TECHNISCHE
UNIVERSITÄT
DRESDEN

Technische Universität Dresden
Institute for Theoretical Computer Science
Chair for Automata Theory

LTCS–Report

Exploration by Confidence

Daniel Borchmann

LTCS-Report 13-04

Postal Address:
Lehrstuhl für Automatentheorie
Institut für Theoretische Informatik
TU Dresden
01062 Dresden

http://lat.inf.tu-dresden.de

Visiting Address:
Nöthnitzer Str. 46
Dresden

Exploration by Confidence

Daniel Borchmann[*]

October 25, 2013

Abstract

Within formal concept analysis, attribute exploration is a powerful tool to semi-automatically check data for completeness with respect to a given domain. However, the classical formulation of attribute exploration does not take into account possible errors which are present in the initial data. We present in this work a generalization of attribute exploration based on the notion of *confidence*, which will allow for the exploration of implications which are not necessarily valid in the initial data, but instead enjoy a minimal confidence therein.

1 Introduction

Attribute exploration is one of the most important algorithms in the area of formal concept analysis [GW99], a branch of mathematical order theory with applications in artificial intelligence, machine learning and data mining. The main purpose of this algorithm is to check a given set of *initial data* for completeness, in the sense that this algorithm assists a domain expert in checking whether this initial data completely represents the particular domain the expert is interested in. In doing so, the algorithm presents *implications* to the expert, who has to either validate them or has to provide a counterexample from the domain of discourse. When the algorithm has finished, the initial data has been extended to a *complete* set of examples whose valid implications completely describe all implications valid in the domain.

However, this approach requires the initial data to be free of *errors* in the sense that all the data really stems from the domain. In practical applications, this may not be reasonable to assume, as it may likewise not be reasonable to check the data for correctness. However, the data itself may still be of “high quality” and therefore could be useful, yet only applying attribute exploration is not possible anymore.

One way to consider of a data set to be of “high quality” is to say that errors occur only “rarely.” To handle a scenario like this, we propose an approach based on the notion of *confidence* from data mining [AIS93]. The idea of this approach is not only to explore the implications which are valid in the initial data set, but also those who satisfy a certain lower bound on their confidence. Of course, this will only provide us with a

[*] Supported by DFG Graduiertenkolleg 1763 (QuantLA)

heuristic algorithm, but in a setting like this, where errors can occur randomly, this is the best we can expect to get. Moreover, an *exploration by confidence* has to be thought of as a first step in a completion process, where the resulting set of implications and set of data should be used further on. As an example, the implications obtained from the exploration by confidence could be used as a background knowledge for a classical attribute exploration which starts out with an empty data set.

The contributions of this work are twofold. Of course, the main result is the development of algorithms for exploration by confidence, which we shall discuss in Section 4. However, these algorithms will be instances of more general formulations of attribute exploration, both with and without optimal interaction strategies. These generalizations are discussed in Section 3 and are derived from the classical case of attribute exploration. Because of their abstract formulation, they might be of use beyond the goals of this work, which is why we consider them as our second main result.

2 Implications and Confidence

In order to make our considerations self-contained we shall introduce the necessary definitions from formal concept analysis in this section. To this end, we shall first provide the basic notions in Section 2.1. Thereafter, we shall also introduce the notion of confidence and confident implications in Section 2.2.

2.1 Basic Notions from Formal Concept Analysis

The main research direction of formal concept analysis we are interested in here is *attribute exploration.* For this, we need to introduce the notions of *formal contexts* and *implications* in formal contexts.

A *formal context* is a triple $\mathbb{K} = (G, M, I)$ consisting of two sets G, M and a relation $I \subseteq G \times M$. The set G is normally thought of as the set of *objects* and the set M as the set of *attributes* of the formal context $\mathbb{K}$. We shall furthermore say that an object $g \in G$ *has* an attribute $m \in M$ (in the formal context $\mathbb{K}$) if and only if $(g, m) \in I$. In this case, we shall also write $g \mathrel{I} m$.

For a set $A \subseteq M$ of attributes we shall denote the set of all objects which share all the attributes in A by A'. More formally,

$$A' = \{\, g \in G \mid \forall m \in A \colon g \mathrel{I} m \,\}.$$

Likewise, for a set $B \subseteq G$ of objects we denote with B' the set of all common attributes of all objects in B, i. e.

$$B' = \{\, m \in M \mid \forall g \in B \colon g \mathrel{I} m \,\}.$$

The sets A' and B' are called the *derivations* of A and B in $\mathbb{K}$, respectively. The corresponding mappings $A \mapsto A'$ and $B \mapsto B'$ are called the *derivation operators* of $\mathbb{K}$. Note that, despite the fact that these two functions share the same name, it is usually clear from the context which one is meant. If not, then we shall add extra syntax to make the distinction clear.

Let $A, B \subseteq M$. In the formal context $\mathbb{K}$ it might be the case that whenever an objects g has all the attributes from A it also has all the attributes from B. In other words, there is a *dependency* between the attribute sets A and B in that A *implies* B. We can formalize this notion by calling a pair (A, B) of sets $A, B \subseteq M$ of attributes an *implication* on M. We shall write $A \to B$ instead of (A, B) to make clear that we consider the pair as an implication. The set of all implications on M is denoted by $\operatorname{Imp}(M)$. We shall say that $A \to B$ *holds* (is *valid*) in $\mathbb{K}$ if and only if every objects that has all the attributes from A also has all the attributes from B, or equivalently $A' \subseteq B'$. We shall write $\mathbb{K} \models (A \to B)$ if and only if $A \to B$ *holds* in $\mathbb{K}$. The subset of $\operatorname{Imp}(M)$ of all implications which are valid in $\mathbb{K}$ is denoted by $\operatorname{Th}(\mathbb{K})$ and is called the *theory* of $\mathbb{K}$.

Let $\mathcal{L} \subseteq \operatorname{Imp}(M)$ be a set of implications, and let $(A \to B) \in \operatorname{Imp}(M)$. The set $\mathcal{L}$ *entails* $(A \to B)$ if and only if in all formal contexts $\mathbb{L}$ with attribute set M, it is true that if all implications from $\mathcal{L}$ are valid in $\mathbb{L}$, then $(A \to B)$ is valid in $\mathbb{L}$ as well. In other words,

$$\mathbb{L} \models \mathcal{L} \implies \mathbb{L} \models (A \to B),$$

if we write $\mathbb{L} \models \mathcal{L}$ to mean that all implications in $\mathcal{L}$ are valid in $\mathbb{L}$. If $\mathcal{L}$ entails $(A \to B)$ we shall also write $\mathcal{L} \models (A \to B)$. The subset of $\operatorname{Imp}(M)$ which is entailed by $\mathcal{L}$ is denoted by $\operatorname{Cn}_M(\mathcal{L})$. We shall drop the subscript if the set M is clear from the context.

Entailment between implications can be characterized in a different manner. For this we introduce the notion of *closure operators* induced by sets of implications. More precisely, we define for $A \subseteq M$

$$\begin{aligned} \mathcal{L}^1(A) &:= A \cup \{ Y \mid (X \to Y) \in \mathcal{L}, X \subseteq A \}, \\ \mathcal{L}^{i+1}(A) &:= \mathcal{L}^i(\mathcal{L}^1(A)) \quad (i \in \mathbb{N}_{>0}), \\ \mathcal{L}(A) &:= \bigcup_{i \in \mathbb{N}_{>0}} \mathcal{L}^i(A). \end{aligned}$$

We shall call the mapping $A \mapsto \mathcal{L}(A)$ the *closure operator* induced by $\mathcal{L}$, and we shall call the set A to be *$\mathcal{L}$-closed* if and only if $A = \mathcal{L}(A)$. The closure operator induced by $\mathcal{L}$ can now be used to characterize entailment of implications as follows:

$$\mathcal{L} \models (A \to B) \iff B \subseteq \mathcal{L}(A).$$

Let $\mathcal{K} \subseteq \operatorname{Imp}(M)$ be another set of implications. We shall call $\mathcal{L}$ a *base* of $\mathcal{K}$ if and only if $\operatorname{Cn}(\mathcal{L}) = \operatorname{Cn}(\mathcal{K})$. In other words, all implications in $\mathcal{K}$ are entailed by $\mathcal{L}$ and vice versa. If $\mathcal{K} = \operatorname{Th}(\mathbb{K})$, then we shall call $\mathcal{L}$ a *base of* $\mathbb{K}$. Note that a base of $\mathcal{K}$ is always a base of $\operatorname{Cn}(\mathcal{K})$, and vice versa.

Bases allow us to represent sets $\mathcal{K}$ of implications in a different way. This fact is mostly exploited by searching for bases of $\mathcal{K}$ which are of considerable smaller size than $\mathcal{K}$ itself. Those bases are preferably *non-redundant* or even *minimal.* More precisely, if $\mathcal{L}$ is a base of $\mathcal{K}$, then $\mathcal{L}$ is called *non-redundant* if no proper subset of $\mathcal{L}$ is a base of $\mathcal{K}$ as well. $\mathcal{L}$ is called *minimal* if and only if there does not exist another base $\mathcal{L}'$ of $\mathcal{K}$ satisfying $|\mathcal{L}'| < |\mathcal{L}|$.

If we search for bases of $\mathcal{K}$, it might be the case that we do not want to include a certain set $\mathcal{L}_{\text{back}}$ of implications which we already "know." We can think of these implications as given *a-priori*, or as *background knowledge*. If we have given such background knowledge, to find a base of $\mathcal{K}$ it then suffices to find a base of all those implications in $\mathcal{K} \setminus \operatorname{Cn}(\mathcal{L}_{\text{back}})$. Therefore, we shall call a set $\mathcal{L} \subseteq \operatorname{Imp}(M)$ a *base of $\mathcal{K}$ relative to $\mathcal{L}_{back}$* (or a *base of $\mathcal{K}$ with background knowledge $\mathcal{L}_{back}$*) if and only if $\mathcal{L} \cup \mathcal{L}_{\text{back}}$ is a base of $\mathcal{K}$. The notions of non-redundancy and minimality for relative bases are the same as in the case of bases. Note that if the background knowledge is empty, then relative bases are just bases.

A particular relative base from which it is known to have minimal cardinality is the *canonical base* $\operatorname{Can}(\mathcal{K}, \mathcal{L}_{\text{back}})$. To define this base, we need to introduce the notion of *$\mathcal{L}_{back}$-pseudo-closed sets of $\mathcal{K}$*. Let $P \subseteq M$. Then P is called an $\mathcal{L}_{\text{back}}$-pseudo-closed set of $\mathcal{K}$ if and only if the following conditions hold.

i. $P = \mathcal{L}_{\text{back}}(P)$;

ii. $P \neq \mathcal{K}(P)$;

iii. for all $Q \subsetneq P$, which are $\mathcal{L}_{\text{back}}$-pseudo-closed sets of $\mathcal{K}$ it is true that $\mathcal{K}(Q) \subseteq P$.

Then

$$\operatorname{Can}(\mathcal{K}, \mathcal{L}_{\text{back}}) := \{\, P \to \mathcal{K}(P) \mid P \subseteq M \text{ an } \mathcal{L}_{\text{back}}\text{-pseudo-closed set of } \mathcal{K}\,\}.$$

It is well-known that $\operatorname{Can}(\mathcal{K}, \mathcal{L}_{\text{back}})$ is a base of $\mathcal{K}$ with background-knowledge $\mathcal{L}_{\text{back}}$ of minimal cardinality; see [GW99; Dis11] for a proof on this.[1]

Computing the canonical base can be done using the NextClosure algorithm [Gan10; GW99]. This algorithm makes use of particular order relations on the subsets of M, which are called *lectic orders*. Since attribute exploration also uses lectic orders, we shall briefly recall the main definitions.

Let $<$ be a strict linear order on M, i.e. $<$ is an irreflexive and transitive relation on M. Let $A, B \subseteq M$, and let $i \in M$. We shall say that A is *lectically smaller than B at position i* if and only if

$$\min\nolimits_{<}(A \mathbin{\triangle} B) = i \quad \text{and} \quad i \in B$$

where $A \mathbin{\triangle} B := (A \setminus B) \cup (B \setminus A)$ is the symmetric difference of A and B. In other words, A is lectically smaller than B at position i if and only the $<$-smallest element in which A and B differ is equal to i, and $i \in B$. We shall write $A \prec_i B$ in this case. We say that A is *lectically smaller* than B, written $A \prec B$, if and only if $A \prec_i B$ for some $i \in M$. Finally, we write $A \preceq B$ if and only if $A \prec B$ or $A = B$. The relation $\preceq$ is called the *lectic order* on $\mathfrak{P}(M)$ induced by $<$.

It is easy to see that $\preceq$ is a linear order relation on $\mathfrak{P}(M)$, i.e. it is reflexive, antisymmetric, transitive, and for each two $A, B \in \mathfrak{P}(M)$ it is true that $A \preceq B$ or $B \preceq A$. Furthermore, it is easy to see that $\preceq$ extends the usual subset order on $\mathfrak{P}(M)$, i.e. if $A \subseteq B$, then $A \preceq B$.

[1] This proof is only for the special case $\mathcal{K} = \operatorname{Th}(\mathbb{K})$, which however is equivalent to our general case.

2.2 Confidence and Confident Implications

Implications may not be valid in a formal context. However, it might be the case that such implications are “valid most of the time,” i. e. the number of *counterexamples* for implications can be “small” compared to the number of examples where it is valid. To formalize these rather subjective conception, we shall introduce the notion of *confidence* for implications. It is a straight-forward adaption of the same notion from data mining [AIS93].

Let $\mathbb{K} = (G, M, I)$ be a formal context, and let $(A \to B) \in \operatorname{Imp}(M)$. A *counterexample* (*negative example*) for $(A \to B)$ in $\mathbb{K}$ is an object $g \in A' \setminus B'$. It is obvious that $A \to B$ is valid in $\mathbb{K}$ if and only if $\mathbb{K}$ does not contain counterexample for $A \to B$. Conversely, we call g a *model* of $A \to B$ if and only if $g \notin A'$ or $g \in B'$.

The *confidence* of $(A \to B)$ in $\mathbb{K}$ is defined as

$$\operatorname{conf}_{\mathbb{K}}(A \to B) := \begin{cases} 1 & \text{if } A' = \varnothing \\ \frac{|(A \cup B)'|}{|A'|} & \text{otherwise .} \end{cases}$$

In other words, $\operatorname{conf}_{\mathbb{K}}(A \to B)$ is the conditional probability that a randomly chosen objects $g \in G$ (in a uniform way), which has all the attributes from A also has all the attributes from B. It is clear that $A \to B$ holds in $\mathbb{K}$ if and only if its confidence in $\mathbb{K}$ is 1.

Let $c \in [0, 1]$. We shall denote the set of all implications in $\operatorname{Imp}(M)$ whose confidence is at least c by $\operatorname{Imp}_c(\mathbb{K})$. If c is chosen properly, we may think of $\operatorname{Imp}_c(\mathbb{K})$ as the set of implications which are “almost valid” in $\mathbb{K}$. Finding a base $\mathcal{L}$ for this set might therefore be desirable, as introduced in the previous section. However, the set $\operatorname{Imp}_c(\mathbb{K})$ is not closed under entailment, and thus $\mathcal{L} \subseteq \operatorname{Imp}_c(\mathbb{K})$ may not necessarily be true. However, a base of $\operatorname{Imp}_c(\mathbb{K})$ might be of more use if the element of the base are also “almost valid,” i. e. have a confidence in $\mathbb{K}$ which is at least c. We shall therefore call $\mathcal{L}$ a *confident base* of $\operatorname{Imp}_c(\mathbb{K})$ (or just $\mathbb{K}$, if c is clear from the context) if and only if $\mathcal{L}$ is a base of $\operatorname{Imp}_c(\mathbb{K})$ and $\mathcal{L} \subseteq \operatorname{Imp}_c(\mathbb{K})$.

3 Attribute Exploration

It is the purpose of this section to introduce attribute exploration as it is needed in the exposition of this paper. This shall include a description of the classical attribute exploration algorithm, which we shall give in the following Section 3.1. Thereafter, we shall discuss a generalized form of attribute exploration in Section 3.2, which uses similar ideas but is different from the one given in [Bor13]. Finally, we shall discuss in Section 3.3 a weaker generalization of attribute exploration, which lacks the optimality of the classical case, but provides more freedom in the way questions are generated.

3.1 Classical Attribute Exploration

As already mentioned, attribute exploration is an algorithm which assists experts in completing implicational knowledge about a certain domain of interest. More specifically, let us suppose that we have fixed a set M of *attributes* which are relevant for our considerations. We then can understand the *domain of interest* as a collection $\mathcal{D}$ of objects where each possesses some attributes from M. In other words, a domain $\mathcal{D}$ on a set M is nothing else than a formal context. Let us furthermore suppose that we are given a set $\mathcal{K}$ of implications from which we definitively know that they are valid in our domain $\mathcal{D}$. Finally, we assume that we have an initial collection of some *examples* from our domain, given again as a formal context.

We are now interested to find all implications which hold in our domain, i. e. to find all implications which are not invalidated by objects from the domain $\mathcal{D}$. The difficulty of this problems stems from the fact that enumerating all these objects may be algorithmically infeasible. What we can assume, however, is that we have given an *expert* which is able to provide us with the information whether there *exists*, for a given implication $(A \to B) \in \operatorname{Imp}(M)$, an object in our domain $\mathcal{D}$ which is a counterexample for (i. e. not a model of) $A \to B$.

Abstractly, attribute exploration now proceeds as follows. From all implications in $\operatorname{Cn}(\mathcal{K})$, we already known that they are valid in our domain $\mathcal{D}$. Furthermore, for all implications which are invalidated by objects from $\mathbb{K}$, we known that they are not valid in $\mathcal{D}$. For all other implications, we do not know whether they hold in $\mathcal{D}$ or not, i. e. all implications in

$$U(\mathbb{K}, \mathcal{K}) := \operatorname{Th}(\mathbb{K}) \setminus \operatorname{Cn}(\mathcal{K})$$

are *undecided* in the sense that they could be valid in $\mathcal{D}$ or not. For those implications in $U(\mathbb{K}, \mathcal{K})$, we have to consult the expert. Attribute exploration now does this in a systematic and somehow efficient way, provided that M is finite.

To make this more precise, we shall proceed by describing attribute exploration in a formal way. This description shall be much more formal as usual, to provide the necessary notions we need for our generalized attribute exploration. First, we shall first provide some necessary definitions. After that, we give a formal description of the algorithm. Finally, we shall note some well-known properties of attribute exploration.

We shall start by formalizing our initial, subjective notion of a *domain expert.* Intuitively, a domain expert for a domain $\mathcal{D}$ is just a "function" p that, given an implication $A \to B$, returns "true" if $A \to B$ is not invalidated in $\mathcal{D}$, or returns an object from $\mathcal{D}$ which is a counterexample for $A \to B$. We shall take this understanding as the motivation for the following definition. See also [Bor13].

3.1 Definition Let M be a set. A *domain expert* on M is a function

$$p \colon \operatorname{Imp}(M) \to \{ \top \} \cup \mathfrak{P}(M),$$

where $\top \notin \mathfrak{P}(M)$, such that the following conditions hold:

i. If $(X \to Y) \in \operatorname{Imp}(M)$ such that $p(X \to Y) = C \neq \top$, then $C \not\models (X \to Y)$, i. e. $X \subseteq C, Y \nsubseteq C$. (*$p$ gives counterexamples for false implications*)

ii. If $(A \to B), (X \to Y) \in \operatorname{Imp}(M)$ such that $p(A \to B) = \top, p(X \to Y) = C \neq \top$, then $C \models (A \to B)$. (*counterexamples do not invalidate correct implications*)

We say that p *confirms* an implication $A \to B$ if and only if $p(A \to B) = \top$. Otherwise, we say that p *rejects* $A \to B$ with counterexample $p(A \to B)$. The *theory* $\operatorname{Th}(p)$ of p is the set of all implications on M confirmed by p. ♢

3.2 Lemma *Let $\mathcal{D}$ be domain on a set M. For each $(A \to B) \in \operatorname{Imp}(M)$ for which there exists a counterexample in $\mathcal{D}$, let $C_{A\to B}$ such a counterexample. Then the mapping*

$$p_{\mathcal{D}}(X \to Y) := \begin{cases} C_{X\to Y} & \text{if } C_{X\to Y} \text{ exists} \\ \top & \text{otherwise} \end{cases}$$

is a domain expert on M.

Note that the definition of $p_{\mathcal{D}}$ depends on the particular choice of the counterexamples, therefore $\mathcal{D}$ may give rise to more then one domain expert.

Proof For the first claim we observe that $p_{\mathcal{D}}$ gives counterexamples to false implication by definition. Furthermore, all counterexamples returned by $p_{\mathcal{D}}$ are elements of $\mathcal{D}$, and $p_{\mathcal{D}}(X \to Y) = \top$ if and only if there does not exist a counterexample for $X \to Y$ in $\mathcal{D}$. This shows that counterexamples provided by $p_{\mathcal{D}}$ do not invalidate correct implications.□

Let

$$\mathcal{D}_p := \{\, p(A \to B) \mid (A \to B) \in \operatorname{Imp}(M) \,\} \setminus \{\, \top \,\}.$$

Then clearly $\mathcal{D}_p$ is a domain, and it is easy to see that each domain expert p on M can be obtained as a domain expert of the form $p_{\mathcal{D}_p}$, and that for each domain $\mathcal{D}$ on M it is true that $\mathcal{D} = \mathcal{D}_{p_{\mathcal{D}}}$.

The crucial observation is now that domain experts can answer the question of *validity* in the domains they represent.

3.3 Lemma *Let M be a set and let p be a domain expert on M. The for each $(A \to B) \in \operatorname{Imp}(M)$ it is true that*

$$(A \to B) \text{ is valid in } \mathcal{D}_p \iff p(A \to B) = \top.$$

Proof Let $A \to B$ be valid in $\mathcal{D}_p$. Then $p(A \to B)$ cannot be a subset of M, since this would be an element of $\mathcal{D}_p$ falsifying $A \to B$. Therefore, $p(A \to B) = \top$ and hence $(A \to B) \in \operatorname{Th}(p)$.

Conversely, let $p(A \to B) = \top$. Since counterexamples from p do not invalidate correct implications, for each implication $(X \to Y) \in \operatorname{Imp}(M)$ where $p(X \to Y) \subseteq M$, the set $p(X \to Y)$ is not a counterexample for $A \to B$. Hence, $\mathcal{D}_p$ does not contain a counterexample for $A \to B$, i. e. $A \to B$ is valid in $\mathcal{D}_p$. □

We have formally captured the notion of an expert, and we are now ready to describe the algorithm of attribute exploration. In this exposition, we assume that the set M

is equipped with a strict linear order, which then gives rise to a lectic order as it has been described in Section 2. For better readability, we denote a formal context that arises from another formal context $\mathbb{K}$ by adding a new object with attributes from C by $\mathbb{K} + C$.

Algorithm 1

Input A domain expert p on a finite set M, a set $\mathcal{K} \subseteq \operatorname{Imp}(M)$ and a formal context $\mathbb{K}$ with attribute set M such that $\mathcal{K} \subseteq \operatorname{Th}(p) \subseteq \operatorname{Th}(\mathbb{K})$.

Procedure

i. Initialize $i := 0, P_i := \mathcal{K}(\varnothing), \mathcal{K}_i := \mathcal{K}, \mathbb{K}_i := \mathbb{K}$.

ii. Let P_{i+1} be the smallest $\mathcal{K}_i$-closed set lectically larger or equal to P_i, which is not an intent of $\mathbb{K}_i$. If no such set exists, terminate.

iii. If p confirms $P \to P''$, then
   - $\mathcal{K}_{i+1} := \mathcal{K}_i \cup \{ P \to P'' \}$,
   - $\mathbb{K}_{i+1} := \mathbb{K}_i$.

iv. If p provides a counterexample C for $P \to P''$, then
   - $\mathcal{K}_{i+1} := \mathcal{K}_i$,
   - $\mathbb{K}_{i+1} := \mathbb{K}_i + C$.

v. Set $i := i + 1$ and go to ii.

Output Return $\mathcal{K}_i$ and $\mathbb{K}_i$. □

Note that the computation of the set P_{i+1} from P_i, $\mathcal{K}_i$ and $\mathbb{K}_i$ and $\mathcal{L}_i$ can be done by using the well-known Next-Closure algorithm [Gan10; GW99]. As the details are not relevant for our further discussion, we refer the interested reader to the given resources.

The following results are well known properties of Algorithm 1.

3.4 Theorem *Let p, $\mathcal{K}$ and $\mathbb{K}$ be valid input for Algorithm 1. Then Algorithm 1 terminates with input p, $\mathcal{K}$ and $\mathbb{K}$. If $\mathcal{K}'$ and $\mathbb{K}'$ are the corresponding values returned by the algorithm, then the following statements are true:*

i. $\mathcal{K} \subseteq \mathcal{K}' \subseteq \operatorname{Th}(\mathbb{K}') \subseteq \operatorname{Th}(\mathbb{K})$.

ii. $\operatorname{Th}(p) = \operatorname{Th}(\mathbb{K}') = \operatorname{Cn}(\mathcal{K}')$.

iii. *The cardinality of $\mathcal{K}' \setminus \mathcal{K}$ is the smallest possible with respect to* $\operatorname{Th}(p) = \operatorname{Cn}(\mathcal{K}')$. *More specifically,* $\mathcal{K}' \setminus \mathcal{K} = \operatorname{Can}(\mathbb{K}', \mathcal{K})$.

The claims of this theorem are shown in [GW99; Gan99; Stu96; Dis11].

3.2 Generalized Form of Attribute Exploration

We have given a precise formulation of attribute exploration in the previous section. However, this formulation is not applicable to our setting of exploring implications with a certain minimal confidence. To address this issue, we shall develop in this section a more general formulation of attribute exploration which goes beyond the classical one.

To make this a reasonable undergoing, we shall first clarify which problem we want to solve with attribute exploration. In the classical case, we have given a formal context $\mathbb{K}$ and a set of implications $\mathcal{K} \subseteq \operatorname{Th}(\mathbb{K})$. Furthermore, we have given a domain expert p, who confirms all implications in $\mathcal{K}$ and where all implications confirmed by p are contained in $\operatorname{Th}(\mathbb{K})$. The task attribute exploration then solves is to provide a method to guide the expert p through all implications in $\operatorname{Th}(\mathbb{K}) \setminus \operatorname{Cn}(\mathcal{K})$ for deciding whether these implications are valid in the domain or not. At the end, attribute exploration both provides a set of implications which is a relative base of all valid implications of the domain p represents, and a set of objects from the domain such that an implication is valid in the domain if and only if all these objects are models of this implication. This set of objects forms itself a domain, and it can be thought of as a sufficient excerpt of the domain represented by p.

We want to try to lift this description of attribute exploration to the case of exploration by confidence. There, our setting is a bit more involved. As in the case of classical attribute exploration, we have given a domain expert p, a formal context $\mathbb{K}$ and a set of implications $\mathcal{K}$. Additionally, we have given a $c \in [0, 1]$, the *confidence threshold* for our exploration. Then, in contrast to the classical setting, exploration by confidence considers not only the implications $\operatorname{Th}(\mathbb{K})$, but also those in $\operatorname{Imp}_c(\mathbb{K})$. Therefore, we can assume that $\mathcal{K}$ is a set of implications with confidence at least c, that all implications in $\mathcal{K}$ are confirmed by p and that all implications which are in $\mathcal{K}$ are also contained in $\operatorname{Imp}_c(\mathbb{K})$. In other words, $\mathcal{K} \subseteq \operatorname{Th}(p)$ and $\mathcal{K} \subseteq \operatorname{Imp}_c(\mathbb{K})$.

An attribute exploration algorithm which then works in this setting should guide the expert through the implications in $\operatorname{Imp}_c(\mathbb{K}) \setminus \operatorname{Cn}(\mathcal{K})$, asking whether some implications are correct or not. The counterexamples which are provided by the expert are then used to falsify certain implications in $\operatorname{Imp}_c(\mathbb{K})$. They are not used, however, for computing the confidence; this is solely done in the initial context $\mathbb{K}$. At the end, the attribute exploration algorithm should both compute a set $\mathcal{L}$ of implications and a formal context $\mathbb{L}$ such that each implication in $\operatorname{Imp}_c(\mathbb{K})$ is either not valid in $\mathbb{L}$ or follows from $\mathcal{L} \cup \mathcal{K}$.

One might be tempted to think that this problem actually has an easy (theoretical) solution. Namely, one can observe that the set $\operatorname{Imp}_c(\mathbb{K})$, as any set of implications, induces a closure operator. Then it is possible to represent this closure operator by means of a formal context $\mathbb{L}_{\operatorname{Imp}_c(\mathbb{K})}$ [GW99; Bor11]. Using this formal context, one could then just do classical attribute exploration, since the set $\mathcal{K}$ of initially given implications is now valid in $\mathbb{L}_{\operatorname{Imp}_c(\mathbb{K})}$.

However, this approach solves a different problem than the one we have described above. The problem is that the set $\operatorname{Imp}_c(\mathbb{K})$ cannot be represented by a formal context, since it is not closed under entailment. In other words, it is in general true that

$$\operatorname{Cn}(\operatorname{Imp}_c(\mathbb{K})) \supsetneq \operatorname{Imp}_c(\mathbb{K}).$$

However, if we would consider the approach sketched in the last paragraph, we would actually consider all implications in $\operatorname{Cn}(\operatorname{Imp}_c(\mathbb{K}))$ instead of only those in $\operatorname{Imp}_c(\mathbb{K})$. This is also the reason why we cannot use the generalized attribute exploration algorithm as described in [Bor13].

That this could be a problem can be seen as follows. It may be the case that for two implications $(A \to B), (B \to C) \in \operatorname{Imp}_c(\mathbb{K}), A \subseteq B \subseteq C$, it is true that $(A \to C) \notin \operatorname{Imp}_c(\mathbb{K})$ (because $\operatorname{conf}_{\mathbb{K}}(A \to C) = \operatorname{conf}_{\mathbb{K}}(A \to B) \cdot \operatorname{conf}_{\mathbb{K}}(B \to C)$.) However, $(A \to C) \in \operatorname{Cn}(\operatorname{Imp}_c(\mathbb{K}))$. Now, assume that the expert may provide a counterexample for the implication $A \to B$. Then in the case of only considering the set $\operatorname{Imp}_c(\mathbb{K})$, the implication $A \to C$ could immediately be falsified as well, since $A \to B$ and $B \to C$ might have been the only reason that $(A \to C) \in \operatorname{Cn}(\operatorname{Imp}_c(\mathbb{K}))$. However, if we consider the whole set $\operatorname{Cn}(\operatorname{Imp}_c(\mathbb{K}))$, then the implication $A \to C$ is still present, and it might very well happen that the counterexample provided by the expert does not invalidate this implication. In the worst case, the implication $A \to C$ has to be considered separately by the expert, which is not what we want, since $\operatorname{conf}_{\mathbb{K}}(A \to C) \leqslant c$.

Therefore, what we really need is a modified (i. e. more general) formulation of attribute exploration that is applicable to the above setting of exploration by confidence. For this, we shall develop in the remainder of this section a general formulation of attribute exploration that works with a set of *certain implications* and a set of *interesting implications* and provides a method to guide an expert through the set of *undecided implications*, until no more are left. The properties this algorithm should have should be the same as in the classical case, as far as this is possible. Then later on, we shall apply this algorithm to our setting of exploration of confidence.

To this end, let us recapitulate our setting for the exploration algorithm, this time a bit more general: we have given a finite set M, a domain expert p on M, and two sets $\mathcal{K}, \mathcal{L}$ of implications. In our classical case, $\mathcal{L} = \operatorname{Th}(\mathbb{K})$ for some formal context $\mathbb{K}$; in our setting of exploration by confidence, we would have $\mathcal{L} = \operatorname{Imp}_c(\mathbb{K})$, again for some formal context $\mathbb{K}$. We assume that $\mathcal{K} \subseteq \operatorname{Th}(p)$ and $\mathcal{K} \subseteq \mathcal{L}$. We then consider the set $\mathcal{K}$ as the set of implications which we *definitively know* to be confirmed by p. Let us therefore call this set the (initial) set of *certain implications*. Furthermore, for our exploration we only consider implications in $\mathcal{L}$, therefore we shall call this set the set of *interesting implications*. Finally, for each implication in $\mathcal{L} \setminus \operatorname{Cn}(\mathcal{K})$ it is not clear yet whether p confirms it or not. Therefore, we call this set the (current) set of *undecided implications*.

An exploration for this abstract setting now should compute a relative base of $\mathcal{L} \cap \operatorname{Th}(p)$ with background knowledge $\mathcal{K}$ by interacting with the expert p. At best, this interaction is kept at a minimum (i. e. the number of times the expert is invoked is as small as possible), as expert interaction can normally be assumed to be expensive.

Considering the classical attribute exploration, it is not very difficult to come up with a reformulation which is reasonably applicable to this general setting. For the remainder of this section, let us fix a finite set M and a lectic order $\preceq$ on $\mathfrak{P}(M)$.

Algorithm 2 (General Attribute Exploration)

Input A domain expert p on a finite set M and sets $\mathcal{K}, \mathcal{L} \subseteq \operatorname{Imp}(M)$ such that $\mathcal{K} \subseteq \operatorname{Th}(p)$ and $\mathcal{K} \subseteq \mathcal{L}$.

Procedure

i. Initialize $i := 0, P_i := \mathcal{K}(\varnothing), \mathcal{K}_i := \mathcal{K}, \mathcal{L}_i := \mathcal{L}, \mathbb{L}_i := (\varnothing, M, \varnothing)$.

ii. Let P_{i+1} be the smallest $\mathcal{K}_i$-closed set lectically larger or equal to P_i, which is not $\mathcal{L}_i$-closed. If no such set exists, terminate.

iii. If p confirms $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, then
  - $\mathcal{K}_{i+1} := \mathcal{K}_i \cup \{ P_{i+1} \to \mathcal{L}_i(P_{i+1}) \}$,
  - $\mathcal{L}_{i+1} := \mathcal{L}_i$,
  - $\mathbb{L}_{i+1} := \mathbb{L}_i$.

iv. If p provides a counterexample C for $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, then
  - $\mathcal{K}_{i+1} := \mathcal{K}_i$,
  - $\mathcal{L}_{i+1} := \{ (A \to B) \in \mathcal{L}_i \mid C \models (A \to B) \}$,
  - $\mathbb{L}_{i+1} := \mathbb{L}_i + C$.

v. Set $i := i + 1$ and go to ii.

Output Return $\mathcal{K}_i$ and $\mathbb{L}_i$. □

The properties we now require this algorithm to have are *correctness* and *optimality*. In other words, we want the algorithm to terminate for every legal input $M, p, \mathcal{K}, \mathcal{L}$, and if $\hat{\mathcal{K}}$ and $\mathbb{L}$ are the computed results, we want that

$$\operatorname{Cn}(\hat{\mathcal{K}}) = \operatorname{Cn}(\operatorname{Th}(p) \cap \mathcal{L}),$$

and for each implication in $\mathcal{L}$, we demand that either p confirms it or it is invalidated by a counterexample in $\mathbb{L}$. Finally, we want that the number of implications in $\hat{\mathcal{K}} \backslash \mathcal{K}$ to be *as small as possible* under the assumption of correctness, i. e. the number of times the expert confirms an implication is as small possible while the algorithm still returns the correct result.

Before we are going to prove these claims, we start by some simple observations. First of all, it is easy to see that $\mathcal{K}_i \subseteq \operatorname{Th}(\mathbb{L}_i)$ is true for every iteration i, since all implications in $\mathcal{K}_i$ have been confirmed by p, and $\mathbb{L}_i$ only contains counterexamples from p and p does not provide counterexamples which invalidate confirmed implications. Secondly, we also see easily that $\mathcal{L}_i \supseteq \operatorname{Th}(p) \cap \mathcal{L}$, i. e. implications which would be confirmed by p are never removed from $\mathcal{L}_i$. Again, the argument is that the counterexamples from p do not invalidate any implication in $\operatorname{Th}(p) \cap \mathcal{L}$.

We first consider termination of the algorithm, and show as a byproduct that $\mathcal{K}_i \subseteq \operatorname{Cn}(\mathcal{L}_i)$ is true for all possible i.

3.5 Proposition *Let $p, \mathcal{K}, \mathcal{L}$ be valid input for Algorithm 2. Then for iteration i of the algorithm for this input, $\mathcal{K}_i \subseteq \operatorname{Cn}(\mathcal{L}_i)$. Furthermore, $\mathcal{K}_i \subseteq \mathcal{K}_{i+1}$ and $\mathcal{L}_i \supseteq \mathcal{L}_{i+1}$, and exactly one of these inclusions is proper.*

Proof Suppose that we are in iteration i. It is immediate from the formulation of the algorithm that $\mathcal{K}_i \subseteq \mathcal{K}_{i+1}$ and $\mathcal{L}_i \supseteq \mathcal{L}_{i+1}$. If p confirms $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, then $\mathcal{K}_i \subsetneq \mathcal{K}_{i+1}$. If p rejects $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, then p provides a counterexample C, i. e.

$C \not\models (P_{i+1} \to \mathcal{L}_i(P_{i+1}))$. Then there exists at least one implication in $\mathcal{L}_i$ for which C is not a model, i. e. $\mathcal{L}_i \supsetneq \mathcal{L}_{i+1}$.

We show the claim $\mathcal{K}_i \subsetneq \operatorname{Cn}(\mathcal{L}_i)$ by induction. For $i = 0$, it is true that $\mathcal{K}_i = \mathcal{K}$ and $\mathcal{L}_i = \mathcal{L}$ and thus $\mathcal{K} \subseteq \mathcal{L} \subseteq \operatorname{Cn}(\mathcal{L})$.

Now assume $\mathcal{K}_i \subseteq \operatorname{Cn}(\mathcal{L}_i)$, and we consider the set P_{i+1}. Then $P_{i+1} = \mathcal{K}_i(P_{i+1})$ and $P_{i+1} \neq \mathcal{L}_i(P_{i+1})$.

If p confirms $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, then $\mathcal{L}_i = \mathcal{L}_{i+1}$. Since $\mathcal{K}_i \subseteq \operatorname{Cn}(\mathcal{L}_i)$ and $(P_{i+1} \to \mathcal{L}_i(P_{i+1})) \in \operatorname{Cn}(\mathcal{L}_i)$, it is true that $\mathcal{K}_{i+1} \subseteq \operatorname{Cn}(\mathcal{L}_{i+1})$.

If p rejects $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, then let C be the counterexample given by p. Then

$$\mathcal{L}_{i+1} = \operatorname{Th}(\mathbb{L}_{i+1}) \cap \mathcal{L}_i,$$

where $\mathbb{L}_{i+1}$ arises from $\mathbb{L}_i$ by adding C as a counterexample. As already noted, $\mathcal{K}_{i+1} \subseteq \operatorname{Th}(\mathbb{L}_{i+1})$. Therefore,

$$\begin{aligned}
\operatorname{Cn}(\mathcal{L}_{i+1}) &= \operatorname{Cn}(\operatorname{Th}(\mathbb{L}_{i+1}) \cap \mathcal{L}_i) \\
&= \operatorname{Cn}(\operatorname{Th}(\mathbb{L}_{i+1}) \cap \operatorname{Cn}(\mathcal{L}_i)) \\
&\supseteq \operatorname{Cn}(\mathcal{K}_{i+1} \cap \mathcal{K}_i) \\
&= \mathcal{K}_{i+1},
\end{aligned}$$

since $\operatorname{Cn}(\mathcal{L}_i) \supseteq \mathcal{K}_i$ by induction hypothesis, and $\mathcal{K}_i = \mathcal{K}_{i+1}$. □

From Proposition 3.5, we immediately obtain the termination of Algorithm 2 on valid input.

3.6 Theorem *Let $p, \mathcal{K}, \mathcal{L}$ be valid input for Algorithm 2. Then the algorithm with this input terminates after finitely many steps.*

Proof Note that the algorithm stops if $\operatorname{Cn}(\mathcal{K}_i) = \operatorname{Cn}(\mathcal{L}_i)$. From Proposition 3.5 we know that $\mathcal{K}_i \subsetneq \mathcal{K}_{i+1}$ or $\mathcal{L}_i \supsetneq \mathcal{L}_{i+1}$, and that $\mathcal{K}_i \subseteq \operatorname{Cn}(\mathcal{L}_i)$. Since all sets are finite, the condition $\operatorname{Cn}(\mathcal{K}_i) = \operatorname{Cn}(\mathcal{L}_i)$ will eventually be reached and the algorithm stops. □

Now that we established termination of our algorithm, we consider its correctness. The following result is crucial for our following argumentation.

3.7 Proposition *In every iteration i of Algorithm 2, it is true that for all $A < P_{i+1}$, if A is $\mathcal{K}_i$-closed, then A is also $\mathcal{L}_i$-closed.*

Proof We show the claim by induction on i. For $i = 0$, the claim is vacuously true, as P_o is the lectically smallest $\mathcal{K}_i$-closed set. Now assume the validity of the proposition for i. We then need to show that for each $A < P_{i+2}$, if A is $\mathcal{K}_{i+1}$-closed, then A is also $\mathcal{L}_{i+1}$-closed.

We first consider the case $A < P_{i+1}$, and we assume that A is $\mathcal{K}_{i+1}$-closed. Since $\mathcal{K}_i \subseteq \mathcal{K}_{i+1}$, A is also $\mathcal{K}_i$-closed. By induction hypothesis, A is $\mathcal{L}_i$-closed. Since $\mathcal{L}_{i+1} \subseteq \mathcal{L}_i$, A is also $\mathcal{L}_{i+1}$-closed, completing the proof for this case.

If $P_{i+1} = P_{i+2}$, then nothing remains to be shown. We therefore assume $P_{i+1} < P_{i+2}$. Then the remaining case $P_{i+1} \leq A < P_{i+2}$ can be reduced to $P_{i+1} = A$, since all sets lectically between P_{i+1} and P_{i+2}, which are $\mathcal{K}_{i+1}$-closed are also $\mathcal{L}_{i+1}$-closed, by construction of P_{i+2}.

Hence we consider the case $A = P_{i+1}$, and we assume that A is $\mathcal{K}_{i+1}$-closed. Since $P_{i+1} \neq P_{i+2}$, we know that then P_{i+1} must also be $\mathcal{L}_{i+1}$-closed, as otherwise the algorithm would choose $P_{i+2} = P_{i+1}$. This finishes the induction step and the proof of the proposition. □

3.8 Corollary *Let $p, \mathcal{K}, \mathcal{L}$ be valid input for Algorithm 2, and let n be the last iteration of the algorithm. Then*

$$\operatorname{Cn}(\mathcal{K}_n) = \operatorname{Cn}(\mathcal{L}_n).$$

Proof As we have already noted, $\mathcal{K}_n \subseteq \operatorname{Cn}(\mathcal{L}_n)$ is true. Furthermore, by the previous proposition we know that when the algorithm finishes, all $\mathcal{K}_n$-closed subsets of M are also $\mathcal{L}_n$-closed. Now if $(X \to Y) \in \operatorname{Cn}(\mathcal{L}_n)$, then $Y \subseteq \mathcal{L}_n(X) \subseteq \mathcal{L}_n(\mathcal{K}_n(X))$. Now $\mathcal{K}_n(X)$ is $\mathcal{K}_n$-closed, therefore $\mathcal{L}_n(\mathcal{K}_n(X)) = \mathcal{K}_n(X)$. Thus, $Y \subseteq \mathcal{K}_n(X)$ and $(X \to Y) \in \operatorname{Cn}(\mathcal{K}_n)$, showing the claimed equality. □

3.9 Corollary *Let $p, \mathcal{K}, \mathcal{L}$ be valid input for Algorithm 2, and let n be the last iteration of the algorithm. Then*

$$\mathcal{L}_n = \operatorname{Th}(p) \cap \mathcal{L}.$$

Proof We already know that $\mathcal{L}_n \supseteq \operatorname{Th}(p) \cap \mathcal{L}$. Since $\operatorname{Cn}(\mathcal{L}_n) = \operatorname{Cn}(\mathcal{K}_n)$, all implications in $\operatorname{Cn}(\mathcal{L}_n)$ are confirmed by p. In particular, $\mathcal{L}_n \subseteq \operatorname{Th}(p)$, and together with $\mathcal{L}_n \subseteq \mathcal{L}$ we obtain the desired equality. □

From the two corollaries just given, the main property of our generalized attribute exploration algorithm follows immediately.

3.10 Theorem *Let $p, \mathcal{K}, \mathcal{L}$ be valid input for Algorithm 2, and let $\mathcal{K}_n$ and $\mathbb{L}_n$ be the values returned by the algorithm. Then*

$$\operatorname{Cn}(\mathcal{K}_n) = \operatorname{Cn}(\operatorname{Th}(p) \cap \mathcal{L}).$$

In our argumentation given so far, the formal contexts $\mathbb{L}_i$ do not appear. Indeed, they are not necessary for the correctness of the algorithm. However, the counterexamples accumulated in these contexts allow us to formulate the following result, which is useful on its own.

3.11 Theorem *Let $p, \mathcal{K}, \mathcal{L}$ be valid input for Algorithm 2, and let $\mathcal{K}_n$ and $\mathbb{L}_n$ be the values returned by the algorithm. Then for each $(A \to B) \in \mathcal{L}$, either $(A \to B) \in \operatorname{Cn}(\mathcal{K}_n)$ or $(A \to B) \notin \operatorname{Th}(\mathbb{L}_n)$.*

Proof We know that $\operatorname{Th}(p) \cap \mathcal{L} = \mathcal{L}_n = \operatorname{Th}(\mathbb{L}_n) \cap \mathcal{L}$. Suppose that $(A \to B) \notin \operatorname{Cn}(\mathcal{K}_n)$. Because of $\operatorname{Cn}(\mathcal{K}_n) = \operatorname{Cn}(\operatorname{Th}(p) \cap \mathcal{L}) = \operatorname{Cn}(\operatorname{Th}(\mathbb{L}_n) \cap \mathcal{L})$, we obtain $(A \to B) \notin \operatorname{Cn}(\operatorname{Th}(\mathbb{L}_n) \cap \mathcal{L})$. But this implies $(A \to B) \notin \operatorname{Th}(\mathbb{L}_n)$, since $(A \to B) \in \mathcal{L}$. □

We have now established the correctness of our algorithm. However, one crucial feature of classical attribute exploration is its *optimality* with respect to the number of implications asked to the expert. We shall see now that this property also holds in our general setting.

3.12 Theorem *Let $p, \mathcal{K}, \mathcal{L}$ be valid input for Algorithm 2, and let n be the last iteration of the algorithm. Then*

$$\operatorname{Can}(\operatorname{Th}(p) \cap \mathcal{L}, \mathcal{K}) = \mathcal{K}_n \backslash \mathcal{K}.$$

Proof We first observe that for each implication $(P_k \to \mathcal{L}_{k-1}(P_k)) \in \mathcal{K}_n \backslash \mathcal{K}$, it is true that $\mathcal{L}_{k-1}(P_k) = \mathcal{K}_n(P_k)$. This follows from the fact that

$$\mathcal{K}_n(P_k) \subseteq \mathcal{L}_{k-1}(P_k) \subseteq \mathcal{L}_n(P_k) = \mathcal{K}_n(P_k).$$

Secondly, it is easy to see that $\mathcal{K}_n \backslash \mathcal{K}$ is an irredundant base of $\mathcal{K}_n$ with background knowledge $\mathcal{K}$.

We are now going to show that the premises in $\mathcal{K}_n \backslash \mathcal{K}$ are actually all $\mathcal{K}$-pseudoclosed sets of $\mathcal{K}_n$. For this, we shall show by induction the following claim:

> For every iteration i, the $k := |\mathcal{K}_i \backslash \mathcal{K}|$ lectically first $\mathcal{K}$-pseudoclosed sets of $\mathcal{K}_n$ are precisely the premises of $\mathcal{K}_i \backslash \mathcal{K}$. If the $(k+1)$st $\mathcal{K}$-pseudoclosed set of $\mathcal{K}_n$ exists, then P_{i+1} exists and if Q denotes this set, then $P_{i+1} \preceq Q$.

The base case $i = 0$ is clear, as $|\mathcal{K}_0 \backslash \mathcal{K}| = 0$. If Q is the first $\mathcal{K}$-pseudoclosed set of $\mathcal{K}_n$, it is $\mathcal{K}$-closed, but not $\mathcal{K}_n$-closed. Then Q is also not $\mathcal{L}$-closed, since $\mathcal{K}_n \subseteq \mathcal{L}$. By construction, P_1 exists and $P_1 \preceq Q$.

For the induction step, assume that the claim is true for an iteration i. Let $k := |\mathcal{K}_i \backslash \mathcal{K}|$. If there is no more $\mathcal{K}$-pseudoclosed set of $\mathcal{K}_n$, then $\mathcal{K}_i$ is a $\mathcal{K}$-base of $\mathcal{K}_n$. Hence $\mathcal{K}_i = \mathcal{K}_n$, since $\mathcal{K}_n$ is irredundant. Then $i = n$, or $|\mathcal{K}_{i+1} \backslash \mathcal{K}| = k$ and the claim is true for iteration $i + 1$ as well.

Now assume that Q is the $(k+1)$st $\mathcal{K}$-pseudoclosed set of $\mathcal{K}_n$. By induction hypothesis, P_{i+1} exists and $Q \preceq P_{i+1}$. We distinguish two cases.

Case $Q = P_{i+1}$: If $\mathcal{L}_i(P_{i+1}) \subseteq \mathcal{K}_n(P_{i+1})$, then p accepts the implication $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, which is then an element of $\mathcal{K}_{i+1}$. Thus the $|\mathcal{K}_{i+1} \backslash \mathcal{K}| = k + 1$ lectically first $\mathcal{K}$-pseudoclosed sets of $\mathcal{K}_n$ are precisely the premises of $\mathcal{K}_{i+1} \backslash \mathcal{K}$.

If $\bar{Q}$ is the $(k+2)$nd $\mathcal{K}$-pseudoclosed set of $\mathcal{K}_n$, then in particular $\bar{Q}$ is closed under $\mathcal{K}_{i+1}$, as for each $\mathcal{K}$-pseudoclosed set $P_\ell \subseteq \bar{Q}$, it is true that $\mathcal{K}_n(P_\ell) \subseteq \bar{Q}$. Furthermore, $\bar{Q}$ is not $\mathcal{K}_n$-closed, thus also not $\mathcal{L}_{i+1}$-closed. Therefore, by construction the set P_{i+2} exists and $P_{i+2} \preceq Q$.

Now if $\mathcal{L}_i(P_{i+1}) \nsubseteq \mathcal{K}_n(P_{i+1})$, then p rejects the implication $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, and provides a counterexample. Then $\mathcal{K}_{i+1} = \mathcal{K}_i$. Since $P_{i+1} = Q$, the set P_{i+1} will not be $\mathcal{L}_{i+1}$-closed, since otherwise it would also be $\mathcal{K}_n$-closed. Thus $P_{i+2} = P_{i+1}$ and $P_{i+2} \preceq Q$, as required.

Case $P_{i+1} \prec Q$: In this case, the set P_{i+1} must be $\mathcal{K}_n$-closed, since otherwise it would be a $\mathcal{K}$-pseudoclosed set of $\mathcal{K}_n$ and $P_{i+1} \succeq Q$ would hold. To see this, first observe that

P_{i+1} is $\mathcal{K}_i$-closed by construction, hence also $\mathcal{K}$-closed. Furthermore, if $\bar{Q} \subsetneq P_{i+1}$ is a $\mathcal{K}$-pseudoclosed set of $\mathcal{K}_n$, then by induction hypothesis, $\bar{Q}$ is a premise of $\mathcal{K}_i$, hence $\mathcal{K}_n(\bar{Q}) \subseteq P_{i+1}$ is true as well.

Furthermore, P_{i+1} is not $\mathcal{L}_i$-closed by construction. However, since P_{i+1} is $\mathcal{K}_n$-closed, the expert p rejects the implication $P_{i+1} \to \mathcal{L}_i(P_{i+1})$. Then $\mathcal{K}_{i+1} = \mathcal{K}_i$ and P_{i+2} is the lectically smallest $\mathcal{K}_{i+1}$-closed, not $\mathcal{L}_{i+1}$-closed set which is lectically greater or equal to P_{i+1}. Since Q is also $\mathcal{K}_{i+1}$-closed (by induction hypothesis, as it is a $\mathcal{K}$-pseudoclosed set of $\mathcal{K}_n$) but not $\mathcal{L}_{i+1}$-closed (since Q is not $\mathcal{K}_n$-closed), it is true that $P_{i+2} \preceq Q$ by construction. The claim follows.

This finishes the proof of the inductive step.

We shall now use the claim to show the theorem. Since n is the last iteration of the algorithm, the set P_{n+1} does not exist. Therefore, there does not exist a $\mathcal{K}$-pseudoclosed set of $\mathcal{K}_n$ that is not a premise in $\mathcal{K}_n \backslash \mathcal{K}$. Thus

$$\begin{aligned}\mathcal{K}_n \backslash \mathcal{K} &= \operatorname{Can}(\mathcal{K}_n, \mathcal{K})\\ &= \operatorname{Can}(\mathcal{L}_n, \mathcal{K})\\ &= \operatorname{Can}(\operatorname{Th}(p) \cap \mathcal{L}, \mathcal{K}).\end{aligned}$$

Here, the third equality holds since $\mathcal{K}_n$ and $\mathcal{L}_n$ are equivalent, and the last equality holds since $\mathcal{L}_n = \operatorname{Th}(p) \cap \mathcal{L}$. □

3.3 A Weaker Generalization of Attribute Exploration

For our considerations about exploration by confidence it may be necessary to drop the optimality property of attribute exploration as it has been stated in Theorem 3.12. This may be due to the fact that computing $\mathcal{L}_i(P_{i+1})$ may be too costly, as the set $\mathcal{L}_i$ may not be given explicitly. For example, the set $\mathcal{L}_i$ can just be the set of all implications with confidence greater or equal for some $c \in [0, 1]$, which are invalidated by some already given counterexample (this will be the case for setting of exploration by confidence). Then, computing $\mathcal{L}_i(P_{i+1})$ may be infeasible.

When computing the closure $\mathcal{L}_i(P_{i+1})$ is too expensive, we have to resort to the heuristic that it is feasible to find at least *some* elements $P_{i+1} \subsetneq Q \subseteq \mathcal{L}_i(P_{i+1})$.[2] If we have found such elements, we simply ask the expert the resulting implication $P_{i+1} \to Q$. For all other elements $\mathcal{L}_i(P_{i+1}) \backslash Q$, we have to ensure that they are asked in another iteration. It may very well be that to ensure this, we also have to ask *additional* implications $X \to Y$ for which the set X is not closed under the set of currently known implications.

Finally, we have to be able to determine whether a set P is closed under the set $\mathcal{L}_i$ or not. Note that this may not require to compute the set $\mathcal{L}_i(P)$.

We collect these ideas in the following algorithm.

Algorithm 3 (A Weaker General Attribute Exploration)

[2] If even this is not feasible, then it is questionable whether exploration itself can be conducted effectively.

Input A domain expert p on a finite set M and sets $\mathcal{K}, \mathcal{L} \subseteq \operatorname{Imp}(M)$ such that $\mathcal{K} \subseteq \operatorname{Th}(p)$ and $\mathcal{K} \subseteq \mathcal{L}$.

Procedure

i. Initialize $i := 0, \mathcal{K}_i := \mathcal{K}, \mathcal{L}_i := \mathcal{L}, \mathbb{L}_i := (\varnothing, M, \varnothing)$. Let P_i be such that $P_i \preceq \mathcal{K}(\varnothing)$.

ii. Let P be the smallest $\mathcal{K}_i$-closed set lectically larger or equal to P_i, which is not $\mathcal{L}_i$-closed. If no such set exists, terminate.

   Otherwise, let P_{i+1} be such that $P_i \preceq P_{i+1} \preceq P$ and P_{i+1} not $\mathcal{L}_i$-closed. Let Q be such that $P_{i+1} \subsetneq Q \subseteq \mathcal{L}_i(P_{i+1})$.

iii. If p confirms $P_{i+1} \to Q$, then
   - $\mathcal{K}_{i+1} := \mathcal{K}_i \cup \{ P_{i+1} \to Q \}$,
   - $\mathcal{L}_{i+1} := \mathcal{L}_i$,
   - $\mathbb{L}_{i+1} := \mathbb{L}_i$.

iv. If p provides a counterexample C for $P_{i+1} \to Q$, then
   - $\mathcal{K}_{i+1} := \mathcal{K}_i$,
   - $\mathcal{L}_{i+1} := \{ (A \to B) \in \mathcal{L}_i \mid C \models (A \to B) \}$,
   - $\mathbb{L}_{i+1} := \mathbb{L}_i + C$.

v. Set $i := i + 1$ and go to ii.

Output Return $\mathcal{K}_i$ and $\mathbb{L}_i$. □

As this algorithm is quite similar to Algorithm 2, most of its argumentation is also valid for it. The main difference is that this algorithm is allowed to ask some questions "in between" those that would be asked in Algorithm 2.

We shall now show that this weaker form of attribute exploration is still correct. However, it can be seen easily that this algorithm is not optimal anymore.

Firstly, termination can be argued as in the case of Algorithm 2. Thus the analogue of Theorem 3.6 holds.

3.13 Theorem *Let $p, \mathcal{K}, \mathcal{L}$ be valid input for Algorithm 3. Then the algorithm with this input terminates after finitely many steps.*

Furthermore, it can be seen easily that Proposition 3.7 is also valid for our weaker form of attribute exploration, with the same proof. We shall repeat this proof here nevertheless for easier comparison.

3.14 Proposition *In every iteration i of Algorithm 3, it is true that for all $A \prec P_{i+1}$, if A is $\mathcal{K}_i$-closed, then A is also $\mathcal{L}_i$-closed.*

Proof We show the claim by induction on i. For $i = 0$, the claim is vacuously true, as P_0 is not lectically greater than the lectically smallest $\mathcal{K}_i$-closed set. Now assume the validity of the proposition for i. We then need to show that for each $A \prec P_{i+2}$, if A is $\mathcal{K}_{i+1}$-closed, then A is also $\mathcal{L}_{i+1}$-closed.

We only have to consider that case that $P_{i+1} \leq A < P_{i+2}$ and A being $\mathcal{K}_{i+1}$-closed. In particular, this means that $P_{i+1} < P_{i+2}$, i. e. $P_{i+1} \neq P_{i+2}$. Then $P_{i+1} = A$ is the only remaining case, since all sets lectically between P_{i+1} and P_{i+2} which are $\mathcal{K}_{i+1}$-closed are also $\mathcal{L}_{i+1}$-closed by construction of P_{i+2}. Since $P_{i+1} \neq P_{i+2}$ we know that then P_{i+1} must also be $\mathcal{L}_{i+1}$-closed, as otherwise the algorithm would choose $P_{i+2} = P_{i+1}$. This finishes the induction step and the proof of the proposition. □

3.15 Theorem *Let $p, \mathcal{K}, \mathcal{L}$ be valid input for Algorithm 3. Let n be the last iteration of the algorithm. Then $\operatorname{Cn}(\mathcal{K}_n) = \operatorname{Cn}(\mathcal{L}_n)$, $\mathcal{L}_n = \operatorname{Th}(p) \cap \mathcal{L}$ and $\operatorname{Cn}(\mathcal{K}_n) = \operatorname{Cn}(\operatorname{Th}(p) \cap \mathcal{L})$. Furthermore, for each $(A \to B) \in \mathcal{L}$, either $(A \to B) \in \operatorname{Cn}(\mathcal{K}_n)$ or $(A \to B) \notin \operatorname{Th}(\mathbb{L}_n)$.*

4 Exploration by Confidence

Let $\mathbb{K} = (G, M, I)$ be a finite and non-empty formal context, $c \in [0, 1]$, $\mathcal{K} \subseteq \operatorname{Imp}_c(\mathbb{K})$ and let p be a domain expert on M. In the following two subsections, we shall make use of our generalized attribute exploration algorithms from the previous section to develop two different algorithms which provide an exploration by confidence, as we it has already been described in Section 3.2. In a nutshell, we are interested in representing the set of implications in $\operatorname{Imp}_c(\mathbb{K})$ which are confirmed by p in a compact way using $\mathcal{K}$ as our background knowledge, i. e. we are looking for a base $\mathcal{B}$ of the set $\operatorname{Th}(p) \cap \operatorname{Imp}_c(\mathbb{K})$ with background knowledge $\mathcal{K}$. Preferably, this base is small, and minimal at best.

For the first algorithm, we shall use the generalization of Section 3.2. For this it is enough to discuss a way to compute the closure operator under a set of implications of the form

$$\operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(\mathbb{L}_i) \tag{1}$$

in a certain iteration i of Algorithm 2. As soon as we can compute this closure operator, Algorithm 2 can be used directly as an algorithm for exploration by confidence. This is what will be done in Section 4.1.

However, the computation of the closure under the implications from (1) may be too expensive for a practical exploration process. Because of this, we shall present in Section 4.2 another approach to exploration by confidence which is based on our weaker form of attribute exploration which we have discussed in Section 3.3. Within this algorithm, computing undecided implications will be much easier. However, we have to pay for this by loosing optimality of the number of questions confirmed by the expert.

4.1 Exploration by Confidence with Optimality

The approach of this section is quite simple: we instantiate the general attribute exploration algorithm from Section 1 with our particular setup. The only thing that remains to be done is to discuss how to compute the closure operator $\mathcal{L}_i(P_{i+1})$ (using the notation from Algorithm 2).

To make the following considerations easier to follow, let us first restate the generalized exploration algorithm from Section 3.1 adapted for our particular setting.

Algorithm 4 (Optimal Exploration by Confidence)

```
Input  A domain expert $p$ on a finite set $M$, a formal context $\mathbb{K}$, $c \in [0,1]$ and a set
       $\mathcal{K} \subseteq \operatorname{Imp}_c(\mathbb{K})$ such that $\mathcal{K} \subseteq \operatorname{Th}(p)$.

Procedure
    i.   Initialize $i := 0, P_i := \mathcal{K}(\varnothing), \mathcal{K}_i := \mathcal{K}, \mathcal{L}_i := \operatorname{Imp}_c(\mathbb{K}), \mathbb{L}_i := (\varnothing, M, \varnothing)$.
    ii.  Let $P_{i+1}$ be the smallest $\mathcal{K}_i$-closed set lectically larger or equal to $P_i$, which
         is not $\mathcal{L}_i$-closed. If no such set exists, terminate.
    iii. If $p$ confirms $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, then
           • $\mathcal{K}_{i+1} := \mathcal{K}_i \cup \{ P_{i+1} \to \mathcal{L}_i(P_{i+1}) \}$,
           • $\mathcal{L}_{i+1} := \mathcal{L}_i$,
           • $\mathbb{L}_{i+1} := \mathbb{L}_i$.
    iv.  If $p$ provides a counterexample $C$ for $P_{i+1} \to \mathcal{L}_i(P_{i+1})$, then
           • $\mathcal{K}_{i+1} := \mathcal{K}_i$,
           • $\mathcal{L}_{i+1} := \{ (A \to B) \in \mathcal{L}_i \mid C \models (A \to B) \}$,
           • $\mathbb{L}_{i+1} := \mathbb{L}_i + C$.
    v.   Set $i := i + 1$ and go to ii.

Output  Return $\mathcal{K}_i$ and $\mathbb{L}_i$.                                          □
```

From Theorem 3.6, Corollary 3.9 and Theorem 3.12 we immediately obtain that Algorithm 4 is a correct and optimal with respect to the number of questions confirmed by the expert.

4.1 Corollary *Let $\mathbb{K} = (G, M, I)$ be a finite and non-empty formal context, $c \in [0,1]$, p be a domain expert on M and $\mathcal{K} \subseteq \operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(p)$. Then Algorithm 4 terminates with input p, c and $\mathcal{K}$. Let n be the last iteration of this run of the algorithm. Then*

i. $\operatorname{Cn}(\mathcal{K}_n) = \operatorname{Cn}(\operatorname{Th}(p) \cap \operatorname{Imp}_c(\mathbb{K}))$.

ii. $\operatorname{Can}(\operatorname{Th}(p) \cap \operatorname{Imp}_c(\mathbb{K}), \mathcal{K}) = \mathcal{K}_n \backslash \mathcal{K}$.

To make this algorithm effectively usable, we need a way to compute $\mathcal{L}_i(A)$ for sets $A \subseteq M$. Of course, since $\mathcal{L}_i = \operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(\mathbb{L}_i)$, this could be done by just enumerating all elements of $\operatorname{Imp}_c(\mathbb{K})$, and using these implications directly. Clearly, this is not very practical. Therefore, in the remainder of this section we concentrate on different ways on how to compute the closure of A under the set $\operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(\mathbb{L}_i)$.

The following considerations will involve derivations in a number of formal contexts, namely in $\mathbb{K}$, $\mathbb{L}_i$ and in the *subposition* of $\mathbb{K}$ and $\mathbb{L}_i$, i. e. in the formal context

$$\frac{\mathbb{K}}{\mathbb{L}_i} := (G \cup G_i, M, I \cup I_i),$$

where $\mathbb{L}_i = (G_i, M, I_i)$, and where we assume both G, G_i and I, I_i to be disjoint. To cause no confusion, we shall denote the derivation in $\mathbb{K}$ by $\cdot''_{\mathbb{K}}$, the derivation in $\mathbb{L}_i$ by $\cdot''_{\mathbb{L}_i}$ and the derivation in $\frac{\mathbb{K}}{\mathbb{L}_i}$ by $''^{i}$.

Let us start with some simple observations. First of all, it is quite easy to see that

$$A''^{i} \subseteq \mathcal{L}_i(A). \tag{2}$$

The fact that this is true is because the implication $A \to A''_{\mathbb{K}}$ is valid in $\mathbb{K}$ and thus $(A \to A''_{\mathbb{K}}) \in \operatorname{Imp}_c(\mathbb{K})$. Since $A \subseteq A''^{i} \subseteq A''_{\mathbb{K}}$, the implication $A \to A''^{i}$ is valid in $\mathbb{K}$ as well. Furthermore, this implication is also valid in $\mathbb{L}_i$, hence

$$(A \to A''^{i}) \in \operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(\mathbb{L}_i) = \mathcal{L}_i.$$

Therefore, $A''^{i} \subseteq \mathcal{L}_i(A)$.

Recall that the set $\mathcal{L}_i(A)$ is computed as

$$\mathcal{L}_i(A) = \bigcup_{k \in \mathbb{N}_{>0}} \mathcal{L}_i^k(A),$$

where

$$\begin{aligned} \mathcal{L}_i^1(A) &= A \cup \{\, Y \mid (X \to Y) \in \mathcal{L}_i, X \subseteq A \,\} \\ \mathcal{L}_i^{k+1}(A) &= \mathcal{L}_i^k(\mathcal{L}_i^1(A)) \end{aligned}$$

for $k \in \mathbb{N}$. By our previous considerations we know that $A''^{i} \subseteq \mathcal{L}_i^1(A)$. To compute the set $\mathcal{L}_i^1(A)$ completely, we have to find all implications $(X \to Y) \in \mathcal{L}_i$ such that $X \subseteq A$. Note that if $X \to Y$ is a valid implication of $\mathbb{K}$ and $\mathbb{L}_i$, then we already know that $Y \subseteq A''^{i}$. Thus it is sufficient to search only for those implications $X \to Y$ which are not valid in $\mathbb{K}$, i. e. where

$$1 > \operatorname{conf}_{\mathbb{K}}(X \to Y) \geqslant c \tag{3}$$

is true. Note that we can assume that $|Y| = 1$, since

$$\operatorname{conf}_{\mathbb{K}}(X \to \{\, y \,\}) \geqslant \operatorname{conf}_{\mathbb{K}}(X \to Y)$$

is true for all $y \in Y$. Note also that $(X \to Y) \in \operatorname{Th}(\mathbb{L}_i)$, therefore $Y \subseteq X''_{\mathbb{L}_i}$.

To sum up, we essentially have to consider all subsets $X \subseteq A$, and all elements in $y \in X''_{\mathbb{L}_i} \setminus A$ for which we have to check whether

$$\operatorname{conf}_{\mathbb{K}}(X \to \{\, y \,\}) \geqslant c$$

is true or not. All such y then constitute the missing part of $\mathcal{L}_i^1(A)$.

4.2 Proposition *Let $\mathbb{K}$ and $\mathbb{L}_i$ be two formal contexts with attribute set M and disjoint object sets. Furthermore, let $c \in [0,1]$. Define $\mathcal{L}_i = \operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(\mathbb{L}_i)$. Then for each $A \subseteq M$ it is true that*

$$\mathcal{L}_i^1(A) = A''^{i} \cup \{\, y \in A''_{\mathbb{L}_i} \mid \exists X \subseteq A \colon y \in X''_{\mathbb{L}_i} \setminus A''^{i} \wedge \operatorname{conf}_{\mathbb{K}}(X \to \{\, y \,\}) \geqslant c \,\},$$

where the derivation operators are denoted as before.

It may be quite difficult for an element $y \in \mathcal{L}_i(A) \backslash A''^{i}$ to find a set X such that $y \in X''_{\mathbb{L}_i} \backslash A$ and $\operatorname{conf}_{\mathbb{K}}(X \to \{\, y \,\}) \geqslant c$. It is thusly desirable to reduce the number of sets X we have to consider for this search.

A first observation into this direction is the well-known fact that

$$\operatorname{conf}_{\mathbb{K}}(X \to \{\, y \,\}) = \operatorname{conf}_{\mathbb{K}}(X''_{\mathbb{K}} \to \{\, y \,\}).$$

This idea is known in the realm of data mining as the approach to consider only *frequent closed itemsets*: it is enough to consider only intents of $\mathbb{K}$ as premises for the implications we are looking for.

Using this idea we can proceed as follows: when searching for a set for y, it is not necessary to consider two different sets $X_1, X_2 \subseteq A$ if $(X_1)''_{\mathbb{L}_i} = (X_2)''_{\mathbb{L}_i}$ and $(X_1)''_{\mathbb{K}} = (X_2)''_{\mathbb{K}}$. However, this means that $(X_1)''^{i} = (X_2)''^{2}$, since

$$\begin{aligned}(X_1)''^{i} &= (X_1)''_{\mathbb{L}_i} \cap (X_1)''_{\mathbb{K}} \\ &= (X_2)''_{\mathbb{L}_i} \cap (X_2)''_{\mathbb{K}} \\ &= (X_2)''^{i}.\end{aligned}$$

On the other hand $(X_1)''^{i} = (X_2)''^{2}$ readily implies

$$(X_1)''_{\mathbb{L}_i} = ((X_1)''^{i})''_{\mathbb{L}_i} = ((X_2)''^{i})''_{\mathbb{L}_i} = (X_2)''_{\mathbb{L}_i}$$

since $X_1 \subseteq (X_1)''^{i} \subseteq (X_1)''_{\mathbb{L}_i}$. In the same way it can be shown that $(X_1)''_{\mathbb{K}} = (X_2)''_{\mathbb{K}}$. Therefore,

$$(X_1)''_{\mathbb{L}_i} = (X_2)''_{\mathbb{L}_i} \wedge (X_1)''_{\mathbb{K}} = (X_2)''_{\mathbb{K}} \iff (X_1)''^{i} = (X_2)''^{i}. \tag{4}$$

The essence of this equivalence is that in Proposition 4.2, instead of considering all subsets of A, we only have to consider intents X of the subposition of $\mathbb{K}$ and $\mathbb{L}_i$ which are generated by subsets of A. But we can also relax this condition in the following way:

$$\begin{aligned}\mathcal{L}_i^{1,\mathrm{conf}}(A) &= A''^{i} \cup \{\, y \in A''_{\mathbb{L}_i} \mid \exists X''^{i} \subseteq A''^{i} \colon y \in X''_{\mathbb{L}_i} \backslash A''^{i} \wedge \operatorname{conf}_{\mathbb{K}}(X''_{\mathbb{K}} \to \{\, y \,\}) \geqslant c \,\}, \\ \mathcal{L}_i^{k+1,\mathrm{conf}}(A) &= \mathcal{L}_i^{k,\mathrm{conf}}(\mathcal{L}_i^{1,\mathrm{conf}}(A)), \\ \mathcal{L}_i^{\mathrm{conf}}(A) &= \bigcup_{i \in \mathbb{N}_{>0}} \mathcal{L}_i^{1,\mathrm{conf}}(A),\end{aligned}$$

(note that we now search for intents which are subsets of A''^{i} instead of subsets of A). Then the following statement is true.

4.3 Proposition *Let $\mathbb{K}$ and $\mathbb{L}_i$ be two formal contexts with attribute set M and disjoint object sets. Furthermore, let $c \in [0,1]$. Define $\mathcal{L}_i = \operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(\mathbb{L}_i)$. Then for each $A \subseteq M$ it is true that*

$$\mathcal{L}_i^{1}(A) \subseteq \mathcal{L}_i^{1,\mathrm{conf}}(A) \subseteq \mathcal{L}_i(A).$$

In particular, $\mathcal{L}_i^{\mathrm{conf}}(A) = \mathcal{L}_i(A)$.

Note that we can use the NextClosure algorithm to efficiently enumerate the intents of the subposition of $\mathbb{K}$ and $\mathbb{L}_i$ which are subsets of A''^{i}.

Our final reduction of the search space for the sets X uses a lower bound size of $|X'_{\mathbb{K}}|$. More specifically, we know that the sets X we are interested in satisfy the constraint

$$1 > \operatorname{conf}_{\mathbb{K}}(X \to \{\, y \,\}) \geqslant c.$$

This immediately implies $X'_{\mathbb{K}} \neq \varnothing$. More generally, it is true that

$$|X'_{\mathbb{K}}| > |X'_{\mathbb{K}} \cap \{\, y \,\}'_{\mathbb{K}}|$$

because there exists at least on object in $X'_{\mathbb{K}}$ which does not have the attribute y. Moreover, the condition $\operatorname{conf}_{\mathbb{K}}(X \to \{\, y \,\}) \geqslant c$ means nothing else but

$$|X'_{\mathbb{K}} \cap \{\, y \,\}'_{\mathbb{K}}| \geqslant c \cdot |X'_{\mathbb{K}}|.$$

From the two inequalities above, we can infer $|X'_{\mathbb{K}}| - 1 \geqslant c \cdot |X'_{\mathbb{K}}|$, or equivalently $|X'_{\mathbb{K}}| \geqslant (1-c)^{-1}$, if $c \neq 1$. So, to compute $\mathcal{L}_1^{1,\mathrm{conf}}(A)$, it is enough to consider only those intents X of the subposition of $\mathbb{K}$ and $\mathbb{L}_i$ which satisfy $|X'_{\mathbb{K}}| \geqslant (1-c)^{-1}$. Note that the NextClosure algorithm can be adapted to this setting, i. e. it can be modified in such a way that it only enumerates intents X which satisfy this cardinality constraint.

4.2 A Faster Exploration by Confidence

The algorithm for computing closures of the form $\mathcal{L}_i(A)$ which we have described in the previous section might be not efficient enough for practical purposes. Even worse, in classical attribute exploration it may be the case that the time between to questions grows exponentially with the number of questions already asked [Dis10], see [BD11] for an example of this on a real-world data set. The computation of closures under $\mathcal{L}_i$ combined with this quite disadvantageous property may render the whole algorithm impractical.

To circumvent this defect of our current form of exploration by confidence, we shall make use of our weaker form of attribute exploration we have developed in Section 3.3. To obtain from the weak form of attribute exploration an algorithm for exploration by confidence, we have to consider the questions on how to decide whether a set is closed under $\mathcal{L}_i$ and on how to compute the sets P_{i+1} and Q from the algorithm.

Let $A \subseteq M$ be a set. To decide whether A is closed under the set $\mathcal{L}_i$, it would be ideal if we could do the following (recall that $\mathcal{L}_i = \operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(\mathbb{L}_i)$):

$$A = \mathcal{L}_i(A) \iff \forall m \in A''_{\mathbb{L}_i} \setminus A \colon \operatorname{conf}_{\mathbb{K}}(A \to \{\, m \,\}) < c.$$

In other words, the set A is closed if and only if all attributes $m \in A''_{\mathbb{L}_i}$ such that $\operatorname{conf}_{\mathbb{K}}(A \to \{\, m \,\}) \geqslant c$ are already elements of A. So, instead of considering all subsets of X and to check whether $\operatorname{conf}_{\mathbb{K}}(X \to \{\, m \,\}) \geqslant c$, it would be sufficient to just check the set A itself.

Of course, the above mentioned equivalence is not true in general, only the direction from left to right is correct. On the other hand, the sets we check for closeness in our exploration algorithm always have an additional property: they are closed under the set

$\mathcal{K}_i$ of already confirmed implications. We shall therefore try to construct the set $\mathcal{K}_i$ in such a way that the above equivalence is true, thus enormously simplifying the test for closeness under $\mathcal{L}_i$.

To this end, we start with the following observation.

4.4 Proposition *Let $\mathbb{K} = (G, M, I)$ be a formal context and let $c \in [0, 1]$. Let $\mathbb{L}_i$ be another formal context with attribute set M and object set disjoint to G. Define $\mathcal{L}_i = \operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(\mathbb{L}_i)$.*

Let $A \subseteq M$ and let $\mathcal{K}_i \subseteq \operatorname{Cn}(\mathcal{L}_i)$ be such that for every intent $X \subsetneq A$ of $\frac{\mathbb{K}}{\mathbb{L}_i}$, it is true that

$$\forall m \in X''_{\mathbb{L}_i}\colon\ \operatorname{conf}_{\mathbb{K}}(X \to \{\, m \,\}) \geqslant c \implies m \in \mathcal{K}_i(X). \tag{5}$$

Suppose that A is closed under $\mathcal{K}_i$. Then A is closed under $\mathcal{L}_i$ if and only if

$$A = A''^{i} \text{ and } \forall m \in A''_{\mathbb{L}_i} \backslash A\colon\ \operatorname{conf}_{\mathbb{K}}(A \to \{\, m \,\}) < c. \tag{6}$$

Proof Suppose that A is closed under $\mathcal{L}_i$. Since the implication $A \to A''^{i}$ is valid in $\mathbb{K}$ and $\mathbb{L}_i$, it is true that $(A \to A''^{i}) \in \mathcal{L}_i$. Therefore, $A = \mathcal{L}(A) \supseteq A''^{i}$. On the other hand, $A \subseteq A''^{i}$, therefore $A = A''^{i}$. Furthermore, if $m \in A''_{\mathbb{L}_i}$ and $\operatorname{conf}_{\mathbb{K}}(A \to \{\, m \,\}) \geqslant c$, then $(A \to \{\, m \,\}) \in \mathcal{L}_i$, thus $m \in \mathcal{L}_i(A) = A$.

Now suppose that A is not closed under $\mathcal{L}_i$ and additionally that $A = A''^{i}$ is true. Then there exists a set $X \subseteq A$ and an attribute $m \in A''_{\mathbb{L}_i} \backslash A$ such that $(X \to \{\, m \,\}) \in \mathcal{L}_i$. Since $A = A''^{i}$, $X \subseteq A$ implies $X''^{i} \subseteq A$. Suppose that $X''^{i} \subsetneq A$. Then $\operatorname{conf}_{\mathbb{K}}(X''^{i} \to \{\, m \,\}) = \operatorname{conf}_{\mathbb{K}}(X \to \{\, m \,\}) \geqslant c$, thus $m \in \mathcal{K}_i(X''^{i}) \subseteq \mathcal{K}_i(A) = A$, a contradiction. Therefore, $X''^{i} = A$ and we have shown that the attribute $m \in A''_{\mathbb{L}_i} \backslash A$ satisfies $\operatorname{conf}_{\mathbb{K}}(A \to \{\, m \,\}) \geqslant c$. □

This proposition will guide our further development of the algorithm. Essentially, what we have to ensure is the correctness of property (5) for all intents of $\frac{\mathbb{K}}{\mathbb{L}_i}$ which are lectically before the premise we consider in iteration i. To this end, we shall ask additional questions: in addition to asking the expert implications $A \to B$, where the set A is $\mathcal{K}_i$-closed, we shall also ask questions of the form $A \to \{\, m \,\}$, where A is an intent of $\frac{\mathbb{K}}{\mathbb{L}_i}$ and $m \in A''_{\mathbb{L}_i} \backslash \mathcal{K}_i(A)$ satisfies $\operatorname{conf}_{\mathbb{K}}(A \to \{\, m \,\}) \geqslant c$. We formalize this idea in Algorithm 5.

To prove that this algorithm indeed implements our perception of an exploration by confidence, we shall show that Algorithm 5 is of the form of Algorithm 3, i. e. is an instance of our weak generalization of attribute exploration, where $\mathcal{L} = \operatorname{Imp}_c(\mathbb{K})$. To this end, we have to argue that

i. $P_0 \preceq \mathcal{K}(\varnothing)$ and

ii. in every iteration i, the lectically smallest $\mathcal{K}_i$-closed, not $\mathcal{L}_i$-closed set $\bar{P}$ lectically greater or equal to P_i satisfies

$$P_i \preceq P_{i+1} \preceq \bar{P}. \tag{7}$$

Furthermore, P_{i+1} is not $\mathcal{L}_i$-closed and Q satisfies

$$P_{i+1} \subsetneq Q \subseteq \mathcal{L}_i(P_{i+1}).$$

Algorithm 5 (Exploration by Confidence with Faster Generation of Questions)

Input A domain expert p on a finite set M, a formal context $\mathbb{K}$, $c \in [0,1]$ and a set $\mathcal{K} \subseteq \operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(p)$.

Procedure

i. Initialize $i := 0, \mathcal{K}_i := \mathcal{K}, \mathcal{L}_i := \operatorname{Imp}_c(\mathbb{K}), \mathbb{L}_i := (\varnothing, M, \varnothing)$. Let $P_i := \min_{\leq}(\mathcal{K}_i(\varnothing), \varnothing^{\prime\prime_i})$.

ii. Let P^1_{i+1} be the lectically smallest intent of $\frac{\mathbb{K}}{\mathbb{L}_i}$ greater or equal to P_i such that there exists an attribute $m \in (P^1_{i+1})''_{\mathbb{L}_i} \backslash \mathcal{K}_i(P^1_{i+1})$ satisfying $\operatorname{conf}_{\mathbb{K}}(P^1_{i+1} \to \{ m \}) \geqslant c$. If no such set exists, let $P^1_{i+1} = M$. Otherwise, set $Q^1_{i+1} = P^1_{i+1} \cup \{ m \}$.

    Let P^2_{i+1} be the lectically smallest closed set of $\mathcal{K}_i$ greater or equal to P_i which is not an intent of $\frac{\mathbb{K}}{\mathbb{L}_i}$. If no such set exists, set $P^2_{i+1} = M$. Otherwise, set $Q^2_{i+1} = (P^2_{i+1})^{\prime\prime_i}$.

    Define $P_{i+1} = \min_{\leq}(P^1_{i+1}, P^2_{i+1})$. If $P_{i+1} = M$, terminate. Otherwise, if $P_{i+1} = P^1_{i+1}$, then define $Q_{i+1} = Q^1_{i+1}$, else $Q_{i+1} = Q^2_{i+1}$.

iii. If p confirms $P_{i+1} \to Q_{i+1}$, then

- $\mathcal{K}_{i+1} := \mathcal{K}_i \cup \{ P_{i+1} \to Q_{i+1} \}$,
- $\mathcal{L}_{i+1} := \mathcal{L}_i$,
- $\mathbb{L}_{i+1} := \mathbb{L}_i$.

iv. If p provides a counterexample C for $P_{i+1} \to Q_{i+1}$, then

- $\mathcal{K}_{i+1} := \mathcal{K}_i$,
- $\mathcal{L}_{i+1} := \{ (A \to B) \in \mathcal{L}_i \mid C \models (A \to B) \}$,
- $\mathbb{L}_{i+1} := \mathbb{L}_i + C$.

v. Set $i := i + 1$ and go to ii.

Output Return $\mathcal{K}_i$ and $\mathbb{L}_i$. □

This is enough, since the steps iii, iv and v are identical in both algorithms.

The first statement is obviously true, since $P_0 := \min_{\leq}(\mathcal{K}(\varnothing), \varnothing''^{i})$. Also the condition on Q is clearly satisfied: if $P_{i+1} = P^1_{i+1}$, then $Q = P^1_{i+1} \cup \{ m \}$, where $m \in (P^1_{i+1})''_{\mathbb{L}_i}$ and $\operatorname{conf}_{\mathbb{K}}(P^1_{i+1} \to \{ m \}) \geqslant c$. Then $(P^1_{i+1} \to \{ m \}) \in \operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(\mathbb{L}_i) = \mathcal{L}_i$ and therefore $Q \subseteq \mathcal{L}_i(P_{i+1})$. If $P_{i+1} = P^2_{i+1}$, then $Q = (P^2_{i+1})''^{i}$. Then $P^2_{i+1} \to Q$ is valid in $\mathbb{K}$ and thus has confidence 1. It is also valid in $\mathbb{L}_i$, thus $(P^2_{i+1} \to Q) \in \operatorname{Imp}_c(\mathbb{K}) \cap \operatorname{Th}(\mathbb{L}_i) = \mathcal{L}_i$, therefore $Q \subseteq \mathcal{L}_i(P_{i+1})$ also holds in this case.

It thus only remains to prove the correctness of (7). For this, we shall first show that condition (5) is indeed satisfied in every iteration of the algorithm.

4.5 Proposition *Let i be an iteration of a run of Algorithm 5. Then for all intents $X \prec P_i$ of $\frac{\mathbb{K}}{\mathbb{L}_i}$ it is true that*

$$\forall m \in X''_{\mathbb{L}_i} \colon \operatorname{conf}_{\mathbb{K}}(X \to \{ m \}) \geqslant c \implies m \in \mathcal{K}_i(X). \tag{8}$$

Proof We show the claim by induction. For $i = 0$, the claim is vacuously true. Thus suppose the claim (8) for i. Then to show (8) for $i+1$, let $X \prec P_{i+1}$ be an intent of $\frac{\mathbb{K}}{\mathbb{L}_{i+1}}$ and $m \in X''_{\mathbb{L}_{i+1}}$ such that $\operatorname{conf}_{\mathbb{K}}(X \to \{ m \}) \geqslant c$. We need to show that $m \in \mathcal{K}_{i+1}(X)$. To this end, we distinguish two cases.

Case $X \prec P_i$: If $\mathbb{L}_i = \mathbb{L}_{i+1}$, then X being an intent of $\frac{\mathbb{K}}{\mathbb{L}_{i+1}}$ is trivially also an intent of $\frac{\mathbb{K}}{\mathbb{L}_i}$. By induction hypothesis we obtain that $m \in \mathcal{K}_i(X)$. Since $\mathcal{K}_i \subseteq \mathcal{K}_{i+1}$, we obtain $m \in \mathcal{K}_{i+1}(X)$ as required.

If $\mathbb{L}_i \neq \mathbb{L}_{i+1}$, then a counterexample C has been added to $\mathbb{L}_i$ to obtain $\mathbb{L}_{i+1}$. Since C is a counterexample for the implication $P_{i+1} \to Q_{i+1}$, it follows that $P_{i+1} \subseteq C$. In particular, $X \prec P_{i+1} \preceq C$. Therefore, $C \nsubseteq X$ and it follows that $X''_{\mathbb{L}_i} = X''_{\mathbb{L}_{i+1}}$. Thus X being an intent of $\frac{\mathbb{K}}{\mathbb{L}_{i+1}}$ is also an intent of $\frac{\mathbb{K}}{\mathbb{L}_i}$. Again, by induction hypothesis, it is true that $m \in \mathcal{K}_i(X) = \mathcal{K}_{i+1}(X)$.

Case $P_i \preceq X \prec P_{i+1}$: Since $P_{i+1} \preceq P^1_{i+1}$, it follows that X being an intent of $\frac{\mathbb{K}}{\mathbb{L}_i}$ also satisfies $m \in \mathcal{K}_i(X)$. □

We now use this result to show (7).

4.6 Proposition *Let i be an iteration of a run of Algorithm 5, and let $\bar{P}$ be the lectically smallest $\mathcal{K}_i$-closed lectically greater or equal to P_i which is not $\mathcal{L}_i$-closed. Then $P_{i+1} \preceq \bar{P}$.*

Proof We first observe that P^2_{i+1} is $\mathcal{K}_i$-closed, but not $\mathcal{L}_i$-closed, since $(P^2_{i+1})''^{i} \neq P^2_{i+1}$. This implies $\bar{P} \preceq P^2_{i+1}$. If $\bar{P} = P^2_{i+1}$, we are done. Therefore, we assume that $\bar{P} \prec P^2_{i+1}$.

In this case, by the construction of P^2_{i+1} and since $\bar{P}$ is $\mathcal{K}_i$-closed, the set $\bar{P}$ must be an intent of $\frac{\mathbb{K}}{\mathbb{L}_i}$. Since $\bar{P}$ is not $\mathcal{L}_i$-closed, there exists $m \in \bar{P}''_{\mathbb{L}_i} \setminus \bar{P}$ and $\hat{P} \subseteq \bar{P}$ satisfying $\operatorname{conf}_{\mathbb{K}}(\hat{P} \to \{ m \}) \geqslant c$. Additionally, we can assume that $\hat{P}$ is an intent of $\frac{\mathbb{K}}{\mathbb{L}_i}$. If $\hat{P} \prec P_i$, then by Proposition 4.5 it would be true that $m \in \mathcal{K}_i(\hat{P}) \subseteq \mathcal{K}_i(\bar{P}) = \bar{P}$, a contradiction. Therefore, $P_i \preceq \hat{P}$. Since $\hat{P} \subseteq \bar{P}$, it is true that $\hat{P} \preceq \bar{P}$. By construction, $P^1_{i+1} \preceq \hat{P}$, and thus $P^1_{i+1} \preceq \hat{P} \preceq \bar{P}$, as required. □

By establishing these results we have shown that Algorithm 5 is indeed of the form of our weak generalization of attribute exploration, namely Algorithm 3. In particular, Algorithm 5 has all the properties Algorithm 3 has.

4.7 Corollary *Let $p, \mathbb{K}, c$ and $\mathcal{K}$ be valid input for Algorithm 5. Then the algorithm with this input will terminate after finitely many steps. Let n be the last iteration of the algorithm. Then $\mathrm{Cn}(\mathcal{K}_n) = \mathrm{Cn}(\mathcal{L}_n), \mathcal{L}_n = \mathrm{Th}(p) \cap \mathrm{Imp}_c(\mathbb{K})$ and for each $(A \to B) \in \mathrm{Imp}_c(\mathbb{K})$ it is true that either $(A \to B) \in \mathrm{Cn}(\mathcal{K}_n)$ or $(A \to B) \notin \mathrm{Th}(\mathbb{L}_n)$.*

5 Future Work

The goal of this work was to develop a generalization of the classical attribute exploration algorithm to allow for the exploration of the confident implications of a formal context. For this generalized formulation we have shown in Section 3.2 that the main properties of attribute exploration are retained. In particular, this generalized formulation is optimal in the sense that the number of confirmed implications is as small as possible. We have also seen, by developing a weaker form of our generalized attribute exploration in Section 3.3, that we can relax the constraints on how to choose implications to be asked to the expert. However, this relaxation causes in general the exploration algorithm not to be optimal anymore. On the other hand, the relaxation may allow a computational simplification.

In Section 4, following the considerations on generalized formulations of attribute exploration, we have turned our attention to the special case of exploration by confidence. After formally specifying the actual problem, we have used our generalized formulations of attribute exploration to develop two algorithms which provide exploration by confidence. The first algorithm has been based on the generalization which retained optimality, and the only step that remained to make it applicable for exploration by confidence was to argue on how to compute the closures under sets of implications of the form $\mathrm{Imp}_c(\mathbb{K}) \cap \mathrm{Th}(\mathbb{L}_i)$. This has been discussed in detail in Section 4.1. Although this can be done effectively, the actual computation might be too costly. For this case, we have made use of the weaker form of attribute exploration to exploit a special strategy of undecided implications, discussed in Section 4.2. This strategy allowed us to compute these implications more easily, accepting the fact that we may loose the optimality of the exploration algorithm.

The algorithms we have presented in Section 4 are effective and can easily be implemented. Such an implementation would give more insight into the actual behavior of these algorithms, both in runtime and resulting bases. In particular, it would be interesting to know if the computation of closures under $\mathrm{Imp}_c(\mathbb{K}) \cap \mathrm{Th}(\mathbb{L}_i)$ is really expensive or not. Furthermore, experimentally comparing the questions asked by Algorithm 4 and Algorithm 5 would give additional insight into the actual overhead which results from the non-optimal asking behavior of Algorithm 5.

Another direction of research which should be quite promising is to generalize our algorithms for exploration by confidence to the setting of the description logic $\mathcal{EL}^{\perp}$, in the same way as classical attribute exploration has been generalized into this setting by [BD09]. This would extent the results of [Bor12b; Bor12a] to allow the construction of TBoxes from erroneous data to include both confidence and expert interaction.

References

[AIS93] R. Agrawal, T. Imielinski, and A. Swami. “Mining Association Rules between Sets of Items in Large Databases”. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data.* May 1993, pp. 207–216.

[BD09] Franz Baader and Felix Distel. “Exploring Finite Models in the Description Logic $\mathcal{EL}_{\text{gfp}}$”. In: *Proceedings of the 7th International Conference on Formal Concept Analysis.* (Darmstadt, Germany). Ed. by Sébastien Ferré and Sebastian Rudolph. Vol. 5548. Lecture Notes in Computer Science. Springer, May 21–24, 2009, pp. 146–161.

[BD11] Daniel Borchmann and Felix Distel. “Mining of $\mathcal{EL}$-GCIs”. In: *ICDM Workshops.* Ed. by Myra Spiliopoulou et al. IEEE, 2011, pp. 1083–1090. ISBN: 978-0-7695-4409-0.

[Bor11] Daniel Borchmann. “Decomposing Finite Closure Operators by Attribute Exploration”. In: *Contributions to ICFCA 2011.* (Nicosia, Cyprus). Ed. by Florent Domenach, Robert Jäschke, and Petko Valtchev. University of Nicosia, May 2011.

[Bor12a] Daniel Borchmann. *Axiomatizing Confident $\mathcal{EL}^{\perp}_{gfp}$-GCIs of Finite Interpretations.* Report MATH-AL-08-2012. Dresden, Germany: Chair of Algebraic Structure Theory, Institute of Algebra, Technische Universität Dresden, Sept. 2012.

[Bor12b] Daniel Borchmann. *On Confident GCIs of Finite Interpretations.* LTCS-Report 12-06. See `http://lat.inf.tu-dresden.de/research/reports.html`. Dresden: Institute for Theoretical Computer Science, TU Dresden, 2012.

[Bor13] Daniel Borchmann. *A General Form of Attribute Exploration.* LTCS-Report 13-02. See http://lat.inf.tu-dresden.de/research/reports.html. Dresden, Germany: Chair of Automata Theory, Institute of Theoretical Computer Science, Technische Universität Dresden, 2013.

[Dis10] Felix Distel. “Hardness of Enumerating Pseudo-intents in the Lectic Order”. In: *Proceedings of the 8th Interational Conference of Formal Concept Analysis.* (Agadir, Morocco). Ed. by Léonard Kwuida and Barış Sertkaya. Vol. 5986. Lecture Notes in Computer Science. Springer, Mar. 2010, pp. 124–137.

[Dis11] Felix Distel. “Learning Description Logic Knowledge Bases from Data Using Methods from Formal Concept Analysis”. PhD thesis. TU Dresden, 2011.

[Gan10] Bernhard Ganter. “Two Basic Algorithms in Concept Analysis”. In: *Proceedings of the 8th Interational Conference of Formal Concept Analysis.* (Agadir, Morocco). Ed. by Léonard Kwuida and Barış Sertkaya. Vol. 5986. Lecture Notes in Computer Science. Springer, Mar. 2010, pp. 312–340.

[Gan99] Bernhard Ganter. “Attribute Exploration with Background Knowledge”. In: *Theoretical Computer Science* 217.2 (1999), pp. 215–233.

[GW99] Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Berlin-Heidelberg: Springer, 1999.

[KS10] Léonard Kwuida and Barış Sertkaya, eds. (Agadir, Morocco). Vol. 5986. Lecture Notes in Computer Science. Springer, Mar. 2010.

[Stu96] Gerd Stumme. “Attribute Exploration with Background Implications and Exceptions”. In: *Data Analysis and Information Systems. Statistical and Conceptual approaches. Proc. GfKl’95. Studies in Classification, Data Analysis, and Knowledge Organization 7*. Ed. by H.-H. Bock and W. Polasek. Heidelberg: Springer, 1996, pp. 457–469.