Distributive Concept Exploration – A Knowledge Acquisition Tool in Formal Concept Analysis

Gerd Stumme

Technische Universität Darmstadt, Fachbereich Mathematik
Schloßgartenstr. 7, D–64289 Darmstadt, stumme@mathematik.tu-darmstadt.de

1 Introduction

Formal Concept Analysis ([9], [1]) provides a mathematical model of the concept 'concept' which is used in data analysis for examining conceptual hierarchies in data tables. If these data tables are too large to be completely given, then the conceptual structure has to be determined in an interactive knowledge acquisition process from an expert of the domain. *Exploration tools* suggest, starting with the concepts to be examined, hierarchical relationships. The expert is asked either to confirm them or to provide typical counter-examples. The result of the exploration is a lattice that is generated by adding all largest common subconcepts and/or least common superconcepts.

In [12] and [4], an overview over different exploration tools in Formal Concept Analysis is given. While *Attribute Exploration* considers largest common subconcepts only and *Object Exploration* least common superconcepts only ([1]), *Concept Exploration* treats largest common subconcepts (*infima*) and least common superconcepts (*suprema*) equally ([3], [6], [7], [11], [12]). It determines the lattice of all combinations of infima and suprema of the starting concepts (which are also called the *basic concepts*).

A big problem of Concept Exploration is the fact that the resulting lattice (and the exploration dialogue) may be infinite. Even only three concepts can generate an infinite lattice! In practice however, this case does not appear. We can overcome this principal difficulty if general knowledge about the domain provides more information about the structure of the intended lattice. If we know in advance that the lattice is distributive, then the finiteness of the result is ensured.

Which additional assumptions imply the distributivity of the lattice? This is especially the case, if we know that the attributes which generate the conceptual hierarchy are closed under disjunction. One interesting application is within *Description Logics* where disjunction is usually used as constructor. For logics having a complete subsumption algorithm, this algorithm can be considered as 'expert' for the exploration procedure. By combining both algorithms, one obtains a completely automatic knowledge acqusition tool ([5], [7]).

With distributivity of the resulting lattice known in advance, we can use its much stronger structure in the algorithm. This is the underlying idea of *Distributive Concept Exploration*. In particular, Distributive Concept Exploration uses the tensor product of lattices ([10]), which is the co-product in the category of completely distributive complete lattices. This approach cannot be adapted to Concept Exploration, since there is no co-product in the category of complete lattices.

During the exploration the user is asked questions of the form "Is s a subconcept of t?", where s and t are lattice terms built with the basic concepts. If the user replies "No", he must justify his answer by an object belonging to s and an attribute belonging to t such that the object does not have the attribute. The result of the exploration is the concept lattice of all combinations of infima and suprema of the basic concepts, together with a list of objects and attributes which separate the concepts. The algorithm is implemented by B. Groh.

In the next section the basic notions of Formal Concept Analysis are introduced. The algorithm of Distributive Concept Exploration is described in Section 3 and illustrated by an example in Section 4. Because of space limitation, the mathematical part is quite condensed. In order to get an idea of the exploration procedure, the reader may first read the next section until the example and then have a look at Section 4 before going in the details in Section 3.

2 Formal Concept Analysis

Tensor products of lattices and congruence relations on lattices are the essential constructions for Cistributive Concept Exploration. Both can adequately be described in terms of Formal Concept Analysis. Formal Concept Analysis (cf. [9], [1]) is a mathematical approach which reflects the philosophical understanding of concepts as units of thought consisting of two parts: the extension containing all objects which belong to the concept and the intension containing the attributes shared by all those objects (cf. [8]). In Formal Concept Analysis this is modeled by *formal concepts* that are derived from a *formal context*. We briefly recall some basic definitions:

A *(formal) context* is a triple $\mathbb{K} := (G, M, I)$ where G and M are sets and I is a relation between G and M. The elements of G and M are called *objects* and *attributes*, respectively, and gIm is read *"the object g has the attribute m"*. For $A \subseteq G$ and $B \subseteq M$ we define $A' := \{m{\in}M \mid \forall g{\in}A : gIm\}$ and dually $B' := \{g{\in}G \mid \forall m{\in}B : gIm\}$. Now a *(formal) concept* is a pair (A, B) with $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. The set A is called the *extent* and the set B the *intent* of the concept. The hierarchical subconcept–superconcept–relation of concepts is formalized by $(A, B) \leq (C, D) :\iff A \subseteq C$ $(\iff B \supseteq D)$. The set of all concepts of the context $\mathbb{K}$ together with this order relation is a complete lattice that is called the *concept lattice* of $\mathbb{K}$ and is denoted by $\underline{\mathfrak{B}}(\mathbb{K})$. Each complete lattice can be viewed as a concept lattice: A complete lattice L is isomorphic to the concept lattice $\underline{\mathfrak{B}}(L, L, \leq)$.

Example. Figure 1 shows a formal context about the potential of gaseous pollutants. Gases are objects, and possible perils are attributes. In the line diagram of the concept lattice, we label, for each object $g \in G$, its *object concept* $\gamma g := (\{g\}'', \{g\}')$ with the name of the object and, for each attribute $m \in M$,

	greenhouseeffect	ozone depletion potential	acidification	nutrification	human toxicity potential	carcinogenity/toxicity	ecotoxicity
NOx	×		×	×	×		×
N2O	×	×			×		
NH3			×	×		×	×
SO2	×		×		×		×
CO	×				×		×
CO2	×						

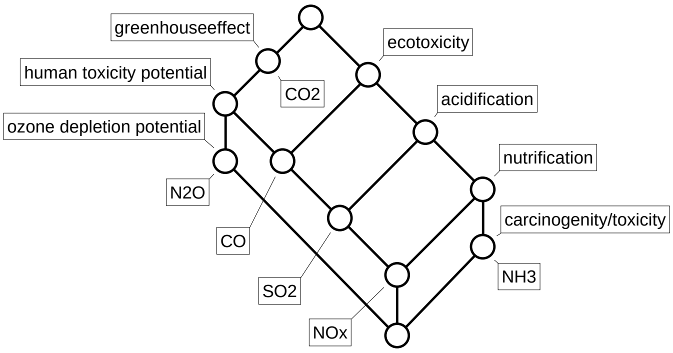


Fig. 1. Formal context and concept lattice of gaseous pollutants

its *attribute concept* $\mu m := (\{m\}', \{m\}'')$ with the name of the attribute. This labeling allows us to determine for each concept its extent and its intent: The extent [intent] of a concept contains all objects [attributes] whose object concepts [attribute concepts] can be reached from the concept on a descending [ascending] path of straight line segments. For instance, the concept labeled with CO has $\{$CO, SO$_2$, NO$_x\}$ as extent, and $\{$human toxicity potential, greenhouse effect, ecotoxicity$\}$ as intent. The concept lattice combines the view of different pollution scenarios with the influence of individual polluants. Such an integrated view can be of interest for the planning of chimneys for plants generating specific polluants. Generally spoken, Formal Concept Analysis treats intensional and extensional aspects equally and in an integrative way.

The *tensor product* of two complete lattices L_1 and L_2 is defined to be the concept lattice $L_1 \otimes L_2 := \underline{\mathfrak{B}}(L_1 \times L_2, L_1 \times L_2, \nabla)$ with $(x_1, x_2)\nabla(y_1, y_2) :\Longleftrightarrow (x_1 \leq y_1$ or $x_2 \leq y_2)$. R. Wille showed in [10] that the tensor product is the co-product in the category of completely distributive complete lattices with complete homomorphisms. Hence the tensor product of L_1 and L_2 is, in a certain sense, the largest complete distributive lattice that can be generated by L_1 and L_2.

We define the *direct product* of two contexts $\mathbb{K}_1 := (G_1, M_1, I_1)$ and $\mathbb{K}_2 := (G_2, M_2, I_2)$ to be the context $\mathbb{K}_1 \times \mathbb{K}_2 := (G_1 \times G_2, M_1 \times M_2, \nabla)$ with the incidence $(g_1, g_2)\nabla(m_1, m_2) :\Longleftrightarrow ((g_1, m_1) \in I_1$ or $(g_2, m_2) \in I_2)$. The tensor product of two concept lattices is (up to isomorphism) just the concept lattice of the direct product of their contexts: We have $\underline{\mathfrak{B}}(\mathbb{K}_1) \otimes \underline{\mathfrak{B}}(\mathbb{K}_2) \cong \underline{\mathfrak{B}}(\mathbb{K}_1 \times \mathbb{K}_2)$.

A context is called *reduced* if each object concept is $\bigvee$-irreducible (i. e., is not supremum of smaller elements) and each attribute concept is $\bigwedge$-irreducible (i. e., is not infimum of larger elements). For a $\bigvee$–irreducible ($\bigwedge$–irreducible) element x of a finite lattice we write x_* (x^*) for its unique lower (upper) cover. If $\mathbb{K}_1$ and $\mathbb{K}_2$ are reduced, then $\mathbb{K}_1 \times \mathbb{K}_2$ is also reduced (cf. [10]).

Congruence relations of complete lattices appear in a quite natural way in Formal Concept Analysis. For finite concept lattices they can always be described by *compatible subcontexts*: A context (H, N, J) is called a *subcontext* of a context

(G, M, I) if $H \subseteq G$, $N \subseteq M$ and $J = I \cap (H \times N)$. It is called *compatible* if for each concept (A, B) of (G, M, I) the pair $(A \cap H, B \cap N)$ is also a concept of the subcontext. Factorizing a concept lattice is equivalent to providing a compatible subcontext, i.e. to deleting suitable rows and columns in the context. The rows and columns that have to be deleted can be described by using the relation $\swarrow\!\!\!\!\nearrow$:

For $g \in G$ and $m \in M$ we define $g \swarrow\!\!\!\!\nearrow m$ if $g \not I m$, $g' \subset h'$ implies hIm for all $h \in G$, and $m' \subset n'$ implies gIn for all $n \in M$. For two elements u and v of a complete lattice L, we write $u \swarrow\!\!\!\!\nearrow v$, if u is maximal in $\{x {\in} L \mid x {\not\geq} v\}$ and v is minimal in $\{x {\in} L \mid x {\not\leq} u\}$.

For two elements u and v of a complete lattice, $u \swarrow\!\!\!\!\nearrow v$ implies that u is $\bigvee$–irreducible and v is $\bigwedge$–irreducible and that $u {\not\leq} v$, $u_* {\leq} v$, and $u {\leq} v^*$ hold. It should not be confusing that we use $\swarrow\!\!\!\!\nearrow$ at the same time as a relation between elements of a lattice and between objects and attributes of a context because $g \swarrow\!\!\!\!\nearrow m$ in $\mathbb{K}$ is equivalent to $\gamma g \swarrow\!\!\!\!\nearrow \mu m$ in the concept lattice $\underline{\mathfrak{B}}(\mathbb{K})$.

A context is called *distributive* if its concept lattice is distributive. All the contexts needed for Distributive Concept Exploration are distributive reduced finite contexts. In these contexts the $\swarrow\!\!\!\!\nearrow$-relation is a bijection between the set of objects and the set of attributes. According to [1], in a distributive reduced finite context the compatible subcontexts are exactly those of the form $(H, N, I \cap (H \times N))$ where for each $m \in N$ exists $g \in H$ s.t. $g \swarrow\!\!\!\!\nearrow m$. The following theorem describes the correspondence between compatible subcontexts and congruence relations. It is a consequence of Lemmata 34 and 36 in [1].

Theorem 1. *Let (G, M, I) be a distributive reduced finite context, $g \in G$ and $m \in M$ with $g \swarrow\!\!\!\!\nearrow m$. Then the kernel of the complete homomorphism*

$$\pi \colon \underline{\mathfrak{B}}(G, M, I) \to \underline{\mathfrak{B}}(G \setminus \{g\}, M \setminus \{m\}, I \setminus (\{g\} \times M \cup G \times \{m\}))$$

with $(A, B) \mapsto (A \setminus \{g\}, B \setminus \{m\})$ is the congruence relation on $\underline{\mathfrak{B}}(G, M, I)$ that is generated by forcing $\gamma g \leq \mu m$.

3 Distributive Concept Exploration

Let $\mathfrak{b}_1, \mathfrak{b}_2, \ldots, \mathfrak{b}_n$ be names of the concepts the user wants to explore. They are called *basic concepts*. We assume that they generate (by taking greatest common subconcepts and least common superconcepts) a (yet unknown) distributive lattice L_n. Distributive Concept Exploration determines the lattice L_n together with a list of objects and attributes which are separating different concepts.

The lattice L_n is isomorphic to a quotient lattice $\mathrm{FBD}(\mathfrak{b}_1, \ldots, \mathfrak{b}_n)/\Theta$ of the free bounded distributive lattice generated by the basic concepts. The congruence relation Θ reflects the answers given by the user. We use the fact that $\mathrm{FBD}(\mathfrak{b}_1, \ldots, \mathfrak{b}_i) \cong \mathrm{FBD}(\mathfrak{b}_1, \ldots, \mathfrak{b}_{i-1}) \otimes \mathrm{FBD}(\mathfrak{b}_i)$ for splitting the determination of Θ into smaller parts: For $i = 0, \ldots, n$, the exploration algorithm subsequently determines the lattice L_i that is completely generated by the basic concepts $\mathfrak{b}_1, \ldots, \mathfrak{b}_i$ with respect to their hierarchical relationships. The lattice L_i is obtained from L_{i-1} by $L_i \cong (L_{i-1} {\otimes} \mathrm{FBD}(\mathfrak{b}_i))/\Theta_i$, where Θ_i reflects the hierarchical relationship between $\mathfrak{b}_i$ and the elements of L_{i-1}. The result of the exploration is then given by the lattice L_n.

For each $i \in \{0, \ldots, n\}$, the lattice L_i will be determined in two steps: First $\widetilde{L}_i$, the tensor product of L_{i-1} with $\mathrm{FBD}(\mathfrak{b}_i)$ (which is the three element chain $\bot < \mathfrak{b}_i < \top$), is calculated. Then the user is asked questions of the kind "Is s a subconcept of t?" with s and t being lattice terms built with $\mathfrak{b}_1, \ldots, \mathfrak{b}_i$. The congruence relation Θ_i on $\widetilde{L}_i$ is deduced from the answers given by the user. The factorization of $\widetilde{L}_i$ by the congruence relation yields the lattice L_i.

In the algorithm, the lattices L_i are *represented* by reduced contexts $\mathbb{K}_i := (G_i, M_i, I_i)$, i.e. the lattice L_i is isomorphic to the concept lattice $\underline{\mathfrak{B}}(\mathbb{K}_i)$. As this context is the result of a repeated use of the direct product of contexts, its objects and attributes are tuples. They are of the form $\vec{x} := (x_0, \ldots, x_i) \in G_i$ with $x_0 = \top$ and $x_k \in \{\top, \mathfrak{b}_k\}$ for $k = 1, \ldots, n$ and $\vec{y} := (y_0, \ldots, y_i) \in M_i$ with $y_0 = \bot$ and $y_k \in \{\bot, \mathfrak{b}_k\}$ for $k = 1, \ldots, n$. The incidence $\vec{x} I_i \vec{y}$ represents the inequality $\bigwedge \vec{x} \leq \bigvee \vec{y}$ with $\bigwedge \vec{x} := \bigwedge_{k=0}^{i} x_i$ and $\bigvee \vec{y} := \bigvee_{k=0}^{i} y_i$.

As mentioned above, the lattice $\widetilde{L}_i$ has to be calculated as intermediate step in the determination of the lattice L_i. This tensor product of L_{i-1} with the chain $\bot < \mathfrak{b}_i < \top$ will be represented by the context $\widetilde{\mathbb{K}}_i := (\widetilde{G}_i, \widetilde{M}_i, \widetilde{I}_i)$ being the direct product of $\mathbb{K}_{i-1}$ with the context $(\{\mathfrak{b}_i, \top\}, \{\bot, \mathfrak{b}_i\}, \{(\mathfrak{b}_i, \mathfrak{b}_i)\})$. The context $\mathbb{K}_i$ will then be derived from $\widetilde{\mathbb{K}}_i$ by deleting suitable rows and columns. This corresponds to finding a suitable congruence relation on the tensor product. Theorem 1 indicates the questions needed for determining these rows and columns: For all $\vec{x} \in \widetilde{G}_i$ and $\vec{y} \in \widetilde{M}_i$ with $\vec{x} \swarrow\!\!\!\nearrow \vec{y}$ the user is asked: "Is the infimum of $\vec{x}$ a subconcept of the supremum of $\vec{y}$?" This question is equivalent to "Does each object belonging to all concepts $x_0, \ldots, x_i$ belong to at least one of the concepts $y_0, \ldots, y_i$?". If the user agrees to the question, the object $\vec{x}$ and the attribute $\vec{y}$ will be deleted, otherwise they will be kept in G_i and M_i, respectively.

Observe that the $\swarrow\!\!\!\nearrow$-relation is inherited and can thus easily be calculated: For $\vec{x} \swarrow\!\!\!\nearrow \vec{y}$ in $\mathbb{K}_{i-1}$, we have $(\vec{x}, \top) \swarrow\!\!\!\nearrow (\vec{y}, \mathfrak{b}_i)$ and $(\vec{x}, \mathfrak{b}_i) \swarrow\!\!\!\nearrow (\vec{y}, \bot)$ in $\widetilde{\mathbb{K}}_i$. Deleting corresponding rows and columns does not change the $\swarrow\!\!\!\nearrow$-relation.

The algorithm starts with the determination of L_0 out of the two element lattice $\widetilde{L}_0 := \mathrm{FBD}(\emptyset) = (\bot{<}\top)$. The elements $\bot$ and $\top$ are the concepts *nothing* and *everything (in our field of interest)*. The lattice $\widetilde{L}_0$ is represented by the context $\widetilde{\mathbb{K}}_0 := (\{\top\}, \{\bot\}, \emptyset)$. As we have $\top \swarrow\!\!\!\nearrow \bot$ in $\widetilde{\mathbb{K}}_0$, the first question in each exploration is "Is $\top$ (*everything*) a subconcept of $\bot$ (*nothing*)?" Usually, this will be denied. If however the user agrees, the exploration is terminated because he obtains $\mathbb{K}_0 = (\emptyset, \emptyset, \emptyset)$ which is the absorbing element for the direct product of contexts. Its concept lattice $\underline{\mathfrak{B}}(\emptyset, \emptyset, \emptyset)$ is the one element lattice which is the absorbing element for the tensor product of lattices.

Next we introduce *separating pairs*. They are justifications for the claim that two concepts are different. More precisely, they justify that one of the concepts is not a subconcept of the other. For two concepts $\mathfrak{a}$ and $\mathfrak{b}$ with $\mathfrak{a}$ not a subconcept of $\mathfrak{b}$, a pair (g, m) is called a *separating pair* if g is an object of the concept $\mathfrak{a}$ and m is an attribute of the concept $\mathfrak{b}$ such that g does not have the attribute m.

The algorithm computes for each L_i with $i = 0, \ldots, n$ a minimal list of pairs of objects and attributes, such that for two concepts $\mathfrak{a}$ and $\mathfrak{b}$ of L_i with $\mathfrak{a} \not\leq \mathfrak{b}$ there is at least one pair in this list which is a separating pair for $\mathfrak{a}$ and $\mathfrak{b}$. It is sufficient to have a list of separating pairs for elements $\mathfrak{c}$ and $\mathfrak{d}$ of L_i with $\mathfrak{c} \swarrow\!\!\!\nearrow \mathfrak{d}$, as for two elements $\mathfrak{a}$ and $\mathfrak{b}$ of L_i with $\mathfrak{a} \not\leq \mathfrak{b}$ there always exist such $\mathfrak{c}$ and $\mathfrak{d}$ with $\mathfrak{c} \leq \mathfrak{a}$ and $\mathfrak{b} \leq \mathfrak{d}$, because L_i is finite and distributive. The separating pair for $\mathfrak{c}$ and $\mathfrak{d}$ is also a separating pair for $\mathfrak{a}$ and $\mathfrak{b}$. On the other hand there must be different separating pairs for different $\mathfrak{c} \swarrow\!\!\!\nearrow \mathfrak{d}$, so that in fact this list is minimal.

During the exploration, the user is asked for separating pairs: Whenever he denies the question "Is the infimum of $\vec{x}$ a subconcept of the supremum of $\vec{y}$?", he is prompted for a separating pair for $\bigwedge \vec{x}$ and $\bigvee \vec{y}$. The pair will be denoted by $(\mathbf{g}_i(\vec{x}), \mathbf{m}_i(\vec{y}))$. Thus we obtain two mappings: $\mathbf{g}_i$ maps from G_i to the set of objects of the separating pairs, and $\mathbf{m}_i$ maps from M_i to the attributes. These mappings indicate that the object $\mathbf{g}_i(\vec{x})$ belongs to the concept $\bigwedge \vec{x}$, and that the attribute $\mathbf{m}_i(\vec{y})$ belongs to the concept $\bigvee \vec{y}$. Because of $\bigwedge \vec{x} \swarrow\!\!\!\nearrow \bigvee \vec{y}$, we know that $\mathbf{g}_i(\vec{x})$ and $\mathbf{m}_i(\vec{y})$ form a separating pair. The mappings $\mathbf{g}_i$ and $\mathbf{m}_i$ however do not indicate whether an object or attribute does *not* belong to a concept. This information cannot be deduced from the answers given by the expert during the exploration dialogue. I. e., because the expert is not asked how objects and attributes of different separating pairs are related.

Unfortunately, $\bigwedge \vec{x} \swarrow\!\!\!\nearrow \bigvee \vec{y}$ in L_i does not imply $\bigwedge \vec{x} \swarrow\!\!\!\nearrow \bigvee \vec{y}$ in L_{i+1}. This means that the separating pair $(\mathbf{g}_i(\vec{x}), \mathbf{m}_i(\vec{y}))$ will in general not remain in the minimal list for L_{i+1}: If neither $\mathbf{g}_i(\vec{x})$ nor $\mathbf{m}_i(\vec{y})$ belong to $\mathfrak{b}_{i+1}$, then there is no $\mathfrak{c} \swarrow\!\!\!\nearrow \mathfrak{d}$ in L_{i+1} separated by this pair. However it can be used to find new separating pairs for the minimal list: $\mathbf{g}_i(\vec{x})$ might appear in a separating pair for $\bigwedge(\vec{x}, \top)$ and $\bigvee(\vec{y}, \mathfrak{b}_{i+1})$ and $\mathbf{m}_i(\vec{y})$ might appear in a separating pair for $\bigwedge(\vec{x}, \mathfrak{b}_{i+1})$ and $\bigvee(\vec{y}, \bot)$. If the object $\mathbf{g}_i(\vec{x})$ belongs to the concept $\mathfrak{b}_{i+1}$ and the attribute $\mathbf{m}_i(\vec{y})$ does not, then they are a separating pair for $\bigwedge(\vec{x}, \mathfrak{b}_{i+1}) \swarrow\!\!\!\nearrow \bigvee(\vec{y}, \bot)$ in L_{i+1} and remain therefore in the minimal list. If the object $\mathbf{g}_i(\vec{x})$ does not belong to the concept $\mathfrak{b}_{i+1}$ and the attribute $\mathbf{m}_i(\vec{y})$ does, then they are a separating pair for $\bigwedge(\vec{x}, \top) \swarrow\!\!\!\nearrow \bigvee(\vec{y}, \mathfrak{b}_{i+1})$ in L_{i+1} and remain in the list. Because the object $\mathbf{g}_i(\vec{x})$ does not have the attribute $\mathbf{m}_i(\vec{y})$, it is not possible that both belong to the concept $\mathfrak{b}_{i+1}$. This justifies the following definition:

$$
\begin{aligned}
\widetilde{\mathbf{g}}_{i+1}(\vec{x}, \mathfrak{b}_{i+1}) &:= \begin{cases} \mathbf{g}_i(\vec{x}) & \text{if } \mathbf{g}_i(\vec{x}) \text{ belongs to } \mathfrak{b}_{i+1} \\ \text{undefined} & \text{else} \end{cases} \\
\widetilde{\mathbf{g}}_{i+1}(\vec{x}, \top) &:= \begin{cases} \mathbf{g}_i(\vec{x}) & \text{if } \mathbf{g}_i(\vec{x}) \text{ does not belong to } \mathfrak{b}_{i+1} \\ \text{undefined} & \text{else} \end{cases} \\
\widetilde{\mathbf{m}}_{i+1}(\vec{y}, \mathfrak{b}_{i+1}) &:= \begin{cases} \mathbf{m}_i(\vec{y}) & \text{if } \mathbf{m}_i(\vec{y}) \text{ belongs to } \mathfrak{b}_{i+1} \\ \text{undefined} & \text{else} \end{cases} \\
\widetilde{\mathbf{m}}_{i+1}(\vec{y}, \bot) &:= \begin{cases} \mathbf{m}_i(\vec{y}) & \text{if } \mathbf{m}_i(\vec{y}) \text{ does not belong to } \mathfrak{b}_{i+1} \\ \text{undefined} & \text{else} \end{cases}
\end{aligned}
$$

Thus, for each separating pair $(\mathbf{g}_i(\vec{x}), \mathbf{m}_i(\vec{y}))$ in L_i, the user has to answer the two following questions: "Does the object $\mathbf{g}_i(\vec{x})$ belong to the concept $\mathfrak{b}_{i+1}$?" and "Does the attribute $\mathbf{m}_i(\vec{y})$ belong the concept $\mathfrak{b}_{i+1}$?". The algorithm uses the fact that the answer "Yes" to one of the questions implies the answer "No" to the other one.

The problem of finding the rows and columns in $\widetilde{\mathbb{K}}_i$ that have to be deleted, now turns out to be equivalent to completing the partial mappings $\tilde{\mathbf{g}}_i$ and $\tilde{\mathbf{m}}_i$: If, for $\vec{x} \in \widetilde{G}_i$ and $\vec{y} \in \widetilde{M}_i$ with $\vec{x} \swarrow\!\!\!\nearrow \vec{y}$, at least one of $\tilde{\mathbf{g}}_i(\vec{x})$ and $\tilde{\mathbf{m}}_i(\vec{y})$ is undefined and the user is not able to find an object or attribute for completing the separating pair, then the row $\vec{x}$ and the column $\vec{y}$ have to be deleted. In two cases we can benefit from the already given knowledge:

1. If $\tilde{\mathbf{g}}_i(\vec{x})$ is undefined, $\tilde{\mathbf{m}}_i(\vec{y})$ is defined and $\vec{x} = (\top, \ldots, \top, \mathfrak{b}_i)$, then we already know that there must exist an object that belongs to $\mathfrak{b}_i$ and that does not have the attribute $\tilde{\mathbf{m}}_i(\vec{y})$. The user is then asked for such an object.
2. If $\tilde{\mathbf{g}}_i(\vec{x})$ is defined and $\tilde{\mathbf{m}}_i(\vec{y})$ is undefined then there must exist an attribute of $\bigvee \vec{y}$ that $\tilde{\mathbf{g}}_i(\vec{x})$ does not have. The user is then asked for such an attribute.

We are now ready to list the algorithm of Distributive Concept Exploration:

```
Algorithm: Given is the list $\mathfrak{b}_1, \mathfrak{b}_2, \ldots, \mathfrak{b}_n$ of basic concepts.
1. $i := 0$, $\widetilde{\mathbb{K}}_0 := (\{\top\}, \{\bot\}, \emptyset)$, $\tilde{\mathbf{g}}_0(\top)$ := undefined, $\tilde{\mathbf{m}}_0(\bot)$ := undefined.
2. For each $(\vec{x}, \vec{y}) \in \widetilde{G}_i \times \widetilde{M}_i$ with $\vec{x} \swarrow\!\!\!\nearrow \vec{y}$,
   where $\tilde{\mathbf{g}}_i(\vec{x})$ or $\tilde{\mathbf{m}}_i(\vec{y})$ are undefined, do:
     • If $\tilde{\mathbf{g}}_i(\vec{x})$ is undefined:
         – If $\tilde{\mathbf{m}}_i(\vec{y})$ is defined and $\vec{x} = (\top, \ldots, \top, \mathfrak{b}_i)$:
             Prompt: "Name an object belonging to $\mathfrak{b}_i$ and not having
             the attribute $\tilde{\mathbf{m}}_i(\vec{y})$!" Set $\tilde{\mathbf{g}}_i(\vec{x})$ according to the answer.
         – Else do:
             Ask the user: "Is the infimum of $\vec{x}$ a subconcept
                                of the supremum of $\vec{y}$?"
             "Yes": Delete $\vec{x}$ in $\widetilde{G}_i$, $\vec{y}$ in $\widetilde{M}_i$,
                    and the corresponding row and column in $\widetilde{I}_i$.
             "No":  Prompt: "Give a separating pair for $\bigwedge \vec{x}$ and $\bigvee \vec{y}$!"
                    If $\tilde{\mathbf{m}}_i(\vec{y})$ is defined, add:
                       "Eventually you can use $\tilde{\mathbf{m}}_i(\vec{y})$ as attribute."
                    Set $\tilde{\mathbf{g}}_i(\vec{x})$ and $\tilde{\mathbf{m}}_i(\vec{y})$ according to the answer.
     • Else (i.e. $\tilde{\mathbf{g}}_i(\vec{x})$ is defined and $\tilde{\mathbf{m}}_i(\vec{y})$ is undefined) do:
         Prompt: "Name an attribute of $\bigvee \vec{y}$ that $\tilde{\mathbf{g}}_i(\vec{x})$ does not have!"
         Set $\tilde{\mathbf{m}}_i(\vec{y})$ according to the answer.
3. Set $\mathbb{K}_i := \widetilde{\mathbb{K}}_i$, $\mathbf{g}_i := \tilde{\mathbf{g}}_{i\,|G_i}$, $\mathbf{m}_i := \tilde{\mathbf{m}}_{i\,|M_i}$.
4. If $i{=}n$, then STOP.
5. Set $\widetilde{\mathbb{K}}_{i+1} := \mathbb{K}_i \times (\{\mathfrak{b}_{i+1}, \top\}, \{\bot, \mathfrak{b}_{i+1}\}, \{(\mathfrak{b}_{i+1}, \mathfrak{b}_{i+1})\})$.
6. For each $(\vec{x}, \vec{y}) \in G_i \times M_i$ with $\vec{x} \swarrow\!\!\!\nearrow \vec{y}$:
   – Ask the user: "Is $\mathbf{g}_i(\vec{x})$ a $\mathfrak{b}_{i+1}$?"
   – If "No", ask "Has each object in $\mathfrak{b}_{i+1}$ the attribute $\mathbf{m}_i(\vec{y})$?"[1]
   – Set $\tilde{\mathbf{g}}_{i+1}(\vec{x}, \mathfrak{b}_{i+1})$ and $\tilde{\mathbf{g}}_{i+1}(\vec{x}, \top)$ as defined above.
```

[1] These two questions are equivalent to "Does the object $\mathbf{g}_i(\vec{x})$ belong to the concept $\mathfrak{b}_{i+1}$?" and "Does the attribute $\mathbf{m}_i(\vec{y})$ belong to the concept $\mathfrak{b}_{i+1}$?", resp.

```
    – Set $\widetilde{\mathbf{m}}_{i+1}(\vec{y}, \mathfrak{b}_{i+1})$ and $\widetilde{\mathbf{m}}_{i+1}(\vec{y}, \perp)$ as defined above.
7. Set $i := i + 1$.
8. Goto Step 2.
```

The result of the algorithm can be shown by a line diagram of $\underline{\mathfrak{B}}(\mathbb{K}_n)$. It is not necessary to label all the object and attribute concepts in the diagram. Only the concepts $\bigvee\{\gamma\vec{x} \mid \vec{x}{\in}G_n,\, x_i{=}\mathfrak{b}_i\}$ $(= \bigwedge\{\mu\vec{y} \mid \vec{y}{\in}M_n,\, y_i{=}\mathfrak{b}_i\})$ of $\underline{\mathfrak{B}}(\mathbb{K}_n)$ have to be labeled by $\mathfrak{b}_i$, as they correspond to the basic concepts which completely generate the whole lattice. The resulting list of separating pairs can be displayed in the same diagram: For each pair $\vec{x} \swarrow\!\!\!\nearrow \vec{y}$ in $\mathbb{K}_n$, there is exactly one separating pair $(\mathbf{g}_n(\vec{x}), \mathbf{m}_n(\vec{y}))$. We label the concept $\gamma\vec{x}$ by $\mathbf{g}_n(\vec{x})$ and the concept $\mu\vec{y}$ by $\mathbf{m}_n(\vec{y})$ and mark $\gamma\vec{x}$ and $\mu\vec{y}$ with the same symbol. An example can be seen in the next section.

4 An Exploration of Zinks

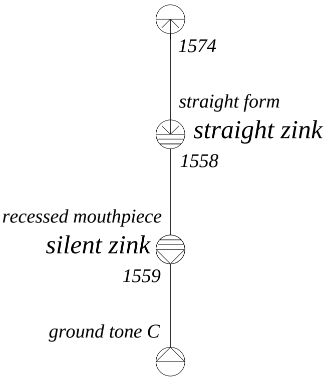
As an example, we want to explore a family of musical instruments: *Zinks* are wind instruments with a conical wide–bored tube, a shortening hole–system and a mouth piece played like a trumpet. We start the exploration with the following basic concepts: $\mathfrak{b}_1$ = *straight zink* [*gerader Zink*], $\mathfrak{b}_2$ = *silent zink* [*stiller Zink*], $\mathfrak{b}_3$ = *curved zink* [*krummer Zink*], $\mathfrak{b}_4$ = *cornettino*, and $\mathfrak{b}_5$ = *cornetto*. The exploration is based on information given by the catalogue of the museum of musical instruments of the University of Leipzig ([2]). The zinks used for separating pairs are named by their catalogue number.

Figure 2 shows the result of the exploration of the two first basic concepts *straight zink* and *silent zink* (i. e., after Step 4 of the algorithm with $i = 2$). For instance, one can see in the diagram that *silent zink* is a subconcept of *straight zink*. The fact that not *everything* is a *straight zink* is asserted by the separating pair *Zink 1574* and *straight form*. The relation $\swarrow\!\!\!\nearrow$ is indicated in the diagram by using the same symbol (e. g., *everything*$\swarrow\!\!\!\nearrow$*straight zink* by ⦶ and ⊛) Next we determine the largest lattice that is possibly generated by adding the next basic concept *curved zink* (Steps 5 & 6):

> "Is *Zink 1559* a *curved zink*?" — "No!" — "Has every *curved zink* the attribute *ground tone C*?" — "No!" — "Is *Zink 1558* a *curved zink*?" — "No!" — "Has each *curved zink* the attribute *recessed mouthpiece*?" — "No!" — "Is *Zink 1574* a *curved zink*?" — "No!" — "Has each *curved zink* the attribute *straight form*?" — "No!"

Figure 3 shows the context $\widetilde{\mathbb{K}}_3$ and the mappings $\widetilde{g}_3$ and $\widetilde{m}_3$. Steps 2 & 3 then determine the congruence relation on $\underline{\mathfrak{B}}(\widetilde{\mathbb{K}}_3)$ that reflects the dependencies between the concept *curved zink* and the concepts *straight zink* and *silent zink*.

> "Is the infimum of *straight zink*, *silent zink* and *curved zink* a subconcept of *nothing*?" — "Yes!" — "Is the infimum of *straight zink* and *curved zink* a subconcept of *silent zink*?" — "Yes!" — "Name a *curved zink* not having



		$\mathbf{m}_2$:	ground tone C	recessed mouthpiece	straight form
g_2:	$\mathbb{K}_2$		$\bot$	$\mathfrak{b}_2$	$\mathfrak{b}_1 \vee \mathfrak{b}_2$
1559	$\mathfrak{b}_1 \wedge \mathfrak{b}_2$		⤢	×	×
1558	$\mathfrak{b}_1$			⤢	×
1574	$\top$				⤢

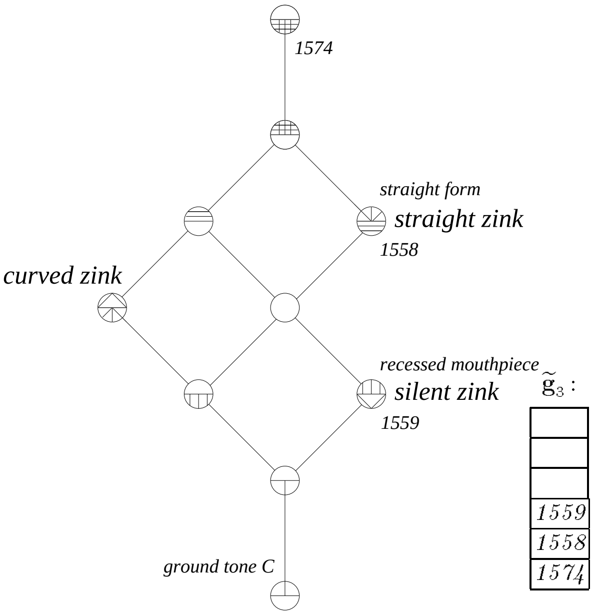
Fig. 2. The result of the exploration of the first two basic concepts

> *straight form*!" — "*Zink 1563.*" — "Name an attribute of *curved zinks* that *Zink 1559* does not have!" — "*Attached mouthpiece*." — "Name an attribute of the supremum of *silent zink* and *curved zink* that *Zink 1558* does not have!" — "*Recessed mouthpiece or curved form.*" — "Name an attribute of the supremum of *straight zink*, *silent zink* and *curved zink* that *Zink 1574* does not have!" — "*More than 6 finger holes*."

Up to now (at Step 4 with $i = 3$), we have determined the complete lattice generated by the first three basic concepts *straight zink*, *silent zink*, and *curved zink*. It is shown in Figure 4. We continue the exploration in the same way with the remaining two basic concepts *cornetto* and *cornettino*. Finally, we get the context $\mathbb{K}_5$ as shown in Figure 5. Its line diagram shows all information about the hierarchical relationships between the five basic concepts. For example, we can deduce from it that there are no *silent zinks* that are also *cornettos*, because the infimum of *silent zink* and *cornetto* is *nothing*. We can further deduce that there are other zinks than those we chose for the exploration, because the supremum of all basic concepts is different from *everything*. The observation that the supremum of *cornetto* and *cornettino* is *curved zink* and their infimum is *nothing* reflects the fact that the *curved zinks* can be divided in two disjoint classes: *cornettos* and *cornettinos*.

Let us remark that, in the Fig. 5, *Zink 1558* is not laying below *attached mouthpiece*, even though Zink 1558 has an attached mouthpiece! *Zink 1558* and *attached mouthpiece* belong to different separating pairs, and so their relationship has not been asked from the expert.

If there are other subconcepts of *zink* we are interested in (for example *tenor zink*, *serpent* or *violoncel serpent*) we can continue the exploration by starting with the context $\mathbb{K}_5$ and adding the new basic concepts. This serial approach allows also to extend the acquired knowledge at a later time.



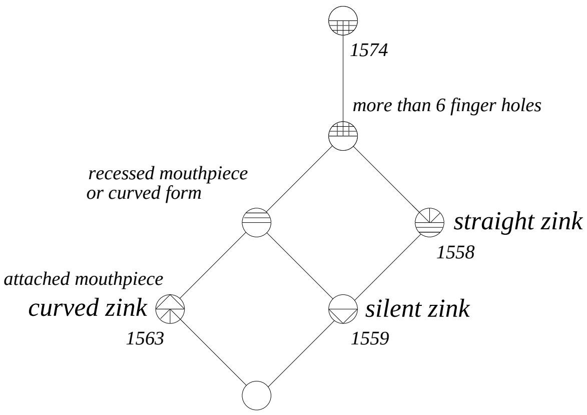
	$\widetilde{\mathbf{m}}_3$:	*ground tone C*	*recessed mouthpiece*	*straight form*			
$\widetilde{\mathbf{g}}_3$:	$\widetilde{\mathbb{K}}_3$	$\bot$	$\mathfrak{b}_2$	$\mathfrak{b}_1 \vee \mathfrak{b}_2$	$\mathfrak{b}_3$	$\mathfrak{b}_2 \vee \mathfrak{b}_3$	$\mathfrak{b}_1 \vee \mathfrak{b}_2 \vee \mathfrak{b}_3$
	$\mathfrak{b}_1 \wedge \mathfrak{b}_2 \wedge \mathfrak{b}_3$	⤢	×	×	×	×	×
	$\mathfrak{b}_1 \wedge \mathfrak{b}_3$		⤢	×	×	×	×
	$\mathfrak{b}_3$			⤢	×	×	×
1559	$\mathfrak{b}_1 \wedge \mathfrak{b}_2$		×	×	⤢	×	×
1558	$\mathfrak{b}_1$			×		⤢	×
1574	$\top$						⤢

Fig. 3. The intermediate result $\widetilde{\mathbb{K}}_3$ and its concept lattice

5 Conclusion

The algorithm as described above is not able to treat incomplete knowledge. The user is assumed to reply to each question during the exploration either with "Yes" or "No". With a little change we can allow the answer "I don't know" to the question "Is the infimum of $\vec{x}$ a subconcept of the supremum of $\vec{y}$?": In this case the row $\vec{x}$ and the column $\vec{y}$ will not be deleted in $\widetilde{\mathbb{K}}_i$ and $\widetilde{\mathbf{g}}_i(\vec{x})$ and $\widetilde{\mathbf{m}}_i(\vec{y})$ will be set to the default value *?*. In Step 6 of the algorithm all $\widetilde{\mathbf{g}}_{i+1}(\vec{x}, \mathfrak{b}_{i+1})$, $\widetilde{\mathbf{g}}_{i+1}(\vec{x}, \top)$, $\widetilde{\mathbf{m}}_{i+1}(\vec{y}, \mathfrak{b}_{i+1})$ and $\widetilde{\mathbf{m}}_{i+1}(\vec{y}, \bot)$ will then automatically be set to *?*. These *?* play the role of "possible separating pairs". During and after the exploration procedure the user can either replace them by a real separating pair or he can delete the corresponding row and column (if he is then sure that the inequality $\bigwedge \vec{x} \leq \bigvee \vec{y}$ holds). The result of the exploration can be shown by a list of line diagrams — one for each possibility of deleting corresponding rows and columns that are not confirmed by a concrete separating pair.

The algorithm generates in the worst case (i.e., the user denies all dependencies between the basic concepts) the free bounded distributive lattice $\text{FBD}(\mathfrak{b}_1, \ldots, \mathfrak{b}_n)$, which is growing super-exponentially. However, the algorithm is working on the level of the formal contexts only, whose sizes are logarithmic in the sizes of the concept lattices. Hence, if the basic concepts are sufficiently related, then the exploration can be done in reasonable time. Its efficiency also depends on the ordering of the basic concepts: The stronger the first basic concepts are related, the smaller the contexts can be kept during the exploration.



	$\mathfrak{m}_3$:	*straight form*	*attached mouthpiece*	*recessed mouthpiece or curved form*	*more than 6 finger holes*
$\mathfrak{g}_3$:	$\mathbb{K}_3$	$\mathfrak{b}_1 \vee \mathfrak{b}_2$	$\mathfrak{b}_3$	$\mathfrak{b}_2 \vee \mathfrak{b}_3$	$\mathfrak{b}_1 \vee \mathfrak{b}_2 \vee \mathfrak{b}_3$
1563	$\mathfrak{b}_3$	⤢	×	×	×
1559	$\mathfrak{b}_1 \wedge \mathfrak{b}_2$	×	⤢	×	×
1558	$\mathfrak{b}_1$	×		⤢	×
1574	$\top$				⤢

Fig. 4. Result of the exploration of the first three basic concepts

The final result, of course, is independent of this ordering. For basic concepts that are only weakly related, the whole lattice generated by them is often not requested. Then the basic concepts can be divided in stronger related classes which are explored separately (cf. [4]).

References

1. B. Ganter, R. Wille: Formale Begriffsanalyse: Mathematische Grundlagen. Springer, Heidelberg 1996
2. H. Heyde: Hörner und Zinken. Musikinstrumenten–Museum der Universität Leipzig. Katalog Bd. 5. VEB Deutscher Verlag für Musik, Leipzig 1982
3. U. Klotz, A. Mann: Begriffexploration. Diplomarbeit, TH Darmstadt 1988
4. G. Stumme: Exploration tools in formal concept analysis. In: *Ordinal and symbolic data analysis.* Studies in classification, data analysis, and knowledge organization **8**, Springer, Heidelberg 1996, 31–44
5. G. Stumme: The concept classification of a terminology extended by conjunction and disjunction. In: N. Foo, R. Goebel (eds.): PRICAI'96: Topics in artificial intelligence. LNAI **1114**, Springer, Heidelberg 1996, 121–131
6. G. Stumme: Concept Exploration – A Tool for Creating and Exploring Conceptual Hierarchies. In: D. Lukose, H. Delugach, M. Keeler, L. Searle, J. F. Sowa (eds.): *Conceptual Structures: Fulfilling Peirce's Dream.* LNAI **1257**, Springer, Berlin 1997, 318–331

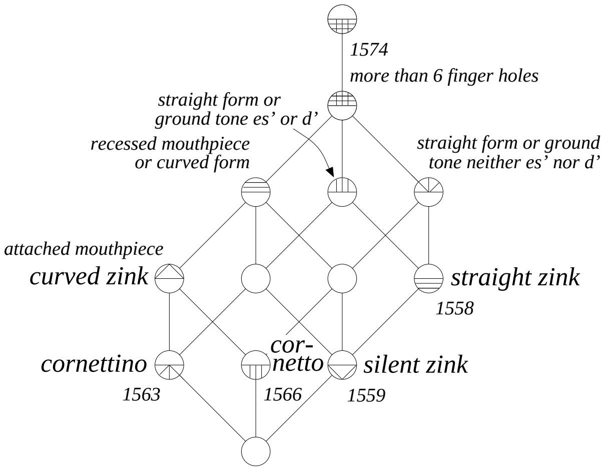


Fig. 5. Result of the Distributive Concept Exploration of zinks

7. G. Stumme: Concept Exploration – Knowledge Acquisition in Conceptual Knowledge Systems. Dissertation. Shaker, Aachen 1997
8. H. Wagner: Begriff. In: H. M. Baumgartner, C. Wild (eds.): *Handbuch philosophischer Grundbegriffe.* Kösel Verlag, München 1973, 191–209
9. R. Wille: Restructuring lattice theory: an approach based on hierarchies of concepts. In: I. Rival (ed.): *Ordered sets.* Reidel, Dordrecht–Boston 1982, 445–470
10. R. Wille: Tensorial decomposition of concept lattices. In: *Order* **2**, 1985, 81–95
11. R. Wille: Bedeutungen von Begriffsverbänden. In: B. Ganter, R. Wille, K. E. Wolff (eds.): *Beiträge zur Begriffsanalyse.* B. I.–Wissenschaftsverlag, Mannheim 1987, 161–211
12. R. Wille: Knowledge acquisition by methods of formal concept analysis. In: E. Diday (ed.): *Data analysis, learning symbolic and numeric knowledge.* Nova Science Publisher, New York, Budapest 1989, 365–380