Concept Exploration – A Tool for Creating and Exploring Conceptual Hierarchies

Gerd Stumme

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

Abstract. Concept exploration is a knowledge acquisition tool for interactively exploring the hierarchical structure of finitely generated lattices. Applications comprise the support of knowledge engineers by constructing a type lattice for conceptual graphs, and the exploration of large formal contexts in formal concept analysis.

1 Introduction

Lattices are a popular mathematical structure for modeling conceptual hierarchies. The existence of greatest common subconcepts and least common superconcepts provides additional algebraic structure to models based on ordered sets, and makes them more suitable for computation. Exploration tools, as developed in formal concept analysis,[1] benefit from this algebraic structure (cf. [12]). The best known and most often used of these tools is *attribute exploration* ([4], [2], [6]), which uses only infima (greatest common subconcepts) for the computation. It determines – in interaction with the user – the implicational logic of attributes of a given formal context, or, more algebraically spoken, the $\bigwedge$-subsemilattice of a finite lattice generated by some subset.

When, in addition, suprema (least common superconcepts) are considered, the problem turns out to determine the sublattice of a given lattice (of concepts) generated by some subset. This is the aim of *concept exploration.* If a priori the lattice is known to be distributive, then the exploration can benefit from the much stronger algebraic structure induced by the distributive law. The corresponding exploration tool is called *distributive concept exploration* ([13], [15]).

Already in the first publication on formal concept analysis, [17], the basic conception of concept exploration is mentioned. Its scheme is demonstrated by examples in [18] and [19]. U. Klotz and A. Mann further elaborated the method in [7]. Unfortunately, their work is, with more than hundred pages, very extensive, so that the results became too complicated for an efficient implementation. This gave the impulse to develop a more transparent tool using existing algorithms which have been proven successful.

Concept exploration is designed to explore a lattice L that is generated by a subset $B \subseteq L$. The algorithm generates questions about the lattice (i. e., the

[1] An introduction into formal concept analysis can for instance be found in [6], [17], or [20].

conceptual hierarchy) of the kind "Is $\mathfrak{c}_1$ a subconcept of $\mathfrak{c}_2$?" that are answered by a (human) user, usually an expert of the field of interest. There are (at least) two interesting applications for such a knowledge acquisition tool.

Firstly in formal concept analysis: Suppose we have a formal context $\mathbb{K}$ which is too large (e. g., infinite) to be completely given. The aim is to determine the structure of the concept lattice, or at least a part of it. Therefore one might consider formal concepts of the concept lattice one is particularly interested in. They are called *basic concepts* (thus the letter B). Concept exploration interactively computes the sublattice L of the concept lattice of $\mathbb{K}$ generated by B. Information on the context $\mathbb{K}$ is acquired from the user.

Secondly for conceptual graphs: The type hierarchy for conceptual graphs is assumed to be a lattice ([10]). This lattice needs to be constructed for modeling a situation by conceptual graphs. Concept exploration can support this creative process: B is now the set of "basic types", i. e., the set of types a knowledge engineer assumes necessary for his purpose. Concept exploration supports him in generating the type lattice L.

2 Concept Exploration

Let L be a lattice generated by a finite subset $B \subseteq L$. The exploration follows the generation process of the lattice. First it computes the set $B^{(1)} := B^{\wedge}$ consisting of all infima of the elements of B. Then it computes $B^{(2)} := B^{\wedge\vee}$, the set of all suprema of $B^{\wedge}$. The next step provides $B^{(3)} := B^{\wedge\vee\wedge}$, and so on. $B^{(k)}$ is considered as a weak sublattice of L in which, for k odd (even), all infima (suprema) are defined, and suprema (infima) are only defined for subsets of $B^{(k-1)} \subseteq B^{(k)}$. If the lattice is finite, then $B^{(k)} = B^{(k+1)} = L$ for some $k \in \mathbb{N}$. Otherwise the exploration would not terminate (which is possible for all B with $\operatorname{card}(B) \geq 3$, because the free lattice over the set B is then infinite) and has to be interrupted at some moment; but the exploration can approach L as much as desired. The resulting partial lattice may then be completed by other tools, as described for instance in [17].

For the step from $B^{(k)}$ to $B^{(k+1)}$, where k is even, concept exploration uses $\bigwedge$-*exploration*, a modification of the algorithm of attribute exploration with background implications ([11]). For odd k, the situation is dual to the former, and $\bigvee$-*exploration*, a modification of object exploration is applied. Object exploration can be understood as an attribute exploration of the dual lattice, i. e., with objects and attributes interchanging their roles ([12]). The idea to use attribute and object exploration alternatively for concept exploration is due to P. Burmeister.

The intermediate results of concept exploration are the weak sublattices $B^{(k)}$ as defined above. They are stored as formal contexts (G, M, I), where G and M contain lattice terms over B. For odd (even) k, G contains exactly one term for every element in $B^{(k)}$ $(B^{(k-1)})$ denoting it, and M contains exactly one term for every element in $B^{(k-1)}$ $(B^{(k)})$ denoting it. The relation I reflects the hierarchical order: $(s,t) \in I :\iff s^L \leq t^L$ for $s \in G$ and $t \in M$. The weak partial lattice $B^{(k)}$ is isomorphic to the weak partial lattice $\mathfrak{B}(G, M, I)$ in which infima are defined on $\gamma(G)$ (which is the whole lattice for odd k), and suprema are defined on $\mu(M)$ (which is the whole lattice for even k). The represenation of partial lattices by concept lattices is described in detail in [16]. In the sequel, we identify the lattice terms with the corresponding elements of L. For L being finite, the final result of the exploration is the context $(L, L, \leq)$.

The first step in concept exploration is a $\bigwedge$-exploration starting with a context having the basic concepts as attributes ($M := B^{(0)} = B$), and having no objects ($G := B^{(-1)} = \emptyset$). The relation I is empty as well. More general, for odd k, the kth exploration step, a $\bigwedge$-exploration, will transform the context $(B^{(k-2)}, B^{(k-1)}, \leq)$ into $(B^{(k)}, B^{(k-1)}, \leq)$. The next exploration step, a $\bigvee$-exploration, then transforms it into the context $(B^{(k)}, B^{(k+1)}, \leq)$, and so on. This yields a first sketch of the algorithm of concept exploration:

```
(0)  Start with $k := 0$ and $(G, M, I) := (B^{(-1)}, B^{(0)}, \leq) = (\emptyset, B, \emptyset)$.
(I)  Increase $k$. Determine $(B^{(k)}, B^{(k-1)}, \leq)$ by $\bigwedge$-exploration.
     If $B^{(k)} = B^{(k-1)}$ then STOP.
(II) Increase $k$. Determine $(B^{(k-1)}, B^{(k)}, \leq)$ by $\bigvee$-exploration.
     If $B^{(k)} = B^{(k-1)}$ then STOP, else go to Step (I).
```

Since the situation is perfectly symmetric, we only describe Step (I) in detail. Let k be odd in the sequel.

Step (I) extends the set $G = B^{(k-2)}$ to $B^{(k)}$ by lattice terms, one for each element in $B^{(k)} \setminus B^{(k-2)}$. For elements in $B^{(k-1)} \setminus B^{(k-2)}$, they can just be copied from M to G. For every copied element x, the relation I is extended by

$$(\dagger) \qquad (x, y) \in I :\iff \{x\}' \subseteq \{y\}', \text{ where } X' := \{g \in G \mid \forall m \in X\colon (g, m) \in I\}.$$

This "copying" is only omitted for $k = 1$, because the first $\bigwedge$-exploration is also used for determining the ordering on B which is not known at the beginning.

For determining the elements in $B^{(k)}$ which are not included in $B^{(k-1)}$, Step (I) is concluded by a $\bigwedge$-exploration.

2.1 Discovering New Concepts by $\bigwedge$-Exploration

The elements in G correspond to infima of elements in $M = B^{(k-1)}$, since the lattice equality $x = \bigwedge(\{x\}')$ holds for every element x in G. For $x \in B^{(k-1)}$, this is obvious, since x is also an element of M. For $x \in B^{(k)} \setminus B^{(k-1)}$, the intent x' is just constructed such that this equality holds.

Since $Y \subseteq x' \iff x \in Y' \iff \forall y \in Y\colon x \leq y \iff x \leq \bigwedge Y$ holds for every lattice element x in G and every subset Y of M, we can also identify every subset $Y \subseteq M = B^{(k-1)}$ with the infimum $\bigwedge Y$ of its elements. At the beginning of the $\bigwedge$-exploration, there may be subsets $Y \subseteq M$ such that $\bigwedge Y \neq \bigwedge(Y'')$ in L. The aim of $\bigwedge$-exploration is to add new objects to G in order to change the derivation operator $''$ such that $\bigwedge Y = \bigwedge(Y'')$ holds in L for all $Y \subseteq M$. As we will see below, it is not necessary to test all subsets Y, but we can restrict ourselves to *pseudo-intents*, which are defined next.

The following definitions and results are cited from [11], where the original algorithm of attribute exploration of B. Ganter ([4], [6], [5]) was modified to accept additional knowledge in form of background implications.

Let $\mathbb{K} := (G, M, I)$ be a formal context. We assume in the following that M is finite.

Definition. An *implication* $X \to Y$ of $\mathbb{K}$ is a pair of subsets X and Y of M, such that every object having all attributes in X also has all attributes in Y.

Let $\mathcal{L}$ be a set of implications of $\mathbb{K}$ (called *background implications*). The closure operator on the set M of attributes induced by the background implications is denoted by $P \mapsto \overline{P} := P \cup P^{\mathcal{L}} \cup P^{\mathcal{L}\mathcal{L}} \cup \dots$ with

$$Q^{\mathcal{L}} := Q \cup \bigcup \{Y \subseteq M | X \subseteq Q,\, X \to Y \in \mathcal{L}\}.$$

A subset P of M is called an *$\mathcal{L}$-pseudo-intent* of $\mathbb{K}$ if $P = \overline{P} \neq P''$ and if, for every $\mathcal{L}$-pseudo-intent Q with $Q \subset P$, the inclusion $Q'' \subseteq P$ holds.

Theorem 1. *The set $\mathcal{B}_{\mathcal{L}} := \{P \to P'' | P$ is a $\mathcal{L}$-pseudo-intent$\}$ of implications is an irredundant set of implications such that every implication of $\mathbb{K}$ can be deduced from $\mathcal{L} \cup \mathcal{B}_{\mathcal{L}}$.*

The set of all intents and $\mathcal{L}$-pseudo-intents of a finite context (G, M, I) is a closure system on M; with the closure operator $A \mapsto A^{\bullet} := A \cup A^{\circ} \cup A^{\circ\circ} \cup \dots$, where $A^{\circ} := \overline{A \cup \bigcup \{Y \subseteq M | X \to Y \in \mathcal{B}_{\mathcal{L}}, X \subset A\}}$.

In our application, the set $\mathcal{L}$ will store information about the hierarchical structure that is computed in previous exploration steps. Intents will correspond to already existing elements, i.e., elements in $B^{(k)}$, while pseudo-intents correspond to potentially new elements which may be added to $B^{(k)}$ in order to obtain $B^{(k+1)}$, depending whether $\bigwedge P = \bigwedge(P'')$ (which is equivalent to $P \to P''$) holds or not – which will be asked from the user.

$\bigwedge$-Exploration uses Ganter's Next-Closure-algorithm (cf. [4]) to compute all intents and all $\mathcal{L}$-pseudo-intents in a *lectical order*. For the sake of simplicity we assume that $M := \{1, \dots, n\}$.

Definition. For $X, Y \subseteq M$ and $i \in M$ we define[2]

$$X <_i Y :\iff (i \in Y \setminus X \text{ and } X \cap \{i+1, \dots n\} = Y \cap \{i+1, \dots n\}).$$

The *lectical order* on $\mathfrak{P}(M)$ is defined by

$$X < Y :\iff \exists i : X <_i Y \quad .$$

For $X \subseteq M$ and $i \in M$ let

$$X \oplus i := ((X \cap \{i+1, \dots, n\}) \cap \{i\})^{\bullet} \quad .$$

[2] In contrast to [11], in this definition the ordering on M is reversed, since this matches better the generation process of concept exploration.

Theorem 2. *The lecticly first intent or $\mathcal{L}$-pseudo-intent is $\overline{\emptyset}$. For a given subset $Y \subseteq M$ the lecticly next intent or $\mathcal{L}$-pseudo-intent is the set $Y \oplus i$, where i is the minimal element in $M \setminus Y$ with $Y <_i Y \oplus i$. The lecticly last intent or $\mathcal{L}$-pseudo-intent is M.*

Attribute exploration and $\bigwedge$-exploration are based on the fact that, during the computation, the list of already computed intents and $\mathcal{L}$-pseudo-intents is stable under adding new objects which respect the previously computed implications (and the background implications).

Additionally to the contexts, $\bigwedge$-exploration produces a list $\mathcal{L}$ of implications. In concept exploration, these implications will be used as background implications for later $\bigwedge$-explorations. (For the $\bigvee$-explorations, a similar list $\mathcal{L}'$ will be kept.) At the beginning of the first $\bigwedge$-exploration, $\mathcal{L}$ is empty. The following is the algorithm described in [11] adapted[3] to concept exploration.

Algorithm 1. Set $n :=$card$(B^{(k-1)})$, $P := \emptyset$.

```
(a) Ask the user: "Which of the concepts ≪Here all elements in $P'' \setminus P$ are listed≫
    are superconcepts of $\bigwedge P$?" The answer is a set $Q \subseteq P'' \setminus P$.
(b) Add the implication $P \to Q$ to $\mathcal{L}$.[4]
(c) If there is no $g \in G$ with $g' = P \cup Q$ then add the lattice term $\bigwedge P$ to $G$, and
    extend $I$ as follows:   $(\bigwedge P, m) \in I :\iff m \in P \cup Q$   for $m \in M$.
(d) Set $Y := P$.
(e) Determine the next intent or $\mathcal{L}$-pseudo-intent $P$ following $Y$ by applying Theorem 2:
    • $i := 1$
    • While $Y \not<_i Y \oplus i$ increase $i$.
    • $P := Y \oplus i$.
(f) If $P = M$ then STOP.
(g) If $P = P''$ then go to Step (e), else go to step (a).
```

The algorithm for $\bigvee$-exploration is exactly the same, only G and M have to be interchanged, "superconcepts" has to be replaced by "subconcepts", "infimum" by "supremum", $\bigwedge$ by $\bigvee$, and $\mathcal{L}$ by $\mathcal{L}'$. The implications $P \to P''$ in $\mathcal{L}'$ correspond to equalities $\bigvee P = \bigvee(P'')$.

[3] In Step (c), *one* element (the lattice term $\bigwedge P$) is added to G having *exactly* the attributes in $P \cup Q$. In other applications of attribute exploration, there may be more than one object necessary in order to restrict the conclusion of the implication to $P \cup Q$.

[4] Even if Q is empty! This will be needed in the over-next exploration, where P may again be a $\mathcal{L}$-pseudo-intent (instead of an intent) in the extended context. (See Steps (4) and (8) in Algorithm 2.)

2.2 Concept Exploration

Now we are ready to present the algorithm of concept exploration. Step (8) and its dual, Step (4), are described below, as well as Steps (9) and (5).

Algorithm 2. Set $k := 1$, $(G, M, I) := (\emptyset, B, \emptyset)$, $\mathcal{L} := \mathcal{L}' := \emptyset$.

(1) Determine $(B^{(1)}, B^{(0)}, \leq)$ by Algorithm 1.
(2) Increase k.
(3) Copy $B^{(k-1)} \setminus B^{(k-2)}$ from G to M. Extend I as described dually at (†).
(4) Change every implication $X \to Y$ in $\mathcal{L}'$ to $X \to X''$ (where X'' is computed in the modified context). For every $x \in B^{(k-1)} \setminus B^{(k-2)}$ ($x \in B^{(1)}$ for $k = 2$) add the implication $\{x\} \to \{x\}''$ to $\mathcal{L}'$.
(5) Determine $(B^{(k-1)}, B^{(k)}, \leq)$ by the dual of Algorithm 1 (for $k > 2$ starting at Step (e) with $Y := B^{(k-3)}$ and answering "All" automatically while $P \subseteq B^{(k-2)}$).
If M is not changed (i. e., all questions are answered by "All") then STOP.
(6) Increase k.
(7) Copy $B^{(k-1)} \setminus B^{(k-2)}$ from M to G. Extend I as described at (†).
(8) Change every implication $X \to Y$ in $\mathcal{L}$ to $X \to X''$ (where X'' is computed in the modified context). For every $x \in B^{(k-1)} \setminus B^{(k-2)}$ add the implication $\{x\} \to \{x\}''$ to $\mathcal{L}$.
(9) Determine $(B^{(k)}, B^{(k-1)}, \leq)$ by Algorithm 1, starting at Step (e) with $Y := B^{(k-3)}$ and answering "All" automatically while $P \subseteq B^{(k-2)}$.
If G is not changed (i. e., all questions are answered by "All") then STOP.
(10) Go to step (2).

In this algorithm, Steps (1) and (6)–(9) correspond to Step (I) in the first sketch, Steps (2)–(5) to Step (II). In Step (8) (and dually in Step (4)), the previously computed implications are adapted to the newly generated attributes. More precisely, the conclusions of the implications are extended by the appropriate attributes from $B^{(k-1)} \setminus B^{(k-2)}$. (The premises remain subsets of $B^{(k-2)}$.) The implications $\{x\} \to \{x\}''$ with $x \in B^{(k-1)} \setminus B^{(k-2)}$ are added because $\{x\}''$ is simply the order filter generated by x, which can be determined automatically. Otherwise, these implications had to be confirmed by the user in the next $\bigwedge$-exploration.

The numbering of the newly generated elements (which is relevant for the variable i of $\bigvee/\bigwedge$-exploration) is just done in the way the elements are generated. This provides the optimization in Step (9) (and dually in Step (5) for $k > 2$): All intents and $\mathcal{L}$-pseudo-intents being subsets of $B^{(k-3)}$ remain unchanged from the previous $\bigwedge$-exploration, and need not be determined again. Since all elements of $B^{(k-2)}$ are also represented *as objects* in the current context, all equalities $\bigwedge P = \bigwedge(P'')$ with $P \subseteq B^{(k-2)}$ hold. For completing the set $\mathcal{L}$ of implications, we can thus provide the answer "All" automatically while $P \subseteq B^{(k-2)}$.

For an implementation, it might be useful to sort the list of objects in the context $(B^{(1)}, B^{(0)}, \leq)$ after Step (1) such that the basic concepts come first (and in the same ordering as in $B^{(0)}$). Then the ith object and the ith attribute will always be the same lattice term in the sequel.

There are three possibilities to simplify the exploration dialogue for the user. Firstly, the lattice terms in $P'' \setminus P$ can be listed in a linear extension of the hierarchical order. When the user chooses a term t for the set Q, then all elements in the order filter $\{x \in B^{(k)} \mid x \geq t\}$ will be chosen automatically as well.

Secondly, the lattice terms may be simplified automatically. For instance, the subterm $\bigwedge \emptyset$ may be omitted in any $\bigwedge$-term. This, and its dual, will be applied in the following example.

Thirdly, the user has the option to introduce a *name* for every lattice term added to the context. For instance, with MAN and WOMAN being basic concepts, the user may define HUMAN := MAN $\vee$ WOMAN. This shortens the lattice terms and supports the readability of the questions generated by the process. For the creation of a type lattice for conceptual graphs, this option is essential.

3 Example

Let us demonstrate concept exploration by an example. Imagine a knowledge engineer who wants to model knowledge about ancient Greek musical instruments by conceptual graphs. He uses concept exploration for supporting the creation for an adequate type lattice. For instance, he might want to start with the following four basic types: CHORD INSTRUMENT, KYTHARA, WIND INSTRUMENT, and AULOS. A kythara is a harp which, in Greek mythology, is the symbol of Apollo, while an aulos is an oboe like instrument associated with Dionysus. The newly generated types are named using the naming mechanism described at the end of the previous section. The dialogue of the first $\bigwedge$-exploration (i. e., Step (1) of Algorithm 2) consists of eight questions:

"Which of the concepts CHORD INSTRUMENT, KYTHARA, WIND INSTRUMENT, and AULOS are superconcepts of $\bigwedge \emptyset$?" – "None!" – "Name for $\bigwedge \emptyset$?" – "INSTRUMENT."

The first $\mathcal{L}$-pseudo-intent is the empty set. The infimum $\bigwedge \emptyset$ of the empty set is always the largest element of the lattice, and can thus be understood as the concept "everything" comprising all objects of the field of interest, which, in this example, are INSTRUMENT*s. It is the first object added to G. The relation I remains empty. The (trivial) implication* INSTRUMENT $\to \emptyset$ *is added to $\mathcal{L}$, because it will be needed in Step (8).*

"Which of the concepts KYTHARA, WIND INSTRUMENT, and AULOS are superconcepts of CHORD INSTRUMENT?" – "None!"

The name CHORD INSTRUMENT *is added to G, and the (trivial) implication* CHORD INSTRUMENT $\to \emptyset$ *to $\mathcal{L}$.*

"Which of the concepts CHORD INSTRUMENT, WIND INSTRUMENT, and AULOS are superconcepts of KYTHARA?" – "CHORD INSTRUMENT!"

The name KYTHARA *is added to G. The first non-trivial implication,* KYTHARA $\to$ CHORD INSTRUMENT, *is added to $\mathcal{L}$. In this way the extension of G and $\mathcal{L}$ continues ...*

“Which of the concepts CHORD INSTRUMENT, KYTHARA, and AULOS are superconcepts of WIND INSTRUMENT?” – “None!”

“Which of the concepts KYTHARA and AULOS are superconcepts of CHORD INSTRUMENT $\wedge$ WIND INSTRUMENT?” – “None!” – “Name for CHORD INSTRUMENT $\wedge$ WIND INSTRUMENT?” – “AEOLS-HARP.”

An aeols-harp is a harp where the vibration of the chords is induced by wind (either natural wind or generated by bellows).

“Is the concept AULOS a superconcept of CHORD INSTRUMENT $\wedge$ KYTHARA $\wedge$ WIND INSTRUMENT?” – “Yes!” – “Name for CHORD INSTRUMENT $\wedge$ KYTHARA $\wedge$ WIND INSTRUMENT?” – “NOTHING.”

There is no kythara being a wind instrument. We obtain the “absurd type”.

“Is the concept KYTHARA a superconcept of CHORD INSTRUMENT $\wedge$ WIND INSTRUMENT $\wedge$ AULOS?” – “Yes!”

We obtain again the concept NOTHING. *Since there is no lattice term added to G, we are not asked to provide a name.*

The result of the first $\bigwedge$-exploration is shown in Fig. 1. Observe that the diagram has to be read as $\bigwedge$-semilattice, since only all infima are defined, but no suprema. In the following $\bigvee$-exploration, for instance, the supremum of KYTHARA and AEOLS-HARP will not be identified with CHORD INSTRUMENT, but with a real subconcept of it, namely HARP.

	chord instrument	kythara	wind instrument	aulos
chord instument	×			
kythara	×	×		
wind instrument			×	
aulos			×	×
instrument				
aeolsharp	×		×	
nothing	×	×	×	×

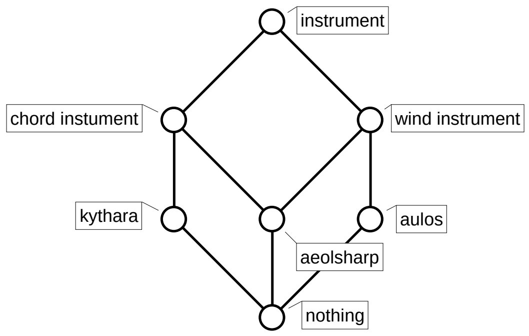

Fig. 1. The result of the first $\bigwedge$-exploration

In Step (3), the names INSTRUMENT, AEOLS-HARP, and NOTHING are copied from G to M, and I is extended such that it represents the ordering shown in the line diagram in Fig. 1. The list $\mathcal{L}'$ of implications is empty up to now, but in Step (4), for every attribute m, the implication $m \to \{x \mid x < m\}$ is added, for example WIND INSTRUMENT $\to$ AULOS, AEOLS-HARP, NOTHING. Then (Step (5)) the next exploration, the first $\bigvee$-exploration, starts:

“Which of the concepts CHORD INSTRUMENT, KYTHARA, WIND INSTRUMENT, AULOS, INSTRUMENT, AEOLS-HARP and NOTHING are subconcepts of $\bigvee \emptyset$?” – “NOTHING!”

This justifies the choice of the name NOTHING *in the former* $\bigwedge$*-exploration, since* $\bigvee\emptyset$ *is always the smallest element of a lattice.*

"Which of the concepts CHORD INSTRUMENT, WIND INSTRUMENT, INSTRUMENT, and AEOLS-HARP are subconcepts of KYTHARA $\vee$ AULOS?" – "None!" – "Name for KYTHARA $\vee$ AULOS?" – "ANCIENT GREEK INSTRUMENT."

Here and in the sequel, the concept NOTHING *is omitted in the lattice terms, since it is not relevant for the supremum.*

"Is the concept CHORD INSTRUMENT a subconcept of KYTHARA $\vee$ AEOLS-HARP?" – "No!" – "Name for KYTHARA $\vee$ AEOLS-HARP?" – "HARP."

"Is the concept WIND INSTRUMENT a subconcept of AULOS $\vee$ AEOLS-HARP?" – "No!" – "Name for AULOS $\vee$ AEOLS-HARP?" – "WIND INSTRUMENT WITH VIBRATING PART."

"Which of the concepts CHORD INSTRUMENT, WIND INSTRUMENT, and INSTRUMENT are subconcepts of KYTHARA $\vee$ AULOS $\vee$ AEOLS-HARP?" – "All!"

Here we decide to identify the concept KYTHARA $\vee$ AULOS $\vee$ AEOLS-HARP *with* INSTRUMENT.

The result of this $\bigvee$-exploration is shown in Fig. 2. In the resulting weak partial lattice, all suprema are defined. Infima are only defined for its $\bigwedge$-subsemilattice which is shown in Fig. 1.

	chord instrument	kythara	wind instrument	aulos	instrument	aeolsharp	nothing	ancient Greek instrument	harp	wind instr. w. vibrating part
chord instument	×				×					
kythara	×	×			×			×	×	
wind instrument			×		×					
aulos			×	×	×			×		×
instrument					×					
aeolsharp	×		×		×	×			×	×
nothing	×	×	×	×	×	×	×	×	×	×

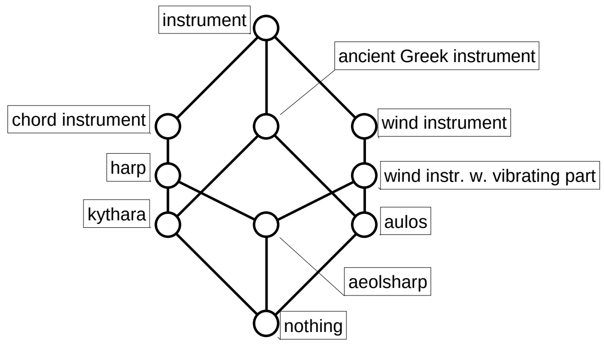

Fig. 2. The result of the first $\bigvee$-exploration

In the next $\bigwedge$-exploration, the names ANCIENT GREEK CHORD INSTRUMENT, ANCIENT GREEK HARP, ANCIENT GREEK WIND INSTRUMENT, and ANCIENT GREEK WIND INSTRUMENT WITH VIBRATING PART are introduced. The following $\bigvee$-exploration provides only one new name, CHORD INSTRUMENT WITHOUT BOW. Finally, the 5th exploration (which is a $\bigwedge$-exploration again) runs through without generating any question. Hence, the concept exploration is terminated. The resulting lattice is shown in Fig. 3. Of course, all infima and suprema are defined now.

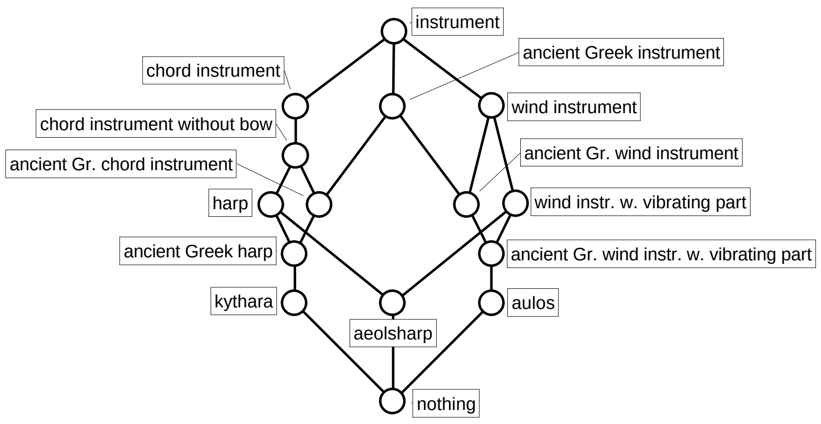
instrument
chord instrument
ancient Greek instrument
wind instrument
chord instrument without bow
ancient Gr. wind instrument
ancient Gr. chord instrument
harp
wind instr. w. vibrating part
ancient Greek harp
ancient Gr. wind instr. w. vibrating part
kythara
aulos
aeolsharp
nothing

Fig. 3. The result of the concept exploration

The "degree of exactitude", i. e., the decision whether or not to identify two concepts, essentially depends on the purpose the exploration was done for. For instance, a very pedantic user might only accept the two implications KYTHARA $\rightarrow$ CHORD INSTRUMENT and AULOS $\rightarrow$ WIND INSTRUMENT, and deny all others. Then every question generates a new concept. This process can be continued ad infinitum, converging towards the (infinite) free lattice generated by two two-element chains, as shown in Fig. 4. The ellipses in the diagram indicate which of the elements of the free lattice were identified in the former exploration.

In general, the knowledge engineer can greatly benefit from the line diagram of the lattice freely generated by the ordered set $(B, \leq)$. In fact, the line diagram of $\mathrm{FL}(\underline{2}+\underline{2})$ in Fig. 4 was used for the exploration of instruments. Unfortunately, there are only very few free lattices over partially ordered sets which can be "drawn" ([8]).

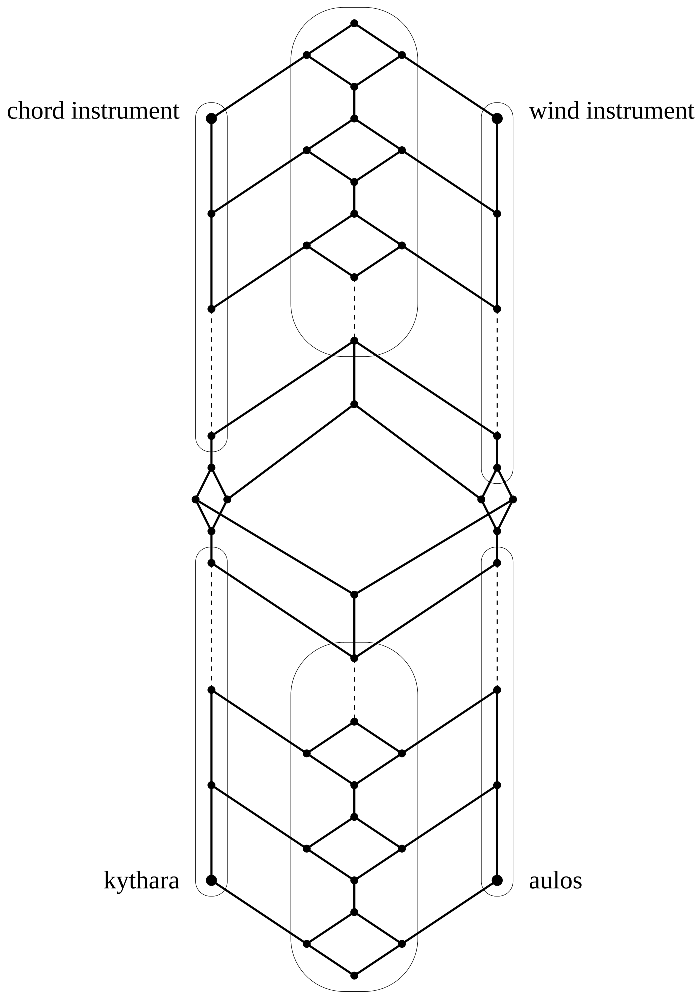

Fig. 4. The free lattice FL($\underline{2} + \underline{2}$).

4 Discussion

The main problem for any knowledge acquisition tool like concept exploration is the fact, that, in principal, the exploration may not terminate, because there are infinite lattices generated by only three elements. Hence any such exploration tool must provide the possibility to stop the exploration at any point such that the level of completeness gained so far is known. This was the crucial point for the decision to choose a breath first exploration: After the kth exploration, it is assured that all lattice elements denotable with a lattice term of complexity less or equal than k are generated, and that their hierarchical relationships are determined. A depth first exploration tool (like, e.g., distributive concept exploration) would risk, for arbitrary (e.g., non distributive) lattices, to explore some of the basic concepts so extensively that it will not reach the other ones. Unfortunately, this breath first approach forces the user to confirm all newly generated elements one by one. One might ask to construct another tool that allows to confirm or deny more than one element at a time. E.g., it might be based on A. Day's famous doubling method ([3]) for the computation of free lattices over partial lattices. Since the method is not able to reach all (finite) lattices, but only certain ones (the so-called bounded lattices), the right balance between applying the doubling method and factorizing by suitable congruences has to be kept. First experiments have shown that a too early factorization may prevent further doubling (i.e., generating potentially new elements) resulting in a break down of the exploration, while a too late factorization produces an abundance of potential elements which are not really different.

An interesting question for any exploration tool is, whether the exploration can be continued such that it will not terminate. In our algorithm, this is equivalent to the question, whether, for the intermediate result $B^{(k)}$, the free lattice $\mathrm{FL}(B^{(k)})$ is infinite or not. In [9], V. Slavík presents an effective algorithm for testing its finiteness. This free lattice is generated by concept exploration when all further questions are answered with "None" (i.e., with $Q := \emptyset$). On the other hand, it is always possible to terminate the algorithm in the next exploration step by answering "All" to all further questions.

Sometimes, the user does not want to start an exploration from the scratch, but he already has some background knowledge. In [5], B. Ganter extends attribute exploration to *background knowledge* formulated in predicate formulas. In particular, this comprises background implications as used in our algorithm. Hence his results can also be adapted to concept exploration. Since Ganter's algorithm can handle partial information, this also provides the possibility to use counterexamples. A *counterexample* (or *separating pair*) for $\bigwedge P \neq \bigwedge(P'')$ consists of an attribute belonging to the intent of $\bigwedge(P'')$ and an object not having the attribute but belonging to the extent of $\bigwedge P$. Unlike in distributive concept exploration – where counterexamples are used –, the capability for treating partial knowledge is necessary, because an object may belong to the supremum of two concepts although it does not belong to any of the two concepts. (The dual is true for attributes.) Hence, if complete knowledge is required, the user has to be asked for every newly generated supremum, which objects additionally belong to it. This makes the exploration unnecessarily complex.

Graphical support for the user showing the already computed hierarchy and the next potential elements is very desirable. By drawing only the potential elements of the next $\bigvee/\bigwedge$-exploration one could avoid the problem discussed at the end of the example, that most free lattices are not "drawable" (in the sense of Fig. 4). Unfortunately, there aren't any really satisfying fully automatic drawing algorithms for lattices yet. As with many other lattice oriented tools, concept exploration would certainly benefit from any progress in this field.

As for attribute exploration, concept exploration is designed to be an interactive tool, where a user acts as expert. In order to obtain a completely automatic tool, concept exploration may be combined with a subsumption algorithm (i. e., an (automatic) algorithm able to answer this type of questions) as it was done for the description logic $\mathcal{ALC}$ with attribute exploration ([1]) and distributive concept exploration ([14]), or resort to an online semantic dictionary.

References

1. F. Baader: Computing a minimal representation of the subsumption lattice of all conjunctions of concepts defined in a terminology. In: G. Ellis, R. A. Levinson, A. Fall, V. Dahl (eds.): *Proceedings of the International KRUSE Symposium: Knowledge Retrieval, Use and Storage for Efficiency.* Santa Cruz, CA, USA, August 11–13, 1995, 168–178
2. P. Burmeister: ConImp – A program for formal concept analysis. Technische Hochschule Darmstadt, 1987 (Latest version 1996 for MS DOS)
3. A. Day: Doubling constructions in lattice theory. *Can. J. Math.* **44**(2), 1992, 252–269
4. B. Ganter: Algorithmen zur Begriffsanalyse. In: B. Ganter, R. Wille, K. E. Wolff (eds.): *Beiträge zur Begriffsanalyse.* B. I.-Wissenschaftsverlag, Mannheim, Wien, Zürich 1987, 241–254
5. B. Ganter: Attribute exploration with background knowledge. *Proceedings of the conference on Order and Decision-Making.* Ottawa, Canada, August 5–9, 1996
6. B. Ganter, R. Wille: Formal Concept Analysis: Mathematical Foundations. Springer, Heidelberg 1997 (Translation of: Formale Begriffsanalyse: Mathematische Grundlagen. Springer, Heidelberg 1996)
7. U. Klotz, A. Mann: Begriffexploration. Diplomarbeit, TH Darmstadt 1988
8. I. Rival, R. Wille: Lattices freely generated by partially ordered sets: which can be "drawn"? J. Reine Angew. Math. **310**, 1979, 55–80
9. V. Slavík: Lattices with finite W-covers. (To appear)
10. J. F. Sowa: Conceptual structures: Information processing in mind and machine. Adison-Wesley, Reading 1984
11. G. Stumme: Attribute exploration with background implications and exceptions. In: H.-H. Bock, W. Polasek (eds.): *Data analysis and information systems. Statistical and conceptual approaches.* Studies in classification, data analysis, and knowledge organization **7**, Springer, Heidelberg 1996, 457–469
12. 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
13. G. Stumme: Knowledge acquisition by distributive concept exploration. In: G. Ellis, R. A. Levinson, W. Rich, J. F. Sowa (eds.): *Supplementary proceedings of the third international conference on conceptual structures*, Santa Cruz, CA, USA, August 14–18, 1995, 98–111
14. 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
15. G. Stumme: Distributive concept exploration – a knowledge acquisition tool in formal concept analysis. *Mathematics in artificial intelligence* (submitted)

16. G. Stumme: Free distributive completions of partial complete lattices. *Order* (submitted)
17. R. Wille: Restructuring lattice theory: an approach based on hierarchies of concepts. In: I. Rival (ed.): *Ordered sets.* Reidel, Dordrecht–Boston 1982, 445–470
18. R. Wille: Bedeutungen von Begriffsverbänden. In: B. Ganter, R. Wille, K. E. Wolff (eds.): *Beiträge zur Begriffsanalyse.* B. I.–Wissenschaftsverlag, Mannheim 1987, 161–211
19. 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
20. R. Wille: Conceptual structures of multicontexts. In: P. W. Eklund, G. Ellis, G. Mann (eds.): *Conceptual structures: Knowledge representation as interlingua.* LNAI **1115**, Springer, Heidelberg 1996, 23–39