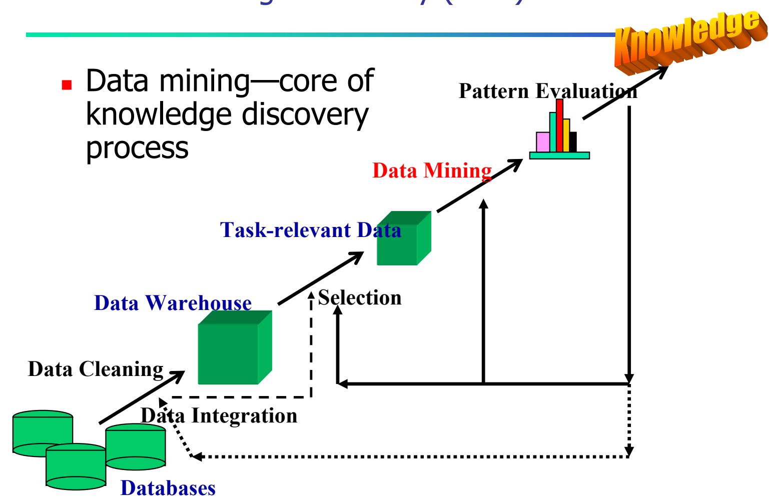
Data Mining:
Concepts and Techniques

— Chapter 5 —

Jiawei Han
Department of Computer Science
University of Illinois at Urbana-Champaign
www.cs.uiuc.edu/~hanj

Knowledge Discovery (KDD) Process

- Data mining—core of knowledge discovery process

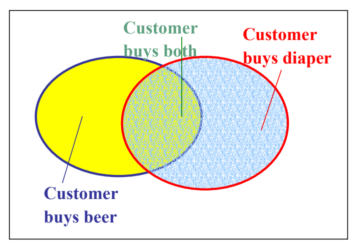


What Is Frequent Pattern Analysis?

- Frequent pattern: a pattern (a set of items, subsequences, substructures, etc.) that occurs frequently in a data set
- First proposed by Agrawal, Imielinski, and Swami [AIS93] in the context of frequent itemsets and association rule mining
- Motivation: Finding inherent regularities in data
  - What products were often purchased together?
  - What are the subsequent purchases after buying a PC?
  - What kinds of DNA are sensitive to this new drug?
  - Can we automatically classify web documents?
- Applications
  - Basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis.

Basic Concepts: Frequent Patterns and Association Rules

Transaction-id	Items bought
10	A, B, D
20	A, C, D
30	A, D, E
40	B, E, F
50	B, C, D, E, F



- Itemset X = $\{x_1, \ldots, x_k\}$
- Find all the rules $X \rightarrow Y$ with minimum support and confidence
  - support, *s*, probability that a transaction contains $X \cup Y$
  - confidence, *c*, conditional probability that a transaction having X also contains *Y*

Let $sup_{min} = 50\%$, $conf_{min} = 50\%$
Freq. Pat.: {A:3, B:3, D:4, E:3, AD:3}
Association rules:
$A \rightarrow D$ (60%, 100%)
$D \rightarrow A$ (60%, 75%)

Closed Patterns and Max-Patterns

- A long pattern contains a combinatorial number of sub-patterns, e.g., $\{a_1, \ldots, a_{100}\}$ contains $\binom{100}{1} + \binom{100}{2} + \ldots + \binom{100}{100} = 2^{100} - 1 = 1.27 \times 10^{30}$ sub-patterns!
- Solution: *Mine closed patterns and max-patterns instead*
- An itemset X is closed if X is *frequent* and there exists *no super-pattern* Y ⊃ X, *with the same support* as X (proposed by Pasquier, et al. @ ICDT'99)
- An itemset X is a max-pattern if X is frequent and there exists no frequent super-pattern Y ⊃ X (proposed by Bayardo @ SIGMOD'98)
- Closed pattern is a lossless compression of freq. patterns
  - Reducing the # of patterns and rules

Scalable Methods for Mining Frequent Patterns

- The downward closure property of frequent patterns
  - Any subset of a frequent itemset must be frequent
  - If **{beer, diaper, nuts}** is frequent, so is **{beer, diaper}**
  - i.e., every transaction having {beer, diaper, nuts} also contains {beer, diaper}
- Scalable mining methods: Three major approaches
  - Apriori (Agrawal & Srikant@VLDB'94)
  - Freq. pattern growth (FPgrowth—Han, Pei & Yin @SIGMOD'00)
  - Vertical data format approach (Charm—Zaki & Hsiao @SDM'02)

Bottleneck of Frequent-pattern Mining

- Multiple database scans are costly
- Mining long patterns needs many passes of scanning and generates lots of candidates
  - To find frequent itemset $i_1 i_2 \ldots i_{100}$
    - # of scans: 100
    - # of Candidates: $\binom{100}{1} + \binom{100}{2} + \ldots + \binom{100}{100} = 2^{100} - 1 = 1.27 \times 10^{30}$!
- Bottleneck: candidate-generation-and-test
- Can we avoid candidate generation?

Mining Frequent Patterns Without Candidate Generation

- Grow long patterns from short ones using local frequent items
  - "abc" is a frequent pattern
  - Get all transactions having "abc": DB|abc
  - "d" is a local frequent item in DB|abc → abcd is a frequent pattern

Construct FP-tree from a Transaction Database

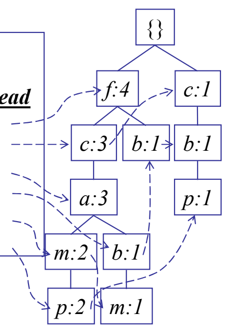
TID	Items bought	(ordered) frequent items
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}
200	{a, b, c, f, l, m, o}	{f, c, a, b, m}
300	{b, f, h, j, o, w}	{f, b}
400	{b, c, k, s, p}	{c, b, p}
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}

min_support = 3

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

Header Table

Item	frequency	head
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



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

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 counting the *node-links* and the *count* field)
  - Some databases show a compression ratio of over 100

Partition Patterns and Databases

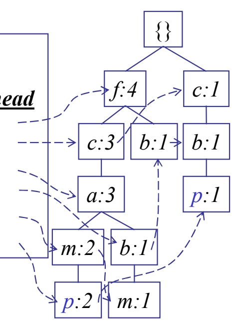
- Frequent patterns can be partitioned into subsets according to f-list
  - F-list=f-c-a-b-m-p
  - Patterns containing p
  - Patterns having m but no p
  - …
  - Patterns having c but no a nor b, m, p
  - Pattern f
- Completeness and non-redundancy

Find Patterns Having P From P-conditional Database

- Starting at the frequent item header table in the FP-tree
- Traverse the FP-tree by following the link of each frequent item *p*
- Accumulate all the *transformed prefix paths* of item *p* to form *p*'s conditional pattern base

Header Table

Item	frequency	head
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



***Conditional* pattern bases for all items from F-list:**

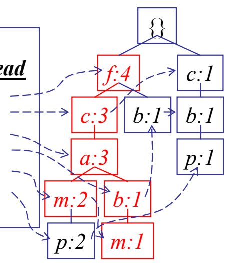
item	cond. pattern base
c	f:3
a	fc:3
b	fca:1, f:1, c:1
m	fca:2, fcab:1
p	fcam:2, cb:1

From Conditional Pattern-bases to Conditional FP-trees

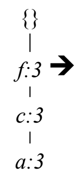
- For each pattern-base
  - Accumulate the count for each item in the base
  - Construct the FP-tree for the frequent items of the pattern base

Header Table

Item	*frequency*	*head*
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



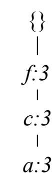
***m*-conditional** **pattern base:**
fca:2, fcab:1



m-conditional **FP-tree**

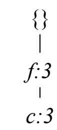
All frequent (min-sup = 3) patterns relate to *m*
m,
fm, cm, am,
fcm, fam, cam,
fcam

Recursion: Mining Each Conditional FP-tree



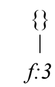
m-conditional **FP-tree**

Cond. pattern base of "am": (fc:3)



am-conditional **FP-tree**

Cond. pattern base of "cm": (f:3)



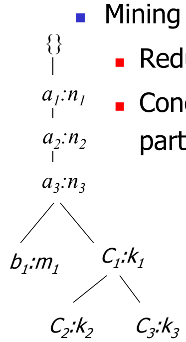
cm-conditional **FP-tree**

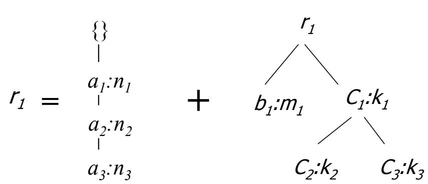
Cond. pattern base of "cam": (f:3)

cam-conditional **FP-tree**

A Special Case: Single Prefix Path in FP-tree

- Suppose a (conditional) FP-tree T has a shared single prefix-path P
- Mining can be decomposed into two parts
  - Reduction of the single prefix path into one node
  - Concatenation of the mining results of the two parts





Mining Frequent Patterns With FP-trees

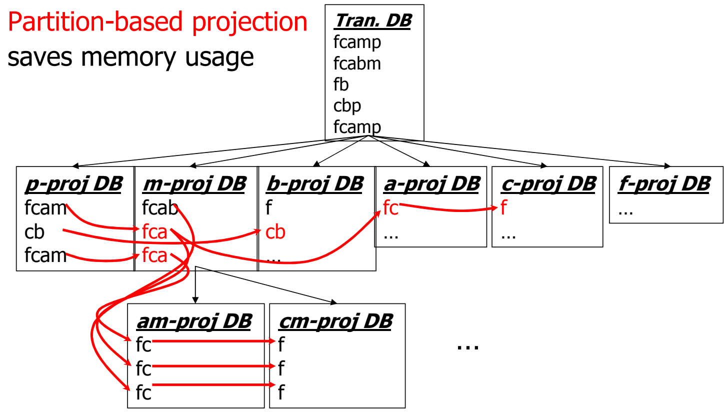
- Idea: Frequent pattern growth
  - Recursively grow frequent patterns by pattern and database partition
- Method
  - For each frequent item, construct its conditional pattern-base, and then its conditional FP-tree
  - Repeat the process on each newly created conditional FP-tree
  - Until the resulting 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 DB Projection

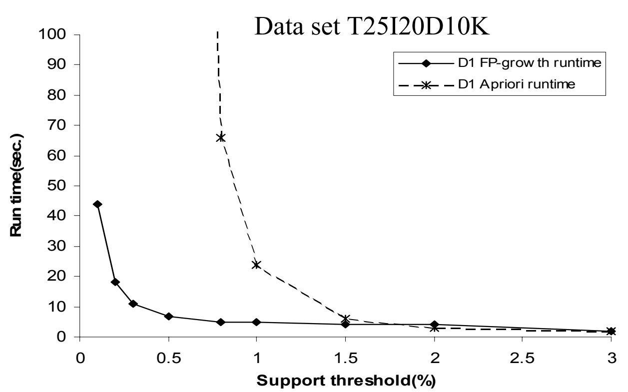
- FP-tree cannot fit in memory? => DB projection
  1. First partition a database into a set of projected DBs
  2. Then construct and mine the FP-tree for each projected DB

Partition-based Projection

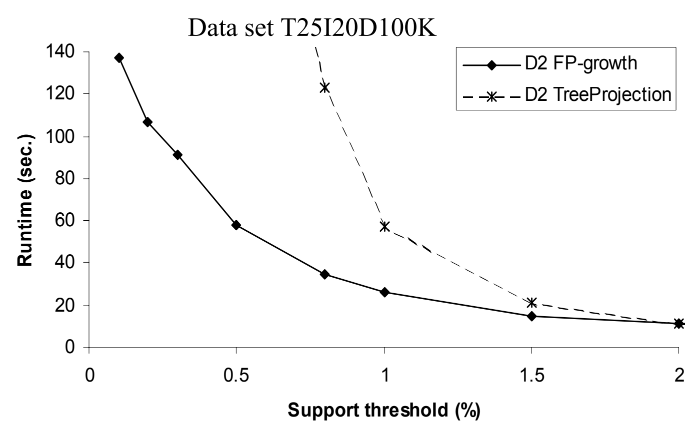
- Partition-based projection saves memory usage



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



FP-Growth vs. Tree-Projection: Scalability with the Support Threshold



Why Is FP-Growth the Winner?

- Divide-and-conquer:
  - decompose both the mining task and the database according to the frequent patterns obtained so far
  - leads 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 operations—counting local freq items and building sub FP-tree, no pattern search and matching

Implications of the Methodology

- Mining closed frequent itemsets and max-patterns
  - CLOSET (DMKD'00)
- Mining sequential patterns
  - FreeSpan (KDD'00), PrefixSpan (ICDE'01)
- Constraint-based mining of frequent patterns
  - Convertible constraints (KDD'00, ICDE'01)
- Computing iceberg data cubes with complex measures
  - H-tree and H-cubing algorithm (SIGMOD'01)

Closed Patterns and Max-Patterns

Recall:

- An itemset X is a max-pattern if X is frequent and there exists no frequent super-pattern Y ⊃ X.

MaxMiner: Mining Max-patterns

Tid	Items
10	A,B,C,D,E
20	B,C,D,E,
30	A,C,D,F

- 1st scan: find frequent items
  - A, B, C, D, E
- 2nd scan: find support for
  - AB, AC, AD, AE, ABCDE
  - BC, BD, BE, BCDE
  - CD, CE, CDE, DE,

Potential max-patterns

- Since BCDE is a max-pattern, no need to check BCD, BDE, CDE in later scan
- R. Bayardo. Efficiently mining long patterns from databases. In *SIGMOD'98*

Mining Frequent Closed Patterns: CLOSET

Recall:

- An itemset X is closed if X is *frequent* and there exists *no super-pattern* Y ⊃ X, *with the same support* as X (proposed by Pasquier, et al. @ ICDT'99)
- Closed pattern is a lossless compression of freq. patterns
  - Reducing the # of patterns and rules

Mining Frequent Closed Patterns: CLOSET

- F-list: list of all frequent items in support ascending order
  - F-list: d-a-f-e-c support: [2,3,4,4,4]
- Divide search space
  - Patterns having d
  - Patterns having d but no a, etc.
- Find frequent closed pattern recursively
  - Every transaction having d also has cfa → cfad is a frequent closed pattern
- J. Pei, J. Han & R. Mao. CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets", DMKD'00.

Min_sup=2

TID	Items
10	a, c, d, e, f
20	a, b, e
30	c, e, f
40	a, c, d, f
50	c, e, f

CLOSET+: Mining Closed Itemsets by Pattern-Growth

- Itemset merging: if Y appears in every occurrence of X, then Y is merged with X
- Sub-itemset pruning: if Y ⊃ X, and sup(X) = sup(Y), X and all of X's descendants in the set enumeration tree can be pruned
- Hybrid tree projection [see article]
  - Bottom-up physical tree-projection
  - Top-down pseudo tree-projection
- Item skipping [see article]: if a local frequent item has the same support in several header tables at different levels, one can prune it from the header table at higher levels
- Efficient subset checking [see article]

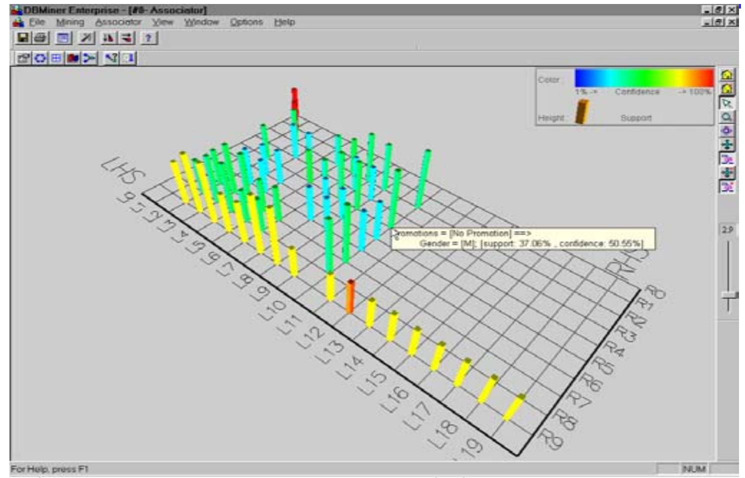
CHARM: Mining by Exploring Vertical Data Format

- Vertical format: t(AB) = $\{T_{11}, T_{25}, \ldots\}$
  - tid-list: list of trans.-ids containing an itemset
- Deriving closed patterns based on vertical intersections
  - t(X) = t(Y): X and Y always happen together
  - t(X) ⊂ t(Y): transaction having X always has Y
- Using diffset to accelerate mining
  - Only keep track of differences of tids
  - t(X) = $\{T_1, T_2, T_3\}$, t(XY) = $\{T_1, T_3\}$
  - Diffset (XY, X) = $\{T_2\}$
- Eclat/MaxEclat (Zaki et al. @KDD'97), VIPER(P. Shenoy et al.@SIGMOD'00), CHARM (Zaki & Hsiao@SDM'02)

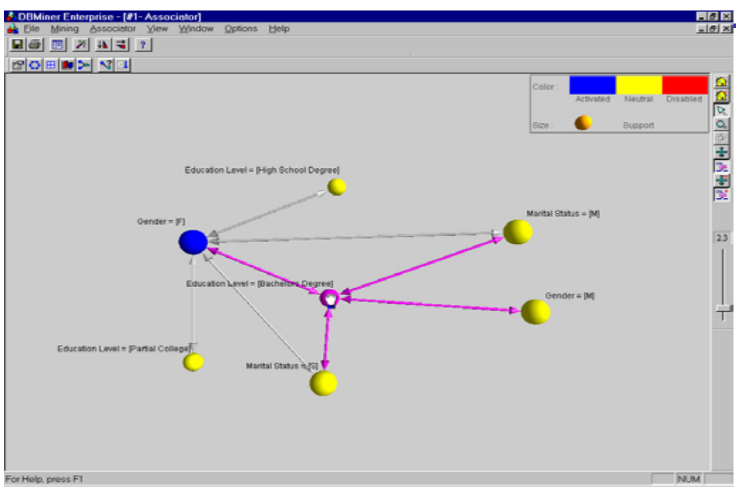
Further Improvements of Mining Methods

- AFOPT (Liu, et al. @ KDD'03)
  - A "push-right" method for mining condensed frequent pattern (CFP) tree
- Carpenter (Pan, et al. @ KDD'03)
  - Techniques for sparse data sets
  - Mine data sets with small rows but numerous columns
  - Construct a row-enumeration tree for efficient mining

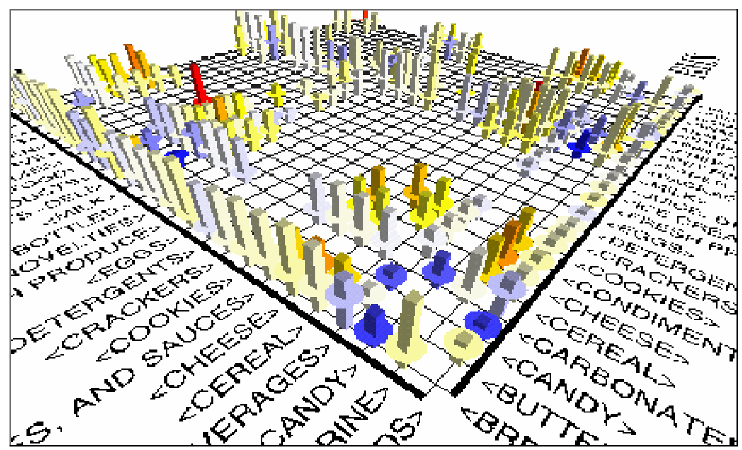
Visualization of Association Rules: Plane Graph



Visualization of Association Rules: Rule Graph



Visualization of Association Rules (SGI/MineSet 3.0)



Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map
- Efficient and scalable frequent itemset mining methods
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary

Mining Various Kinds of Association Rules

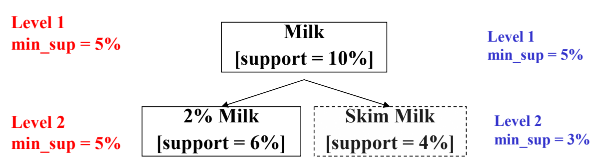
- Mining multilevel association
- Miming multidimensional association
- Mining quantitative association
- Mining interesting correlation patterns

Mining Multiple-Level Association Rules

- Items often form hierarchies
- Flexible support settings
  - Items at the lower (more specialized) level are expected to have lower support
- Exploration of *shared* multi-level mining (Agrawal & Srikant@VLB'95, Han & Fu@VLDB'95)

uniform support

reduced support



Multi-level Association: Redundancy Filtering

- Some rules may be redundant due to "ancestor" relationships between items.
- Example
  - milk $\Rightarrow$ wheat bread [support = 8%, confidence = 70%]
  - 2% milk $\Rightarrow$ wheat bread [support = 2%, confidence = 72%]
- We say the first rule is an ancestor of the second rule.
- A rule is redundant if its support is close to the "expected" value, based on the rule's ancestor.

(Note, 2% milk is reduced fat milk.)

Mining Multi-Dimensional Association

- Single-dimensional rules:
  buys(X, "milk") ⇒ buys(X, "bread")
- Multi-dimensional rules: ≥ 2 dimensions or predicates
  - Inter-dimension assoc. rules (*no repeated predicates*)
    age(X,"19-25") ∧ occupation(X,"student") ⇒ buys(X, "coke")
  - hybrid-dimension assoc. rules (*repeated predicates*)
    age(X,"19-25") ∧ buys(X, "popcorn") ⇒ buys(X, "coke")
- Categorical Attributes: finite number of possible values, no ordering among values—data cube approach
- Quantitative Attributes: numeric, implicit ordering among values—discretization, clustering, and gradient approaches

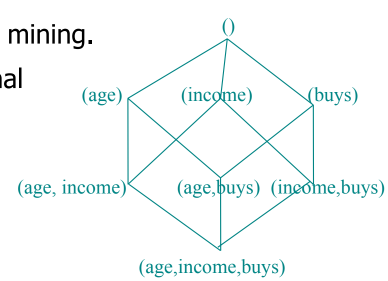
Mining Quantitative Associations

- Techniques can be categorized by how numerical attributes, such as age or salary are treated

1. Static discretization based on predefined concept hierarchies (data cube methods)
2. Dynamic discretization based on data distribution (quantitative rules, e.g., Agrawal & Srikant@SIGMOD96)
3. Clustering: Distance-based association (e.g., Yang & Miller@SIGMOD97)
   - one dimensional clustering then association
4. Deviation: (such as Aumann and Lindell@KDD99)
   Sex = female => Wage: mean=$7/hr (overall mean = $9)

Static Discretization of Quantitative Attributes

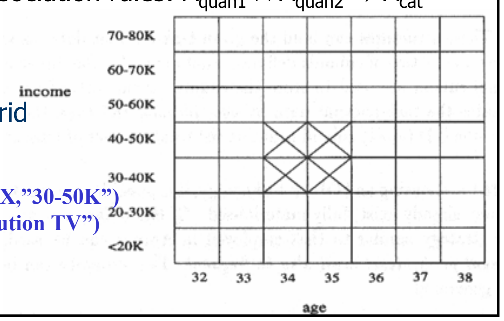
- Discretized prior to mining using concept hierarchy.
- Numeric values are replaced by ranges.
- In relational database, finding all frequent k-predicate sets will require *k* or *k*+1 table scans.
- Data cube is well suited for mining.
- The cells of an n-dimensional cuboid correspond to the predicate sets.
- Mining from data cubes can be much faster.



Quantitative Association Rules

- Proposed by Lent, Swami and Widom ICDE'97
- Numeric attributes are *dynamically* discretized
  - Such that the confidence or compactness of the rules mined is maximized
- 2-D quantitative association rules: $A_{quan1} \wedge A_{quan2} \Rightarrow A_{cat}$
- Cluster *adjacent* association rules to form general rules using a 2-D grid
- Example

age(X,"34-35") ∧ income(X,"30-50K") ⇒ buys(X,"high resolution TV")



Mining Other Interesting Patterns

- Flexible support constraints (Wang et al. @ VLDB'02)
  - Some items (e.g., diamond) may occur rarely but are valuable
  - Customized sup_{min} specification and application
- Top-K closed frequent patterns (Han, et al. @ ICDM'02)
  - Hard to specify sup_{min}, but top-k with $length_{min}$ is more desirable
  - Dynamically raise sup_{min} in FP-tree construction and mining, and select most promising path to mine

Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map
- Efficient and scalable frequent itemset mining methods
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary

Interestingness Measure: Correlations (Lift)

- *play basketball* $\Rightarrow$ *eat cereal* [40%, 66.7%] is misleading
  - The overall % of students eating cereal is 75% > 66.7%.
- *play basketball* $\Rightarrow$ *not eat cereal* [20%, 33.3%] is more accurate, although with lower support and confidence
- Measure of dependent/correlated events: lift

$$lift = \frac{P(A \cup B)}{P(A)P(B)}$$

	Basketball (B)	Not basketball	Sum (row)
Cereal (C)	2000	1750	3750
Not cereal	1000	250	1250
Sum(col.)	3000	2000	5000

$$lift(B,C) = \frac{2000/5000}{3000/5000 * 3750/5000} = 0.89 \qquad lift(B,\neg C) = \frac{1000/5000}{3000/5000 * 1250/5000} = 1.33$$

Are *lift* and χ^2 Good Measures of Correlation?

- "*Buy walnuts* $\Rightarrow$ *buy milk* [1%, 80%]" is misleading
  - if 85% of customers buy milk
- Support and confidence are not good to represent correlations
- So many interestingness measures? (Tan, Kumar, Sritastava @KDD'02)

$$coh = \frac{P(A,B)}{P(A)+P(B)-P(A,B)}$$

$$all\_conf = \frac{P(A,B)}{\max\{P(A),P(B)\}}$$

$$lift = \frac{P(A \cup B)}{P(A)P(B)}$$

	Milk	No Milk	Sum (row)
Coffee	m, c	~m, c	c
No Coffee	m, ~c	~m, ~c	~c
Sum(col.)	m	~m	Σ

DB	m, c	~m, c	m~c	~m~c	lift	all-conf	coh	$\chi 2$
A1	1000	100	100	10,000	9.26	0.91	0.83	9055
A2	100	1000	1000	100,000	8.44	0.09	0.05	670
A3	1000	100	10000	100,000	9.18	0.09	0.09	8172
A4	1000	1000	1000	1000	1	0.5	0.33	0

Which Measures Should Be Used?

- **lift** *and* χ^2 are not good measures for correlations in large transactional DBs
- **all-conf** or **coherence** could be good measures (Omiecinski@TKDE'03)
- Both **all-conf** and **coherence** have the downward closure property
- Efficient algorithms can be derived for mining (Lee et al. @ICDM'03sub)

symbol	measure	range	formula
ϕ	ϕ-coefficient	-1 … 1	$\frac{P(A,B)-P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
Q	Yule's Q	-1 … 1	$\frac{P(A,B)P(\overline{A},\overline{B})-P(A,\overline{B})P(\overline{A},B)}{P(A,B)P(\overline{A},\overline{B})+P(A,\overline{B})P(\overline{A},B)}$
Y	Yule's Y	-1 … 1	$\frac{\sqrt{P(A,B)P(\overline{A},\overline{B})}-\sqrt{P(A,\overline{B})P(\overline{A},B)}}{\sqrt{P(A,B)P(\overline{A},\overline{B})}+\sqrt{P(A,\overline{B})P(\overline{A},B)}}$
k	Cohen's	-1 … 1	$\frac{P(A,B)+P(\overline{A},\overline{B})-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A)P(B)-P(\overline{A})P(\overline{B})}$
PS	Piatetsky-Shapiro's	-0.25 … 0.25	$P(A,B)-P(A)P(B)$
F	Certainty factor	-1 … 1	$\max(\frac{P(B\|A)-P(B)}{1-P(B)}, \frac{P(A\|B)-P(A)}{1-P(A)})$
AV	added value	-0.5 … 1	$\max(P(B\|A)-P(B), P(A\|B)-P(A))$
K	Klosgen's Q	-0.33 … 0.38	$\sqrt{P(A,B)}\max(P(B\|A)-P(B), P(A\|B)-P(A))$
g	Goodman-kruskal's	0 … 1	$\frac{\sum_j \max_k P(A_j,B_k)+\sum_k \max_j P(A_j,B_k)-\max_j P(A_j)-\max_k P(B_k)}{2-\max_j P(A_j)-\max_k P(B_k)}$
M	Mutual Information	0 … 1	$\frac{\sum_i \sum_j P(A_i,B_j)\log\frac{P(A_i,B_j)}{P(A_i)P(B_j)}}{\min(-\sum_i P(A_i)\log P(A_i), -\sum_i P(B_i)\log P(B_i))}$
J	J-Measure	0 … 1	$\max(P(A,B)\log(\frac{P(B\|A)}{P(B)})+P(A\overline{B})\log(\frac{P(\overline{B}\|A)}{P(\overline{B})}),$ $P(A,B)\log(\frac{P(A\|B)}{P(A)})+P(\overline{A}B)\log(\frac{P(\overline{A}\|B)}{P(\overline{A})}))$
G	Gini index	0 … 1	$\max(P(A)[P(B\|A)^2+P(\overline{B}\|A)^2]+P(\overline{A})[P(B\|\overline{A})^2+P(\overline{B}\|\overline{A})^2]-P(B)^2-P(\overline{B})^2,$ $P(B)[P(A\|B)^2+P(\overline{A}\|B)^2]+P(\overline{B})[P(A\|\overline{B})^2+P(\overline{A}\|\overline{B})^2]-P(A)^2-P(\overline{A})^2)$
s	support	0 … 1	$P(A,B)$
c	confidence	0 … 1	$max(P(B\|A), P(A\|B))$
L	Laplace	0 … 1	$\max(\frac{NP(A,B)+1}{NP(A)+2}, \frac{NP(A,B)+1}{NP(B)+2})$
IS	Cosine	0 … 1	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
γ	coherence(Jaccard)	0 … 1	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$
α	all_confidence	0 … 1	$\frac{P(A,B)}{\max(P(A),P(B))}$
o	odds ratio	0 … ∞	$\frac{P(A,B)P(\overline{A},\overline{B})}{P(\overline{A},B)P(A,\overline{B})}$
V	Conviction	0.5 … ∞	$\max(\frac{P(A)P(\overline{B})}{P(A\overline{B})}, \frac{P(B)P(\overline{A})}{P(B\overline{A})})$
λ	lift	0 … ∞	$\frac{P(A,B)}{P(A)P(B)}$
S	Collective strength	0 … ∞	$\frac{P(A,B)+P(\overline{AB})}{P(A)P(B)+P(\overline{A})P(\overline{B})}\times\frac{1-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A,B)-P(\overline{AB})}$
χ^2	χ^2	0 … ∞	$\sum_i \frac{(P(A_i)-E_i)^2}{E_i}$

Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map
- Efficient and scalable frequent itemset mining methods
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary

Constraint-based (Query-Directed) Mining

- Finding all the patterns in a database autonomously? — unrealistic!
  - The patterns could be too many but not focused!
- Data mining should be an interactive process
  - User directs what to be mined using a data mining query language (or a graphical user interface)
- Constraint-based mining
  - User flexibility: provides constraints on what to be mined
  - System optimization: explores such constraints for efficient mining—constraint-based mining

Constraints in Data Mining

- Knowledge type constraint:
  - classification, association, etc.
- Data constraint — using SQL-like queries
  - find product pairs sold together in stores in Chicago in Dec.'02
- Dimension/level constraint
  - in relevance to region, price, brand, customer category
- Rule (or pattern) constraint
  - small sales (price < $10) triggers big sales (sum > $200)
- Interestingness constraint
  - strong rules: min_support ≥ 3%, min_confidence ≥ 60%

Constrained Mining vs. Constraint-Based Search

- Constrained mining vs. constraint-based search/reasoning
  - Both are aimed at reducing search space
  - Finding all patterns satisfying constraints vs. finding some (or one) answer in constraint-based search in AI
  - Constraint-pushing vs. heuristic search
  - It is an interesting research problem on how to integrate them
- Constrained mining vs. query processing in DBMS
  - Database query processing requires to find all
  - Constrained pattern mining shares a similar philosophy as pushing selections deeply in query processing

Anti-Monotonicity in Constraint Pushing

- Anti-monotonicity
  - *When an intemset S **violates** the constraint, so does any of its superset*
  - $sum(S.Price) \le v$ is anti-monotone
  - $sum(S.Price) \ge v$ is not anti-monotone
- Example. C: range(S.profit) $\le$ 15 is anti-monotone
  - Itemset *ab* violates C: profit range is [0,40]
  - So does every superset of *ab*

TDB (min_sup=2)

TID	Transaction
10	a, b, c, d, f
20	b, c, d, f, g, h
30	a, c, d, e, f
40	c, e, f, g

Item	Profit
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

Monotonicity for Constraint Pushing

- Monotonicity
  - *When an intemset S **satisfies** the constraint, so does any of its superset*
  - $sum(S.Price) \ge v$ is monotone
  - $min(S.Price) \le v$ is monotone

  (where Price $\ge$ 0)
- Example. C: range(S.profit) $\ge$ 15
  - Itemset *ab* satisfies C (range [0,40])
  - So does every superset of *ab*

TDB (min_sup=2)

TID	Transaction
10	a, b, c, d, f
20	b, c, d, f, g, h
30	a, c, d, e, f
40	c, e, f, g

Item	Profit
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

Succinctness

- Succinctness:
  - Given A_1, the set of items satisfying a succinctness constraint *C*, then any set *S* satisfying *C* is based on A_1 , i.e., *S* contains a subset belonging to A_1
  - Idea: Without looking at the transaction database, whether an itemset *S* satisfies constraint C can be determined based on the selection of items (in A_1)
  - $min(S.Price) \le v$ is succinct
  - $sum(S.Price) \ge v$ is not succinct
- Optimization: If *C* is succinct, *C* is pre-counting pushable

The Apriori Algorithm — Example

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Scan D → C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

Scan D → C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

itemset
{2 3 5}

Scan D → L_3

itemset	sup
{2 3 5}	2

Naïve Algorithm: Apriori + Constraint

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Scan D → C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
~~{5}~~	~~3~~

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

Scan D → C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

L_2

itemset	sup
{1 3}	2
~~{2 3}~~	~~2~~
~~{2 5}~~	~~3~~
~~{3 5}~~	~~2~~

C_3

itemset
{2 3 5}

Scan D → L_3

itemset	sup
~~{2 3 5}~~	~~2~~

Constraint:

Sum{S.price} < 5

The Constrained Apriori Algorithm: Push an Anti-monotone Constraint Deep

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Scan D → C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
~~{5}~~	~~3~~

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
~~{5}~~	~~3~~

C_2

itemset
{1 2}
{1 3}
~~{1 5}~~
{2 3}
~~{2 5}~~
~~{3 5}~~

Scan D → C_2

itemset	sup
{1 2}	1
{1 3}	2
~~{1 5}~~	~~1~~
~~{2 3}~~	~~2~~
~~{2 5}~~	~~3~~
~~{3 5}~~	~~2~~

L_2

itemset	sup
{1 3}	2
~~{2 3}~~	~~2~~
~~{2 5}~~	~~3~~
~~{3 5}~~	~~2~~

C_3

itemset
~~{2 3 5}~~

Scan D → L_3

itemset	sup
~~{2 3 5}~~	~~2~~

Constraint:

Sum{S.price} < 5

The Constrained Apriori Algorithm: Push a Succinct Constraint Deep

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Scan D → C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
~~{2 3}~~
~~{2 5}~~
~~{3 5}~~

not immediately to be used

Scan D → C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
~~{2 3}~~	~~2~~
~~{2 5}~~	~~3~~
~~{3 5}~~	~~2~~

L_2

itemset	sup
{1 3}	2
~~{2 3}~~	~~2~~
~~{2 5}~~	~~3~~
~~{3 5}~~	~~2~~

C_3

itemset
~~{2 3 5}~~

Scan D → L_3

itemset	sup
~~{2 3 5}~~	~~2~~

Constraint:

min{S.price } <= 1

Converting "Tough" Constraints

- Convert tough constraints into anti-monotone or monotone constraints by properly ordering items
- Examine C: avg(*S*.profit) ≥ 25
  - Order items in value-descending order
    - <*a, f, g, d, b, h, c, e*>
  - If an itemset *afb* violates C
    - So does *afbh, afb**
    - It becomes anti-monotone!

TDB (min_sup=2)

TID	Transaction
10	a, b, c, d, f
20	b, c, d, f, g, h
30	a, c, d, e, f
40	c, e, f, g

Item	Profit
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

Strongly Convertible Constraints

- avg(X) ≥ 25 is convertible anti-monotone w.r.t. item value descending order R: <*a, f, g, d, b, h, c, e*>
  - If an itemset *af* violates a constraint C, so does every itemset with *af* as prefix, such as *afd*
- avg(X) ≥ 25 is convertible monotone w.r.t. item value ascending order R^{-1}: <*e, c, h, b, d, g, f, a*>
  - If an itemset *d* satisfies a constraint *C*, so does itemsets *df* and *dfa*, which having *d* as a prefix
- Thus, avg(X) ≥ 25 is strongly convertible

Item	Profit
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

Can Apriori Handle Convertible Constraint?

- A convertible, not monotone nor anti-monotone nor succinct constraint (like avg(X)>=25) cannot be pushed deep into an Apriori mining algorithm:
  - Within the level wise framework, no direct pruning based on the constraint can be made
  - Itemset df (10,30) violates constraint C: avg(X)>=25
  - Since adf (40, 10, 30) satisfies C, Apriori needs df to assemble adf => it is clear that df cannot be pruned
- But it can be pushed into the frequent-pattern growth framework!

Item	Value
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

Mining With Convertible Constraints

- C: avg(X) >= 25, min_sup=2
- List items in every transaction in value descending order R: <a, f, g, d, b, h, c, e>
  - C is convertible anti-monotone w.r.t. R
- Scan TDB once
  - remove infrequent items
    - Item h is dropped
  - Itemsets a and f are good, …
- Projection-based mining
  - Imposing an appropriate order on item projection
  - Many tough constraints can be converted into (anti)-monotone

Item	Value
a	40
f	30
g	20
d	10
b	0
h	-10
c	-20
e	-30

TDB (min_sup=2)

TID	Transaction
10	a, f, d, b, c
20	f, g, d, b, c
30	a, f, d, c, e
40	f, g, h, c, e

Handling Multiple Constraints

- Different constraints may require different or even conflicting item-ordering
- If there exists an order *R* such that both C_1 and C_2 are convertible w.r.t. *R*, then there is no conflict between the two convertible constraints
- If there exists conflict on order of items
  - Try to satisfy one constraint first
  - Then using the order for the other constraint to mine frequent itemsets in the corresponding projected database

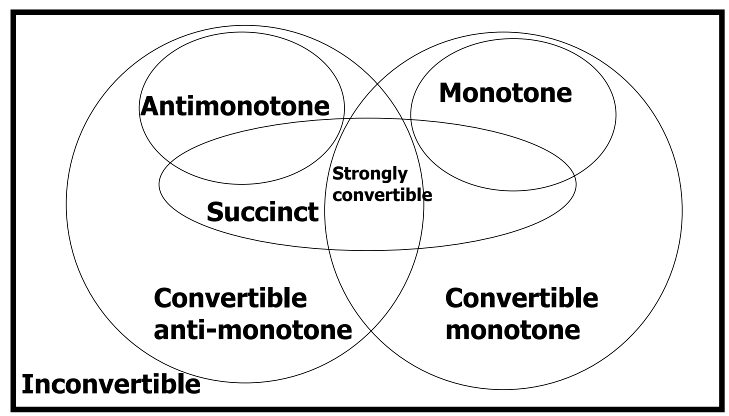
What Constraints Are Convertible?

Constraint	Convertible anti-monotone	Convertible monotone	Strongly convertible
avg(S) ≤ , ≥ v	Yes	Yes	Yes
median(S) ≤ , ≥ v	Yes	Yes	Yes
sum(S) ≤ v (where items could be of any value, v ≥ 0)	Yes	No	No
sum(S) ≤ v (where items could be of any value, v ≤ 0)	No	Yes	No
sum(S) ≥ v (where items could be of any value, v ≥ 0)	No	Yes	No
sum(S) ≥ v (where items could be of any value, v ≤ 0)	Yes	No	No
……			

Constraint-Based Mining—A General Picture

Constraint	Antimonotone	Monotone	Succinct
v ∈ S	no	yes	yes
S ⊇ V	no	yes	yes
S ⊆ V	yes	no	yes
min(S) ≤ v	no	yes	yes
min(S) ≥ v	yes	no	yes
max(S) ≤ v	yes	no	yes
max(S) ≥ v	no	yes	yes
count(S) ≤ v	yes	no	weakly
count(S) ≥ v	no	yes	weakly
sum(S) ≤ v (a ∈ S, a ≥ 0)	yes	no	no
sum(S) ≥ v (a ∈ S, a ≥ 0)	no	yes	no
range(S) ≤ v	yes	no	no
range(S) ≥ v	no	yes	no
avg(S) θ v, θ ∈ { =, ≤, ≥ }	convertible	convertible	no
support(S) ≥ ξ	yes	no	no
support(S) ≤ ξ	no	yes	no

A Classification of Constraints



Chapter 5: Mining Frequent Patterns, Association and Correlations

- Basic concepts and a road map
- Efficient and scalable frequent itemset mining methods
- Mining various kinds of association rules
- From association mining to correlation analysis
- Constraint-based association mining
- Summary

Frequent-Pattern Mining: Summary

- Frequent pattern mining—an important task in data mining
- Scalable frequent pattern mining methods
  - Apriori (Candidate generation & test)
  - Projection-based (FPgrowth, CLOSET+, ...)
  - Vertical format approach (CHARM, ...)
- Mining a variety of rules and interesting patterns
- Constraint-based mining
- Mining sequential and structured patterns
- Extensions and applications

Frequent-Pattern Mining: Research Problems

- Mining fault-tolerant frequent, sequential and structured patterns
  - Patterns allows limited faults (insertion, deletion, mutation)
- Mining truly interesting patterns
  - Surprising, novel, concise, …
- Application exploration
  - E.g., DNA sequence analysis and bio-pattern classification
  - "Invisible" data mining

Ref: Basic Concepts of Frequent Pattern Mining

- (Association Rules) R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. SIGMOD'93.
- (Max-pattern) R. J. Bayardo. Efficiently mining long patterns from databases. SIGMOD'98.
- (Closed-pattern) N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. ICDT'99.
- (Sequential pattern) R. Agrawal and R. Srikant. Mining sequential patterns. ICDE'95

Ref: Apriori and Its Improvements

- R. Agrawal and R. Srikant. Fast algorithms for mining association rules. VLDB'94.
- H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient algorithms for discovering association rules. KDD'94.
- A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. VLDB'95.
- J. S. Park, M. S. Chen, and P. S. Yu. An effective hash-based algorithm for mining association rules. SIGMOD'95.
- H. Toivonen. Sampling large databases for association rules. VLDB'96.
- S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket analysis. SIGMOD'97.
- S. Sarawagi, S. Thomas, and R. Agrawal. Integrating association rule mining with relational database systems: Alternatives and implications. SIGMOD'98.

Ref: Depth-First, Projection-Based FP Mining

- R. Agarwal, C. Aggarwal, and V. V. V. Prasad. A tree projection algorithm for generation of frequent itemsets. J. Parallel and Distributed Computing:02.
- J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. SIGMOD' 00.
- J. Pei, J. Han, and R. Mao. CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets. DMKD'00.
- J. Liu, Y. Pan, K. Wang, and J. Han. Mining Frequent Item Sets by Opportunistic Projection. KDD'02.
- J. Han, J. Wang, Y. Lu, and P. Tzvetkov. Mining Top-K Frequent Closed Patterns without Minimum Support. ICDM'02.
- J. Wang, J. Han, and J. Pei. CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets. KDD'03.
- G. Liu, H. Lu, W. Lou, J. X. Yu. On Computing, Storing and Querying Frequent Patterns. KDD'03.

Ref: Vertical Format and Row Enumeration Methods

- M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. Parallel algorithm for discovery of association rules. DAMI:97.
- Zaki and Hsiao. CHARM: An Efficient Algorithm for Closed Itemset Mining, SDM'02.
- C. Bucila, J. Gehrke, D. Kifer, and W. White. DualMiner: A Dual-Pruning Algorithm for Itemsets with Constraints. KDD'02.
- F. Pan, G. Cong, A. K. H. Tung, J. Yang, and M. Zaki , CARPENTER: Finding Closed Patterns in Long Biological Datasets. KDD'03.

Ref: Mining Multi-Level and Quantitative Rules

- R. Srikant and R. Agrawal. Mining generalized association rules. VLDB'95.
- J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. VLDB'95.
- R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. SIGMOD'96.
- T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. SIGMOD'96.
- K. Yoda, T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Computing optimized rectilinear regions for association rules. KDD'97.
- R.J. Miller and Y. Yang. Association rules over interval data. SIGMOD'97.
- Y. Aumann and Y. Lindell. A Statistical Theory for Quantitative Association Rules KDD'99.

Ref: Mining Correlations and Interesting Rules

- M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A. I. Verkamo. Finding interesting rules from large sets of discovered association rules. CIKM'94.
- S. Brin, R. Motwani, and C. Silverstein. Beyond market basket: Generalizing association rules to correlations. SIGMOD'97.
- C. Silverstein, S. Brin, R. Motwani, and J. Ullman. Scalable techniques for mining causal structures. VLDB'98.
- P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the Right Interestingness Measure for Association Patterns. KDD'02.
- E. Omiecinski. Alternative Interest Measures for Mining Associations. TKDE'03.
- Y. K. Lee, W.Y. Kim, Y. D. Cai, and J. Han. CoMine: Efficient Mining of Correlated Patterns. ICDM'03.

Ref: Mining Other Kinds of Rules

- R. Meo, G. Psaila, and S. Ceri. A new SQL-like operator for mining association rules. VLDB'96.
- B. Lent, A. Swami, and J. Widom. Clustering association rules. ICDE'97.
- A. Savasere, E. Omiecinski, and S. Navathe. Mining for strong negative associations in a large database of customer transactions. ICDE'98.
- D. Tsur, J. D. Ullman, S. Abitboul, C. Clifton, R. Motwani, and S. Nestorov. Query flocks: A generalization of association-rule mining. SIGMOD'98.
- F. Korn, A. Labrinidis, Y. Kotidis, and C. Faloutsos. Ratio rules: A new paradigm for fast, quantifiable data mining. VLDB'98.
- K. Wang, S. Zhou, J. Han. Profit Mining: From Patterns to Actions. EDBT'02.

Ref: Constraint-Based Pattern Mining

- R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. KDD'97.
- R. Ng, L.V.S. Lakshmanan, J. Han & A. Pang. Exploratory mining and pruning optimizations of constrained association rules. SIGMOD'98.
- M.N. Garofalakis, R. Rastogi, K. Shim: SPIRIT: Sequential Pattern Mining with Regular Expression Constraints. VLDB'99.
- G. Grahne, L. Lakshmanan, and X. Wang. Efficient mining of constrained correlated sets. ICDE'00.
- J. Pei, J. Han, and L. V. S. Lakshmanan. Mining Frequent Itemsets with Convertible Constraints. ICDE'01.
- J. Pei, J. Han, and W. Wang, Mining Sequential Patterns with Constraints in Large Databases, CIKM'02.

Ref: Mining Sequential and Structured Patterns

- R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. EDBT'96.
- H. Mannila, H Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. DAMI:97.
- M. Zaki. SPADE: An Efficient Algorithm for Mining Frequent Sequences. Machine Learning:01.
- J. Pei, J. Han, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. ICDE'01.
- M. Kuramochi and G. Karypis. Frequent Subgraph Discovery. ICDM'01.
- X. Yan, J. Han, and R. Afshar. CloSpan: Mining Closed Sequential Patterns in Large Datasets. SDM'03.
- X. Yan and J. Han. CloseGraph: Mining Closed Frequent Graph Patterns. KDD'03.

Ref: Mining Spatial, Multimedia, and Web Data

- K. Koperski and J. Han, Discovery of Spatial Association Rules in Geographic Information Databases, SSD'95.
- O. R. Zaiane, M. Xin, J. Han, Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs. ADL'98.
- O. R. Zaiane, J. Han, and H. Zhu, Mining Recurrent Items in Multimedia with Progressive Resolution Refinement. ICDE'00.
- D. Gunopulos and I. Tsoukatos. Efficient Mining of Spatiotemporal Patterns.   SSTD'01.

Ref: Mining Frequent Patterns in Time-Series Data

- B. Ozden, S. Ramaswamy, and A. Silberschatz. Cyclic association rules. ICDE'98.
- J. Han, G. Dong and Y. Yin, Efficient Mining of Partial Periodic Patterns in Time Series Database, ICDE'99.
- H. Lu, L. Feng, and J. Han. Beyond Intra-Transaction Association Analysis: Mining Multi-Dimensional Inter-Transaction Association Rules. TOIS:00.
- B.-K. Yi, N. Sidiropoulos, T. Johnson, H. V. Jagadish, C. Faloutsos, and A. Biliris. Online Data Mining for Co-Evolving Time Sequences. ICDE'00.
- W. Wang, J. Yang, R. Muntz. TAR: Temporal Association Rules on Evolving Numerical Attributes. ICDE'01.
- J. Yang, W. Wang, P. S. Yu. Mining Asynchronous Periodic Patterns in Time Series Data. TKDE'03.

Ref: Iceberg Cube and Cube Computation

- S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the computation of multidimensional aggregates. VLDB'96.
- Y. Zhao, P. M. Deshpande, and J. F. Naughton. An array-based algorithm for simultaneous multidi-mensional aggregates. SIGMOD'97.
- J. Gray, et al. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. DAMI: 97.
- M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani, and J. D. Ullman. Computing iceberg queries efficiently. VLDB'98.
- S. Sarawagi, R. Agrawal, and N. Megiddo. Discovery-driven exploration of OLAP data cubes. EDBT'98.
- K. Beyer and R. Ramakrishnan. Bottom-up computation of sparse and iceberg cubes. SIGMOD'99.

Ref: Iceberg Cube and Cube Exploration

- J. Han, J. Pei, G. Dong, and K. Wang, Computing Iceberg Data Cubes with Complex Measures. SIGMOD' 01.
- W. Wang, H. Lu, J. Feng, and J. X. Yu. Condensed Cube: An Effective Approach to Reducing Data Cube Size. ICDE'02.
- G. Dong, J. Han, J. Lam, J. Pei, and K. Wang. Mining Multi-Dimensional Constrained Gradients in Data Cubes. VLDB'01.
- T. Imielinski, L. Khachiyan, and A. Abdulghani. Cubegrades: Generalizing association rules. DAMI:02.
- L. V. S. Lakshmanan, J. Pei, and J. Han. Quotient Cube: How to Summarize the Semantics of a Data Cube. VLDB'02.
- D. Xin, J. Han, X. Li, B. W. Wah. Star-Cubing: Computing Iceberg Cubes by Top-Down and Bottom-Up Integration. VLDB'03.

Ref: FP for Classification and Clustering

- G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. KDD'99.
- B. Liu, W. Hsu, Y. Ma. Integrating Classification and Association Rule Mining. KDD'98.
- W. Li, J. Han, and J. Pei. CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules. ICDM'01.
- H. Wang, W. Wang, J. Yang, and P.S. Yu. Clustering by pattern similarity in large data sets. SIGMOD' 02.
- J. Yang and W. Wang. CLUSEQ: efficient and effective sequence clustering. ICDE'03.
- B. Fung, K. Wang, and M. Ester. Large Hierarchical Document Clustering Using Frequent Itemset. SDM'03.
- X. Yin and J. Han. CPAR: Classification based on Predictive Association Rules. SDM'03.

Ref: Stream and Privacy-Preserving FP Mining

- A. Evfimievski, R. Srikant, R. Agrawal, J. Gehrke. Privacy Preserving Mining of Association Rules. KDD'02.
- J. Vaidya and C. Clifton. Privacy Preserving Association Rule Mining in Vertically Partitioned Data. KDD'02.
- G. Manku and R. Motwani. Approximate Frequency Counts over Data Streams. VLDB'02.
- Y. Chen, G. Dong, J. Han, B. W. Wah, and J. Wang. Multi-Dimensional Regression Analysis of Time-Series Data Streams. VLDB'02.
- C. Giannella, J. Han, J. Pei, X. Yan and P. S. Yu. Mining Frequent Patterns in Data Streams at Multiple Time Granularities, Next Generation Data Mining:03.
- A. Evfimievski, J. Gehrke, and R. Srikant. Limiting Privacy Breaches in Privacy Preserving Data Mining. PODS'03.

Ref: Other Freq. Pattern Mining Applications

- Y. Huhtala, J. Kärkkäinen, P. Porkka, H. Toivonen. Efficient Discovery of Functional and Approximate Dependencies Using Partitions. ICDE'98.
- H. V. Jagadish, J. Madar, and R. Ng. Semantic Compression and Pattern Extraction with Fascicles. VLDB'99.
- T. Dasu, T. Johnson, S. Muthukrishnan, and V. Shkapenyuk. Mining Database Structure; or How to Build a Data Quality Browser. SIGMOD'02.