Horn Representation of a Concept Lattice

Kamel Ben-Khalifa, Susanne Motameny

ZAIK, Center for Applied Computer Science,
University of Cologne, Germany

Abstract. Concept lattices are the central notion of formal concept analysis. They are applied in many different areas such as data mining, knowledge representation or ontology engineering and are subject to ongoing research. In order to understand better the nature of concept lattices it is useful to consider their links to other mathematical notions. For example, a concept lattice can be viewed as a special kind of poset or closure system. In this paper we consider another view of concept lattices by establishing a link to propositional formulas and a special closure property of relations. The main result is an elementary derivation of a Horn formula that uniquely represents a concept lattice based on prime implicates. Using the derived Horn representation, we reestablish the #P-completeness of the concept counting problem and find that the Horn representation is closely related to the stem base of a concept lattice.

1 Introduction

Concept lattices structure the formal concepts of their corresponding formal contexts in a hierarchical manner. They form a central element of formal concept analysis (FCA) and are used in many applications to structure object-attribute data in an informative way. To fully determine a concept lattice, it suffices to know one of the closure systems of concept intents or concept extents (as a formal concept is determined uniquely by its intent or extent, respectively). Our goal in this paper is to derive a conjunctive normal form (CNF) Horn formula which is satisfied exactly by the concept intents of a given context.
Relationships between FCA and Horn formulas have been pointed out by several other authors during the recent years. In [3] Horn functions of general closure systems are constructed (for example the closure systems of concept intents) in order to achieve a translation of certain notions of FCA (in case of [3] that of domination) into a logical framework. For our representation of concept lattices by Horn formulas we decided on different initial conditions and definitions so that the Horn formulas produced by our approach differ from those of [3] in decisive aspects. In [2] the authors show that if a formal context is viewed as a theory in the sense of propositional logic, the attribute implications holding in the context are equivalent to the empirical Horn approximation or Horn least upper bound (as defined in [18]) of that theory. Equivalence here means that the attribute sets respecting all the attribute implications are in one-to-one correspondence

with the tuples of the resulting Horn theory. Thus a link between FCA and knowledge compilation is established. The authors stop at this point, but their result eventually entails an equivalence of concept lattices and certain Horn formulas. In contrast to [2] where the authors introduce an artificial attribute that is satisfied by no object, we can do without such extensions and provide a direct translation of FCA notions into propositional logic.
The paper is organized as follows: In Section 2 we recall the necessary basic definitions and facts from FCA and propositional logic. The derivation of the Horn formula is accomplished in Section 3 and we discuss the relationship of our result to attribute implications and the counting of concepts in Section 4. We close the paper with a summary in Section 5.

2 Basic Definitions and Facts

We now give the basic definitions and facts that we use from formal concept analysis and propositional logic. The notions from propositional logic are well known and only repeated to make the paper self contained. Concerning the FCA terminology we follow the notations from [5].

2.1 Formal Concept Analysis

Let G be a set of objects, M a set of attributes and $I \subseteq G \times M$ an incidence relation between G and M. For $g \in G$ and $m \in M$, gIm is read as "object g has the attribute m" or dually "attribute m is satisfied by object g". The triple $K = (G, M, I)$ is called a *(formal) context.* We consider the usual derivation operators: For $A \subseteq G$ and $B \subseteq M$

$$\begin{aligned} A' &= \{m \in M \mid gIm \; \forall g \in A\} \\ B' &= \{g \in G \mid gIm \; \forall m \in B\}. \end{aligned}$$

A set $B \subseteq M$ is an *intent* of (G, M, I) if and only if $B = B''$. M is always an intent of (G, M, I) as all objects in the empty set trivially have all attributes. Also the set $\{g\}'$ is an intent for each object g and is called *object intent* of $g \in G$. It is well known that every intent of (G, M, I) (except for M) is an intersection of object intents. This fact forms the basis for our derivation of the Horn representation of a concept lattice.

2.2 Propositional Logic

We first introduce the syntactical objects of propositional logic as far as needed for the later discussion and then define their semantics. A lower case letter or an indexed lower case letter, x or x_i, respectively, is a *variable.* We will only use variables that can take values in {0,1} and call these *propositional* *variables*. Furthermore, we use the symbols $\vee, \wedge$, and $^{-}$ as well as (and). If x is a propositional variable, then $\bar{x}$ is called its *negation*. A *literal* is a propositional variable (called *positive* literal) or its negation (called *negative* literal). A *clause* is a string of the form $(x_1 \vee x_2 \vee \cdots \vee x_n)$, where each $x_i, i \in \{1, \ldots, n\}$ is a literal. A clause is *Horn* if it contains at most one positive literal. It is called *definite Horn* if it contains exactly one positive literal. We will denote the set of literals of a clause C by $\mathrm{lit}(C)$. A *formula in conjunctive normal form (CNF)* is a string of the form

$$C_1 \wedge C_2 \wedge \cdots \wedge C_n,$$

where each $C_i, i \in \{1, \ldots, n\}$ is a clause. If every clause in a CNF formula F is Horn, F is called a *Horn formula*.
Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of n propositional variables with a fixed numbering and $L_X = \{x, \bar{x} \ : \ x \in X\}$ be the set of literals over X. An *assignment* is a mapping $\mathcal{A} : X \longrightarrow \{0, 1\}$. An assignment $\mathcal{A}$ can be extended to an assignment $\bar{\mathcal{A}} : L_X \longrightarrow \{0, 1\}$ by setting

$$\bar{\mathcal{A}}(x) = \begin{cases} \mathcal{A}(x) : x \text{ is a positive literal} \\ 1 - \mathcal{A}(x) : x \text{ is a negative literal.} \end{cases} \tag{1}$$

By Equation (1), an assignment $\bar{\mathcal{A}}$ on the set of literals is determined uniquely by an assignment $\mathcal{A}$ on the set of variables. Therefore we will represent an assignment $\bar{\mathcal{A}} : L_X \longrightarrow \{0, 1\}$ by the underlying assignment $\mathcal{A} : X \longrightarrow \{0, 1\}$. With this convention we will use the notation $\mathcal{A}(x)$ for $x \in L_X$ (instead of $\bar{\mathcal{A}}(x)$) throughout the remainder of this paper. An assignment $\mathcal{A}$ *satisfies* a clause C over X if there is at least one literal x in C with $\mathcal{A}(x) = 1$. We then call $\mathcal{A}$ a *model* of C and write $\mathcal{A} \models C$. An assignment $\mathcal{A}$ is called a model of a CNF formula F over X if $\mathcal{A}$ satisfies every clause in F. In this case we write $\mathcal{A} \models F$. The set of all models of a formula F is denoted by $\mathrm{Mod}(F)$. We write $\mathcal{A} \not\models F$ if $\mathcal{A}$ is not a model of F. An assignment $\mathcal{A}$ can be represented by a tuple $t \in \{0, 1\}^n$ by setting $t_i = \mathcal{A}(x_i)$ for each variable x_i. This representation is unique and we will identify assignments and their representing tuples in the following. In particular, we will write $t(x)$ instead of $\mathcal{A}(x)$ and $t \models F$ instead of $\mathcal{A} \models F$. Throughout the remainder of the paper we will refer to a set of tuples $R \subseteq \{0, 1\}^n$ as an *n-ary relation* (or simply *relation* if the arity is not of interest). When we refer to the conjunction of tuples we mean the entry-wise application of the logical $\wedge$-operation, for example

$$(1, 0, 0, 1) \wedge (0, 1, 0, 1) = (0, 0, 0, 1).$$

The essential notion needed in the following is that of a *prime implicate*. We will use prime implicates to construct a CNF formula that is equivalent to a given relation.

Definition prime implicate: Let R be a relation. A clause C is called *implicate* of R if $R \subseteq \mathrm{Mod}(C)$. C is called *prime implicate* if no clause D with $\mathrm{lit}(D) \subset \mathrm{lit}(C)$ is an implicate of R.

It is a well known fact that the CNF formula F that consists of the conjunction of all prime implicates of a relation R is equivalent to R in the following sense:

$$t \in R \Longleftrightarrow t \models F \text{ (that is } R = \mathrm{Mod}(F)\text{)} \tag{2}$$

3 Derivation of the Horn Representation

With the definitions from Section 2 we are now ready to derive the Horn representation of a concept lattice. We first note a special property of Horn clauses.

Lemma 1: Let $t^{(i)}, i \in \{1, \ldots, k\}$ be k n-ary tuples and C a Horn clause in the propositional variables $x_1, x_2, \ldots, x_n$. It holds that

$$\forall i \in \{1, \ldots, k\}\ t^{(i)} \in \mathrm{Mod}(C) \implies \bigwedge_{i=1}^{k} t^{(i)} \in \mathrm{Mod}(C).$$

This is a well known fact which follows for example from [1], Lemma 3. A relation R is called $\wedge$-*closed* if with every two tuples $t^{(1)}, t^{(2)} \in R$ also $t = t^{(1)} \wedge t^{(2)}$ is an element of R. We continue with an important and well known fact about $\wedge$-closed relations which dates back to [10, 15] and can also be found in [9], [11], and [13] in a more general form than the one we prove. Our formulation however is sufficient and more appropriate for our considerations.

Proposition 1: R is $\wedge$-closed if and only if all prime implicates of R are Horn clauses.

Proof: Let R be $\wedge$-closed and C an arbitrary prime implicate of R. Assume that C contains at least two positive literals y and z. If we denote the disjunction of the remaining literals by D we have $C = (y \vee z \vee D)$. By definition $R \subseteq \mathrm{Mod}(C)$. If every tuple in R was a model of D, then D would be an implicate of R. This cannot happen because C is a prime implicate. Therefore R contains a nonempty set of tuples T with $T \cap \mathrm{Mod}(D) = \emptyset$. To fulfill the condition $t \models C$ for a $t \in T$, either $t(y) = 1$ or $t(z) = 1$ must hold. If we had $t(y) = 1$ (or analogously $t(z) = 1$) for all $t \in T$, then $(y \vee D)$ (or $(z \vee D)$, respectively) would be an implicate of R and C could not be a prime implicate. This implies the existence of tuples $t^{(y)}$ with $t^{(y)}(y) = 1$ and $t^{(y)}(z) = 0$ and $t^{(z)}$ with $t^{(z)}(y) = 0$ and $t^{(z)}(z) = 1$ in T. Let $t^* = t^{(y)} \wedge t^{(z)}$. For t^* it holds that

$$\begin{aligned} t^* &\not\models D \\ t^*(y) &= 0 \\ t^*(z) &= 0 \end{aligned}$$

and therefore $t^* \notin \mathrm{Mod}(C)$. This means that $t^* \notin R$, which is a contradiction to the precondition that R is $\wedge$-closed.
Now let R be a relation whose prime implicates $C_j, j \in \{1,\dots,k\}$ are all Horn. Consider two arbitrary tuples $t^{(1)}$ and $t^{(2)}$ in R. Let $t^* = t^{(1)} \wedge t^{(2)}$. According to Lemma 1 $t^{(1)} \in \mathrm{Mod}(C_j)$ and $t^{(2)} \in \mathrm{Mod}(C_j)$ implies $t^* \in \mathrm{Mod}(C_j)$ for all $j \in \{1,\dots,k\}$. From $R = \mathrm{Mod}(\bigwedge_{j=1}^{k} C_j) = \bigcap_{j=1}^{k} \mathrm{Mod}(C_j)$ it follows that $t^* \in R$ which proves the $\wedge$-closedness of R. $\square$

For the purpose of our derivation of a Horn representation of a concept lattice we now provide an obvious translation of a formal context $K = (G, M, I)$ into a $|M|$-ary relation. Let $m_1, m_2, \dots, m_{|M|}$ be the attributes in the cross table of K. We associate with every attribute m_i a propositional variable x_i and replace the crosses in the table by ones and the empty cells by zeroes. The row corresponding to an object $g \in G$ now represents a tuple from $\{0,1\}^{|M|}$ denoted by t^g. Finally, the tuple t^M which contains only ones is added to the table. The relation gained from a context K by this translation is denoted by R_K. Table 1 illustrates the translation by an example.

K	m_1	m_2	m_3	m_4
a		X	X	
b				X
c		X		X
d	X	X		X

R_K	x_1	x_2	x_3	x_4
t^a	0	1	1	0
t^b	0	0	0	1
t^c	0	1	0	1
t^d	1	1	0	1
t^M	1	1	1	1

Table 1. A formal context K and its corresponding relation R_K

Remark: By adding t^M to R_K, we ensure that we arrive at a definite Horn representation (called pure Horn representation in [3]), that is each clause in the representation contains exactly one positive literal. This is what makes our approach different from [3] and turns out to yield a very nice correspondence to attribute implications as described in Section 5.

The result of our translation procedure is a $|M|$-ary relation R_K which contains one tuple for each object intent as well as the tuple t^M that corresponds to the intent consisting of all attributes. Recall that all other concept intents are intersections of object intents. The intersection of object intents translates exactly to the conjunction of their corresponding tuples. Consider for example the objects a and c from Table 1. Their object intents are $\{a\}' = \{m_2, m_3\}$ and $\{c\}' = \{m_2, m_4\}$. The concept intent resulting from their intersection corresponds to the tuple $t = t^a \wedge t^c$:

$$\begin{array}{rcccc} \{a\}' \cap \{c\}' & = & \{m_2, m_3\} \cap \{m_2, m_4\} & = & \{m_2\} \\ t^a \wedge t^c & = & (0,1,1,0) \wedge (0,1,0,1) & = & (0,1,0,0) \end{array}$$

With this procedure we can inductively construct to each concept intent a corresponding tuple by simply computing the conjunction of the tuples belonging to the involved object intents. We add all thus constructed tuples to R_K and call the resulting relation $\hat{R}_K$. Obviously there is a bijection between concept intents of $K = (G, M, I)$ and tuples in $\hat{R}_K$. Also, because we collected all possible conjunctions of tuples from R_K in $\hat{R}_K$, $\hat{R}_K$ is $\wedge$-closed. In fact it is exactly the $\wedge$-closure of R_K. By Proposition 1, $\hat{R}_K$ can be represented by the Horn formula F_K consisting of its prime implicates. This formula is the Horn representation of the concept lattice we wanted to derive.

Definition Horn representation: Let $C_1, C_2, \dots, C_k$ be the prime implicates of $\hat{R}_K$. The formula

$$F_K = C_1 \wedge C_2 \wedge \cdots \wedge C_k$$

is called *Horn representation* of the concept lattice defined by $K = (G, M, I)$.

Note that we use all prime implicates of $\hat{R}_K$ in the above definition. In general, F_K can contain redundant prime implicates. If these are removed, the obtained formula is still equivalent to $\hat{R}_K$ in the sense of (2). We use all prime implicates in our definition of a Horn representation for reasons of uniqueness. The Horn representation of the context from Table 1 for example is

$$F_K = (\bar{x}_1 \vee x_2) \ \wedge \ (\bar{x}_1 \vee x_4) \ \wedge \ (\bar{x}_3 \vee x_2) \ \wedge \ (\bar{x}_3 \vee \bar{x}_4 \vee x_1).$$

The tuples of $\hat{R}_K$ correspond to the concept intents of K bijectively. The hierarchical order of the concept intents is given by $B_1 \leq B_2 \Longleftrightarrow B_1 \supseteq B_2$ for two concept intents B_1 and B_2 (see [5]). It corresponds to the following order on $\hat{R}_K$:

$$t^{(1)} \leq t^{(2)} \Longleftrightarrow t^{(1)} \wedge t^{(2)} = t^{(2)}$$

The concept extent corresponding to the intent represented by the tuple $t \in \hat{R}_K$ is denoted by t' and can be determined by

$$t' = \{g \in G \mid t^g \wedge t = t\}.$$

In theory, F_K can be computed from R_K directly, that is one does not need to determine the (possibly exponentially sized) $\wedge$-closure of R_K.

Proposition 2: The prime implicates of $\hat{R}_K$ are exactly the Horn prime implicates of R_K.

Proof: Let C be a Horn prime implicate of R_K. Because of $R \subseteq \mathrm{Mod}(C)$ it follows from Lemma 1 that $\hat{R}_K \subseteq \mathrm{Mod}(C)$. Therefore C is an implicate of $\hat{R}_K$. C is also a prime implicate of $\hat{R}_K$, because for all D with $\mathrm{lit}(D) \subset \mathrm{lit}(C)$ there is a tuple $t \in R_K$ (and in particular $t \in \hat{R}_K$) satisfying $t \not\models D$.

Let now conversely C be a prime implicate of $\hat{R}_K$. According to Proposition 1 C is Horn. Because of $\hat{R}_K \supseteq R_K$, C is an implicate of R_K. Assume that C is no prime implicate of R_K. Then there is an implicate D of R_K with $\text{lit}(D) \subset \text{lit}(C)$. C is a prime implicate of $\hat{R}_K$ and therefore there is a tuple $t \in \hat{R}_K$ that is no model for D. This tuple is a conjunction of tuples $t^{(i)} \in R_K$ and as C is Horn so is D. Lemma 1 implies the existence of a tuple $t^{(i^*)} \in R_K$ with $t^{(i^*)} \notin \text{Mod}(D)$. But this is a contradiction to our presupposition of D being an implicate of R_K, which completes the proof. $\square$

Proposition 2 shows that it suffices to compute the prime implicates of R_K, select those that are Horn, and their conjunction is exactly the Horn representation F_K. However, there are several complexity results from propositional logic that imply limitations for the practical implementation of such a procedure. From a result shown in [12] it follows that there exist contexts $K = (G, M, I)$ containing only join-irreducible objects (called characteristic models in [12]) so that the shortest Horn formula equivalent to $\hat{R}_K$ is exponential in $|G|$. Thus a method running in input polynomial time that computes the Horn representation of any concept lattice from its context does not exist. Another result in [8] states that a Horn formula that is equivalent to an $\wedge$-closed n-ary relation R can be found in $O(|R|\, n\, (|R| + n))$ time. Moreover, the constructed formula contains no more than $|R|\, n$ clauses. This means that we can obtain a Horn formula that is equivalent to F_K in $O(|\hat{R}_K|\, |M|\, (|\hat{R}_K| + |M|))$ time. However, the input to this algorithm would be the relation $\hat{R}_K$ which may be exponentially larger than R_K. Thus, the algorithm from [8] cannot be used to generate a Horn formula representing the concept lattice of a given context in time polynomial in the size of the context. As in general large relations can be equivalent to short Horn formulas, the time complexity of such an approach would not even be polynomial relative to the input and output size. In [13] it is proved that the computation of a Horn formula that is equivalent to $\hat{R}_K$ and takes R_K as input is at least as difficult as the generation of all the transversals of a hypergraph. To our knowledge the complexity of this hypergraph problem is still open.

4 Discussion

In this section we discuss the relationship of our result to certain areas and problems of Formal Concept Analysis.

4.1 Counting of Concepts

In the previous section we derived a Horn formula F_K that represents the intent set of a formal context $K = (G, M, I)$ and therefore is also a representation of its concept lattice. If we count the models of F_K we actually count the concepts of $K = (G, M, I)$ and our result thus offers a possibility to determine the size of a concept lattice. However, in [4] a theorem is proven which implies that the counting of the models of a definite Horn formula is $\#P$-complete in general, that is for a definite Horn formula there is a nondeterministic Turing machine whose number of accepting computation paths equals the number of models of that formula and that runs in polynomial time (see for example [16] for a rigorous definition of the complexity class $\#P$). This is in accordance with [14], where the $\#P$-completeness of the determination of the size of a concept lattice is also proved. The fact that the counting of concepts is $\#P$-complete also follows easily from the $\#P$-completeness of the counting of ideals of a poset which was proved in [17]. If $(P, \leq)$ is an arbitrary poset, then the formal context $(P, P, \not\leq)$ contains as many concepts as there are ideals in $(P, \leq)$. This also answers the question raised in [14] whether the $\#P$-completeness also holds for distributive concept lattices because the concept lattice of $(P, P, \not\leq)$ is always distributive.

4.2 Attribute Implications

We defined R_K in such a way that $t^M \in R_K$. This causes all clauses of F_K to contain exactly one positive literal and they can be equivalently written as implications:

$$(\bar{x}_1 \vee \bar{x}_2 \vee \cdots \vee \bar{x}_k \vee y) \Longleftrightarrow x_1 \wedge x_2 \wedge \cdots \wedge x_k \longrightarrow y$$

If we write instead of the propositional variable x_i the attribute m_i which it represents, we can translate our Horn representation to a set of attribute implications. We recall some basic definitions from FCA concerning attribute implications (see [5]).

Definition attribute implications: The implication $X \longrightarrow Y$ is equivalent to $Y \subseteq X''$. A set $T \subseteq M$ is said to *respect* an implication $X \longrightarrow Y$ if $X \not\subseteq T$ or $Y \subseteq T$. A set of attributes T respects a set of implications $\mathcal{L}$ if it respects every single implication in $\mathcal{L}$.
An implication $X \longrightarrow Y$ *follows* from an implication set $\mathcal{L}$ if every set $T \subseteq M$ that respects $\mathcal{L}$ also respects $X \longrightarrow Y$. An implication $X \longrightarrow Y$ *holds* in a context (G, M, I) if every concept intent of (G, M, I) respects the implication $X \longrightarrow Y$.

Because all concept intents of $K = (G, M, I)$ respect all the implications obtained from the prime implicates of $\hat{R}_K$ by definition, the set of all these implications is complete for K, that is all attribute implications holding in K follow from it. Merging implications with the same premise which can generally occur in F_K, we obtain the set $\mathcal{H}$ of attribute implications. We illustrate this procedure using the Horn representation of the context K from Table 1:

Horn Representation	Implications	Merged Implications
$F_K = \bar{x}_1 \vee x_2 \wedge$	$\{m_1 \longrightarrow m_2$	$\mathcal{H} = \{m_1 \longrightarrow m_2, m_4$
$\bar{x}_1 \vee x_4 \wedge$	$m_1 \longrightarrow m_4$	$m_3 \longrightarrow m_2$
$\bar{x}_3 \vee x_2 \wedge$	$m_3 \longrightarrow m_2$	$m_3, m_4 \longrightarrow m_1\}$
$\bar{x}_3 \vee \bar{x}_4 \vee x_1$	$m_3, m_4 \longrightarrow m_1\}$	

Unfortunately $\mathcal{H}$ can contain redundant implications because the set of prime implicates it is based on can be redundant. Recall that an implication base of K is a set of implications that is complete for K and minimal in the sense that it does not contain redundant implications. Therefore, if the redundant implications are removed from $\mathcal{H}$ an implication base of K is the result. In FCA the usual implication base of interest is the stem base or Duquenne-Guigues base first described in [6]. Its definition is based on the notion of a pseudo intent (see [5]).

Definition pseudo intent: A subset $P \subseteq M$ of attributes is called *pseudo intent* of a context (G, M, I) if

(1) $P \neq P''$
(2) $\forall Q \subset P, Q$ pseudo intent $\Longrightarrow Q'' \subset P$.

In the above definition M is considered to be finite. The stem base is then defined as:

Definition stem base: The set of implications

$$\mathcal{L} = \{P \longrightarrow P'' \mid P \text{ is a pseudo intent of } (G, M, I)\}$$

is an implication base of the context (G, M, I) and is called *stem base* of the context.

When comparing $\mathcal{H}$ to the stem base $\mathcal{L}$ of $K = (G, M, I)$, we can see immediately that for each premise P of the stem base there must be a premise X of $\mathcal{H}$ satisfying $X \subseteq P$. This is because the premises in $\mathcal{H}$ correspond to prime implicates and every implication holding in $K = (G, M, I)$ is an implicate of $\hat{R}_K$ and thus must contain a prime implicate. We can see this already in our simple example:

$$\begin{array}{l|l} \mathcal{H} = \{m_1 \longrightarrow m_2, m_4 & \mathcal{L} = \{m_1 \longrightarrow m_2, m_4 \\ \quad m_3 \longrightarrow m_2 & \quad m_3 \longrightarrow m_2 \\ \quad m_3, m_4 \longrightarrow m_1\} & \quad m_2, m_3, m_4 \longrightarrow m_1\} \end{array}$$

The premise of the last implication in $\mathcal{L}$ is a superset of the premise of the last implication in $\mathcal{H}$. We have already seen that each pseudo intent contains the premise of an implication in $\mathcal{H}$. Moreover, we have the following conjecture.

Conjecture: Each premise in $\mathcal{H}$ is contained in a pseudo intent.

We believe that this conjecture is true but could not find a rigorous proof for this claim up to now. However, Proposition 3 shows that there is a subset of premises in $\mathcal{H}$ each of which is not only contained in a pseudo intent but is a pseudo intent itself.

Proposition 3: Let $X \longrightarrow Y$ be an implication in $\mathcal{H}$. If $X \cup Y$ is closed (that is $(X \cup Y)'' = X \cup Y$) then X is a pseudo intent.

Proof : Let $R \subset X$. Because $X \longrightarrow y$ is a prime implicate for all $y \in Y$ it follows that $R \not\longrightarrow y$ and thus $R'' \cap Y = \emptyset$. Because $X \longrightarrow Y$ is equivalent to $Y \subseteq X''$ we obtain $X \cup Y \subseteq X''$. On the other hand we have $X'' \subseteq (X \cup Y)'' = X \cup Y$ and thus $X'' = X \cup Y$ which implies $R'' \subset X$. Therefore X satisfies the definition of a pseudo intent. $\square$

For the case when $X \cup Y$ is not closed we have not been able to prove that X is contained in a pseudo intent nor could we construct a counterexample. Thus, the question whether our conjecture is true or not is currently open.
We close this section with a remark about the minimality of the Duquenne-Guigues base. The comparison of $\mathcal{H}$ and $\mathcal{L}$ from our example context shows that both contain the same number of implications but the last implication in $\mathcal{H}$ has a shorter premise. In general, the stem base $\mathcal{L}$ allows for a more compact representation (that is one that uses less attributes), which can be computed from $\mathcal{L}$ in the following way: First $\mathcal{L}$ is translated into a Horn formula $F_{\mathcal{L}}$ (for example by using the translation of F_K to $\mathcal{H}$ demonstrated above backwards). Then for each clause it is checked whether it contains a subclause that is an implicate of $F_{\mathcal{L}}$. If so, the clause is replaced by that subclause. When iterated, this procedure yields a formula consisting of prime implicates that is equivalent to $\mathcal{L}$ and the translation back into a set of implications is the desired compact representation of $\mathcal{L}$. The basics of such a method that requires at most quadratic time in the length of the input formula $F_{\mathcal{L}}$ can be found for example in [7].

5 Summary

We suggested a representation of a concept lattice by a special Horn formula. This formula was constructed as a conjunction of the prime implicates of a relation R_K that was derived from the formal context $K = (G, M, I)$. Theoretically, the Horn representation can be computed from the context directly but several results from propositional logic imply limitations for the practical implementation. We believe that despite of this drawback our result is interesting insofar as it reveals a close relationship between FCA and propositional logic that can be exploited to tackle research questions in both scientific areas. By using the Horn representation we reproduced the $\#P$-completeness of the concept counting problem. Also we pointed to a close relationship between the Horn representation and the stem base of a formal context.
In our future research we hope to find more links between FCA and propositional logic so that one can use the knowledge and formalism in the one area to solve problems and gain insights in the other one.

6 Acknowledgements

We would like to thank Ulrich Faigle and three anonymous referees whose suggestions helped to improve the paper.

References

1. D. Angluin, M. Frazier, and L. Pitt. Learning conjunctions of Horn clauses. *Machine Learning*, 9:147–164, 1992.
2. J. L. Balcázar and J. Baixeries. Discrete deterministic data mining as knowledge compilation. *Workshop Discr Math and Data Mining at SIAM DM Conference*, 2003.
3. A. Berry, E. SanJuan, and A. Sigayret. Generalized domination in closure systems. *Discrete Applied Mathematics*, 154:1064–1084, 2006.
4. N. Creignou and M. Hermann. Complexity of generalized satisfiability counting problems. *Information and Computation*, 125:1–12, 1996.
5. B. Ganter and R. Wille. *Formal Concept Analysis, Mathematical Foundations.* Springer, 1999.
6. J.-L. Guigues and V. Duquenne. Familles minimales d'implications informatives resultant d'un tableau de données binaires. *Math. Sci. Humaines*, 95:5–18, 1986.
7. P. L. Hammer and A. Kogan. Horn functions and their DNFs. *Information Processing Letters*, 44:23–29, 1992.
8. J.-J. Herbrard and B. Zanuttini. An efficient algorithm for Horn description. *Information Processing Letters*, 88:177–182, 2003.
9. J. Hooker. *Logic-Based Methods for Optimization.* John Wiley & Sons, 2000.
10. A. Horn. On sentences which are true of direct unions of algebras. *Journal of Symbolic Logic*, 16:14–21, 1951.
11. H. A. Kautz, M. J. Kearns, and B. Selman. Reasoning with characteristic models. In *National Conference on Artificial Intelligence*, pages 34–39, 1993.
12. H. A. Kautz, M. J. Kearns, and B. Selman. Horn approximations of empirical data. *Artificial Intelligence*, 74(1):129–145, 1995.
13. D. Kavvadias, C. H. Papadimitriou, and M. Sideri. On Horn envelopes and hypergraph transversals. In K. W. Ng, P. Raghavan, N. V. Balasubramanian, and F. Y.L. Chin, editors, *Algorithms and Computation, Proceedings of the 4th International Symposium on Algorithms and Computation*, volume 762 of *Lecture Notes in Computer Science*, pages 399–405. Springer, 1993.
14. S. O. Kuznetsov. On computing the size of a lattice and related decision problems. *Order*, 18:313–321, 2001.
15. J. C. C. McKinsey. The decision problem for some classes without quantifiers. *Journal of Symbolic Logic*, 8:61–76, 1943.
16. C. H. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.
17. J. S. Provan and M. O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal of Computing*, 12(4):777–788, 1983.
18. B. Selman and H. Kautz. Knowledge compilation and theory approximation. *Journal of the ACM*, 43(2):193–224, 1996.