An Introduction to Formal Concept Analysis for Biclustering Applications

Mehdi Kaytoue

mehdi.kaytoue@insa-lyon.fr
http://liris.cnrs.fr/mehdi.kaytoue

15 April 2016

The Knowledge Discovery Process

Identified domain(s)

↓ Data acquisition (crawling, scraping, interviews)

Rough data

↓ Selection and preparation

↓ Transformation : cleaning and formatting

Prepared data

↓ Data mining (Numerical & symbolic methods)

Extracted units

↓ Interpretation and evaluation

↓ Knowledge representation formalism

Knowledge units

↓

Knowledge based systems

An interactive and iterative process guided by an analyst and knowledge of the domain

The Knowledge Discovery Process

- Large volumes of data from which useful, significant and reusable units should be extracted
- Involves several tasks of data and knowledge processing
  - Mining: ((closed) frequent ...) pattern mining (itemset, sequences, graphs,...)
  - Modeling: hierarchy of concepts and relations
  - Representing: Concepts and relations as knowledge units
  - Reasoning and solving problems: classification and case based reasonning
- Many domains of applications
  - Scientific data (agronomy, astronomy, chemistry, cooking, medicine)
  - Sensors data ((interactions) traces of human/system behaviors)

A basic example: What can say a binary table?

Assume a **binary** table M_{ij} obtained by an interview

- A set of clients c_i
- A set of products p_j
- The **relation** states that some clients bought some products
- The table may of course be "**big**" (millions of lines, thousands of columns)
- The table may contain **errors**

c/p	p1	p2	p3	p4	p5
c1	x				x
c2	x	x	x	x	x
c3	x		x		
c4			x		x
c5		x	x	x	x
c6	x	x	x		x
c7	x		x	x	x
c8	x	x	x		
c9	x			x	
c10		x	x		x

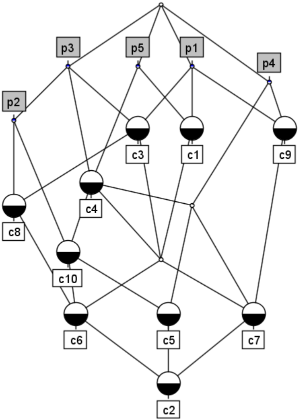
Let's make the table speak!

- $\{p2, p3, p5\}$ is an **itemset** of frequency $4/10 = 0.4$.
- $\{p3, p5\}$ has $6/10 = 0.6$ as frequency
- $p3 \wedge p5 \longrightarrow p2$ is an **association rule** with a confidence of $4/6 = 0.66$: if a client buys $p3$ and $p5$, 0.66 is the probability he buys also $p2$.

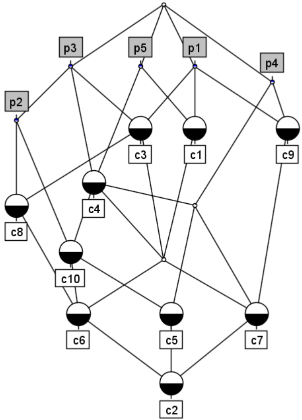
c/p	p1	p2	p3	p4	p5
c1	x				x
c2	x	x	x	x	x
c3	x		x		
c4			x		x
c5		x	x	x	x
c6	x	x	x		x
c7	x		x	x	x
c8	x	x	x		
c9	x			x	
c10		x	x		x

$$conf(X \to Y) = sup(X \cup Y)/sup(X)$$

FCA and the Concept lattice, a synthetic view



- What can we say about $\{p2\}$? and $\{p2, p3\}$?
- What about $p2 \rightarrow p3$?
- What about $p3 \rightarrow p5$?
- How to classify object described by $\{p2, p3\}$?
- What if lines are products and columns their attributes?



Formal concepts can be represented in a KR formalism (eg.. DLs)

- $Concept_1 \equiv \exists hasAwR.p3$
- $Concept_2 \equiv \exists hasAwR.p2$
- $Concept_2 \sqsubseteq Concept_1$
- ...

FCA and the Concept lattice, a synthetic view

Useful for many tasks of DM, DB, KR; Gives a formalism

- (frequent (closed)) itemsets
- (partial) implications or association rules
- Possible knowledge units to be reused for problem solving

What happens when

- When there are too much patterns ?
  Closure, iceberg, stability, ... – See the previous classes!
- When the table is not binary?
  Scaling, pattern structures
- When the table is *n*-dimensional?
  Triadic and polyadic concept analysis
- When relations arise between objects themselves?
  Relational concept analysis

Today

Objectives of this class

- Basics of formal concept analysis: from a binary table to a concept lattice
- Handling numerical data with scaling and pattern structures
- Handling multi-dimensional data with triadic concepts analysis
- Understanding the problem of biclustering
- Solving some of the biclustering problems with FCA

Outline

Binary relations

Definition (Binary relation)

A binary relation R between two arbitrary sets M and N is defined on the Cartesian product $M \times N$ and consists of pairs (m, n) with $m \in M$ and $n \in N$. When $(m, n) \in R$, we usually write mRn.

Definition (Order relation)

A binary relation R on a set M is called an order relation (or shortly order) if it satisfies the following conditions for all elements $x, y, z \in M$:

1. (reflexivity) xRx
2. (antisymmetry) xRy and $x \neq y \Rightarrow$ not yRx
3. (transitivity) xRy and $yRz \Rightarrow xRz$

Total and partial orders

Definition (Ordered set)

Given an order relation $\leq$ on a set M, an ordered set is a pair $(M, \leq)$. When $\leq$ is a partial order, $(M, \leq)$ is called partially ordered set, or poset for short.

Example: Given a set E, $(2^E, \subseteq)$

Definition (Total order)

For any $a, b \in M$, either $a \leq b$ or $b \leq a$.

Example: real numbers

Infimum, Supremum

Definition (Infimum, supremum)

Let $(M, \leq)$ be an ordered set and A a subset of M. A lower bound of A is an element s of M with $s \leq a$ for all $a \in A$. An upper bound of A is defined dually. If it exists a largest element in the set of all lower bounds of A, it is called the infimum of A and is denoted by "inf A" or $\bigwedge A$; dually, a least upper bound is called supremum and denoted by "sup A" or $\bigvee A$. Infimum and supremum are frequently called respectively meet and join, also denoted respectively by the symbols $\sqcap$ and $\sqcup$.

Lattice

Definition (Lattice, complete lattice)

A poset $\mathcal{V} = (V, \leq)$ is a lattice, if for any two elements $x, y \in V$ the supremum $x \vee y$ and the infimum $x \wedge y$ always exist. $\mathcal{V}$ is called a complete lattice if for any subset $X \subseteq V$, the supremum $\bigvee X$ and the infimum $\bigwedge X$ exist. Every complete lattice $\mathcal{V}$ has a largest element $\bigvee$ called the unit element denoted by 1_V. Dually, the smallest element 0_V is called the zero element. We will rather use the symbol bottom $\perp$ for 0_V and top $\top$ for the unit element in the following.

Remark

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

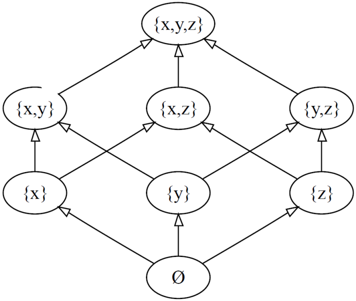
$$x \leq y \iff x = x \wedge y \iff x \vee y = y$$

$$\{a\} \leq \{a, b\} \iff \{a\} = \{a\} \cap \{a, b\}$$

$$\{a\} \leq \{a, b\} \iff \{a\} \cup \{a, b\} = \{a, b\}$$

This remark is important for understanding pattern structures

Hasse diagram of the powerset lattice

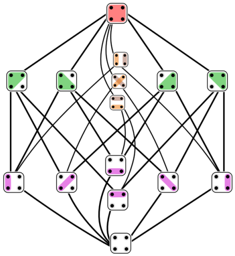


$$x \leq y \iff x = x \wedge y \iff x \vee y = y$$

$$\{a\} \leq \{a, b\} \iff \{a\} = \{a\} \cap \{a, b\}$$

$$\{a\} \leq \{a, b\} \iff \{a\} \cup \{a, b\} = \{a, b\}$$

Hasse diagram of the partition lattice



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

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

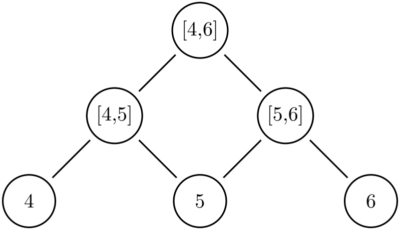
$$\{\{a, b, c\}, \{d\}\} \wedge \{\{a, b, d\}, \{c\}\} = \{\{a, b\}, \{c\}, \{d\}\}$$

Semi-lattices

Definition (Join-semi-lattice and meet-semi-lattice)

A poset $\mathcal{V} = (V, \leq)$ is a join-semi-lattice if for any two elements $x, y \in V$ the supremum $x \vee y$ always exists. Dually, it is a meet-semi-lattice if the infimum $x \wedge y$ always exists. A lattice is a poset that is both a meet- and join-semi-lattice with respect to the same partial order.

Hasse diagram of a semi-lattice



How can we formulate here $\leq$ and $\wedge$?

Closure operator

Let S be a set and ψ a mapping from the power set[1] of S into the power set of S, i.e. $\psi : \mathcal{P}(S) \longrightarrow \mathcal{P}(S)$.

Definition (Closure operator)

ψ is called a closure operator on S if, for any $A, B \subseteq S$, it is:

1. extensive: $A \subseteq \psi(A)$,
2. monotone: $A \subseteq B$ implies that $\psi(A) \subseteq \psi(B)$, and
3. idempotent: $\psi(\psi(A)) = \psi(A)$.

A subset $A \subseteq S$ is ψ-closed if $A = \psi(A)$. The set of all ψ-closed $\{A \subseteq S \mid A = \psi(A)\}$ is called a closure system.

[1] The power set of any set S, written $\mathcal{P}(S)$, or 2^S, is the set of all subsets of S, including the empty set and S itself, hence composed of $2^{|S|}$ elements.

Outline

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	×		×		×	×

$G = \{g_1, \ldots, g_7\}$ “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.

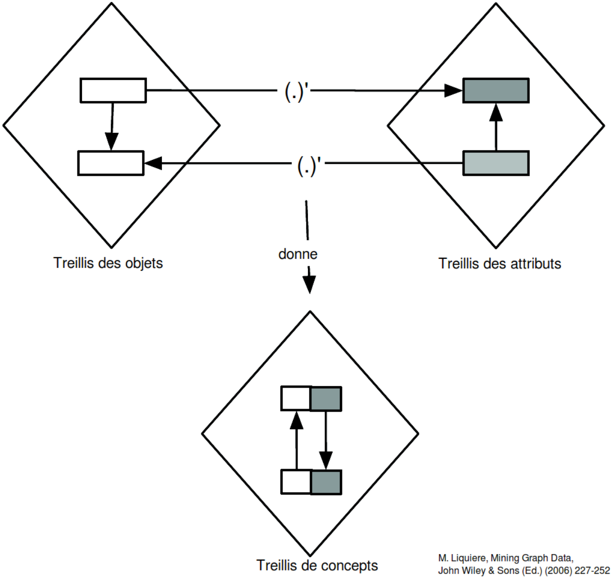
Galois connection

Let P and Q be ordered sets. A pair of maps $\phi : P \rightarrow Q$ and $\psi : Q \rightarrow P$ 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 $(\mathcal{P}(G), \subseteq)$ and $(\mathcal{P}(M), \subseteq)$ with $\leq \equiv \subseteq$.

Galois connection illustration



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

Theorem (The Basic Theorem on Concept Lattices)

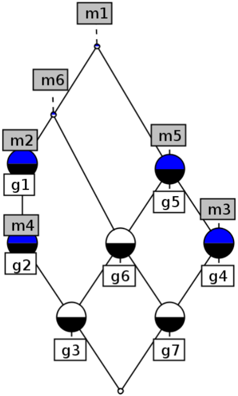
The concept lattice $\underline{\mathfrak{B}}(G, M, I)$ is a complete lattice in which infimum and supremum are given by:

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

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

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

A basic algorithm for computing formal concepts

Remember that

Each concept of a formal context (G, M, I) has the form (A'', A') for some subset $A \subseteq G$ and the form (B', B'') for some subset $B \subseteq M$.

One does naively apply the closure operator $(.)''$ on all possible subsets of G (dually all subsets of M), and remove all redundant concepts (How to generate these subsets?)

Inefficient

Several algorithms exist. Their performance is usually linked with the density $d = \dfrac{|I|}{|G| \times |M|}$ of a context (G, M, I). Time complexity is generally $O(|G|^2|M||L|)$ (L being the set of concepts).

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			×

$\emptyset$

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

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

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

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

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

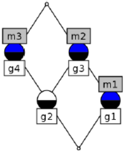
$\cancel{\{g_1, g_3\}' = \{m_2\}}$
$\cancel{\{g_1, g_3\}'' = \{g_1, g_2, g_3\}}$

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

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

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

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



Outline

Definition (Many-valued context)

A many-valued context (G, M, W, I) consists of sets G, M and W and a ternary relation I between those three sets, i.e. $I \subseteq G \times M \times W$, for which it holds that

$$(g, m, w) \in I \text{ and } (g, m, v) \in I \quad \text{always imply} \quad w = v$$

The fact $(g, m, w) \in I$ means "the attribute m takes value w for object g", simply written as $m(g) = w$.

	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

Conceptual scale

Definition

A (conceptual) scale for the attribute m of a many-valued context is a (one-valued) context $\mathcal{S}_m = (G_m, M_m, I_m)$ with $m(G) = \{m(g), \forall g \in G\} \subseteq G_m$. The objects of a scale are called scale values, the attributes are called scale attributes.

Basic scales

- **Nominal scale** is defined by the context $(W_m, W_m, =)$. We obtain the following scales, respectively for attribute m_1, m_2 and m_3:

=	4	5	6
4	×		
5		×	
6			×

=	7	8	9
7	×		
8		×	
9			×

=	4	5	6	8
4	×			
5		×		
6			×	
8				×

$$W_m \subseteq W, \forall m \in M$$

Resulting context

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

Basic scales

- **Ordinal scale** is given by the context $(W_m, W_m, \leq)$ where $\leq$ denotes classical real number order. We obtain for each attribute the following scales:

$\leq$	4	5	6
4	×	×	×
5		×	×
6			×

$\leq$	7	8	9
7	×	×	×
8		×	×
9			×

$\leq$	4	5	6	8
4	×	×	×	×
5		×	×	×
6			×	×
8				×

- **Interordinal scale** is given by $(W_m, W_m \leq) \mid (W_m, W_m \geq)$ where $\mid$ denotes the apposition of two contexts[2]. We obtain for attribute m_1 the following scale[3]:

	≤ 4	≤ 5	≤ 6	≥ 4	≥ 5	≥ 6
4	×	×	×	×		
5		×	×	×	×	
6			×	×	×	×

[2] The *apposition of two contexts* with identical sets of objects, denoted by $\mid$, returns the context with the same set of objects and the set of attributes being the disjoint union of attribute sets of the original contexts.

[3] The double-line column separator intuitively corresponds to context apposition.

Is scaling a valid way to consider non binary data?

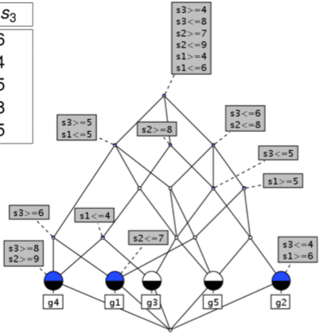
- Consider interordinal scaling.
- What is the concept lattice of its context?
- What does it represent?
- What are the problem?
- Can we do better?

Pattern structures formalize a nice alternative.

Concept lattice with interordinal scaling

	$s_1 \leq 4$	$s_1 \leq 5$	$s_1 \leq 6$	$s_1 \geq 4$	$s_1 \geq 5$	$s_1 \geq 6$
g_1		×	×	×	×	
g_2			×	×	×	×
g_3	×	×	×	×		
g_4	×	×	×	×		
g_5		×	×	×	×	

	s_1	s_2	s_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



Outline

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 $\subseteq$

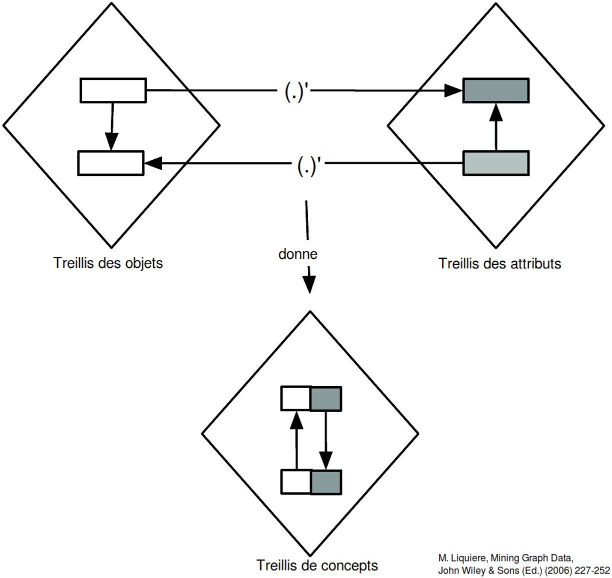
$$N \cap O = N \iff N \subseteq O$$
$$\{m_1\} \cap \{m_1, m_2\} = \{m_1\} \iff \{m_1\} \subseteq \{m_1, m_2\}$$

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

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

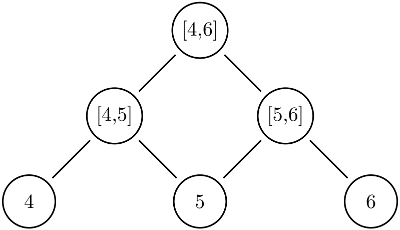
A. Tversky
Features of similarity.
In *Psychological Review, 84 (4), 1977.*

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

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

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

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

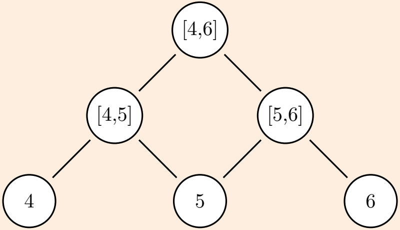
$$d^{\square} = \{g \in G | d \sqsubseteq \delta(g)\} \qquad \text{for } d \in (D, \sqcap)$$

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



$$[a_1, b_1] \sqcap [a_2, b_2] = [min(a_1, a_2), max(b_1, b_2)]$$
$$[4, 4] \sqcap [5, 5] = [4, 5]$$

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

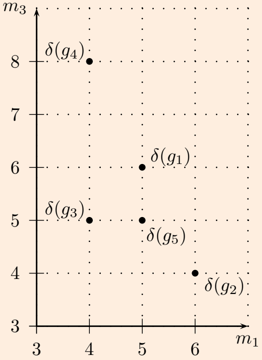
$$\begin{aligned}\{g_1, g_2\}^{\square} &= \bigsqcap_{g \in \{g_1, g_2\}} \delta(g) \\ &= \langle 5, 7, 6\rangle \sqcap \langle 6, 8, 4\rangle \\ &= \langle [5, 6], [7, 8], [4, 6]\rangle\end{aligned}$$

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

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

Interval patterns as (hyper) rectangles

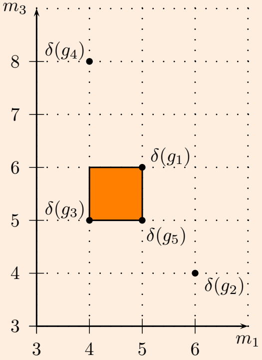
	m_1		m_3
g_1	5		6
g_2	6		4
g_3	4		5
g_4	4		8
g_5	5		5



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

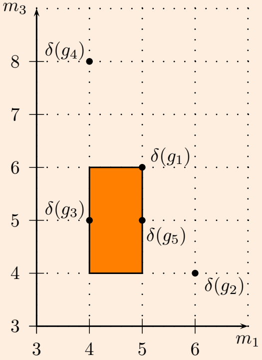
$$\langle [4,5],[5,6] \rangle^{\square} = \{g_1, g_3, g_5\}$$



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

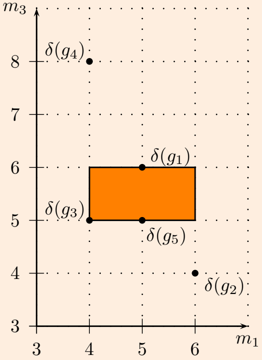
$$\langle [4,5],[5,6] \rangle^{\square} = \{g_1, g_3, g_5\}$$
$$\langle [4,5],[4,6] \rangle^{\square} = \{g_1, g_3, g_5\}$$



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

$$\langle [a_{m_1}, b_{m_1}], [a_{m_2}, b_{m_2}], \ldots \rangle$$
$$\text{where } a_{m_i}, b_{m_i} \in W_{m_i}$$

	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

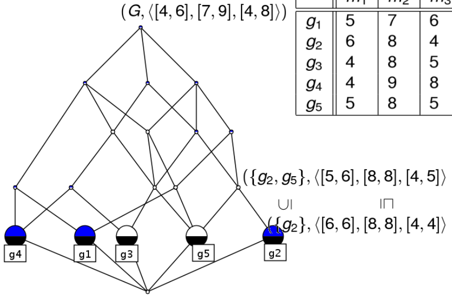
$$\prod_{i \in \{1,\ldots,|M|\}} \frac{|W_{m_i}| \times (|W_{m_i}| + 1)}{2}$$

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

Elements of order theory
Formal Concept Analysis
Algorithms
Conceptual Scaling
Pattern structures
Triadic Concepts
Biclustering
Motivations
A naive approach
Mining biclusters of constant values
Mining biclusters of similar values
Mining *n*-dimensional clusters
Conclusion
References



	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).

Equivalence classes of interval patterns

Two interval patterns with same image are said to be equivalent

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

Equivalence class of a pattern d

$$[d] = \{c | c \cong d\}$$

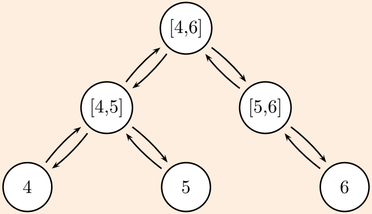
- with a unique closed pattern: the smallest rectangle
- and one or several generators: the largest rectangles

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

Outline

1. Elements of order theory
2. Formal Concept Analysis
3. Algorithms
4. Conceptual Scaling
5. Pattern structures
6. Triadic Concepts
7. Biclustering
   - Motivations
   - A naive approach
   - Mining biclusters of constant values
   - Mining biclusters of similar values
   - Mining *n*-dimensional clusters
8. Conclusion
9. References

Triadic Concept Analysis

“Extension” of FCA to ternary relation

- An object has an attribute for a given condition
- Triadic context (G, M, B, Y)
- Several derivation operators allowing to characterize “triadic concepts” as maximal cubes of $\times$

b_1

	m_1	m_2	m_3
g_1			$\times$
g_2	$\times$	$\times$	
g_3		$\times$	$\times$
g_4		$\times$	$\times$
g_5		$\times$	$\times$

b_2

	m_1	m_2	m_3
g_1	$\times$	$\times$	$\times$
g_2	$\times$	$\times$	
g_3	$\times$	$\times$	$\times$
g_4		$\times$	$\times$
g_5		$\times$	$\times$

b_3

	m_1	m_2	m_3
g_1	$\times$		$\times$
g_2	$\times$		
g_3	$\times$	$\times$	$\times$
g_4		$\times$	$\times$
g_5	$\times$	$\times$	$\times$

$(\{g_3, g_4, g_5\}, \{m_2, m_3\}, \{b_1, b_2, b_3\})$ is a triadic concept

F. Lehmann and R. Wille.
A Triadic Approach to Formal Concept Analysis.
In International Conference on Conceptual Structures (ICCS), 1995.

Derivation operators

Definition

Triconcept forming operators - outer closure

$$\Phi : X \rightarrow X^{(i)} : \{(a_j, a_k) \in K_j \times K_k \mid (a_i, a_j, a_k) \in Y \text{ forall } a_i \in X\}$$

$$\Phi' : Z \rightarrow Z^{(i)} : \{a_i \in K_i \mid (a_i, a_j, a_k) \in Y \text{ for all } (a_j, a_k) \in Z\}$$

Definition

Triconcept forming operators - inner (dyadic) closure

$$\Psi : X_i \rightarrow X_i^{(i,j,A_k)} : \{a_j \in K_j \mid (a_i, a_j, a_k) \in Y \text{ for all } (a_i, a_k) \in X_i \times A_k\}$$

$$\Psi' : X_j \rightarrow X_j^{(i,j,A_k)} : \{a_i \in K_i \mid (a_i, a_j, a_k) \in Y \text{ for all } (a_j, a_k) \in X_j \times A_k\}$$

Without going into details...

A Naive approach

- Start with a set of conditions and a context (G, M, J) which involves all these conditions
- Compute all dyadic concepts (inner closure)
- For any dyadic concept, compute the set of conditions that contains it (outer closure).
- Do it for any subset of conditions
- Remove redundant tri-concepts.

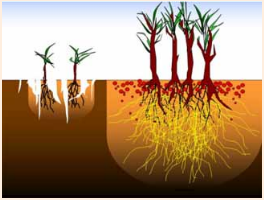
What happens if we have n dimensions?

Data-peeler: An algorithm based on a binary tree enumeration: For each node, choose a dimension and an element, generates two n-sets one with the element, the other without. Constraints are used to prune the search space and detect maximal n-sets.

See also Trias algorithm

Outline

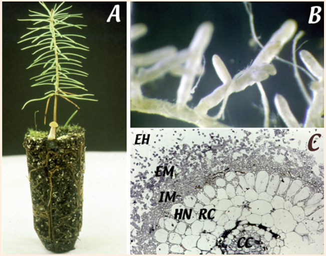
Somewhere... in a temperate forest...



A biological problem

INRA: How does symbiosis work at the cellular level?

- Analyse biological processes
- Find genes involved in symbiosis
- Choose a model for understanding symbiosis: ***Laccaria bicolor***



Analysing Gene Expression Data (GED)

F. Martin et al.
The Genome of Laccaria Bicolor Provides Insights into Mycorrhizal Symbiosis.
In *Nature.*, 2008.

Microarray Data

Gene Expression Matrix (GEM): $E = (e_{ij})_{1 \leq i \leq n, 1 \leq j \leq m}$
Gene Expression Profile (GEP): e_i *with* $1 \leq i \leq n$
A Gene Expression Value: e_{ij}

Gene \ Situation	a	b	c
g_1	11050	11950	1503
g_2	13025	14100	1708
g_3	6257	5057	6500
g_4	5392	6020	7300
g_5	13070	12021	15548

Integer values in $[0, 65535]$ (NimbleGen Systems Oligonucleotide Arrays Technology)

(Lee et al, 2002): genes having a similar expression **profile**

- interact together within the same biological **process**,
- or have a similar biological **function**.

Numerical Clustering Methods

Many clustering methods

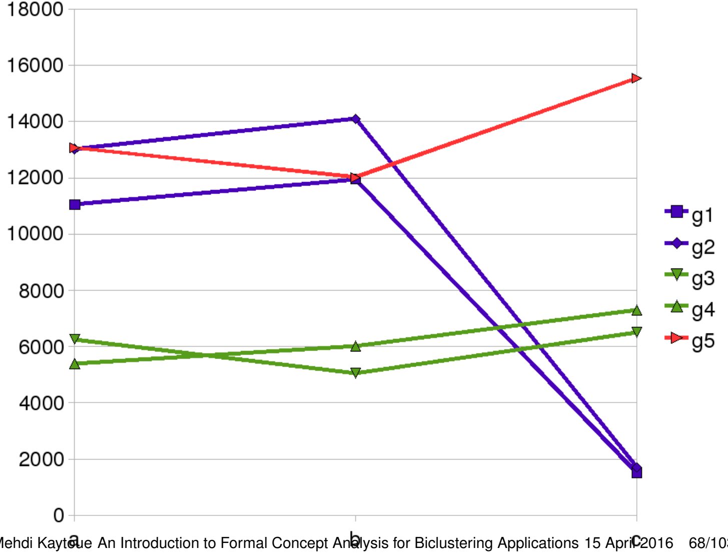
- K-means (Gasch et al, 1999)
- Self Organizing Maps (Tamayo et al, 1999)
- Hierarchical clustering (Eisen et al, 1998)

Gene \ Situation	a	b	c
g_1	11050	11950	1503
g_2	13025	14100	1708
g_3	6257	5057	6500
g_4	5392	6020	7300
g_5	13070	12021	15548

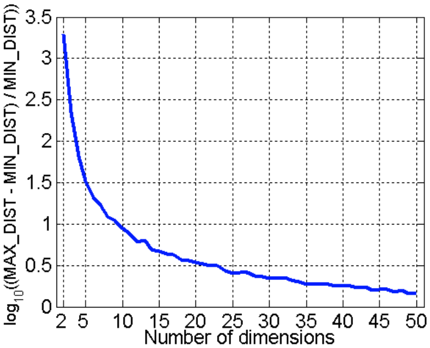
A cluster is a set of similar gene expression profiles (GEP)

See (Jiang et al., 2004) for a survey

Same example where each line is a GEP representation



First problem



With 500 random points

Clustering methods have to respect the following biological properties

1. A gene can participate in several processes
2. A situation can describe several processes
3. A process involves a small subset of genes
4. A process is active in none, some or all situations
5. High variations of expression values are not frequent between two situations

In Clustering Words . . .

1. A gene may belong to many clusters (overlapping)
2. A situation may belong to many clusters (overlapping)
3. Control gene dimension size of clusters
4. Control situation dimension size of clusters
5. Groups of genes are interesting if showing high similar changes of expression

Gene \ Situation	a	b	c
g_1	11050	11950	1503
g_2	13025	14100	1708
g_3	6257	5057	6500
g_4	5392	6020	7300
g_5	13070	12021	15548

A bi-cluster is a set of similar GEP in some situations

Extracting (maximal) rectangles in numerical data

A set of genes co-expressed in some biological situations

- **Local patterns**: biological processes may be activated in some situations only
- **Overlapping patterns**: a gene may be involved in several biological process

	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	6
g_2	2	1	1	0	6
g_3	2	2	1	**7**	**6**
g_4	8	9	2	**6**	**7**

Biclustering numerical data

Bicluster should reflect

- a **local** phenomena in the data: “rectangles of values”
- **connectedness** of values: e.g. similar values
- **overlapping**: objects/attributes may belong to several patterns
- a partial **order**, e.g. for algorithmic issues
- **maximality** of rectangles

Several types of biclusters

1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0

1.0	1.0	1.0	0.0
2.0	2.0	2.0	2.0
3.0	3.0	3.0	3.0
4.0	4.0	40	40

1.0	2.0	3.0	4.0
1.0	2.0	3.0	4.0
1.0	2.0	3.0	4.0
1.0	2.0	3.0	4.0

1.0	2.0	5.0	0.0
2.0	3.0	6.0	1.0
4.0	5.0	8.0	3.0
5.0	6.0	9.0	4.0

1.0	2.0	0.5	1.5
2.0	4.0	1.0	3.0
4.0	8.0	2.0	6.0
3.0	6.0	1.5	4.5

Several applications...

- Collaborative filtering and recommender systems
- Finding web communities
- Gene expression analysis, ...

Several algorithms

- Iterative Row and Column Clustering Combination
- Divide and Conquer / Distribution Parameter Identification
- Greedy Iterative Search / Exhaustive Bicluster Enumeration

A difficult problem generally relying on heuristics

S. C. Madeira and A. L. Oliveira
Biclustering Algorithms for Biological Data Analysis: a survey.
In *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2004.

Different Stages

In terms of data analysis:

In terms of formal concept analysis (FCA):

In the following: **a concept (A, B) represents a set of genes A having similar expression values in situations of B. The notion of similarity will be given by an interval of values apriori computed.**

Stage 1: a GEM as a numerical data table

A GEM is represented by a many valued context (G, S, W, I_1)

Example

- $G = \{Gene\ 1, \ldots, Gene\ 5\}$, a set of objects.
- $S = \{a, b, c\}$, a set of attributes.
- $W = \{11050, 11950, \ldots\}$, a set of unique values.
- I_1 is illustrated, for example, by $Gene\ 1(a) = 11050$.

I_1	a	b	c
Gene 1	11050	11950	1503
Gene 2	13025	14100	1708
Gene 3	6257	5057	6500
Gene 4	5392	6020	7300
Gene 5	13070	12021	15548

We need a **formal context** (binary) which is a triple (G, S_T, I_2):
I_2 says an object $g \in G$ possesses an attribute $s \in S_T$ or not.

Interval scaling (discretization):
Each attribute is cut into p attributes considering p disjoint ordered intervals of the set $\{[0, u_1],]u_1, u_2], \ldots,]u_{p-1}, u_p]\}$.

Given an index set T on the set of intervals, $t \in T$,

$$g(s) = x \text{ becomes } (g, (s, t)) \text{ if } x \in [u_{t-1}, u_t]$$

$$(g, (s, t)) \in I_2$$

means

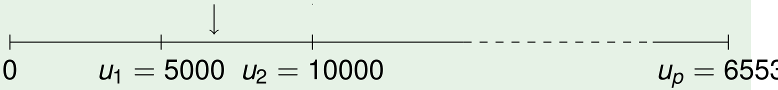
g has an expression value in the t^{th} interval for situation s

Stage 2: Interval Scaling

Example

T is an index set on
$$\{[0, 5000],]5000, 10000],]10000, 65535]\}$$

$g_3(a) = 6257$ becomes $(g_3, (a, 2))$



The interval borders are choosen by the biologists and directly influence the number of concepts.

Stage 3: Concept Lattice Construction

I_2	$(a,1)$	$(a,2)$	$(a,3)$	$(b,1)$	$(b,2)$	$(b,3)$	$(c,1)$	$(c,2)$	$(c,3)$
g_1			×			×	×		
g_2			×			×	×		
g_3		×			×			×	
g_4		×			×			×	
g_5			×			×			×

Definition

A **formal concept** is a pair (A, B) where $A \subseteq G$, $B \subseteq S_T$ such as $A' = B$ and $B' = A$ given the following derivation operators:

$$' : 2^G \rightarrow 2^S_T; A' = \{s \in S_T; \forall g \in A : (g, s) \in I_2\}$$

$$' : 2^S_T \rightarrow 2^G; B' = \{g \in G; \forall s \in B : (g, s) \in I_2\}$$

A is extent of the concept and B the intent.

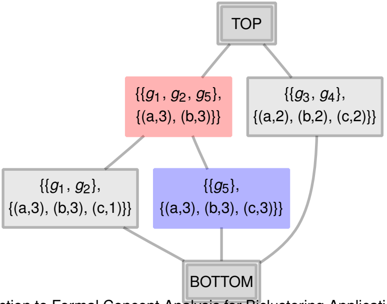
For a given concept, the genes of the extent A are co-expressed in sense of the intent B.

Stage 3: Concept Lattice Construction

Definition

The **concept lattice** of the context (G, S, I_2) is the hierarchy of the whole set of concepts partially ordered by :

$$(A_1, B_1) \sqsubseteq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2 \ (or\ B_2 \subseteq B_1)$$



Stage 4: Concept Filtering

Too many patterns?

- A concept is a relevant bi-cluster if the extent is not composed of “too many” genes, and if the intent contains a least “a few” situations.
- A first filtering step keeps only concepts (A, B), where $|A| \leq x$ and $|B| \geq y$.
  x and y are chosen by the biologist
- Many concepts describe groups of co-expressed genes having a similar expression with no radical change of expression.

Example

$(\{g_3, g_4\}, \{(a, 2), (b, 2), (c, 2)\})$ presents no change. Keep those with the maximal and strongest changes!

Starting from

- $\mathbb{K}_1 = (G, S, W, I)$ with $|G| = 22,294$ and $|S| = 7$ biological situations like roots, fruit and in symbiosis root cells,
- an index set T on the set of disjoint intervals whose borders are: 0, 100, 250, 500, 1000, 2500, 5000, 7500, 10000, 12500, 15000, 17500, 20000, 30000, 40000, 65535,
- $\mathbb{K}_2 = (G, S \times T, I)$ where $|S \times T| = 98$,

we obtain $146,504$ **formal concepts. We filter out concepts** (A, B) **such as**

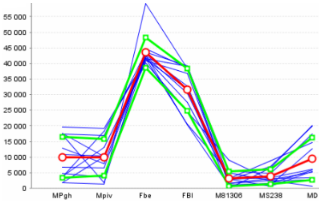
- $|A| \leq 50$,
- $|B| \geq 4$ and
- B is a $(4,4) - \textit{variant}$ intent.

We finally obtain 156 concepts.

Some results

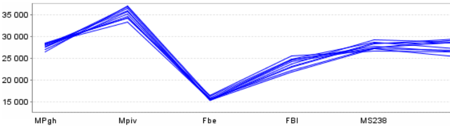
Some genes involved in fructification ?

$|A| = 9$ and $|B| = 7$
(A, B) represents a cluster ($|B|$ is maximal)



Some genes sharing a similar function ?

$|A| = 9$ and $|B| = 6$
(A, B) represents a bicluster ($|B|$ is not maximal)



Experimental validation required

A first type of biclusters

Bicluster of equal values

A bicluster (A, B) is a bicluster of similar values if

$$m_i(g_j) = m_k(g_l), \forall g_j, g_l \in A, \forall m_i, m_k \in B$$

	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	**6**
g_2	2	1	1	0	**6**
g_3	2	2	1	7	**6**
g_4	8	9	2	6	7

Maximal bicluster of equal values

(A, B) is maximal if either

- $(A \cup g, B)$, $g \in G \backslash A$ is not a bicluster of equal values
- $(A, B \cup m)$, $m \in M \backslash B$ is not a bicluster of equal values

A natural solution!

<table>
<tr><th>$w \in W$</th><th>$\mathbb{K}_w$</th><th>$\mathfrak{B}_w$</th><th>Bicluster corresponding to first concept on left list</th></tr>
<tr><td>1</td><td>

	m_1	m_2	m_3	m_4	m_5
g_1	×			×	
g_2		×	×		
g_3			×		
g_4					

</td><td>$(\{g_2, g_3\}, \{m_3\})$
$(\{g_2\}, \{m_2, m_3\})$
$(\{g_1\}, \{m_1, m_4\})$</td><td>

	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	6
g_2	2	1	**1**	0	6
g_3	2	2	**1**	7	6
g_4	8	9	2	6	7

</td></tr>
<tr><td>6</td><td>

	m_1	m_2	m_3	m_4	m_5
g_1					×
g_2					×
g_3					×
g_4				×	

</td><td>$(\{g_1, g_2, g_3\}, \{m_5\})$
$(\{g_4\}, \{m_4\})$</td><td>

	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	**6**
g_2	2	1	1	0	**6**
g_3	2	2	1	7	**6**
g_4	8	9	2	6	7

</td></tr>
<tr><td>...</td><td>...</td><td>...</td><td>...</td></tr>
</table>

Allows to directly find groups of users with same ratings (1-5 stars data)! Will not work for data with many attribute values: a notion of similarity is needed.

Biclusters of similar values

A similarity relation

$$w_1 \simeq_\theta w_2 \iff |w_1 - w_2| \leq \theta \text{ with } \theta \in \mathbb{R}, w_1, w_2 \in W$$

Bicluster of similar values

A bicluster (A, B) is a bicluster of similar values if

$$m_i(g_j) \simeq_\theta m_k(g_l), \forall g_j, g_l \in A, \forall m_i, m_k \in B$$

	m_1	m_2	m_3	m_4	m_5
g_1	**1**	**2**	**2**	1	6
g_2	**2**	**1**	**1**	0	6
g_3	**2**	**2**	**1**	7	6
g_4	8	9	2	6	7

$\theta = 1$

and maximal if no object/attribute can be added

J. Besson, C. Robardet, L. De Raedt, J.-F. Boulicaut
Mining Bi-sets in Numerical Data
In KDID 2006: 11-23

Concept example

$(\{g_2, g_3\}, \langle [2,2], [1,2], [1,1], [0,7], [6,6] \rangle)$

	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	6
g_2	**2**	**1**	**1**	**0**	**6**
g_3	**2**	**2**	**1**	**7**	**6**
g_4	8	9	2	6	7

$\theta = 1$

3 statements to verify

- Some intervals have a “size” larger than θ
- Some values in two different columns may not be similar
- Rectangle may not be maximal

M. Kaytoue, S. O. Kuznetsov, and A. Napoli
Biclustering Numerical Data in Formal Concept Analysis

First statement

Avoiding intervals with size larger than θ

$$[a_1, b_1] \sqcap [a_2, b_2] = \begin{cases} [min(a_1, a_2), max(b_1, b_2)] & \text{if} |max(b_1, b_2) - min(a_1, a_2)| \leq \theta \\ * & \text{otherwise} \end{cases}$$

Going back to our example, with $\theta = 1$

$$(\{g_2, g_3\}, \langle [2, 2], [1, 2], [1, 1], *, [6, 6] \rangle)$$

	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	6
g_2	**2**	**1**	**1**	0	**6**
g_3	**2**	**2**	**1**	7	**6**
g_4	8	9	2	6	7

Second statement

Values from two columns should be similar

From

$$(\{g_2, g_3\}, \langle [2,2], [1,2], [1,1], *, [6,6] \rangle)$$

we group attributes such as their values form a class of tolerance:

	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	6
g_2	**2**	**1**	**1**	0	6
g_3	**2**	**2**	**1**	7	6
g_4	8	9	2	6	7

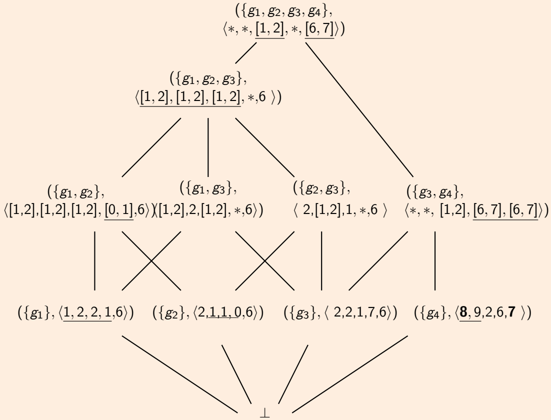
$$(\{g_2, g_3\}, \{m_1, m_2, m_3\})$$

	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	6
g_2	2	1	1	0	**6**
g_3	2	2	1	7	**6**
g_4	8	9	2	6	7

$$(\{g_2, g_3\}, \{m_5\})$$

Third statement

Maximal bicluster of similar values



Constructing maximal biclusters: bottom-up/top-down

What about a bicluster at some period of time? In the summer? for young people? ...:
Many dimensions can be added

Exercise

How to discovery maximal *n*-rectangles of constant values?

What about biclusters of similar values on their columns with a discretization?

What about biclusters of similar values?
i.e. *n*-dimensional rectangles of pairwise similar values

Basic idea

Principle

- Start from a numerical dataset (G, M, W, I)
- Build a triadic context (G, M, B, Y) with same objects, same attributes, and discretized dimension
- Extract triadic concepts

Interordinal scaling

B and all its intersections characterize any interval over W

We show interesting links between biclusters of similar values and triadic concepts

Mehdi Kaytoue, Sergei O. Kuznetsov, Juraj Macko, Amedeo Napoli:
Biclustering meets triadic concept analysis.
Ann. Math. Artif. Intell. 70(1-2): 55-79 (2014)

Discretization method

Interodinal scaling (existing discretization scale)

- Let (G, M, W, I) be a numerical dataset (with W the set of data-values.
- Now consider the set $T = \{[min(W), w], \forall w \in W\} \cup \{[w, max(W)], \forall w \in W\}$.
- Known fact: T and all its intersections characterize any interval of values on W.

Example

With $W = \{0, 1, 2, 6, 7, 8, 9\}$, one has

- $T = \{[0, 0], [0, 1], [0, 2], ..., [0, 9], [1, 9], [2, 9], ..., [9, 9]\}$
- and for example $[0, 8] \cap [2, 9] = [2, 8]$

Building a triadic context

Transformation procedure

From a numerical dataset (G, M, W, I), build a triadic context (G, M, T, Y) such as $(g, m, t) \in Y \iff m(g) \in t$

	$t_1 = [0,0]$					$t_2 = [0,1]$					$t_3 = [0,2]$					$t_4 = [0,6]$					$t_5 = [0,7]$				
	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5
g_1						×			×		×	×	×	×		×	×	×	×	×	×	×	×	×	×
g_2				×			×	×	×		×	×	×	×		×	×	×	×	×	×	×	×	×	×
g_3								×			×	×	×			×	×	×		×	×	×	×	×	×
g_4													×					×	×				×	×	×

	$t_6 = [0,8]$					$t_7 = [0,9]$					$t_8 = [1,9]$					$t_9 = [2,9]$					$t_{10} = [6,9]$				
	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5
g_1	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×		×	×		×					×
g_2	×	×	×	×	×	×	×	×	×	×	×	×	×		×	×				×					×
g_3	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×				×	×
g_4	×		×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×		×	×

	$t_{11} = [7,9]$					$t_{12} = [8,9]$					$t_{13} = [9,9]$				
	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5
g_1															
g_2															
g_3				×											
g_4	×	×			×	×	×					×			

Theorem

There is a 1-1-correspondence between

- (i) Triadic concepts of the resulting triadic context
- (ii) Biclusters of similar values maximal for some $\theta \geq 0$

Interesting facts

- Efficient algorithm for concept extraction (Data-Peeler, handling several constraints)

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

- Top-k biclusters: Concept (A, B, C) with high $|A|$, $|B|$, and $|C|$ corresponds to bicluster (A, B) as a large rectangle of close values (by properties of interordinal scale)
- This formalization allows us to design a new algorithm to extract maximal biclusters for a given parameter θ

If we really need to specify θ?

Compute all max. biclusters for a given θ

- Use another (but similar) discretization procedure to build the triadic context based on tolerance blocks
- Standard algorithms output biclusters of similar values but not necessarily maximal
- We design a new algorithm TriMax for that task

TriMax **is flexible, uses standard FCA algorithms in its core, seems better than its competitors, can be extended to *n*-ary relations and distributed.**

New transformation procedure

Tolerance blocks based scaling

- Compute the set C of all blocks of tolerance over W
- From the numerical dataset (G, M, W, I), build the triadic context (G, M, C, Z) such that
  $(g, m, c) \in Z \iff m(g) \in c$
- Actually, we remove "useless information"

	label 1					label 2					label 3					label 4					label 5				
	$[0, 1]$					$[1, 2]$					$[6, 7]$					$[7, 8]$					$[8, 9]$				
	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5
g_1	×			×		×	×	×	×						×										
g_2		×	×	×		×	×	×							×										
g_3			×			×	×	×						×	×				×						
g_4								×						×	×	×				×	×	×			

$$\theta = 1$$

Second contribution

Algorithm TriMax

- Any triadic concept corresponds to a bicluster of similar values, but not necessarily maximal!
- It lead us to the algorithm TriMax that:
  - Process each formal context (one for each block of tolerance) with any existing FCA algorithm
  - Any resulting concept is a maximal bicluster candidate and
  - Each context can be processed separately
- TriMax **allows a complete, correct and non redundant extraction of all maximal biclusters of similar values for a user defined similarity parameter** θ

Leveraging the problem of biclustering with FCA

- Roots of closed pattern mining
- A set of ”tools” and algorithms
- Allows a generic and direct way of computing biclusters of many kinds
- Extends to multi-dimensional data
- Allows parallel computing

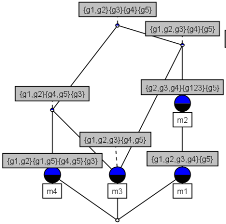
Thanks for your attention.

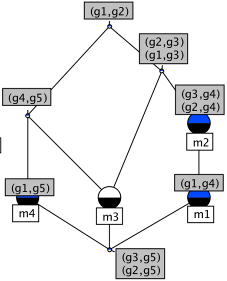
Biclusters with similar values on columns

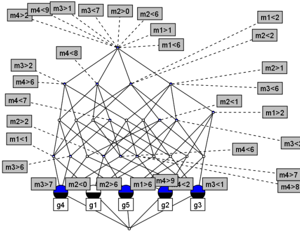
Given a numerical dataset (G, M, W, I), a pair (A, B) (where $A \subseteq G, B \subseteq M$) is called a bicluster of similar values on columns when the following statement holds: $\forall g, h \in A, \forall m \in B, m(g) \simeq_\theta m(h)$ A bicluster (A, B) is maximal if $\nexists g \in G \backslash A$ such that $(A \cup \{g\}, B)$ is a bicluster, and $\nexists m \in M \backslash B$ such that $(A, B \cup \{m\})$ is a bicluster.

- Can you find all of them in a pattern concept lattice?
- First solution: each object is described by a vector of intervals (easier that what's been done before)
- Second solution: each attribute is described by a partition if $\theta = 0$, a *tolerance* otherwise (generalizes partitions)
- Third solution: a partition can be described by a formal context (one for each attribute, thus a triadic context).

Elements of solution







	m_1	m_2	m_3	m_4
g_1	1	2	2	8
g_2	2	1	2	9
g_3	2	1	1	2
g_4	1	0	7	6
g_5	6	6	6	7

Elements of solution

m_1	g_1	g_2	g_3	g_4	g_5
g_1	×	×	×	×	
g_2	×	×	×	×	
g_3	×	×	×	×	
g_4	×	×	×	×	
g_5					×

m_2	g_1	g_2	g_3	g_4	g_5
g_1	×	×	×		
g_2	×	×	×	×	
g_3	×	×	×	×	
g_4		×	×	×	
g_5					×

m_3	g_1	g_2	g_3	g_4	g_5
g_1	×	×	×		
g_2	×	×	×		
g_3	×	×	×		
g_4				×	×
g_5				×	×

m_4	g_1	g_2	g_3	g_4	g_5
g_1	×	×			×
g_2	×	×			
g_3			×		
g_4				×	×
g_5	×			×	×

References

- Marc Barbut and Bernard Monjardet, Ordre et classification, Hachette, 1970.
- Nathalie Caspard, Bruno Leclerc, Bernard Monjardet, Ensembles ordonnés finis concepts, résultats et usages (Mathématiques et Applications), 2007
- Bernhard Ganter and Rudolph Wille, Formal Concept Analysis, Springer, 1999
- Oded Maimon, Lior Rokach (Eds.), The Data Mining and Knowledge Discovery Handbook, Springer, 2005.
- Claudio Carpineto and Giovanni Romano, Concept Data Analysis: Theory and Applications, John Wiley & Sons, 2004.
- Sergei O. Kuznetsov, Galois Connections in Data Analysis: Contributions from the Soviet Era and Modern Russian Research, Formal Concept Analysis 2005: 196-225
- Bernhard Ganter, Sergei O. Kuznetsov: Pattern Structures and Their Projections. ICCS 2001: 129-142
- Amedeo Napoli, An Introduction to Symbol Methods for Knowledge Discovery. Handbook of Categorization in Cognitive Science, 1st Edition, Cohen & Lefebvre (Eds.), 2005.
- Sergei O. Kuznetsov, Sergei A. Obiedkov: Comparing performance of algorithms for generating concept lattices. J. Exp. Theor. Artif. Intell. 14(2-3): 189-216 (2002)
- Franz Baader, Bernhard Ganter, Baris Sertkaya, Ulrike Sattler, Completing Description Logic Knowledge Bases Using Formal Concept Analysis. IJCAI 2007: 230-235