Formal Concept Analysis: Themes and Variations for Knowledge Discovery

Sergei O. Kuznetsov[1] and Amedeo Napoli[2]
[1] School of Applied Mathematics and Information Science, National Research University Higher School of Economics, Moscow, Russia
[2] LORIA (CNRS – INRIA Nancy Grand-Est – Université de Lorraine)
B.P. 239, 54506 Vandoeuvre les Nancy, France
skuznetsov@yandex.ru;Amedeo.Napoli@loria.fr

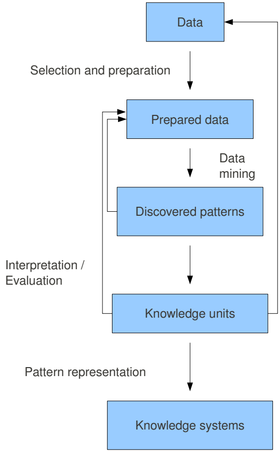
Tutorial on Formal Concept Analysis at IJCAI 2013

IJCAI 2013, Beijing, August 3rd 2013

Summary of the presentation

Knowledge Discovery guided by Domain Knowledge (1)

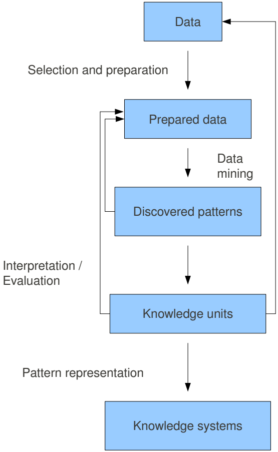
- The process of Knowledge Discovery guided by Domain Knowledge (KDDK) is applied on large volumes of data for extracting information units which are useful, significant, and reusable.
- KDDK is based on four main operations: data preparation, data mining, interpretation and representation of the extracted units.
- KDDK is iterative and interactive, guided by an analyst, and by domain knowledge.



KDDK is an interactive and iterative process that can be replayed.

Knowledge Discovery guided by Domain Knowledge (2)

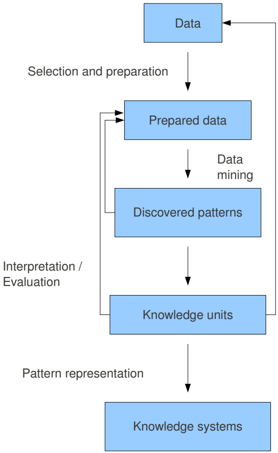
- One the core idea of KDDK is classification, which is involved in all tasks of data and knowledge processing:
- mining: Formal Concept Analysis (FCA), pattern mining...
- modeling: hierarchy of concepts and relations,
- representing: concepts and relations as knowledge units,
- reasoning and problem solving: classification-based and case-based reasoning.



KDDK is an interactive and iterative process that can be replayed.

Knowledge Discovery guided by Domain Knowledge (3)

- KDDK is used for knowledge engineering and problem-solving activities in some application domains:
  - agronomy
  - astronomy
  - biology
  - chemistry
  - cooking
  - medicine



FCA, Formal Concepts and Concept Lattices

- Marc Barbut and Bernard Monjardet, Ordre et classification, Hachette, 1970.
- Claudio Carpineto and Giovanni Romano, Concept Data Analysis: Theory and Applications, John Wiley & Sons, 2004.
- Bernhard Ganter and Rudolph Wille, Formal Concept Analysis, Springer, 1999.

The FCA process

- The basic procedure of Formal Concept Analysis (FCA) is based on a simple representation of data, i.e. a binary table called a formal context.
- Each formal context is transformed into a mathematical structure called concept lattice.
- The information contained in the formal context is preserved.
- The concept lattice is the basis for data analysis.
It is represented graphically to support analysis, mining, visualization, interpretation...

The notion of a formal context

Objects / Attributes	m1	m2	m3	m4	m5
g1		x	x		x
g2	x		x	x	
g3	x	x	x	x	
g4	x			x	
g5	x	x	x	x	
g6	x		x	x	

- $(\mathtt{G}, \mathtt{M}, \mathtt{I})$ is called a formal context where $\mathtt{G}$ (*Gegenstände*) and $\mathtt{M}$ (*Merkmale*) are sets, and $\mathtt{I} \subseteq \mathtt{G} \times \mathtt{M}$ is a binary relation between $\mathtt{G}$ and $\mathtt{M}$.
- The elements of $\mathtt{G}$ are the objects, while the elements of $\mathtt{M}$ are the attributes, $\mathtt{I}$ is the incidence relation of the context $(\mathtt{G}, \mathtt{M}, \mathtt{I})$.

Two derivation operators

- For $\mathtt{A} \subseteq \mathtt{G}$: $\mathtt{A}' = \{\mathtt{m} \in \mathtt{M}/(\mathtt{g},\mathtt{m}) \in \mathtt{I} \texttt{ for all } \mathtt{g} \in \mathtt{A}\}$
- Dually, for $\mathtt{B} \subseteq \mathtt{M}$: $\mathtt{B}' = \{\mathtt{g} \in \mathtt{G}/(\mathtt{g},\mathtt{m}) \in \mathtt{I} \texttt{ for all } \mathtt{m} \in \mathtt{B}\}$

$\{\mathtt{g3}\}'$ and $\{\mathtt{m3}\}'$:

Objects / Attributes	m1	m2	m3	m4	m5
g1		x	x		x
g2	x		x	x	
g3	x	x	x	x	
g4	x			x	
g5	x	x	x	x	
g6	x		x	x	

Two derivation operators

$\{g3, g5\}'$

Objects / Attributes	m1	m2	m3	m4	m5
g1		x	x		x
g2	x		x	x	
g3	x	x	x	x	
g4	x			x	
g5	x	x	x	x	
g6	x		x	x	

Two derivation operators

$\{\mathrm{m3}, \mathrm{m4}\}'$

Objects / Attributes	m1	m2	m3	m4	m5
g1		x	x		x
g2	x		x	x	
g3	x	x	x	x	
g4	x			x	
g5	x	x	x	x	
g6	x		x	x	

The derivation operators and the Galois connection

- The derivation operators establish a Galois connection between the power sets $\wp(\mathtt{G})$ and $\wp(\mathtt{M})$ (and thereby a dual isomorphism between two closure systems).

A Galois connection is defined as follows:

- Let $\mathtt{P}$ and $\mathtt{Q}$ be ordered sets.
  A pair of maps $\phi : \mathtt{P} \longrightarrow \mathtt{Q}$ and $\psi : \mathtt{Q} \longrightarrow \mathtt{P}$ is called a Galois connection between $\mathtt{P}$ and $\mathtt{Q}$ if:
- (i) $\mathtt{p}_1 \leq \mathtt{p}_2 \Longrightarrow \phi(\mathtt{p}_1) \geq \phi(\mathtt{p}_2)$
  (ii) $\mathtt{q}_1 \leq \mathtt{q}_2 \Longrightarrow \psi(\mathtt{q}_1) \geq \psi(\mathtt{q}_2)$
  (iii) $\mathtt{p} \leq \psi \circ \phi(\mathtt{p})$ and $\mathtt{q} \leq \phi \circ \psi(\mathtt{q})$

The Galois connection and the closure operators

- $' : \wp(\mathtt{G}) \longrightarrow \wp(\mathtt{M})$ with $\mathtt{A} \longrightarrow \mathtt{A}'$
- $' : \wp(\mathtt{M}) \longrightarrow \wp(\mathtt{G})$ with $\mathtt{B} \longrightarrow \mathtt{B}'$
- These two applications induce a Galois connection between $\wp(\mathtt{G})$ and $\wp(\mathtt{M})$ when sets are ordered by set inclusion relation.

The Galois connection and the closure operators

A closure operator on a set $\mathtt{H}$ is a map κ such that:

- $\kappa : \wp(\mathtt{H}) \longrightarrow \wp(\mathtt{H})$
- For all $\mathtt{A}_1, \mathtt{A}_2 \subseteq \mathtt{H}$:
    - (i) $\mathtt{A}_1 \subseteq \kappa(\mathtt{A}_1)$
    - (ii) $\mathtt{A}_1 \subseteq \mathtt{A}_2$ then $\kappa(\mathtt{A}_1) \subseteq \kappa(\mathtt{A}_2)$
    - (iii) $\kappa(\kappa(\mathtt{A}_1)) = \kappa(\mathtt{A}_1)$
- $\mathtt{A}$ is a closed set whenever $\kappa(\mathtt{A}) = \mathtt{A}$.
- The composition operators $''$ (composition of $'$ and $'$ are closure operators.

Formal concept

Given a formal context $(\mathtt{G}, \mathtt{M}, \mathtt{I})$:

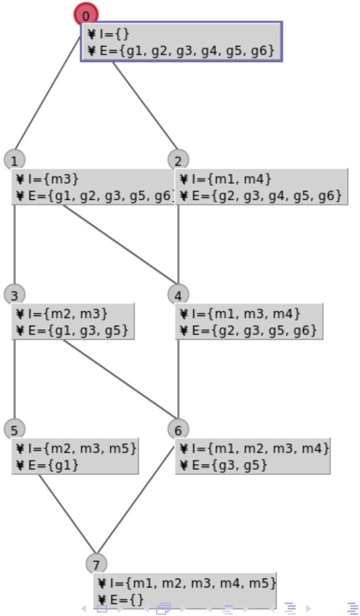
- $\mathtt{A}' = \{\mathtt{m} \in \mathtt{M} / (\mathtt{g}, \mathtt{m}) \in \mathtt{I} \texttt{ for all } \mathtt{g} \in \mathtt{A}\}$
- $\mathtt{B}' = \{\mathtt{g} \in \mathtt{G} / (\mathtt{g}, \mathtt{m}) \in \mathtt{I} \texttt{ for all } \mathtt{m} \in \mathtt{B}\}$
- $(\mathtt{A}, \mathtt{B})$ is a formal concept of $(\mathtt{G}, \mathtt{M}, \mathtt{I})$ iff:
  $\mathtt{A} \subseteq \mathtt{G}$, $\mathtt{B} \subseteq \mathtt{M}$, $\mathtt{A}' = \mathtt{B}$, and $\mathtt{A} = \mathtt{B}'$.
- $\mathtt{A}$ is the extent and $\mathtt{B}$ is the intent of $(\mathtt{A}, \mathtt{B})$.
- The mappings $\mathtt{A} \longrightarrow \mathtt{A}''$ and $\mathtt{B} \longrightarrow \mathtt{B}''$ are closure operators.

The concept lattice

- Formal concepts can be ordered by:
  $(A_1, B_1) \leq (A_2, B_2) \iff A_1 \subseteq A_2$ (dually $B_2 \subseteq B_1$).
- The set $\underline{\mathfrak{B}}(G, M, I)$ of all formal concepts of (G, M, I) with this order is a complete lattice called the concept lattice of (G, M, I).
- Recall that: a set $(P, \leq)$ is a complete lattice if the supremum $\bigvee S$ and the infimum $\bigwedge S$ exist for any subset S of P.
- Every complete lattice has a top or unit element denoted by $\top$, and a bottom or zero element denoted by $\perp$.

The concept lattice

G / M	m1	m2	m3	m4	m5
g1		×	×		×
g2	×		×	×	
g3	×	×	×	×	
g4	×			×	
g5	×	×	×	×	
g6	×		×	×	



The basic theorem of FCA

- The concept lattice $\underline{\mathfrak{B}}(G, M, I)$ is a complete lattice in which the infimum and the supremum are given by:
- $\bigwedge_{k \in K}(A_k, B_k) = (\bigcap_{k \in K} A_k, (\bigcup_{k \in K} B_k)'')$
- $\bigvee_{k \in K}(A_k, B_k) = ((\bigcup_{k \in K} A_k)'', \bigcap_{k \in K} B_k)$
- Note: an intersection of closed sets is a closed set but a union of closed sets is not necessarily a closed set.

The properties of the derivation operators

Iterating derivation

- $\mathtt{A}' = \{\mathtt{m} \in \mathtt{M}/(\mathtt{g},\mathtt{m}) \in \mathtt{I} \text{ for all } \mathtt{g} \in \mathtt{A}\}$
- $\mathtt{B}' = \{\mathtt{g} \in \mathtt{G}/(\mathtt{g},\mathtt{m}) \in \mathtt{I} \text{ for all } \mathtt{m} \in \mathtt{B}\}$
- The derivation operators can be combined: Starting with a set $\mathtt{A} \subseteq \mathtt{G}$, we obtain that $\mathtt{A}'$ is a subset of $\mathtt{M}$.
- Applying the second operator on this set, we get $(\mathtt{A}')'$, or $\mathtt{A}''$ for short, which is a set of objects.
- Continuing, we obtain $\mathtt{A}'''$, $\mathtt{A}''''$, and so on.

Properties of the derivation operators

- $\mathtt{A}' = \{\mathtt{m} \in \mathtt{M}/(\mathtt{g},\mathtt{m}) \in \mathtt{I} \texttt{ for all } \mathtt{g} \in \mathtt{A}\}$
- $\mathtt{B}' = \{\mathtt{g} \in \mathtt{G}/(\mathtt{g},\mathtt{m}) \in \mathtt{I} \texttt{ for all } \mathtt{m} \in \mathtt{B}\}$

The derivation operator $'$ satisfy the following rules:

- $\mathtt{A}_1 \subseteq \mathtt{A}_2 \Longrightarrow \mathtt{A}_2' \subseteq \mathtt{A}_1'$
- $\mathtt{B}_1 \subseteq \mathtt{B}_2 \Longrightarrow \mathtt{B}_2' \subseteq \mathtt{B}_1'$
- $\mathtt{A} \subseteq \mathtt{A}''$ and $\mathtt{A}' = \mathtt{A}'''$
- $\mathtt{B} \subseteq \mathtt{B}''$ and $\mathtt{B}' = \mathtt{B}'''$

Examples

G / M	m1	m2	m3	m4	m5
g1		x	x		x
g2	x		x	x	
g3	x	x	x	x	
g4	x			x	
g5	x	x	x	x	
g6	x		x	x	

- $A_1 \subseteq A_2 \Longrightarrow A_2' \subseteq A_1'$
  $B_1 \subseteq B_2 \Longrightarrow B_2' \subseteq B_1'$
- $A \subseteq A''$ and $A' = A'''$
  $B \subseteq B''$ and $B' = B'''$

Extent closure, intent closure

The derivation operator $'$ satisfy the following rules:

- Combining the derivation operators, we get two operators of the form: $\mathtt{X} \longrightarrow \mathtt{X}''$, one on $\mathtt{G}$, the other on $\mathtt{M}$.
- For $\mathtt{A} \subseteq \mathtt{G}$ we have that $\mathtt{A}'' \subseteq \mathtt{G}$.
  The set $\mathtt{A}''$ is called the extent closure of $\mathtt{A}$.
- Dually, when $\mathtt{B} \subseteq \mathtt{M}$ we have also that $\mathtt{B}'' \subseteq \mathtt{M}$.
  The set $\mathtt{B}''$ is called the intent closure of $\mathtt{A}$.

Extent closure, intent closure

From $A \subseteq A''$ and $B \subseteq B''$ it comes:

- (i) whenever all objects from a set $A \subseteq G$ have a common attribute m, then also all objects from A'' have that attribute.
- (ii) whenever an object $g \in G$ has all attributes from $B \subseteq M$, then this object also has all attributes from B''.

Other properties of the derivation operators

For $\mathtt{A}_1, \mathtt{A}_2 \subseteq \mathtt{G}$, and dually for $\mathtt{B}_1, \mathtt{B}_2 \subseteq \mathtt{M}$, we have:

- $\mathtt{A}_1 \subseteq \mathtt{A}_2 \Longrightarrow \mathtt{A}_1'' \subseteq \mathtt{A}_2''$
- $\mathtt{B}_1 \subseteq \mathtt{B}_2 \Longrightarrow \mathtt{B}_1'' \subseteq \mathtt{B}_2''$
- $(\mathtt{A}'')'' = \mathtt{A}''$
- $(\mathtt{B}'')'' = \mathtt{B}''$

Closure operators

- As already mentioned, the operators $'$ and $''$ satisfying the above properties are closure operators.
- The sets which are images of a closure operator are the closed sets.
- Thus, in the case of a closure operator $\mathtt{X} \longrightarrow \mathtt{X}''$ the closed sets are the sets of the form $\mathtt{X}''$.

Closed sets are intents and extents

- If $(\mathtt{G}, \mathtt{M}, \mathtt{I})$ is a formal context and $\mathtt{A} \subseteq \mathtt{G}$, then $\mathtt{A}''$ is an extent.
- Conversely, if $\mathtt{A}$ is an extent of $(\mathtt{G}, \mathtt{M}, \mathtt{I})$, then $\mathtt{A} = \mathtt{A}''$.
- Dually if $\mathtt{B}$ is an intent of $(\mathtt{G}, \mathtt{M}, \mathtt{I})$, then $\mathtt{B} = \mathtt{B}''$, and every every intent $\mathtt{B}$ satisfies $\mathtt{B} = \mathtt{B}''$.
- This follows from the fact that for each subset $\mathtt{A} \subseteq \mathtt{G}$, the pair $(\mathtt{A}'', \mathtt{A}')$ is a formal concept, and that similarly, for each subset $\mathtt{B} \subseteq \mathtt{M}$, $(\mathtt{B}', \mathtt{B}'')$ is a formal concept.
- Therefore, the closed sets of the closure operator $\mathtt{A} \longrightarrow \mathtt{A}'', \mathtt{A} \subseteq \mathtt{G}$ are precisely the extents of $(\mathtt{G}, \mathtt{M}, \mathtt{I})$, and the closed sets of the operator $\mathtt{B} \longrightarrow \mathtt{B}'', \mathtt{B} \subseteq \mathtt{M}$, are precisely the intents.

The structure of the concept lattice

The reduced labeling

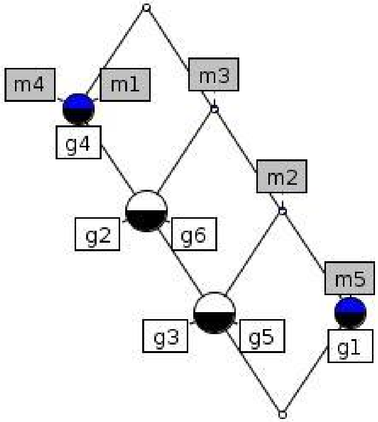
- A reduced labeling may be used allowing that each object and each attribute is entered only once in a diagram.
- The name of the object g is attached to the "lower half" of the corresponding object concept $\gamma(g) = (\{g\}'', \{g\}')$.
- The object concept of an object $g \in \mathtt{G}$ is the concept $(\{g\}'', \{g\}')$ where $\{g\}'$ is the object intent $\{\mathtt{m} \in \mathtt{M}/\mathtt{gIm}\}$ of g.
- The object concept of g, denoted by $\gamma(g)$, is the smallest concept (for the lattice order) with g in its extent.
- Example:
  $\gamma(g4) = (\{g4\}'', \{g4\}') = (\{g2, g3, g4, g5, g6\}, \{\mathtt{m1}, \mathtt{m4}\})$
  $\gamma(g1) = (\{g1\}'', \{g1\}') = (\{g1\}, \{\mathtt{m2}, \mathtt{m3}, \mathtt{m5}\})$

The reduced labeling

- The name of the attribute $\mathtt{m}$ is located to the "upper half" of the corresponding attribute concept $\mu(\mathtt{m}) = (\{\mathtt{m}\}', \{\mathtt{m}\}'')$.
- Correspondingly, the attribute concept of an attribute $\mathtt{m} \in \mathtt{M}$ is the concept $(\{\mathtt{m}\}', \{\mathtt{m}\}'')$ where $\{\mathtt{m}\}'$ is the attribute extent $\{\mathtt{g} \in \mathtt{G}/\mathtt{gIm}\}$ of $\mathtt{m}$.
- The attribute concept of $\mathtt{m}$, denoted by $\mu(\mathtt{m})$ is the largest concept (for the lattice order) with $\mathtt{m}$ in its intent.
- Example:
  $\mu(\mathtt{m1}) = (\{\mathtt{m1}\}', \{\mathtt{m1}\}'') = (\{\mathrm{g2}, \mathrm{g3}, \mathrm{g4}, \mathrm{g5}, \mathrm{g6}\}, \{\mathtt{m1}, \mathtt{m4}\})$
  $\mu(\mathtt{m1}) = \mu(\mathtt{m4})$
  $\mu(\mathtt{m2}) = (\{\mathtt{m2}\}', \{\mathtt{m2}\}'') = (\{\mathrm{g1}, \mathrm{g3}, \mathrm{g5}\}, \{\mathtt{m2}, \mathtt{m3}\})$

The reduced labeling

G / M	m1	m2	m3	m4	m5
g1		×	×		×
g2	×		×	×	
g3	×	×	×	×	
g4	×			×	
g5	×	×	×	×	
g6	×		×	×	



Reduced labeling: The attributes “at the highest” and the objects “at the lowest”.

The reduced labeling

- For any concept $(\mathtt{A}, \mathtt{B})$ we have:
- $\mathtt{g} \in \mathtt{A} \Longleftrightarrow \gamma(\mathtt{g}) \leq (\mathtt{A}, \mathtt{B})$
- $\mathtt{m} \in \mathtt{B} \Longleftrightarrow (\mathtt{A}, \mathtt{B}) \leq \mu(\mathtt{m})$

An extent is an ideal (down-set)

- The extent of an arbitrary concept can be found as the set of objects in the principal ideal generated by the concept.
- Let $(\mathtt{P}, \leq)$ be an ordered set.
  A subset $\mathtt{Q} \subseteq \mathtt{P}$ is an order ideal or a down-set if $\mathtt{x} \in \mathtt{Q}$ and $\mathtt{y} \leq \mathtt{x}$ imply that $\mathtt{y} \in \mathtt{Q}$.
- $\downarrow \mathtt{Q} = \{\mathtt{y} \in \mathtt{P} / \exists \mathtt{x} \in \mathtt{Q} : \mathtt{y} \leq \mathtt{x}\}$
  $\downarrow \mathtt{x} = \{\mathtt{y} \in \mathtt{P} / \mathtt{y} \leq \mathtt{x}\}$

An intent is a filter (up-set)

- The intent of an arbitrary concept can be found as the set of objects in the principal filter generated by the concept.
- Let $(\mathtt{P}, \leq)$ be an ordered set.
  A subset $\mathtt{Q} \subseteq \mathtt{P}$ is an order filter or an up-set if $\mathtt{x} \in \mathtt{Q}$ and $\mathtt{x} \leq \mathtt{y}$ imply that $\mathtt{y} \in \mathtt{Q}$.
- $\uparrow \mathtt{Q} = \{\mathtt{y} \in \mathtt{P} / \exists \mathtt{x} \in \mathtt{Q} : \mathtt{x} \leq \mathtt{y}\}$
  $\uparrow \mathtt{x} = \{\mathtt{y} \in \mathtt{P} / \mathtt{x} \leq \mathtt{y}\}$

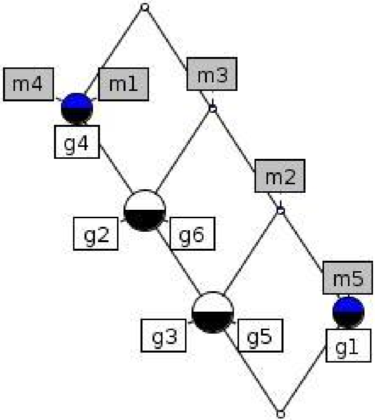
Largest and smallest concepts

- Given a concept lattice, the context associated to the lattice can be read using the following general rule:
  $(\mathtt{g}, \mathtt{m}) \in \mathtt{I} \iff \gamma(\mathtt{g}) \leq \mu(\mathtt{m})$
- Just as a set of concepts can be uniquely determined from a given context, so the context can be reconstructed from its concepts.
- The set $\mathtt{G}$ is the extent of the largest concept $(\mathtt{G}, \mathtt{G}')$.
- The set $\mathtt{M}$ is the intent of the smallest concept $(\mathtt{M}', \mathtt{M})$.
- The incidence relation is given by:
  $\mathtt{I} = \bigcup\{\mathtt{A} \times \mathtt{B} / (\mathtt{A}, \mathtt{B}) \in \mathtt{C}(\mathtt{G}, \mathtt{M}, \mathtt{I})\}$
  where $\mathtt{C}(\mathtt{G}, \mathtt{M}, \mathtt{I})$ denotes the set of all formal concepts for $(\mathtt{G}, \mathtt{M}, \mathtt{I})$.

Types of attributes

- Introducing and attribute: an attribute α is introduced in a concept `C` when it is not present in any ascendant (super-concept) of `C`, i.e. the concept `C` corresponds to the attribute concept of α (sometimes called the introducer of α).
- Inheriting an attribute: an attribute α is inherited by a concept `C` when it is already present in an ascendant of `C`, i.e. `C` is lower for the lattice order than the attribute-concept or introducer of α.

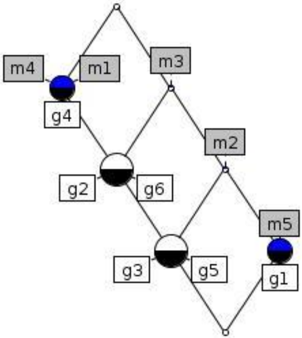
Types of attributes (example)



- m3 is an attribute introduced in the concept (g12356, m3), a and d are attributes introduced in the concept (g23456, m14), b is an attribute introduced in the concept (g135, m23).
- m3 is an attribute inherited by (g135, m23), m1, m3, and m4, are attributes inherited by (g2356, m134), and so on.

Extracting rules from a concept lattice

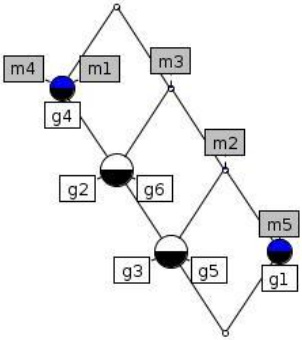
Mutual implications between attributes having the same attribute-concept



- Attributes having the same attribute-concept or introducer are equivalent: for example $\mathtt{m1} \longleftrightarrow \mathtt{m4}$ for $(\mathtt{g23456}, \mathtt{m14})$.

Extracting rules from a concept lattice (continued)

Introduced attributes imply inherited attributes



- When an attribute α is introduced, it implies every inherited attribute in the attribute-concept of α: for example $\mathtt{m2} \longrightarrow \mathtt{m3}$ for $(\mathtt{g135}, \mathtt{m23})$ and $\mathtt{m5} \longrightarrow \mathtt{m23}$ for $(\mathtt{g1}, \mathtt{m235})$.

Scaling

When the data set is large or complex

When the size of the data set is growing, it becomes unreasonable to display the full data in a single lattice diagram.
Formal Concept Analysis allows to:

- split large diagrams into smaller ones, so that the information content is preserved,
- browse through lattices and thereby build conceptual views of data.

The need for scaling

- There are many series of formal contexts that have an suggestive interpretation. Such formal contexts will be called scales.
- So formally, a scale is the same as a formal context. But it is meant to have a special interpretation.
- Examples of scales are nominal, ordinal, and dichotomic scales.

Conceptual scaling

- The formal context is the basic data type of Formal Concept Analysis.
- However data are often given in form of a many-valued context.
- Many-valued contexts are translated to one-valued context via conceptual scaling.
- But this is not automatic and some arbitrary choices have to be made.

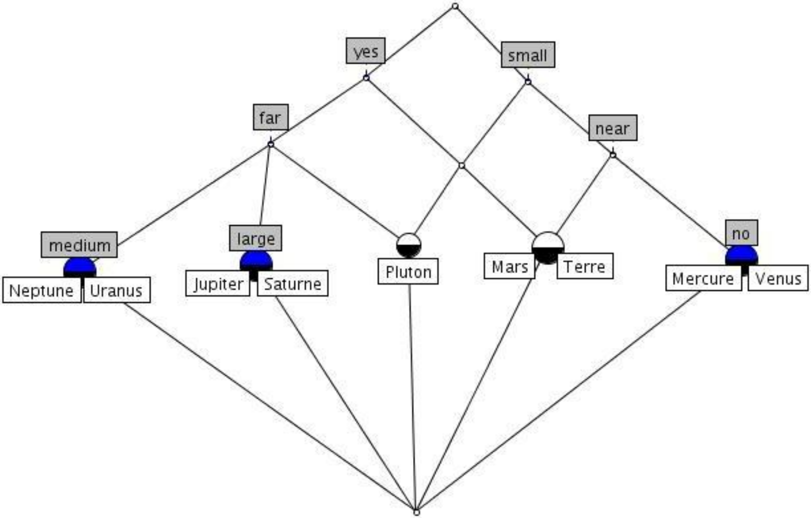
The example of the context of planets

Planet	Size	Distance to Sun	Moon(s)
Jupiter	large	far	yes
Mars	small	near	yes
Mercury	small	near	no
Neptune	medium	far	yes
Pluto	small	far	yes
Saturn	large	far	yes
Earth	small	near	yes
Uranus	medium	far	yes
Venus	small	near	no

The context of planets after nominal scaling

Planet	Size			Distance to Sun		Moon(s)	
	small	medium	large	near	far	yes	no
Jupiter			x		x	x	
Mars	x			x		x	
Mercury	x			x			x
Neptune		x			x	x	
Pluto	x				x	x	
Saturn			x		x	x	
Earth	x			x		x	
Uranus		x			x	x	
Venus	x			x			x

The concept lattice of planets (after scaling)



Examples of scaling

- Nominal: $\mathtt{K} = (\mathtt{N}, \mathtt{N}, =)$
- Ordinal: $\mathtt{K} = (\mathtt{N}, \mathtt{N}, \leq)$
- Interordinal: $\mathtt{K} = (\mathtt{N}, \mathtt{N}, \leq \cup \geq)$

Two algorithms for building formal concepts and the concept lattice

An algorithm for computing the formal concepts

- A rectangle in a binary table corresponds to a pair $(\mathtt{X}, \mathtt{Y})$ –where $\mathtt{X}$ denotes an extension and $\mathtt{Y}$ denotes an intension– only contains crosses x.
  Such an extension and intension are not necessarily extents and intents respectively.
- A rectangle $(\mathtt{X}, \mathtt{Y})$ is contained in another rectangle $(\mathtt{X}_1, \mathtt{Y}_1)$ whenever $\mathtt{X} \subseteq \mathtt{X}_1$ and $\mathtt{Y} \subseteq \mathtt{Y}_1$.
- A rectangle $(\mathtt{X}, \mathtt{Y})$ is maximal when it is not included in any other rectangle: any rectangle $(\mathtt{X}_1, \mathtt{Y}_1)$ containing a maximal rectangle $(\mathtt{X}, \mathtt{Y})$ is such that $\mathtt{X}_1$ and/or $\mathtt{Y}_1$ contain at least a "void place", i.e. a place without a cross x.

An algorithm for constructing the concept lattice

- Step 1: build the rectangles $(\mathtt{X}, \mathtt{Y})$ whose extension $\mathtt{X}$ is of size 1 (the cardinality of $\mathtt{X}$ is equal to 1).
- Step 2 with union of rectangles: build the rectangles $(\mathtt{X}, \mathtt{Y})$ of size 2 making the union of extensions of size 1 and the intersection of intensions.
  An intersection should not be empty otherwise the corresponding rectangle is no more considered.
- Step 3: Check and reomve the rectangles that are not maximal, then assemble rectangles with the same intension.
- Step 4: continue in the same way the process considering rectangles of size 3, 4, and so on. . .
- Final step: the building process stops as soon as there is no more rectangle to be built.

An example of construction of a concept lattice (1)

G / M	m1 (a)	m2 (b)	m3 (c)	m4 (d)	m5 (e)
g1		x	x		x
g2	x		x	x	
g3	x	x	x	x	
g4	x			x	
g5	x	x	x	x	
g6	x		x	x	

For better readibility: $\mathtt{M} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}\}$

The rectangles of size 1:

{g1}x{b,c,e}, {g2}x{a,c,d}, {g3}x{a,b,c,d}, {g4}x{a,d},
{g5}x{a,b,c,d}, {g6}x{a,c,d}

An example of construction of a concept lattice (2)

The rectangles of size 1:
{g1}x{b,c,e}, {g2}x{a,c,d}, {g3}x{a,b,c,d}, {g4}x{a,d},
{g5}x{a,b,c,d}, {g6}x{a,c,d}

Build the rectangles of size 2 by union of rectangles of size 1:
{g1,g2}x{c}, {g1,g3}x{b,c}, {g1,g5}x{b,c}, {g1,g6}x{c},
{g2,g3}x{a,c,d}, {g2,g4}x{a,d}, {g2,g5}x{a,c,d}, {g2,g6}x{a,c,d},
{g3,g4}x{a,d}, {g3,g5}x{a,b,c,d}, {g3,g6}x{a,c,d},
{g4,g5}x{a,d}, {g4,g6}x{a,d},
{g5,g6}x{a,c,d}, ...

An example of construction of a concept lattice (3)

Remove the non maximal rectangles
(the rectangles with the same intension and a smaller extension):
For example:
{g2}x{a,c,d} because of {g2,g3}x{a,c,d},
{g3}x{a,b,c,d} and {g5}x{a,b,c,d} because of {g3,g5}x{a,b,c,d},
{g4}x{a,d} because of {g4,g5}x{a,d},
{g6}x{a,c,d} because of {g5,g6}x{a,c,d}
...

An example of construction of a concept lattice (4)

Fusion of rectangles with the same intension:

For example:

{g1,g2}x{c} and {g1,g6}x{c} give {g1,g2,g6}x{c}

{g1,g3}x{b,c} and {g1,g5}x{b,c} give {g1,g3,g5}x{b,c}

{g2,g3}x{a,c,d}, {g2,g5}x{a,c,d} and {g2,g6}x{a,c,d} give {g2,g3,g5,g6}x{a,c,d}

this removes the fusion: {g3,g6}x{a,c,d} and {g5,g6}x{a,c,d} give {g3,g5,g6}x{a,c,d}

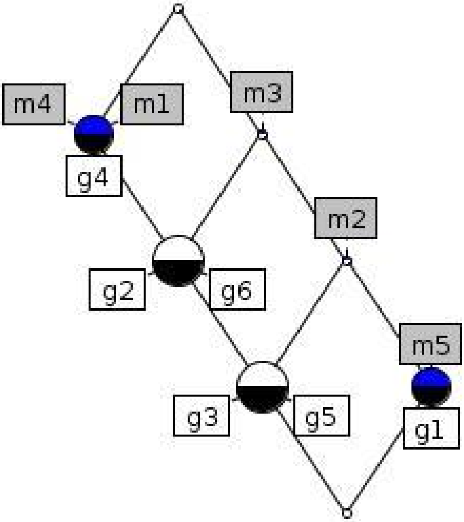
{g2,g4}x{a,d}, {g3,g4}x{a,d}, {g4,g5}x{a,d} and {g4,g6}x{a,d} give {g2,g3,g4,g5,g6}x{a,d}

An example of construction of a concept lattice (5)

Listing the rectangles by size:
Size 1: {g1}x{b,c,e},
Size 2: {g3,g5}x{a,b,c,d},
Size 3: {g1,g3,g5}x{b,c}, {g1,g2,g6}x{c},
Size 4: {g2,g3,g5,g6}x{a,c,d},
Size 5: {g2,g3,g4,g5,g6}x{a,d}

A last construction by union of rectangles is possible:
{g1,g3,g5}x{b,c} and {g1,g2,g6}x{c} give {g1,g2,g3,g5,g6}x{c},
this removes {g1,g2,g6}x{c}

An example of construction of a concept lattice (6)



The Bordat algorithm

Let us consider a set of objects $\mathtt{G}$ and a set of attributes $\mathtt{M}$.

- The Bordat algorithm is able to build the formal concepts and the order between the concepts, thus the whole concept lattice.
- Let us consider a set of objects $\mathtt{G}$ and a set of attributes $\mathtt{M}$.
- Set the bottom concept as the pair $\perp = (\mathtt{M}', \mathtt{M})$ unless there is one object owning all attributes (then $\mathtt{M}' \neq \emptyset$).

The Bordat algorithm

- Let $C_k = (X_k, Y_k)$ be a formal concept.
- The concept $C_{k+1} = (X_{k+1}, Y_{k+1})$ is a subsumer of C_k in the lattice whever it verifies:
- (i) $X_{k+1} = X_k \cup \{x \in G \setminus X_k | f_{C_k}(x) = Y_{k+1}\}$ where f_{C_k} computes the intension of x restricted to $(G \setminus X_k) \times Y_k$ (extensions are growing).
- (ii) $Y_{k+1} \in \mathrm{Max}\{f_{C_k}(x) | x \notin X_k\}$

The Bordat algorithm

G / M	a	b	c	d	e
g1		x	x		x
g2	x		x	x	
g3	x	x	x	x	
g4	x			x	
g5	x	x	x	x	
g6	x		x	x	

- $\mathtt{G} = \{g1, g2, g3, g4, g5, g6\}$ and $\mathtt{M} = \{a, b, c, d, e\}$
- $\perp = (\emptyset, \mathtt{M}) = (\mathtt{X}_0, \mathtt{Y}_0)$

The Bordat algorithm

- $G = \{g1, g2, g3, g4, g5, g6\}$ and $M = \{a, b, c, d, e\}$
- $\perp = (\emptyset, M) = (X_0, Y_0)$
- $X_1 = X_0 \cup \{x \in G \setminus X_0 | f_{C_0}(x) = Y_1\}$
- $Y_1 \in Max\{f_{C_0}(x) | x \notin X_0\}$
- $f_{C_0} : (G \setminus \emptyset) \times \{a, b, c, d, e\}$
- $\{f_{C_0}(x) | x \notin X_0\} = \{f_{C_0}(x) | x \in G\} =$ $\{f_{C_0}(g1), f_{C_0}(g2), f_{C_0}(g3), f_{C_0}(g4), f_{C_0}(g5), f_{C_0}(g6)\} =$ $\{\{b, c, e\}, \{a, c, d\}, \{a, b, c, d\}, \{a, d\}, \{a, b, c, d\}, \{a, c, d\}\}$

The Bordat algorithm

- $\mathtt{Max}\{\{b,c,e\},\{a,c,d\},\{a,b,c,d\},\{a,d\},\{a,b,c,d\},\{a,c,d\}\}$
  $= \{\{b,c,e\},\{a,b,c,d\}\}$
- $Y_1 = \{b,c,e\}$
  $X_1 = X_0 \cup \{x \in G \setminus X_0 | f_{C_0}(x) = Y_1\}$
  $= \{x \in G | f_{C_0}(x) = \{b,c,e\}\} = \{1\}$
- $Y_2 = \{a,b,c,d\}$
  $X_2 = X_0 \cup \{x \in G \setminus X_0 | f_{C_0}(x) = Y_2\}$
  $= \{x \in G | f_{C_0}(x) = \{a,b,c,d\}\} = \{3,5\}$
- $C_1 = (X_1, Y_1) = (\{g1\}, \{b,c,e\})$ and $C_2 = (X_2, Y_2) = (\{g3, g5\}, \{a,b,c,d\})$ are the direct subsumers of (X_0, Y_0) in the lattice.

The Bordat algorithm

- $X_3 = X_1 \cup \{x \in G \setminus X_1 | f_{C_1}(x) = Y_3\}$
- $f_{C_1} : (G \setminus \{g1\}) \times \{b, c, e\}$
- $Y_3 \in Max\{f_{C_1}(x) | x \in \{g2, g3, g4, g5, g6\}\}$
- $Y_3 \in Max\{f_{C_1}(g2), f_{C_1}(g2), f_{C_1}(g3) f_{C_1}(g4), f_{C_1}(g5), f_{C_1}(g6)\}$
  $= Max\{\{c\}, \{b, c\}, \emptyset, \{b, c\}, \{c\}\}$
- Thus $C_3 = (X_3, Y_3) = (\{g1, g3, g5\}, \{b, c\})$

The Bordat algorithm

- $X_4 = X_2 \cup \{x \in G \setminus X_2 | f_{C_2}(x) = Y_4\}$
- $f_{C_2} : (G \setminus \{g3, g5\}) \times \{a, b, c, d\}$
- $Y_4 \in \text{Max}\{f_{C_2}(x) | x \in \{g1, g2, g4, g6\}\}$
- $Y_4 \in \text{Max}\{f_{C_2}(g1), f_{C_2}(g2), f_{C_2}(g4), f_{C_2}(g6)\}$
  $= \text{Max}\{\{b, c\}, \{a, c, d\}, \{a, d\}, \{a, c, d\}\}$
  $= \{\{b, c\}, \{a, c, d\}\}$
- Thus the first subsumer is identical to $C_3 = (X_3, Y_3) = (\{g1, g3, g5\}, \{b, c\})$
- The second subsumer $C_4 = (X_4, Y_4) = (\{g2, g3, g5, g6\}, \{a, c, d\})$

The Bordat algorithm

- $X_5 = X_3 \cup \{x \in G \setminus X_3 | f_{C_3}(x) = Y_5\}$
- $f_{C_3} : (G \setminus \{g1, g3, g5\}) \times \{b, c\}$
- $Y_5 \in Max\{f_{C_3}(x) | x \in \{g2, g4, g6\}\}$
- $Y_5 \in Max\{f_{C_3}(g2), f_{C_3}(g4), f_{C_3}(g6)\}$
  $= Max\{\{c\}, \emptyset, \{c\}\} = \{c\}$
- Thus $C_5 = (X_5, Y_5) = (\{g1, g2, g3, g5, g6\}, \{c\})$

The Bordat algorithm

- $X_6 = X_4 \cup \{x \in G \setminus X_4 | f_{C_4}(x) = Y_6\}$
- $f_{C_4} : (G \setminus \{g2, g3, g5, g6\}) \times \{a, c, d\}$
- $Y_6 \in \mathrm{Max}\{f_{C_4}(x) | x \in \{g1, g4\}$
- $Y_6 \in \mathrm{Max}\{f_{C_4}(g1), f_{C_4}(g4)\}\}$
  $= \mathrm{Max}\{\{c\}, \{d\}\}$
- The first subsumer is identical to
  $C_5 = (X_5, Y_5) = (\{g1, g2, g3, g5, g6\}, \{c\})$
- The second subsumer
  $C_6 = (X_6, Y_6) = (\{g2, g3, g4, g5, g6\}, \{d\})$
- ...

Relational Concept Analysis

- Mohamed Rouane-Hacene, Marianne Huchard, Amedeo Napoli and Petko Valtchev. A proposal for combining Formal Concept Analysis and description Logics for mining relational data, in Proceedings of ICFCA-2007, LNAI 4390, Springer, pages 51–65, 2007.
- Mohamed Rouane-Hacene, Amedeo Napoli, Petko Valtchev, Yannick Toussaint and Rokia Bendaoud. Ontology Learning from Text using Relational Concept Analysis, in International Conference on eTechnologies (MCETECH 08), Montréal, IEEE Computer Society, pages 154–163, 2008.
- Lian Shi, Yannick Toussaint, Amedeo Napoli and Alexandre Blansché. Mining for Reengineering: an Application to Semantic Wikis using Formal and Relational Concept Analysis, Proceedings of ESWC 2011, LNCS 6644, Springer, pages 421–435, 2011.

Introducing Relational Concept Analysis (RCA)

- The objective of RCA is to extend the purpose of FCA for taking into account relations between objects.
- The RCA process relies on the following main points:
  - a relational model based on the entity-relationship model,
  - a conceptual scaling process allowing to represent relations between objects as relational attributes,
  - an iterative process for designing a concept lattice where concept intents include binary and relational attributes.
- The RCA process provides "relational structures" that can be represented as ontology concepts within a knowledge representation formalism such as description logics (DLs).

The RCA data model

- The RCA data model relies on a so-called relational context family denoted by $\mathcal{RCF} = (\mathbf{K}, \mathbf{R})$, where:
- $\mathbf{K}$ is a set of formal contexts $\mathcal{K}_i = (\mathtt{G_i}, \mathtt{M_i}, \mathtt{I_i})$,
- $\mathbf{R}$ is a set of relations $\mathtt{r_k} \subseteq \mathtt{G_i} \times \mathtt{G_j}$, where $\mathtt{G_i}$ and $\mathtt{G_j}$ are sets of objects from the formal contexts $\mathcal{K}_i$ and $\mathcal{K}_j$.
- A relation $\mathtt{r} \subseteq \mathtt{G_i} \times \mathtt{G_j}$ has a domain and a range where:
- $\mathtt{dom(r)} = \mathtt{G_i}$ and $\mathtt{ran(r)} = \mathtt{G_j}$.

An example

- Suppose that we have a context Papers $\times$ Topics where Papers denotes a set of papers –from "a" to "ℓ"– and Topics denotes a set of three attributes, namely "lt" for "lattice theory", "mmi" for "man-machine interface", and "se" for "software engineering".
- There are two relations:
  - cites $\subseteq$ Papers $\times$ Papers indicates that a paper is citing another paper,
  - develops $\subseteq$ Papers $\times$ Papers indicates that a paper is developing another paper.

The initial relational context

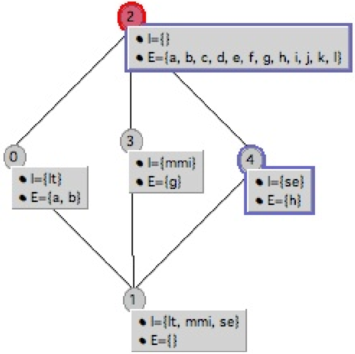
- Relational context:

  $(\mathbf{K}, \mathbf{R}) = (\mathcal{K}_0, \{\texttt{cites}, \texttt{develops}\})$ with $\mathcal{K}_0 = (\texttt{Papers}, \texttt{Topics}, \texttt{I})$

	lt	mmi	se	a	b	g	h	c	d	i	j
a	x										
b	x										
c				x		x					
d					x		x				
e								x			
f									x		
g		x									
h			x								
i				x							
j					x						
k										x	
ℓ											x

The $\mathcal{L}_0$ concept lattice built from formal context $\mathcal{K}_0$

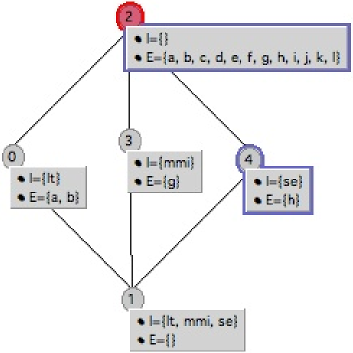
	lt	mmi	se
a	x		
b	x		
c			
d			
e			
f			
g		x	
h			x
i			
j			
k			
ℓ			



Introducing relational scaling

- The first step consists in building an initial concept lattice $\mathcal{L}_0$ from the the initial context $\mathcal{K}_0$ using standard FCA.
- The second step takes into account relations $\mathtt{r}(\mathtt{o_i}, \mathtt{o_j})$ for building a new context $\mathcal{K}_1$:
  - $\mathtt{r}(\mathtt{o_i}, \mathtt{o_j})$ means that object $\mathtt{o_i} \in \mathtt{G_i}$ is related through relation $\mathtt{r}$ with object $\mathtt{o_j} \in \mathtt{G_j}$,
  - then a relational attribute of the form $\exists\mathtt{r}.\mathtt{C_k}$ is associated to object $\mathtt{o_i}$ in $\mathcal{K}_1$, where $\mathtt{C_k}$ is **any** concept instantiating $\mathtt{o_j}$ in $\mathcal{L}_0$.
- When all relations between objects have been examined, the new context $\mathcal{K}_1$ is completed and a new concept lattice $\mathcal{L}_1$ is built accordingly.

Relational scaling in $\mathcal{L}_0$

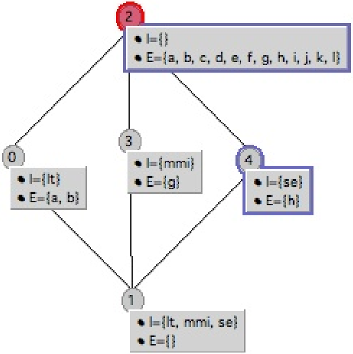


- Object i is in relation with object a through relation `cites`.
- Object a is in the extent of concepts C_0 and C_2 of the initial lattice $\mathcal{L}_0$.
- Thus, object i is given two new relational attributes, namely $\exists$cites:C_0 and $\exists$cites:C_2.
- Object j is in relation with object b through `cites`: by the same way, object j is given two relational attributes $\exists$cites:C_0 and $\exists$cites:C_2.

The relational context $\mathcal{K}_0$

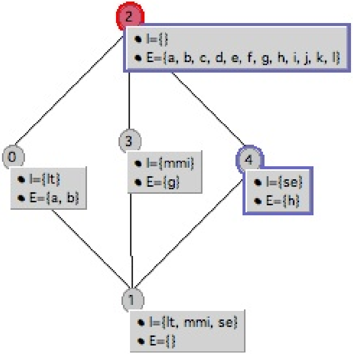
	lt	mmi	se	a	b	g	h	c	d	i	j
a	x										
b	x										
c				x		x					
d					x		x				
e								x			
f									x		
g		x									
h			x								
i				x							
j					x						
k										x	
ℓ											x

Relational scaling in $\mathcal{L}_0$



- Object c is in relation with objects a and g through relation `cites`.
- Object a is in the extent of concepts C_0 and C_2 in $\mathcal{L}_0$ while object g is in the extent of concepts C_3 and C_2 in $\mathcal{L}_0$.
- Thus, object c is given three new relational attributes, namely $\exists$cites:C_0, $\exists$cites:C_2, and $\exists$cites:C_3.

Relational scaling in $\mathcal{L}_0$

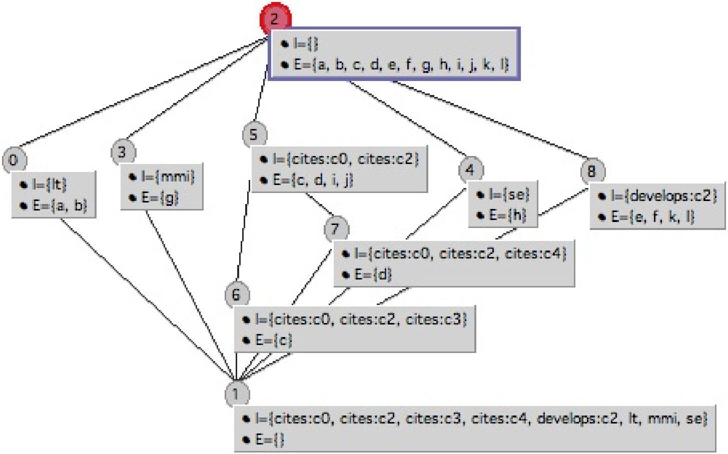


- The same process is applied to `develops`:
- `e` is in relation with `c`, `f` with `d`, `k` with `i`, and ℓ with `j`.
- The four objects `e`, `f`, `k`, and ℓ, are given the relational attribute $\exists\texttt{develops}{:}\texttt{C}_2$.

The new relational context $\mathcal{K}_1$

	lt	mmi	se	cites:c2	cites:c0	cites:c3	cites:c4	develops:c2
a	x							
b	x							
c				x	x	x		
d				x	x		x	
e								x
f								x
g		x						
h			x					
i				x	x			
j				x	x			
k								x
ℓ								x

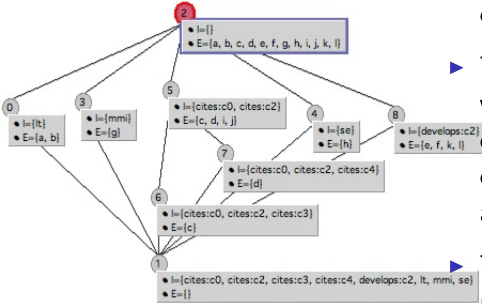
The concept lattice $\mathcal{L}_1$



Three forms of relational attributes

- Existential scaling $\exists \mathtt{r.C}$: $\mathtt{r(o)} \cap \mathtt{Extent(C)} \neq \emptyset$
- Universal scaling $\forall \mathtt{r.C}$: $\mathtt{r(o)} \subseteq \mathtt{Extent(C)}$
- Universal-Existential scaling $\forall\exists$r.C: $\mathtt{r(o)} \subseteq \mathtt{extent(C)}$ and $\mathtt{r(o)} \neq \emptyset$
- With relational scaling, the homogeneity of concept descriptions is kept: all attributes –included relational attributes– are considered as binary attributes.
- Standard algorithms for building concept lattices can be straightforwardly reused.

Relational scaling in $\mathcal{L}_1$

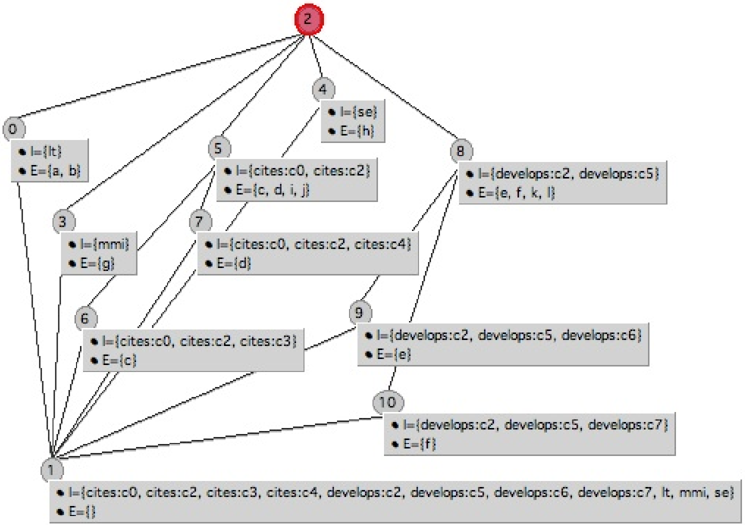


- The process is applied a second time for relation `develops`.
- The object `e` `develops` `c` whose description has changed, i.e. `c` is in the extent of concepts $\mathtt{C}_2$, $\mathtt{C}_5$, and $\mathtt{C}_6$.
- Thus object `e` is given the relational attributes $\exists$`develops`:$\mathtt{C}_5$ and $\exists$`develops`:$\mathtt{C}_6$.

The new relational context $\mathcal{K}_2$

	lt	mmi	se	ct:2	ct:0	ct:3	ct:4	dvl:2	dvl:5	dvl:6	dvl:7
a	x										
b	x										
c				x	x	x					
d				x	x		x				
e								x	x	x	
f								x	x		x
g		x									
h			x								
i				x	x						
j				x	x						
k								x	x		
ℓ								x	x		

The concept lattice $\mathcal{L}_2$



The completion of the RCA process

- Relational scaling is still applied for `cites` and `develops` but the final context and the associated concept lattice are obtained after the second step.
- More generally, relational scaling is applied and either there are new modifications (continuation) or there is no more modification (fix-point).
- The relational scaling process reaches a fix-point and the final context is stationary: no more changes need to be made and the associated final lattice is reached (the relational scaling process terminates).

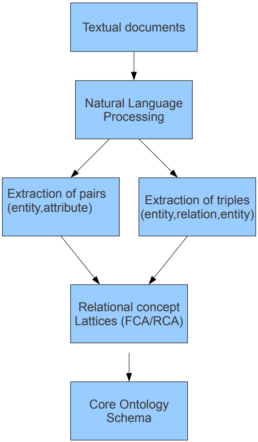
From a relational concept lattice to an ontology schema

- The concepts of the final concept lattice can be represented within a DL formalism such as $\mathcal{ALE}$ for designing an ontology schema supported by the lattice.
- Some problems about knowledge representation are arising for representing binary and relational attributes.
  Binary attributes can be represented as atomic concepts.
- Thanks to the semantics associated with relational scaling and operators, roles can be attached to defined concepts in a "natural" way using a construction such as $\exists \mathtt{r.C}$.

Text Mining

The text mining process

- Text mining implies the manipulation of textual documents in depth, i.e. w.r.t. their structure and their content.
- Text mining can be guided by domain ontologies and supports ontology engineering.
- In turn, ontologies can be used for text annotation, information retrieval, and for guiding the mining of documents w.r.t. their content.



From a relational concept lattice to an ontology schema

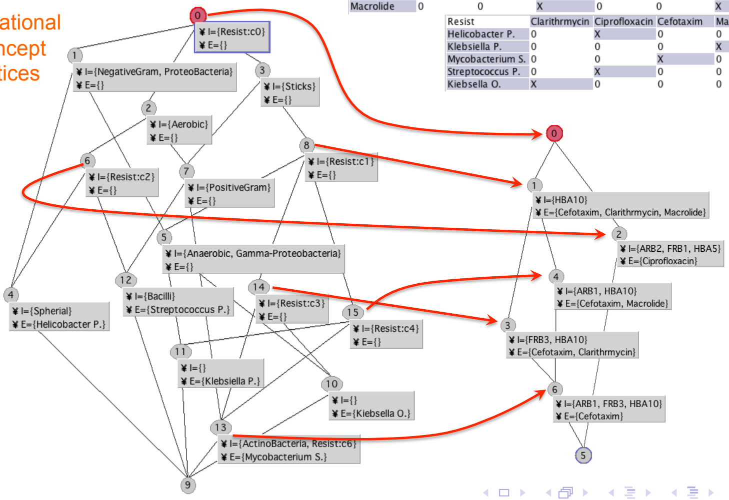
Relational Context Family

Bacteria	ProteoBacteria	Gamma-Proteobacteria	ActinoBacteria	Bacilli	Spherial	Sticks	NegativeGram	PositiveGram	Aerobic	Anaerobic
Helicobacter P.	X	0	0	0	X	0	X	0	X	0
Klebsiella P.	X	X	0	0	0	X	X	0	0	X
Mycobacteriu...	0	0	X	0	0	X	0	X	X	0
Streptococcus P.	0	0	0	X	0	X	0	X	X	0
Kiebsella O.	X	X	0	0	0	X	X	0	0	X

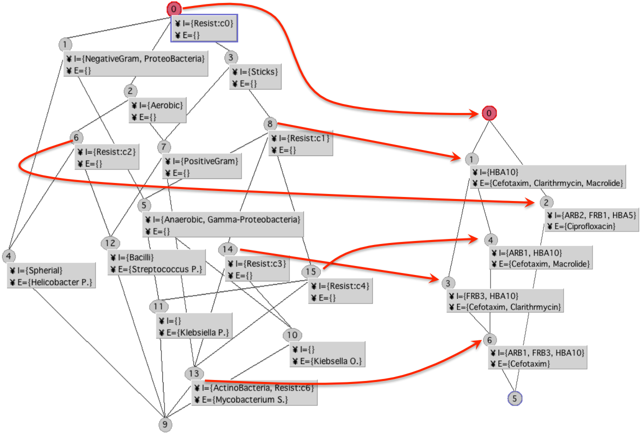
Antibiotics	FRB1	FRB3	ARB1	ARB2	HBA5	HBA10
Clarithrmycin	0	X	0	0	0	X
Ciprofloxacin	X	0	0	X	X	0
Cefotaxim	0	X	X	0	0	X
Macrolide	0	0	X	0	0	X

Resist	Clarithrmycin	Ciprofloxacin	Cefotaxim	Macrolide
Helicobacter P.	0	X	0	0
Klebsiella P.	0	0	0	X
Mycobacterium S.	0	0	X	0
Streptococcus P.	0	X	0	0
Kiebsella O.	X	0	0	0

Relational Concept Lattices



From a relational concept lattice to an ontology schema



Relational concept lattice:

$C_8 : \mathtt{Intent} = \{\mathtt{sticks}, \mathtt{resist} : \mathtt{A0}, \mathtt{resist} : \mathtt{A1}\}$

$C_8 : \mathtt{Extent} = \{\mathtt{klebsiella\_P}, \mathtt{klebsiella\_O}, \mathtt{mycobacterium\_S}\}$

DL expressions:

$C_8 \equiv \exists \mathtt{sticks}.\top \sqcap \exists \mathtt{resist}.\mathtt{A0} \sqcap \exists \mathtt{resist}.\mathtt{A1}$

$C_8(\mathtt{klebsiella\_P}), C_8(\mathtt{klebsiella\_O}), C_8(\mathtt{mycobacterium\_S})$

Pattern Structures

- Mehdi Kaytoue, Sergei O. Kuznetsov, Amedeo Napoli and Sébastien Duplessis. Mining Gene Expression Data with Pattern Structures in Formal Concept Analysis, Information Science, 181(10):1989–2001, 2011.
- Mehdi Kaytoue, Sergei O. Kuznetsov and Amedeo Napoli. Revisiting Numerical Pattern Mining with Formal Concept Analysis, in Proceedings of 22nd International Joint Conference on Artificial Intelligence (IJCAI-11), Barcelona, Spain, 2011.
- Zainab Assaghir, Mehdi Kaytoue, and Amedeo Napoli and Henri Prade. Managing Information Fusion with Formal Concept Analysis, in Proceedings of 7th International Conference on Modeling Decisions for Artificial Intelligence (MDAI 2010), LNCS 6408, Springer, pages 104–115, 2010.

Handling numerical data with FCA?

Conceptual scaling (discretization or binarization)

An object has an attribute if its value lies in a predefined interval

	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

	$m_1, [4,5]$	$m_2, [4,7]$	$m_3, [5,6]$
g_1	×	×	×
g_2			
g_3	×		×
g_4	×		
g_5	×		×

Different scalings: different interpretations of the data

General problem

How to directly build a concept lattice from numerical data?

How to handle complex 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

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

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

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

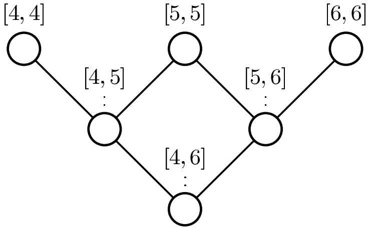
$$d^{\square} = \{g \in G | d \sqsubseteq \delta(g)\} \qquad \text{for } d \in (D, \sqcap)$$

Interval Pattern Structure

- A meet-semi-lattice for intervals $(\mathtt{D}, \sqcap)$ where $\mathtt{D}$ is a set of intervals,
- a possible choice for the meet operator is the convexification of intervals:

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

$$\begin{array}{rcl} [a_1, b_1] \sqsubseteq [a_2, b_2] & \iff & [a_2, b_2] \subseteq [a_1, b_1] \\ [4,5] \sqsubseteq [5,5] & \iff & [5,5] \subseteq [4,5] \end{array}$$



Interval Pattern Structure

- An interval pattern p is an n-dimensional vector of intervals: $\mathtt{p} = \langle[\mathtt{a_i}, \mathtt{b_i}]\rangle_{\mathtt{i}\in[1,\mathtt{n}]}$
- Operation $\sqcap$ and order of interval patterns:
  Given interval patterns $\mathtt{p} = \langle[\mathtt{a_i}, \mathtt{b_i}]\rangle_{\mathtt{i}\in[1,\mathtt{n}]}$ and $\mathtt{q} = \langle[\mathtt{c_i}, \mathtt{d_i}]\rangle_{\mathtt{i}\in[1,\mathtt{n}]}$:
- $$p \sqcap q = \langle[a_i, b_i]\rangle_{i\in[1,n]} \sqcap \langle[c_i, d_i]\rangle_{i\in[1,n]}$$
  $$p \sqcap q = \langle[a_i, b_i] \sqcap [c_i, d_i]\rangle_{i\in[1,n]}$$
- $$p \sqcap q = p \Leftrightarrow p \sqsubseteq q$$
  $$p \sqsubseteq q \Leftrightarrow [a_i, b_i] \sqsubseteq [c_i, d_i], \forall i \in [1, n]$$

Interval pattern structures based on convexification

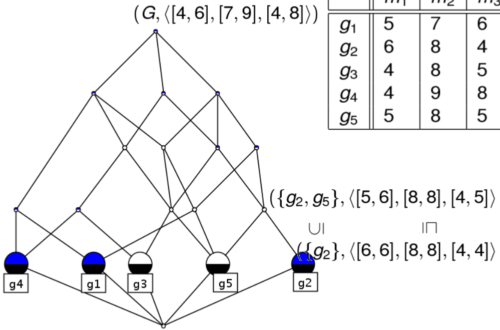
	m_1	m_2	m_3
g_1	5	7	6
g_2	6	8	4
g_3	4	8	5
g_4	4	9	8
g_5	5	8	5

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

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

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

Interval pattern concept lattice



	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

- Highest concepts: largest extents and largest intervals (smallest intents)
- Lowest concepts: smallest extents and smallest intervals (largest intents)
- Problem: efficient pattern mining.

Links with conceptual scaling

Interordinal scaling [Ganter & Wille]

- A scale to encode intervals of attribute values

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

- Equivalent concept lattice
- Example

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

Why should we use pattern structures as we have scaling?

Processing a pattern structure is more efficient

Interval pattern search space

Counting all possible interval patterns with interordinal scaling

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

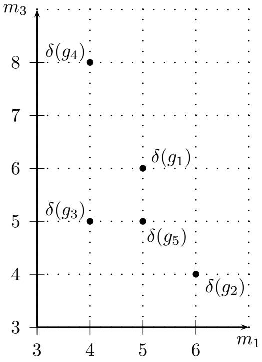
Questions on interval pattern mining

- What are the links between numerical pattern structures and pattern mining?
- How can we reuse (good) ideas from pattern mining, i.e. closed patterns, generators and equivalence classes, in the framework of pattern structures?

Semantics for interval patterns

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

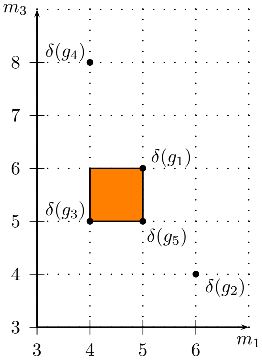


Semantics for interval patterns

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

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

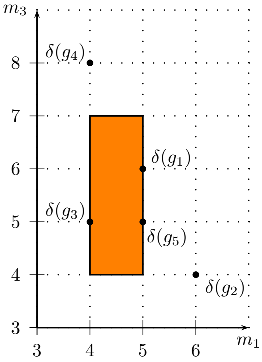


Semantics for interval patterns

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

$\langle[4,5],[5,6]\rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle[4,5],[4,7]\rangle^{\square} = \{g_1, g_3, g_5\}$

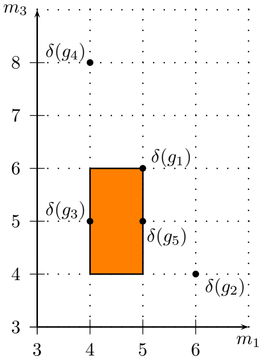


Semantics for interval patterns

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

$$\langle [4,5],[5,6] \rangle^{\Box} = \{g_1, g_3, g_5\}$$
$$\langle [4,5],[4,7] \rangle^{\Box} = \{g_1, g_3, g_5\}$$
$$\langle [4,5],[4,6] \rangle^{\Box} = \{g_1, g_3, g_5\}$$

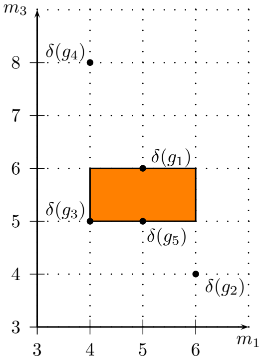


Semantics for interval patterns

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

$\langle[4,5],[5,6]\rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle[4,5],[4,7]\rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle[4,5],[4,6]\rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle[4,6],[5,6]\rangle^{\square} = \{g_1, g_3, g_5\}$

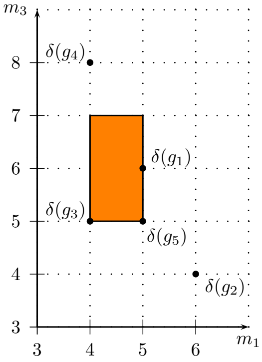


Semantics for interval patterns

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

$\langle [4,5], [5,6] \rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle [4,5], [4,7] \rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle [4,5], [4,6] \rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle [4,6], [5,6] \rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle [4,5], [5,7] \rangle^{\square} = \{g_1, g_3, g_5\}$

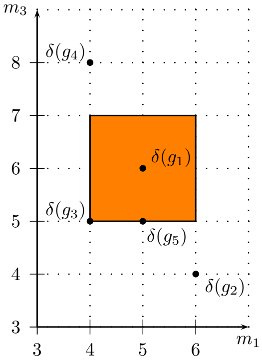


Semantics for interval patterns

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

$\langle[4,5],[5,6]\rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle[4,5],[4,7]\rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle[4,5],[4,6]\rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle[4,6],[5,6]\rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle[4,5],[5,7]\rangle^{\square} = \{g_1, g_3, g_5\}$
$\langle[4,6],[5,7]\rangle^{\square} = \{g_1, g_3, g_5\}$



A condensed representation

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

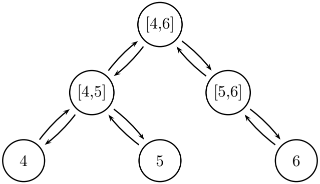
$$[d] = \{c | c \cong d\}$$

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

In the example: 360 patterns ; 18 closed patterns ; 44 generators

Algorithms & experiments

Algorithms: MintIntChange, MinIntChangeG[t|h]

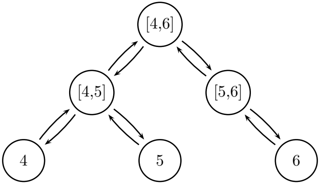


Principle with an example

1. Start from the most general interval pattern: $\langle[4,6],[7,9],[4,8]\rangle$
2. Apply next minimal change following a canonical order $c = \langle[4,5],[7,9],[4,8]\rangle$
3. Apply closure operator $c^{\square\square} = \langle[4,5],[7,9],[5,8]\rangle$
4. If canonicity test fails: backtrack (in the depth first traversal)
5. Otherwise go to 2. with $c^{\square\square} = \langle[4,5],[7,9],[5,8]\rangle$

Algorithms & experiments

Algorithms: MintIntChange, MinIntChangeG[t|h]



Experiments

- Mining several datasets from Bilkent University Repository
- Compression rate varies between 10^7 and 10^9
- Interordinal scaling
  - not efficient even with best algorithms (e.g. LCMv2)
  - redundancy problem discarding its use for generator extraction

Discussion

- Potential applications:
  - data privacy and k-anonymisation
  - k-box problem in computational geometry
  - quantitative association rule mining
  - data summarization
- Extension: focus on generator extraction
- Problems:
  - compression is not enough when considering very large data set
  - numerical data are noisy: this calls for fault-tolerant condensed representations

Conclusion

- FCA is a well-founded mathematical theory equipped with efficient algorithmic tools.
- FCA is a polymorphic process and addresses problems ranging from knowledge discovery to knowledge representation and reasoning, and pattern recognition as well.
- FCA is rather mature and times are there for important variations, e.g. RCA and pattern structures (intervals and graphs).
- There is still room for many improvements, especially in dealing with trees and graphs, in taking into account domain knowledge, similarity, and in combining FCA with numerical processes.

Tools for building and visualizing concept lattices

- The Conexp program: http://sourceforge.net/projects/conexp
- The Galicia Platform: http://www.iro.umontreal.ca/~galicia/
- The Toscana platform: http://tockit.sourceforge.net/toscanaj/index.html
- The Formal Concept Analysis Homepage: http://www.upriss.org.uk/fca/fca.html

Elements of bibliography on concept lattices

- Marc Barbut and Bernard Monjardet, Ordre et classification, Hachette, 1970.
- Introduction to Formal Concept Analysis. Radim Belohlavek, Palacky University, Olomouc, `http://phoenix.inf.upol.cz/esf/ucebni/formal.pdf`
- Claudio Carpineto and Giovanni Romano, Concept Data Analysis: Theory and Applications, John Wiley & Sons, 2004.
- Finite Ordered Sets: Concepts, Results and Uses. Nathalie Caspard, Bruno Leclerc and Bernard Monjardet, Cambridge University Press, 2012.
- Bernhard Ganter and Rudolph Wille, Formal Concept Analysis, Springer, 1999.
- Applied Lattice Theory: Formal Concept Analysis. Bernhard Ganter and Rudolf Wille, `www.math.tu-dresden.de/~ganter/psfiles/concept.ps`
- Formal Concept Analysis, Foundations and Applications. Bernhard Ganter, Gerd Stumme and Rudolf Wille editors, Lecture Notes in Computer Science 3626, Springer, 2005.