

Constraint-Based Pattern Mining

Marc Plantevit

Université Claude Bernard Lyon 1 – LIRIS CNRS UMR5205



* Slides from the research school lectures and different tutorials.

ENS Lyon, March 2018

About me.

marc.plantevit@univ-lyon1.fr or marc.plantevit@liris.cnrs.fr

Associate Professor

Computer Science Dept.

University Claude Bernard Lyon 1.

Lab: LIRIS UMR 5205

Team: Data Mining & Machine Learning

**Interest: Foundations of constraint-based
pattern mining, sequences,
augmented graphs.**



Outline

Introduction

Frequent Itemset Mining

- Frequent Itemset Mining
- Condensed Representations

Constraint-based Pattern Mining

- Constraint properties
- Algorithmic principles

Evolution of Sciences

Before 1600: Empirical Science

- ▶ Babylonian mathematics: 4 basis operations done with tablets and the resolution of practical problems based on words describing all the steps. \Rightarrow that worked and they manage to solve 3 degree equations.
- ▶ Ancient Egypt: No theorization of algorithms. We give only examples made empirically, certainly repeated by students and scribes. Empirical knowledge, transmitted as such, and not a rational mathematical science.
- ▶ Aristotle also produced many biological writings that were empirical in nature, focusing on biological causation and the diversity of life. He made countless observations of nature, especially the habits and attributes of plants and animals in the world around him, classified more than 540 animal species, and dissected at least 50.
- ▶ ...

1600-1950s: Theoretical Science

Each discipline has grown a theoretical component. Theoretical models often motivate experiments and generalize our understanding.

- ▶ Physics: Newton, Max Planck, Albert Einstein, Niels Bohr, Schrödinger
- ▶ Mathematics: Blaise Pascal, Newton, Leibniz, Laplace, Cauchy, Galois, Gauss, Riemann
- ▶ Chemistry: R. Boyle, Lavoisier, Dalton, Mendeleev,
- ▶ Biology, Medicine, Genetics: Darwin, Mendel, Pasteur



1950s–1990s, Computational Science

- ▶ Over the last 50 years, most disciplines have grown a third, computational branch (e.g. empirical, theoretical, and computational ecology, or physics, or linguistics.)
- ▶ Computational Science traditionally meant simulation. It grew out of our inability to find closed form solutions for complex mathematical models.



The Data Science Era

1990's-now, Data Science

- ▶ The flood of data from new scientific instruments and simulations
- ▶ The ability to economically store and manage petabytes of data online
- ▶ The Internet and computing Grid that makes all these archives universally accessible
- ▶ Scientific info. management, acquisition, organization, query, and visualization tasks scale almost linearly with data volumes.

The Fourth Paradigm: Data-Intensive Scientific Discovery

Data mining is a major new challenge!



The Fourth Paradigm. Tony Hey, Stewart Tansley, and Kristin Tolle. Microsoft Research, 2009.

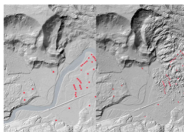
Evolution of Database Technology

- ▶ 1960s: Data collection, database creation, IMS and network DBMS
- ▶ 1970s : Relational data model, relational DBMS implementation
- ▶ 1980s: RDBMS, advanced data models (extended-relational, OO, deductive, etc.), application-oriented DBMS (spatial, scientific, engineering, etc.)
- ▶ 1990s: Data mining, data warehousing, multimedia databases, and Web databases
- ▶ 2000s: Stream data management and mining, Data mining and its applications, Web technology (XML, data integration) and global information systems, NoSQL, NewSQL.

Why Data Mining?

- ▶ The Explosive Growth of Data: from terabytes to petabytes
 - ▶ Data collection and data availability
 - ▶ Automated data collection tools, database systems, Web, computerized society
- ▶ Major sources of abundant data
 - ▶ Business: Web, e-commerce, transactions, stocks, ...
 - ▶ Science: Remote sensing, bioinformatics, scientific simulation, ...
 - ▶ Society and everyone: news, digital cameras, social network, ...
 - ▶ "We are drowning in data, but starving for knowledge!" – John Naisbitt, 1982 –

Applications

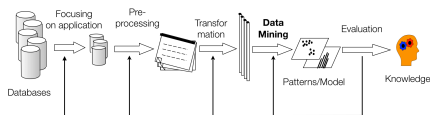


- ▶ Human mobility (ANR VEL'INNOV 2012–2016)
- ▶ Social media (GRAISearch - FP7-PEOPLE-2013-IAPP, Labex IMU project RESALI 2015–2018)
- ▶ Soil erosion (ANR Foster 2011–2015)
- ▶ Neuroscience (olfaction)
- ▶ Chemoinformatics
- ▶ Fact checking (ANR ContentCheck 2016 – 2019)
- ▶ Industry (new generation of product, failure detection)
- ▶ ...


What is Data Mining

- ▶ Data mining (knowledge discovery from data)
 - ▶ Extraction of interesting (non-trivial, implicit, previously unknown and potentially useful) patterns or knowledge from huge amount of data
- ▶ Alternative names:
 - ▶ KDD, knowledge extraction, data/pattern analysis, data archeology, data dredging, information harvesting, business intelligence, etc.
- ▶ Watch out: Is everything “data mining”?
 - ▶ simple search or query processing
 - ▶ (Deductive) expert systems

KDD Process



Iterative and Interactive Process

 Fayad et al., 1996

Data Mining

- ▶ Core of KDD
- ▶ Search for knowledge in data

Functionalities

- ▶ **Descriptive data mining** vs Predictive data mining
- ▶ **Pattern mining**, classification, clustering, regression
- ▶ Characterization, discrimination, association, classification, clustering, outlier and trend analysis, etc.







Major Issues In Data Mining

- ▶ Mining methodology
 - ▶ Mining different kinds of knowledge from diverse data types, e.g., bio, stream, Web.
 - ▶ Performance: efficiency, effectiveness, and scalability
 - ▶ Pattern evaluation: the interestingness problem
 - ▶ Incorporation of background knowledge.
 - ▶ Handling noise and incomplete data
 - ▶ Parallel, distributed and incremental mining methods.
 - ▶ Integration of the discovered knowledge with existing one: knowledge fusion.
 - ▶ Completeness or not.
- ▶ User interaction
 - ▶ Data mining query languages and ad-hoc mining.
 - ▶ Expression and visualization of data mining results.
 - ▶ Interactive mining of knowledge at multiple levels of abstraction
- ▶ Applications and social impacts
 - ▶ Domain-specific data mining & invisible data mining
 - ▶ Protection of data security, integrity, and privacy.

Where to Find References? DBLP, Google Scholar

- ▶ Data Mining and KDD
 - ▶ Conferences: ACM-SIGKDD, IEEE-ICDM, SIAM-DM, PKDD, PAKDD, etc.
 - ▶ Journals: Data Mining and Knowledge Discovery, ACM TKDD
- ▶ Database Systems
 - ▶ Conferences: : ACM-SIGMOD, ACM-PODS, (P)VLDB, IEEE-ICDE, EDBT, ICDT, DASFAA
 - ▶ Journals: IEEE-TKDE, ACM-TODS/TOIS, JIIS, J. ACM, VLDB J., Info. Sys., etc.
- ▶ AI & Machine Learning
 - ▶ Conferences: Int. Conf. on Machine learning (ICML), AAAI, IJCAI, COLT (Learning Theory), CVPR, NIPS, etc
 - ▶ Journals: Machine Learning, Artificial Intelligence, Knowledge and Information Systems, IEEE-PAMI, etc.
- ▶ Web and IR
 - ▶ Conferences: SIGIR, WWW, CIKM, etc
 - ▶ Journals: WWW: Internet and Web Information Systems,

Recommended Books

-  U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press, 1996
-  J. Han and M. Kamber. Data Mining: Concepts and Techniques. Morgan Kaufmann, 2nd ed., 2006
-  D. J. Hand, H. Mannila, and P. Smyth, Principles of Data Mining, MIT Press, 2001
-  P.-N. Tan, M. Steinbach and V. Kumar, Introduction to Data Mining, Wiley, 2005
-  Charu C. Aggarwal, Data Mining, Springer, 2015.
-  Mohammed J. Zaki, Wagner Meira, Jr. Data Mining and Analysis Fundamental Concepts and Algorithms. Cambridge University Press, 2014.

ML versus DM

Predictive (global) modeling

- ▶ Turn the data into an as accurate as possible prediction machine.
- ▶ Ultimate purpose is **automatization**.
- ▶ E.g., autonomously driving a car based on sensor inputs



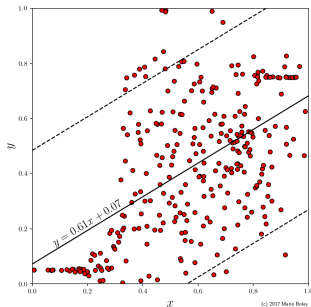
M. Boley www.realkd.org

Exploratory data analysis.

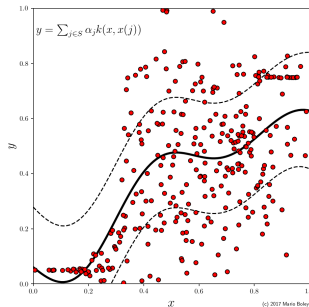
- ▶ Automatically discover novel insights about the domain in which the data was measured.
- ▶ Use machine discoveries to synergistically **boost** human expertise.
- ▶ E.g., understanding commonalities and differences among PET scans of Alzheimers patients.

ML versus DM

“A good prediction machine does not necessarily provide explicit insights into the data domains”



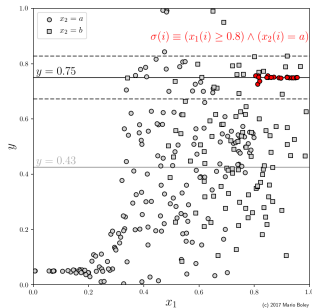
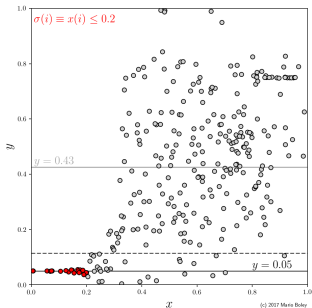
Global linear regression model



Gaussian process model.

ML versus DM

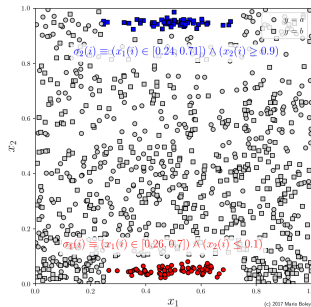
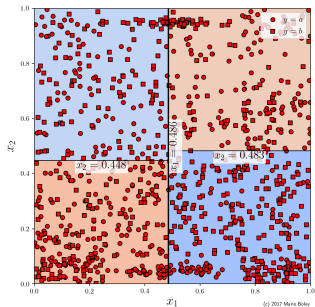
“A complex theory of everything might be of less value than a simple observation about a specific part of the data space”



Identifying interesting subspace and the power of saying “I don’t know for other points”

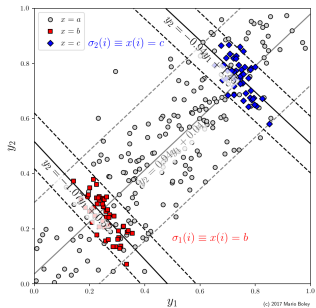
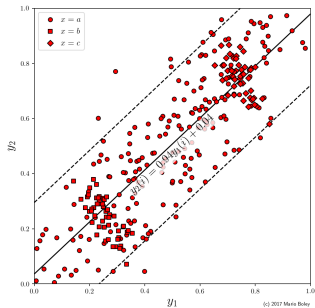
ML versus DM

“Subgroups look similar to decision trees but good tree learners are forced to brush over some local structure in favor of the global picture”



ML versus DM

“Going one step further, we can find local trends that are opposed to the global trend”




Roadmap

We will focus on **descriptive data mining** especially on Constraint-based Pattern Mining with an **inductive database vision**.

$$Th(\mathcal{L}, \mathcal{D}, \mathcal{C}) = \{\psi \in \mathcal{L} \mid \mathcal{C}(\psi, \mathcal{D}) \text{ is true}\}$$

- ▶ Pattern domain: (itemset, sequences, graphs, dynamic graphs, etc.)
- ▶ Constraints: How to efficiently push them?

 Imielinski and Mannila: Communications of the ACM (1996).

Outline

Introduction

Frequent Itemset Mining

Constraint-based Pattern Mining

- Constraint properties

- Algorithmic principles

Itemset: definition

Definition

Given a set of attributes \mathcal{A} , an *itemset* X is a subset of attributes, i. e., $X \subseteq \mathcal{A}$.

Input:

	a_1	a_2	\dots	a_n
o_1	$d_{1,1}$	$d_{1,2}$	\dots	$d_{1,n}$
o_2	$d_{2,1}$	$d_{2,2}$	\dots	$d_{2,n}$
\vdots	\vdots	\vdots	\ddots	\vdots
o_m	$d_{m,1}$	$d_{m,2}$	\dots	$d_{m,n}$

Question

How many itemsets are there?

where $d_{i,j} \in \{\text{true}, \text{false}\}$

Itemset: definition

Definition

Given a set of attributes \mathcal{A} , an *itemset* X is a subset of attributes, i. e., $X \subseteq \mathcal{A}$.

Input:

	a_1	a_2	\dots	a_n
o_1	$d_{1,1}$	$d_{1,2}$	\dots	$d_{1,n}$
o_2	$d_{2,1}$	$d_{2,2}$	\dots	$d_{2,n}$
\vdots	\vdots	\vdots	\ddots	\vdots
o_m	$d_{m,1}$	$d_{m,2}$	\dots	$d_{m,n}$

Question

How many itemsets are there? $2^{|\mathcal{A}|}$.

where $d_{i,j} \in \{\text{true}, \text{false}\}$

Transactional representation of the data

Relational representation:

$$\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$$

	a_1	a_2	\dots	a_n
o_1	$d_{1,1}$	$d_{1,2}$	\dots	$d_{1,n}$
o_2	$d_{2,1}$	$d_{2,2}$	\dots	$d_{2,n}$
\vdots	\vdots	\vdots	\ddots	\vdots
o_m	$d_{m,1}$	$d_{m,2}$	\dots	$d_{m,n}$

where $d_{i,j} \in \{\text{true}, \text{false}\}$

Transactional representation: \mathcal{D} is an array of subsets of \mathcal{A}

$$\begin{array}{c} t_1 \\ t_2 \\ \vdots \\ t_m \end{array}$$

where $t_i \subseteq \mathcal{A}$

Example

	a_1	a_2	a_3
o_1	\times	\times	\times
o_2	\times	\times	
o_3		\times	
o_4			\times

t_1	a_1, a_2, a_3
t_2	a_1, a_2
t_3	a_2
t_4	a_3

Frequency: definition

Definition (absolute frequency)

Given the objects in \mathcal{O} described with the Boolean attributes in \mathcal{A} , the absolute *frequency* of an itemset $X \subseteq \mathcal{A}$ in the dataset $\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$ is $|\{o \in \mathcal{O} \mid \{o\} \times X \subseteq \mathcal{D}\}|$.

Definition (relative frequency)

Given the objects in \mathcal{O} described with the Boolean attributes in \mathcal{A} , the relative *frequency* of an itemset $X \subseteq \mathcal{A}$ in the dataset $\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$ is $\frac{|\{o \in \mathcal{O} \mid \{o\} \times X \subseteq \mathcal{D}\}|}{|\mathcal{O}|}$.

The relative frequency is a joint probability.

Frequent itemset mining

Problem Definition

Given the objects in \mathcal{O} described with the Boolean attributes in \mathcal{A} , listing every itemset having a frequency above a given threshold $\mu \in \mathbb{N}$.

Input:

	a_1	a_2	\dots	a_n
o_1	$d_{1,1}$	$d_{1,2}$	\dots	$d_{1,n}$
o_2	$d_{2,1}$	$d_{2,2}$	\dots	$d_{2,n}$
\vdots	\vdots	\vdots	\ddots	\vdots
o_m	$d_{m,1}$	$d_{m,2}$	\dots	$d_{m,n}$

and a minimal frequency $\mu \in \mathbb{N}$.

where $d_{i,j} \in \{\text{true}, \text{false}\}$



R. Agrawal; T. Imielinski; A. Swami: Mining Association Rules Between Sets of Items in Large Databases, SIGMOD, 1993.

Frequent itemset mining

Problem Definition

Given the objects in \mathcal{O} described with the Boolean attributes in \mathcal{A} , listing every itemset having a frequency above a given threshold $\mu \in \mathbb{N}$.

Output: every $X \subseteq \mathcal{A}$ such that there are at least μ objects having all attributes in X .



R. Agrawal; T. Imielinski; A. Swami: Mining Association Rules Between Sets of Items in Large Databases, SIGMOD, 1993.

Frequent itemset mining: illustration

Specifying a minimal absolute frequency $\mu = 2$ objects (or, equivalently, a minimal relative frequency of 50%).

	a_1	a_2	a_3
o_1	×	×	×
o_2	×	×	
o_3		×	
o_4			×

Frequent itemset mining: illustration

Specifying a minimal absolute frequency $\mu = 2$ objects (or, equivalently, a minimal relative frequency of 50%).

	a_1	a_2	a_3
o_1	×	×	×
o_2	×	×	
o_3		×	
o_4			×

The frequent itemsets are: \emptyset (4), $\{a_1\}$ (2), $\{a_2\}$ (3), $\{a_3\}$ (2) and $\{a_1, a_2\}$ (2).

Completeness

Both the clustering and the classification schemes *globally* model the data: every object influences the output. That is the fundamental reason for these tasks to be solved in an *approximate* way.

In contrast, *local* patterns, such as itemsets, describe “anomalies” in the data and all such anomalies usually can be *completely* listed.

Inductive database vision

Querying data:

$$\{d \in \mathcal{D} \mid q(d, \mathcal{D})\}$$

where:

- ▶ \mathcal{D} is a dataset (tuples),
- ▶ q is a query.

Inductive database vision

Querying patterns:

$$\{X \in P \mid Q(X, \mathcal{D})\}$$

where:

- ▶ \mathcal{D} is the dataset,
- ▶ P is the pattern space,
- ▶ Q is an inductive query.

Inductive database vision

Querying **the frequent itemsets**:

$$\{X \in P \mid Q(X, \mathcal{D})\}$$

where:

- ▶ \mathcal{D} is the dataset,
- ▶ P is the pattern space,
- ▶ Q is an inductive query.

Inductive database vision

Querying **the frequent itemsets**:

$$\{X \in P \mid Q(X, \mathcal{D})\}$$

where:

- ▶ \mathcal{D} is a subset of $\mathcal{O} \times \mathcal{A}$, i. e., objects described with Boolean attributes,
- ▶ P is the pattern space,
- ▶ Q is an inductive query.

Inductive database vision

Querying **the frequent itemsets**:

$$\{X \in P \mid Q(X, \mathcal{D})\}$$

where:

- ▶ \mathcal{D} is a subset of $\mathcal{O} \times \mathcal{A}$, i. e., objects described with Boolean attributes,
- ▶ P is $2^{\mathcal{A}}$,
- ▶ Q is an inductive query.

Inductive database vision

Querying **the frequent itemsets**:

$$\{X \in P \mid Q(X, \mathcal{D})\}$$

where:

- ▶ \mathcal{D} is a subset of $\mathcal{O} \times \mathcal{A}$, i. e., objects described with Boolean attributes,
- ▶ P is $2^{\mathcal{A}}$,
- ▶ Q is $(X, \mathcal{D}) \mapsto |\{o \in \mathcal{O} \mid \{o\} \times X \subseteq \mathcal{D}\}| \geq \mu$.

Inductive database vision

Querying **the frequent itemsets**:

$$\{X \in P \mid Q(X, \mathcal{D})\}$$

where:

- ▶ \mathcal{D} is a subset of $\mathcal{O} \times \mathcal{A}$, i. e., objects described with Boolean attributes,
- ▶ P is $2^{\mathcal{A}}$,
- ▶ Q is $(X, \mathcal{D}) \mapsto f(X, \mathcal{D}) \geq \mu$.

Inductive database vision

Querying **the frequent itemsets**:

$$\{X \in P \mid Q(X, \mathcal{D})\}$$

where:

- ▶ \mathcal{D} is a subset of $\mathcal{O} \times \mathcal{A}$, i. e., objects described with Boolean attributes,
- ▶ P is $2^{\mathcal{A}}$,
- ▶ Q is $(X, \mathcal{D}) \mapsto f(X, \mathcal{D}) \geq \mu$.

Listing the frequent itemsets is NP-hard.

Naive algorithm

Input: $\mathcal{O}, \mathcal{A}, \mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}, \mu \in \mathbb{N}$

Output: $\{X \subseteq \mathcal{A} \mid f(X, \mathcal{D}) \geq \mu\}$

for all $X \subseteq \mathcal{A}$ **do**

if $f(X, \mathcal{D}) \geq \mu$ **then**

output(X)

end if

end for

Question

How many itemsets are enumerated? $2^{|\mathcal{A}|}$.

Transactional representation of the data

Relational representation:

$$\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$$

	a_1	a_2	\dots	a_n
o_1	$d_{1,1}$	$d_{1,2}$	\dots	$d_{1,n}$
o_2	$d_{2,1}$	$d_{2,2}$	\dots	$d_{2,n}$
\vdots	\vdots	\vdots	\ddots	\vdots
o_m	$d_{m,1}$	$d_{m,2}$	\dots	$d_{m,n}$

where $d_{i,j} \in \{\text{true}, \text{false}\}$

Transactional representation: \mathcal{D} is an array of subsets of \mathcal{A}

$$\begin{array}{c} t_1 \\ t_2 \\ \vdots \\ t_m \end{array}$$

where $t_i \subseteq \mathcal{A}$

Transactional representation of the data

Relational representation:

$$\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$$

	a_1	a_2	\dots	a_n
o_1	$d_{1,1}$	$d_{1,2}$	\dots	$d_{1,n}$
o_2	$d_{2,1}$	$d_{2,2}$	\dots	$d_{2,n}$
\vdots	\vdots	\vdots	\ddots	\vdots
o_m	$d_{m,1}$	$d_{m,2}$	\dots	$d_{m,n}$

where $d_{i,j} \in \{\text{true}, \text{false}\}$

Transactional representation: \mathcal{D} is an array of subsets of \mathcal{A}

t_1
 t_2
 \vdots
 t_m

where $t_i \subseteq \mathcal{A}$

For a linear time verification of “ X being a subset of t_i ”, the transactions are sorted (arbitrary order on \mathcal{A}) in a pre-processing step and any enumerated itemset X respects this order.

Transactional representation of the data

Relational representation:

$$\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$$

	a_1	a_2	\dots	a_n
o_1	$d_{1,1}$	$d_{1,2}$	\dots	$d_{1,n}$
o_2	$d_{2,1}$	$d_{2,2}$	\dots	$d_{2,n}$
\vdots	\vdots	\vdots	\ddots	\vdots
o_m	$d_{m,1}$	$d_{m,2}$	\dots	$d_{m,n}$

where $d_{i,j} \in \{\text{true}, \text{false}\}$

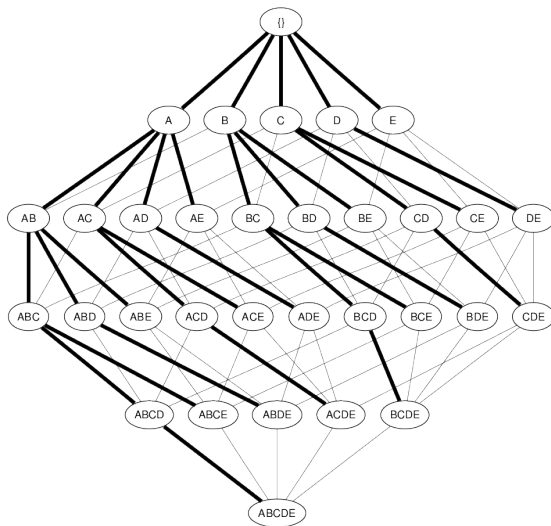
Transactional representation: \mathcal{D} is an array of subsets of \mathcal{A}

t_1
 t_2
 \vdots
 t_m

where $t_i \subseteq \mathcal{A}$

For a linear time verification of “ X being a subset of t_i ”, the transactions are sorted (arbitrary order on \mathcal{A}) in a pre-processing step and **any enumerated itemset X respects this order.**

Prefix-based enumeration



Complexity of the naive approach

Question

How many itemsets are enumerated? $2^{|\mathcal{A}|}$.

Question

What is the worst-case complexity of computing $f(X, \mathcal{D})$?

Question

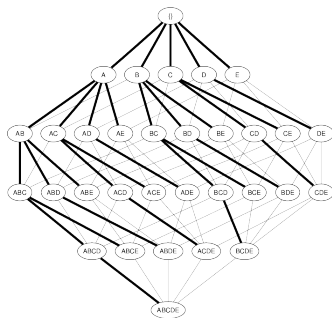
What is the worst-case complexity of computing $f(X, \mathcal{D})$? $O(|\mathcal{O} \times \mathcal{A}|)$.

Question

What is the worst-case complexity of the naive approach?
 $O(2^{|\mathcal{A}|} |\mathcal{O} \times \mathcal{A}|)$.

How to efficiently mine frequent itemsets?

Taking advantage of an important property



- ▶ *Anti-monotonicity of the frequency*
- ▶ in a levelwise enumeration (e.g. Apriori)



R. Agrawal; T. Imielinski; A. Swami: Mining Association Rules Between Sets of Items in Large Databases, SIGMOD, 1993.

- ▶ in a depthfirst enumeration (e.g. Eclat)



Mohammed J. Zaki, Scalable Algorithms for Association Mining. IEEE TKDE, 2000.

Anti-monotonicity of the frequency

Theorem

Given a dataset \mathcal{D} of objects described with Boolean attributes in \mathcal{A} :

$$\forall (X, Y) \in 2^{\mathcal{A}} \times 2^{\mathcal{A}}, X \subseteq Y \Rightarrow f(X, \mathcal{D}) \geq f(Y, \mathcal{D}) .$$

	a_1	a_2	a_3
o_1	×	×	×
o_2	×	×	
o_3		×	
o_4			×

$$f(\emptyset, \mathcal{D}) = 4$$

$$f(\{a_1\}, \mathcal{D}) = 2$$

$$f(\{a_1, a_2\}, \mathcal{D}) = 2$$

$$f(\{a_1, a_2, a_3\}, \mathcal{D}) = 1$$

Anti-monotonicity of the frequency

Theorem

Given a dataset \mathcal{D} of objects described with Boolean attributes in \mathcal{A} :

$$\forall (X, Y) \in 2^{\mathcal{A}} \times 2^{\mathcal{A}}, X \subseteq Y \Rightarrow f(X, \mathcal{D}) \geq f(Y, \mathcal{D}) .$$

	a_1	a_2	a_3
o_1	×	×	×
o_2	×	×	
o_3		×	
o_4			×

$$f(\emptyset, \mathcal{D}) = 4$$

$$f(\{a_3\}, \mathcal{D}) = 2$$

$$f(\{a_1, a_3\}, \mathcal{D}) = 1$$

$$f(\{a_1, a_2, a_3\}, \mathcal{D}) = 1$$

Anti-monotonicity of the frequency

Corollary

Given a dataset \mathcal{D} of objects described with Boolean attributes in \mathcal{A} and a minimal frequency $\mu \in \mathbb{N}$:

$$\forall (X, Y) \in 2^{\mathcal{A}} \times 2^{\mathcal{A}}, X \subseteq Y \Rightarrow \left(f(Y, \mathcal{D}) \geq \mu \Rightarrow f(X, \mathcal{D}) \geq \mu \right) .$$

	a_1	a_2	a_3
o_1	×	×	×
o_2	×	×	
o_3		×	
o_4			×

$$\begin{aligned} f(\emptyset, \mathcal{D}) &= 4 \\ f(\{a_3\}, \mathcal{D}) &= 2 \\ f(\{a_1, a_3\}, \mathcal{D}) &= 1 \\ f(\{a_1, a_2, a_3\}, \mathcal{D}) &= 1 \end{aligned}$$

Anti-monotonicity of the frequency

Corollary

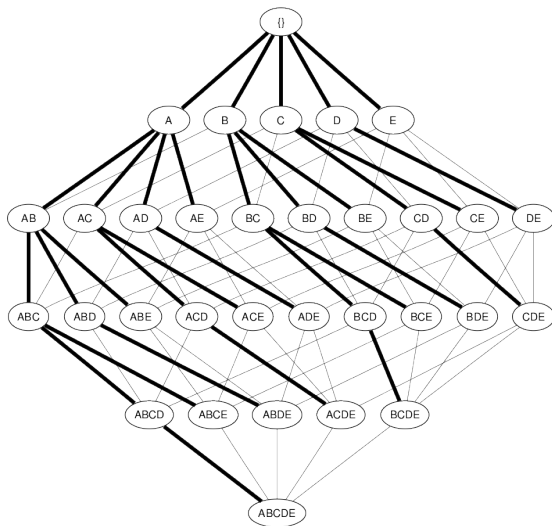
Given a dataset \mathcal{D} of objects described with Boolean attributes in \mathcal{A} and a minimal frequency $\mu \in \mathbb{N}$:

$$\forall (X, Y) \in 2^{\mathcal{A}} \times 2^{\mathcal{A}}, X \subseteq Y \Rightarrow \left(f(X, \mathcal{D}) < \mu \Rightarrow f(Y, \mathcal{D}) < \mu \right) .$$

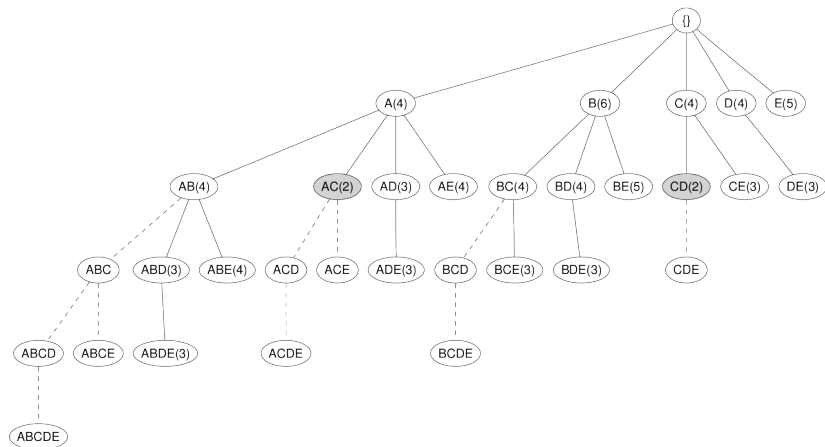
	a_1	a_2	a_3
o_1	×	×	×
o_2	×	×	
o_3		×	
o_4			×

$$\begin{aligned} f(\emptyset, \mathcal{D}) &= 4 \\ f(\{a_3\}, \mathcal{D}) &= 2 \\ f(\{a_1, a_3\}, \mathcal{D}) &= 1 \\ f(\{a_1, a_2, a_3\}, \mathcal{D}) &= 1 \end{aligned}$$

Pruning the enumeration tree ($\mu = 3$)



Pruning the enumeration tree ($\mu = 3$)



APriori enumeration

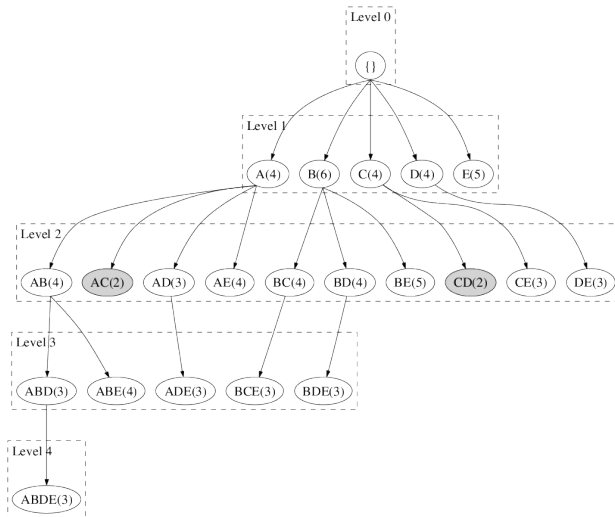
To check the frequency of every parent, the enumeration tree must be traversed breadth-first.

APriori enumeration

To check the frequency of every parent, the enumeration tree must be traversed breadth-first.

The two first parents (in the lexicographic order \preceq) are close to each other in the prefix-based tree. Indeed, they only differ by the last attribute. Instead of considering all possible children of a parent, APriori searches this second parent and, if found, enumerate, by union, their child.

Level-wise enumeration of the itemsets



APriori algorithm

Input: \mathcal{A}, \mathcal{D} as an array of subsets of $\mathcal{A}, \mu \in \mathbb{N}$

Output: $\{X \subseteq \mathcal{A} \mid f(X, \mathcal{D}) \geq \mu\}$

$\mathcal{P} \leftarrow \{\{a\} \mid a \in \mathcal{A}\}$

while $\mathcal{P} \neq \emptyset$ **do**

$\mathcal{P} \leftarrow \text{output_frequent}(\mathcal{P}, \mathcal{D}, \mu)$

$\mathcal{P} \leftarrow \text{children}(\mathcal{P})$

end while

children

Input: A lexicographically ordered collection $\mathcal{P} \subseteq 2^A$

Output: $\{X \subseteq 2^A \mid \forall a \in X, X \setminus \{a\} \in \mathcal{P}\}$ lexico. ordered

$\mathcal{P}' \leftarrow \emptyset$

for all $P_1 \in \mathcal{P}$ **do**

for all $P_2 \in \{P_2 \in \mathcal{P} \mid P_1 \prec P_2 \wedge P_2 \setminus \{\text{last}(P_2)\} = P_1 \setminus \{\text{last}(P_1)\}\}$

do

$X \leftarrow P_1 \cup P_2$

if $\forall P \in \{X \setminus \{\text{member}(X)\} \mid P_2 \prec P\}, P \in \mathcal{P}$ **then**

$\mathcal{P}' \leftarrow \mathcal{P}' \cup \{X\}$

end if

end for

end for

return \mathcal{P}'

children

Input: A lexicographically ordered **collection** $\mathcal{P} \subseteq 2^A$

Output: $\{X \subseteq 2^A \mid \forall a \in X, X \setminus \{a\} \in \mathcal{P}\}$ lexico. ordered

$\mathcal{P}' \leftarrow \emptyset$

for all $P_1 \in \mathcal{P}$ **do**

for all $P_2 \in \{P_2 \in \mathcal{P} \mid P_1 \prec P_2 \wedge P_2 \setminus \{\text{last}(P_2)\} = P_1 \setminus \{\text{last}(P_1)\}\}$

do

$X \leftarrow P_1 \cup P_2$

if $\forall P \in \{X \setminus \{\text{member}(X)\} \mid P_2 \prec P\}, P \in \mathcal{P}$ **then**

$\mathcal{P}' \leftarrow \mathcal{P}' \cup \{X\}$

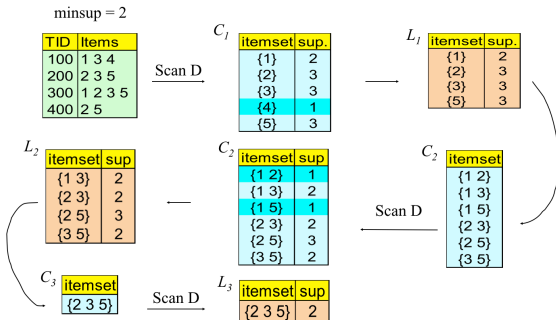
end if

end for

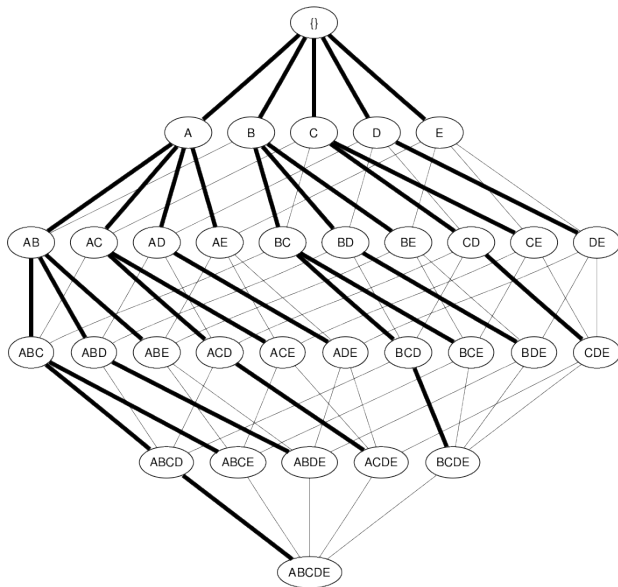
end for

return \mathcal{P}'

Example



Depth-first enumeration of the itemsets



Fail-first principle

Observation

An itemset has a greater probability to be infrequent if the frequencies of its attributes, taken individually, are low.

Fail-first principle

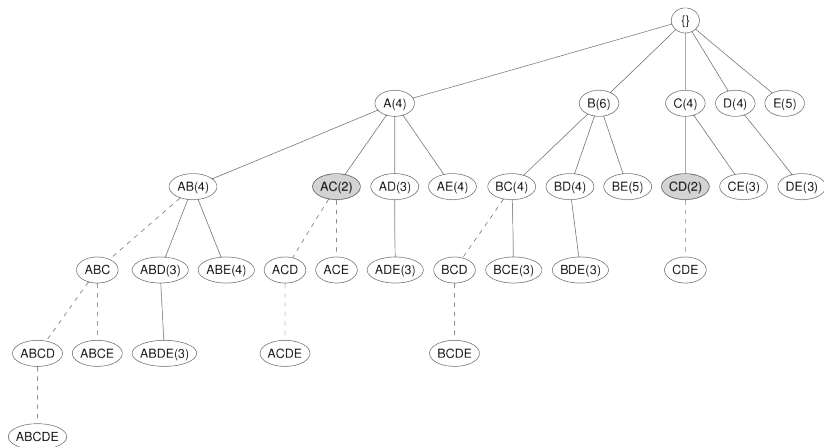
Observation

An itemset has a greater probability to be infrequent if the frequencies of its attributes, taken individually, are low.

Fail-first principle

Taking advantage of the anti-monotonicity of the frequency, it is better to enumerate the infrequent itemsets first.

The *unbalanced* enumeration tree



Heuristic choice of a lexicographic order

Input: \mathcal{A}, \mathcal{D} as an array of subsets of $\mathcal{A}, \mu \in \mathbb{N}$

Output: $\{X \subseteq \mathcal{A} \mid f(X, \mathcal{D}) \geq \mu\}$

$\mathcal{P} \leftarrow \{\{a\} \mid a \in \mathcal{A}\}$

while $\mathcal{P} \neq \emptyset$ **do**

$\mathcal{P} \leftarrow \text{output_frequent}(\mathcal{P}, \mathcal{D}, \mu)$

$\mathcal{P} \leftarrow \text{children}(\mathcal{P})$

end while

Whatever the order on \mathcal{A} , the frequent itemsets are correctly and completely listed...

Heuristic choice of a lexicographic order

Input: \mathcal{A}, \mathcal{D} as an array of subsets of \mathcal{A} , $\mu \in \mathbb{N}$

Output: $\{X \subseteq \mathcal{A} \mid f(X, \mathcal{D}) \geq \mu\}$

$\mathcal{P} \leftarrow \{\{a\} \mid a \in \mathcal{A}\}$ **ordered by increasing $f(\{a\}, \mathcal{D})$**

while $\mathcal{P} \neq \emptyset$ **do**

$\mathcal{P} \leftarrow \text{output_frequent}(\mathcal{P}, \mathcal{D}, \mu)$

$\mathcal{P} \leftarrow \text{children}(\mathcal{P})$

end while

Whatever the order on \mathcal{A} , the frequent itemsets are correctly and completely listed... but this heuristic choice usually leads to the enumeration of much less infrequent itemsets.

Iterative computation of the supports

Theorem

Given the objects in \mathcal{O} described with the Boolean attributes in \mathcal{A} , i. e., the dataset $\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$ and $k \in \mathbb{N}$ itemsets $(P_i)_{i=1..k} \in (2^{\mathcal{A}})^k$:

$$\{o \in \mathcal{O} \mid \{o\} \times \cup_{i=1}^k P_i \subseteq \mathcal{D}\} = \cap_{i=1}^k \{o \in \mathcal{O} \mid \{o\} \times P_i \subseteq \mathcal{D}\} .$$

	a_1	a_2	a_3
o_1	x	x	x
o_2	x	x	
o_3		x	
o_4			x

$$\begin{array}{rcl} \{o \in \mathcal{O} \mid \{o\} \times \{a_1\} \subseteq \mathcal{D}\} & = & \{o_1, o_2\} \\ \{o \in \mathcal{O} \mid \{o\} \times \{a_2\} \subseteq \mathcal{D}\} & = & \{o_1, o_2, o_3\} \\ \{o \in \mathcal{O} \mid \{o\} \times \{a_3\} \subseteq \mathcal{D}\} & = & \{o_1, o_4\} \\ \hline \{o \in \mathcal{O} \mid \{o\} \times \{a_1, a_2, a_3\} \subseteq \mathcal{D}\} & = & \{o_1\} \end{array}$$

Iterative computation of the supports

Theorem

Given the objects in \mathcal{O} described with the Boolean attributes in \mathcal{A} , i. e., the dataset $\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$ and $k \in \mathbb{N}$ itemsets $(P_i)_{i=1..k} \in (2^{\mathcal{A}})^k$:

$$\{o \in \mathcal{O} \mid \{o\} \times \cup_{i=1}^k P_i \subseteq \mathcal{D}\} = \cap_{i=1}^k \{o \in \mathcal{O} \mid \{o\} \times P_i \subseteq \mathcal{D}\} .$$

	a_1	a_2	a_3
o_1	x	x	x
o_2	x	x	
o_3		x	
o_4			x

$$\begin{array}{rcl} \{o \in \mathcal{O} \mid \{o\} \times \{a_1, a_2\} \subseteq \mathcal{D}\} & = & \{o_1, o_2\} \\ \{o \in \mathcal{O} \mid \{o\} \times \{a_3\} \subseteq \mathcal{D}\} & = & \{o_1, o_4\} \\ \hline \{o \in \mathcal{O} \mid \{o\} \times \{a_1, a_2, a_3\} \subseteq \mathcal{D}\} & = & \{o_1\} \end{array}$$

Iterative computation of the supports

Theorem

Given the objects in \mathcal{O} described with the Boolean attributes in \mathcal{A} , i. e., the dataset $\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$ and $k \in \mathbb{N}$ itemsets $(P_i)_{i=1..k} \in (2^{\mathcal{A}})^k$:

$$\{o \in \mathcal{O} \mid \{o\} \times \cup_{i=1}^k P_i \subseteq \mathcal{D}\} = \cap_{i=1}^k \{o \in \mathcal{O} \mid \{o\} \times P_i \subseteq \mathcal{D}\} .$$

	a_1	a_2	a_3
o_1	x	x	x
o_2	x	x	
o_3		x	
o_4			x

$$\begin{array}{rcl} \{o \in \mathcal{O} \mid \{o\} \times \{a_1, a_2\} \subseteq \mathcal{D}\} & = & \{o_1, o_2\} \\ \{o \in \mathcal{O} \mid \{o\} \times \{a_1, a_3\} \subseteq \mathcal{D}\} & = & \{o_1\} \\ \hline \{o \in \mathcal{O} \mid \{o\} \times \{a_1, a_2, a_3\} \subseteq \mathcal{D}\} & = & \{o_1\} \end{array}$$

Vertical representation of the data

Relational representation:

$$\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$$

	a_1	a_2	\dots	a_n
o_1	$d_{1,1}$	$d_{1,2}$	\dots	$d_{1,n}$
o_2	$d_{2,1}$	$d_{2,2}$	\dots	$d_{2,n}$
\vdots	\vdots	\vdots	\ddots	\vdots
o_m	$d_{m,1}$	$d_{m,2}$	\dots	$d_{m,n}$

where $d_{i,j} \in \{\text{true}, \text{false}\}$

Vertical representation: \mathcal{D} is an array of subsets of \mathcal{O}

$$i_1 \quad i_2 \quad \dots \quad i_n$$

where $i_j \subseteq \mathcal{O}$

Vertical representation of the data

Relational representation:

$$\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$$

	a_1	a_2	\dots	a_n
o_1	$d_{1,1}$	$d_{1,2}$	\dots	$d_{1,n}$
o_2	$d_{2,1}$	$d_{2,2}$	\dots	$d_{2,n}$
\vdots	\vdots	\vdots	\ddots	\vdots
o_m	$d_{m,1}$	$d_{m,2}$	\dots	$d_{m,n}$

where $d_{i,j} \in \{\text{true}, \text{false}\}$

Vertical representation: \mathcal{D} is an array of subsets of \mathcal{O}

$$i_1 \quad i_2 \quad \dots \quad i_n$$

where $i_j \subseteq \mathcal{O}$

For a linear time intersection of the i_j , they are sorted (arbitrary order on \mathcal{O}) in a pre-processing step and the support of any enumerated itemset X will respect this order.

Vertical representation of the data

Relational representation:

$$\mathcal{D} \subseteq \mathcal{O} \times \mathcal{A}$$

	a_1	a_2	\dots	a_n
o_1	$d_{1,1}$	$d_{1,2}$	\dots	$d_{1,n}$
o_2	$d_{2,1}$	$d_{2,2}$	\dots	$d_{2,n}$
\vdots	\vdots	\vdots	\ddots	\vdots
o_m	$d_{m,1}$	$d_{m,2}$	\dots	$d_{m,n}$

where $d_{i,j} \in \{\text{true}, \text{false}\}$

Vertical representation: \mathcal{D} is an array of subsets of \mathcal{O}

$$i_1 \quad i_2 \quad \dots \quad i_n$$

where $i_j \subseteq \mathcal{O}$

Unless the minimal relative frequency is very low, storing the support on *bitsets* provide the best space and time performances.

Eclat enumeration

Like APriori:

- ▶ The anti-monotonicity of the frequency prunes the enumeration tree;

Eclat enumeration

Like APriori:

- ▶ The anti-monotonicity of the frequency prunes the enumeration tree;
- ▶ the two first parents (in the lexicographic order \preceq) are searched to generate by union their child;

Eclat enumeration

Like APriori:

- ▶ The anti-monotonicity of the frequency prunes the enumeration tree;
- ▶ the two first parents (in the lexicographic order \preceq) are searched to generate by union their child;
- ▶ Ordering the attributes by increasing frequency heuristically leads to the enumeration of much less infrequent itemsets.

Eclat enumeration

Like APriori:

- ▶ The anti-monotonicity of the frequency prunes the enumeration tree;
- ▶ the two first parents (in the lexicographic order \preceq) are searched to generate by union their child;
- ▶ Ordering the attributes by increasing frequency heuristically leads to the enumeration of much less infrequent itemsets.

However:

- ▶ the frequency of the other parents is not checked;

Eclat enumeration

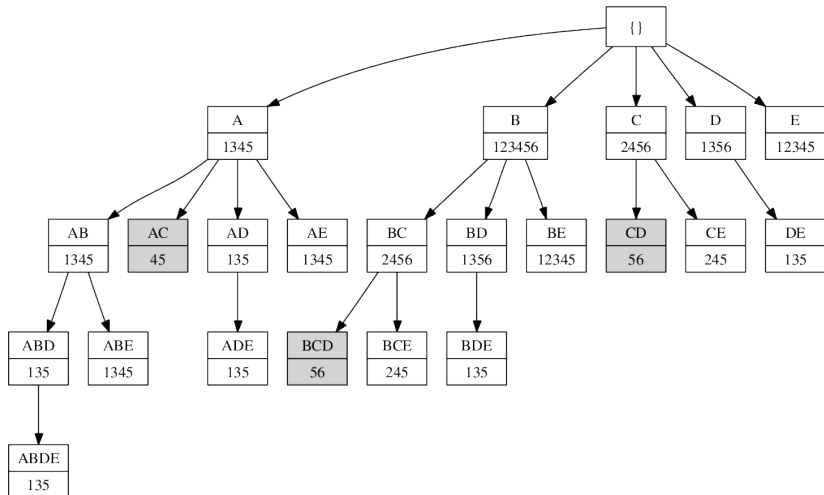
Like APriori:

- ▶ The anti-monotonicity of the frequency prunes the enumeration tree;
- ▶ the two first parents (in the lexicographic order \preceq) are searched to generate by union their child;
- ▶ Ordering the attributes by increasing frequency heuristically leads to the enumeration of much less infrequent itemsets.

However:

- ▶ the frequency of the other parents is not checked;
- ▶ thanks to that, the enumeration tree is traversed in a less memory-hungry way (but, contrary to APriori, the supports of the frequent itemsets are stored too).

Pruning the enumeration tree ($\mu = 3$)



Eclat algorithm

Input: \mathcal{A}, \mathcal{D} as an array of subsets of $\mathcal{O}, \mu \in \mathbb{N}$

Output: $\{X \subseteq \mathcal{A} \mid f(X, \mathcal{D}) \geq \mu\}$

Eclat(\mathcal{P}, μ) {Initial call: $\mathcal{P} = \{(\{a_j\}, i_j) \mid j = 1..m \wedge |i_j| \geq \mu\}$ }

for all $(P_1, i_{P_1}) \in \mathcal{P}$ **do**

output(P_1)

$\mathcal{P}' \leftarrow \emptyset$

for all $(P_2, i_{P_2}) \in \{(P_2, i_{P_2}) \in \mathcal{P} \mid P_1 \prec P_2\}$ **do**

$i \leftarrow i_{P_1} \cap i_{P_2}$

if $|i| \geq \mu$ **then**

$\mathcal{P}' \leftarrow \mathcal{P}' \cup \{(P_1 \cup P_2, i)\}$

end if

end for

Eclat(\mathcal{P}', μ)

end for

Pattern flooding

$$\mu = 2$$

\mathcal{O}	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
o_1	x	x	x	x	x										
o_2	x	x	x	x	x										
o_3	x	x	x	x	x										
o_4						x	x	x	x	x					
o_5						x	x	x	x	x					
o_6						x	x	x	x	x					
o_7											x	x	x	x	x
o_8											x	x	x	x	x

- How many frequent patterns?

Pattern flooding

$$\mu = 2$$

\mathcal{O}	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
o_1	x	x	x	x	x										
o_2	x	x	x	x	x										
o_3	x	x	x	x	x										
o_4						x	x	x	x	x					
o_5						x	x	x	x	x					
o_6						x	x	x	x	x					
o_7											x	x	x	x	x
o_8											x	x	x	x	x

- How many frequent patterns? $1 + (2^5 - 1) \times 3 = 94$ patterns

Pattern flooding

$$\mu = 2$$

\mathcal{O}	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
\mathcal{O}_1	x	x	x	x	x										
\mathcal{O}_2	x	x	x	x	x										
\mathcal{O}_3	x	x	x	x	x										
\mathcal{O}_4						x	x	x	x	x					
\mathcal{O}_5						x	x	x	x	x					
\mathcal{O}_6						x	x	x	x	x					
\mathcal{O}_7											x	x	x	x	x
\mathcal{O}_8											x	x	x	x	x

- How many frequent patterns? $1 + (2^5 - 1) \times 3 = 94$ patterns but actually 4 interesting ones:
 $\{\}, \{a_1, a_2, a_3, a_4, a_5\}, \{a_6, a_7, a_8, a_9, a_{10}\}, \{a_{11}, a_{12}, a_{13}, a_{14}, a_{15}\}.$

Pattern flooding

$$\mu = 2$$

\mathcal{O}	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
o_1	x	x	x	x	x										
o_2	x	x	x	x	x										
o_3	x	x	x	x	x										
o_4						x	x	x	x	x					
o_5						x	x	x	x	x					
o_6						x	x	x	x	x					
o_7											x	x	x	x	x
o_8											x	x	x	x	x

- ▶ How many frequent patterns? $1 + (2^5 - 1) \times 3 = 94$ patterns but actually 4 interesting ones:
 $\{\}, \{a_1, a_2, a_3, a_4, a_5\}, \{a_6, a_7, a_8, a_9, a_{10}\}, \{a_{11}, a_{12}, a_{13}, a_{14}, a_{15}\}.$

☞ the need to focus on a **condensed representation** of frequent patterns.

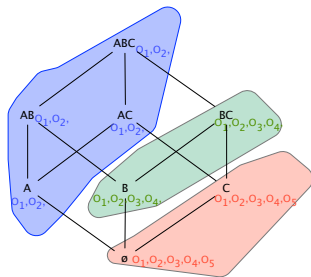


Toon Calders, Christophe Rigotti, Jean-François Boulicaut: A Survey on Condensed Representations for Frequent Sets. Constraint-Based Mining and Inductive Databases 2004: 64-80.

Closed and Free Patterns

Equivalence classes based on support.

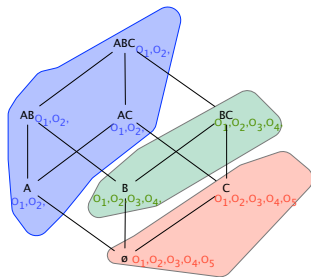
\mathcal{O}	A	B	C
\mathcal{O}_1	×	×	×
\mathcal{O}_2	×	×	×
\mathcal{O}_3		×	×
\mathcal{O}_4		×	×
\mathcal{O}_5			×




Closed and Free Patterns

Equivalence classes based on support.

\mathcal{O}	A	B	C
o_1	×	×	×
o_2	×	×	×
o_3		×	×
o_4		×	×
o_5			×



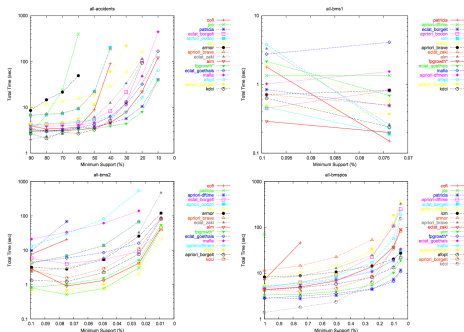
- ▶ **Closed** patterns are maximal element of each equivalence class: ABC , BC , and C .
- ▶ **Generators** or **Free** patterns are minimal elements (not necessary unique) of each equivalent class: $\{\}$, A and B

 Y. Bastide, et al. Mining frequent patterns with counting inference. SIGKDD Expl., 2000.

Few researchers (in DM) are aware about this strong intersection.

A strong intersection with Formal Concept Analysis (Ganter and Wille, 1999).

- ▶ transactional DB \equiv **formal context** is a triple $K = (G, M, I)$, where G is a set of objects, M is a set of attributes, and $I \subseteq G \times M$ is a binary relation called incidence that expresses which objects have which attributes.
- ▶ closed itemset \equiv **concept intent**
- ▶ FCA gives the mathematical background about closed patterns.
- ▶ Algorithms: **LCM** is an efficient implementation of **Close By One**. (Sergei O. Kuznetsov, 1993).



(FIMI Workshop@ICDM, 2003 and 2004)

The FIM Era: during more than a decade, only ms were worth it!

Even if the complete collection of frequent itemsets is known useless, the main objective of many algorithms is to earn ms according to their competitors!!

What about the end-user (and the pattern interestingness)?

→ partially answered with constraints.

Outline

Introduction

Frequent Itemset Mining

Frequent Itemset Mining

Condensed Representations

Constraint-based Pattern Mining

Pattern constraints

Constraints are needed for:

- ▶ only retrieving patterns that describe an interesting subgroup of the data
- ▶ making the extraction feasible

Pattern constraints

Constraints are needed for:

- ▶ only retrieving patterns that describe an interesting subgroup of the data
- ▶ making the extraction feasible

Constraint properties are used to infer constraint values on (many) patterns without having to evaluate them individually.

Pattern constraints

Constraints are needed for:

- ▶ only retrieving patterns that describe an interesting subgroup of the data
- ▶ making the extraction feasible

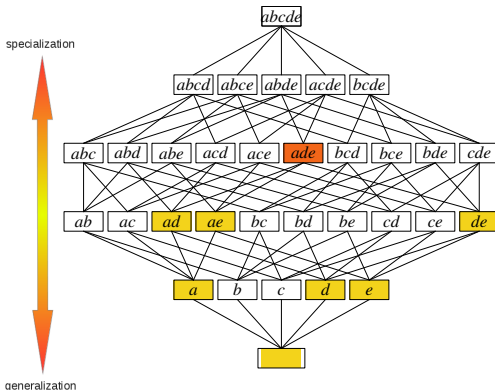
Constraint properties are used to infer constraint values on (many) patterns without having to evaluate them individually.

→ They are defined up to the partial order \preceq used for listing the patterns

Constraint properties - 1

Monotone constraint

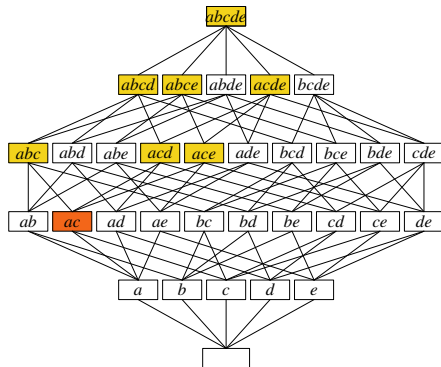
$$\forall \varphi_1 \preceq \varphi_2, \mathcal{C}(\varphi_1, \mathcal{D}) \Rightarrow \mathcal{C}(\varphi_2, \mathcal{D})$$



$$\mathcal{C}(\varphi, \mathcal{D}) \equiv b \in \varphi \vee c \in \varphi$$

Anti-monotone constraint

$$\forall \varphi_1 \preceq \varphi_2, \mathcal{C}(\varphi_2, \mathcal{D}) \Rightarrow \mathcal{C}(\varphi_1, \mathcal{D})$$

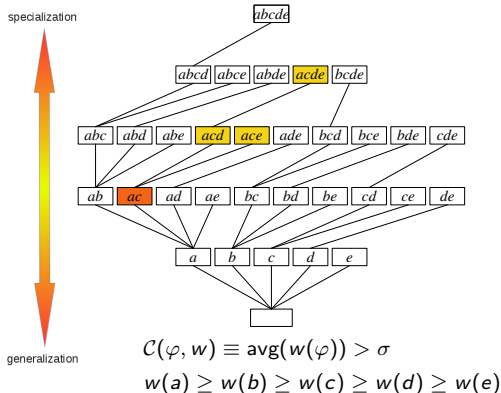


$$\mathcal{C}(\varphi, \mathcal{D}) \equiv a \notin \varphi \wedge c \notin \varphi$$

Constraint properties - 2

Convertible constraints

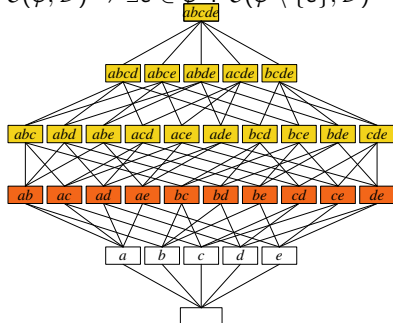
\preceq is extended to the prefix order \leq so that $\forall \varphi_1 \leq \varphi_2, \mathcal{C}(\varphi_2, \mathcal{D}) \Rightarrow \mathcal{C}(\varphi_1, \mathcal{D})$



Pei and Han – 2000

Loose AM constraints

$\mathcal{C}(\varphi, \mathcal{D}) \Rightarrow \exists e \in \varphi : \mathcal{C}(\varphi \setminus \{e\}, \mathcal{D})$



Bonchi and Lucchese – 2007

Examples

$v \in P$	M
$P \supseteq S$	M
$P \subseteq S$	AM
$\min(P) \leq \sigma$	AM
$\min(P) \geq \sigma$	M
$\max(P) \leq \sigma$	M
$\max(P) \geq \sigma$	AM
$\text{range}(P) \leq \sigma$	AM
$\text{range}(P) \geq \sigma$	M
$\text{avg}(P) \theta \sigma, \theta \in \{\leq, =, \geq\}$	Convertible
$\text{var}(w(\varphi)) \leq \sigma$	LAM

Outline

Introduction

Frequent Itemset Mining

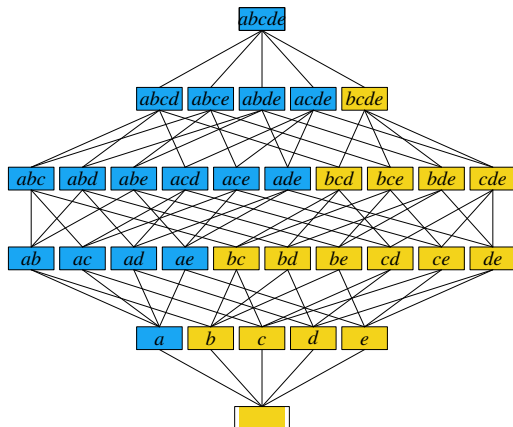
Constraint-based Pattern Mining

Constraint properties

Algorithmic principles

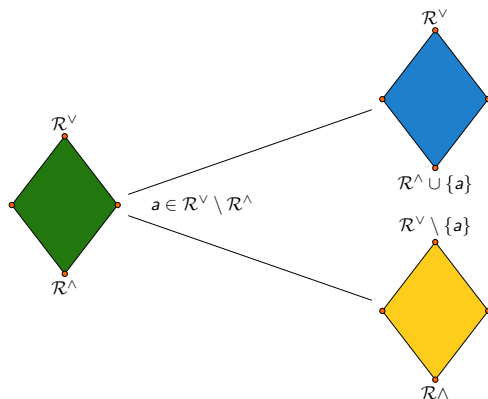
Enumeration strategy

Binary partition: the element 'a' is enumerated



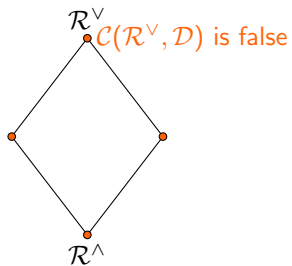
Enumeration strategy

Binary partition: the element 'a' is enumerated



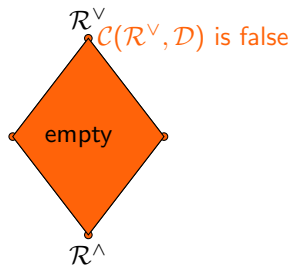
Constraint evaluation

Monotone constraint



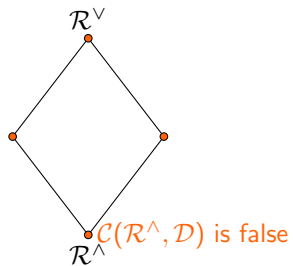
Constraint evaluation

Monotone constraint



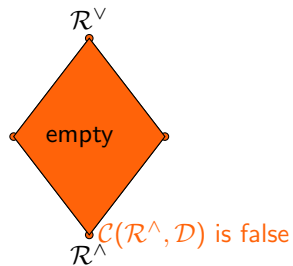
Constraint evaluation

Anti-monotone constraint



Constraint evaluation

Anti-monotone constraint





A new class of constraints

Piecewise monotone and anti-monotone constraints¹


1. \mathcal{C} involves p times the pattern φ : $\mathcal{C}(\varphi, \mathcal{D}) = f(\varphi_1, \dots, \varphi_p, \mathcal{D})$
2. $f_{i,\varphi}(x) = (\varphi_1, \dots, \varphi_{i-1}, x, \varphi_{i+1}, \dots, \varphi_p, \mathcal{D})$
3. $\forall i = 1 \dots p$, $f_{i,\varphi}$ is either monotone or anti-monotone:

$$\forall x \preceq y, \begin{cases} f_{i,\varphi}(x) \Rightarrow f_{i,\varphi}(y) \text{ iff } f_{i,\varphi} \text{ is monotone} \\ f_{i,\varphi}(y) \Rightarrow f_{i,\varphi}(x) \text{ iff } f_{i,\varphi} \text{ is anti-monotone} \end{cases}$$

 L. Cerf, J. Besson, C. Robardet, J-F. Boulicaut: Closed patterns meet n-ary relations. TKDD 3(1) (2009)

 A. Buzmakov, S. O. Kuznetsov, A. Napoli: Fast Generation of Best Interval Patterns for Nonmonotonic Constraints. ECML/PKDD (2) 2015: 157-172

¹A.k.a. primitive-based constraints

 A. Soulet, B. Crémilleux: Mining constraint-based patterns using automatic relaxation. Intell. Data Anal. 13(1): 109-133 (2009)

An example

- ▶ $\forall e, w(e) \geq 0$
- ▶ $\mathcal{C}(\varphi, w) \equiv \text{avg}(w(\varphi)) > \sigma \equiv \frac{\sum_{e \in \varphi} w(e)}{|\varphi|} > \sigma$.

$\mathcal{C}(\varphi, \mathcal{D})$ is piecewise monotone and anti-monotone with

$$f(\varphi_1, \varphi_2, \mathcal{D}) = \frac{\sum_{e \in \varphi_1} w(e)}{|\varphi_2|}$$

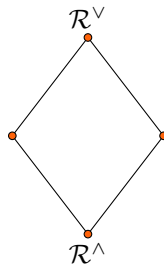
$\forall x \preceq y$,

- ▶ $f_{1,\varphi}$ is monotone: $f(x, \varphi_2, \mathcal{D}) = \frac{\sum_{e \in x} w(e)}{|\varphi_2|} > \sigma \Rightarrow \frac{\sum_{e \in y} w(e)}{|\varphi_2|} > \sigma$
- ▶ $f_{2,\varphi}$ is anti-monotone:
 $f(\varphi_1, y, \mathcal{D}) = \frac{\sum_{e \in \varphi_1} w(e)}{|y|} > \sigma \Rightarrow \frac{\sum_{e \in \varphi_1} w(e)}{|x|} > \sigma$

Piecewise constraint exploitation

Evaluation

$$\text{If } f(\mathcal{R}^\vee, \mathcal{R}^\wedge, \mathcal{D}) = \frac{\sum_{e \in \mathcal{R}^\vee} w(e)}{|\mathcal{R}^\wedge|}$$



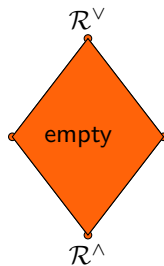
Propagation

- ▶ $\exists e \in \mathcal{R}^\vee \setminus \mathcal{R}^\wedge$ such that $f(\mathcal{R}^\vee \setminus \{e\}, \mathcal{R}^\wedge, \mathcal{D}) \leq \sigma$, then e is moved in \mathcal{R}^\wedge
- ▶ $\exists e \in \mathcal{R}^\vee \setminus \mathcal{R}^\wedge$ such that $f(\mathcal{R}^\vee, \mathcal{R}^\wedge \cup \{e\}, \mathcal{D}) \leq \sigma$, then e is removed from \mathcal{R}^\vee

Piecewise constraint exploitation

Evaluation

If $f(\mathcal{R}^\vee, \mathcal{R}^\wedge, \mathcal{D}) = \frac{\sum_{e \in \mathcal{R}^\vee} w(e)}{|\mathcal{R}^\wedge|} \leq \sigma$
then \mathcal{R} is empty.



Propagation

- ▶ $\exists e \in \mathcal{R}^\vee \setminus \mathcal{R}^\wedge$ such that $f(\mathcal{R}^\vee \setminus \{e\}, \mathcal{R}^\wedge, \mathcal{D}) \leq \sigma$, then e is moved in \mathcal{R}^\wedge
- ▶ $\exists e \in \mathcal{R}^\vee \setminus \mathcal{R}^\wedge$ such that $f(\mathcal{R}^\vee, \mathcal{R}^\wedge \cup \{e\}, \mathcal{D}) \leq \sigma$, then e is removed from \mathcal{R}^\vee


Algorithmic principles

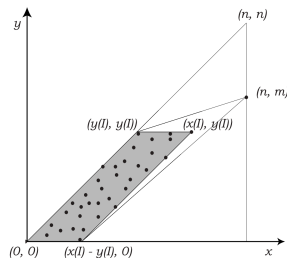
Function Generic_CBPM_enumeration($\mathcal{R}^\vee, \mathcal{R}^\wedge$)

```
1: if Check_constraints( $\mathcal{R}^\wedge, \mathcal{R}^\vee$ ) then
2:   ( $\mathcal{R}^\wedge, \mathcal{R}^\vee$ )  $\leftarrow$  Constraint_Propagation( $\mathcal{R}^\wedge, \mathcal{R}^\vee$ )
3:   if  $\mathcal{R}^\wedge = \mathcal{R}^\vee$  then
4:     output  $\mathcal{R}^\wedge$ 
5:   else
6:     for all  $e \in \mathcal{R}^\vee \setminus \mathcal{R}^\wedge$  do
7:       Generic_CBPM_Enumeration( $\mathcal{R}^\wedge \cup \{e\}, \mathcal{R}^\vee$ )
8:       Generic_CBPM_Enumeration( $\mathcal{R}^\wedge, \mathcal{R}^\vee \setminus \{e\}$ )
9:     end for
10:  end if
11: end if
```

Tight Upper-bound computation

- ▶ Convex measures can be taken into account by computing some upper bounds with \mathcal{R}^\wedge and \mathcal{R}^\vee .
- ▶ Branch and bound enumeration

 Shinichi Morishita, Jun Sese: Traversing Itemset Lattice with Statistical Metric Pruning. PODS 2000: 226-236



Studying constraints \equiv looking for efficient and effective upper bound in a branch and bound algorithm !

Case Studies

Mining of

- ▶ Multidimensional and multi-level sequences [ACM TKDD 2010]
- ▶ Maximal homogeneous clique set [KAIS 2014]
- ▶ Rules in Boolean tensors/dynamic graphs [SDM 11, IDA J. 2013]
- ▶ Topological patterns in static attributed graphs [TKDE 2013]
- ▶ Temporal dependencies in streams [KDD'13, IDA J. 2016]
- ▶ Trend dynamic sub-graphs [DS 12, PKDD 13, IDA 14]
- ▶ δ -free sequential patterns [ICDM'14]
- ▶ Triggering patterns [ASONAM 14, Social Network Analysis J. 2015]
- ▶ Events in geo-localized social medias [ECMLPKDD'15]
- ▶ Pairwise change behavior [ECMLPKDD'17]
- ▶ Exceptional attributed Graphs [Machine Learning 2017, ICDM'16, ComplexNetwork17]

Toward declarativity

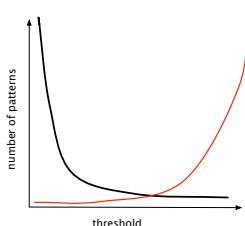
Why declarative approaches?

- ▶ for each problem, do not write a solution from scratch

Declarative approaches:

- ▶ CP approaches (Khiari et al., CP10, Guns et al., TKDE 2013)
- ▶ SAT approaches (Boudane et al., IJCAI16, Jabbour et al., CIKM13)
- ▶ ILP approaches (Mueller et al., DS10, Babaki et al., CPAIOR14, Ouali et al. IJCAI16)
- ▶ ASP approaches (Gebser et al., IJCAI16)

Thresholding problem



- ▶ A too stringent threshold: trivial patterns
- ▶ A too weak threshold: too many patterns, unmanageable and diversity not necessary assured.
- ▶ Some attempts to tackle this issue:
 - ▶ Interestingness is not a dichotomy! [?]
 - ▶ Taking benefit from hierarchical relationships [?, ?]
- ▶ But setting thresholds remains an issue in pattern mining.

Constraint-based pattern mining:

concluding remarks

- ▶ how to fix thresholds?
- ▶ how to handle numerous patterns including non-informative patterns? how to get a global picture of the set of patterns?
- ▶ how to design the proper constraints/preferences?

END