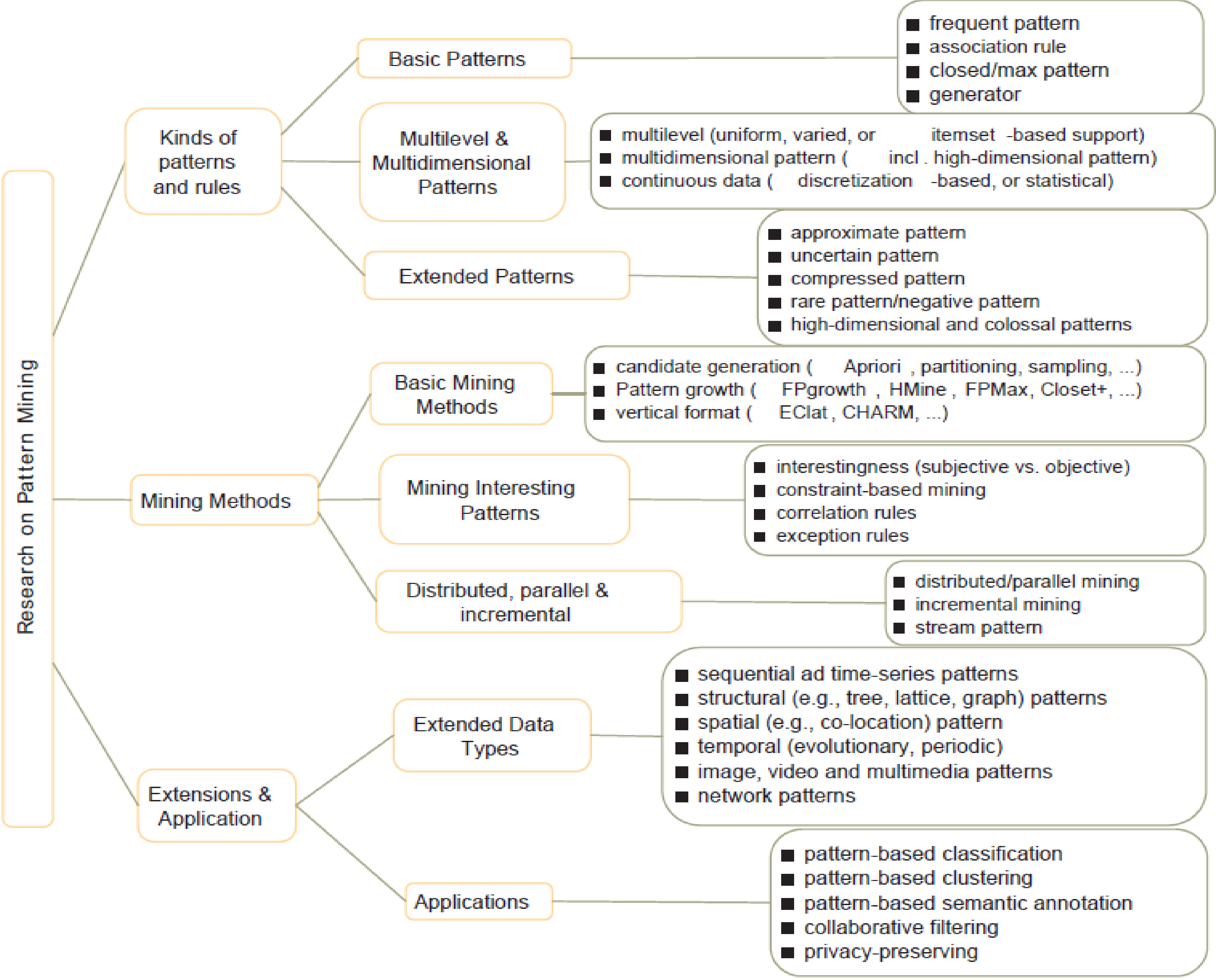
Mining Frequent Patterns & Associations: Advanced Methods

Excerpt from
"Data Mining: Concepts and Techniques", 3rd Ed.
Jiawei Han, Micheline Kamber, and Jian Pei
Chapter 7

Research on Pattern Mining: A Road Map



Outline

- Pattern Mining: A Road Map
- Pattern Mining in Multi-Level, Multi-Dimensional Space
- Constraint-Based Frequent Pattern Mining
- Mining High-Dimensional Data and Colossal Patterns
- Mining Compressed or Approximate Patterns
- Pattern Exploration and Application
- Summary

Outline

- Pattern Mining: A Road Map
- Pattern Mining in Multi-Level, Multi-Dimensional Space
  - Mining Multi-Level Association
  - Mining Multi-Dimensional Association
  - Mining Quantitative Association Rules
  - Mining Rare Patterns and Negative Patterns
- Constraint-Based Frequent Pattern Mining
- Mining High-Dimensional Data and Colossal Patterns
- Mining Compressed or Approximate Patterns
- Pattern Exploration and Application
- Summary

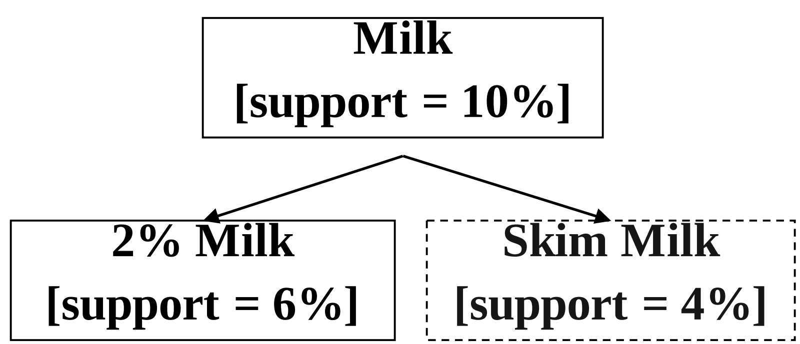
Mining Multiple-Level Association Rules

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

uniform support

reduced support

Level 1
min_sup = 5%



Level 1
min_sup = 5%

Level 2
min_sup = 5%

Level 2
min_sup = 3%

Multi-level Association: Flexible Support and Redundancy filtering

- Flexible min-support thresholds: Some items are more valuable but less frequent
  - Use non-uniform, group-based min-support
  - E.g., {diamond, watch, camera}: 0.05%; {bread, milk}: 5%; ...
- Redundancy Filtering: Some rules may be redundant due to "ancestor" relationships between items
  - milk $\Rightarrow$ wheat bread [support = 8%, confidence = 70%]
  - 2% milk $\Rightarrow$ wheat bread [support = 2%, confidence = 72%]

  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

Outline

- Pattern Mining: A Road Map
- Pattern Mining in Multi-Level, Multi-Dimensional Space
  - Mining Multi-Level Association
  - Mining Multi-Dimensional Association
  - Mining Quantitative Association Rules
  - Mining Rare Patterns and Negative Patterns
- Constraint-Based Frequent Pattern Mining
- Mining High-Dimensional Data and Colossal Patterns
- Mining Compressed or Approximate Patterns
- Pattern Exploration and Application
- Summary

Mining Multi-Dimensional Association

- Single-dimensional rules:

  buys(X, “milk”) $\Rightarrow$ buys(X, “bread”)

- Multi-dimensional rules: $\geq$ 2 dimensions or predicates
  - Inter-dimension assoc. rules (*no repeated predicates*)

    age(X,”19-25”) $\wedge$ occupation(X,“student”) $\Rightarrow$ buys(X, “coke”)
  - hybrid-dimension assoc. rules (*repeated predicates*)

    age(X,”19-25”) $\wedge$ buys(X, “popcorn”) $\Rightarrow$ 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

Outline

- Pattern Mining: A Road Map
- Pattern Mining in Multi-Level, Multi-Dimensional Space
  - Mining Multi-Level Association
  - Mining Multi-Dimensional Association
  - Mining Quantitative Association Rules
  - Mining Rare Patterns and Negative Patterns
- Constraint-Based Frequent Pattern Mining
- Mining High-Dimensional Data and Colossal Patterns
- Mining Compressed or Approximate Patterns
- Pattern Exploration and Application
- Summary

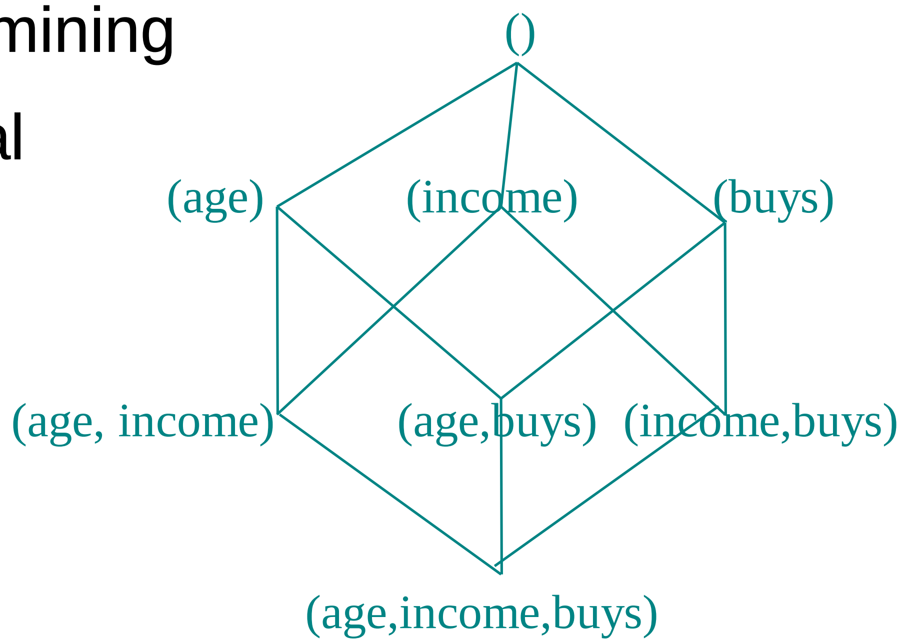
Mining Quantitative Associations

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

- Static discretization based on predefined concept hierarchies (data cube methods)
- Dynamic discretization based on data distribution (quantitative rules, e.g., Agrawal & Srikant@SIGMOD96)
- Clustering: Distance-based association (e.g., Yang & Miller@SIGMOD97)
  - One dimensional clustering then association
- 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



Outline

- Pattern Mining: A Road Map
- Pattern Mining in Multi-Level, Multi-Dimensional Space
  - Mining Multi-Level Association
  - Mining Multi-Dimensional Association
  - Mining Quantitative Association Rules
  - Mining Rare Patterns and Negative Patterns
- Constraint-Based Frequent Pattern Mining
- Mining High-Dimensional Data and Colossal Patterns
- Mining Compressed or Approximate Patterns
- Pattern Exploration and Application
- Summary

Negative and Rare Patterns

- Rare patterns: Very low support but interesting
  - E.g., buying Rolex watches
  - Mining: Setting individual-based or special group-based support threshold for valuable items
- Negative patterns
  - Since it is unlikely that one buys Ford Expedition (an SUV car) and Toyota Prius (a hybrid car) together, Ford Expedition and Toyota Prius are likely negatively correlated patterns
- Negatively correlated patterns that are also infrequent/rare tend to be more interesting than those that are frequent

Defining Negative Correlated Patterns (I)

- Definition 1 (support-based)
  - If itemsets X and Y are both frequent but rarely occur together, i.e.,

    $\sup(X \cup Y) < \sup(X) * \sup(Y)$
  - Then X and Y are negatively correlated
- Problem: A store sold two types of needles A and B, 100 transactions including each type, only one transaction containing both A and B.
  - When there are in total 200 transactions, we have

    $s(A \cup B) = 0.005$, $s(A) * s(B) = 0.25$, $s(A \cup B) < s(A) * s(B)$
  - When there are 10^5 transactions, we have

    $s(A \cup B) = 1/10^5$, $s(A) * s(B) = 1/10^3 * 1/10^3$, $s(A \cup B) > s(A) * s(B)$
  - Where is the problem? —Null transactions, i.e., the support-based definition is not null-invariant!

Defining Negative Correlated Patterns (II)

- Definition 2 (Kulzynski measure-based) If itemsets X and Y are frequent, but $(P(X|Y) + P(Y|X))/2 < \epsilon$, where ϵ is a negative-pattern threshold, then X and Y are negatively correlated.

- Ex. For the same needle package problem, when no matter there are 200 or 10^5 transactions, if $\epsilon = 0.01$, we have

$$(P(A|B) + P(B|A))/2 = (0.01 + 0.01)/2 < \epsilon$$

- Null invariant!

Outline

- Pattern Mining: A Road Map
- Pattern Mining in Multi-Level, Multi-Dimensional Space
- Constraint-Based Frequent Pattern Mining
- Mining High-Dimensional Data and Colossal Patterns
- Mining Compressed or Approximate Patterns
- Pattern Exploration and Application
- 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
  - Optimization: explores such constraints for efficient mining — constraint-based mining: constraint-pushing, similar to push selection first in DB query processing
  - Note: still find all the answers satisfying constraints

Constraints in Data Mining

- Knowledge type constraint:
  - correlation, association, etc.
- Data constraint — using SQL-like queries
  - find product pairs sold together in stores in Chicago this year
- Dimension/level constraint
  - in relevance to region, price, brand, customer category
- Interestingness constraint
  - strong rules: min_support $\geq$ 50%, min_confidence $\geq$ 60%
- Rule (or pattern) constraint
  - small sales (price < $10) triggers big sales (sum > $200)

Meta-Rule Guided Mining

- Meta-rule can be in the rule form with partially instantiated predicates and constants

  $P_1(X, Y) \wedge P_2(X, W) => buys(X, \text{“iPad”})$

- The resulting rule derived can be

  age(X, “15-25”) ^ profession(X, “student”) => buys(X, “iPad”)

- In general, it can be in the form of

  $P_1 \wedge P_2 \wedge \ldots \wedge P_l => Q_1 \wedge Q_2 \wedge \ldots \wedge Q_r$

- Method to find meta-rules
  - Find frequent (l+r) predicates (based on min-support threshold)
  - Push constants deeply when possible into the mining process (see the remaining discussions on constraint-push techniques)

Constraint-Based Frequent Pattern Mining

- Pattern space pruning constraints
  - Anti-monotonic: If constraint c is violated, its further mining can be terminated as it won't be satisfied with itemset extension
  - Monotonic: If c is satisfied, no need to check c again as it will for sure be satisfied with every itemset extension
  - Succinct: c must be satisfied, so one can start with the data sets satisfying c
  - Convertible: c is not monotonic nor anti-monotonic, but it can be converted into it if items in the transaction can be properly ordered
- Data space pruning constraint
  - Data succinct: Data space can be pruned at the initial pattern mining process
  - Data anti-monotonic: If a transaction t does not satisfy c, t can be pruned from its further mining

Pattern Space Pruning with Anti-Monotonicity Constraints

- A constraint C is *anti-monotone* if the super pattern satisfies C, all of its sub-patterns do so too
- In other words, *anti-monotonicity:* If an itemset S **violates** the constraint, so does any of its superset
- Ex. 1. *sum(S.price)* $\leq$ *v* is anti-monotone
- Ex. 2. range(S.profit) $\leq$ 15 is anti-monotone
  - Itemset *ab* violates C
  - So does every superset of *ab*
- Ex. 3. *sum(S.Price)* $\geq$ *v* is not anti-monotone
- Ex. 4. *support count* is anti-monotone: core property used in Apriori

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

Pattern Space Pruning with Monotonicity Constraints

- A constraint C is *monotone* if the pattern satisfies C, we do not need to check C in subsequent mining
- Alternatively, monotonicity: *If an itemset S* ***satisfies*** *the constraint, so does any of its superset*
- Ex. 1. $sum(S.Price) \geq v$ is monotone
- Ex. 2. $min(S.Price) \leq v$ is monotone
- Ex. 3. C: range(S.profit) $\geq$ 15
  - Itemset *ab* satisfies C
  - 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

Data Space Pruning with Data Anti-monotonicity

- A constraint c is *data anti-monotone* if for a pattern p cannot satisfy a transaction t under c, p's superset cannot satisfy t under c either
- The key for data anti-monotone is *recursive data reduction*
- Ex. 1. $sum(S.Price) \geq v$ is data anti-monotone
- Ex. 2. $min(S.Price) \leq v$ is data anti-monotone
- Ex. 3. C: $range(S.profit) \geq 25$ is data anti-monotone
  - Itemset {b, c}'s projected DB:
    - T10': {d, f, h}, T20': {d, f, g, h}, T30': {d, f, g}
  - since C cannot satisfy T10', T10' can be pruned

TDB (min_sup=2)

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

Item	Profit
a	40
b	0
c	-20
d	-15
e	-30
f	-10
g	20
h	-5

Pattern Space Pruning with 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
  - $min(S.Price) \geq v$ is succinct
  - $sum(S.Price) \geq v$ is not succinct

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

Constrained Apriori : 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:

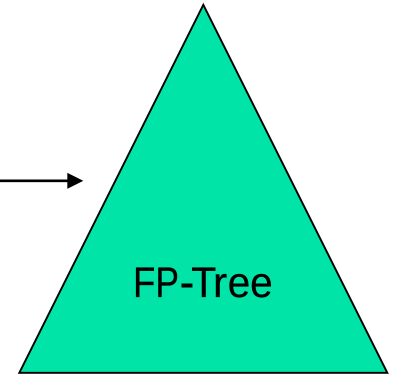
min{S.price } <= 1

Constrained FP-Growth: Push a Succinct Constraint Deep

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

Remove infrequent length 1

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



1-Projected DB

TID	Items
100	3
300	2 3 5

No Need to project on 2, 3, or 5

Constraint:

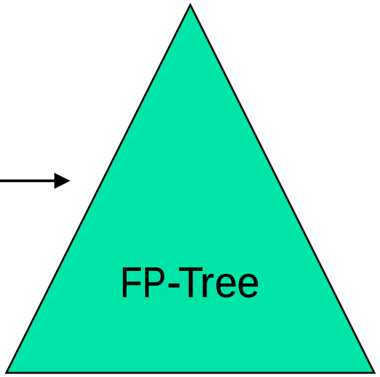
min{S.price } <= 1

Constrained FP-Growth: Push a Data Anti-monotonic Constraint Deep

Remove from data

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

TID	Items
100	1 3
300	1 3



Single branch, we are done

Constraint:

min{S.price } <= 1

Convertible Constraints: Ordering Data in Transactions

- Convert tough constraints into anti-monotone or monotone 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

Can Apriori Handle Convertible Constraints?

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

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

Can FP-Growth Handle 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* s.t. 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
$avg(S) \leq , \geq v$	Yes	Yes
$median(S) \leq , \geq v$	Yes	Yes
$sum(S) \leq v$ (items could be of any value, $v \geq 0$)	Yes	No
$sum(S) \leq v$ (items could be of any value, $v \leq 0$)	No	Yes
$sum(S) \geq v$ (items could be of any value, $v \geq 0$)	No	Yes
$sum(S) \geq v$ (items could be of any value, $v \leq 0$)	Yes	No
……		

Constraint-Based Mining — A General Picture

Constraint	Anti-monotone	Monotone	Succinct
$v \in S$	no	yes	yes
$S \supseteq V$	no	yes	yes
$S \subseteq V$	yes	no	yes
$min(S) \leq v$	no	yes	yes
$min(S) \geq v$	yes	no	yes
$max(S) \leq v$	yes	no	yes
$max(S) \geq v$	no	yes	yes
$count(S) \leq v$	yes	no	weakly
$count(S) \geq v$	no	yes	weakly
$sum(S) \leq v$ ($a \in S, a \geq 0$)	yes	no	no
$sum(S) \geq v$ ($a \in S, a \geq 0$)	no	yes	no
$range(S) \leq v$	yes	no	no
$range(S) \geq v$	no	yes	no
$avg(S)\ \theta\ v, \theta \in \{ =, \leq, \geq \}$	convertible	convertible	no
$support(S) \geq \xi$	yes	no	no
$support(S) \leq \xi$	no	yes	no

Outline

- Pattern Mining: A Road Map
- Pattern Mining in Multi-Level, Multi-Dimensional Space
- Constraint-Based Frequent Pattern Mining
- Mining High-Dimensional Data and Colossal Patterns
- Mining Compressed or Approximate Patterns
- Pattern Exploration and Application
- Summary

Mining Colossal Frequent Patterns

- F. Zhu, X. Yan, J. Han, P. S. Yu, and H. Cheng, "Mining Colossal Frequent Patterns by Core Pattern Fusion", ICDE'07.
- We have many algorithms, but can we mine large (i.e., colossal) patterns? — such as just size around 50 to 100? Unfortunately, not!
- Why not? — the curse of "downward closure" of frequent patterns
  - The "downward closure" property
    - Any sub-pattern of a frequent pattern is frequent.
  - Example. If $(a_1, a_2, \ldots, a_{100})$ is frequent, then $a_1, a_2, \ldots, a_{100}, (a_1, a_2), (a_1, a_3), \ldots, (a_1, a_{100}), (a_1, a_2, a_3), \ldots$ are all frequent! There are about 2^{100} such frequent itemsets!
  - No matter using breadth-first search (e.g., Apriori) or depth-first search (FPgrowth), we have to examine so many patterns
- Thus the downward closure property leads to explosion!

Colossal Patterns: A Motivating Example

Let's make a set of 40 transactions

T1 = 1 2 3 4 ..... 39 40

T2 = 1 2 3 4 ..... 39 40

: .

: .

: .

: .

T40=1 2 3 4 ..... 39 40

Then delete the items on the diagonal

T_1 = 2 3 4 ..... 39 40

T_2 = 1 3 4 ..... 39 40

: .

: .

: .

: .

T_{40}=1 2 3 4 39

Closed/maximal patterns may partially alleviate the problem but not really solve it: We often need to mine scattered large patterns!

Let the minimum support threshold σ= 20

There are $\binom{40}{20}$ frequent patterns of size 20

Each is closed and maximal

$$\# \text{ patterns} = \binom{n}{n/2} \approx \sqrt{2/\pi}\,\frac{2^n}{\sqrt{n}}$$

The size of the answer set is exponential to n

A Show of Colossal Pattern Mining!

T_1 = 2 3 4 ….. 39 40
T_2 = 1 3 4 ….. 39 40
: .
: .
: .
: .
T_{40}=1 2 3 4 …… 39
T_{41}= 41 42 43 ….. 79
T_{42}= 41 42 43 ….. 79
: .
: .
T_{60}= 41 42 43 … 79

Let the min-support threshold σ= 20

Then there are $\binom{40}{20}$ closed/maximal frequent patterns of size 20

However, there is only one with size greater than 20, (*i.e.,* colossal):

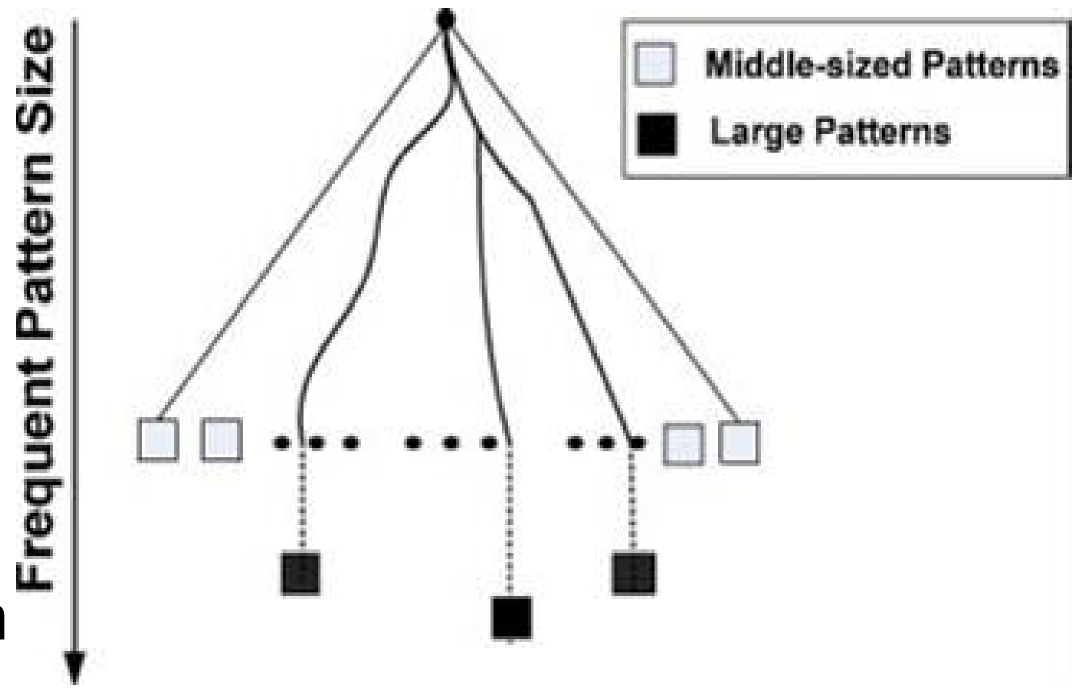
α= {41,42,…,79} of size 39

The existing fastest mining algorithms fail to complete running

Colossal Pattern Set: Small but Interesting

- It is often the case that only a small number of patterns are colossal, i.e., of large size

- Colossal patterns are usually attached with greater importance than those of small pattern sizes



Mining Colossal Patterns: Motivation and Philosophy

- Motivation: Many real-world tasks need mining colossal patterns
  - Micro-array analysis in bioinformatics (when support is low)
  - Biological sequence patterns
  - Biological/sociological/information graph pattern mining
- *No hope for completeness*
  - If the mining of mid-sized patterns is explosive in size, there is no hope to find colossal patterns efficiently by insisting “complete set” mining philosophy
- *Jumping out of the swamp of the mid-sized results*
  - What we may develop is a philosophy that may jump out of the swamp of mid-sized results that are explosive in size and jump to reach colossal patterns
- *Striving for mining almost complete colossal patterns*
  - The key is to develop a mechanism that may quickly reach colossal patterns and discover most of them

Methodology of Pattern-Fusion Strategy

- Pattern-Fusion traverses the tree in a bounded-breadth way
  - Always pushes down a frontier of a bounded-size candidate pool
  - Only a fixed number of patterns in the current candidate pool will be used as the starting nodes to go down in the pattern tree — thus avoids the exponential search space
- Pattern-Fusion identifies "shortcuts" whenever possible
  - Pattern growth is not performed by single-item addition but by leaps and bounded: agglomeration of multiple patterns in the pool
  - These shortcuts will direct the search down the tree much more rapidly towards the colossal patterns

Robustness of Colossal Patterns

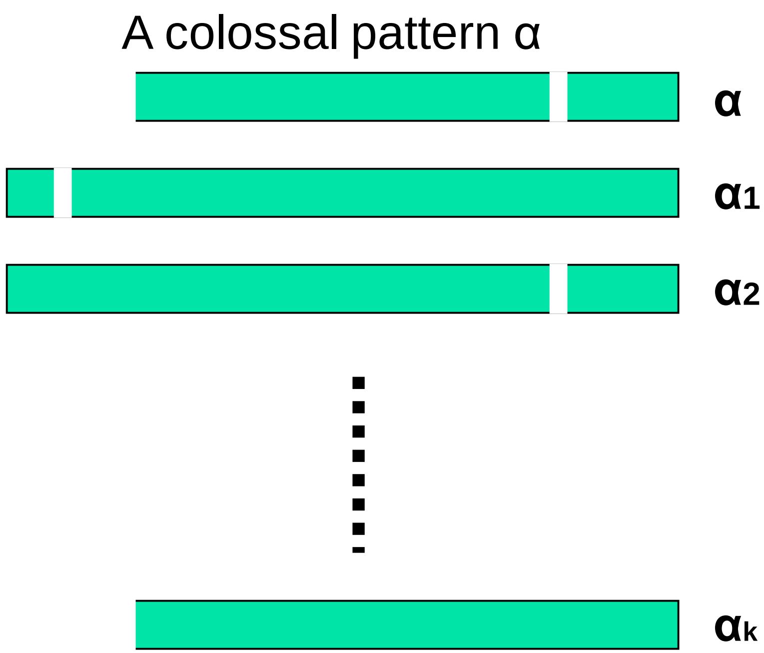
- Core Patterns: Intuitively, for a frequent pattern α, a subpattern β is a τ-core pattern of α if β shares a similar support set with α, i.e.,

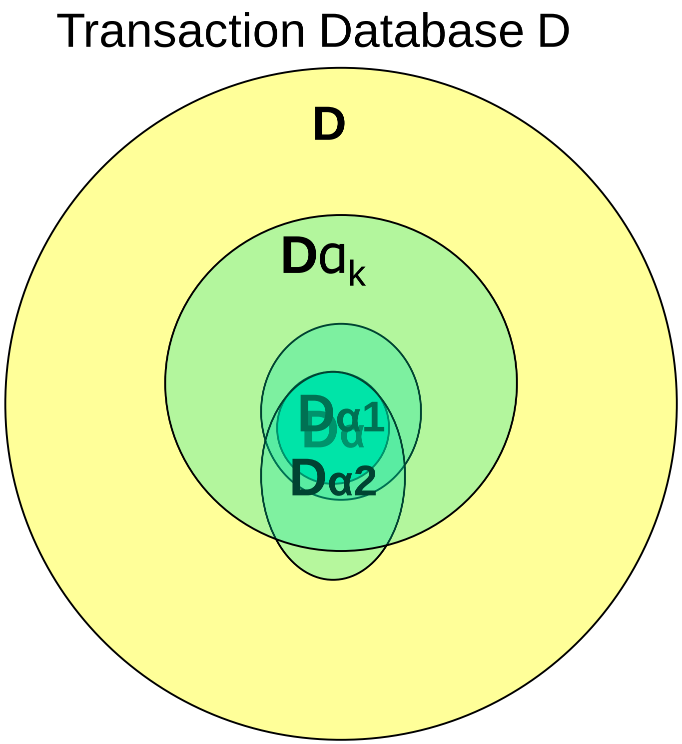
$$\frac{|D_\alpha|}{|D_\beta|} \geq \tau \qquad 0 < \tau \leq 1$$

  where τ is called the core ratio and D is dataset support for a given pattern

- Robustness of Colossal Patterns: A colossal pattern is (d,τ)-robust if d is the maximum number of items that can be removed from α while the remaining pattern is still a core pattern of α
  - Colossal patterns are robust, hence they tend to have much more core patterns than small patterns

Observation: Colossal Patterns and Core Patterns





Subpatterns α_1 to α_k cluster tightly around the colossal pattern α by sharing a similar support. We call such subpatterns *core patterns* of α

Example: Core Patterns

- A colossal pattern has far more core patterns than a small-sized pattern
- A colossal pattern has far more core descendants of a smaller size c
- A random draw from a complete set of pattern of size c would more likely to pick a core descendant of a colossal pattern
- A colossal pattern can be generated by merging a small subset of core patterns

Transaction (# of Ts)	Core Patterns ($\tau = 0.5$)
(abe) (100)	(abe), (ab), (be), (ae), (e)
(bcf) (100)	(bcf), (bc), (bf)
(acf) (100)	(acf), (ac), (af)
(abcef) (100)	(ab), (ac), (af), (ae), (bc), (bf), (be) (ce), (fe), (e), (abc), (abf), (abe), (ace), (acf), (afe), (bcf), (bce), (bfe), (cfe), (abcf), (abce), (bcfe), (acfe), (abfe), (abcef)

Robustness of Colossal Patterns

- (d,τ)-robustness: A pattern α is *(d, τ)-robust* if *d* is the maximum number of items that can be removed from α for the resulting pattern to remain a τ-core pattern of α
- For a (d,τ)-robust pattern α, it has $\Omega(2^d)$ core patterns
- Pattern distance: For patterns α and β, the pattern distance of α and β is defined to be

$$Dist(\alpha,\beta) = 1 - \frac{|D_\alpha \cap D_\beta|}{|D_\alpha \cup D_\beta|}$$

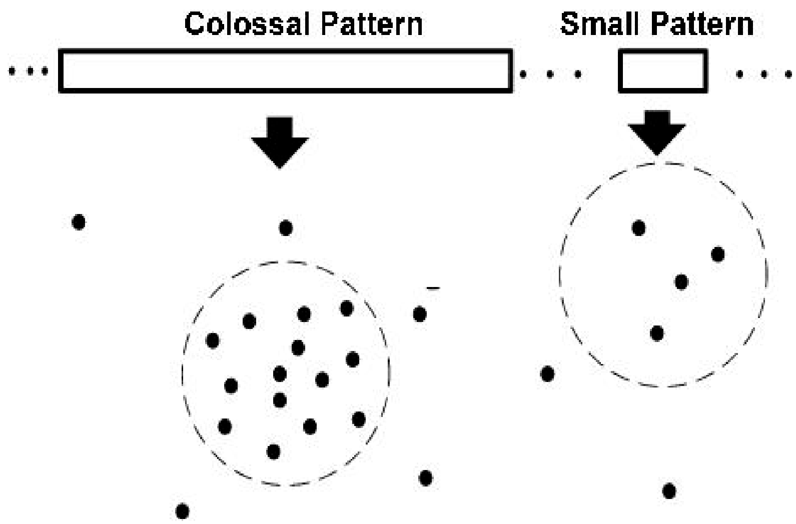
- If two patterns α and β are both core patterns of a same pattern, they would be bounded by a "ball" of a radius specified by their core ratio τ

$$Dist(\alpha,\beta) \le 1 - \frac{1}{2/\tau - 1} = r(\tau)$$

- Once we identify one core pattern, we will be able to find all the other core patterns by a bounding ball of radius r(τ)

Colossal Patterns Correspond to Dense Balls

- Due to their robustness, colossal patterns correspond to dense balls
  - Ω(2^d) in population
- A random draw in the pattern space will hit somewhere in the ball with high probability



Idea of Pattern-Fusion Algorithm

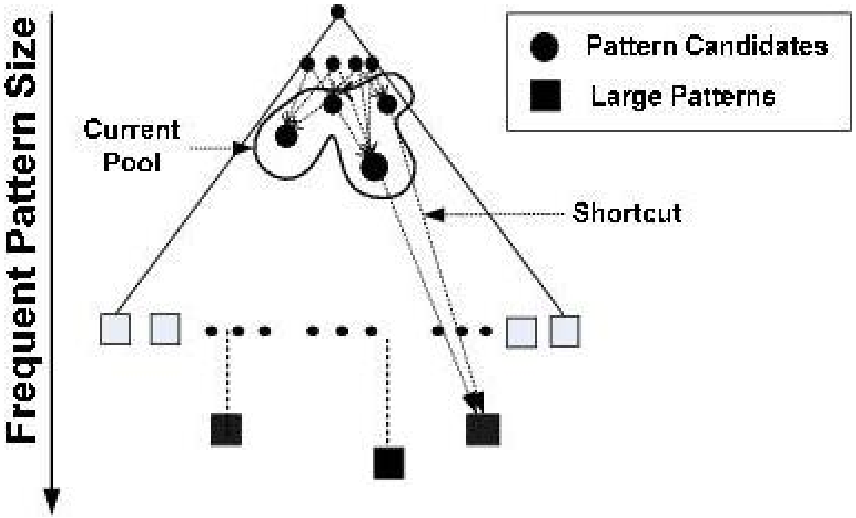
- Generate a complete set of frequent patterns up to a small size
- Randomly pick a pattern β, and β has a high probability to be a core-descendant of some colossal pattern α
- Identify all α's descendants in this complete set, and merge all of them — This would generate a much larger core-descendant of α
- In the same fashion, we select K patterns. This set of larger core-descendants will be the candidate pool for the next iteration

Pattern-Fusion: The Algorithm

- Initialization (Initial pool): Use an existing algorithm to mine all frequent patterns up to a small size, e.g., 3
- Iteration (Iterative Pattern Fusion):
  - At each iteration, k seed patterns are randomly picked from the current pattern pool
  - For each seed pattern thus picked, we find all the patterns within a bounding ball centered at the seed pattern
  - All these patterns found are fused together to generate a set of super-patterns. All the super-patterns thus generated form a new pool for the next iteration
- Termination: when the current pool contains no more than K patterns at the beginning of an iteration

Why Is Pattern-Fusion Efficient?

- A bounded-breadth pattern tree traversal
  - It avoids explosion in mining mid-sized ones
  - Randomness comes to help to stay on the right path
- Ability to identify "short-cuts" and take "leaps"
  - fuse small patterns together in one step to generate new patterns of significant sizes
  - Efficiency



Outline

- Pattern Mining: A Road Map
- Pattern Mining in Multi-Level, Multi-Dimensional Space
- Constraint-Based Frequent Pattern Mining
- Mining High-Dimensional Data and Colossal Patterns
- Mining Compressed or Approximate Patterns
- Pattern Exploration and Application
- Summary

Mining Compressed Patterns: δ-clustering

- Why compressed patterns?
  - too many, but less meaningful
- Pattern distance measure

$$D(P_1, P_2) = 1 - \frac{|T(P_1) \cap T(P_2)|}{|T(P_1) \cup T(P_2)|}$$

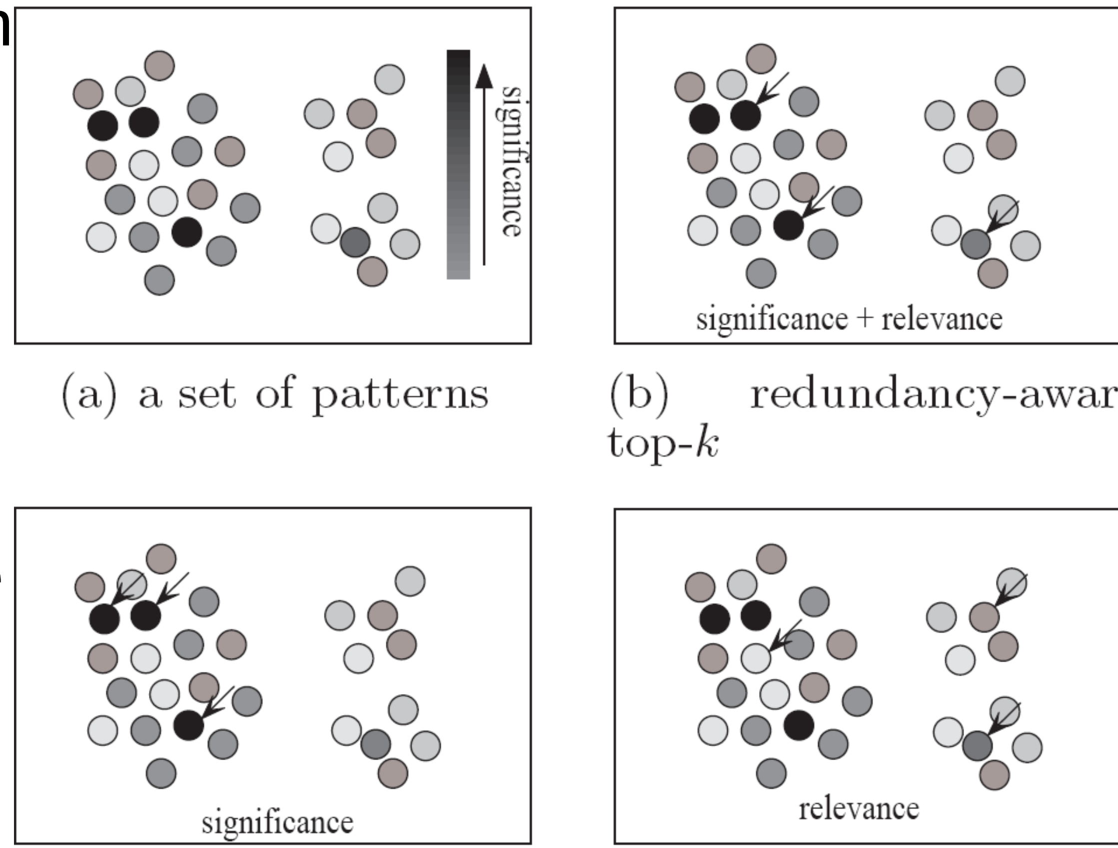
- δ-clustering: For each pattern P, find all patterns which can be expressed by P and their distance to P are within δ (δ-cover)
- All patterns in the cluster can be represented by P
- Xin et al., "Mining Compressed Frequent-Pattern Sets", VLDB'05

ID	Item-Sets	Support
P1	{38,16,18,12}	205227
P2	{38,16,18,12,17}	205211
P3	{39,38,16,18,12,17}	101758
P4	{39,16,18,12,17}	161563
P5	{39,16,18,12}	161576

- Closed frequent pattern
  - Report P1, P2, P3, P4, P5
  - Emphasize too much on support
  - no compression
- Max-pattern, P3: info loss
- A desirable output: P2, P3, P4

Redundancy-Aware Top-k Patterns

- Why redundancy-aware top-k patterns?
- Desired patterns: high significance & low redundancy
- Propose the MMS (Maximal Marginal Significance) for measuring the combined significance of a pattern set
- Xin et al., Extracting Redundancy-Aware Top-K Patterns, KDD'06



(a) a set of patterns

(b) redundancy-aware top-k

(c) traditional top-k

(d) summarization

Outline

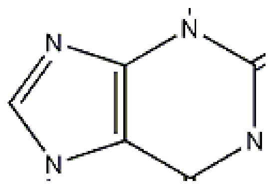
- Pattern Mining: A Road Map
- Pattern Mining in Multi-Level, Multi-Dimensional Space
- Constraint-Based Frequent Pattern Mining
- Mining High-Dimensional Data and Colossal Patterns
- Mining Compressed or Approximate Patterns
- Pattern Exploration and Application
- Summary

How to Understand and Interpret Patterns?

diaper beer

- Do they all make sense?
- What do they mean?
- How are they useful?

female sterile (2) tekele



morphological info. and simple statistics

Semantic Information

Not all frequent patterns are useful, only meaningful ones …

Annotate patterns with semantic information

A Dictionary Analogy

Word: “pattern” – from Merriam-Webster

Non-semantic info.

Main Entry: [1]**pat**·*tern*
Pronunciation: ˈpa-tərn
Function: *noun*
Etymology: Middle English *patron*, from Middle French, from Medieval Latin *patronus*
Date: 14th century

1 : a form or model proposed for imitation **:** EXEMPLAR
2 : something designed or used as a model for making things ⟨a dressmaker’s *pattern*⟩
3 : a model for making a mold into which molten metal is poured to form a casting
4 : an artistic, musical, literary, or mechanical design or form
5 : a natural or chance configuration ⟨frost *pattern*⟩ ⟨the *pattern* of events⟩
11 : a discernible coherent system based on the intended interrelationship of component parts ⟨foreign policy *patterns*⟩
12 : frequent or widespread incidence ⟨a *pattern* of dissent⟩
synonyms see MODEL
–**pat**·*terned* \-tərnd\ *adjective*
–**pat**·*ternless adjective*

A Dictionary Analogy

Word: “pattern” – from Merriam-Webster

Non-semantic info.

Main Entry: [1]**pat***tern*
Pronunciation: ˈpa-tərn
Function: *noun*
Etymology: Middle English *patron*, from Middle French, from
Latin *patronus*
Date: 14th century

Definitions indicating semantics

1 : a form or model proposed for imitation **:** EXEMPLAR
2 : something designed or used as a model for making things ⟨a dressmaker's *pattern*⟩
3 : a model for making a mold into which molten metal is poured to form a casting
4 : an artistic, musical, literary, or mechanical design or form
5 : a natural or chance configuration ⟨frost *pattern*⟩ ⟨the *pattern* of events⟩
11 : a discernible coherent system based on the intended interrelationship of component parts ⟨foreign policy *patterns*⟩
12 : frequent or widespread incidence ⟨a *pattern* of dissent⟩
synonyms see MODEL
–**pat***terned* \\-tərnd\\ *adjective*
–**pat***ternless* *adjective*

A Dictionary Analogy

Word: "pattern" – from Merriam-Webster

Non-semantic info.

Main Entry: ^{1}**pat***tern*
Pronunciation: ˈpa-tərn
Function: *noun*
Etymology: Middle English *patron*, from Middle French, from Medieval
Latin *patronus*
Date: 14th century

Definitions indicating semantics

1 : a form or model proposed for imitation : EXEMPLAR
2 : something designed or used as a model for making things ‹a dressmaker's *pattern*›
3 : a model for making a mold into which molten metal is poured to form a casting
4 : an artistic, musical, literary, or mechanical design or form
5 : a natural or chance configuration ‹frost *pattern*› ‹the *pattern* of ev…

Examples of Usage

11 : a discernible coherent system based on the intended interrelationship of component parts ‹foreign policy *patterns*›
12 : frequent or widespread incidence ‹a *pattern* of dissent›
synonyms see MODEL
–**pat***terned* \-tərnd\ *adjective*
–**pat***ternless* *adjective*

A Dictionary Analogy

Word: "pattern" – from Merriam-Webster

Main Entry: [1]**pat***tern*
Pronunciation: 'pa-tərn
Function: *noun*
Etymology: Middle English *patron*, from Middle French, from M...
Latin *patronus*
Date: 14th century

Non-semantic info.

Definitions indicating semantics

1 : a form
2 : somethi
pattern>
3 : a mode
casting
4 : an artis
5 : a natur
11 : a disc
component
12 : freque
synonyms
–**pat***tern*
–**pat***tern*

Main Entry: **pattern**
Function: *noun*

1
Synonyms MODEL 2, archetype, beau ideal, ensample, example, exemplar, ideal, mirror, paradigm, standard
Related Word original
2
Synonyms FIGURE 3, design, device, motif, motive
Related Word patterning
3
Synonyms ORDER 8, method, orderliness, plan, system
Related Word arrangement, constellation

ge

A Dictionary Analogy

Word: “pattern” – from Merriam-Webster

Main Entry: [1]pat·tern
Pronunciation: 'pa-tərn
Function: *noun*
Etymology: Middle English *patron*, from Middle French, from
Latin *patronus*
Date: 14th century

1 : a form
2 : somethi
pattern〉
3 : a mode
casting
4 : an artis
component
12 : freque
synonyms
–**pat**·*tern*
–**pat**·*tern*

Non-semantic info.

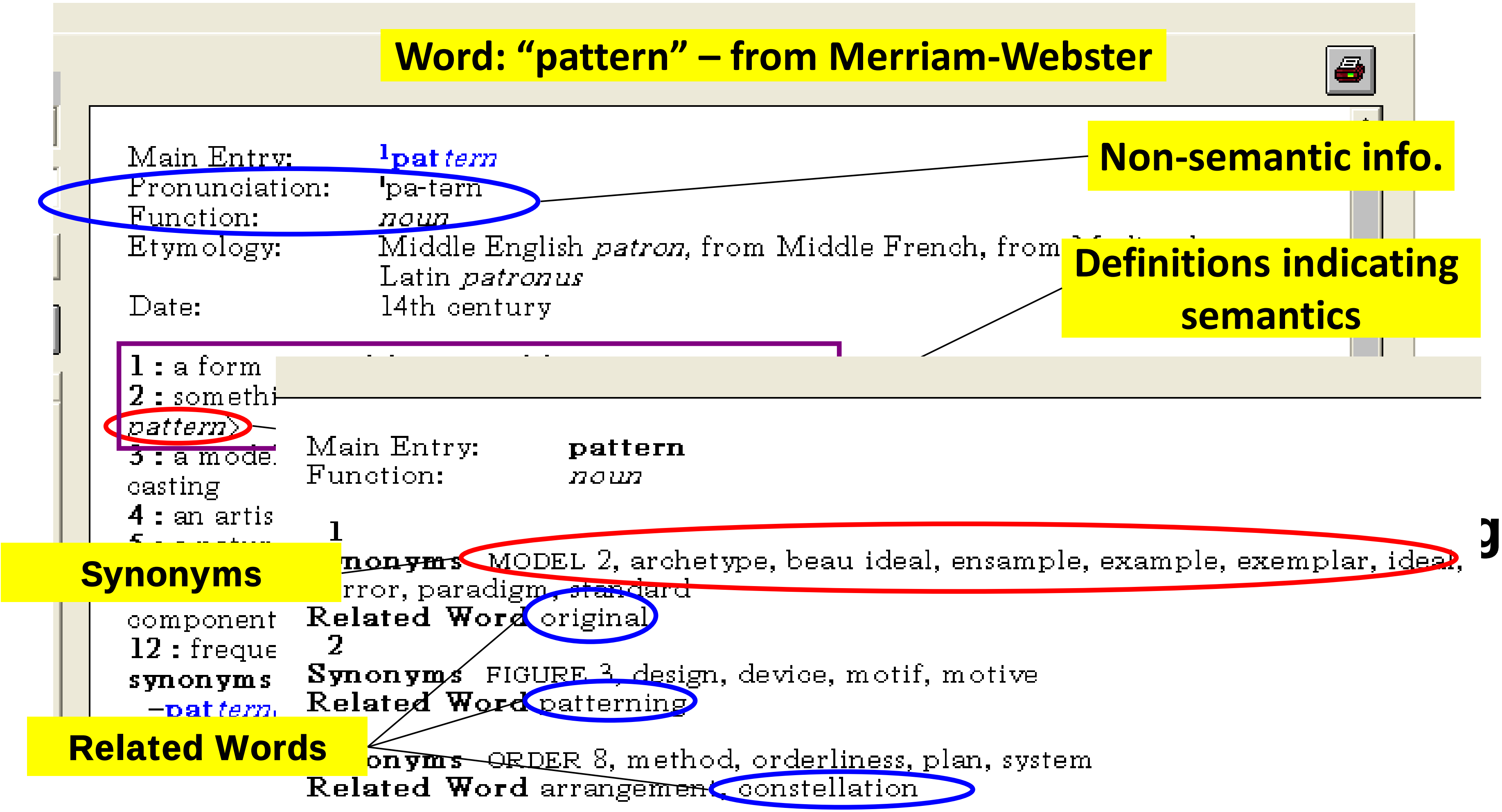
Definitions indicating semantics

Synonyms

Main Entry: **pattern**
Function: *noun*

1
Synonyms MODEL 2, archetype, beau ideal, ensample, example, exemplar, ideal,
mirror, paradigm, standard
Related Word original
2
Synonyms FIGURE 3, design, device, motif, motive
Related Word patterning
3
Synonyms ORDER 8, method, orderliness, plan, system
Related Word arrangement, constellation

A Dictionary Analogy



Semantic Analysis with Context Models

- Task1: Model the context of a frequent pattern

  Based on the Context Model…

- Task2: Extract strongest context indicators
- Task3: Extract representative transactions
- Task4: Extract semantically similar patterns

Annotating DBLP Co-authorship & Title Pattern

Database:

Authors	Title
X.Yan, P. Yu, J. Han	Substructure Similarity Search in Graph Databases
…	…
…	…

Frequent Patterns

P$_1$: { x_yan, j_han }

Frequent Itemset

P$_2$: "substructure search"

Semantic Annotations

Pattern	{ x_yan, j_han}
Non	Sup = …
CI	{p_yu}, graph pattern, …
Trans.	**gSpan: graph-base……**
SSPs	{ j_wang }, {j_han, p_yu}, …

Context Units

< { p_yu, j_han}, { d_xin }, … , "graph pattern", … "substructure similarity", … >

Pattern = {xifeng_yan, jiawei_han}

Annotation Results:

Context Indicator (CI)	graph; {philip_yu}; mine close; graph pattern; sequential pattern; …
Representative Transactions (Trans)	**> gSpan: graph-base substructure pattern mining;** **> mining close relational graph connect constraint; …**
Semantically Similar Patterns (SSP)	{jiawei_han, philip_yu}; {jian_pei, jiawei_han}; {jiong_yang, philip_yu, wei_wang}; …

Summary

- Roadmap: Many aspects & extensions on pattern mining
- Mining patterns in multi-level, multi dimensional space
- Mining rare and negative patterns
- Constraint-based pattern mining
- Specialized methods for mining high-dimensional data and colossal patterns
- Mining compressed or approximate patterns
- Pattern exploration and understanding: Semantic annotation of frequent patterns

Ref: Mining Multi-Level and Quantitative Rules

- Y. Aumann and Y. Lindell. A Statistical Theory for Quantitative Association Rules, KDD'99
- T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. SIGMOD'96.
- J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. VLDB'95.
- R.J. Miller and Y. Yang. Association rules over interval data. SIGMOD'97.
- R. Srikant and R. Agrawal. Mining generalized association rules. VLDB'95.
- R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. SIGMOD'96.
- K. Wang, Y. He, and J. Han. Mining frequent itemsets using support constraints. *VLDB'00*
- K. Yoda, T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Computing optimized rectilinear regions for association rules. KDD'97.

Ref: Mining Other Kinds of Rules

- F. Korn, A. Labrinidis, Y. Kotidis, and C. Faloutsos. Ratio rules: A new paradigm for fast, quantifiable data mining. VLDB'98
- 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
- B. Lent, A. Swami, and J. Widom. Clustering association rules. ICDE'97.
- R. Meo, G. Psaila, and S. Ceri. A new SQL-like operator for mining association rules. VLDB'96.
- 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.

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
- 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
- F. Bonchi, F. Giannotti, A. Mazzanti, and D. Pedreschi. ExAnte: Anticipated Data Reduction in Constrained Pattern Mining, PKDD'03
- F. Zhu, X. Yan, J. Han, and P. S. Yu, “gPrune: A Constraint Pushing Framework for Graph Pattern Mining”, PAKDD'07

Ref: Mining Sequential Patterns

- X. Ji, J. Bailey, and G. Dong. Mining minimal distinguishing subsequence patterns with gap constraints. ICDM'05
- H. Mannila, H Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. DAMI:97.
- 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.
- R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. EDBT’96.
- X. Yan, J. Han, and R. Afshar. CloSpan: Mining Closed Sequential Patterns in Large Datasets. SDM'03.
- M. Zaki. SPADE: An Efficient Algorithm for Mining Frequent Sequences. Machine Learning:01.

Mining Graph and Structured Patterns

- A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. *PKDD'00*
- M. Kuramochi and G. Karypis. Frequent Subgraph Discovery. ICDM'01.
- X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. ICDM'02
- X. Yan and J. Han. CloseGraph: Mining Closed Frequent Graph Patterns. KDD'03
- X. Yan, P. S. Yu, and J. Han. Graph indexing based on discriminative frequent structure analysis. *ACM TODS*, 30:960–993, 2005
- X. Yan, F. Zhu, P. S. Yu, and J. Han. Feature-based substructure similarity search. *ACM Trans. Database Systems*, 31:1418–1453, 2006

Ref: Mining Spatial, Spatiotemporal, Multimedia Data

- H. Cao, N. Mamoulis, and D. W. Cheung. Mining frequent spatiotemporal sequential patterns. ICDM'05
- D. Gunopulos and I. Tsoukatos. Efficient Mining of Spatiotemporal Patterns. SSTD'01
- K. Koperski and J. Han, Discovery of Spatial Association Rules in Geographic Information Databases, SSD'95
- H. Xiong, S. Shekhar, Y. Huang, V. Kumar, X. Ma, and J. S. Yoo. A framework for discovering co-location patterns in data sets with extended spatial objects. *SDM'04*
- J. Yuan, Y. Wu, and M. Yang. Discovery of collocation patterns: From visual words to visual phrases. *CVPR'07*
- O. R. Zaiane, J. Han, and H. Zhu, Mining Recurrent Items in Multimedia with Progressive Resolution Refinement. ICDE'00

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.
- J. Shieh and E. Keogh. iSAX: Indexing and mining terabyte sized time series. *KDD'08*
- 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
- L. Ye and E. Keogh. Time series shapelets: A new primitive for data mining. *KDD'09*

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.
- X. Yin and J. Han. CPAR: Classification based on Predictive Association Rules. SDM'03.
- H. Cheng, X. Yan, J. Han, and C.-W. Hsu, Discriminative Frequent Pattern Analysis for Effective Classification”, ICDE'07

Ref: Privacy-Preserving FP Mining

- A. Evfimievski, R. Srikant, R. Agrawal, J. Gehrke. Privacy Preserving Mining of Association Rules. KDD'02.
- A. Evfimievski, J. Gehrke, and R. Srikant. Limiting Privacy Breaches in Privacy Preserving Data Mining. PODS'03
- J. Vaidya and C. Clifton. Privacy Preserving Association Rule Mining in Vertically Partitioned Data. KDD'02

Mining Compressed Patterns

- D. Xin, H. Cheng, X. Yan, and J. Han. Extracting redundancy-aware top-k patterns. *KDD'06*
- D. Xin, J. Han, X. Yan, and H. Cheng. Mining compressed frequent-pattern sets. *VLDB'05*
- X. Yan, H. Cheng, J. Han, and D. Xin. Summarizing itemset patterns: A profile-based approach. *KDD'05*

Mining Colossal Patterns

- F. Zhu, X. Yan, J. Han, P. S. Yu, and H. Cheng. Mining colossal frequent patterns by core pattern fusion. ICDE'07
- F. Zhu, Q. Qu, D. Lo, X. Yan, J. Han. P. S. Yu, Mining Top-K Large Structural Patterns in a Massive Network. VLDB’11

Ref: FP Mining from Data Streams

- Y. Chen, G. Dong, J. Han, B. W. Wah, and J. Wang. Multi-Dimensional Regression Analysis of Time-Series Data Streams. VLDB'02.
- R. M. Karp, C. H. Papadimitriou, and S. Shenker. A simple algorithm for finding frequent elements in streams and bags. *TODS* 2003.
- G. Manku and R. Motwani. Approximate Frequency Counts over Data Streams. VLDB'02.
- A. Metwally, D. Agrawal, and A. El Abbadi. Efficient computation of frequent and top-k elements in data streams. *ICDT'05*

Ref: Freq. Pattern Mining Applications

- T. Dasu, T. Johnson, S. Muthukrishnan, and V. Shkapenyuk. Mining Database Structure; or How to Build a Data Quality Browser. SIGMOD'02
- M. Khan, H. Le, H. Ahmadi, T. Abdelzaher, and J. Han. DustMiner: Troubleshooting interactive complexity bugs in sensor networks., SenSys'08
- Z. Li, S. Lu, S. Myagmar, and Y. Zhou. CP-Miner: A tool for finding copy-paste and related bugs in operating system code. In Proc. 2004 Symp. Operating Systems Design and Implementation (OSDI'04)
- Z. Li and Y. Zhou. PR-Miner: Automatically extracting implicit programming rules and detecting violations in large software code. *FSE'05*
- D. Lo, H. Cheng, J. Han, S. Khoo, and C. Sun. Classification of software behaviors for failure detection: A discriminative pattern mining approach. *KDD'09*
- Q. Mei, D. Xin, H. Cheng, J. Han, and C. Zhai. Semantic annotation of frequent patterns. *ACM TKDD,* 2007.
- K. Wang, S. Zhou, J. Han. Profit Mining: From Patterns to Actions. EDBT’02.