Mining All Non-Derivable Frequent Itemsets

Toon Calders

Bart Goethals

Outline

- Frequent Itemset Mining in 2002
  - Pattern Explosion Problem
  - Condensed Representations
- Non-Derivable Itemsets PKDD 2002
  - Quick Inclusion-Exclusion KDID 2005
  - Depth-first NDI Mining SDM 2005
- Recent Approaches Towards Non-Redundant Pattern Mining

Association Rules

Database

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

L_1

Itemset	Support
{1}	2
{2}	3
{3}	3
{5}	3

L_2

Itemset	Support
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

L_3

Itemset	Support
{2 3 5}	2

{3,5} → {2}
{2,3} → {5}
{1} → {3}
{2} → {5}
{5} → {2}

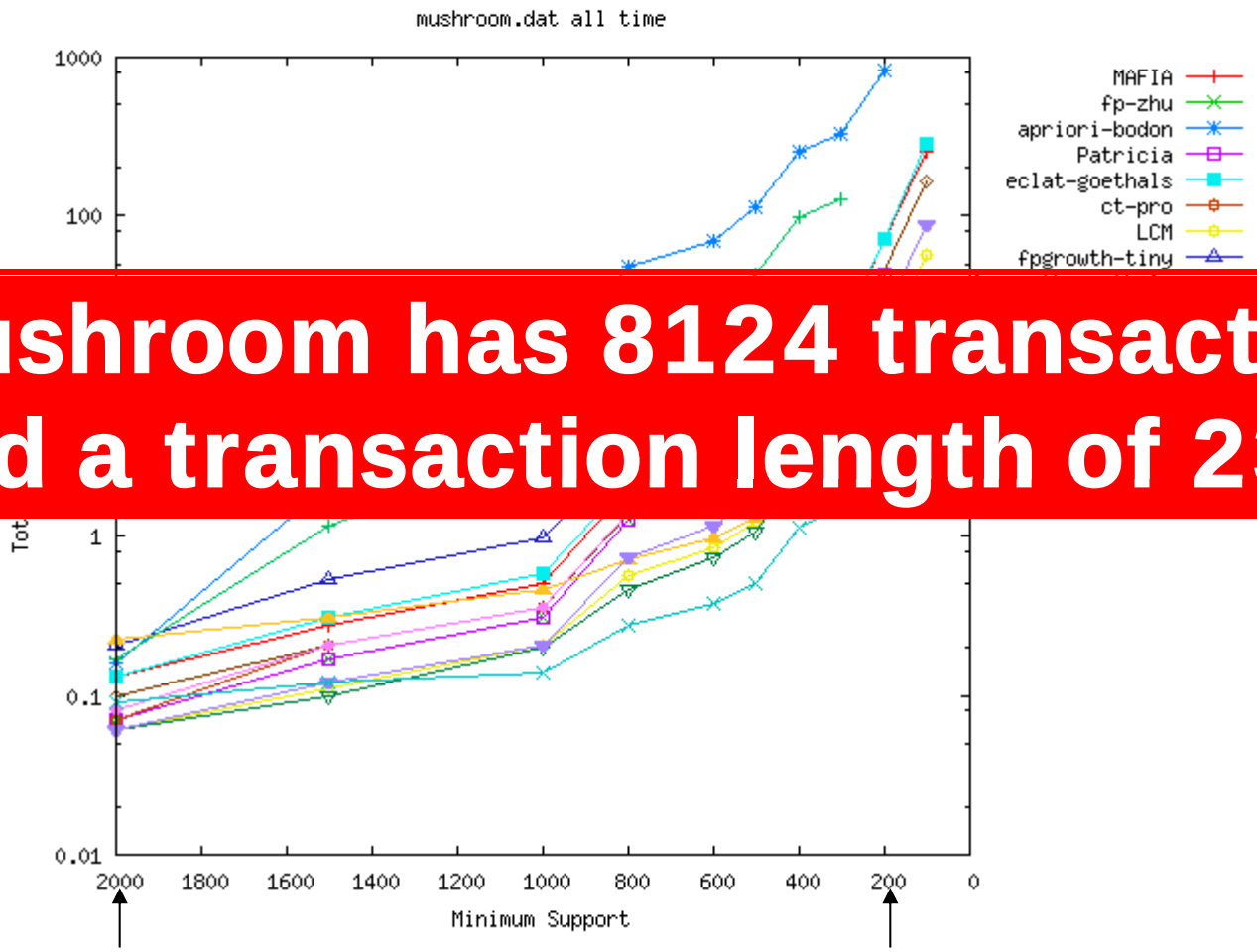
Mining **association rules** between sets of items in large databases
R **Agrawal**, T Imieliński, A Swami - ACM SIGMOD Record, 1993 - dl.acm.org
Cited by 11735

Situation Early 2000's

- Association rules gaining popularity

- Literally hundreds of algorithms:
  AIS, Apriori, AprioriTID, AprioriHybrid, FPGrowth, FPGrowth*, Eclat, dEclat, Pincer-search, ABS, DCI, kDCI, LCM, AIM, PIE, ARMOR, AFOPT, COFI, Patricia, MAXMINER, MAFIA, …

Situation Early 2000's



Mushroom has 8124 transactions, and a transaction length of 23

Over 50 000 patterns

Over 10 000 000 patterns

Situation Early 2000's

- Frequent itemset / Association rule mining
  = find all itemsets / ARs satisfying thresholds

- Many are redundant

  smoker → lung cancer

  smoker, bald → lung cancer

  pregnant → woman

  pregnant, smoker → woman, lung cancer

Condensed Representations

A1	A2	A3	B1	B2	B3	C1	C2	C3
1	1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0
0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	1	1	1

- Number of frequent itemsets = 21
- Need a compact representation

Discovering frequent **closed itemsets** for association rules

N **Pasquier**, Y Bastide, R Taouil, L Lakhal - Database Theory—ICDT'99, 1999

Cited by 1089

Condensed Representations

- Condensed Representation:
  "Compressed" version of the collection of all frequent itemsets (usually a subset) that allows for lossless regeneration of the complete collection.
  - Closed Itemsets (Pasquier et al, ICDT 1999)
  - Free Itemsets (Boulicaut et al, PKDD 2000)
  - Disjunction-Free itemsets (Bykowski and Rigotti, PODS 2001)

Outline

- Frequent Itemset Mining in 2002
  - Pattern Explosion Problem
  - Condensed Representations
- Non-Derivable Itemsets PKDD 2002
  - Quick Inclusion-Exclusion KDID 2005
  - Depth-first NDI Mining SDM 2005
- Recent Approaches Towards Non-Redundant Pattern Mining

2002

Mining All Non-Derivable Frequent Itemsets

Toon Calders
University of Antwerp
Belgium

Bart Goethals
University of Limburg
Belgium

Redundancies

- How do supports interact?
- What information about unknown supports can we derive from known supports?
  - Concise representation: only store relevant part of the supports

Redundancies

- Agrawal et al. (Monotonicity)
  - $Supp(AX) \leq Supp(A)$
- Lakhal et al. (Closed sets)
  Boulicaut et al. (Free sets)
  - **If** $Supp(A) = Supp(AB)$
    Then $Supp(AX) = Supp(AXB)$

Redundancies

- Bayardo (MAXMINER)
  - $\text{Supp}(ABX) \geq \text{Supp}(AX) - \underline{(\text{Supp}(X) - \text{Supp}(BX))}$

drop (X, B)

- Bykowski, Rigotti (Disjunction-free sets)

if $\text{Supp}(ABC) = \text{Supp}(AB) + \text{Supp}(AC) - \text{Supp}(A)$

then

$\text{Supp}(ABCX) = \text{Supp}(ABX) + \text{Supp}(ACX) - \text{Supp}(AX)$

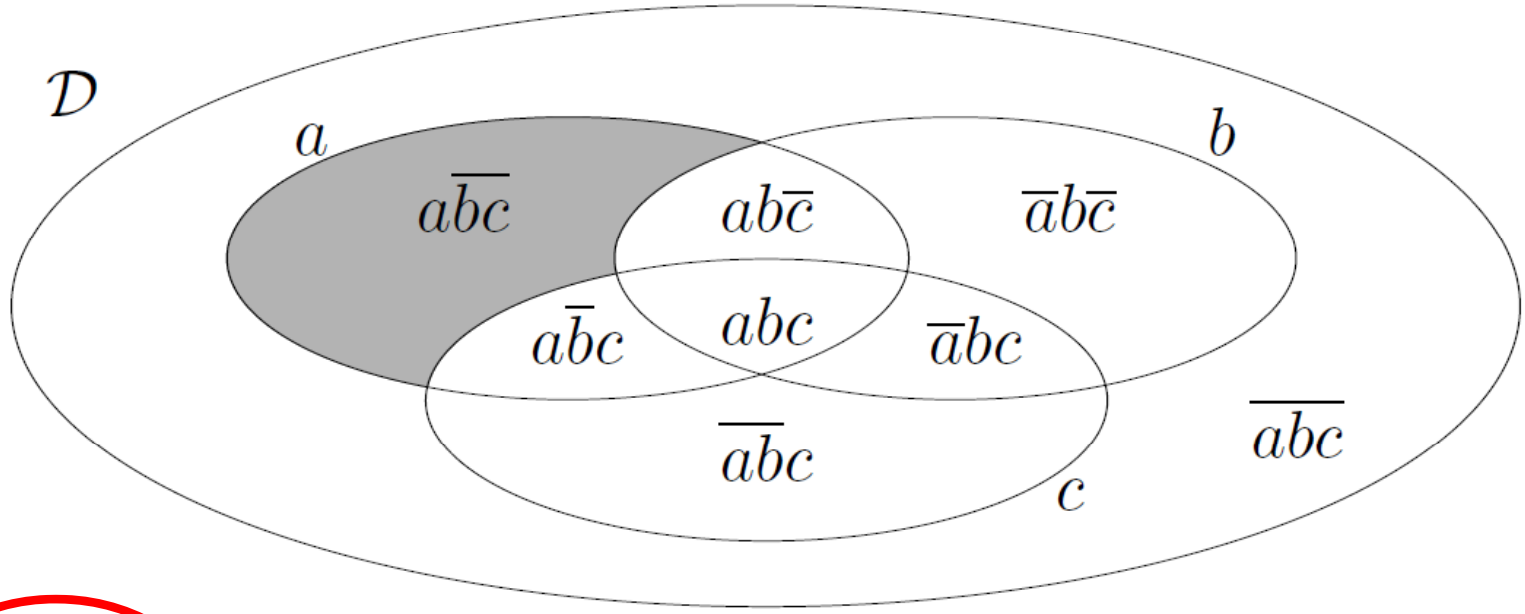
Goal

- Theoretical study – general framework
  - Deduction Rules
- Applicability in
  - Frequent Set Mining
  - Concise Representations

Outline

- Deduction Rules via Inclusion-Exclusion
- Derivable Itemsets
- NDI Algorithm
- Evaluation
- Conclusion & Further Work

I. The Inclusion – Exclusion Principle



$$\underbrace{\text{supp}(a\overline{bc})}_{\geq 0} = \text{supp}(a) - \text{supp}(ab) - \text{supp}(ac) + \text{supp}(abc)$$

$$\text{supp}(abc) \geq \text{supp}(ab) + \text{supp}(ac) - \text{supp}(a)$$

Deduction Rules via Inclusion-Exclusion

- Inclusion-exclusion principle:

$$\text{supp}(\overline{abc}) \geq 0$$

$$= \text{supp}(\{\}) - \text{supp}(a) - \text{supp}(b) - \text{supp}(c) + \text{supp}(ab) + \text{supp}(ac) + \text{supp}(bc) - \text{supp}(abc)$$

$$\text{supp}(ABC) \leq \text{supp}(AB) + \text{supp}(AC) + \text{supp}(BC) - \text{supp}(A) - \text{supp}(B) - \text{supp}(C) + \text{supp}(\{\})$$

Deduction Rules via Inclusion-Exclusion

- One more:

$$\text{supp}(ab\bar{c}) = \text{supp}(ab) - \text{supp}(abc)$$
$$\geq 0$$

supp(ABC) ≥ supp(ab)

Complete Set for supp(abc)

0 $\quad supp(abc) \geq o$

$supp(abc) \leq supp(ab)$ — monotonicity

1 $\quad supp(abc) \leq supp(ac)$

$supp(abc) \leq supp(bc)$

---

free/closed

$supp(abc) \geq supp(ab) + supp(ac) - supp(a)$

2 $\quad supp(abc) \geq supp(ab) + supp(bc) - supp(b)$ — disj-free

$supp(abc) \geq supp(ac) + supp(bc) - supp(c)$

---

3 $\quad supp(abc) \leq supp(ab) + supp(ac) + supp(bc) - supp(a) - supp(b) - supp(c) + supp(\{\})$

(PKDD03)

Example: Deduction Rules

supp({}) = 6

a	b	c
0	0	1
0	1	1
0	1	0
1	1	0
1	0	0
1	1	1

Example: Deduction Rules

supp({}) = 6
supp(a) = 3
supp(b) = 4
supp(c) = 3

a	b	c
0	0	1
0	1	1
0	1	0
1	1	0
1	0	0
1	1	1

Example: Deduction Rules

a	b	c
0	0	1
0	1	1
0	1	0
1	1	0
1	0	0
1	1	1

supp({}) = 6
supp(a) = 3
supp(b) = 4
supp(c) = 3

supp(ab)=2
supp(ac)=1
supp(bc)=2

Example: Deduction Rules

supp({}) = 6	supp(ab)=2
supp(a) = 3	supp(ac)=1
supp(b) = 4	supp(bc)=2
supp(c) = 3	

$$supp(abc) \geq o$$
$$supp(abc) \leq supp(ab)$$
$$supp(abc) \leq supp(ac)$$
$$supp(abc) \leq supp(bc)$$
$$supp(abc) \geq supp(ab) + supp(ac) - supp(a)$$
$$supp(abc) \geq supp(ab) + supp(bc) - supp(b)$$
$$supp(abc) \geq supp(ac) + supp(bc) - supp(c)$$
$$supp(abc) \leq supp(ab) + supp(ac) + supp(bc) - supp(a) - supp(b) - supp(c) + supp(\{\})$$

Example: Deduction Rules

supp({}) = 6	supp(ab)=2
supp(a) = 3	supp(ac)=1
supp(b) = 4	supp(bc)=2
supp(c) = 3	

$$\text{supp}(abc) \geq 0$$
$$\text{supp}(abc) \leq 2$$
$$\text{supp}(abc) \leq 1$$
$$\text{supp}(abc) \leq 2$$
$$\text{supp}(abc) \geq 2 + 1 - 3 = 0$$
$$\text{supp}(abc) \geq 2 + 2 - 4 = 0$$
$$\text{supp}(abc) \geq 1 + 2 - 3 = 0$$
$$\text{supp}(abc) \leq 2 + 1 + 2 - 3 - 4 - 3 + 6 = 1$$

Hence, $\text{supp}(abc) \in [0,1]$

Main Theorem

Theorem

Given: Supp(I) for all $I \subset J$

The deduction rules are **sound, complete, and non-redundant** for deducing upper and lower bounds on Supp(J).

- Complete: if bounds are [L,U], then
  - Exists consistent D_L s.t. supp(J,D) = L
  - Exists consistent D_U s.t. supp(J,D) = U

Example: Completeness

a	b	c
0	0	1
0	1	1
0	1	0
1	1	0
1	0	0
1	1	1

supp({}) = 6
supp(a) = 3
supp(b) = 4
supp(c) = 3
supp(ab)=2
supp(ac)=1
supp(bc)=2

- Lower bound: 0
- Upper bound: 1

Example: Completeness

D_U =

a	b	c
0	0	1
0	1	1
0	1	0
1	1	0
1	0	0
1	**1**	**1**

supp({}) = 6
supp(a) = 3
supp(b) = 4
supp(c) = 3
supp(ab)=2
supp(ac)=1
supp(bc)=2

- Lower bound: 0
- Upper bound: 1

Example: Completeness

D_L =

a	b	c
0	0	**0**
0	1	1
0	1	**1**
1	1	0
1	0	**1**
1	1	**0**

supp({}) = 6
supp(a) = 3
supp(b) = 4
supp(c) = 3
supp(ab)=2
supp(ac)=1
supp(bc)=2

- Lower bound: 0
- Upper bound: 1

II. Derivable Itemsets

Given: Supp(I) for all $I \subset J$
Lower bound on Supp(J) = l
Upper bound on Supp(J) = u

- Without counting : $Supp(J) \in [l,u]$
- J is a ***derivable itemset*** (DI) iff $l = u$
  - We **know** Supp(J) **exactly** without counting!

Derivable Itemsets

J derivable itemset:

- No need to ***count*** supp(J)
- No need to ***store*** supp(J)
  - We can use the deduction rules

- Concise representation:

$$C = \{ (J, supp(J)) \mid J \text{ not derivable from } supp(I), I \subset J \}$$

Example: Derivable Itemset

a	b	c
0	0	0
0	1	1
0	1	1
1	1	0
1	0	0
1	1	1

supp({}) = 6
supp(a) = 3
supp(b) = 4
supp(c) = 3

supp(ab)=2
supp(ac)=1
supp(bc)=3

Example: Derivable Itemset

supp({}) = 6	supp(ab)=2
supp(a) = 3	supp(ac)=1
supp(b) = 4	supp(bc)=3
supp(c) = 3	

$$supp(abc) \geq 0$$
$$supp(abc) \leq supp(ab)$$
$$supp(abc) \leq supp(ac)$$
$$supp(abc) \leq supp(bc)$$
$$supp(abc) \geq supp(ab) + supp(ac) - supp(a)$$
$$supp(abc) \geq supp(ab) + supp(bc) - supp(b)$$
$$supp(abc) \geq supp(ac) + supp(bc) - supp(c)$$
$$supp(abc) \leq supp(ab) + supp(ac) + supp(bc) - supp(a) - supp(b) - supp(c) + supp(\{\})$$

Example: Derivable Itemset

supp({}) = 6	supp(ab)=2
supp(a) = 3	supp(ac)=1
supp(b) = 4	supp(bc)=3
supp(c) = 3	

$supp(abc) \geq 0$
$supp(abc) \leq 2$
$supp(abc) \leq 1$
$supp(abc) \leq 2$
$supp(abc) \geq 2 + 1 - 3 = 0$
$supp(abc) \geq 2 + 3 - 4 = 1$
$supp(abc) \geq 1 + 3 - 3 = 1$
$supp(abc) \leq 2 + 1 + 3 - 3 - 4 - 3 + 6 = 2$

Hence, $supp(abc) \in [1,1]$

Derivable Itemsets

Theorem (Monotonicity)

If $J \subset K$, J derivable, then K derivable.

Hence, being derivable is anti-monotone !!

Theorem (Halving)

The width of the interval for $J \cup \{A\}$ is at most ***half*** the size of the interval for J

Hence, any NDI has length at most log(|D|) !!!

III. Algorithm

- Non-derivability is monotone
  - Levelwise search, apriori-like
  - Also depth-first version (SDM 2005)
- Only evaluate itemset if
  - All its subsets are frequent
  - Bounds cannot derive support exactly

Optimization

Lemma

If bound on I is [l,u] and Supp(I) equals either l or u, then all supersets of I are derivable

- Apriori-style algorithm
  - Itemset of length k is a candidate if: all subsets
    - are frequent
    - are non-derivable
    - lower nor upper bound equal actual support
  - bounds cannot derive support exactly

Quick Inclusion-Exclusion KDID 2005

- Computing the bounds is expensive if done brute-force

- Exploit that many of sums share terms:

$$ab\overline{cd} = ab - abc - abd + abcd$$

$$a\overline{bcd} = a - \mathbf{ab} - ac - ad + \mathbf{abc} + \mathbf{abd} + acd - \mathbf{abcd}$$

Quick Inclusion-Exclusion
KDID 2005

$$support(\overline{a}G) = support(G) - support(aG)$$

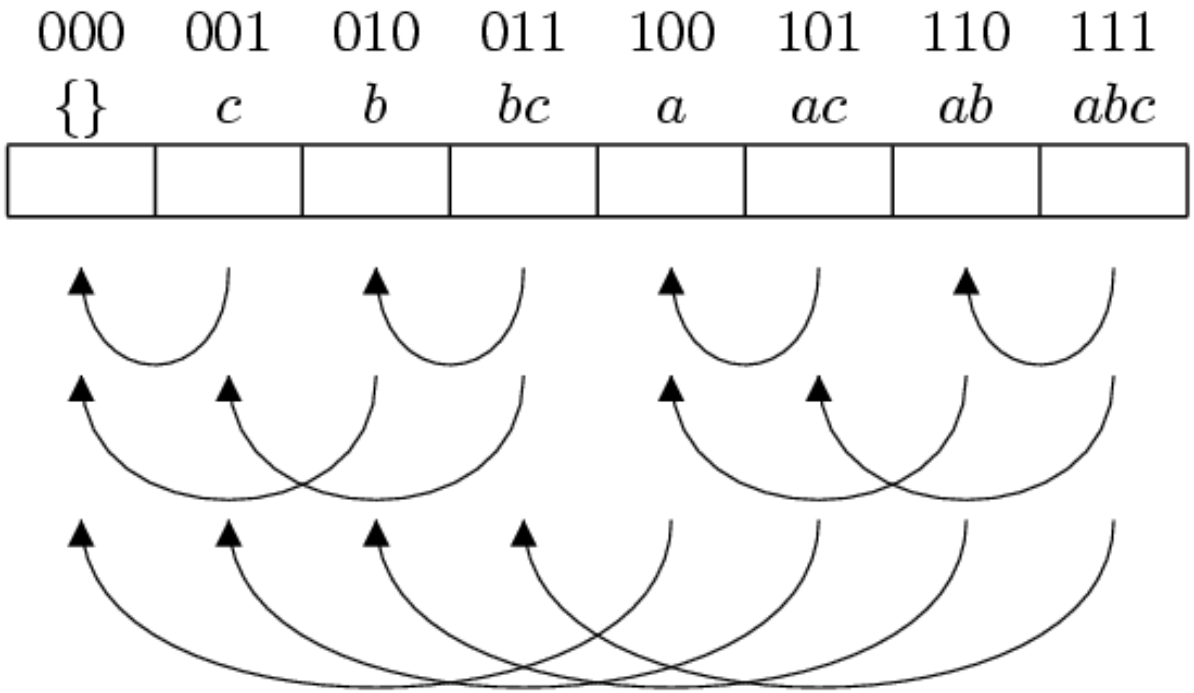
000	001	010	011	100	101	110	111
$\{\}$	c	b	bc	a	ac	ab	abc

000	001	010	011	100	101	110	111
$\overline{c}$	c	$b\overline{c}$	bc	$a\overline{c}$	ac	$ab\overline{c}$	abc

000	001	010	011	100	101	110	111
$\overline{bc}$	$\overline{b}c$	$b\overline{c}$	bc	$a\overline{bc}$	$a\overline{b}c$	$ab\overline{c}$	abc

000	001	010	011	100	101	110	111
$\overline{a}bc$	$\overline{abc}$	$\overline{a}b\overline{c}$	$\overline{a}bc$	$a\overline{bc}$	$a\overline{b}c$	$ab\overline{c}$	abc

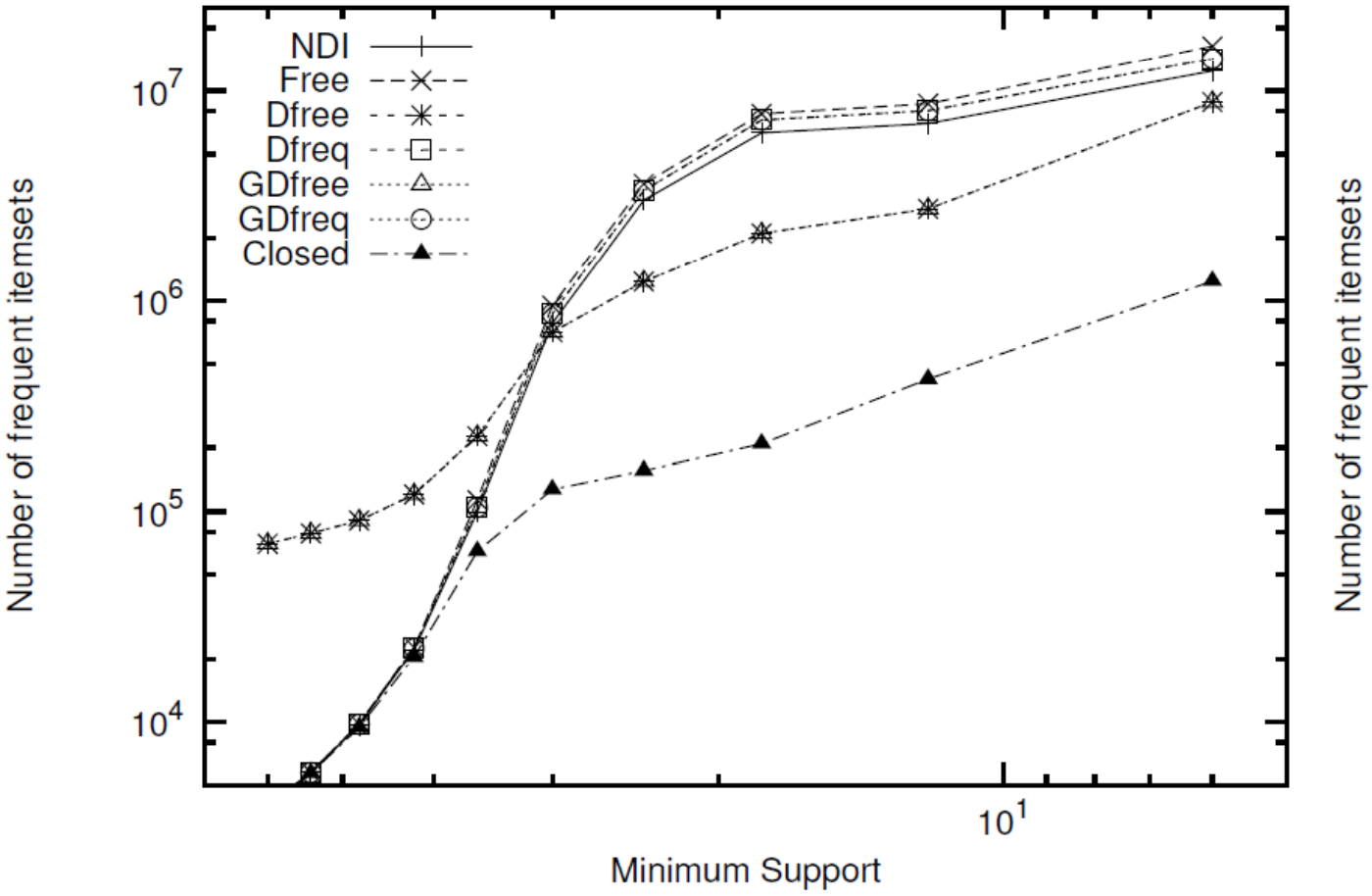
Quick Inclusion-Exclusion
KDID 2005



Evaluation - Empirically

- Comparison with other condensed representations
- Influence of Rule Depth

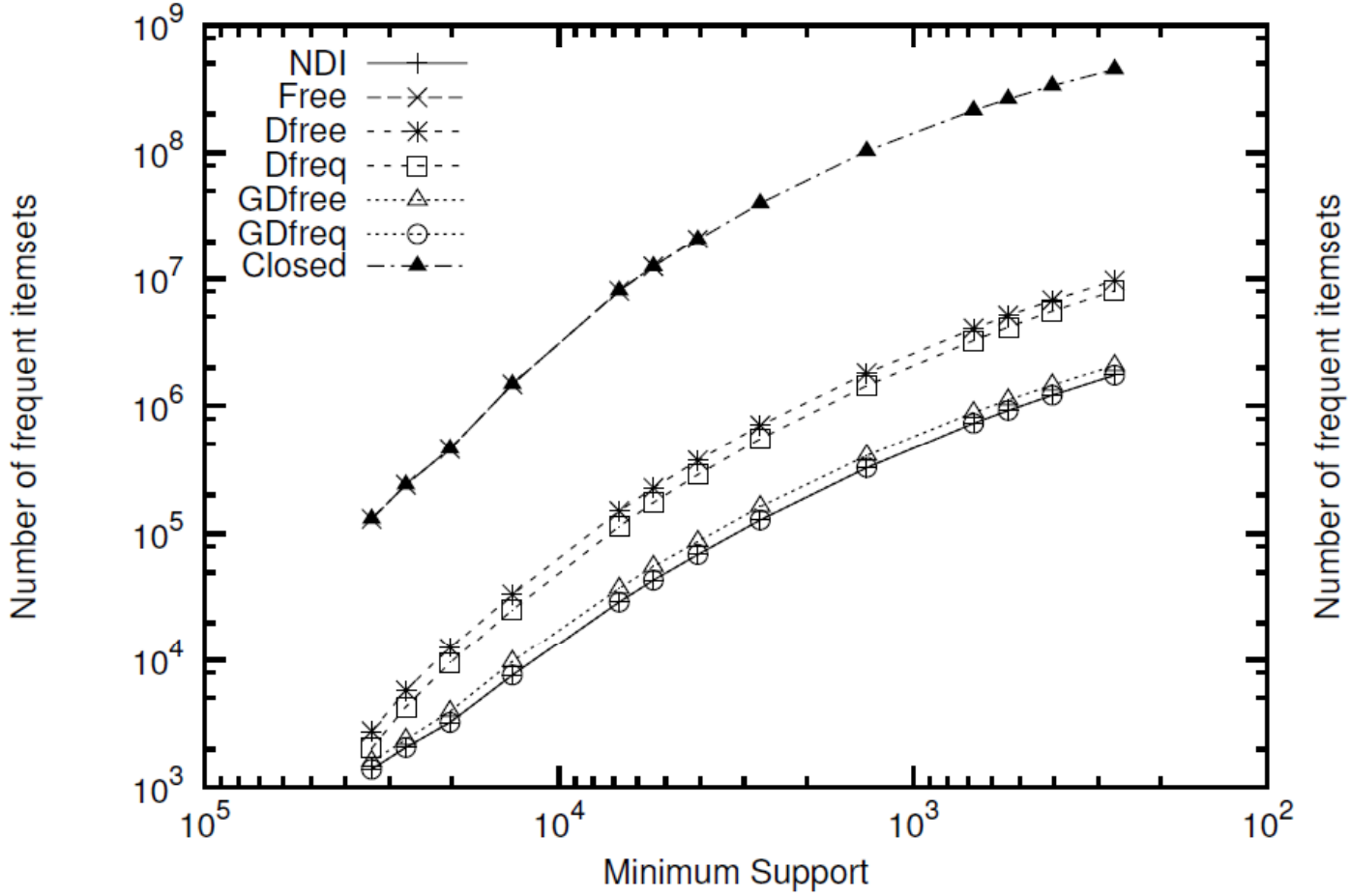
Comparison



(a) BMS-Webview-1

(DAMI 2007)

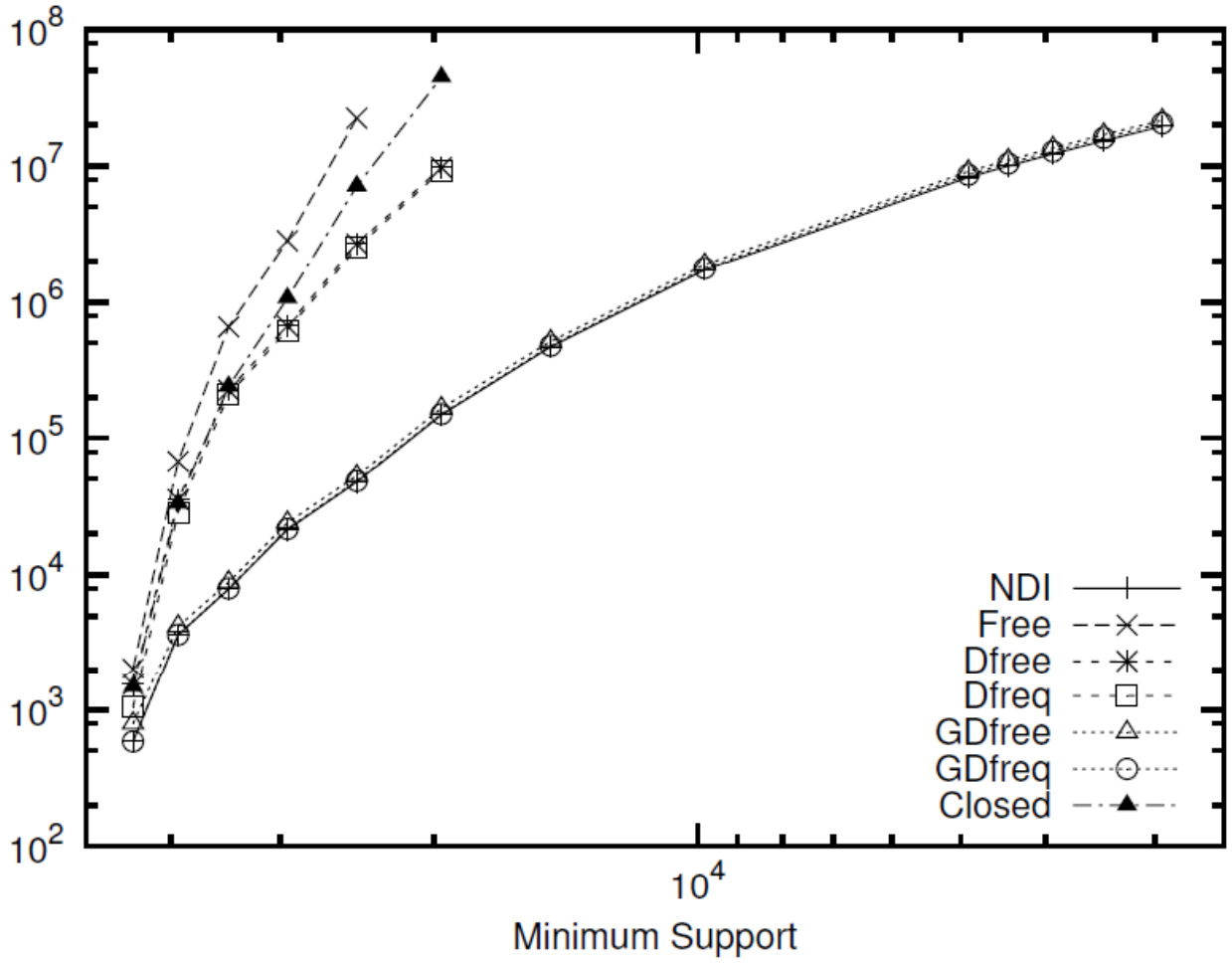
Comparison



(e) Connect-4

(DAMI 2007)

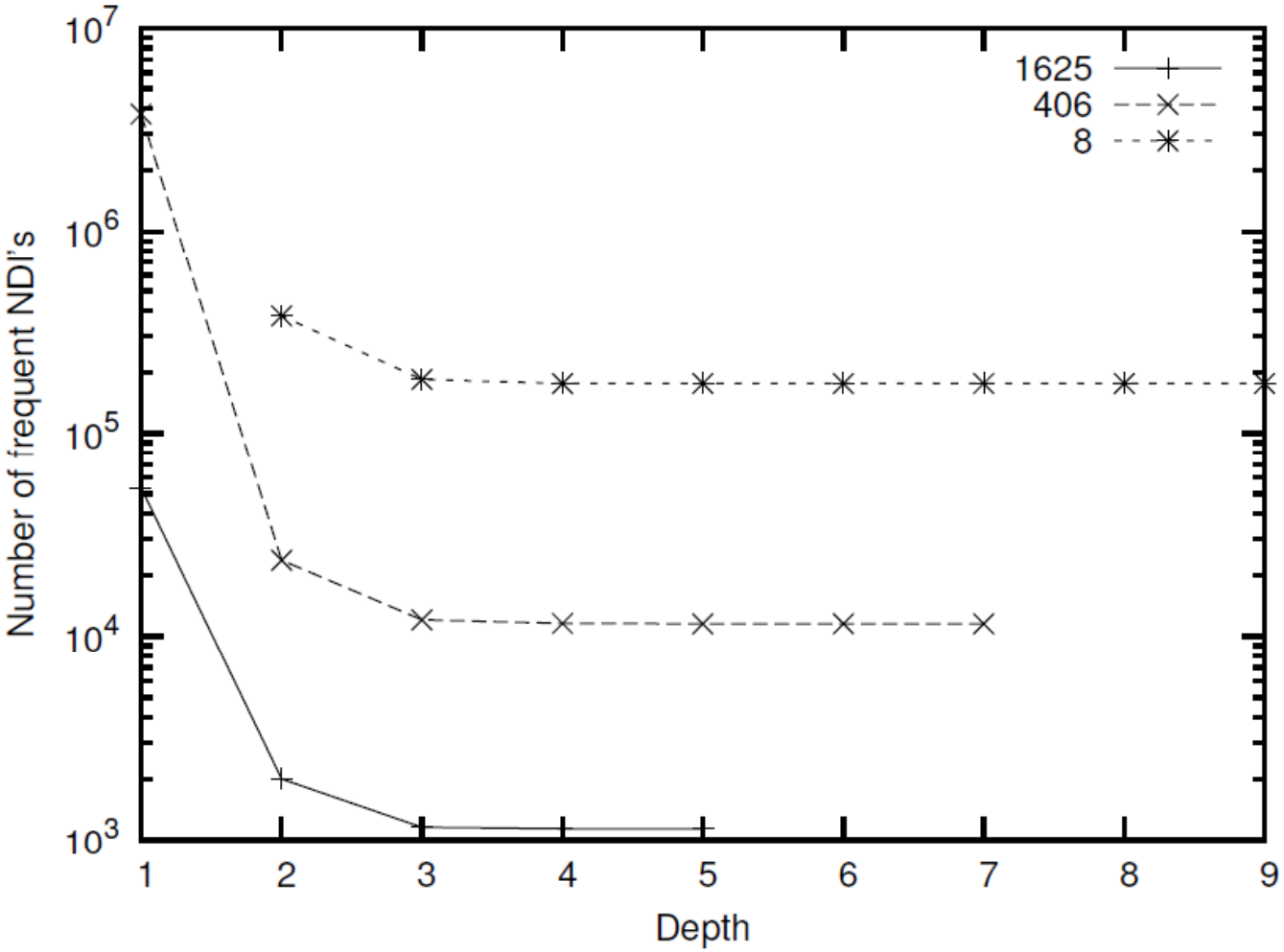
Comparison



(f) PUMSB

(DAMI 2007)

Influence of Rule Depth



Mushroom

(DAMI 2007)

Evaluation

- Number of frequent NDIs considerable smaller than number of frequent itemsets
- Most work is done by rules of limited depth
- Algorithm is efficient
  - Calculating NDI + deducing DIs often outperforms Apriori

V. Conclusion

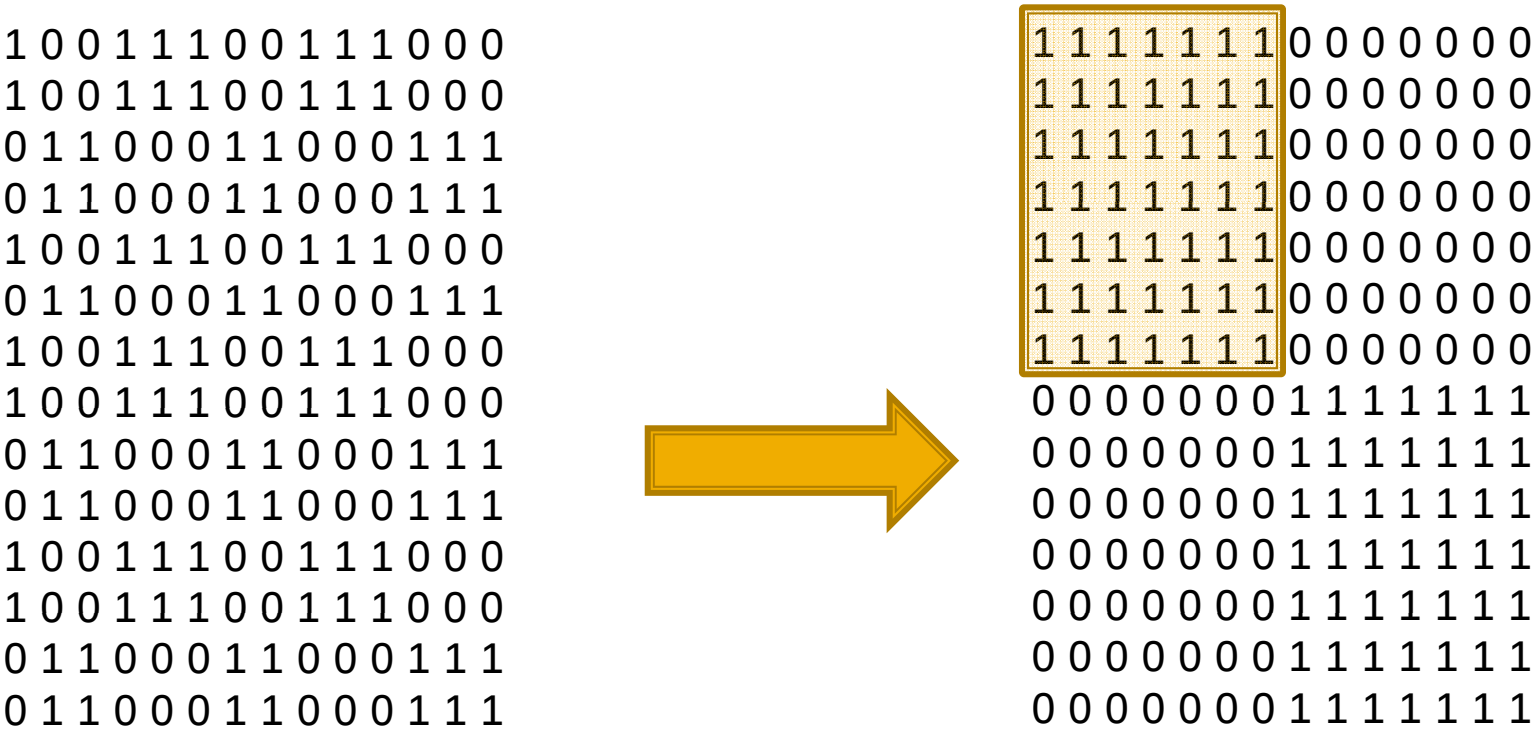
- Deduction rules for support
  - Sound, complete, non-redundant
- NDI as a concise representation
- Efficient algorithm for finding NDI
  - NDI is monotone
  - Intervals at least halve each iteration
- Theoretical and empirical evaluation
  - Sometimes better, sometimes worse than other condensed representations

Outline

- Frequent Itemset Mining in 2002
  - Pattern Explosion Problem
  - Condensed Representations
- Non-Derivable Itemsets PKDD 2002
  - Quick Inclusion-Exclusion KDID 2005
  - Depth-first NDI Mining SDM 2005
- Recent Approaches Towards Non-Redundant Pattern Mining

Illustrative Example: Tiles

- Tile = rectangle of 1's in a 0-1 matrix
  - We can reorder rows and columns



Illustrative Example: Tiles

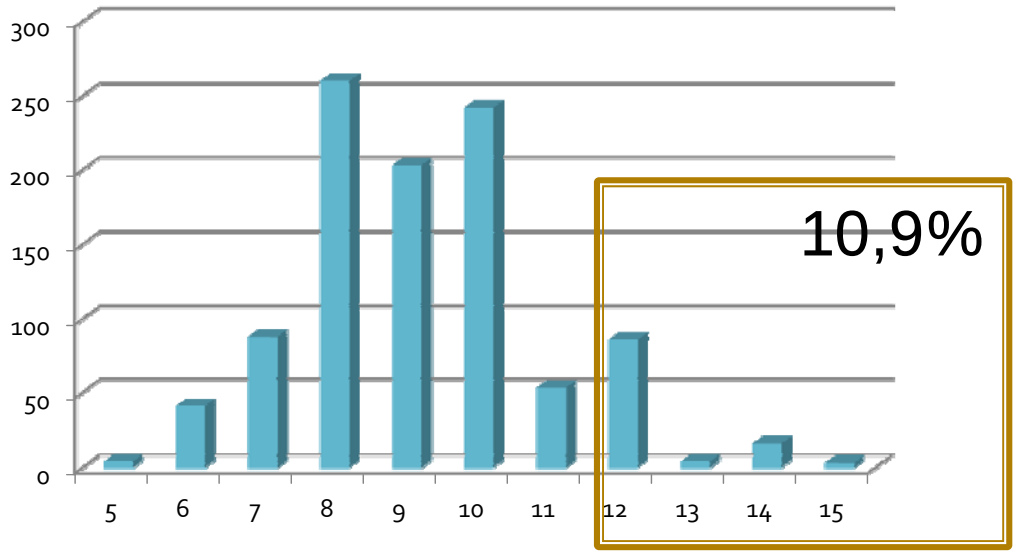
- Suppose we found a 4x3 tile; area = 12
  "Is this significant in a database with 20 tuples, 6 items and all items have a probability of 0.3?"

- Approach:
  - Characterize the distribution of maximal area of tiles in "random" data
  - Compute how likely it is to have a tile of size 16 or more = p-value

Illustrative Example: Tiles

- Characterizing the distribution is not easy
- Therefore: Simulation
  - Sample over all databases
  - Compute empirical p-value

20 tuples, 6 items; all items are independent and have probability 0.3



Statistical Approach

- Null-model expresses the prior believe of the user
- p-values can be computed for every itemset
  - How *extreme* is the observed support given our prior believe?

- Select null-model
- Compute p-values for the itemsets
- Rank itemsets according to p-value

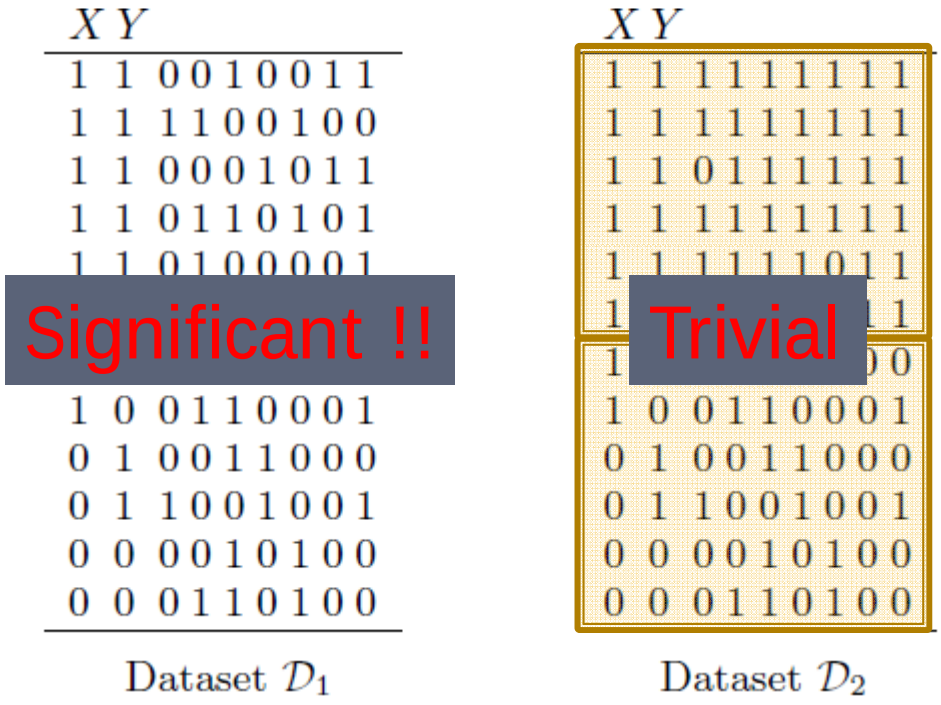
Model I: Swap Randomization

1 1 0 0 ← 2
0 1 1 0 ← 2
1 1 1 0 ← 3
0 0 0 1 ← 1
↑ ↑ ↑ ↑
2 3 2 1

- Uniform over all databases with:
  - the number of rows and columns
  - same row and column marginals

Assessing data mining results via **swap randomization**
A **Gionis**, H Mannila, T Mielikäinen… - ACM Transactions on …, 2007

Swap Randomization



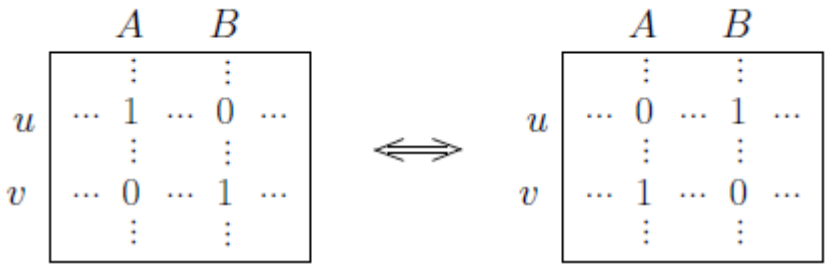
Buying a lot

Buying nothing

Figure 2: Examples of two 0–1 datasets, $\mathcal{D}_1$ and $\mathcal{D}_2$. In both cases we are interested in the correlation between columns (attributes) X and Y. The significance of the correlation result might depend on the overall context of the dataset

Swap Randomization

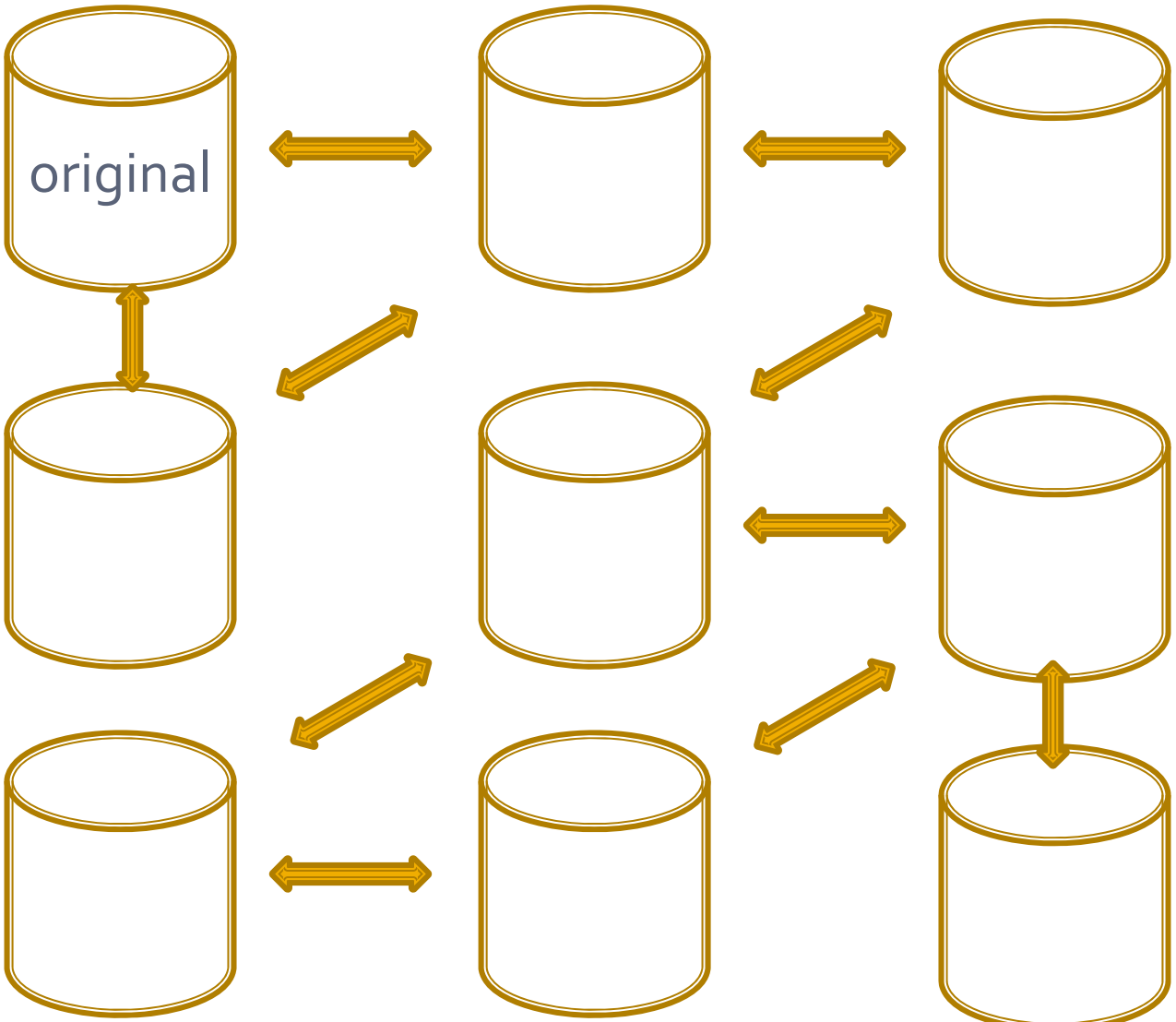
- Observation:
  - When we do have a solution, it is easy to create a new solution



  - By swapping we can reach every other database that satisfies the row and column totals

- We can use this observation to generate other databases!

"Swap graph" = Markov Chain



Markov Chain Monte Carlo

- After *a while*, the distribution will converge to a unique stationary distribution (under conditions)
  - Independent of starting point

- Under some conditions the stationary distribution of a Markov Chain is uniform
  - All nodes are connected
  - The chain is reversible
  - All nodes have the same degree (in general *not true*)

Swap Randomization: Experiments

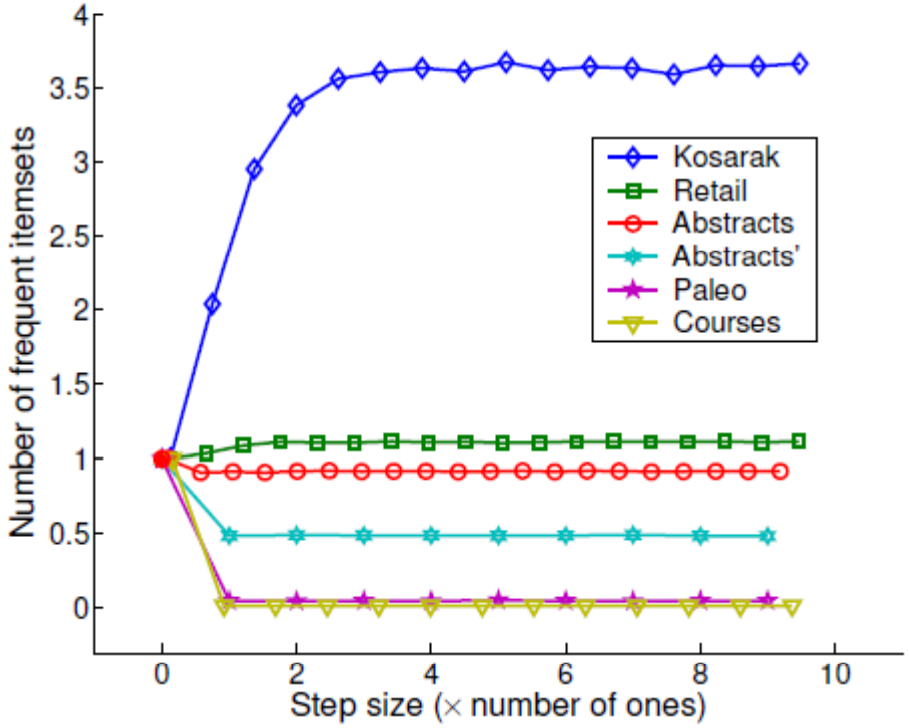
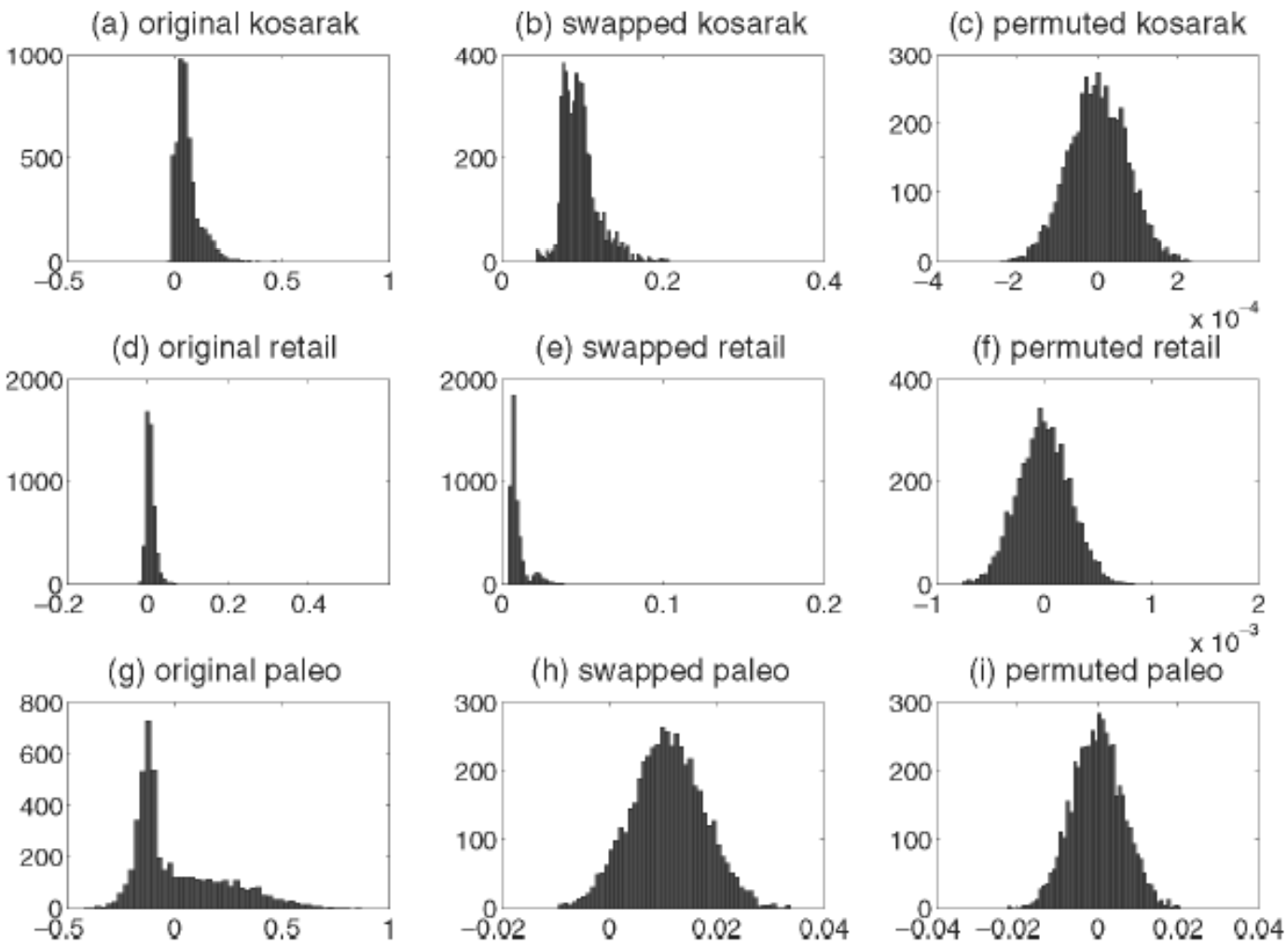


Figure 4: Convergence: x-axis: the number of steps ($\times$ the number of 1's in the data); y-axis: the number of frequent itemsets in the sampled datasets, divided by the number of frequent itemsets in the original dataset.

Swap Randomization: Experiments



II. MaxEntropy Based Models

- Given a set of frequency-constraints
- Null-model: usually based upon maximal entropy
  - Database = distribution (Nikolaj Tatti)

    Constraints must be satisfied

    Of these, select the (unique) MaxEnt distribution

    → one database

  - Distribution over databases (Tijl De Bie)

    Constraints must be satisfied in expectation

    Select the MaxEnt distribution

    →Distribution over databases

    →Depending on the type of constraint: analytical solution

II. MaxEntropy Based Models

- Why MaxEntropy? principle of maximum entropy
  - if nothing known about a distribution except that it belongs to a certain class, pick distribution with the largest entropy. Maximizing entropy minimizes the amount of prior information built into the distribution.

- $H(X) = -\Sigma_a\, p(X=a) \log(p(X=a))$
  - -log(p(X=a)) denotes space required to encode a, given an optimal Shannon encoding for the distribution p; characterizes the *information content of a*
  - p(X=a) denotes the probability that event X=a occurs
  - H(X) = average information content

Database = Distribution

- P(transaction T) = $|\{ tid \mid (tid,T) \in D \}| / |D|$
- Entropy(D) = $- \sum_{T \in D} P(T) \log(P(T))$
- Given a set of constraints

  $C = \{ supp(J_i) = s_i \}_i$

  the null-model is the distribution D that satisfies

  $supp(J_i,D) = s_i$

  and maximizes the entropy

- Comes down to optimizing a convex function under linear constraints

Database = Distribution

- Scoring an itemset J:
  - Compare empirical distribution with the MaxEnt model
    - E.g., KL-divergence
  - Compare support under null-model with true support
  - Compute p-value of the support of J under the null-model
- Itemsets to summarize the database
  - Select the itemset that minimizes KL-divergence between MaxEnt distribution and true dataset

Distribution over Databases

- Original database is n x m
- Consider all 0-1 databases of size n x m
- Every database has a probability
  ➔ distribution over databases
- $E(supp(J)) = \sum_D P(D)\, supp(J,D)$
- Given a set of constraints
  $C = \{ supp(J_i) = s_i \}_i$
  the null-model is the distribution D that satisfies
  $E(supp(J_i)) = s_i$
  and maximizes the entropy

Distribution over Databases

- Depending on the type of constraints it can be solved; e.g.,
  - density of a given tile
  - row and column marginals
  - Anything expressible as a linear constraint in the variables D[i,j]
- Does not work for frequency constraints!
  - freq(ab) = 5 ➔ D[1,a]*D[1,b] + D[2,a]*D[2,b] + … = 5

III. Minimal Description Length

- A good model helps us to compress the data and is compact
  - Let L(M) be the description length of the model,
  - Let L(D|M) be the size of the data when compressed by the model
- Find set of patterns (model M) that minimizes:
  L(M) + L(D|M)
- Explicit trade-off; adding a pattern:
  - Increases L(M),
  - Decreases L(D|M)

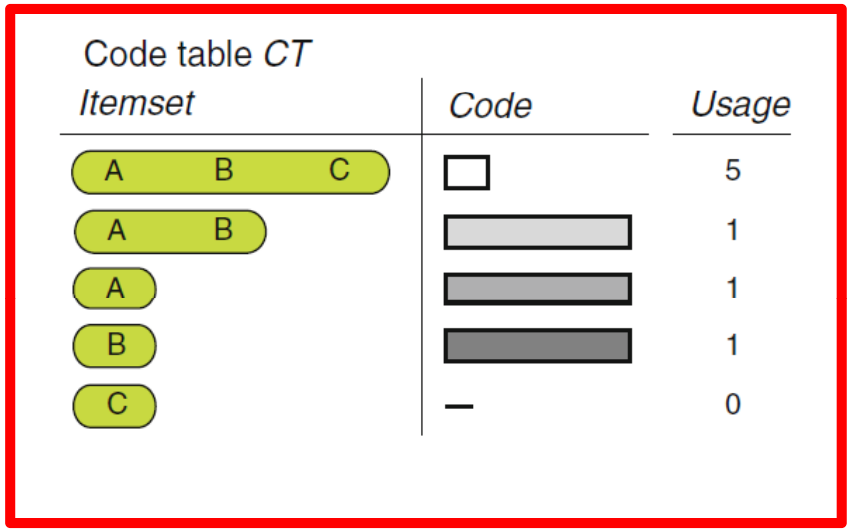
III. Minimal Description Length

- Rank itemsets according to how well they can be used to compress the dataset
  - Property of a set of patterns
- The "Krimp" algorithm was the first to use this paradigm in itemset mining
  - Assumes a seed set of patterns
  - A subset of these patterns is selected to form the "code book"
  - The best codebook is the one that gives the best compression

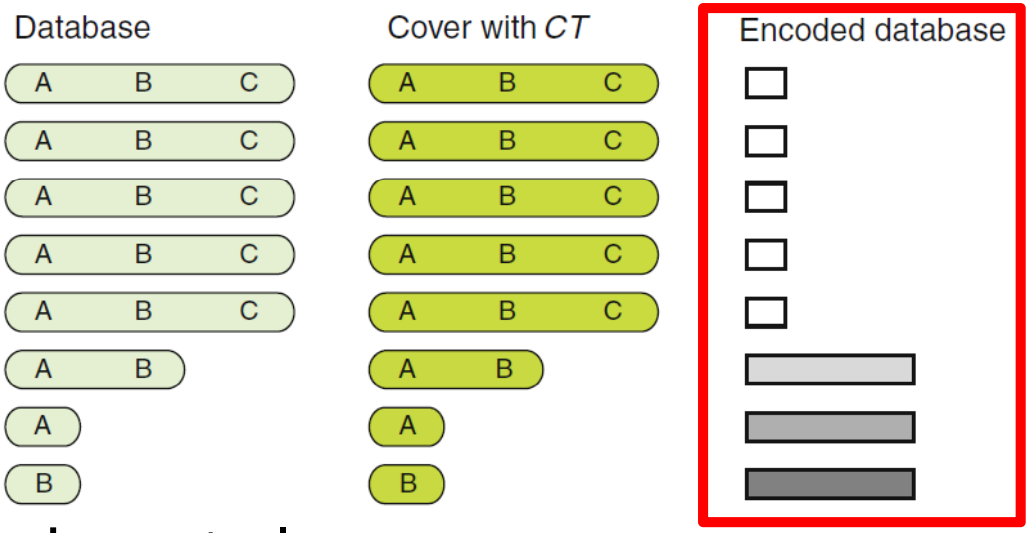
Krimp: mining itemsets that compress

J Vreeken, M van Leeuwen, A Siebes - Data Mining and Knowledge …, 2011

III. Minimal Description Length



L(M)



L(D|M)

Figures of Vreeken et al.

III. Minimal Description Length

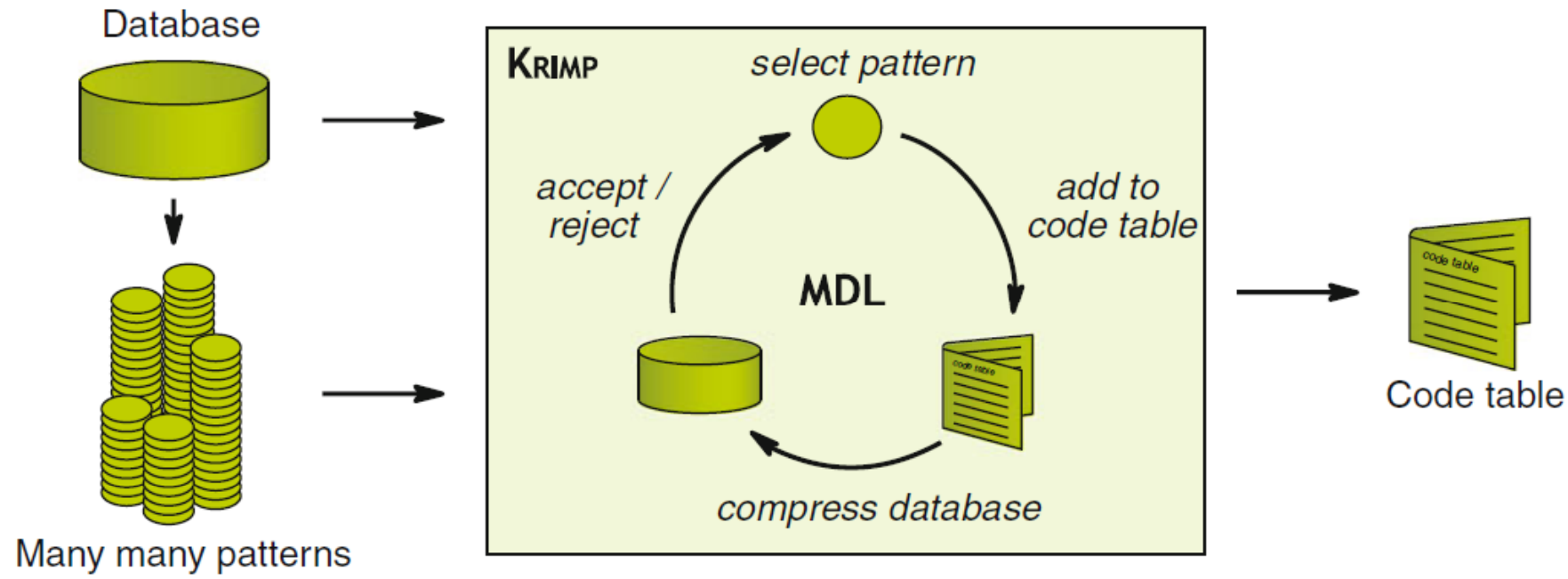


Fig. 4 KRIMP in action

Figure of Vreeken et al.

Conclusion

- Frequent pattern mining introduced in 1993
  - Hundreds of algorithms
  - Pattern explosion problem
- Early 2000: condensed representations
  - NDI was one of them
- Contributions of PKDD 2002 paper:
  - Sound, complete, non-redundant rules
  - Unifying framework (PKDD 2003)
  - Competitive condensed representatios

Conclusion

- More recent approaches:
  - Based on Minimal Description Length Principle:
    - Pattern collection → Model
    - Good models allow for compression
    - Heuristic method for selecting "best" model
  - Based on statistics
    - Null-model expresses "expectation"
    - Measure how "surprising" a pattern is w.r.t. the expectation
    - Update null model when accepting a pattern

Future?

- Make these approaches more practical
  - Work only on toy-examples
- Extend to other pattern domains
  - Sequences, graphs, dynamic graphs

Please, please, please, STOP making new algorithms for mining all frequent itemsets …