Modelling Conceptual Schemata with Formal Concept Analysis

Uta Priss

Ostfalia University, Wolfenbüttel, Germany

Abstract. This paper discusses how to construct conceptual schemata (which are meant to provide conceptual information in a manner close to natural language, easy to memorise and mentally parse) from concept lattices which tend to present a more computational view of conceptual information. Different methods for constructing schemata from concept lattices (such as OR-definitions for reducing the number of attributes and implications of a concept lattice) are considered.

1 Introduction

As the name states, Formal Concept Analysis (FCA) provides a means for analysing concepts. While there are many applications for FCA, it is often easier to employ FCA for *computational* problems than to actually analyse concepts in a manner similar to how *natural language* is processed by humans. For the purpose of structuring and developing teaching materials it would be desirable if FCA could serve as a means for representing concepts in a manner that is close to how students learn domain knowledge. A representation would be desirable that is similar to relations amongst natural language words, for example hyponyms such as “poodle” and “dog”. Lattices that are automatically generated from natural language databases, however, such as WordNet or Roget’s Thesaurus tend to be more computational because their relations are not sufficiently precisely defined (Priss & Old 2010).

In this paper, we are introducing an approach for shifting between word-based natural language representations and more formal representations with FCA. For that purpose we are distinguishing *(conceptual) schemata* which utilise natural language words from *(conceptual) classes* which contain a set of formal contexts. Investigating the connections between schemata and classes is a *semiotic* task because it considers a relationship between words as representations of signs and concepts as meanings of signs. It is of interest to determine how well the information of a class is retained in a schema, how efficiently it is represented and how well a schema covers a class. The semiotic perspective and the relationship to educational research have been discussed elsewhere (Priss 2021a and 2021b) and are not further elaborated in this paper.

The notion of “schema” is influenced by Lakoff’s (1987) “image schema”. But the focus of schemata in this paper is on verbal description, not on images. The words or phrases that are defined by a schema and relate one schema to other schemata are called *head representamens* in this paper in analogy to the headwords of dictionary entries which are also called catchwords, keywords, subject headings, index terms or descriptors in other disciplines. From a semiotic view, head representamens are representamens of signs (Priss 2017). From a computational view, head representamens are just strings that are elements of a set. Head representamens are to be distinguished from other representamens which have an auxiliary function.

Head representamens can serve as building blocks for constructing compound representamens using the operations AND, OR and NOT. For example, head representamens for poodles might be "poodle" and "miniature poodle" whereas a compound representamen might be "poodle AND cute". Such operations are syntactically defined in schemata and semantically defined in classes with *interpretations* mapping schemata into classes[1]. The operations AND and OR for head representamens are similar but not identical to natural language "and" and "or" because, in natural language, "and" is sometimes used for an intersection (such as "dog and cute"), sometimes for a union (such as "dogs and cats") and "or" can be exclusive or inclusive. An interpretation should map an AND-operation amongst head representamens into a meet of concepts in a class, an OR-operation into a join of concepts or a construction involving a union of extensions and a NOT-operation into an extensional set difference.

The basics of FCA can be found in the textbook by Ganter & Wille (1999) and are not repeated in this paper. But it should be mentioned that a concept (a', a'') is called an *attribute concept of* a and a concept (o'', o') an *object concept*. The ordering amongst object concepts is called *object order*. Concepts that are not object concepts are called *supplemental concepts* in this paper. The extension of a supplemental concept equals the union of the extensions of its proper subconcepts. In this paper supplemental concepts are drawn as empty nodes in the Hasse diagrams. Each supplemental concept corresponds to a clause because for such a concept c with extension $ext(c)$ and intension $int(c)$ and the condition $\forall o_i \in ext(c) : \exists c_i < c : o_i \in ext(c_i)$ it follows that $\bigwedge(a_i \in int(c)) \Rightarrow \bigvee(a_i \mid \exists c_i : c_i < c, a_i \in int(c_i), a_i \notin int(c))$ is a clause. It is particularly interesting to consider whether some concepts always have to be supplemental with respect to background knowledge even if more objects are added to a context. An example for this feature is provided in the next section.

Some aspects presented in Section 2 which discusses a certain type of reduction of concept lattices have already been covered elsewhere, for example, by Ganter & Obiedkov (2016). Ganter (2019) discusses how to render an implication basis of a formal context more human readable by changing and grouping some of the implications and Lopez-Rodriguez et al. (2021) provide a means for determining core implications from a basis. In this paper, the focus is on reducing implications combined with representing some of the information by other means (as subconcept hierarchies or prototypical examples) if that renders the information more human readable. OR-reductions are also relevant for reducing a concept lattice to its AOC-poset (Osswald & Petersen, 2002) which consists only of the attribute and object concepts and possibly for feature models of Product Line Representations (Carbonnel et al. 2016).

The definitions of conceptual schemata, classes and interpretations in Section 3 are similar to a standard modelling with formal semantics, for example, Prediger's (1998) K-interpretations which map ordered sets of concept and relation names into power

[1] Contrary to standard formal semantics where interpretations map strings into sets, in this paper interpretations map head representamens into concepts.

context families. The aim of Prediger's work and others who extended it was to establish a connection between FCA and Conceptual Graphs and focused on logical properties. The focus of this paper is on the relationship between representamens and concepts in a more closed world setting. Most established FCA exploration and reduction methods tend to focus on reducing the lower parts of a lattice whereas in this paper mainly supplemental concepts in the upper part of a concept lattice are reduced. Thus, this paper draws on existing research but from a somewhat different perspective.

2 OR-Reduction

This section uses an example of a formal context and lattice from Ganter & Wille (1999) consisting of seven prototypical types of triangles and their defining properties (Fig. 1, left). In this example, the supplemental concepts (represented as empty nodes) must always be supplemental because every triangle must have exactly one of the attributes "acute", "obtuse" or "right" and either be equilateral or not or isosceles or not. Thus according to background knowledge about triangles, the object concepts describe actual examples of triangles whereas the extensions of the supplemental concepts must always be unions of the extensions of their subconcepts even if more triangles are added to the context. The lattice displays subconcept relationships for the types of triangles. If a student wants to learn about triangles, their types and their definitions, it would not be efficient to memorise all of the concepts of the lattice on the left side of Fig. 1. because it displays more a computational view than a natural language view.

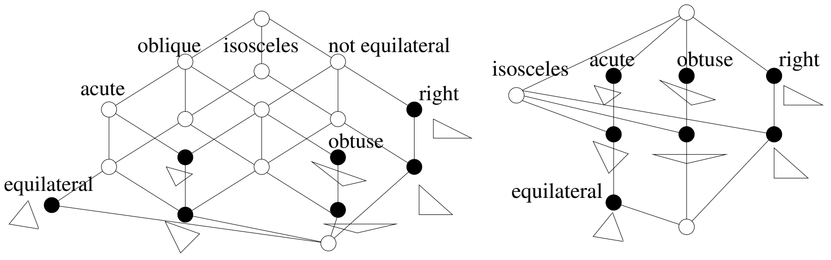


Fig. 1. A lattice of triangles (cf. Ganter & Wille (1999)) and its reduced form

The right side of Fig. 2 shows a reduced version of the lattice on the left. The attributes "oblique" and "not equilateral" have been removed because oblique represents "acute OR obtuse" and "not equilateral" is the negation of "equilateral". Normally in FCA *reducing* means to remove all attributes and objects from a context which are at attribute or object reducible concepts. Another form of reduction is to calculate an AOC-poset which only keeps object and attribute concepts and their ordering (Osswald & Petersen 2002). AOC-posets are compact and can be algorithmically produced (Berry et al. 2014). A lattice can be reconstructed from its AOC-poset if a clause is added for each concept that is neither an attribute nor an object concept. A disadvantage of AOC-posets is that conjunctions of attributes and therefore implications need not correspond to a single node and cannot easily be read from Hasse diagrams. This disadvantage is avoided by the reduction methods in this paper. Other means for reducing the size of concept lattices discussed in the literature tend to rely on statistical or probabilistic methods which cannot easily be reversed (cf. Priss & Old (2011) for an overview).

In this paper, only reducing attributes is of interest. Reducing attributes in the standard manner (called *AND-reduction* in this paper) changes the labelling of a concept lattice but not its structure. Removing attributes that are OR combinations (called *OR-reduction* in this paper) or NOT combinations (*NOT-reduction*) may change the concept lattice itself and reduce its size as demonstrated in Fig. 1. All of the following definitions focus on attributes and assume that the contexts are finite and clarified (or purified) which means that for any two attributes $a \neq b \Longrightarrow a' \neq b'$.

Definition 1. *An attribute a of a formal context (O, A, J) is called* OR-reducible *if a set $A^\star := \{a_1, ..., a_n\} \subseteq A$ exists with $a' = a'_1 \cup ... \cup a'_n$ and $a_i \in A^\star \Longleftrightarrow a'_i \subset a'$ and $\neg\exists b \in A : a'_i \subset b' \subset a'$.*

Definition 2. *For a formal context (O, A, J): For an OR-reducible attribute a, its* OR-definition *is provided by $a := a_1$ OR ... OR a_n for $a_i \in A^\star$. An attribute a with $\exists\{a_1, ..., a_n\} \subseteq A : a' = a'_1 \cap ... \cap a'_n$ is called* (AND-)reducible *with an* AND-definition *provided by $a := a_1$ AND ... AND a_n. An attribute a with $\exists b \in A : a' = O \backslash b'$ is called* NOT-reducible *with its* NOT-definition *provided by $a :=$ NOT b.*

Lemma 1. *If a is OR-reducible, then its set $A^\star := \{a_1, ..., a_n\}$ for its representation as a_1 OR ... OR a_n is uniquely determined. If a is NOT-reducible then its NOT-definition is uniquely determined.*

Proof: For $b \in A$ with $b' \subset a'$: if $b' \setminus \bigcup\{a'_i : a_i \in A^\star, a_i \neq b\} \neq \emptyset$, then $b \in A^\star$. Else $b' \subseteq \bigcup\{a'_i : a_i \in A^\star, a_i \neq b\} = a'$ and either $\exists a_i \in A^\star : b' \subset a'_i$ (thus $b \notin A^\star$) or $\neg\exists a_i \in A^\star : b' \subset a'_i$ (thus $b \in A^\star$). Thus $b \in A^\star$ or $b \notin A^\star$ is uniquely determined. NOT-definitions are unique because the context is clarified.

An attribute b that was removed during clarification can be considered a strong synonym or *SYN-definition* in the form of $a := b$. As mentioned above, AND-reduction corresponds to standard FCA $\wedge$-reduction. AND-definitions are not unique because often several possibilities exist to represent a $\wedge$-reducible concept as a meet of other concepts. OR-reduction focuses on attributes whereas standard FCA $\vee$-reduction focuses on objects. Thus, these two notions are different. An OR-reducible attribute must belong to a $\vee$-reducible concept, but not every $\vee$-reducible concept has an OR-reducible attribute. In fact OR- and NOT-reducible attributes need not exist at all in a lattice. In a similar manner, XOR-definitions could be declared as OR-definitions where the a'_i are pairwise disjoint.

Lemma 2. *i) The AND-definition of an attribute concept (a', a'') is a.*
ii) An object concept (o'', o') cannot be OR-reducible.
iii) A $\vee$-reducible concept that is an attribute concept (a', a'') and not an object concept can always be made OR-reducible by adding further attributes to the formal context.

Proof: i) Trivial. ii) Because o cannot be in the extension of proper subconcepts of (o'', o'). iii) Attributes $a_1, ..., a_n$ can be added so that each lower neighbour of (a', a'')

is an attribute concept (a_i', a_i''). Because the concept is not an object concept, Def. 1 is then fulfilled with $A^\star = \{a_1, ..., a_n\}$.

Thus Lemma 1 only states that an OR-definition is unique with respect to a fixed formal context. Turning each lower neighbour into an attribute concept is always possible but may not be the best strategy. For example in Fig. 1, oblique is definable as "acute OR obtuse" even though only one of its lower neighbours is an attribute concept.

Standard FCA implications only use logical AND. Implications that are formed with combinations of AND and OR are called (cumulated) clauses and are more complicated than standard implications. For example, there is no equivalent to the Duquenne-Guigues basis for clauses (Ganter & Obiedkov 2016). Because the requirements for an OR-definition are more specific than just a logical OR, the implications discussed in the next lemma are not standard FCA clauses. The lemma shows that if attributes are removed from a context as OR-definitions, some of the implications of the original context can be directly reconstructed from the OR-definitions (as background knowledge) and the implications of the reduced context.

Lemma 3. *For implications involving OR-definitions with $a := a_1$ OR ... OR a_n:*
i) $\forall a_i : a_i \to a$
ii) $a \to x \iff (a_1$ *OR ... OR* $a_n) \to x \iff (a_1 \to x)$ *and ... and* $(a_n \to x)$
iii) $x \to a \iff x \to (a_1$ *OR ... OR* $a_n) \iff (x \to a_1)$ *or ... or* $(x \to a_n)$
iv) $a_1 ... \, a_n \to x \Longrightarrow a \to x$
v) $\forall a_i : (x \to a_i y \Longrightarrow x \to ay)$

Proof: i) Because $a_i' \subset a'$. ii) $a_1' \cup ... \cup a_n' \subseteq x' \iff a_1' \subseteq x'$ and ... and $a_n' \subseteq x'$. iii) With $A^\star = \{a_1, ..., a_n\}$ it follows that $x' \subseteq a_1' \cup ... \cup a_n' \iff \exists a_i \in A^\star : x' \subseteq a_i'$ because otherwise $x \in A^\star$. iv) follows from ii) and the Armstrong rule of composition. v) because of transitivity of "$\to$".

Removing an OR-reducible attribute changes a lattice unless the attribute is also AND-reducible. It would be desirable to develop an efficient algorithm for reconstructing the implications of an original non-reduced context from the implications of a reduced context together with the OR-definitions. Lemma 3 contains some rules for such an algorithm, but the list is not complete and it is not clear whether it can be completed. A challenge for such an algorithm is that if several OR-definitions exist, they can mutually affect each other and thus cannot be processed in a linear sequence. It would be even more desirable if such an algorithm were to convert basis implications into basis implications. While all implications of an OR-reduced context are implications of its non-reduced context, applying Lemma 3 to an implication that belongs to a basis does not guarantee that it results in a basis implication of the non-reduced context. If efficiency is not an issue, then the implications of the non-reduced context can always be calculated by adding OR-definitions to a formal context as columns (for attributes) that are unions of other columns. For the purposes of developing schemata as discussed in the next section, it is sufficient to store those implications that cannot be easily reconstructed with Lemma 3 in a separate list in addition to the basis implications.

Presumably reconstructing the implications after NOT-reduction is even more complicated. OR-reduction does not change the object order of a lattice. But adding or deleting NOT-reducible attributes does change the object order as shown in Fig. 1. Therefore removal of NOT-reducible attributes may not in general be advisable. For the same reason, combining AND, OR and NOT in definitions is not even discussed in this paper. A further reason for removing OR-reducible attributes is because their existence is somewhat arbitrary. In the example in Fig. 1, oblique is OR-definable as “acute OR obtuse”. There are no similar attributes for “acute OR right” and “obtuse OR right”, but there could be. It is arbitrary which OR-definitions happen to exist as an attribute and which do not. Successive OR-reduction might reduce a concept lattice to its object ordering. Presumably, implications involving object concepts are particularly important whereas all other implications somewhat depend on how upper level concepts are labelled.

3 Conceptual Schemata and Classes

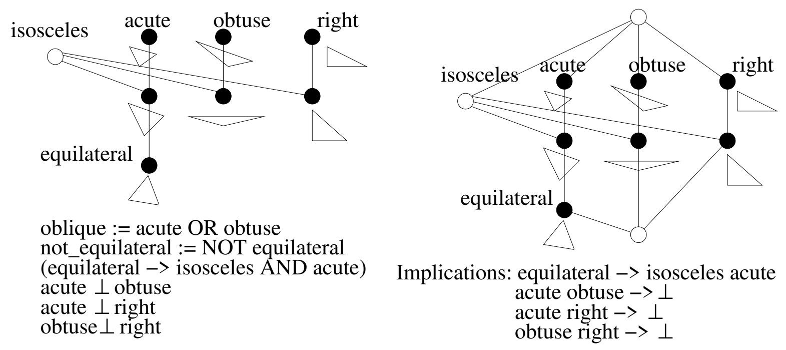
Learning is a complicated task that consists of memorising information but also acquiring skills and modes of thinking. With respect to conceptual knowledge different modes of thinking correspond to structuring content in a variety of manners: some information as concepts, some as implications, clauses or examples and some by techniques for deducing further information from the memorised information. The idea for conceptual schemata is that they present information in a format that is closer to how information would be structured for learning purposes. The role of conceptual classes is then to ensure that the information that is behind a schema is consistent and as complete as possible. The definitions in this section only provide a general framework and will need to be specified with further details for actual applications.

Fig. 2 shows the conceptual schema (on the left) for the concept lattice (on the right) of the example of Fig. 1. In this case the reduced lattice is already quite close to a schema, except that the top and bottom node are not necessary because they can be deduced. Further details about the schema are explained below. Evidence for the adequacy of the diagram on the left of Fig. 2 is provided by the fact that the German Wikipedia page about triangles contains basically the same image[2] for the “hierarchy of triangles”. Thus, the Wikipedia authors appear to consider it a suitable summary of knowledge about basic triangles. Students can memorise that diagram together with the definition of “oblique” and the fact that acute, right and obtuse are mutually exclusive. Students can then deduce further implications (such as “obtuse $\rightarrow$ oblique not_equilateral” and “equilateral $\rightarrow$ isosceles acute”) from the memorised information.

The following definitions specify the relationship between conceptual schemata and classes more precisely. The definitions are similar to standard definitions of formal semantics except that interpretations result in concepts instead of sets.

Definition 3. *A* (conceptual) class (O, A_L, J, N) *consists of a set O of formal objects, a set A_L of predicates (or “attributes”, formed according to some language L), a relation $J \subseteq O \times A_L$ with $oJa \iff (a(o)$ is true) and a set N of formal contexts with $(O_i, A_i, J_i) \in N$ for $O_i \subseteq O, A_i \subseteq A_L, J_i \subseteq J$ and $J_i \subseteq O_i \times A_i$. The set of all concepts that can be derived from any of the contexts is denoted by $\mathcal{C}(O, A_L, J, N)$, the set of all true statements that can be derived from any of the contexts by $\mathcal{T}(O, A_L, J, N)$.*

[2] https://de.wikipedia.org/wiki/Datei:Hierarchie.Dreiecke.png



oblique := acute OR obtuse
not_equilateral := NOT equilateral
(equilateral –> isosceles AND acute)
acute ⊥ obtuse
acute ⊥ right
obtuse⊥ right

Implications: equilateral –> isosceles acute
acute obtuse –> ⊥
acute right –> ⊥
obtuse right –> ⊥

Fig. 2. A conceptual schema (left) and a NOT-OR-reduced concept lattice (right)

A conceptual class is a set of formal contexts which are defined with respect to a common set of objects and attributes. While it would be possible to consider (O, A_L, J) a formal context itself, it may be too big to compute anything useful for it. Therefore concepts and implications are only computed for the contexts in N. It is possible for implications from one $n_1 \in N$ to contradict implications from another $n_2 \in N$, but that can be avoided by renaming attributes and is a matter of how the data of an application is modelled. In this paper, the language L contains expressions formed from unary predicates and the symbols AND, OR, NOT, $\rightarrow$ and $:=$, although the symbol "AND" is usually omitted as the default operation. In general, L can be more complex. The implications of a context are considered true for all objects of the context. In this paper, the examples of classes only consist of a single context where the predicates are unary attributes. But in general, Def. 3 encompasses a wide variety of possibilities. For example, a class can be a computer program of a declarative programming language or a relational database where the elements of O are tuples and a single predicate for each table determines whether or not a tuple exists in the table.

Definition 4. *A (conceptual) schema $(R_H, R_L, \mathcal{B})$ consists of a set R of head representamens, a set R_L of (representamen) expressions that are formed using head representamens and elements of a language L with $R_H \subseteq R_L$ and a set $\mathcal{B}$ of binary (representamen) relations $B \subseteq R_L \times R_L$.*

Further, non-mathematical conditions of conceptual schemata could be formulated, for example, that a schema should be coherent, focused on a single topic and have a certain minimal and maximal size. In this paper, the vocabulary of L is AND, OR and NOT and $\mathcal{B} := \{\rightarrow, \hookrightarrow, =, :=, \perp\}$ with ":=" $\subseteq R_H \times R_L$, $r_1 = r_2 \iff (r_1 \rightarrow r_2, r_2 \rightarrow r_1)$, $(r_1 := r_2 \implies r_1 = r_2)$ and $(r_1 \hookrightarrow r_2 \implies r_1 \rightarrow r_2)$. The relations are *definition* ($:=$), *strong synonymy* ($=$), *hyponymy* ($\rightarrow$ or edge in a Hasse diagram), *distant hyponymy* ($\hookrightarrow$ or arrow in a Hasse diagram) and *mutual exclusivity* ($\perp$). Two expressions are in a distant hyponymy relation if the exact hyponymy chain from one to the other is not specified. Further syntactic conditions need to be provided for actual applications. The Hasse diagram on the left of Fig. 2 is an abbreviation for some of the expressions and the hyponymy relation. Each node corresponds to an expression, either by itself ("acute") or as an AND-definition ("isosceles AND acute"), but only involving hyponyms, not distant hyponyms. The hyponymy relation is the transitive closure of the edges in the diagram. The hyponymy instance "equilateral $\rightarrow$ isosceles AND acute" is in brackets because it is redundant and can be read from the Hasse diagram.

Expressions and relations in a schema are meaningless strings that are manipulated according to the rules of a language. In order to evaluate whether expressions and relations are meaningful or true, they need to be mapped into classes using interpretations. The following definition specifies that head representamens and expressions are mapped onto concepts and relations onto true statements.

Definition 5. *A schema* $(R_H, R_L, \mathcal{B})$ *is* interpretable over *a class* (O, A_L, J, N) *if a set* $\mathcal{I}$ *of partial functions (called* interpretations*) can be defined so that* $\forall r \in R_L \; \exists i \in \mathcal{I} : i(r) \in \mathcal{C}(O, A_L, J, N)$ *and* $\forall B \in \mathcal{B} \; \forall b \in B \; \exists i \in \mathcal{I} : i(b) \in \mathcal{T}(O, A_L, J, N)$.

The definition does not provide any details with respect to how the interpretations are constructed. Further conditions must be supplied for specific applications. For example, if r_1 is a hyponym of r_2, it should be required that $i(r_1) <_n i(r_2)$ in some context n. Ideally, there should be exactly one interpretation for each formal context so that a head representamen can be assigned different concepts for different contexts but only at most one concept within a single context. Because different relation instances in a schema can utilise different interpretations, a certain amount of flexibility, ambiguity or fuzziness is possible. For example, a tomato can be a fruit in one context and a vegetable in another context. A schema should not just be interpretable, but also provide sufficient information about its underlying class as specified in the next definition. All examples of schemata in this paper fulfil Def. 6.

Definition 6. *A schema* $(R_H, R_L, \mathcal{B})$ covers *a class* (O, A_L, J, N) *under a set* $\mathcal{I}$ *of interpretations if each object concept is an interpretation of at least one representamen expression, if the object order and the relationship between an object concept and its attribute concepts is an interpretation of some instances of "$\rightarrow$" and* $\mathcal{T}(O, A_L, J, N)$ *can be logically derived from interpretations of representamen relations.*

4 Two Further Examples of Schemata and Classes

This section provides two further examples for developing conceptual schemata. The example in Fig. 3 is based on Ganter & Obiedkov (2016) where it is utilised for a discussion of clauses. According to the example, a driving license is passed exactly if both the theoretical and the driving part are passed and failed if one of them is failed. Ganter & Obiedkov argue that the 8 implications of the lattice do not represent the information in a natural manner. Instead they are suggesting to use 6 clauses and 2 implications. A difference between the clauses of Ganter & Obiedkov and the OR-definitions in this paper is that OR-defined attributes can be removed from the set of attributes before calculating the remaining implications. Thus the set of implications becomes smaller. The example in Fig. 3 shows that after defining the attribute "license fail" as "driving fail OR theory fail" and then removing it from the formal context, only four relation instances corresponding to implications are left. The first two correspond to both directions of an AND-definition (“license pass” as “driving pass AND theory pass”). The other two state that passing and failing each part of a driving test is mutually exclusive. Thus the schema on the left of Fig. 3 presents all relevant information and covers the class on the right in a succinct manner.

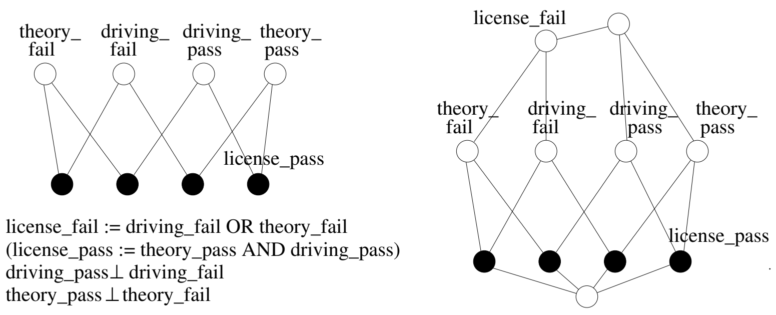

theory_ fail
driving_ fail
driving_ pass
theory_ pass
license_pass
license_fail := driving_fail OR theory_fail
(license_pass := theory_pass AND driving_pass)
driving_pass⊥ driving_fail
theory_pass ⊥ theory_fail
license_fail
theory_ fail
driving_ fail
driving_ pass
theory_ pass
license_pass

Fig. 3. A conceptual schema (left) for the driving license example (right)

The final example of this paper is based on Ganter & Wille (1999) and consists of properties of binary relations as defined in the following table[3].

property	definition
reflexive	$\forall a \in A : aRa$
irreflexive	$\forall a \in A : \neg aRa$
symmetric	$\forall a, b \in A : aRb \rightarrow bRa$
asymmetric	$\forall a, b \in A : aRb \rightarrow \neg(bRa)$
antisymmetric	$\forall a, b \in A : aRb$ and $bRa \rightarrow a = b$
transitive	$\forall a, b, c \in A : aRb$ and $bRc \rightarrow aRc$
semiconnex	$\forall a \neq b \in A : aRb$ or bRa
connex	$\forall a, b \in A : aRb$ or bRa

The concept lattice in Fig. 4 follows Ganter & Wille (1999). But it contains additional AND-defined attributes, such as “preorder := reflexive AND transitive”. It also contains some attributes about extreme cases. The example assumes that the relations R are defined as $R \subseteq S \times S$ for a non-empty set S. It may seem counter-intuitive that a relation can be both an order relation and an equivalence relation or symmetric and antisymmetric at the same time because that is only possible for extreme cases with attributes such as $R = S \times S$, $R = \{(i, j) \mid i = j\}$, $|S| = 1$ or $R = \{\}$. These attributes are included in the lattice.

Supplemental concepts are identified in Fig. 4 using background knowledge about binary relations. OR-definitions are only applicable to supplemental concepts. In the

[3] It should be remarked that the notions “semiconnex” and “connex” are used ambiguously in the literature. Sometimes “connex” is used instead of “semiconnex” and “strong connex” instead of “connex”.

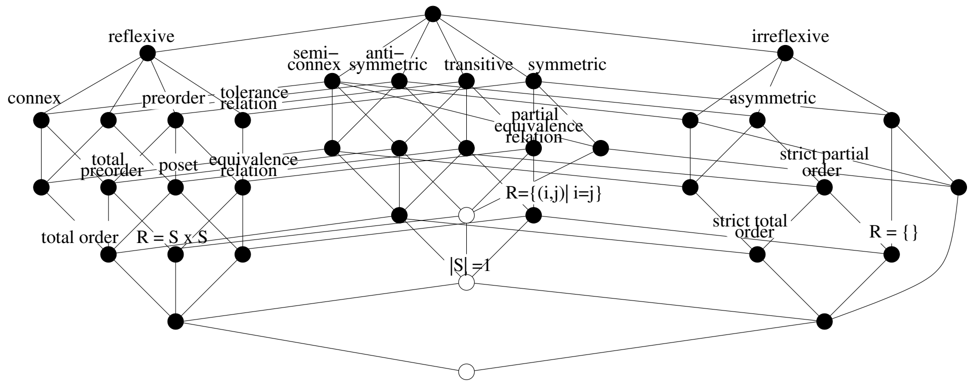
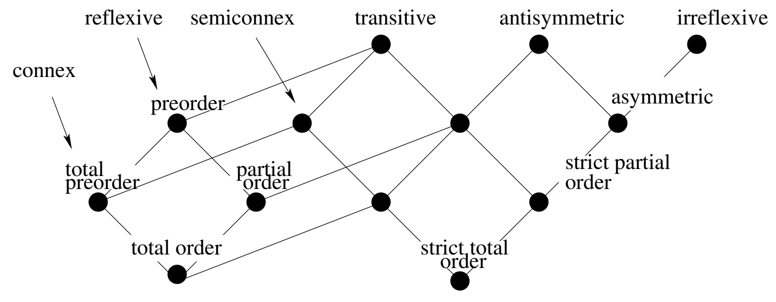


Fig. 4. A conceptual class of types of binary relations

previous examples, the concept lattices had supplemental concepts higher up in the lattice which could be removed using OR-definitions. This example only has very few supplemental concepts which are at the bottom of the lattice. These could be removed by introducing more attributes according to Lemma 2, but that does not reduce the complexity of the lattice significantly and increases the set of implications. Instead, the suggestion for developing a conceptual schema in this example is to extract meaningful parts of the lattice. Fig. 4 indicates a subdivision according to whether a relation is reflexive, irreflexive or neither. But such a division groups equivalence relations closely with order relations which are separated from strict orders. Thus it seems more natural to consider symmetric and NOT-symmetric as the main dividing factor for types of binary relations. Therefore, Fig. 5 and Fig. 6 divide the conceptual schema of binary relations into 3 parts: those that are not symmetric and tend to be orders, those that are symmetric and closely related to equivalence relations and the extreme cases at the bottom of the lattice which are antisymmetric and symmetric at the same time.



(transitive AND irreflexive –> asymmetric)
reflexive$\perp$ irreflexive

Fig. 5. Conceptual schema for types of binary relations: part 1

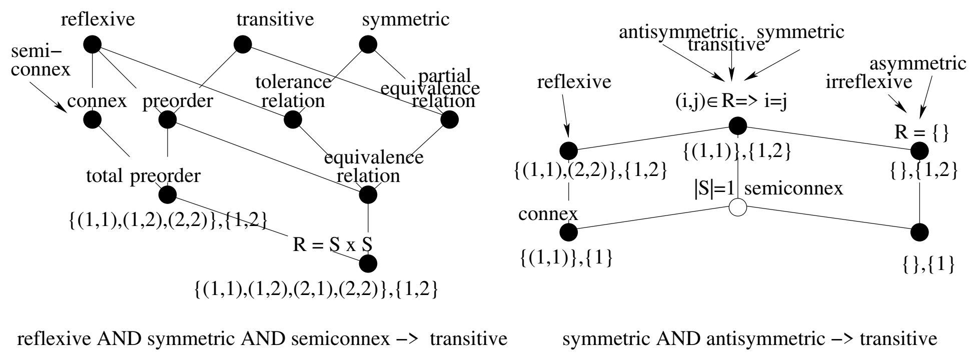


Fig. 6. Conceptual schema for types of binary relations: parts 2 and 3

The three parts of the schema in Fig. 5 and Fig. 6 are derived from the lattice of the class by deleting and restricting some attributes. Restricting means in this case that an attribute is replaced by its meet with another attribute. For example, in Fig. 5, the attribute "symmetric" is deleted and the attributes "reflexive", "semiconnex" and "connex" are replaced by their meet with "transitive" which results in distant hyponyms in the schemata. AND-definitions involving distant hyponyms cannot be read from the Hasse diagrams of the schemata. The left schema in Fig. 6 is derived by deleting the attributes "antisymmetric", "asymmetric", "irreflexive" and "semiconnex". The right schema is derived by restricting all attributes to their meet with "antisymmetric", "symmetric" and "transitive". For the non-restricted attributes, the hyponymy relation of the schema corresponds to the subconcept relation of the class and results in the same implications. The restricted attributes are considered distant hyponyms because their implications are not completely contained in the parts of the schema. All phrases in Fig. 5 are head representamens. In Fig. 6, the phrases and formulas that are written above the nodes are head representamens. The strings below the nodes represent prototypical examples and are not head representamens. The conceptual class contains four implications which are not just AND-definitions. Each of these four implications is included in the part of the schema where it is visible. In this case, even the schemata are still quite complex. But that is due to the subject manner. Learning all the relevant information about the head representamens in Fig. 5 and Fig. 6 will require a significant amount of time.

5 Conclusion

In summary, conceptual classes and schemata mutually influence each other. In some cases, it might be more suitable to extract a class from a schema using some form of conceptual exploration. In other cases, a schema can be constructed after reducing a class. The following strategies can be employed:

- Possibly splitting the context into smaller coherent subcontexts
- Conceptual exploration (for completing the set of objects and attributes)

- Purifying the lattice, adding SYN-definitions
- AND-reduction
- OR-reduction
- Further OR-reduction after adding attributes according to Lemma 2
- NOT-reduction

The motivation behind this strategy is that with respect to relationships between conceptual schemata and classes, the core content of a class is retained in its object concepts, their ordering and the AND-definitions of object concepts. The concepts that are above the object order may be less important, in particular if they are supplemental concepts, because they tend to represent attributes that may be expressible as OR-definitions.

References

1. Berry, A.; Gutierrez, A.; Huchard, M.; Napoli, A.; Sigayret, A. (2014). Hermes: a simple and efficient algorithm for building the AOC-poset of a binary relation. Annals of Mathematics and Artificial Intelligence, 72, 1, p. 45-71.
2. Carbonnel, J.; Bertet, K.; Huchard, M.; Nebut, C. (2016). FCA for software product lines representation: Mixing configuration and feature relationships in a unique canonical representation. In: Concept Lattices and their Applications (CLA'16), CEUR, p. 109-122.
3. Ganter, B.; Wille, R. (1999). Formal Concept Analysis. Mathematical Foundations. Springer.
4. Ganter, B.; Obiedkov, S. (2016). Conceptual Exploration. Springer.
5. Ganter, B. (2019). "Properties of Finite Lattices" by S. Reeg and W. Weiß, Revisited. In: Cristea et al. (eds) Formal Concept Analysis. ICFCA 2019. LNCS 11511, Springer, p. 99-109.
6. Lakoff, G. (1987). Women, Fire, and Dangerous Things. What Categories Reveal about the Mind. The University of Chicago Press.
7. Lopez-Rodriguez D., Cordero P., Enciso M., Mora A. (2021). Clustering and Identification of Core Implications. In: Braud et al. (eds.) Formal Concept Analysis. ICFCA 2021. LNAI 12733, p. 138-154.
8. Osswald, R.; Petersen, W. (2002). Induction of classifications from linguistic data. In: Proc. of the ECAI-Workshop on Advances in Formal Concept Analysis for Knowledge Discovery in Databases.
9. Prediger, S. (1998). Simple concept graphs: A logic approach. In: Mugnier et al. (eds.) Conceptual Structures: Theory, Tools and Applications. ICCS 1998. LNCS 1453, Springer, p. 225-239.
10. Priss, U.; Old, L. J. (2010). Concept Neighbourhoods in Lexical Databases. In: Kwuida; Sertkaya (eds.), Formal Concept Analysis. ICFCA 2010. LNCS 5986, Springer, p. 283-295.
11. Priss, U.; Old, L. J. (2011). Data Weeding Techniques Applied to Roget's Thesaurus. In: Wolff et al. (eds.) Knowledge Processing and Data Analysis. KPP 2007. LNAI 6581, Springer, p. 150-163.
12. Priss, U. (2017). Semiotic-Conceptual Analysis: A Proposal. International Journal of General Systems, 46, 5, p. 569-585.
13. Priss, U. (2021a). Diagrammatic Representation of Conceptual Structures. In: Braud et al. (eds.) Formal Concept Analysis. ICFCA 2021. LNAI 12733, p. 281-289.
14. Priss, U. (2021b). Conceptual Schemata as a Means for Structuring Teaching Materials. In: Concepts in Action: Representation, Learning, and Application (CARLA'21). Available at: https://www.conceptuccino.uni-osnabrueck.de/carla_workshop/carla_2021.html