ENS DE LYON

Pattern Mining (part III)

DBDM, ENS de Lyon, 04th May 2017

Mehdi Kaytoue, mehdi.kaytoue@insa-lyon.fr

Context: understanding a (natural) phenomena

Olfaction

- Ability to perceive odors
- Complex phenomenon from molecule to perception

Challenges

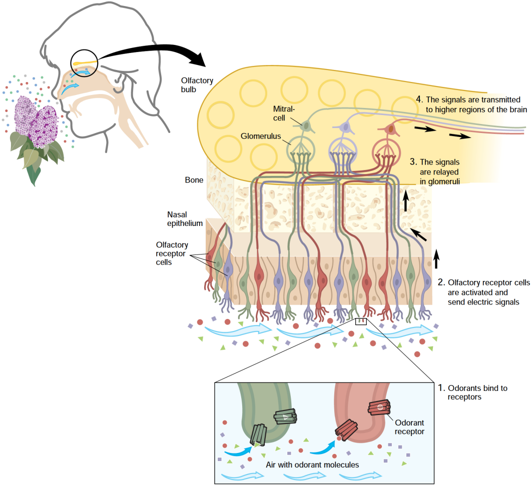
- Established links between physicochemical properties and olfactory qualities of molecules
- Difficulties to formulate/propose rules

Applications

- Fundamental neuroscience research
- Industry (agri-food industry, perfume industry, ...)
- Health (anosmia, ...)

U.J. Meierhenrich, J. Golebiowski, X. Fernandez, and D. Cabrol-Bass
The Molecular Basis of Olfactory Chemoreception.
In *Angewandte Chemie International Edition*, 2004.

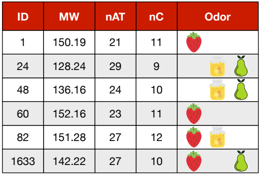
Olfaction


Olfactory bulb
Mitral-cell
Glomerulus
4. The signals are transmitted to higher regions of the brain
3. The signals are relayed in glomeruli
Bone
Nasal epithelium
Olfactory receptor cells
2. Olfactory receptor cells are activated and send electric signals
1. Odorants bind to receptors
Odorant receptor
Air with odorant molecules

Buck L, Axel R.
A novel multigene family may encode odorant receptors: a molecular basis for odor recognition, Nobel Prize in Physiology or Medecine in 2004,

Towards discriminant pattern mining

- How to characterize and describe the relationships between the molecular properties and olfactory qualities?



ID	MW	nAT	nC	Odor
1	150.19	21	11	🍓
24	128.24	29	9	🍯 🍐
48	136.16	24	10	🍯 🍐
60	152.16	23	11	🍓
82	151.28	27	12	🍓 🍯
1633	142.22	27	10	🍓 🍐

Toy dataset

- Can we build predictive models? To some extent only because of inter/intra individual variability and with very specific datasets (Atlas)

  A. Keller et al.
  Predicting human olfactory perception from chemical features of odor molecules,
  In *Science*, 355(6327):820–826, 2017.

- What features? 1800 numerical attributes but several representations are possibles (molecular graph, 2/3D, smiley...)

Encode the data and mining patterns that discriminates odors

- Discover hypotheses, features and build intelligible classifiers

Outline

1. Discriminant Pattern Discovery
2. Complex Data Mining
3. Diverse Pattern Set Discovery
4. Pattern-based classification
5. Concluding remarks

Outline

1. Discriminant Pattern Discovery
    1.1 Problem settings
    1.2 Subgroup Discovery
    1.3 Exceptional Model Mining
    1.4 The problems

Outline

1. Discriminant Pattern Discovery
    1.1 Problem settings
    1.2 Subgroup Discovery
    1.3 Exceptional Model Mining
    1.4 The problems

Mining patterns in labeled data

Definition (Dataset)

Let $\mathcal{O}$, $\mathcal{A}$ and C be respectively a set of objects, a set of attributes and a target attribute (the class). The domain of an attribute $a \in \mathcal{A}$ is $Dom(a)$ where a is either nominal or numerical. Each object is associated to a value from the domain $Dom(C)$ of the target attribute through $class : \mathcal{O} \longmapsto Dom(C)$. $\mathcal{D}(\mathcal{O}, \mathcal{A}, C, class)$ is a dataset.

ID	a	b	c	C
1	150.19	21	11	l_1
2	128.24	29	9	l_2
3	136.16	24	10	l_2
4	152.16	23	11	l_3
5	151.28	27	12	l_2
6	142.22	27	10	l_1

- A dataset is a set of tuples called entry, object, transaction...
- A tuple is described by attributes (numerical, boolean, nominal, graphs, etc.)

What are we seeking?

Intuitive definitions

- Find descriptions, generalizations, that rather cover objects of a single class label
- Find rules describing subsets of the population that are sufficiently large and statistically unusual.
- Find descriptions which induce an exceptional model compared to the whole dataset

ID	a	b	c	C
1	150.19	21	11	l_1
2	128.24	29	9	l_2
3	136.16	24	10	l_2
4	152.16	23	11	l_3
5	151.28	27	12	l_2
6	142.22	27	10	l_1

- The label distribution is known
- Can we find subgroups, sufficiently large, for which the distribution is different?

A simple example

Consider a dataset concerning people, and let the target attribute be whether the person develops lung cancer. Interesting subsets would then include the group of smokers, with an increased incidence of lung cancer, and the group of athletes, with a decreased incidence of lung cancer.

–Duivesteijn et al. 2016.

Outline

1. Discriminant Pattern Discovery
   - 1.1 Problem settings
   - 1.2 Subgroup Discovery
   - 1.3 Exceptional Model Mining
   - 1.4 The problems

Subgroup Discovery

Definition (Subgroup)

The description of a subgroup is given by $d = \langle f_1, \ldots, f_{|\mathscr{A}|} \rangle$ where each f_i is a restriction on the value domain of the attribute $a_i \in \mathscr{A}$. A restriction is either a subset of a nominal attribute domain, or an interval contained in the domain of a numerical attribute. The description d covers a set of objects called the support of the subgroup, denoted $supp(d) \subseteq \mathscr{O}$.

ID	a	b	c	C
1	150.19	21	11	l_1
2	128.24	29	9	l_2
3	136.16	24	10	l_2
4	152.16	23	11	l_3
5	151.28	27	12	l_2
6	142.22	27	10	l_1

- How many subsets of objects?
- How many descriptions?
- How many subgroups?
- hint: Galois connection

How to evaluate the quality of a subgroup?

- The ability of a subgroup to discriminate a class label is evaluated thanks to a quality measure
- The latter reflects the difference between the model induced by the subgroup on the target attribute and the model induced by the entire dataset
- A basic way, comparing label distribution: the model induced by a set of objects S is the proportion of objects of S associated to **one** class label $l \in Dom(C)$

Weighted relative accuracy

- Consider $d = \langle [128.24 \leq a \leq 151.28], [23 \leq b \leq 29] \rangle$
- We have $supp(d) = \{2, 3, 5, 6\}$
- Accuracy for a label l_2 is: $acc(d, l_2) = \frac{|\{o \in supp(d) | class(o) = l_2\}|}{|supp(d)|} = \frac{3}{4}$
- For the whole data we have $\frac{|\{o \in \mathcal{O} | class(o) = l_2\}|}{|\mathcal{O}|} = \frac{1}{2}$
- The relative accuracy is given by $p_d^{l_2} - p_0^{l_2}$
- RAcc may high for very small subgroups: a weight give more importance to frequent subgroups:
  $WRAcc(d, l_2) = \frac{|supp(d)|}{|\mathcal{O}|} \times (p_d^{l_2} - p_0^{l_2}) = \frac{1}{6}$.

ID	a	b	c	C
1	150.19	21	11	l_1
2	128.24	29	9	l_2
3	136.16	24	10	l_2
4	152.16	23	11	l_3
5	151.28	27	12	l_2
6	142.22	27	10	l_1

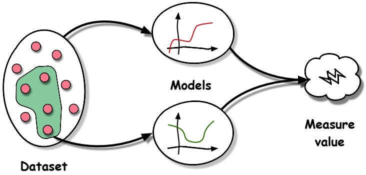
The accuracy $p_d^{l_2}$ should be taken relative to the accuracy obtained by always guessing the class, $p_0^{l_2}$, weighted by the subgroup coverage $\frac{|supp(d)|}{|\mathcal{O}|}$.

Outline

1. Discriminant Pattern Discovery
   1.1 Problem settings
   1.2 Subgroup Discovery
   1.3 Exceptional Model Mining
   1.4 The problems

Exceptional Model Mining

- A generalisation of SD: rather than one single target variable consider a more complex target concept, a numerical target, several targets possibly structured (as a tree, a graph, ...)
- Any model can be built from a subset of objects, e.g., classification, regression and clustering models
- For a chosen model, there are several ways to measure the difference between its instanciation on the the subgroup and dataset, e.g. difference between two dendograms (trees), two classification models, ...



W. Duivesteijn, A. Feelders, and A. J. Knobbe. Exceptional model mining - supervised descriptive local pattern mining with complex target concepts., In *Data Min. Knowl. Discov.*, 30(1):47–98, 2016.

Projections induce different models

- Consider that points have attributes *x*, *y* and a Boolean attribute *diag* which has a high probability to take *true* for points close to the diagonal
- In reality attribute/value combination must be discovered and are not trivial: Such patterns are also useful to propose hypotheses on the models (each point has plenty of other attributes)

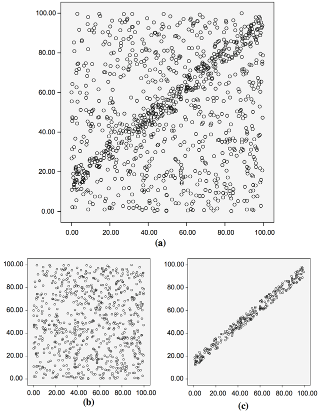


Fig. 1 A mixture of distributions. Ideally we would want to find a way to partition the original dataset from **a** into two parts: the static, depicted in **b**, and the diagonal line, depicted in **c**

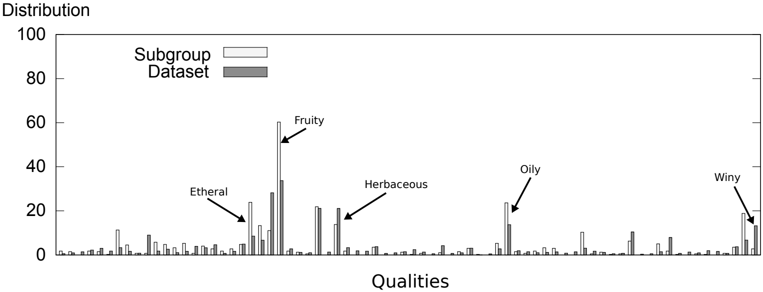
Exceptional model mining

- The (linear) correlation model with two numerical targets y_1, y_2 $\varphi(s) = supp(s) \times (correlation_s(y_1, y_2) - correlation_{\mathscr{D}}(y_1, y_2))$ with a Pearson coeff. The factor here again prevents over-fitting.
- The association model with two nominal targets
- The simple linear regression model
- The classification model
- The Bayesian network model

Exceptional model mining

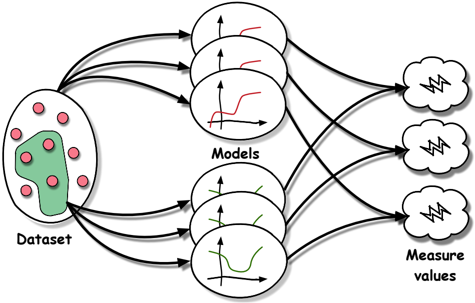
- The multi-class distribution model with WKL

$$WKL(d, L) = \frac{|supp(d)|}{|\mathcal{O}|} \sum_{l \in L} (p_d^l \log_2 \frac{p_d^l}{p_0^l})$$



Exceptional model mining

- Considering target subspaces



- ...

Other formalism

- Learning from positive and negative examples, find hypothesis h with formal concept analysis:
  - $h^{\square} \cap E_{-} = \emptyset$
  - $\exists A \subseteq E_{+} : A^{\square} = h$

Sergei O. Kuznetsov.
Galois Connections in Data Analysis: Contributions from the Soviet Era and Modern Russian Research.

In *Formal Concept Analysis* , 2005: 196-225.

- Redescription mining, given attribute sets X and Y, find $X_1 \subseteq X$ and $Y_1 \subseteq Y$ such that $jaccard(X_1, Y_1) = \frac{X_1 \cap Y_1}{X_1 \cup Y_1}$ highest as possible

Esther Galbrun, Pauli Miettinen.
From black and white to full color: extending redescription mining outside the Boolean world.
In *Statistical Analysis and Data Mining*, 5(4): 284-303 (2012).

Outline

1. Discriminant Pattern Discovery
   - 1.1 Problem settings
   - 1.2 Subgroup Discovery
   - 1.3 Exceptional Model Mining
   - 1.4 The problems

Problems

- Choosing the right model and the appropriate measure
- Representing the information with complex languages
- Mining efficiently the search search space
  - top-k-patterns are highly redundant
  - either exhaustively with smart pruning or with heuristics
  - Returning a diverse collection of non redundant patterns requires to pay attention to an exploration/exploitation trade-off
- Choosing the right patterns to build predictive models: patterns can be used as features making intelligible classifications

Outline

2. Complex Data Mining
   - 2.1 Remember... FCA!
   - 2.2 Revisiting FCA for non binary data
   - 2.3 Algorithms
   - 2.4 Other types of patterns

Outline

2. Complex Data Mining
   - 2.1 Remember... FCA!
   - 2.2 Revisiting FCA for non binary data
   - 2.3 Algorithms
   - 2.4 Other types of patterns

Formal Concept Analysis

- Emerged in the 1980's from attempts to restructure lattice theory in order to promote better communication between lattice theorists and potential users of lattice theory
- A research field leading to a seminal book and FCA dedicated conferences (ICFCA, CLA, ICCS)
- A simple, powerful and well formalized framework useful for several applications: information and knowledge processing including visualization, data analysis (mining) and knowledge management
- See also `http://www.upriss.org.uk/fca/fca.html`

B. Ganter and R. Wille
Formal Concept Analysis.
In *Springer, Mathematical foundations.*, 1999.

Formal Context

A formal context $\mathbb{K} = (G, M, I)$ consists of two sets G and M and a binary relation I between G and M. Elements of G are called objects while elements of M are called attributes of the context. The fact $(g, m) \in I$ is interpreted as “the object g has attribute m”.

	m_1	m_2	m_3	m_4	m_5	m_6
g_1	×	×				×
g_2	×	×		×		×
g_3	×	×		×	×	×
g_4	×		×		×	
g_5	×				×	
g_6	×				×	×
g_7	×		×		×	×

$G = \{g_1, ..., g_7\}$ “ostrich”, “canary”, “duck”, “shark”, “salmon”, “frog”, and “crocodile”

$M = \{m_1, .., m_6\}$ “borned from an egg”, “has feather”, “has tooth”, “fly”, “swim”, “lives in air”

Derivation operators

For a set of objects $A \subseteq G$ we define the set of attributes that all objects in A have in common as follows:

$$A' = \{m \in M \mid gIm \; \forall g \in A\}$$

Dually, for a set of attributes $B \subseteq M$, we define the set of objects that have all attributes from B as follows:

$$B' = \{g \in G \mid gIm \; \forall m \in B\}$$

Some derivation on our example

We have $\{g_1, g_2\}' = \{m_1, m_2, m_6\}$ and
$\{m_1, m_2, m_6\}' = \{g_1, g_2, g_3\}$

Formal Concepts

A formal concept of a context (G, M, I) is a pair

$$(A, B) \text{ with } A \subseteq G, B \subseteq M, A' = B \text{ and } B' = A$$

A is called the extent ; B is called its intent

$\mathfrak{B}(G, M, I)$ is the poset of all formal concepts

$$(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2 \ (\Leftrightarrow B_2 \subseteq B_1)$$

Concepts in our example

$(\{g_1, g_2, g_3\}, \{m_1, m_2, m_6\})$ as a maximal rectangle of crosses with possible row and column permutations

$$(\{g_1, g_2, g_3\}, \{m_1, m_2, m_6\}) \leq (\{g_1, g_2, g_3, g_6, g_7\}, \{m_1, m_6\})$$

Galois connection

It can be shown that operator $(.)''$, applied either to a set of objects or a set of attributes, is a closure operator. Hence we have two closure systems on G and on M. It follows that the pair $\{(.)', (.)'\}$ is a Galois connection between the power set of objects and the power set of attributes.
These mappings put in 1-1-correspondence closed sets of objects and closed sets of attributes, i.e. concept extents and concept intents. In our example, $\{g_1, g_2\}$ is not a closed set of objects, since $\{g_1, g_2\}''$ =$\{g_1, g_2, g_3\}$. Accordingly, $\{g_1, g_2, g_3\}$ is a closed set of objects hence a concept extent.

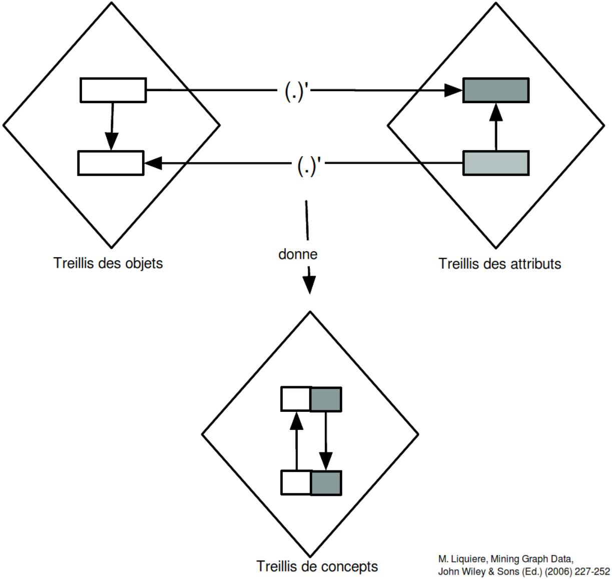
Galois connection

Let P and Q be ordered sets. A pair of maps $\phi : P \longrightarrow Q$ and $\psi : Q \longrightarrow P$ is called a Galois connection if:

- $p_1 \leq p_2 \Rightarrow \phi(p_1) \geq \phi(p_2)$
- $q_1 \leq q_2 \Rightarrow \psi(q_1) \geq \psi(q_2)$
- $p \leq \psi \circ \phi(p)$ and $q \leq \phi \circ \psi(q)$

We here have a Galois connection between $(\mathscr{P}(G), \subseteq)$ and $(\mathscr{P}(M), \subseteq)$ with $\leq \equiv \subseteq$.

Galois connection illustration



M. Liquiere, Mining Graph Data,
John Wiley & Sons (Ed.) (2006) 227-252

The Basic Theorem on Concept Lattices

Theorem

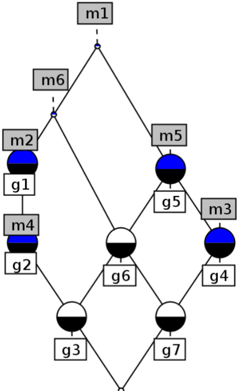
The concept lattice $\underline{\mathfrak{B}}(G, M, I)$ is a complete lattice in which infimum and supremum are given by:

$$\bigwedge_{t\in T}(A_t, B_t) = \left(\bigcap_{t\in T} A_t, \left(\bigcup_{t\in T} B_t\right)''\right)$$

$$\bigvee_{t\in T}(A_t, B_t) = \left(\left(\bigcup_{t\in T} A_t\right)'', \bigcap_{t\in T} B_t\right)$$

Example of formal context and its concept lattice

	m_1	m_2	m_3	m_4	m_5	m_6
g_1	×	×				×
g_2	×	×		×		×
g_3	×	×		×	×	×
g_4	×		×		×	
g_5	×				×	
g_6	×				×	×
g_7	×		×		×	×



Each node is a concept, each a line an order relation between two concepts.

Reduced labeling: the extent of a concept is composed of all objects lying in the extents of its sub-concepts; the intent of a concept is composed of all attributes in the intents of its super-concepts.

The top (resp. bottom) concept is the highest (resp. lowest) w.r.t. $\leq$.

Implications

An implication of a formal context (G, M, I) is denoted by

$$X \rightarrow Y \quad X, Y \subseteq M$$

All objects from G having the attributes in X also have also the attributes in Y, i.e. $X' \subseteq Y'$.

Implications obey the Amstrong rules (reflexivity, augmentation, transitivity). A minimal subset of implications (in sense of its cardinality) from which all implications can be deduced with Amstrong rules is called the Duquenne-Guigues basis.

$$\frac{Y \subseteq X}{X \rightarrow Y} \qquad \frac{X \rightarrow Y}{X \cup Z \rightarrow Y \cup Z} \qquad \frac{X \rightarrow Y, Y \rightarrow Z}{X \rightarrow Z}$$

reflexivity *augmentation* *transitivity*

Outline

2. Complex Data Mining
    - 2.1 Remember... FCA!
    - 2.2 Revisiting FCA for non binary data
    - 2.3 Algorithms
    - 2.4 Other types of patterns

How to handle non binary descriptions

An intersection as a similarity operator

- $\cap$ behaves as *similarity operator*

$$\{m_1, m_2\} \cap \{m_1, m_3\} = \{m_1\}$$

- $\cap$ induces an ordering relation $\subseteq$

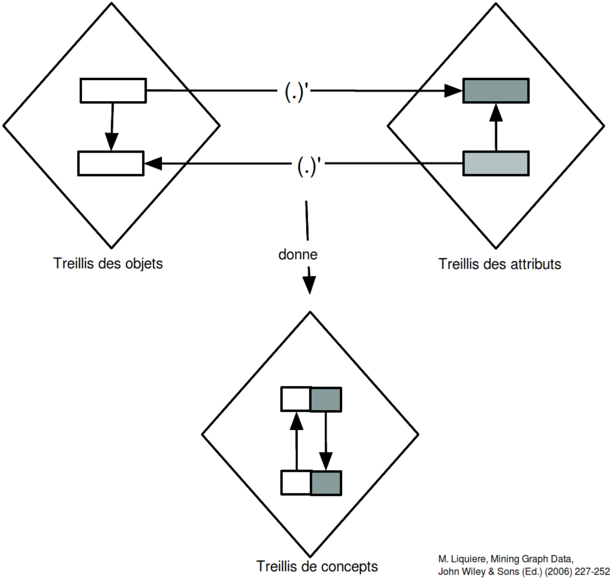
$$N \cap O = N \iff N \subseteq O$$
$$\{m_1\} \cap \{m_1, m_2\} = \{m_1\} \iff \{m_1\} \subseteq \{m_1, m_2\}$$

- $\cap$ has the properties of a meet $\sqcap$ in a semi lattice, a commutative, associative and idempotent operation

$$c \sqcap d = c \iff c \sqsubseteq d$$

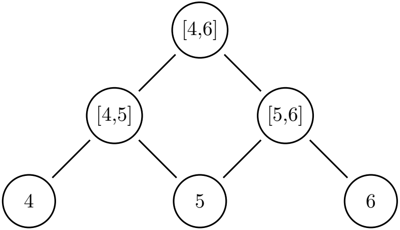
A. Tversky
Features of similarity.
In *Psychological Review, 84 (4), 1977.*

Going a little bit back



M. Liquiere, Mining Graph Data,
John Wiley & Sons (Ed.) (2006) 227-252

Going a little bit back



We can reconstruct the order relation from the lattice operations infimum and supremum by

$$x \leq y \iff x = x \wedge y \iff x \vee y = y$$

Lets do it together.

Pattern structure

Given by $(G, (D, \sqcap), \delta)$

- G a set of *objects*
- $(D, \sqcap)$ a semi-lattice of descriptions or *patterns*
- δ a mapping such as $\delta(g) \in D$ describes object g

A Galois connection

$$A^{\square} = \bigsqcap_{g \in A} \delta(g) \qquad \text{for } A \subseteq G$$

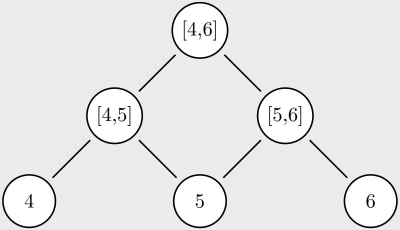
$$d^{\square} = \{g \in G | d \sqsubseteq \delta(g)\} \qquad \text{for } d \in (D, \sqcap)$$

B. Ganter and S. O. Kuznetsov
Pattern Structures and their Projections. In *International Conference on Conceptual Structures*, 2001.

Ordering descriptions in numerical data

$(D, \sqcap)$ as a meet-semi-lattice with $\sqcap$ as a "convexification"

	m_1	m_2	m_3
g_1	5	7	6
g_2	6	8	4
g_3	4	8	5
g_4	4	9	8
g_5	5	8	5



$$
\begin{aligned}
[a_1, b_1] \sqcap [a_2, b_2] &= [min(a_1, a_2), max(b_1, b_2)] \\
[4, 4] \sqcap [5, 5] &= [4, 5]
\end{aligned}
$$

Numerical data are pattern structures

Interval pattern structures

	m_1	m_2	m_3
g_1	5	7	6
g_2	6	8	4
g_3	4	8	5
g_4	4	9	8
g_5	5	8	5

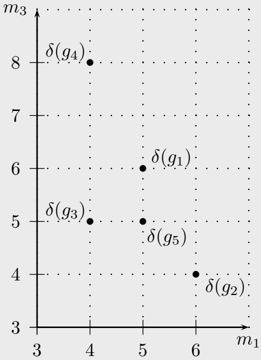
$$\begin{aligned}
\{g_1, g_2\}^{\square} &= \bigsqcap_{g \in \{g_1, g_2\}} \delta(g) \\
&= \langle 5, 7, 6\rangle \sqcap \langle 6, 8, 4\rangle \\
&= \langle [5, 6], [7, 8], [4, 6]\rangle \\
\langle [5, 6], [7, 8], [4, 6]\rangle^{\square} &= \{g \in G | \langle [5, 6], [7, 8], [4, 6]\rangle \sqsubseteq \delta(g)\} \\
&= \{g_1, g_2, g_5\}
\end{aligned}$$

$(\{g_1, g_2, g_5\}, \langle [5, 6], [7, 8], [4, 6]\rangle)$ is a (pattern) concept

n-dimensional intervals

Interval patterns as (hyper) rectangles

	m_1		m_3
g_1	5		6
g_2	6		4
g_3	4		5
g_4	4		8
g_5	5		5

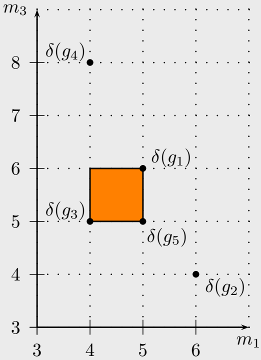


n-dimensional intervals

Interval patterns as (hyper) rectangles

	m_1		m_3
g_1	5		6
g_2	6		4
g_3	4		5
g_4	4		8
g_5	5		5

$$\langle [4,5],[5,6] \rangle^{\square} = \{g_1, g_3, g_5\}$$

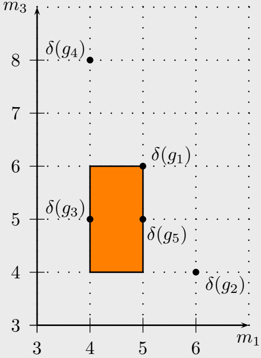


n-dimensional intervals

Interval patterns as (hyper) rectangles

	m_1		m_3
g_1	5		6
g_2	6		4
g_3	4		5
g_4	4		8
g_5	5		5

$\langle [4,5],[5,6] \rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle [4,5],[4,6] \rangle^{\square} = \{g_1, g_3, g_5\}$

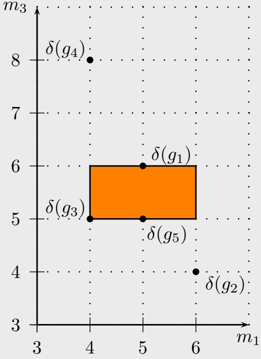


n-dimensional intervals

Interval patterns as (hyper) rectangles

	m_1		m_3
g_1	5		6
g_2	6		4
g_3	4		5
g_4	4		8
g_5	5		5

$\langle [4,5],[5,6] \rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle [4,5],[4,6] \rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle [4,6],[5,6] \rangle^{\square} = \{g_1, g_3, g_5\}$



Interval patterns

Counting all possible interval patterns

$$\langle [a_{m_1}, b_{m_1}], [a_{m_2}, b_{m_2}], \ldots \rangle$$

where $a_{m_i}, b_{m_i} \in W_{m_i}$

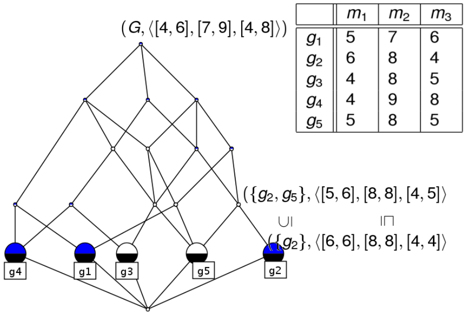
	m_1	m_2	m_3
g_1	5	7	6
g_2	6	8	4
g_3	4	8	5
g_4	4	9	8
g_5	5	8	5

$$\prod_{i \in \{1,\ldots,|M|\}} \frac{|W_{m_i}| \times (|W_{m_i}| + 1)}{2}$$

360 possible interval patterns in our small example

M. Kaytoue, S. O. Kuznetsov, and A. Napoli
Revisiting Numerical Pattern Mining with Formal Concept Analysis.
In International Joint Conference on Artificial Intelligence (IJCAI), 2011.

Interval pattern concept lattice



$(G, \langle [4,6],[7,9],[4,8] \rangle)$

$(\{g_2, g_5\}, \langle [5,6],[8,8],[4,5] \rangle)$

$\rotatebox{90}{$\subseteq$} \qquad \rotatebox{90}{$\sqsubseteq$}$

$(\{g_2\}, \langle [6,6],[8,8],[4,4] \rangle)$

	m_1	m_2	m_3
g_1	5	7	6
g_2	6	8	4
g_3	4	8	5
g_4	4	9	8
g_5	5	8	5

- Existing algorithms
- Lowest concepts: few objects, small intervals
- Highest concepts: many objects, large intervals

Links with conceptual scaling

Interordinal scaling [Ganter & Wille]

- A scale to encode intervals of attribute values, gives rise to equivalent concept lattice

	$m_1 \leq 4$	$m_1 \leq 5$	$m_1 \leq 6$	$m_1 \geq 4$	$m_1 \geq 5$	$m_1 \geq 6$
4	×	×	×	×		
5		×	×	×	×	
6			×	×	×	×

$$(\{g_1, g_2, g_5\}, \quad \{m_1 \leq 6, m_1 \geq 4, m_1 \geq 5, \ \dots, \ \dots \})$$
$$(\{g_1, g_2, g_5\}, \quad \langle [5, 6] \, , \ \dots , \ \dots \rangle)$$

Why pattern structures as we have scaling?

Processing a pattern structure is more efficient

M. Kaytoue, S. O. Kuznetsov, A. Napoli and S. Duplessis
Mining Gene Expression Data with Pattern Structures in Formal Concept Analysis.
In *Information Sciences. Spec. Iss.: Lattices (Elsevier)*, 181(10): 1989-2001 (2011).

A condensed representation

Equivalence classes of interval patterns

Two interval patterns with same image are said to be equivalent

$$c \cong d \iff c^{\Box} = d^{\Box}$$

Equivalence class of a pattern d: $[d] = \{c | c \cong d\}$

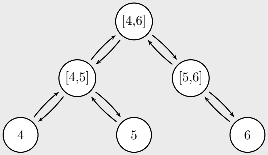
- with a unique closed pattern: the smallest rectangle
- and one or several generators: the largest rectangles

Y. Bastide, R. Taouil, N. Pasquier, G. Stumme, and L. Lakhal.
Mining frequent patterns with counting inference.
SIGKDD Expl., 2(2):66–75, 2000.

In our example: 360 **patterns ;** 18 **closed ;** 44 **generators**

A condensed representation

Remarks



- Compression rate varies between 10^7 and 10^9
- Interordinal scaling: encodes $\simeq$ 30.000 binary patterns
  - not efficient even with best algorithms (e.g. LCMv2)
  - redundancy problem discarding its use for generator extraction

Mehdi Kaytoue, Sergei O. Kuznetsov, Amedeo Napoli:
Revisiting Numerical Pattern Mining with Formal Concept Analysis.
Int. Joint Conference on Artificial Intelligence, IJCAI 2011: 1342-1347

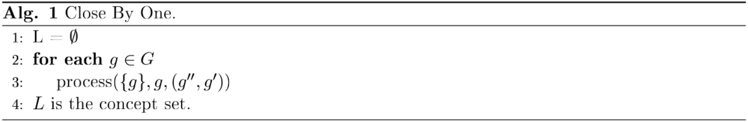
Outline

2. Complex Data Mining
    - 2.1 Remember... FCA!
    - 2.2 Revisiting FCA for non binary data
    - 2.3 Algorithms
    - 2.4 Other types of patterns

Adapt Close By One algorithm

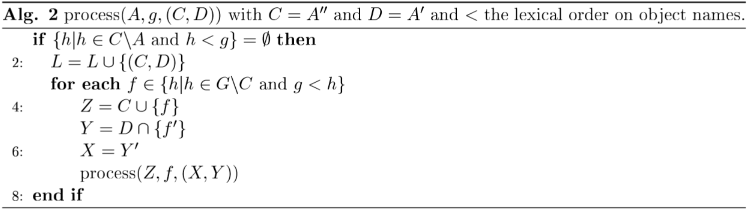
- Bottom-up concepts generation (from min. to max. extents)
- Considers objects one by one starting from the minimal one w.r.t. a linear order < on G (e.g. lexical)
- Given a concept (A, B), the algorithm adds the next object g w.r.t $<$ in A such as $g \notin A$.
- Then it applies the closure operator $(\cdot)''$ for generating the next concept (C, D): intent B is intersected with the description of g, i.e. $D = B \cap g'$, and $C = D'$.
- Induces a tree structure on concepts
- To avoid redundancy, it uses a *canonicity test*: Consider a concept (C, D) obtained from a concept (A, B) by adding object g in A and applying closure. C is said to be canonically generated iff $\{h | h \in C \backslash A \text{ and } h < g\} = \emptyset$, i.e. no object before g has been added in A to obtain C. Backtrack can be ensured.

Closed By One Algorithm



Alg. 1 Close By One.

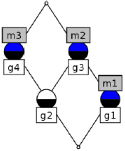
```
L = $\emptyset$
for each $g \in G$
    process($\{g\}, g, (g'', g')$)
$L$ is the concept set.
```



Alg. 2 process$(A, g, (C, D))$ with $C = A''$ and $D = A'$ and $<$ the lexical order on object names.

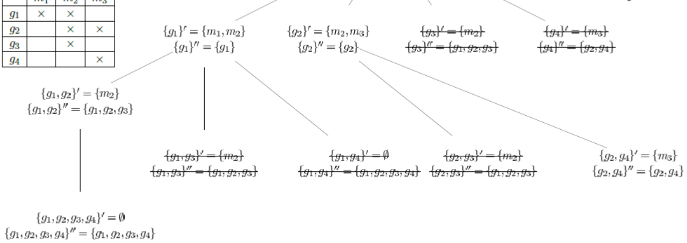
```
if $\{h | h \in C \backslash A$ and $h < g\} = \emptyset$ then
    $L = L \cup \{(C, D)\}$
    for each $f \in \{h | h \in G \backslash C$ and $g < h\}$
        $Z = C \cup \{f\}$
        $Y = D \cap \{f'\}$
        $X = Y'$
        process$(Z, f, (X, Y))$
end if
```

Example



	m_1	m_2	m_3
g_1	×	×	
g_2		×	×
g_3		×	
g_4			×

$\emptyset$



$\{g_1\}' = \{m_1, m_2\}$
$\{g_1\}'' = \{g_1\}$

$\{g_2\}' = \{m_2, m_3\}$
$\{g_2\}'' = \{g_2\}$

~~$\{g_3\}' = \{m_2\}$~~
~~$\{g_3\}'' = \{g_1, g_2, g_3\}$~~

~~$\{g_4\}' = \{m_3\}$~~
~~$\{g_4\}'' = \{g_2, g_4\}$~~

$\{g_1, g_2\}' = \{m_2\}$
$\{g_1, g_2\}'' = \{g_1, g_2, g_3\}$

~~$\{g_1, g_3\}' = \{m_2\}$~~
~~$\{g_1, g_3\}'' = \{g_1, g_2, g_3\}$~~

~~$\{g_1, g_4\}' = \emptyset$~~
~~$\{g_1, g_4\}'' = \{g_1, g_2, g_3, g_4\}$~~

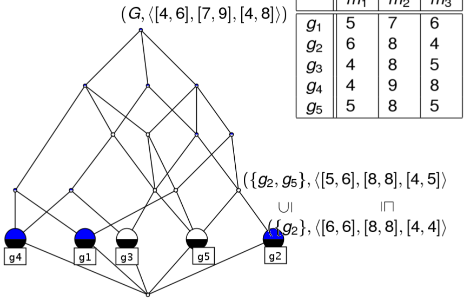
~~$\{g_2, g_3\}' = \{m_2\}$~~
~~$\{g_2, g_3\}'' = \{g_1, g_2, g_3\}$~~

$\{g_2, g_4\}' = \{m_3\}$
$\{g_2, g_4\}'' = \{g_2, g_4\}$

$\{g_1, g_2, g_3, g_4\}' = \emptyset$
$\{g_1, g_2, g_3, g_4\}'' = \{g_1, g_2, g_3, g_4\}$

Now with numerical data: bottom approach

- "Easy", enumerate object sets as we did before: You need $\sqcap$ which is the minimal convex hull and $\sqsubseteq$
- As $c \sqcap d = c \iff c \sqsubseteq d$, for intervals: $c \sqsubseteq d \iff c \supseteq d$.
- e.g., $[4,5] \sqcap [5,5] = [4,5] \iff [4,5] \sqsubseteq [5,5] = [4,5]$



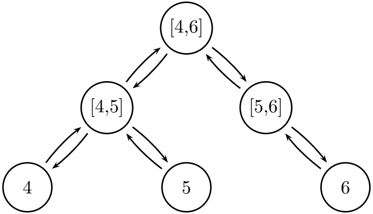
$(G, \langle[4,6],[7,9],[4,8]\rangle)$
$(\{g_2, g_5\}, \langle[5,6],[8,8],[4,5]\rangle)$
$\cup\!|$
$|\!\sqcap$
$(\{g_2\}, \langle[6,6],[8,8],[4,4]\rangle)$
g4
g1
g3
g5
g2

	m_1	m_2	m_3
g_1	5	7	6
g_2	6	8	4
g_3	4	8	5
g_4	4	9	8
g_5	5	8	5

Exercise in class: compute the concepts.

Now with numerical data: top-down approach

A bit trickier but remember:



- We need to define the operation for getting the direct lower neighbor of a pattern w.r.t. $\sqsubseteq$.
- We need a lectic order on these operations and associated canonicity test.
- We need to consider several attributes (but we know how to do that already).

Exercise in class: compute the concepts.

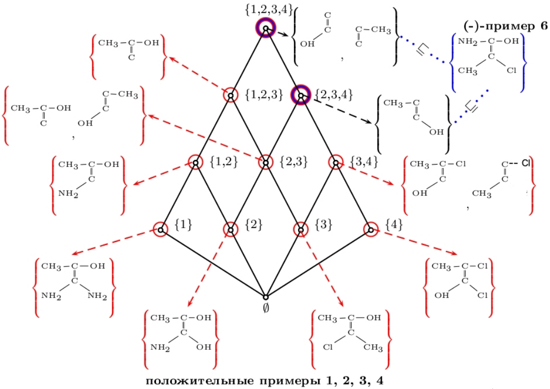
Outline

2. Complex Data Mining
   - 2.1 Remember... FCA!
   - 2.2 Revisiting FCA for non binary data
   - 2.3 Algorithms
   - 2.4 Other types of patterns

Learning closed sets of labeled graphs

$\{X\} \sqcap \{Y\}$ the set of all maximal common subgraphs of graphs X and Y. For non singleton sets of graphs, we have

$$\{X_1, \ldots, X_k\} \sqcap \{Y_1, \ldots, Y_m\} = MAX_{\leq}(\textstyle\bigcup_{i,j}(\{X_i\} \sqcap \{Y_j\}))$$



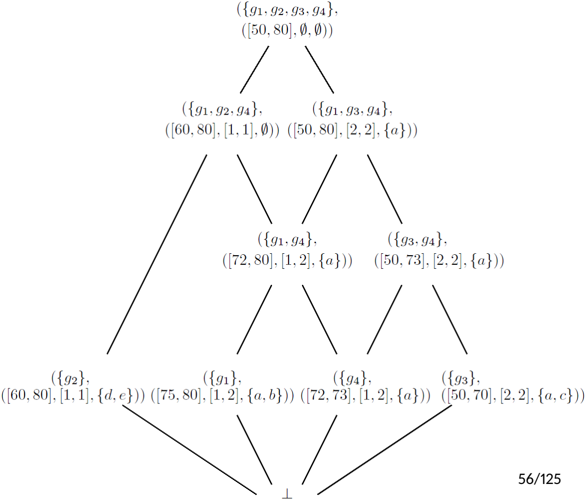
Closed graph mining (Yan & Han, 2003) is covered by this model.

Heterogeneous data

$(D, \sqcap) = (D_{y_1}, \sqcap_{y_1}) \times \ldots \times (D_{y_p}, \sqcap_{y_p})$

	y_1	y_2	y_3
g_1	[75,80]	[1,2]	{a,b}
g_2	[60,80]	[1,1]	{d,e}
g_3	[50,70]	[2,2]	{a,c}
g_4	[72,73]	[1,2]	{a}

M. Alam, A. Buzmakov, V. Codocedo, A. Napoli
Mining Definitions from RDF Annotations Using Formal Concept Analysis.
In International Joint Conference on Artificial Intelligence, IJCAI 2015.



$(\{g_1, g_2, g_3, g_4\}, ([50, 80], \emptyset, \emptyset))$

$(\{g_1, g_2, g_4\}, ([60, 80], [1, 1], \emptyset))$

$(\{g_1, g_3, g_4\}, ([50, 80], [2, 2], \{a\}))$

$(\{g_1, g_4\}, ([72, 80], [1, 2], \{a\}))$

$(\{g_3, g_4\}, ([50, 73], [2, 2], \{a\}))$

$(\{g_2\}, ([60, 80], [1, 1], \{d, e\}))$

$(\{g_1\}, ([75, 80], [1, 2], \{a, b\}))$

$(\{g_4\}, ([72, 73], [1, 2], \{a\}))$

$(\{g_3\}, ([50, 70], [2, 2], \{a, c\}))$

$\perp$

Functional Dependencies

Definition

$X \longrightarrow Y$, holds in T if:

$$\forall t_i, t_j \in T : t_i(X) = t_j(X) \Rightarrow t_i(Y) = t_j(Y)$$

Derive a formal context in which

$X \longrightarrow Y$ holds iif $X'' = XY''$

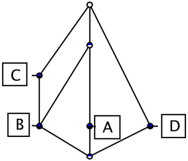
B. Ganter and R. Wille
Formal Concept Analysis.
Springer, Mathematical foundations., 1999.

Example

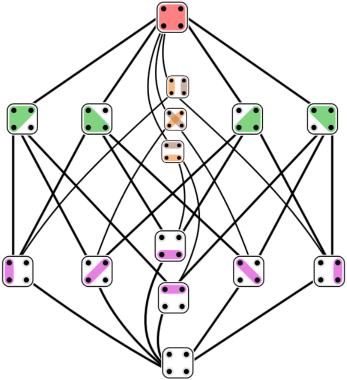
- $C \longrightarrow B$ holds
- $B \longrightarrow C$ does not hold

id	A	B	C	D
t_1	1	3	7	2
t_2	1	3	4	5
t_3	3	5	2	2
t_4	3	3	4	8

$\mathbb{K}$	A	B	C	D
(1,2)	×	×		
(1,3)				×
(1,4)		×		
(2,3)				
(2,4)		×	×	
(3,4)	×			



Hasse diagram of the partition lattice



$$\{\{a,b\},\{c\},\{d\}\} \leq \{\{a,b,c\},\{d\}\}$$

$$\{\{a,b\},\{c\},\{d\}\} \vee \{\{a,c\},\{b\},\{d\}\} = \{\{a,b,c\},\{d\}\}$$

$$\{\{a,b,c\},\{d\}\} \wedge \{\{a,b,d\},\{c\}\} = \{\{a,b\},\{c\},\{d\}\}$$

Characterizing FD with partition pattern structures

Partition pattern structures

id	a	b	c	d
t_1	1	3	4	1
t_2	4	3	4	3
t_3	1	8	4	1
t_4	4	3	7	3

m	$\delta(m) \in (D, \sqcap)$
a	$\{\{t_1, t_3\}, \{t_2, t_4\}\}$
b	$\{\{t_1, t_2, t_4\}, \{t_3\}\}$
c	$\{\{t_1, t_2, t_3\}, \{t_4\}\}$
d	$\{\{t_1, t_3\}, \{t_2, t_4\}\}$

Concept lattice and pattern concept lattice are isomorphic

$X \longrightarrow Y$ **holds iif** $X^{\square\square} = XY^{\square\square}$

J Baixeries, M Kaytoue, A Napoli.
Characterizing functional dependencies in formal concept analysis with pattern structures.
Ann. Math. Artif. Intell. 72(1-2): 129-149, 2014

V. Codocedo and A. Napoli
Lattice-based biclustering using partition pattern structures.
European Conference on Artificial Intelligence (ECAI 2014)

Characterizing SD with partition pattern structures

Un-crisping functional dependencies with a similarity relation

- Two values match when they are close enough

S. Song and C. Lei.
Efficient discovery of similarity constraints for matching dependencies.
Data & Knowledge Engineering, 87, 2013.

L. Caruccio, V. Deufemia and G. Polese.
Relaxed Functional Dependencies - A Survey of Approaches
IEEE Transactions on Knowledge and Data Engineering (TKDE 2015)

Softening partitions with tolerance relations

- Parts of the partition becomes maximal sets of of objects with pairwise similar values
- We still have that $X \rightarrow Y$ holds iif $X^{\square\square} = XY^{\square\square}$ holds

J. Baixeries, M. Kaytoue, and A. Napoli
Computing Similarity Dependencies with Pattern Structures (CLA 2013).

Outline

3. Diverse Pattern Set Discovery
   - 3.1 Problem settings
   - 3.2 Exhaustive Search
   - 3.3 Heuristic Search with Beam-search
   - 3.4 Monte Carlo Tree Search
   - 3.5 Pattern mining with MCTS
   - 3.6 A few experiments

Outline

3. Diverse Pattern Set Discovery
    - 3.1 Problem settings
    - 3.2 Exhaustive Search
    - 3.3 Heuristic Search with Beam-search
    - 3.4 Monte Carlo Tree Search
    - 3.5 Pattern mining with MCTS
    - 3.6 A few experiments

The search space

Most of the SD/EMM algorithms exploit the lattice of subgroups

Definition (Subgroup search space)

- The set of all descriptions is partially ordered and is structured as a lattice. (*Question: can we use closed subgroups?*)
- $s_1 \prec s_2$ and say that the subgroup s_1 is more specific than the subgroup s_2 if the description of s_1 is more specific than the one of s_2 w.r.t. the partial order (s_2 is more general than s_1).

Example

$\langle [23 \le b \le 29] \rangle$ is more general than

$\langle [128.24 \le a \le 151.28], [23 \le b \le 29] \rangle$

Goal: Find the top-k patterns maximizing a quality measure φ

The need of heuristic search

Exhaustive search works in practice when

- For simple pattern languages
- Quality measures that allows pruning/upper bounds
- Search space of reasonable size

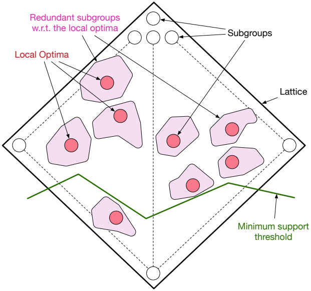
With numerical attributes, this is even more problematic

- Convex: no pattern with empty support!
- Discretization are (almost) always used but it comes with loss of information, which may not be acceptable when dealing with, e.g. spatial attributes describing molecules
- Discretization is always a difficult choice and impacts interpretation

Heuristic search becomes mandatory

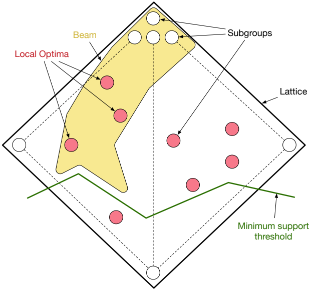
The redundancy problem

- The quality measure of a subgroup close to a local optimum s^* in the lattice is similar to – but lower than – the quality measure of s^*: The slight change in the description of a subgroup s close to s^* induces a slight change of the support of s compared to those of s^*.
- It is desirable to avoid extracting the redundant subgroups close to a local optimum: This is the *redundancy problem*.



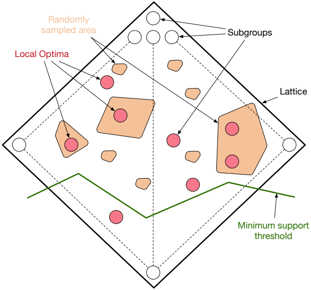
The completeness problem

- What are the guarantees of finding the top-k patterns maximizing a quality measure φ?
- A greedy approach will certainly miss some of them
- Many techniques
  - hill climbing from the top
  - hill climbing with random seeds with restarts
  - beam search
  - genetic algorithms
  - Diversity: **all optima should be present in the pattern set result**.



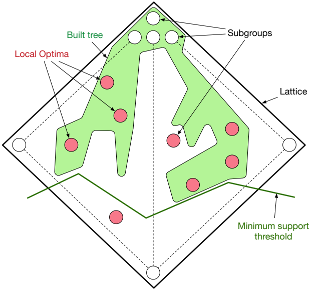
The diversity problem

- What are the guarantees of finding the top-k patters maximizing the quality measures φ?
- A greedy approach will certainly miss some of them,
- Many techniques
  - sampling (see Pattern Mining: Part 2): based on a probability distribution over the subgroup space that gives more chance to an interesting subgroup to be drawn.
- Diversity: **all optima should be present in the pattern set result**.



The diversity problem

- What are the guarantees of finding the top-k patterns maximizing the quality measure φ?
- A greedy approach will certainly miss some of them,
- Many techniques
  - Monte Carlo Tree Search: use a lot of random searches with "memory" **no a priori required**
- Diversity: **all optima should be present in the pattern set result**.



The DSSD Problem

Problem (Diverse Subgroup Set Discovery)

*Given $\mathscr{D}(\mathscr{O}, \mathscr{A}, C, class)$, a quality measure φ, a minimum support threshold minSupp, an integer k, extract a set of the top-k best patterns w.r.t. φ that has as little **redundancy** as possible and the highest number of local optima.*

Matthijs van Leeuwen, Arno J. Knobbe:
Diverse subgroup set discovery.
Data Min. Knowl. Discov., 25(2): 208-242 (2012)

Outline

3. Diverse Pattern Set Discovery
   - 3.1 Problem settings
   - 3.2 Exhaustive Search
   - 3.3 Heuristic Search with Beam-search
   - 3.4 Monte Carlo Tree Search
   - 3.5 Pattern mining with MCTS
   - 3.6 A few experiments

Exhaustive search

- Search space of subgroups: lattice of all possible descriptions $(D, \sqsubseteq)$ where $d_1 \sqsubseteq d_2$ means that subgroup d_1 is more general than d_2, or equivalently $supp(d_2) \subseteq supp(d_1)$.
- This lattice can be explored either in a depth-first (DFS) or in a breadth-first (BFS) search manner.
- During the traversal, the quality measure is computed for each subgroup.
- In the end, a redundancy filter is applied to output the top-k diverse subgroups. (discussed later)

Exhaustive search

- Mining closed patterns when the quality measure is maximized by them: adapt CloseByOne
  - Define $\sqcap$ and $\sqsubset$
  - Define the operation the gives the next more general patterns after a pattern (the neighboors) and a lexicographic order on them.
  - That's it!
- Safe pruning: see the SD-Map* algorithm
  - monotone constraints
  - upper bounds

Outline

3. Diverse Pattern Set Discovery
    - 3.1 Problem settings
    - 3.2 Exhaustive Search
    - 3.3 Heuristic Search with Beam-search
    - 3.4 Monte Carlo Tree Search
    - 3.5 Pattern mining with MCTS
    - 3.6 A few experiments

Beam search

- Level wise exploration with fixed size width
- Can be understood as a set of parallel hill climbing search (ensures fast termination)

The subgroup search space is explored level-wise (BFS) and each level is restricted to a set of diversified high quality patterns. The diversification is done as follows. Subgroups are sorted according to their quality: The best is picked and all the next patterns that are too similar (bounded Jaccard coefficient between their support) are removed. The first of the next patterns that is not similar is kept, and the process is reiterated.

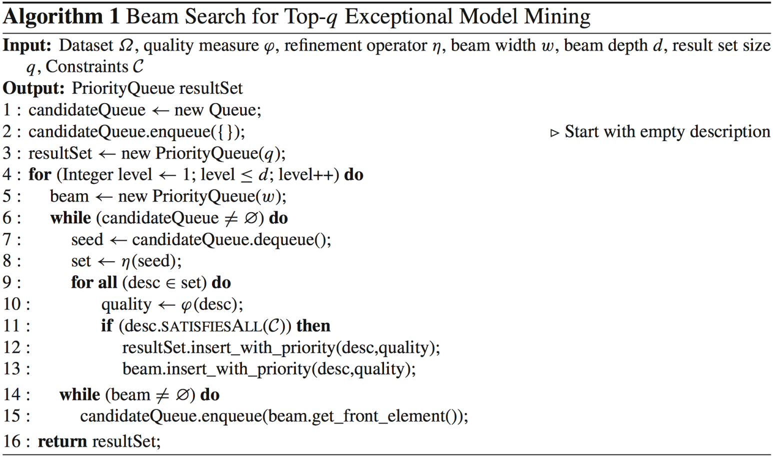
Used in most of the research papers and platforms!

Beam search

Consider a single binary target.

- Starts from the most general subgroup
- Generates next levels by specializing subgroups by restricting an attribute as long as the quality measure is improved
- Choose among those only a constant number of candidates to continue the exploration (the beam width: The *beamWidth* best subgroups w.r.t. the quality measure).
- How many possible actions? *n* for itemsets, 2*n* for *n* numerical attributes, ... what about sequence and graphs?

A Beam search pseudo code

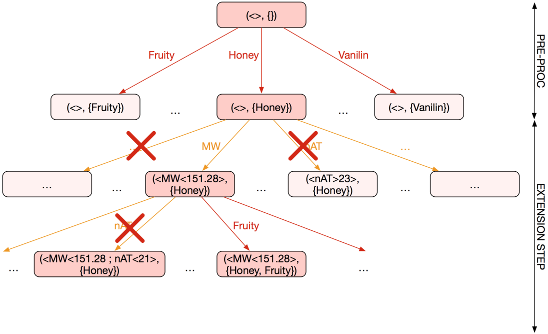


Algorithm 1 Beam Search for Top-q Exceptional Model Mining

```
Input: Dataset $\Omega$, quality measure $\varphi$, refinement operator $\eta$, beam width $w$, beam depth $d$, result set size $q$, Constraints $\mathcal{C}$
Output: PriorityQueue resultSet
candidateQueue ← new Queue;
candidateQueue.enqueue({ });                                   ▷ Start with empty description
resultSet ← new PriorityQueue($q$);
for (Integer level ← 1; level $\leq d$; level++) do
    beam ← new PriorityQueue($w$);
    while (candidateQueue $\neq \emptyset$) do
        seed ← candidateQueue.dequeue();
        set ← $\eta$(seed);
        for all (desc $\in$ set) do
            quality ← $\varphi$(desc);
            if (desc.SATISFIESALL($\mathcal{C}$)) then
                resultSet.insert_with_priority(desc,quality);
                beam.insert_with_priority(desc,quality);
    while (beam $\neq \emptyset$) do
        candidateQueue.enqueue(beam.get_front_element());
return resultSet;
```

In presence of multi label data

- Binary relevance widely used
- Label powerset to keep label correlations but too numerous!
- Jointly explore subgroups and label subset: search for bi-sets



G Bosc, J Golebiowski, M Bensafi, C Robardet, M Plantevit, J-F Boulicaut, M Kaytoue:
Local Subgroup Discovery for Eliciting and Understanding New Structure-Odor Relationships.
Discovery Science, 2016: 19-34

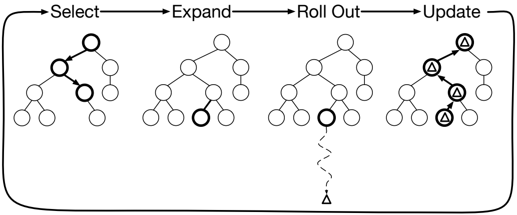
Outline

3. Diverse Pattern Set Discovery
   - 3.1 Problem settings
   - 3.2 Exhaustive Search
   - 3.3 Heuristic Search with Beam-search
   - 3.4 Monte Carlo Tree Search
   - 3.5 Pattern mining with MCTS
   - 3.6 A few experiments

Monte Carlo Tree Search (MCTS)

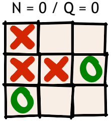
MCTS is an exploration method, initially designed for Artificial Intelligence, that builds iteratively the search tree according to random simulations. The strengths of MCTS are :

- The power of random simulations
- The trade-off between exploration and exploitation of an interesting solution



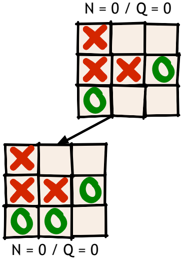
C. Browne, E. J. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. P. Liebana, S. Samothrakis, and S. Colton
A survey of monte carlo tree search methods.
In *IEEE Trans. Comput. Intellig. and AI in Games*, 2012.

Introductory example



X		
X	X	O
O		

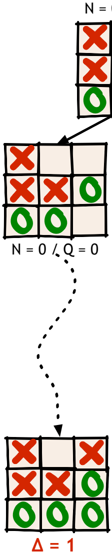
Introductory example



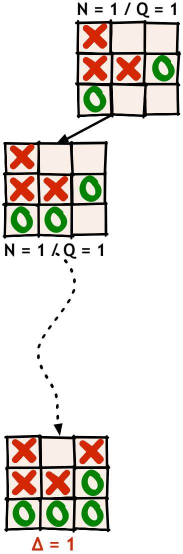
N = 0 / Q = 0

N = 0 / Q = 0

Introductory example



Introductory example

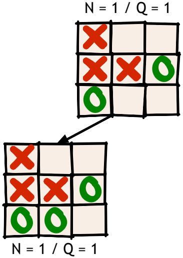


N = 1 / Q = 1

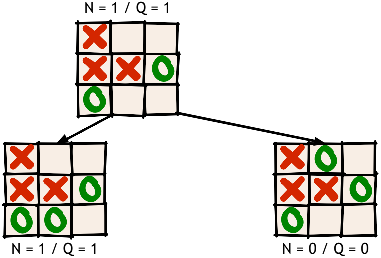
N = 1 / Q = 1

Δ = 1

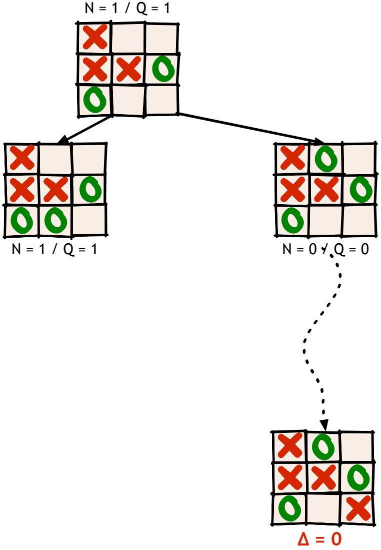
Introductory example



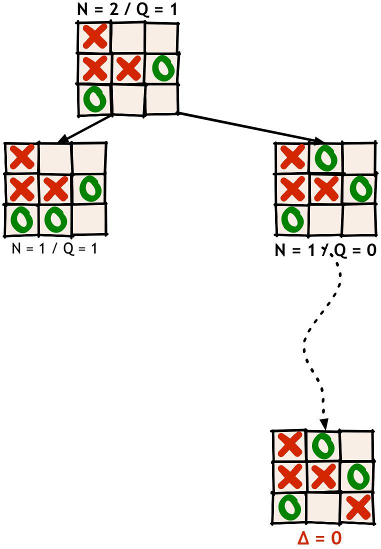
Introductory example



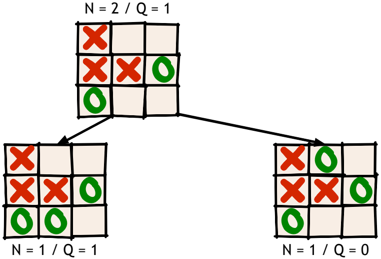
Introductory example



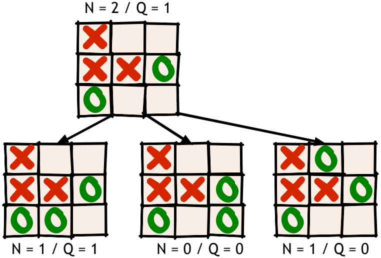
Introductory example



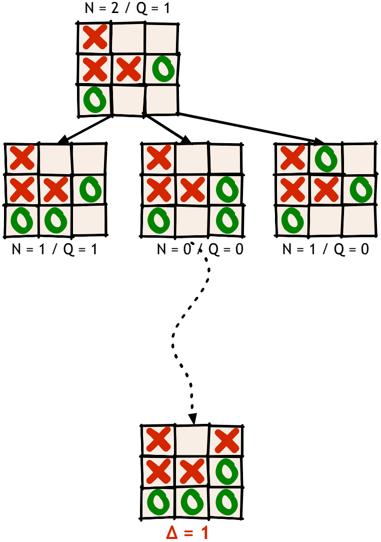
Introductory example



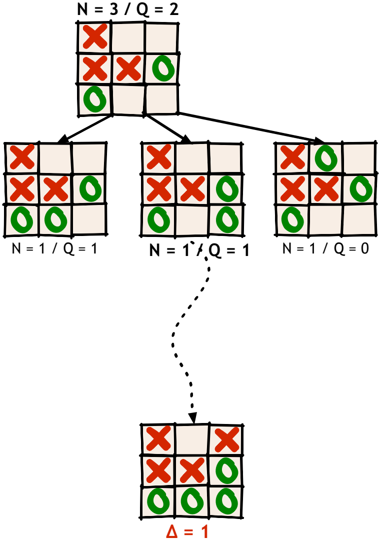
Introductory example



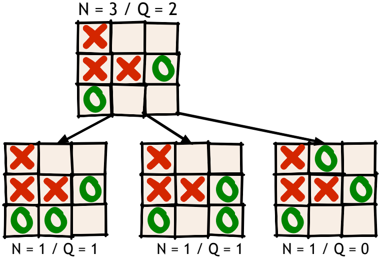
Introductory example



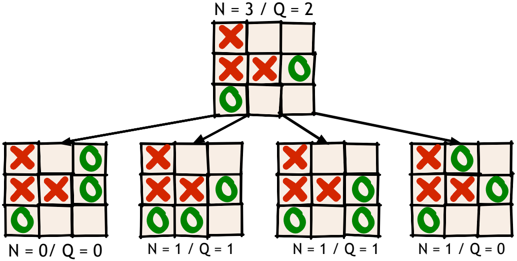
Introductory example



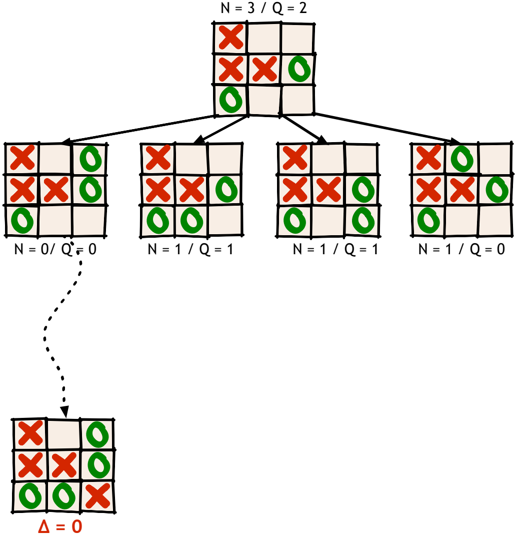
Introductory example



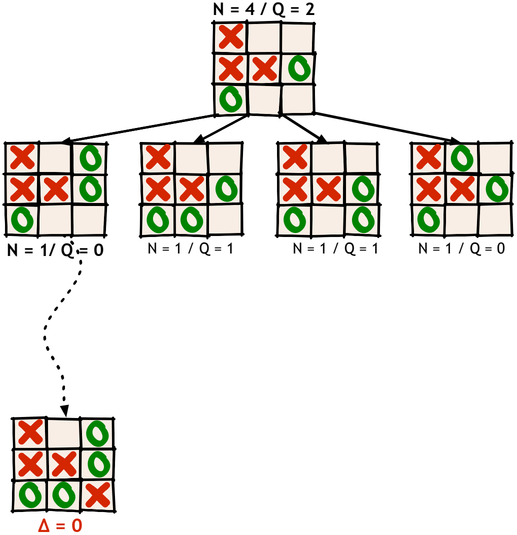
Introductory example



Introductory example

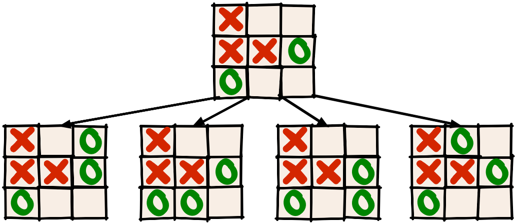


Introductory example



Introductory example

N = 4 / Q = 2



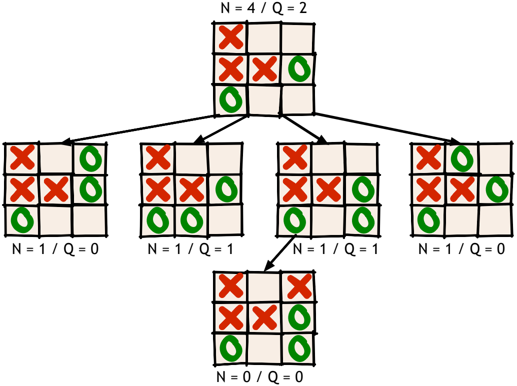
N = 1 / Q = 0

N = 1 / Q = 1

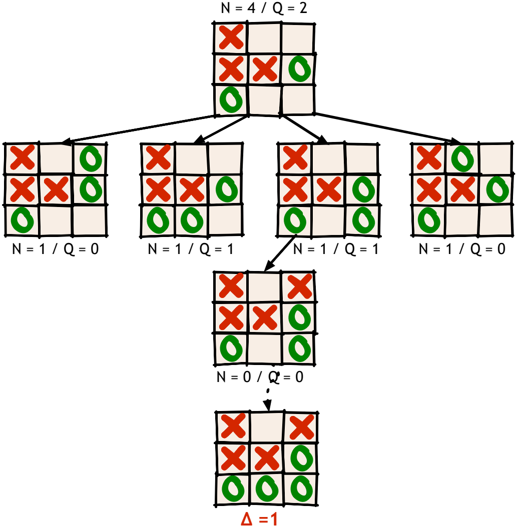
N = 1 / Q = 1

N = 1 / Q = 0

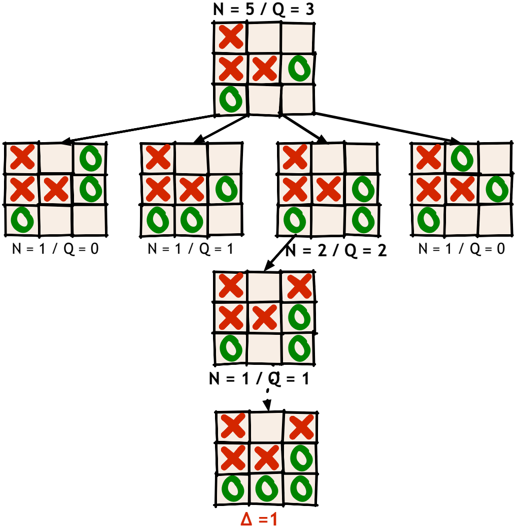
Introductory example



Introductory example

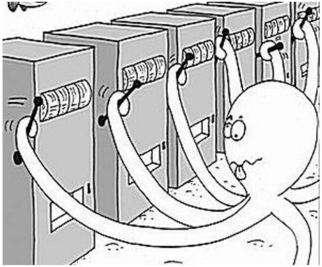


Introductory example



Monte Carlo Tree Search

- Making optimal decisions in artificial intelligence (AI) problems, typically move planning in combinatorial games
- Min-Max cannot work when there is no good heuristic function
- Multi armed bandit problem: pull one arm at each turn in order to find the best arm, minimize regret, maximize expected return
- MCTS combines the generality of random simulation with the precision of tree search



Exploration/exploitation trade-off: e.g. shall I try another restaurant or stick to those I know excellent?

MCYS: Key advance enabling a program to win against a pro Go player! ALPHA GO (Nature,

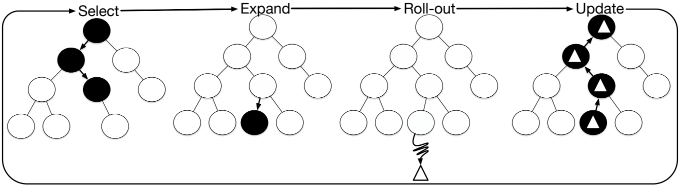
Monte Carlo Tree Search

Games

- Find the best action to play given a current game state
- Build a partial game tree depending on results of previous iterations until max. budget is reached, 4 steps per iter.
  - Selection of a node (depending on the exploration/exploitation trade-off due to the past iterations)
  - Creation of a new node from the selection one
  - Simulation: sequence of actions to a terminal node
  - Update/Backpropagation: Any node s is provided with two values: The number $N(s)$ of times it has been visited, and a value $Q(s)$ that corresponds to the aggregation of rewards of all simulations passed through s so far (mean).
- The aggregated reward of each node is updated through the iterations and becomes more and more accurate.
- When computation budget reached: return the optimal move that leads to the child of the root node with best $Q(.)$.

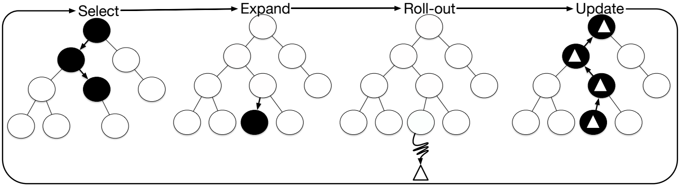
Monte Carlo Tree Search

- Recursively **selects** from the root an action until either a terminal (win/loss/draw) or fully-expanded (no actions) node.
- Selection base on the exploration/exploitation trade-off given by the an Upper Confidence Bound (UCB) which estimate the regret of choosing a non-optimal child.
- Many variants of UCB, e.g. UCT and one of its variant, namely the UCT: $UCT(s, s') = Q(s') + 2C_p\sqrt{\frac{2\ln N(s)}{N(s')}}$ where s' is a child of a node s and $C_p > 0$ is a constant.
- First term: exploitation; second term: exploration



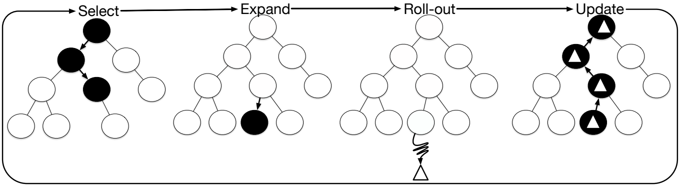
Monte Carlo Tree Search

Expand A new child, denoted s_{exp}, of the selected node s_{sel} is added to the tree according to the available actions. The child s_{exp} is randomly picked among all available children of s_{sel} not yet expanded in the search tree.



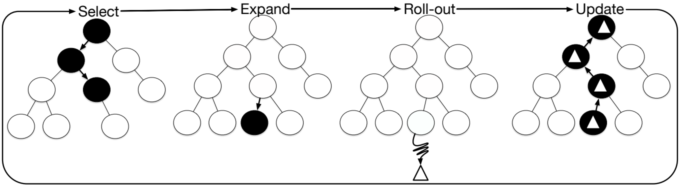
Monte Carlo Tree Search

RollOut From this expanded node s_{exp}, a simulation is played based on a specific policy. This simulation consists of exploring the search space (playing a series of actions) from s_{exp} until a terminal state is reached. It returns the reward Δ of this terminal state: $\Delta = 1$ if the terminal state is a win, $\Delta = 0$ otherwise.



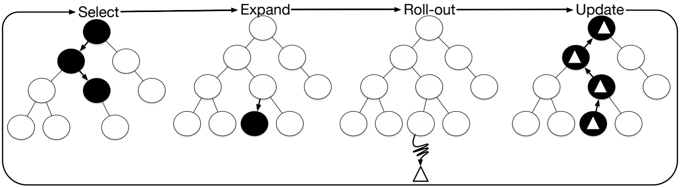
Monte Carlo Tree Search

Update The reward Δ is back-propagated to the root, updating for each parent the number of visits $N(.)$ (incremented by 1) and the aggregation reward $Q(.)$ (the new win rate).



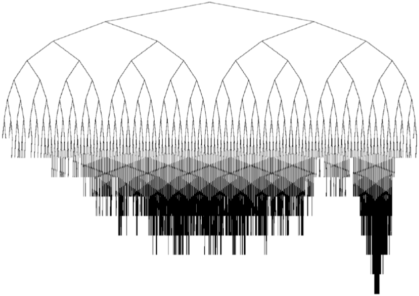
Monte Carlo Tree Search

Next Select will select the most urgent node to be expanded according to the UCT that will consider new values of $N(.)$ and $Q(.)$ recently back-propagated.



Monte Carlo Tree Search

Example of tree during a search



Outline

3. Diverse Pattern Set Discovery
    - 3.1 Problem settings
    - 3.2 Exhaustive Search
    - 3.3 Heuristic Search with Beam-search
    - 3.4 Monte Carlo Tree Search
    - 3.5 Pattern mining with MCTS
    - 3.6 A few experiments

General idea

Problem (EMM when descriptions are itemsets)

Let $\mathscr{I}$ be a set of items. A transaction is a subset of items $t \subseteq \mathscr{I}$. A transaction database is a set of transactions $\mathscr{T} = \{t_1, \ldots, t_n\}$. An itemset is an arbitrary subset of items $P \subseteq \mathscr{I}$. Its support is given by $supp(P) = \{t \in \mathscr{T} | P \subseteq t\}$. Its evaluation measure $\varphi(P)$ depends on the EMM instance that is considered. The problem is to find the best itemsets w.r.t φ.

It follows that the search space is given by $\mathscr{S} = (2^{\mathscr{I}}, \subseteq)$. The initial pattern is the empty set: $s_0 = \emptyset$. The actions that lead to specializations, or supersets, are the items $\mathscr{I}$. A simulation is a random sequence of items additions.

EMM as a *single-turn single-player game*

Let $\mathscr{S}$ be the set of all possible patterns ordered with a specialization/generalization rel. , a poset $(\mathscr{S}, \prec)$, generally a lattice.

- Let $\mathscr{S}$ be the set of game states, or patterns, with support $supp(s)$ and quality measure $\varphi(s)$. The initial game state $s_0 \in \mathscr{S}$, root of tree, is the most general pattern.
- The actions for generating new game states are defined as pattern restrictions (for deriving pattern refinements).
- A simulation is a random sequence of actions, or pattern restrictions. A leaf is a maximal frequent pattern.

The goal is not to decide, at each turn, what is the best action to play, but to explore the search space of patterns with the benefit of the exploitation/exploration trade-off and the memory of the tree.

MCTS for EMM: Select

Goal: select the most urgent node w.r.t. exploration vs. exploitation.

- Any UCB, but empirically, the best is the *Single-Player MCTS* adds a third term to the UCB to take into account the variance σ^2 of the rewards obtained by the child so far. SP-MCTS of a child s' of a node s is:
  $$SP\text{-}MCTS(s, s') = Q(s') + C\sqrt{\frac{2 \ln N(s)}{N(s')}} + \sqrt{\sigma^2(s') + \frac{D}{N(s')}}$$
  where the constant $C = 0.5$ is used to weight the exploration term and the term $\frac{D}{N(s')}$ inflates the standard deviation for infrequently visited children (D is a constant).
- The reward of a node rarely visited is considered as less certain: It is still required to explore it to get a more precise σ estimate
- If the variance is still high, it means that the subspace from this node is not homogeneous: more exploration is needed.
- Pattern evaluation measures φ can be normalized (UCT).

MCTS for EMM: expand

The simple way to expand the selected node s_{sel} is to choose uniformly an item not yet used in s_{sel}, that is to specialize s_{sel} into s_{exp} such that $s_{exp} \prec s_{sel}$: s_{exp} is a refinement of s_{sel}.
but... a lot of redundant nodes!

1. a pattern s can be expanded into a node s' with the same support, thus, the same quality measure (for most of the quality measures)
2. a pattern s may appear in different branches of the enumeration tree.

MCTS for EMM: expand with generators

Definition (Closed descriptions and their generators)

The equivalence class of an pattern s is given by $[s] = \{s' | supp(s) = supp(s')\}$. Each equivalence class has a unique smallest element w.r.t. $\prec$ that is called the closed pattern: s is said to be closed iff $\nexists s'$ such that $s' \prec s$ and $supp(s) = supp(s')$. The (minimal) non-closed patterns are called (minimal) generators.

Avoiding duplicates in a tree branch. A specialization is uniformly picked: If its support does change from the parent, it is used as an expansion. Otherwise, the specialization is considered invalid and another one is picked. Repeat this process until a valid expansion is found. If there are no more valid specializations, then label node as *fully expanded* and start a new iteration.

Removing duplicates and correcting bias

A pattern can be generated in nodes in different branches of the Monte Carlo tree, as the search space is a lattice: For example, with $\mathscr{I} = \{a, b, c\}$, all permutations of the sequence $\langle a, b, c \rangle$ could be generated.

Thus, a part of the search space is sampled several times in different branches of the tree. However, the visit count $N(s)$ of a node s will not count visits of other nodes that depict exactly the same pattern: The UCB is biased!

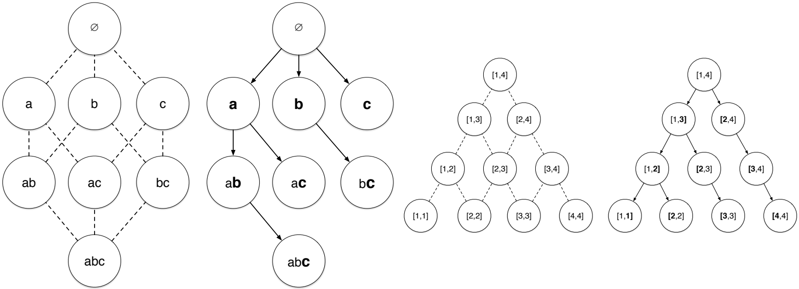
Removing duplicates and correcting bias

Solutions for Avoiding duplicates in the search tree.

- *Lectic order*: Setting an enumeration technique that generates each pattern once and only once is trivial in pattern mining (setting a total order on the set of actions, e.g. lexicographic for itemsets). This restricts the set of available actions at each node. However, it biases the search as some actions are discarded. The UCB should correct this bias.
- *Permutation AMAF* is a solution that allows to keep a unique node for all duplicates of a pattern. This node no longer has a single parent but a list of each duplicates' parent. This list will be used when back-propagating a reward. A hash-map is used to store all the unique patterns encountered so far in the search tree and pointers towards duplicates are set.

Removing duplicates and correcting bias

The problem with a lectic ordering: patterns on the left hand side of the tree have less chances to be generated, e.g., $prob(\{a, b\}) = 1/6$ while $prob(\{b, c\}) = 1/3$.



Removing duplicates and correcting bias

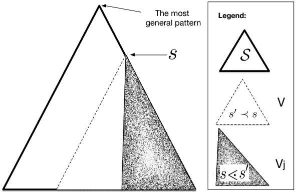
The *DFS-UCT* of a child s_j of s

$$DFS\text{-}UCT(s) = Q(s) + 2C_p\sqrt{\frac{2 \cdot \ln[N(s) \cdot \rho_{norm}(s)]}{N(s_j) \cdot \rho_{norm}(s_j)}}$$

with

$$\rho_{norm}(s) = \frac{V}{V_j} = \frac{|\{s' | s' \prec s \in \mathscr{S}\}|}{|\{s' | s \lessdot s' \wedge s' \prec s \in \mathscr{S}\}|}$$

- Weight the number of visits of a node
- higher weight: smaller proportion of the specialization to explore w.r.t. lectic order $\lessdot$



MCTS for EMM: roll-out

From the expanded node s_{exp} a simulation is run (roll-out).

- With standard MCTS, a simulation is a random sequence of actions that leads to a terminal node: A game state from which a reward can be computed.
- But: any pattern encountered during the simulation could be evaluated
- Define the notion of path (the simulation) and reward computation (which nodes are evaluated and how these different rewards are aggregated) separately.

MCTS for EMM: roll-out

Definition (Path Policy)

Let s_1 the node from which a simulation has to be run (i.e., $s_1 = s_{exp}$). Let $n \geq 1 \in \mathbb{N}$, we define a path $p(s_1, s_n) = \{s_1, \ldots, s_n\}$ as an ordered list of patterns starting from s_1 and ending with s_n such that: $\forall i \in \{1, \ldots, n-1\}$, s_{i+1} is a direct refined pattern of s_i. We denote $\mathscr{P}(s_1, s_n)$ the set of all possible paths from s_1 to s_n.

- *naive-roll-out*: A path length n is randomly picked in $(1, \ldots,$ *pathLength*$)$ where *pathLength* is given by the user (*pathLength* $= |\mathscr{I}|$ by default) using the direct refinement operator. **Fast but fail at finding frequent patterns (or simply with non empty support!)**

MCTS for EMM: roll-out

Definition (Path Policy)

Let s_1 the node from which a simulation has to be run (i.e., $s_1 = s_{exp}$). Let $n \geq 1 \in \mathbb{N}$, we define a path $p(s_1, s_n) = \{s_1, \ldots, s_n\}$ as an ordered list of patterns starting from s_1 and ending with s_n such that: $\forall i \in \{1, \ldots, n-1\}$, s_{i+1} is a direct refined pattern of s_i. We denote $\mathscr{P}(s_1, s_n)$ the set of all possible paths from s_1 to s_n.

- *direct-freq-roll-out*: The path is extended with a randomly chosen restriction until it meets an infrequent pattern s_{n+1} using the direct refinement operator. s_n is a leaf of the tree in our settings. **Slower, but ensures frequent patterns**

MCTS for EMM: roll-out

Definition (Path Policy)

Let s_1 the node from which a simulation has to be run (i.e., $s_1 = s_{exp}$). Let $n \geq 1 \in \mathbb{N}$, we define a path $p(s_1, s_n) = \{s_1, \ldots, s_n\}$ as an ordered list of patterns starting from s_1 and ending with s_n such that: $\forall i \in \{1, \ldots, n-1\}, s_{i+1}$ is a direct refined pattern of s_i. We denote $\mathscr{P}(s_1, s_n)$ the set of all possible paths from s_1 to s_n.

- *large-freq-roll-out* overrides the *direct-freq-roll-out* by using non direct specializations: Several actions are added instead of one to create a new element of the path. The number of added actions is randomly picked in $(1, \ldots, jumpLength)$ (user given). **Quite fast and allows to go deeper in the search space; good trade-off especially for numeric with large domains**

MCTS for EMM: roll-out

Definition (Reward Aggregation Policy)

Let s_1 the node from which a simulation has been run and $p(s_1, s_n)$ the associated random path. Let $\mathcal{E} \subseteq p(s_1, s_n)$ be the subset of nodes to be evaluated. The aggregated reward of the simulation is given by: $\Delta = aggr(\{\varphi(q) \forall q \in \mathcal{E}\}) \in [0; 1]$

- *terminal-reward*: $\mathcal{E} = \{s_n\}$ and *aggr* is the identity function.
- *random-reward*: $\mathcal{E} = \{s_i\}$ with a random $1 \leq i \leq n$ and *aggr* the identity function.
- *max-reward*: $\mathcal{E} = p(s_1, s_n)$ and *aggr* is the *max*$(.)$ function
- *mean-reward*: $\mathcal{E} = p(s_1, s_n)$ and *aggr* is the *mean*$(.)$ function.
- *top-k-mean-reward*: $\mathcal{E} = top\text{-}k(p(s_1, s_n))$, *aggr* is the *mean*$(.)$ function and *top-k(X)* returns the k elements with the highest φ.

For some path policies, node supports are computed, “free” φ

MCTS for EMM: roll-out

A basic MCTS forgets any state encountered during a simulation. A pattern with a high φ should not be forgotten as we might not expand the tree enough to reach it. Add a memory policy!

Definition (Roll-out Memory Policy)

A roll-out memory policy specifies which of the nodes of the path $p = (s_1, s_n)$ shall be kept in an auxiliary data structure M.

- *no-memory*: Any pattern in $\mathcal{E}$ is forgotten **The best pattern may be forgotten!**
- *all-memory*: All evaluated patterns in $\mathcal{E}$ are kept **Too costly**
- *top-k-memory*: A list M stores the best k patterns in $\mathcal{E}$ w.r.t. $\varphi(.)$ **trade-off**

MCTS for EMM: update

A back-propagation policy updates the tree according to a simulation. Let s_{sel} be the selected node and s_{exp} its expansion from which the simulation is run: The policy updates the estimation $Q(.)$ and the number of visits $N(.)$ of each parent of s_{exp} recursively. The number of visits is always incremented by one but for $Q(.)$:

- *mean-update*: $Q(.)$ is the average of the rewards Δ back-propagated through the node so far (basic MCTS).
- *max-update*: $Q(.)$ is the maximum reward Δ back-propagated through the node so far. This strategy allows to identify a local optimum within a part of the search space that contains most of uninteresting patterns.
- *top-k-mean-update*: $Q(.)$ average of the k best rewards Δ back-propagated through the node so far. It favors parts of the search space containing several local optima.

MCTS for EMM: Budget exceeded!

Goal: Pick the *k*-best diverse and non-redundant subgroups within a huge pool of nodes: the tree and the auxiliary memory.

- Let $\mathscr{P} = T \cup M$ be a pool of patterns, where T is the set of patterns stored in the nodes of the tree.
- $\mathscr{P}$ is totally sorted w.r.t. φ in a list ι.
- Recursively, we poll (and remove) the best subgroup s^* from ι, and we add s^* to $\mathscr{R}$ if it is not redundant with any subgroup in $\mathscr{R}$.

It requires however that the pool of patterns has a reasonable cardinality which may be problematic with MCTS. The allowed budget must enable such post-processing (e.g., one million of iterations with 4GB RAM very high branching factors of about 600).

Outline

3. Diverse Pattern Set Discovery
   - 3.1 Problem settings
   - 3.2 Exhaustive Search
   - 3.3 Heuristic Search with Beam-search
   - 3.4 Monte Carlo Tree Search
   - 3.5 Pattern mining with MCTS
   - 3.6 A few experiments

Tuning the MCTS

After a very important number of experiments, it seems that the best tuning is:

- *single player UCB* (SP-MCTS) for the select policy
- *min-gen-expand* policy with AMAF activated **surprising!**
- *direct-freq-roll-out* policy for the simulations *but it depends...*
- *max-reward* policy as aggregation function of the rewards of a simulation *but it depends...*
- *top-10* memory policy *but it depends...*
- *max-update* policy for the back-propagation. *but it depends...*

Finding patterns hidden in artificial data

Definition (Evaluation measure)

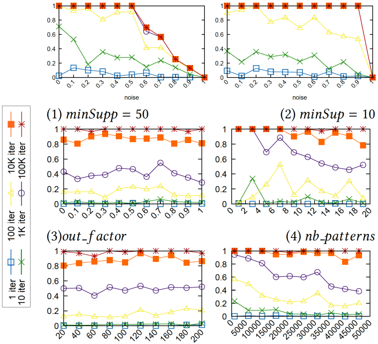
Let $\mathscr{H}$ be the set of hidden patterns, and $\mathscr{F}$ the set of patterns found by an MCTS mining algorithm, the quality of the found collection is given by:

$qual(\mathscr{H}, \mathscr{F}) = avg_{\forall h \in \mathscr{H}}(max_{\forall f \in \mathscr{F}}(Jaccard(supp(h), supp(f))))$,

that is, the average of the quality of each hidden pattern, which is the best Jaccard coefficient with a found pattern. We thus measure the *diversity*. This measure is pessimistic in the sense that it takes its maximum value 1 if and only if **all** patterns are **completely** retrieved.

Finding patterns hidden in artificial data

number of objects $nb\_obj = 50,000$
number of attributes $nb\_attr = 25$
domain size per attribute $domain\_size = 50$
number of hidden patterns $nb\_patterns = 25$
support of each hidden pattern $pattern\_sup = 100$
probability of a pattern labeled − $out\_factor = 0.1$
probability of a object to be noisy $noise\_rate = 0.1$



(1) $minSupp = 50$ (2) $minSup = 10$

(3)out_factor (4) $nb\_patterns$

(5) $pattern\_sup$ (6) nb_obj

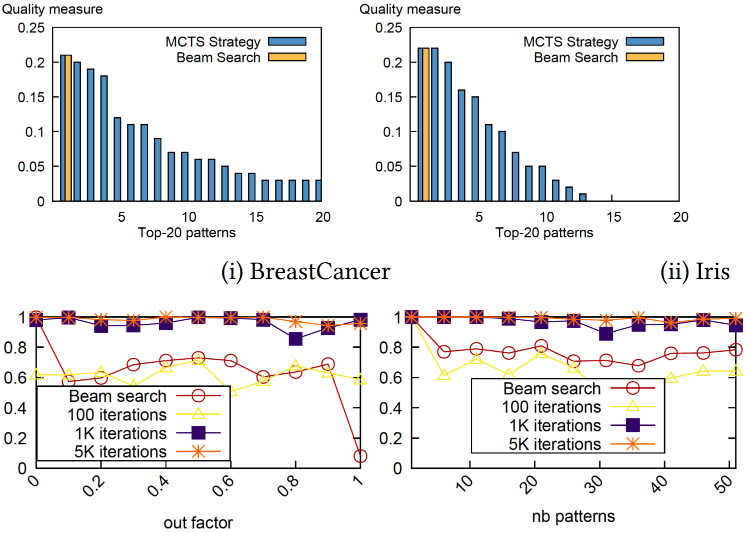
Comparing other paradigms

MCTS4DM	1K iterations		50K iterations		100K iterations	
	t(s)	$max(\varphi)$	t(s)	$max(\varphi)$	t(s)	$max(\varphi)$
BreastCancer	**0.185**	**0.210**	3.254	**0.210**	6.562	**0.210**
Cal500	0.746	0.025	5.717	0.026	12.082	0.027
Emotions	0.770	0.054	7.090	0.068	14.024	0.069
Ionosphere	0.354	0.188	5.688	0.196	10.847	0.198
Iris	**0.105**	**0.222**	2.941	**0.222**	36.302	**0.222**
Mushroom	1.087	0.118	8.141	0.118	21.299	0.118
Nursery	1.334	0.076	5.653	0.076	5.701	0.076
TicTacToe	**0.173**	**0.069**	2.446	**0.069**	2.364	**0.069**
Yeast	4.776	0.027	26.976	0.032	49.785	0.032

Existing	Beam search		ROC-Search		SD-Map	
approaches	t(s)	$max(\varphi)$	t(s)	$max(\varphi)$	t(s)	$max(\varphi)$
BreastCancer	1.334	0.208	4.318	0.207	0.7	0.184
Cal500	**21.609**	**0.044**	>180	-	1.05	0.029
Emotions	**30.476**	**0.117**	>180	-	11.45	0.075
Ionosphere	15.482	0.202	4.618	0.201	0.97	0.069
Iris	1.335	**0.222**	1.664	**0.222**	0.73	0.164
Mushroom	1.591	0.173	10;512	0.173	**2.65**	**0.194**
Nursery	10.667	**0.145**	**4.219**	**0.145**	30.4	**0.145**
TicTacToe	1.335	**0.069**	1.340	**0.069**	0.85	**0.069**
Yeast	>180	-	>180	-	**47.37**	**0.055**

Comparing other paradigms

Redundancy and diversity: beam seach vs. MCTS



(i) BreastCancer

(ii) Iris

Real life dataset: back to olfaction!

Neuroscientist colleagues gave us a dataset contains 1,689 molecules described by 82 physico-chemical properties (e.g., the molecular weight, the number of carbon atoms, etc.) and associated
Goal: Extract subgroups given by a description that are characteristic of the *Musk* odor using φ as the F1-score.

- Best known exhaustive approach SD-Map: 477.8 seconds and best quality measure of $\textit{F1-Score} = 0.45$.
- MCTS: 1 million of iterations in 99 seconds (average over 5 runs) with the best quality measure found is $\textit{F1-Score} = 0.47$.
- SD-Map discretize numerical attributes with a greedy heuristic!
- 300 attributes: MCTS gives results, exhaustive search can't.

Conclusion on MCTS

- Heuristic search of supervised patterns becomes mandatory with large datasets. Standard heuristics lead to a weak diversity in pattern sets: Only few local optima are found.
- MCTS: An exploration strategy leading to *“any-time"* pattern mining that can be adapted with different measures and policies.
- The experiments show that MCTS provides a much better diversity in the result set than existing heuristic approaches.

Conclusion on MCTS

- Interesting subgroups are found in reasonable amount of iterations and the quality of the result iteratively improves.
- MCTS is a powerful exploration strategy that can be applied to several, if not all, pattern mining problems that need to optimize a quality measure given a subset of objects.
- The main difficulties are to be able to deal with large branching factors, and jointly deal with several quality measures and interactions (remember your previous class on pattern mining and preferences), that is, skylines, progressive widening, bandit with infinite arms, streaming data, … **exciting research**!

Outline

4. Pattern-based classification

Pattern-based classification

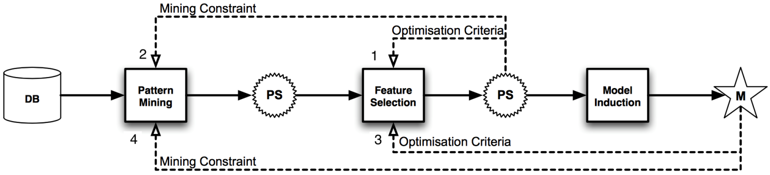
Using patterns in predictive models

- proposed as a means to obtain more accurate and more interpretable models, although there is an tradeoff between accuracy and interpretability
- obtain models for structured domains: graphs, sequences, ...
- Two general dimensions:
  - Post-processing vs. Iterative Mining: mining patterns and build model vs. integrating pattern mining, feature selection and model construction
  - Model dependent vs. model independent: explicitly take into account the type of model in which the pattern will be used or not.

Björn Bringmann, Siegfried Nijssen, and Albrecht Zimmermann
Pattern-Based Classification: A Unifying Perspective.
In Frequent Pattern Mining, Springer, 2014.

General schema

	Model-Independent	Model-dependent
Post-Processing	1	3
Iterative	2	4

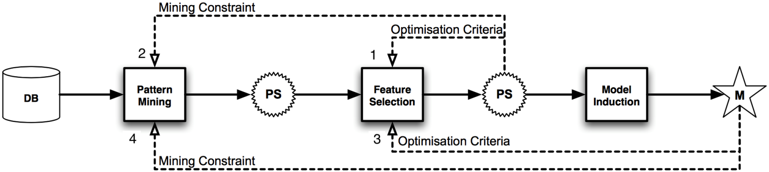


Model-independent approaches

- Use any constraint-based pattern mining algorithm, subgroup discovery, EMM, hypotheses in FCA, ...

  Petra Kralj Novak, Nada Lavrac, and Geoffrey I. Webb.
  Supervised descriptive rule discovery: A unifying survey [...].
  In Journal of Machine Learning Research, 10:377–403, 2009

- Building classifiers directly is inefficient, over-fitting and poor model interpretability: Two possibilities
  - **1.** Mine once, select patterns iteratively
  - **2.** Iterate: mine and pick w.r.t. the previous iterations to increase diversity/coverage (remember, top-k patterns are redundant)



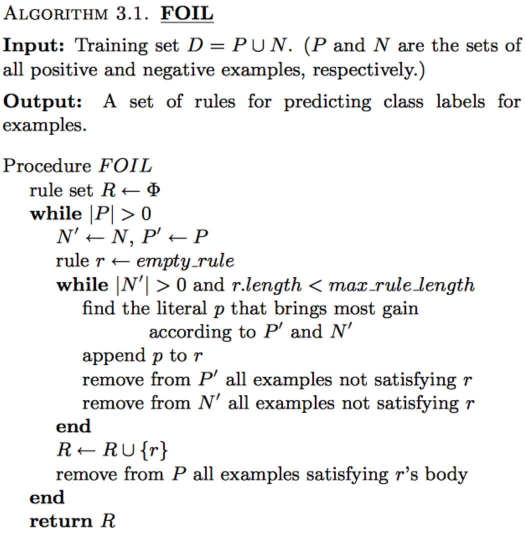
3. Model-dependent post-processing

associative classification, *classification rules*

- Conflict resolution strategy
  1. Compute patterns/rules for each class
  2. When a new example arrives, a score is computed for each class from the patterns for that class.
  3. Vote (e.g. majority)
- sequential covering/weighted covering paradigm
  1. Compute and sort the patterns;
  2. select a pattern according to this sorting order;
  3. optionally remove some of the remaining unselected patterns;
  4. optionally resort remaining unselected patterns according to updated scores
  5. optionally remove/weight transactions
  6. recursively continue selecting a pattern

4. Model-dependent iterative mining

- First Order Inductive Learner (FOIL, Quinlan 1993)



Algorithm 3.1. **FOIL**

Input: Training set $D = P \cup N$. (P and N are the sets of all positive and negative examples, respectively.)

Output: A set of rules for predicting class labels for examples.

```
Procedure $FOIL$
    rule set $R \leftarrow \Phi$
    **while** $|P| > 0$
        $N' \leftarrow N$, $P' \leftarrow P$
        rule $r \leftarrow empty\_rule$
        **while** $|N'| > 0$ and $r.length < max\_rule\_length$
            find the literal $p$ that brings most gain
                    according to $P'$ and $N'$
            append $p$ to $r$
            remove from $P'$ all examples not satisfying $r$
            remove from $N'$ all examples not satisfying $r$
        **end**
        $R \leftarrow R \cup \{r\}$
        remove from $P$ all examples satisfying $r$'s body
    **end**
    **return** $R$
```

4. Model-dependent iterative mining

- Decision trees: iteratively search for discriminant patterns that split data as well as possible according to, e.g., the information gain
- Instance based: patterns are searched when a new instanced is presented (lazy classification)
- Boosting strategies, iteratively discover a rule(s) and weight the examples
- Regression strategies: weighted sum of patterns, weights are found by linear regression

4. Model-dependent iterative mining

- It is not clear what the best methods are as there is no deep experimental comparison but a myriad of (often in favor their authors) experimental studies.

Outline

5. Concluding remarks

Concluding remarks

- Discriminant pattern mining with Exceptional Model Mining
  - A pattern domain: FCA helps with patterns structures
  - A model: You can think about anything!
  - A measure: Compare distribution, trees, classification models, ...
- Computating discriminant patterns
  - eliciting hypotheses from data
  - Building classifiers
  - Exhaustive approaches fail: heuristic required
  - Redundancy, diversity, coverage are keys
  - MCTS a novel a promising paradigm