

**Data Mining and Machine Learning**  
**Bioinspired computational methods**  
**Biological data mining**

---

## **Advanced Frequent Pattern Analysis**

*Francesco Marcelloni*

Department of Information Engineering  
University of Pisa  
ITALY

Some slides belong to the collection


Jiawei Han, Micheline Kamber, and Jian Pei  
University of Illinois at Urbana-Champaign  
Simon Fraser University

©2011 Han, Kamber, and Pei. All rights reserved.

1

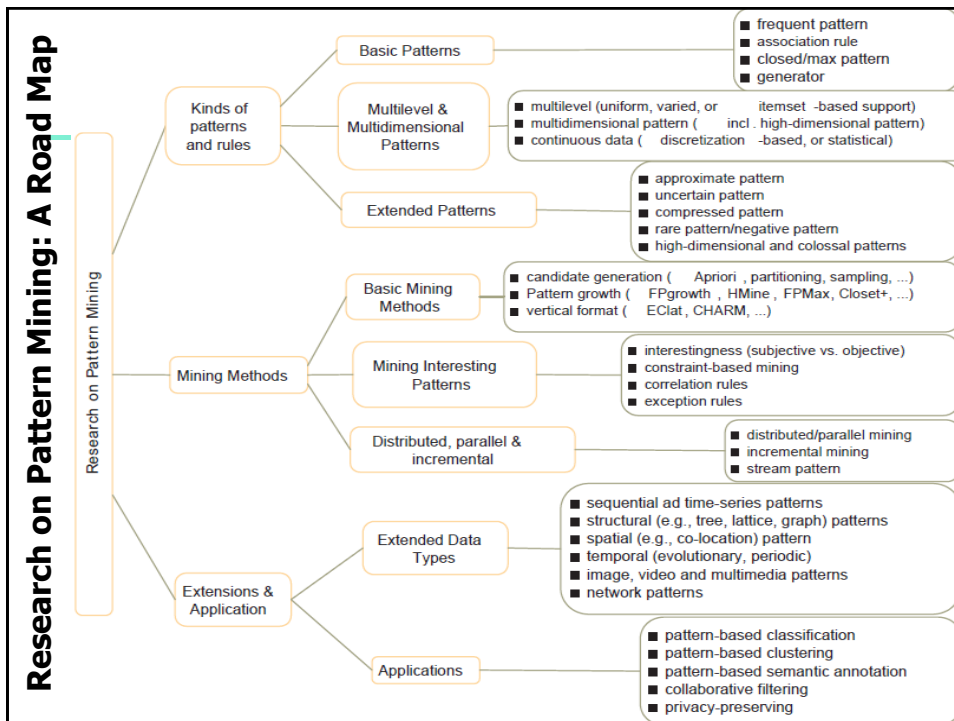
## **Advanced Frequent Pattern Mining**

---

- 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
- Summary

2

2



3

## Advanced Frequent Pattern Mining

- 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
- Summary

4

4

## 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@VLB'95, Han & Fu@VLDB'95)

uniform support

reduced support

Level 1  
min\_sup = 5%

Milk  
[support = 10%]

Level 1  
min\_sup = 5%

Level 2  
min\_sup = 5%

2% Milk  
[support = 6%]

Skim Milk  
[support = 4%]

Level 2  
min\_sup = 3%

5

5

## 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 and confidence are close to the "expected" value, based on the rule's ancestor

6

6

## Advanced Frequent Pattern Mining

- 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
- Summary

7

7

## Mining Multi-Dimensional Association

- Single-dimensional rules:  
 $\text{buys}(X, \text{"milk"}) \Rightarrow \text{buys}(X, \text{"bread"})$
- Multi-dimensional rules:  $\geq 2$  dimensions or predicates
  - Inter-dimensional assoc. rules (*no repeated predicates*)  
 $\text{age}(X, \text{"19-25"}) \wedge \text{occupation}(X, \text{"student"}) \Rightarrow \text{buys}(X, \text{"coke"})$
  - hybrid-dimensional assoc. rules (*repeated predicates*)  
 $\text{age}(X, \text{"19-25"}) \wedge \text{buys}(X, \text{"popcorn"}) \Rightarrow \text{buys}(X, \text{"coke"})$

8

8

## Advanced Frequent Pattern Mining

---

- 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
- Summary

9

9

## Mining Quantitative Associations

---

- **Categorical Attributes:** finite number of possible values, no ordering among values—data cube approach
- **Quantitative Attributes:** Numeric, implicit ordering among values— static discretization (predefined concept hierarchies) and dynamic discretization (binning and clustering)

10

10

## 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)

11

11

## Advanced Frequent Pattern Mining

- 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
- Summary

14

14

## 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 (correlated) 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 infrequent tend to be more interesting than those that are frequent

15

15

## Defining Negative Correlated Patterns (I)

- **Definition 1 (support-based)**
  - If itemsets  $X$  and  $Y$  are both frequent but rarely occur together, i.e.,  
 $\text{sup}(X \cup Y) < \text{sup}(X) * \text{sup}(Y)$   
then  $X$  and  $Y$  are negatively correlated
- **Problem:** A store sold two needle 100 packages  $A$  and  $B$ , 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!

16

16

## Defining Negative Correlated Patterns (II)

- **Definition 2 (negative itemset-based)**

- If X and Y are strongly negatively correlated, then

$$\sup(X \cup \bar{Y}) \times \sup(\bar{X} \cup Y) \gg \sup(X \cup Y) \times \sup(\bar{X} \cup \bar{Y})$$

- Also this definition suffers from the null-variant problem
- 200 transactions

$$\begin{aligned} \sup(A \cup \bar{B}) \times \sup(\bar{A} \cup B) &= 99/200 \times 99/200 = 0.245 \\ &\gg \sup(A \cup B) \times \sup(\bar{A} \cup \bar{B}) = 199/200 \times 1/200 \approx 0.005 \end{aligned}$$

- $10^6$  transactions

$$\begin{aligned} \sup(A \cup \bar{B}) \times \sup(\bar{A} \cup B) &= 99/10^6 \times 99/10^6 = 9.8 \times 10^{-9} \\ &\ll \sup(A \cup B) \times \sup(\bar{A} \cup \bar{B}) = 199/10^6 \times (10^6 - 199)/10^6 \approx 1.99 \times 10^{-4} \end{aligned}$$

17

17

## Defining Negative Correlated Patterns (II)

- **Definition 3 (Kulczynski measure-based)** If itemsets X and Y are frequent, but  $(P(X|Y) + P(Y|X))/2 < \epsilon$ , where  $\epsilon$  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  $\text{min\_sup} = 0.01\%$  and  $\epsilon = 0.02$ , we have

$$\begin{aligned} \sup(A) = \sup(B) &= 100/200 = 0.5 > 0.01\% \\ (P(B|A) + P(A|B))/2 &= (0.01 + 0.01)/2 < 0.02 \end{aligned}$$

$$\begin{aligned} \sup(A) = \sup(B) &= 100/10^6 = 0.01\% \geq 0.01\% \\ (P(B|A) + P(A|B))/2 &= (0.01 + 0.01)/2 < 0.02. \end{aligned}$$


18

18



## Advanced Frequent Pattern Mining

---

- 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
- Summary

19

19

## 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 —
  - Note: still find all the answers satisfying constraints, not finding some answers in “heuristic search”

20

20

## 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 this year
- **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%

21

21

## Meta-Rule Guided Mining

- A **metarule** forms a hypothesis regarding the relationships that the user is interested in probing or confirming
- Meta-rule can be in the rule form with partially instantiated predicates and constants
$$P_1(X, Y) \wedge P_2(X, W) \Rightarrow \text{buys}(X, \text{"iPad"})$$
- We are interested in determining which type of customer buys iPad.
- The resulting rule derived can be
$$\text{age}(X, \text{"15-25"}) \wedge \text{profession}(X, \text{"student"}) \Rightarrow \text{buys}(X, \text{"iPad"})$$
- In general, it can be in the form of
$$P_1 \wedge P_2 \wedge \dots \wedge P_l \Rightarrow Q_1 \wedge Q_2 \wedge \dots \wedge Q_r$$

22

22

## Constraint-Based Frequent Pattern Mining

Suppose our association mining query is “Find the patterns or rules about the sales of which cheap items (where the sum of the prices is less than \$10) may promote (i.e., appear in the same transaction) the sales of which expensive items (where the minimum price is \$50), shown in the sales in Chicago in 2010.”

This query contains the following four constraints: (1)  $\text{sum}(I.\text{price}) < \$10$ , where  $I$  represents the  $\text{item\_ID}$  of a cheap item; (2)  $\text{min}(J.\text{price}) \geq \$50$ , where  $J$  represents the  $\text{item\_ID}$  of an expensive item; (3)  $T.\text{city} = \text{Chicago}$ ; and (4)  $T.\text{year} = 2010$ , where  $T$  represents a  $\text{transaction\_ID}$ . For conciseness, we do not show the mining query explicitly here; however, the constraints’ context is clear from the mining query semantics. ■

- Dimension/level constraints and interestingness constraints can be applied after mining to filter out discovered rules, although it is generally more efficient and less expensive to use them during mining to help prune the search space.

23

23

## Constraint-Based Frequent Pattern Mining

- “How can we use rule constraints to prune the search space? More specifically, what kind of rule constraints can be ‘pushed’ deep into the mining process and still ensure the completeness of the answer returned for a mining query?”
- Pruning pattern search space
  - The former checks candidate patterns and decides whether a pattern can be pruned.
- Pruning data search space
  - checks the data set to determine whether the particular data piece will be able to contribute to the subsequent generation of satisfiable patterns (for a particular pattern) in the remaining mining process.

24

24

# Constraint-Based Frequent Pattern Mining

- Pattern space pruning constraints
  - **Anti-monotonic:** If constraint  $c$  is violated, its further mining can be terminated
  - **Monotonic:** If  $c$  is satisfied, no need to check  $c$  again
  - **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

25

25

## 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.  $\text{sum}(I.\text{price}) \leq v$  is **anti-monotone**
- Ex. 2.  $\text{range}(I.\text{profit}) \leq 15$  is **anti-monotone**
  - Itemset  $ab$  violates  $C$
  - So does every superset of  $ab$
- Ex. 3.  $\text{sum}(I.\text{price}) \geq v$  is not **anti-monotone**
- Ex. 4.  $\text{support count}$  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

26

26

## 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.  $\text{sum}(I.\text{Price}) \geq v$  is **monotone**
- Ex. 2.  $\text{min}(I.\text{Price}) \leq v$  is **monotone**
- Ex. 3.  $C: \text{range}(I.\text{profit}) \geq 15$ 
  - 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

27

27

## Pattern Space Pruning with Succinctness

- **Succinctness:**
    - We can enumerate all and only those sets that are **guaranteed to satisfy the constraint**. That is, if a rule constraint is succinct, we can directly generate precisely the sets that satisfy it, even before support counting begins.
    - This avoids the substantial overhead of the generate-and-test paradigm (constraints are *precounting prunable*).
      - $\text{min}(I.\text{Price}) \geq v$  is succinct
      - $\text{sum}(I.\text{Price}) \geq v$  is not succinct
- because we can explicitly and precisely generate all the itemsets that satisfy the constraint (there exists a precise formula, we do not need check the rule constraint)

28

28

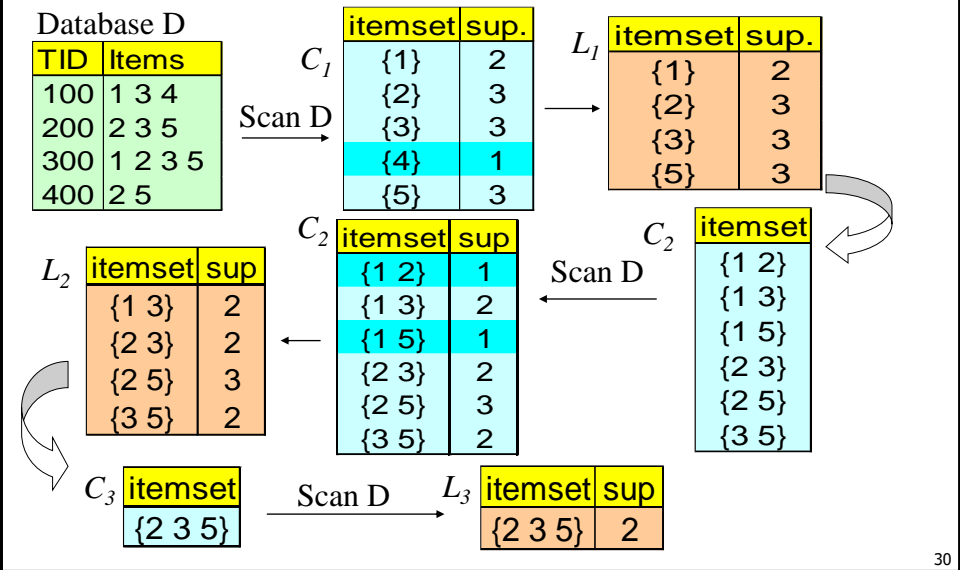
# Pattern Space Pruning with Succinctness

- Succinctness:**
  - Given  $A_1$ , the set of items satisfying a succinctness constraint  $C$ , then any set  $I$  satisfying  $C$  is based on  $A_1$ , i.e.,  $I$  contains a subset belonging to  $A_1$
  - Idea: Without looking at the transaction database, whether an itemset  $I$  satisfies constraint  $C$  can be determined based on the selection of items
- Optimization: If  $C$  is succinct,  $C$  is pre-counting pushable

29

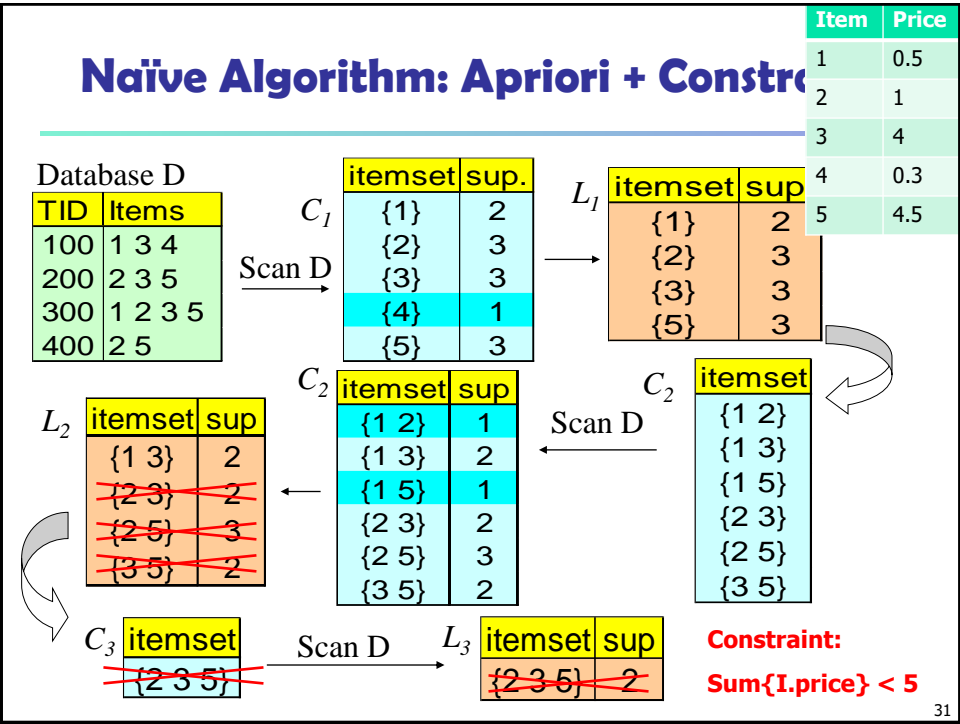
29

## The Apriori Algorithm - Example

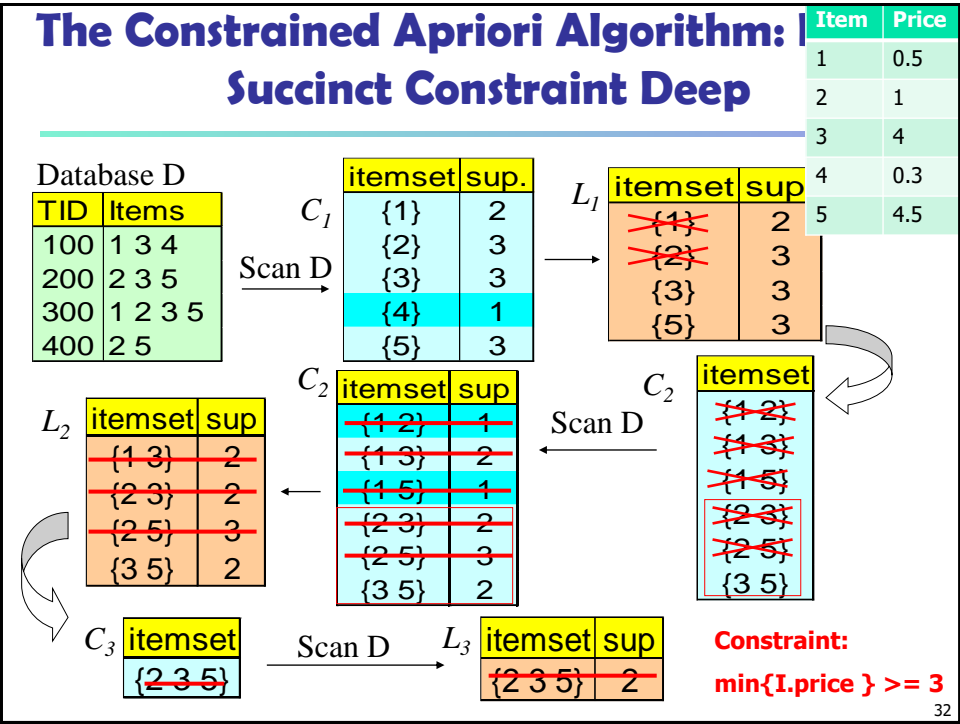


30

30



31



32

# Convertible Constraints: Ordering Data in Transactions

- Convert tough constraints into anti-monotone or monotone by properly ordering items
- Examine C:  $\text{avg}(S.\text{profit}) \geq 25$ 
  - Order items in value-descending order
    - $\langle a, f, g, d, b, h, c, e \rangle$
  - 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

33

33

# Strongly Convertible Constraints

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

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

34

34



## 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  $C$ :  $avg(X) \geq 25$
  - 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	Profit
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

35

35

## Can Apriori Handle Convertible Constraints?

- $C$ :  $avg(X) \geq 25$ ,  $min\_sup=2$
- List items in every transaction in value descending order  $R$ :  $\langle a, f, g, d, b, h, c, e \rangle$ 
  - $C$  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	Profit
a	40
b	0
c	-20
d	10
e	-30
f	30
g	20
h	-10

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

36

## Handling Multiple Constraints

- Different constraints may require different or even conflicting item-ordering
- If there exists an order  $R$  s.t. both  $C1$  and  $C2$  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 try to mine frequent itemsets in the corresponding projected database

37

37

## What Constraints Are Convertible?

Constraint	Convertible anti-monotone	Convertible monotone	Strongly convertible
$\text{avg}(S) \leq, \geq v$	Yes	Yes	Yes
$\text{median}(S) \leq, \geq v$	Yes	Yes	Yes
$\text{sum}(S) \leq v$ (items could be of any value, $v \geq 0$ )	Yes	No	No
$\text{sum}(S) \leq v$ (items could be of any value, $v \leq 0$ )	No	Yes	No
$\text{sum}(S) \geq v$ (items could be of any value, $v \geq 0$ )	No	Yes	No
$\text{sum}(S) \geq v$ (items could be of any value, $v \leq 0$ )	Yes	No	No
.....			

38

38

# Data Space Pruning with Data Anti-monotonicity

TDB (min\_sup=2)

- A constraint  $c$  is *data anti-monotone* if for a pattern  $p$  which cannot be satisfied by a transaction  $t$  under  $c$ ,  $p$ 's superset cannot be satisfied by  $t$  under

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

Note that such pruning cannot be done at the beginning of the mining because at that time, we do not know yet if the total sum of the prices of all the items in  $T_i$  will be over \$100 (e.g., we may have  $i_3.price = \$80$ ). However, during the iterative mining process, we may find some items (e.g.,  $i_3$ ) that are not frequent with  $S$  in the transaction data set, and thus they would be pruned. Therefore, such checking and pruning should be enforced at each iteration to reduce the data search space.

- Ex. 2.  $min(I.Price) \leq v$  is data anti-monotone
- Ex. 3.  $C: range(I.profit) \geq 25$  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 be satisfied by  $T10'$ ,  $T10'$  can be pruned

D	U
c	-20
d	-15
e	-30
f	-10
g	20
h	-5

39

39

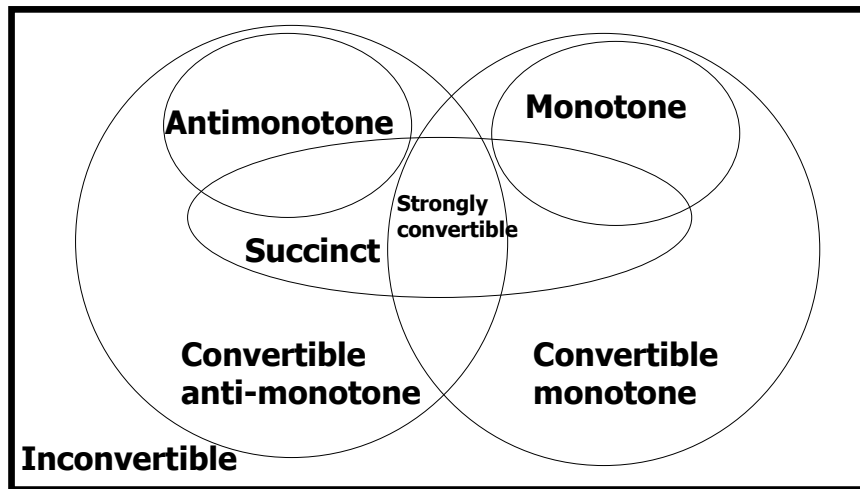
# Constraint-Based Mining — A General Picture

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

40

40

## A Classification of Constraints



41

## Advanced Frequent Pattern Mining

- 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
- Summary

42

# 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  $(a_1, a_2, \dots, a_{100})$  is frequent, then  $a_1, a_2, \dots, a_{100}, (a_1, a_2), (a_1, a_3), \dots, (a_1, a_{100}), (a_1, a_2, a_3), \dots$  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!**

43

43

# Colossal Patterns: A Motivating Example

Let’s make a set of 40 transactions

T<sub>1</sub> = 1 2 3 4 ..... 39 40  
 T<sub>2</sub> = 1 2 3 4 ..... 39 40  
 :  
 :  
 :  
 :  
 :  
 :  
 T<sub>40</sub>=1 2 3 4 ..... 39 40

Then delete the items on the diagonal

T<sub>1</sub> = 2 3 4 ..... 39 40  
 T<sub>2</sub> = 1 3 4 ..... 39 40  
 :  
 :  
 :  
 :  
 :  
 :  
 T<sub>40</sub>=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  $\sigma= 20$

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

Each is closed and maximal

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

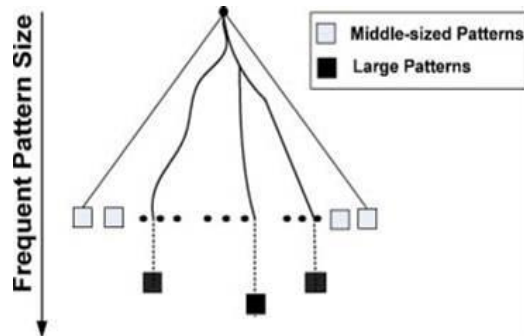
The size of the answer set is exponential to n

44

44

## 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



45

45

## 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

46

46

## Alas, A Show of Colossal Pattern Mining!

$T_1 = 2\ 3\ 4\ \dots\ 39\ 40$

$T_2 = 1\ 3\ 4\ \dots\ 39\ 40$

:

:

:

:

$T_{40} = 1\ 2\ 3\ 4\ \dots\ 39$

$T_{41} = 41\ 42\ 43\ \dots\ 79$

$T_{42} = 41\ 42\ 43\ \dots\ 79$

:

:

$T_{60} = 41\ 42\ 43\ \dots\ 79$

Let the min-support threshold  $\sigma = 20$

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):

$\alpha = \{41, 42, \dots, 79\}$  of size 39

The existing fastest mining algorithms (*e.g.*, FPClose, LCM) fail to complete running

The algorithm we analyse outputs this colossal pattern in seconds

47

47

## Methodology of Pattern-Fusion Strategy

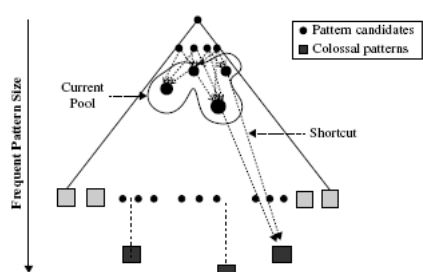
- Pattern-Fusion traverses the tree in a bounded-breadth way
  - Always pushes down a frontier of a bounded-size candidate pool
  - Only a fixed number of patterns in the current candidate pool will be used as the starting nodes to go down in the pattern tree — thus avoids the exponential search space

48

48

# Methodology of Pattern-Fusion Strategy

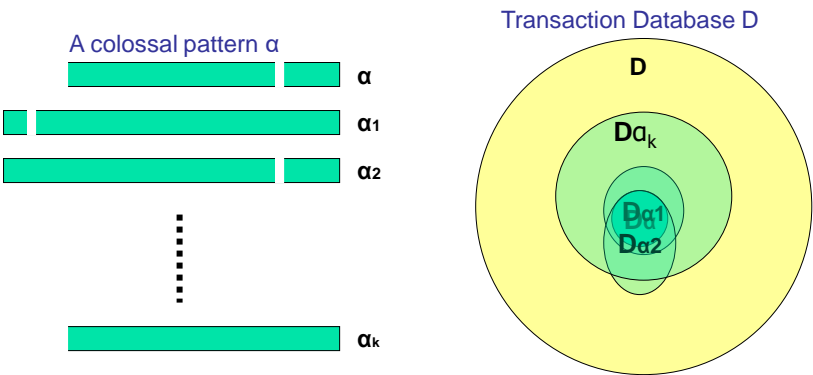
- Pattern-Fusion identifies “shortcuts” whenever possible
  - Pattern growth is not performed by single-item addition but by leaps and bounded: agglomeration of multiple patterns in the pool
  - These shortcuts will direct the search down the tree much more rapidly towards the colossal patterns



49

49

# Observation: Colossal Patterns and Core Patterns



Subpatterns  $\alpha_1$  to  $\alpha_k$  cluster tightly around the colossal pattern  $\alpha$  by sharing a similar support. We call such subpatterns *core patterns* of  $\alpha$

50

50



## Robustness of Colossal Patterns

### ■ Core Patterns

Intuitively, for a frequent pattern  $\alpha$ , a subpattern  $\beta$  is a  $\tau$ -core pattern of  $\alpha$  if  $\beta$  shares a similar support set with  $\alpha$ , i.e.,

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

where  $|D_\alpha|$  is the number of patterns containing  $\alpha$  and  $\tau$  is called the core ratio

### ■ Robustness of Colossal Patterns

A colossal pattern is robust in the sense that it tends to have much more core patterns than small patterns

51

51

## Robustness of Colossal Patterns

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

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

- If two patterns  $\alpha$  and  $\beta$  are both core patterns of a same pattern, they would be bounded by a "ball" of a radius specified by their core ratio  $\tau$

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

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

52

52

## Example: Core Patterns

A colossal pattern has far more core patterns than a small sized pattern. we set  $\tau = 0.5$ , then  $(ab)$  is a core pattern of  $\alpha_1$  because  $(ab)$  is contained only by  $\alpha_1$  and  $\alpha_4$ . Therefore,  $\frac{|D_{\alpha_1}|}{|D_{(ab)}|} = \frac{100}{200} \geq \tau$ .  $\alpha_1$  is  $(2, 0.5)$ -robust while  $\alpha_4$  is  $(4, 0.5)$ -robust. The table also shows that larger patterns (e.g.,  $(abcef)$ ) have far more core patterns than smaller ones (e.g.,  $(bcf)$ ).

- A colossal pattern can be generated by merging a set of core patterns

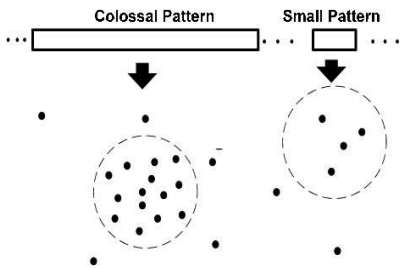
Transaction (# of Ts)	Core Patterns ( $\tau = 0.5$ )
(abe) (100)	(abe), (ab), (be), (ae), (e)
(bcf) (100)	(bcf), (bc), (bf)
(acf) (100)	(acf), (ac), (af)
(abcef) (100)	(ab), (ac), (af), (ae), (bc), (bf), (be), (ce), (fe), (e), (abc), (abf), (abe), (ace), (acf), (afe), (bcf), (bce), (bfe), (cfe), (abcf),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

53

53

## Colossal Patterns Correspond to Dense Balls

- Due to their robustness, colossal patterns correspond to dense balls
  - $\Omega(2^d)$  in population
- A random draw in the pattern space will hit somewhere in the ball with high probability
- In the previous example, the probability of drawing a descendant of  $abcef$  is 0.9.



54

54

## Idea of Pattern-Fusion Algorithm

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

55

55

## 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

56

56

## Pattern-Fusion: The Algorithm

---

### Algorithm 1 Main Algorithm

---

Input: Initial pool  $InitPool$ , Core ratio  $\tau$   
 Maximum number of patterns to mine  $K$ ,  
 Output: Set of frequent patterns  $S$

- 1: **do**
- 2:      $S \leftarrow \text{Pattern\_Fusion}(InitPool, K, \tau)$ ;
- 3:      $InitPool \leftarrow S$
- 4: **while**  $|S| > K$
- 5: **return**  $S$ ;

---

57

57

## Pattern-Fusion: The Algorithm

When the number of such  $\beta$  exceeds a threshold, which is determined by the system, we resort to a sampling technique to decide the set of  $\beta$  to retain.

Output: Set of patterns  $S$

- 1:  $S \leftarrow \emptyset; T \leftarrow \emptyset$ ;
- 2: **for**  $i = 1$  **to**  $K$
- 3:     Randomly draw a seed  $\alpha$  from  $InitPool$ ;
- 4:      $T \leftarrow T \cup \{\alpha\}$
- 5:     **for each**  $\beta \in InitPool$
- 6:         **if**  $\text{Dist}(\alpha, \beta) \leq r(\tau)$
- 7:             Record  $\beta$  in  $\alpha.CoreList$
- 8:     **for each**  $\alpha \in T$
- 9:          $S \leftarrow S \cup \text{Fusion}(\alpha.CoreList)$ ;
- 10: **return**  $S$ ;

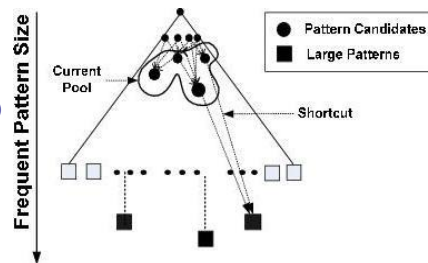
---

58

58

## 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



59

59

## Pattern-Fusion Leads to Good Approximation

- Gearing toward colossal patterns
  - The larger the pattern, the greater the chance it will be generated
- Catching outliers
  - The more distinct the pattern, the greater the chance it will be generated

60

60

## Experimental Setting

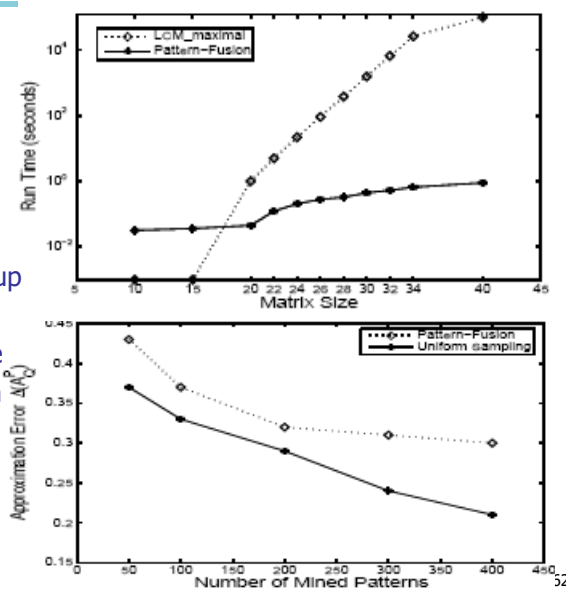
- Synthetic data set
  - $\text{Diag}_n$  an  $n \times (n-1)$  table where  $i^{\text{th}}$  row has integers from 1 to  $n$  except  $i$ . Each row is taken as an itemset.  $\text{min\_support}$  is  $n/2$ .
- Real data set
  - Replace: A program trace data set collected from the "replace" program, widely used in software engineering research
  - ALL: A popular gene expression data set, a clinical data on ALL-AML leukemia ([www.broad.mit.edu/tools/data.html](http://www.broad.mit.edu/tools/data.html)).
    - Each item is a column, representing the activity level of gene/protein in the sample
    - Frequent pattern would reveal important correlation between gene expression patterns and disease outcomes

61

61

## Experiment Results on $\text{Diag}_n$

- LCM run time increases exponentially with pattern size  $n$
- Pattern-Fusion finishes efficiently
- The approximation error of Pattern-Fusion (with  $\text{min\_sup}$  20) in comparison with the complete set is rather close to uniform sampling (which randomly picks  $K$  patterns from the complete answer set)



62

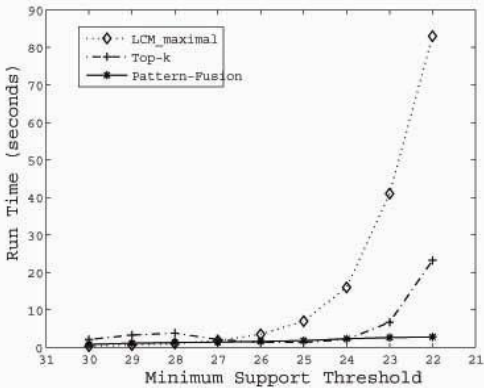
## Experimental Results on ALL

- ALL: A popular gene expression data set with 38 transactions, each with 866 columns
  - There are 1736 items in total
  - The table shows a high frequency threshold of 30

Pattern Size	110	107	102	91	86	84	83
The complete set	1	1	1	1	1	2	6
Pattern-Fusion	1	1	1	1	1	1	4

Pattern Size	82	77	76	75	74	73	71
The complete set	1	2	1	1	1	2	1
Pattern-Fusion	0	2	0	1	1	1	1



63

## Experimental Results on REPLACE

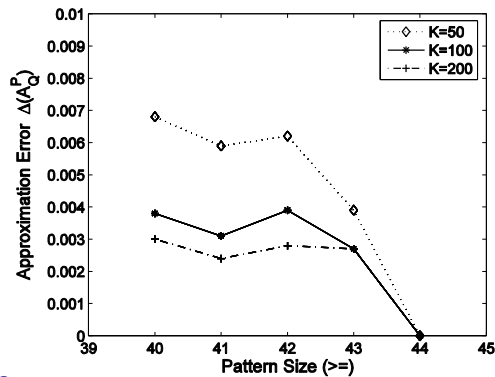
- REPLACE
  - A program trace data set, recording 4395 calls and transitions
  - The data set contains 4395 transactions with 57 items in total
  - AIM: identify frequent, and accordingly normal, program execution structures.
  - With support threshold of 0.03, the largest patterns are of size 44
  - They are all discovered by Pattern-Fusion with different settings of K and  $\tau$ , when started with an initial pool of 20948 patterns of size  $\leq 3$

64

64

## Experimental Results on REPLACE

- Approximation error when compared with the complete mining result
- Example. Out of the total 98 patterns of size  $\geq 42$ , when  $K=100$ , Pattern-Fusion returns 80 of them
- A good approximation to the colossal patterns in the sense that any pattern in the complete set is on average at most 0.17 items away from one of these 80 patterns



65

65

## Advanced Frequent Pattern Mining

- 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
- Summary



66

66



# Mining Compressed Patterns: $\delta$ -clustering

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

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

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

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

67

67

# Redundancy-Award Top-k Patterns

- Mining the top-k most frequent patterns is a strategy for reducing the number of patterns returned during mining.
- However, in many cases, frequent patterns are not mutually independent but often clustered in small regions.
- This is somewhat like finding 20 population centers in the world, which may result in cities clustered in a small number of countries rather than evenly distributed across the globe.

68

68

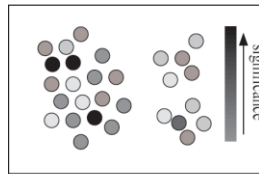
## Redundancy-Award Top-k Patterns

- Why redundancy-aware top-k patterns?

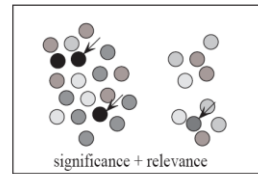
- Desired patterns: high significance & low redundancy

- Propose the MMS (Maximal Marginal Significance) for measuring the combined significance of a pattern set

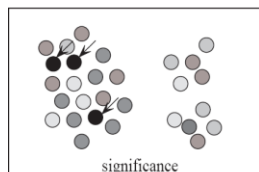
- Xin et al., Extracting Redundancy-Aware Top-K Patterns, KDD'06



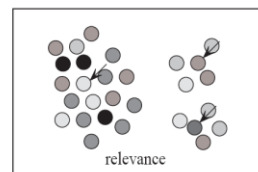
(a) a set of patterns



(b) redundancy-aware top-k



(c) traditional top-k



(d) summarization

69

69

## Redundancy-Award Top-k Patterns

- A significance measure  $S$  is a function mapping a pattern  $p \in P$  to a real value such that  $S(p)$  is the degree of interestingness (or usefulness) of the pattern  $p$ .
  - Objective measures** depend only on the structure of the given pattern and the underlying data used in the discovery process.
  - Subjective measures** are based on user beliefs in the data. They therefore depend on the users who examine the patterns.
- Redundancy  $R$  between two patterns  $p$  and  $q$  is defined as
  - $R(p,q) = S(p) + S(q) - S(p,q)$

70

70

## Redundancy-Award Top-k Patterns

---

- The ideal redundancy measure  $R(p,q)$  is usually hard to obtain. However, we can approximate redundancy using distance between patterns
- The problem of finding redundancy-aware top-k patterns can thus be transformed into finding a k-pattern set that maximizes the marginal significance, which is a well studied problem in information retrieval.
  - A document has high marginal relevance if it is both relevant to the query and contains minimal marginal similarity to previously selected documents, where the marginal similarity is computed by choosing the most relevant selected document.

71

71

## Advanced Frequent Pattern Mining

---

- 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
- Summary 

72

72

## 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

73