
and X is parent of X1, Y is parent of Y1
then $\sigma(X \cup Y1) \geq minsup$, $\sigma(X1 \cup Y) \geq minsup$
$\sigma(X \cup Y) \geq minsup$
  - If $conf(X1 \Rightarrow Y1) \geq minconf$,
then $conf(X1 \Rightarrow Y) \geq minconf$

Multi-level Association Rules

- Approach 1:
  - Extend current association rule formulation by augmenting each transaction with higher level items

Original Transaction: {skim milk, wheat bread}

Augmented Transaction:
{skim milk, wheat bread, milk, bread, food}

- Issues:
  - Items that reside at higher levels have much higher support counts
    - if support threshold is low, too many frequent patterns involving items from the higher levels
  - Increased dimensionality of the data

Multi-level Association Rules

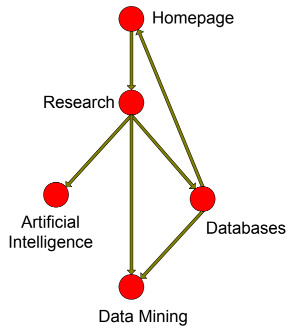
- Approach 2:
  - Generate frequent patterns at highest level first
  - Then, generate frequent patterns at the next highest level, and so on
- Issues:
  - I/O requirements will increase dramatically because we need to perform more passes over the data
  - May miss some potentially interesting cross-level association patterns

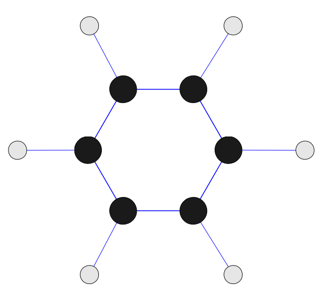
Association Analysis: Advanced Concepts

Subgraph Mining

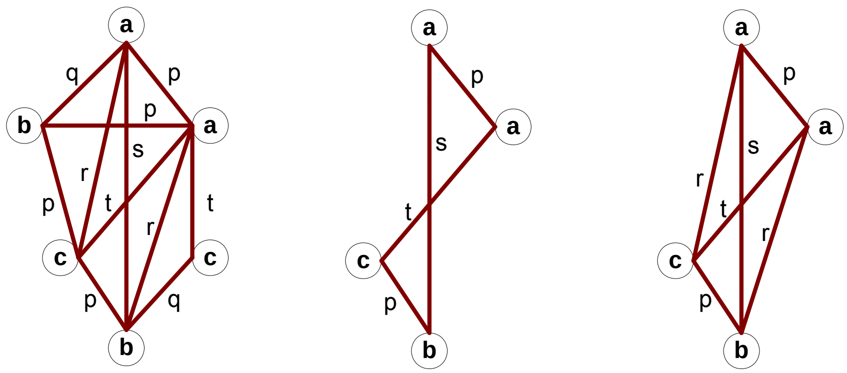
Frequent Subgraph Mining

- Extends association analysis to finding frequent subgraphs
- Useful for Web Mining, computational chemistry, bioinformatics, spatial data sets, etc





Graph Definitions

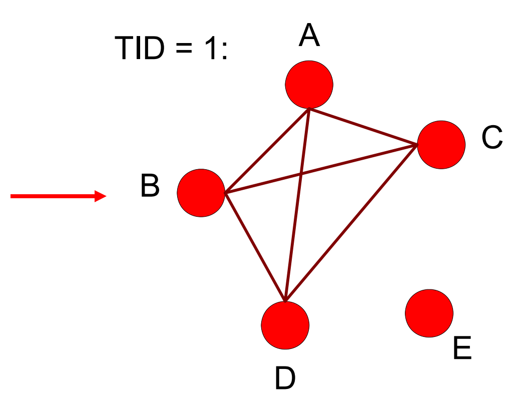


(a) Labeled Graph (b) Subgraph (c) Induced Subgraph

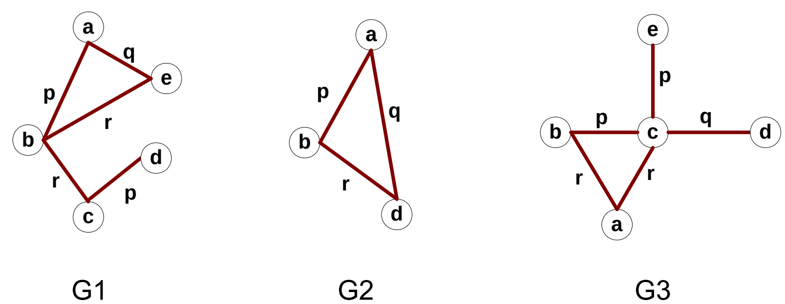
Representing Transactions as Graphs

- Each transaction is a clique of items

Transaction Id	Items
1	{A,B,C,D}
2	{A,B,E}
3	{B,C}
4	{A,B,D,E}
5	{B,C,D}



Representing Graphs as Transactions



	(a,b,p)	(a,b,q)	(a,b,r)	(b,c,p)	(b,c,q)	(b,c,r)	…	(d,e,r)
G1	1	0	0	0	0	1	…	0
G2	1	0	0	0	0	0	…	0
G3	0	0	1	1	0	0	…	0
G3	…	…	…	…	…	…	…	…

Challenges

- Node may contain duplicate labels
- Support and confidence
  - How to define them?
- Additional constraints imposed by pattern structure
  - Support and confidence are not the only constraints
  - Assumption: frequent subgraphs must be connected
- Apriori-like approach:
  - Use frequent k-subgraphs to generate frequent (k+1) subgraphs
    - What is k?

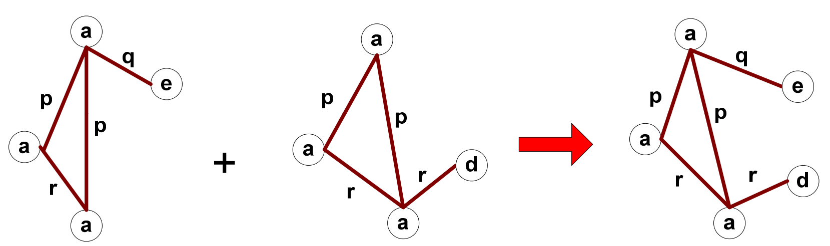
Challenges…

- Support:
  – number of graphs that contain a particular subgraph

- Apriori principle still holds

- Level-wise (Apriori-like) approach:
  – Vertex growing:
    - k is the number of vertices
  – Edge growing:
    - k is the number of edges

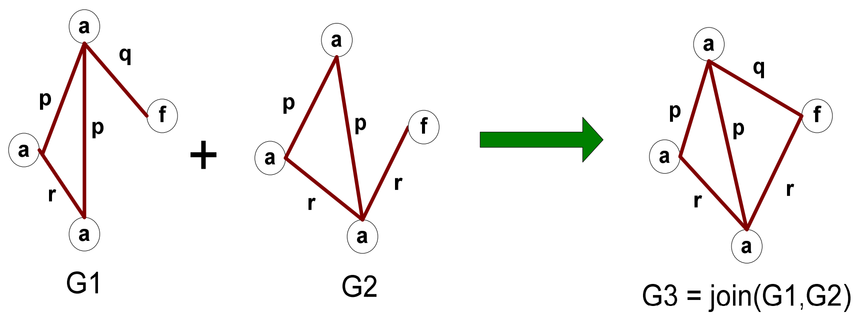
Vertex Growing



G1 G2 G3 = join(G1,G2)

$$M_{G1} = \begin{pmatrix} 0 & p & p & q \\ p & 0 & r & 0 \\ p & r & 0 & 0 \\ q & 0 & 0 & 0 \end{pmatrix} \quad M_{G2} = \begin{pmatrix} 0 & p & p & 0 \\ p & 0 & r & 0 \\ p & r & 0 & r \\ 0 & 0 & r & 0 \end{pmatrix} \quad M_{G3} = \begin{pmatrix} 0 & p & p & q & 0 \\ p & 0 & r & 0 & 0 \\ p & r & 0 & 0 & r \\ q & 0 & 0 & 0 & ? \\ 0 & 0 & r & ? & 0 \end{pmatrix}$$

Edge Growing

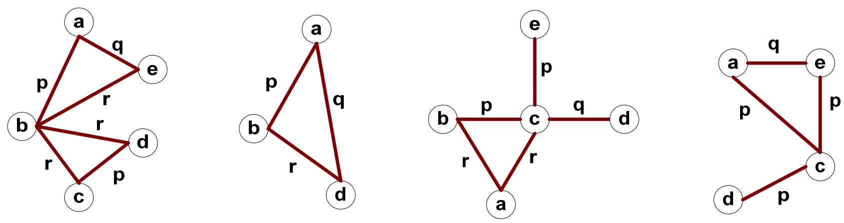


Apriori-like Algorithm

- Find frequent 1-subgraphs
- Repeat
  - Candidate generation
    - Use frequent (*k-1*)-subgraphs to generate candidate *k*-subgraph
  - Candidate pruning
    - Prune candidate subgraphs that contain infrequent (*k-1*)-subgraphs
  - Support counting
    - Count the support of each remaining candidate
  - Eliminate candidate *k*-subgraphs that are infrequent

In practice, it is not as easy. There are many other issues

Example: Dataset



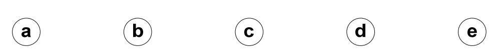
G1 G2 G3 G4

	(a,b,p)	(a,b,q)	(a,b,r)	(b,c,p)	(b,c,q)	(b,c,r)	…	(d,e,r)
G1	1	0	0	0	0	1	…	0
G2	1	0	0	0	0	0	…	0
G3	0	0	1	1	0	0	…	0
G4	0	0	0	0	0	0	…	0

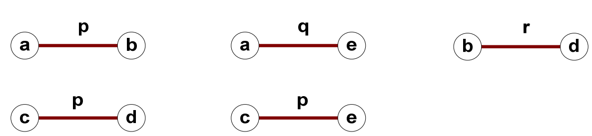
Example

Minimum support count = 2

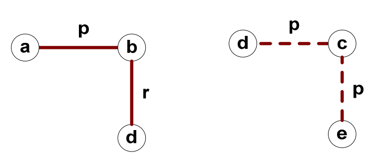
k=1
Frequent
Subgraphs



k=2
Frequent
Subgraphs



k=3
Candidate
Subgraphs



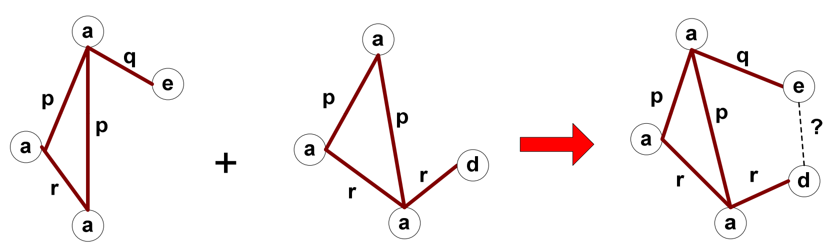
(Pruned candidate due to low support)

Candidate Generation

- In Apriori:
  - Merging two frequent *k*-itemsets will produce a candidate (*k+1*)-itemset

- In frequent subgraph mining (vertex/edge growing)
  - Merging two frequent *k*-subgraphs may produce more than one candidate (*k+1*)-subgraph

Multiplicity of Candidates (Vertex Growing)

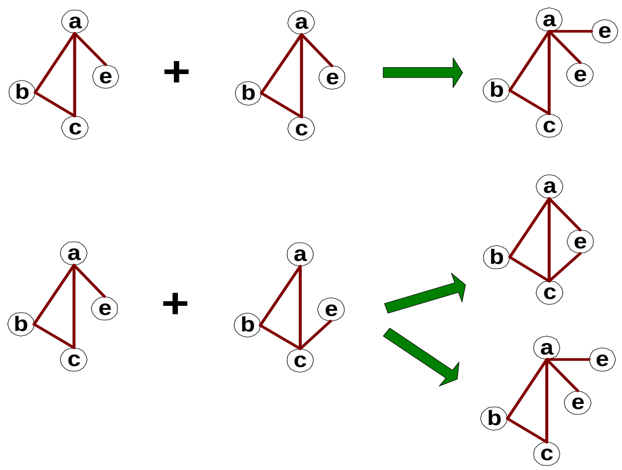


G1 G2 G3 = join(G1,G2)

$$M_{G1} = \begin{pmatrix} 0 & p & p & q \\ p & 0 & r & 0 \\ p & r & 0 & 0 \\ q & 0 & 0 & 0 \end{pmatrix} \quad M_{G2} = \begin{pmatrix} 0 & p & p & 0 \\ p & 0 & r & 0 \\ p & r & 0 & r \\ 0 & 0 & r & 0 \end{pmatrix} \quad M_{G3} = \begin{pmatrix} 0 & p & p & 0 & q \\ p & 0 & r & 0 & 0 \\ p & r & 0 & r & 0 \\ 0 & 0 & r & 0 & ? \\ q & 0 & 0 & ? & 0 \end{pmatrix}$$

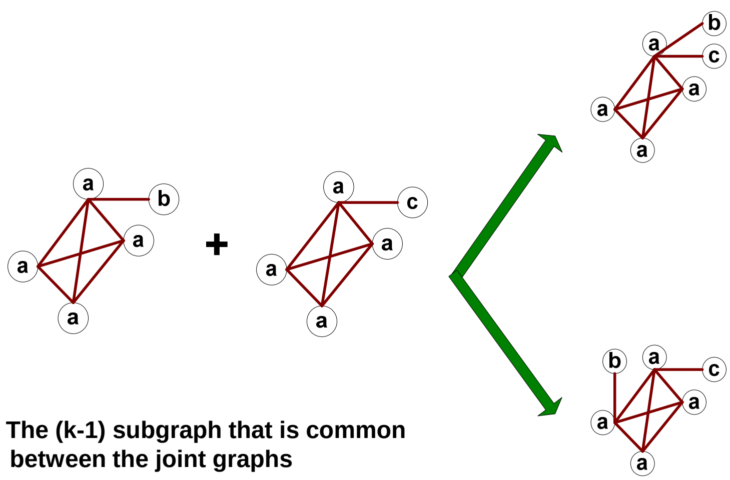
Multiplicity of Candidates (Edge growing)

- Case 1: identical vertex labels



Multiplicity of Candidates (Edge growing)

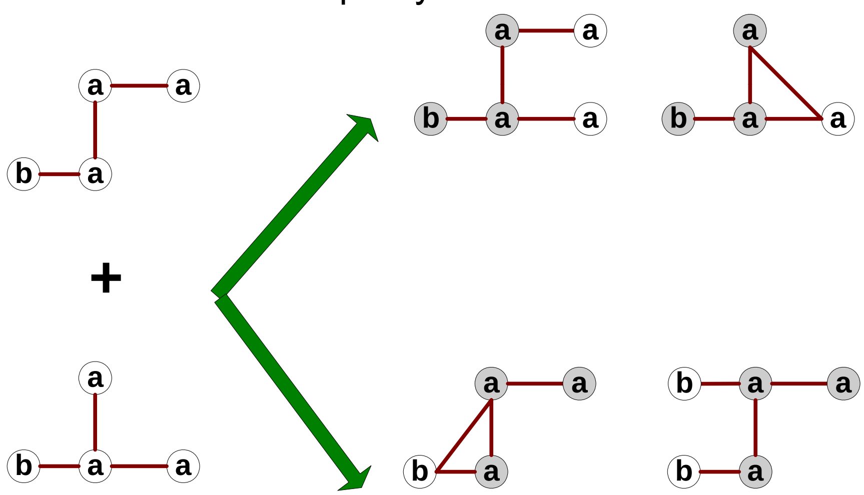
- Case 2: Core contains identical labels



Core: The (k-1) subgraph that is common between the joint graphs

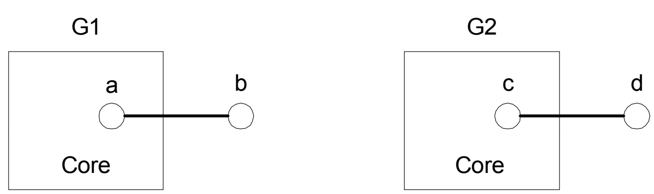
Multiplicity of Candidates (Edge growing)

- Case 3: Core multiplicity

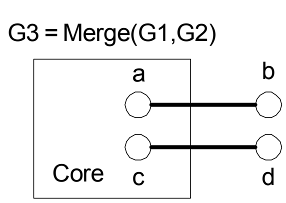


Candidate Generation by Edge Growing

- Given:

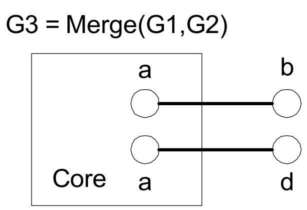


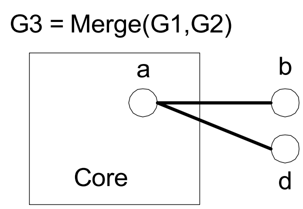
- Case 1: a ≠ c and b ≠ d



Candidate Generation by Edge Growing

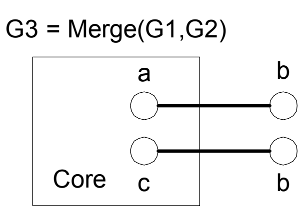
- Case 2: a = c and b ≠ d

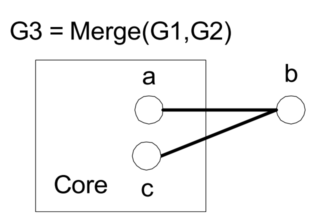




Candidate Generation by Edge Growing

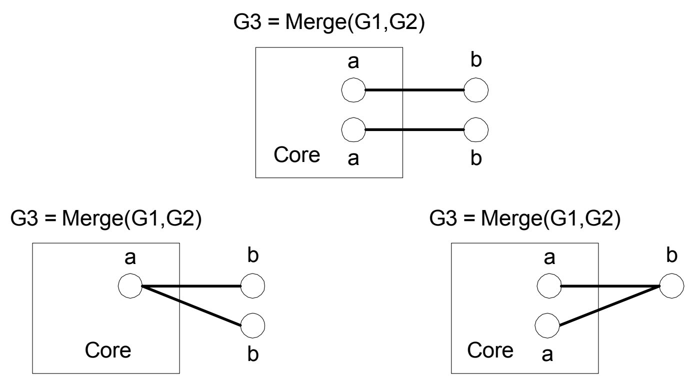
- Case 3: a ≠ c and b = d





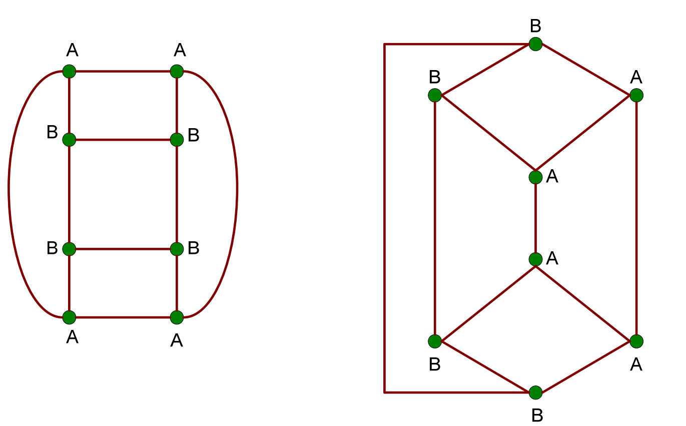
Candidate Generation by Edge Growing

- Case 4: a = c and b = d



Graph Isomorphism

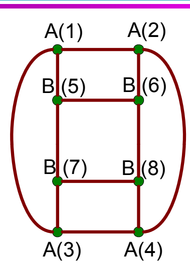
- A graph is isomorphic if it is topologically equivalent to another graph



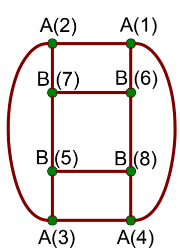
Graph Isomorphism

- Test for graph isomorphism is needed:
  - During candidate generation step, to determine whether a candidate has been generated
  - During candidate pruning step, to check whether its (*k-1*)-subgraphs are frequent
  - During candidate counting, to check whether a candidate is contained within another graph

Graph Isomorphism



	A(1)	A(2)	A(3)	A(4)	B(5)	B(6)	B(7)	B(8)
A(1)	1	1	1	0	1	0	0	0
A(2)	1	1	0	1	0	1	0	0
A(3)	1	0	1	1	0	0	1	0
A(4)	0	1	1	1	0	0	0	1
B(5)	1	0	0	0	1	1	1	0
B(6)	0	1	0	0	1	1	0	1
B(7)	0	0	1	0	1	0	1	1
B(8)	0	0	0	1	0	1	1	1

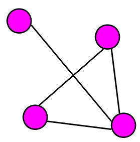


	A(1)	A(2)	A(3)	A(4)	B(5)	B(6)	B(7)	B(8)
A(1)	1	1	0	1	0	1	0	0
A(2)	1	1	1	0	0	0	1	0
A(3)	0	1	1	1	1	0	0	0
A(4)	1	0	1	1	0	0	0	1
B(5)	0	0	1	0	1	0	1	1
B(6)	1	0	0	0	0	1	1	1
B(7)	0	1	0	0	1	1	1	0
B(8)	0	0	0	1	1	1	0	1

- **The same graph can be represented in many ways**

Graph Isomorphism

- Use canonical labeling to handle isomorphism
  - Map each graph into an ordered string representation (known as its code) such that two isomorphic graphs will be mapped to the same canonical encoding
  - Example:
    - Lexicographically largest adjacency matrix



$$\rightarrow \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

String: 011011 **Canonical: 111100**