Parameterized simplification logic I: Reasoning with implications and classes of closure operators

Pablo Cordero, Manuel Enciso, Angel Mora[a] and Vilem Vychodil[b]

[a]Universidad de Málaga, Andalucía Tech, Spain; [b]Czechia

ARTICLE HISTORY
Compiled October 6, 2019

ABSTRACT
In this paper, we present a general inference system for reasoning with if-then dependencies. The dependencies are defined using general lattice-theoretic notions and their semantics is defined using particular closure operators parameterized by systems of isotone Galois connections. In this general setting, we introduce a simplification logic, show its sound and complete axiomatization, and deal with related issues. The presented results can be seen as forming parameterized framework for dealing with if-then dependencies that allows to focus on particular dependencies obtained by choices of parameterizations. In the sequel of this paper, we present efficient algorithms that take advantage of the general framework and are derived from the axiomatization proposed in the present paper.

KEYWORDS
closure operator, lattice theory, completeness, data dependency, isotone Galois connection

1. Introduction

Rule-based systems and their applications are popular in many domains of computer science and data analysis. Among the most widely used rules are simple if-then rules that can be seen as implications between conjunctions of attributes. Such rules have been extensively studied and enjoyed many applications. In database systems, if-then rules of this form are called functional dependencies Maier (1983) and in relational database systems Codd (1983), they are interpreted in relations on relation schemes and serve basically as constraints that allow to design normalized databases. Rules of the same form with different interpretation emerged in several data analytical disciplines. For instance, in formal concept analysis Ganter and Wille (1997); Wille (1982), the rules are called attribute implications and are interpreted in object-attribute incidence data represented by formal contexts. Instead of playing the role of constraints, in formal concept analysis the rules are often considered as dependencies that are discovered from given object-attribute data Ganter (2010); Guigues and Duquenne (1986). If-then rules with a similar role emerged in data mining as association rules Agrawal et al. (1993); Zaki (2004). Indeed, the attribute implications used in formal concept analysis can be seen as particular association rules with confidence equal to 1 and disregarded support. From a broader perspective, if-then rules play central role in

Email: {pcordero,enciso}@uma.es, amora@ctima.uma.es, vilem.vychodil@gmail.com

logic programming languages, logic query systems Lloyd (1984), and many branches of artificial intelligence.

In most of the areas where if-then rules appear, the users rely either directly or indirectly on the notion of entailment of the rules. Indeed, many problems can be solved by deciding whether a given if-then rule logically follows form a set of other if-then rules. One of the early results Fagin (1977); Sagiv et al. (1981) on the entailment of if-then rules in relational databases observed that the entailment of functional dependencies coincides with the entailment of propositional formulas. In fact, functional dependencies can be seen as propositional formulas, namely implications between conjunctions of propositional variables, and their entailment in sense of the classic propositional logic coincides with the one considered in relational databases. Therefore, one might say that the logic of functional dependencies is a fragment of the propositional logic with a limited language. On the one hand, the if-then rules are not as expressive as general propositional formulas but on the other hand, several problems that are known to be hard in the full-fledged propositional logic become tractable for if-then rules. For instance, the entailment problem of if-then rules is decidable in linear time Beeri and Bernstein (1979) and the minimality problem of sets of if-then rules is decidable in polynomial time Maier (1980). Also, the limited language of if-then rules allows us to consider simple inference systems. Perhaps the best known inference system for if-then rules (and functional dependencies in particular) is the one by Armstrong Armstrong (1974) that can be summarized as follows:

$$\frac{}{A \cup B \Rightarrow B}, \qquad \frac{A \Rightarrow B}{A \cup C \Rightarrow B \cup C}, \qquad \frac{A \Rightarrow B,\, B \Rightarrow C}{A \Rightarrow C}, \tag{1}$$

where A, B, C are finite sets of attributes (think of attribute as of names). The rules in (1) are called the (generalized) reflexivity, augmentation, and transitivity, respectively, and it is well-known that they form an system that is sound and complete with respect to several possible interpretations of the rules, including their interpretations as functional dependencies and propositional formulas with a limited language. The inference system can be simplified, e.g., by combining the augmentation and transitivity into a single inference rule that is usually called pseudo-transitivity (or a cut):

$$\frac{A \Rightarrow B,\, B \cup C \Rightarrow D}{A \cup C \Rightarrow D}, \tag{2}$$

for any finite sets A, B, C, D of attributes. Observe that neither the transitivity nor the pseudo-transitivity allows to apply the inference rule to an arbitrary pair of if-then rules, e.g., in case of the transitivity, the premise (the part preceding $\Rightarrow$) of one of the formulas must be the same as the conclusion (the part after $\Rightarrow$) of the other. This might be seen as a practical obstacle in designing automated provers based directly on the inference system. The obstacle is overcome in other equivalent inference system, most notably the system of simplification logic Mora et al. (2012) which uses the following inference rules:

$$\frac{}{A \Rightarrow A}, \qquad \frac{A \Rightarrow B,\, C \Rightarrow D}{A \cup C \Rightarrow B \cup D}, \qquad \frac{A \Rightarrow B,\, C \Rightarrow D}{A \cup (C \setminus B) \Rightarrow D}, \tag{3}$$

where A, B, C are finite sets of attributes and $\setminus$ denotes the usual set difference. The rules in (3) are called the reflexivity, composition, and simplification, respectively. In contrast to the original Armstrong system, both the composition and simplification can be applied to any pair of if-then rules. The simplification logic proved to be useful since several efficient algorithms for sets of if-then rules can be derived directly from the inference system Benito-Picazo et al. (2018); Cordero et al. (2014, 2015). The simplification logic can be seen as a starting point of the present paper.

In many situations, the users need to switch from if-then rules with the classic-style interpretation to if-then rules in similar form but with a non-classic interpretation. For instance, Triska and Vychodil (2017) proposed temporal attribute implications, i.e., attribute implications between attributes annotated by time points. The goal of Triska and Vychodil (2017) and its sequel Triska and Vychodil (2018) was to formalize if-then dependencies between attributes whose presence/absence is subject to change in consecutive points in time. The if-then rules used in the paper can be represented by particular formulas of linear temporal logic. That is, instead of being a fragment of the classic logic, the logic of temporal if-then rules from Triska and Vychodil (2017) can be seen as a fragment of a particular temporal logic. In much the same way as in the classic case, there is a sound and complete system of inference rules that resembles the Armstrong system (1). Other examples of non-classic if-then rules that have an inference system which resembles the one by Armstrong include graded attribute implications Belohlavek and Vychodil (2016, 2017a), similarity-based functional dependencies Belohlavek and Vychodil (2017b, 2019), and earlier approaches to fuzzy functional dependencies Raju and Majumdar (1988), see also Ježková et al. (2017) for a survey. Recent result showing a variant of the Armstrong system is Naumov and Tao (2017).

The goal of this paper is to provide a general framework that covers various flavors of the Armstrong system that have been used in the literature. In order to obtain a reasonable level of generality, we use a general approach to parameterized semantics of if-then rules by systems of isotone Galois connections that originated in Vychodil (2016) and has been extended later in Vychodil (2017). In addition to that, we propose the framework as an abstract metatheory about general inference systems like (1) and (3) where instead of finite sets of attributes we consider compact elements of lattices and instead of the set operations $\cup$ and $\setminus$, we consider a pair of residuated operations Galatos et al. (2007) on lattices. Furthermore, we propose a general semantics of if-then rules that is based on particular closure operators. In this setting, we propose a logic for reasoninng with the rules, called the Parameterized Simplification Logic, and prove its soundness and completeness. In a follow-up paper, we propose a general algorithm for transformations of sets of if-then rules that can be instantiated into concrete algorithms depending on the choice of a parameterization. As a result, we obtain particular simplification logics and basic algorithms for all particular approaches to if-then rules that can be formalized in our framework.

This paper is organized as follows. In Section 2, we present a survey of utilized notions and notation. In Section 3, we introduce the syntax of the logic and its axiomatization. In Section 4, we present general semantics of the logic. In Section 5, we present the results on soundness and completeness of the logic and related observations. In Section 6, we deal with the general notion of completeness in data. Finally, in Section 7, we present conclusions and outline results elaborated in the second part Cordero et al. (2019) of the paper.

2. Preliminary Notions

Throughout this paper, we are going to utilize lattice-based structures that are closely related to structures frequently used in multiple-valued logics Galatos et al. (2007); Gottwald (2008); Hájek (1998). Namely, we are going to use complete dual residuated lattices. A complete dual residuated lattice is a structure $\mathbb{L} = \langle L, \leqslant, \oplus, \ominus, 0, 1\rangle$ satisfying the following conditions:

- $\langle L, \leqslant, 0, 1\rangle$ is a complete lattice where 0 is the least element and 1 is the greatest element. As usual, we use the symbols $\vee$ and $\wedge$ to denote suprema (least upper bounds) and infima (greatest lower bounds), respectively;
- $\langle L, \oplus, 0\rangle$ is a commutative monoid;
- $\ominus$ is a binary operation so that the pair $\langle \oplus, \ominus\rangle$ satisfies the following adjointness property: For all $a, b, c \in L$, we have

$$a \leqslant b \oplus c \text{ if and only if } a \ominus b \leqslant c. \tag{4}$$

The operations $\oplus$ and $\ominus$ shall be called the *addition* and *residuated subtraction*, respectively.

Remark 1. Residuated lattices were initially studied in Ward and Dilworth (1939) and later became popular as general structures of truth values in multiple-valued and fuzzy logics Cintula et al. (2011a,b); Esteva and Godo (2001); Goguen (1969); Hájek (1998), see also Belohlavek et al. (2017) for a historic overview. In fuzzy logics, elements of residuated lattices are considered as degrees of truth and operations of residuated lattices are taken as truth functions for general (fuzzy) logical connectives. Typically, one works with a residuated lattice satisfying the following form of adjointness:

$$a \otimes b \leqslant c \text{ iff } a \leqslant b \rightarrow c, \tag{5}$$

for all $a, b, c \in L$; $\otimes$ (so-called *multiplication*) is considered as truth function of "fuzzy conjunction" and $\rightarrow$ (so-called *residuum*) is considered as truth function of "fuzzy implication". In contrast, we use (4) with the intention to have a general way of expressing operations $\oplus$ and $\ominus$ generalizing set union and set difference, respectively.

There are many examples of dual residuated lattices which, in fact, can be easily obtained from the classic residuated lattices. For instance, $\mathbb{L}$ can be defined on a real unit interval with $\oplus$ being a continuous triangular co-norm Klement et al. (2000) with its adjoint $\ominus$. Other examples include structure defined of finite chains that are important from algorithmic points of view. The following example shows a particular example of a finite structure that will be used throughout the paper in running examples.

Example 2.1. Consider $L = \big\{0, 0.1, 0.2, \ldots 1\big\}$ with $\leqslant$ being the usual order and with the following binary operations:

$$a \oplus b = \begin{cases} a + b, & \text{if } a + b \leqslant 0.5, \\ \max\big\{0.5, a, b\big\}, & \text{otherwise,} \end{cases}$$

$$a \ominus b = \begin{cases} 0, & \text{if } a \leqslant b, \\ 1 - b, & \text{if } 0 \leqslant b < a \leqslant 0.5, \\ \max\{a, b\}, & \text{otherwise.} \end{cases}$$

Then, $\mathbb{L} = \langle L, \leqslant, \oplus, \ominus, 0, 1\rangle$ is a dual complete residuated lattice.

In the following proposition, we recall properties of dual residuated lattices that will be used in the paper.

Proposition 2.2. *For any $a, b, c \in L$, the following conditions hold:*

$$a \leqslant b \text{ if and only if } a \ominus b = 0, \tag{6}$$
$$a \ominus 0 = a, \tag{7}$$
$$a \ominus b \leqslant a \leqslant a \oplus b, \tag{8}$$
$$b \leqslant c \text{ implies } a \oplus b \leqslant a \oplus c,\ \ b \ominus a \leqslant c \ominus a \text{ and } a \ominus c \leqslant a \ominus b, \tag{9}$$
$$a \vee b \leqslant a \oplus (b \ominus a) \leqslant a \oplus b, \tag{10}$$
$$(a \oplus b) \ominus a \leqslant b. \tag{11}$$

Proof. All of (6)–(11) follow by the definition of dual residuated lattices. For illustration, we prove (10). Clearly, $0 = a \ominus a \leqslant b \ominus a$, i.e., using the adjointness, we have $a \leqslant a \oplus (b \ominus a)$. In addition, we get $b \leqslant a \oplus (b \ominus a)$ from $b \ominus a \leqslant b \ominus a$ using the adjointness. Thus, $a \vee b \leqslant a \oplus (b \ominus a)$. The remaining inequality in (10) follows from (9) using the fact that $b \ominus a \leqslant b$. □

The following observation shows equivalent conditions that are necessary and sufficient for the existence of a pair of operations satisfying the adjointness property.

Proposition 2.3. *Let $\langle L, \leqslant, 0, 1\rangle$ be a complete lattice and $\langle L, \oplus, 0\rangle$ be a commutative monoid. Then the following conditions are equivalent.*

- (*i*) *$\mathbb{L} = \langle L, \leqslant, \oplus, \ominus, 0, 1\rangle$ is a complete dual residuated lattice;*
- (*ii*) *$\ominus$ is isotone in the first argument, $\oplus$ is isotone in the second argument, and we have*

$$(a \oplus b) \ominus a \leqslant b \leqslant a \oplus (b \ominus a) \tag{12}$$

  for any $a, b \in L$;
- (*iii*) *We have*

$$a \oplus \textstyle\bigwedge_{i \in I} b_i = \bigwedge_{i \in I} (a \oplus b_i) \tag{13}$$

  for any $a \in L$ and any $b_i \in L$ ($i \in I$).

Proof. The fact that (*i*) implies (*ii*) follows from Proposition 2.2. Now, suppose that (*ii*) holds. By the isotony of $\oplus$, we get $a \oplus \bigwedge_{i \in I} b_i \leqslant a \oplus b_i$ for any $i \in I$ and thus $a \oplus \bigwedge_{i \in I} b_i \leqslant \bigwedge_{i \in I} (a \oplus b_i)$. Conversely, using the isotony of $\ominus$ in the first argument, $(a \oplus b_i) \ominus a \leqslant b_i$ yields $\bigwedge_{i\in} (a \oplus b_i) \ominus a \leqslant b_i$ for any $i \in I$ and so $\bigwedge_{i\in} (a \oplus b_i) \ominus a \leqslant \bigwedge_{i \in I} b_i$. Furthermore, applying the isotony of $\oplus$ and $\bigwedge_{i \in I} (a \oplus b_i) \leqslant a \oplus \big(\bigwedge_{i \in I} (a \oplus b_i) \ominus a\big)$, which is an instance of (12), we get $\bigwedge_{i \in I} (a \oplus b_i) \leqslant a \oplus \bigwedge_{i \in I} b_i$, showing (*iii*). Finally, assume that (*iii*) holds. In that case, we prove that $\oplus$ and $\ominus$ satisfy (4) and, in addition, $a \ominus b = \bigwedge\{x \in L \mid a \leqslant b \oplus x\}$. Suppose that $a \leqslant b \oplus c$. In that case, we have $c \geq \bigwedge\{x \in L \mid a \leqslant b \oplus x\} = a \ominus b$. Conversely, assume that $a \ominus b \leqslant c$ holds true. In that case, applying the isotony of $\oplus$, which is a consequence of (*iii*), we get $b \oplus c \geq b \oplus (a \ominus b) = b \oplus \bigwedge\{x \in L \mid a \leqslant b \oplus x\} = \bigwedge\{b \oplus x \mid a \leqslant b \oplus x\} \geq a$, proving that (*i*) holds. □

As we have outlined in Section 1, some of the elements in $\mathbb{L}$ will be considered as representations of premises and conclusions of the general if-then rules we are going to formalize. For this purpose, we restrict ourselves to compact elements: Following the usual terminology in lattice theory Birkhoff (1940), an element $k \in L$ is said to be *compact* if, for all $J \subseteq L$,

$$\text{if } k \leqslant \bigvee J, \text{there exists a finite } J' \subseteq J \text{ such that } k \leqslant \bigvee J'. \tag{14}$$

From now on, K denotes the set of all compact elements in $\mathbb{L}$ and we assume that $\mathbb{L}$ is *algebraic* (or compactly generated), i.e.,

$$\text{for all } a \in L \text{ there exists } X \subseteq K \text{ such that } a = \bigvee X. \tag{15}$$

Finally, we also assume that K is closed for $\oplus$ and $\ominus$:

$$a, b \in K \text{ implies } a \oplus b, a \ominus b \in K. \tag{16}$$

Example 2.4. Given the finite dual complete residuated lattice presented in Example 2.1, we consider the usual notion of fuzzy sets in the universe $\mathbb{N}$ of natural numbers. For instance, given two fuzzy sets $A, B \in L^{\mathbb{N}}$, we have that $A \subseteq B$ iff $A(n) \leqslant B(n)$ for all $n \in \mathbb{N}$, and the addition and the residuated subtraction are defined componentwise as follows: $(A \oplus B)(n) = A(n) \oplus B(n)$ and $(A \ominus B)(n) = A(n) \ominus B(n)$. Then, $\langle L^{\mathbb{N}}, \subseteq, \oplus, \ominus, \emptyset, \mathbb{N}, \rangle$ is a dual complete residuated lattice and its compact elements are those sets having a finite support, i.e., $A \in L^{\mathbb{N}}$ such that $\{n \in \mathbb{N} \mid A(n) > 0\}$ is finite. Using such structure, we use the usual succinct notation of writing fuzzy sets. For instance $A = \{1/_{0.3}, 3/_{0.6}\}$ denotes that $A(1) = 0.3$, $A(3) = 0.6$, and $A(n) = 0$ otherwise.

Notice that this dual complete residuated lattice is algebraic and, for all $A, B \in L^{\mathbb{N}}$, if A and B are compact then $A \oplus B$ and $A \ominus B$ are also compact.

In order to cover a large family of inference systems, we introduce parameterizations be means of systems of isotone Galois connections. Such systems are utilized in the inference system of our general logic and provide additional deduction rules. Also, in the semantics, the systems influence the interpretation of the formulas. The idea of this type of parameterization comes from Vychodil (2016) and Vychodil (2017).

Recall that a pair $\langle \boldsymbol{f}, \boldsymbol{g} \rangle$ of operators $\boldsymbol{f}\colon L \to L$ and $\boldsymbol{g}\colon L \to L$ is called an *isotone Galois connection* in $\mathbb{L}$ whenever

$$\boldsymbol{f}(a) \leqslant b \text{ iff } a \leqslant \boldsymbol{g}(b) \tag{17}$$

for all $a, b \in L$; $\boldsymbol{f}$ is called the *lower adjoint* of $\boldsymbol{g}$ and, dually, $\boldsymbol{g}$ is called the *upper adjoint* of $\boldsymbol{f}$. For isotone Galois connections $\langle \boldsymbol{f}_1, \boldsymbol{g}_1 \rangle$ and $\langle \boldsymbol{f}_2, \boldsymbol{g}_2 \rangle$ in $\mathbb{L}$, we define a *composition* $\langle \boldsymbol{f}_1, \boldsymbol{g}_1 \rangle \circ \langle \boldsymbol{f}_2, \boldsymbol{g}_2 \rangle$ by

$$\langle \boldsymbol{f}_1, \boldsymbol{g}_1 \rangle \circ \langle \boldsymbol{f}_2, \boldsymbol{g}_2 \rangle = \langle \boldsymbol{f}_1\boldsymbol{f}_2, \boldsymbol{g}_2\boldsymbol{g}_1 \rangle, \tag{18}$$

where $\boldsymbol{f}_1\boldsymbol{f}_2$ is a composed operator such that $\boldsymbol{f}_1\boldsymbol{f}_2(a) = \boldsymbol{f}_1(\boldsymbol{f}_2(a))$ for all $a \in L$ and analogously for $\boldsymbol{g}_2\boldsymbol{g}_1$. Obviously, $\langle \boldsymbol{f}_1, \boldsymbol{g}_1 \rangle \circ \langle \boldsymbol{f}_2, \boldsymbol{g}_2 \rangle$ is an isotone Galois connection in $\mathbb{L}$. We denote by $\mathtt{I}$ the identity operator in L, i.e., $\mathtt{I}(a) = a$ for all $a \in L$; $\langle \mathtt{I}, \mathtt{I} \rangle$ is an isotone Galois connection in $\mathbb{L}$. Furthermore, $\circ$ is associative and $\langle \mathtt{I}, \mathtt{I} \rangle$ is neutral with respect to $\circ$.

The following assertion lists some of the important properties of isotone Galois connections that will be used in our paper.

Proposition 2.5. *Let $\langle \boldsymbol{f}, \boldsymbol{g} \rangle$ be an isotone Galois connection in $\mathbb{L}$. Then, for any $a, b \in L$ and $a_i \in L$ $(i \in I)$, we have:*

$$a \leqslant \boldsymbol{g}(\boldsymbol{f}(a)), \tag{19}$$

$$\boldsymbol{f}(\boldsymbol{g}(b)) \leqslant b \tag{20}$$

$$a \leqslant b \text{ implies } \boldsymbol{f}(a) \leqslant \boldsymbol{f}(b), \tag{21}$$

$$a \leqslant b \text{ implies } \boldsymbol{g}(a) \leqslant \boldsymbol{g}(b), \tag{22}$$

$$\boldsymbol{f}\big(\bigvee\{a_i \mid i \in I\}\big) = \bigvee\big\{\boldsymbol{f}(a_i) \mid i \in I\big\}, \tag{23}$$

$$\boldsymbol{g}\big(\bigwedge\{a_i \mid i \in I\}\big) = \bigwedge\big\{\boldsymbol{g}(a_i) \mid i \in I\big\}. \tag{24}$$

Proof. The properties are well-known consequences of (17). □

Remark 2. Given a pair $\langle \otimes, \rightarrow \rangle$ of operations satisfying (5), we can easily induce an isotone Galois connection on $\mathbf{L}$. Indeed, for any $a \in L$, we may put $\boldsymbol{f}_a(x) = a \otimes x$ and $\boldsymbol{g}_a(x) = a \rightarrow x$ for all $x \in L$. It is straightforward to check that $\langle \boldsymbol{f}_a, \boldsymbol{g}_a \rangle$ is an isotone Galois connection in $\mathbb{L}$. This is an important observation—in practice it allows us to handle $\otimes$ and $\rightarrow$ appearing in the classic residuated lattices as special cases of systems of isotone Galois connections. General systems of that form, called parameterizations, are defined below.

Any set S of isotone Galois connections in $\mathbb{L}$ such that $\langle \mathtt{I}, \mathtt{I} \rangle \in S$ is called an *L-parameterization*. If S is an L-parameterization that is closed under compositions, i.e., if $\langle S, \circ, \langle \mathtt{I}, \mathtt{I} \rangle \rangle$ is a monoid, we call it an *$\mathbb{L}$-parameterization*, see (Vychodil 2016, Definition 1). If for any $a \in K$ (recall that K denotes the set of all compact elements) and any $\langle \boldsymbol{f}, \boldsymbol{g} \rangle \in S$, we have $\boldsymbol{f}(a) \in K$, then S is called *compact*, see (Vychodil 2017, Definition 4).

Example 2.6. Consider the dual commutative residuated lattice $L^{\mathbb{N}}$ introduced in Example 2.4 and, for each $a \in L$, consider two mappings $\boldsymbol{f}_a, \boldsymbol{g}_a \colon L^{\mathbb{N}} \rightarrow L^{\mathbb{N}}$ defined as follows:

$$(\boldsymbol{f}_a(A))(n) = \max\{0, A(n) - a\} \quad \text{and} \quad (\boldsymbol{g}_a(A)(n) = \min\{1, A(n) + a\},$$

for all $A \in L^{\mathbb{N}}$ and $n \in \mathbb{N}$. In particular, for all $A \in L^{\mathbb{N}}$, $\boldsymbol{f}_1(A) = \emptyset$, $\boldsymbol{g}_1(A) = \mathbb{N}$, and $\boldsymbol{f}_0(A) = \boldsymbol{g}_0(A) = A$.

It is easy to check that, for each $a \in L$, the pair $\langle \boldsymbol{f}_a, \boldsymbol{g}_a \rangle$ is an isotone Galois connection, and the set $S = \big\{\langle \boldsymbol{f}_a, \boldsymbol{g}_a \rangle \mid a \in \{0, 0.2, 0.4, \ldots 1\}\big\}$ is a compact $\mathbb{L}$-parameterization.

Let S be an L-parameterization. An operator $\boldsymbol{c} : L \rightarrow L$ is called an S-closure operator (Vychodil 2017, Definition 3) in $\mathbb{L}$ whenever

$$a \leqslant \boldsymbol{c}(a), \tag{25}$$

$$a \leqslant b \text{ implies } \boldsymbol{c}(a) \leqslant \boldsymbol{c}(b), \tag{26}$$

$$\boldsymbol{c}(\boldsymbol{g}(\boldsymbol{c}(a))) \leqslant \boldsymbol{g}(\boldsymbol{c}(a)), \tag{27}$$

are satisfied for all $a, b \in L$ and all $\langle \boldsymbol{f}, \boldsymbol{g} \rangle \in S$. In addition, if S is an $\mathbb{L}$-parameterization, then $\boldsymbol{c}$ is called an $\mathbb{S}$-closure operator.

Remark 3. It is easy to see that classic closure operators can be seen as $\mathbb{S}$-closure operators for $S = \{\langle \mathtt{I}, \mathtt{I} \rangle\}$. Indeed, in this particular case, (25) and (26) are the usual conditions of extensivity and monotony, and (27) becomes the classic condition of idempotency. Since for any L-parameterization S, we have $\langle \mathtt{I}, \mathtt{I} \rangle \in S$, it follows by the same argument that any S-closure operator is a closure operator.

Example 2.7. Let $L^{\mathbb{N}}$ be the dual complete residuated lattice introduced in Example 2.4 and S be the compact $\mathbb{L}$-parameterization defined in Example 2.6. Consider mappings $\boldsymbol{c}_1, \boldsymbol{c}_2 \colon L^{\mathbb{N}} \to L^{\mathbb{N}}$ defined as follows: for each $A \in L^{\mathbb{N}}$ and each $n \in \mathbb{N}$,

$$\boldsymbol{c}_1(A)(n) = \begin{cases} 0.6 & \text{if } A(n) \leqslant 0.6, \\ 0.8 & \text{if } 0.6 < A(n) \leqslant 0.8, \\ 1 & \text{otherwise,} \end{cases} \quad \boldsymbol{c}_2(A)(n) = \begin{cases} 0.5 & \text{if } A(n) \leqslant 0.5, \\ 0.7 & \text{if } 0.5 < A(n) \leqslant 0.7, \\ 1 & \text{otherwise.} \end{cases}$$

One can check that $\boldsymbol{c}_1$ is an $\mathbb{S}$-closure operator. In contrast, $\boldsymbol{c}_2$ is not an $\mathbb{S}$=closure operator since (27) does not hold. Indeed, one can take, e.g., $A = \{1/_{0.3}, 3/_{0.6}\}$ for which $\boldsymbol{c}_1(A) = \{1/_{0.5}, 3/_{0.7}\}$ and so

$$\boldsymbol{c}_2(\boldsymbol{g}_{0.2}(\boldsymbol{c}_2(A)))(3) = 1 \not\leqslant \boldsymbol{g}_{0.2}(\boldsymbol{c}_2(A))(3) = \boldsymbol{g}_{0.2}(0.7) = 0.9.$$

3. Parametrized Simplification Logic

In this section, we present the syntax of our logic and introduce an axiomatic system that will be used in further sections. We present basic properties of provability in our system and conclude the section by showing logical equivalences that are important from the point of view of the algorithms proposed in the sequel of the paper. In the next section, we focus on the semantics of the logic and prove that with respect to the desired semantics, the logic is sound and complete.

From now on, we assume that $\mathbb{L}$ is an algebraic dual commutative residuated lattice in which (16) holds and S is a compact L-parametrization. Furthermore, we let K denote the set of all compact elements of $\mathbb{L}$.

The set of well-formed formulas of the language is:

$$\mathcal{L} = \{a \Rightarrow b \mid a, b \in K\}$$

These well-formed formulas will be named *implications* and, in each implication, the first and the second component will be named *premise* and *conclusion*, respectively. Finally, the sets of implications $\Sigma \subseteq \mathcal{L}$ will be named *theories*.

Definition 3.1. For all $a, b, c, d \in K$ and $\langle \boldsymbol{f}, \boldsymbol{g} \rangle \in S$, the inference system consists of the following axiom scheme:

Reflexivity: Infer $a \Rightarrow a$; (`Ref`)

together with the following inference rules:

Composition: From $a \Rightarrow b$ and $a \Rightarrow c$ infer $a \Rightarrow b \oplus c$; (`Comp`)

Simplification: From $a \Rightarrow b$ and $c \Rightarrow d$ infer $a \oplus (c \ominus b) \Rightarrow d$; (`Simp`)

Extension: From $a \Rightarrow b$ infer $\boldsymbol{f}(a) \Rightarrow \boldsymbol{f}(b)$. (`Ext`)

Remark 4. Notice that (`Comp`)–(`Ext`) infer well-formed formulas from well-formed formulas. This follows from the fact that all premises and conclusions of the formulas are compact elements. Indeed, from (16), $a, b \in K$ implies $a \oplus b \in K$ and $a \ominus b \in K$; and, since S is compact, for all $a \in K$ and $\langle \boldsymbol{f}, \boldsymbol{g} \rangle \in S$, one has $\boldsymbol{f}(a) \in K$. In addition, it can be easily seen that (`Comp`) and (`Simp`) can be seen as generalizations of the rules in (3). In contrast, (`Ext`) can be seen as an additional rule (or a family or rules) that enrich the inference system.

The notion of syntactic derivation is introduced in the standard way.

Definition 3.2. An implication $a \Rightarrow b \in \mathcal{L}$ is said to be syntactically derived or inferred from a theory $\Sigma \subseteq \mathcal{L}$, denoted by $\Sigma \vdash a \Rightarrow b$, if there exists a sequence $\sigma_1, \ldots, \sigma_n \in \mathcal{L}$ such that σ_n is the implication $a \Rightarrow b$ and, for all $1 \leqslant i \leqslant n$, at least one of the following conditions holds:

- $\sigma_i \in \Sigma$; or
- σ_i is an axiom, i.e., it is obtained from (`Ref`); or
- σ_i is obtained by applying an inference rule (`Comp`), (`Simp`), or (`Ext`) to formulas in $\{\sigma_j \mid 1 \leqslant j < i\}$.

Example 3.3. Let $L^{\mathbb{N}}$ be the dual complete residuated lattice introduced in Example 2.4 and S be the compact $\mathbb{L}$-parameterization defined in Example 2.6. Consider the theory

$$\Sigma = \big\{\{1/_{0.3}, 3/_{0.6}\}{\Rightarrow}\{2/_{0.7}\},\ \{3/_{0.7}\}{\Rightarrow}\{1/_{0.6}\}\big\}.$$

The following sequence shows that $\Sigma \vdash \{3/_{0.6}\}{\Rightarrow}\{2/_{0.7}\}$ holds.

(i) $\{1/_{0.3}, 3/_{0.6}\}{\Rightarrow}\{2/_{0.7}\}$ .... by hypothesis.
(ii) $\{3/_{0.7}\}{\Rightarrow}\{1/_{0.6}\}$ .... by hypothesis.
(iii) $\{3/_{0.5}\}{\Rightarrow}\{1/_{0.4}\}$ .... by (ii) and (`Ext`) with $\boldsymbol{f}_{0.2}$.
(iv) $\{3/_{0.6}\}{\Rightarrow}\{2/_{0.7}\}$ .... by (iii), (i) and (`Simp`).

In the following proposition we present a summary of several important inference rules that can be derived from the primitive inference system from Definition 3.1. In order to keep the notation simple, for $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$, we often write $\sigma_1, \ldots, \sigma_n \vdash \varphi$ to denote that $\Sigma \vdash \varphi$.

Proposition 3.4. *The following rules are derived from the axiomatic system:*

Generalized Reflexivity: $\vdash a \Rightarrow b$ *when* $b \leqslant a$ (GRef)

Transitivity: $a \Rightarrow b, b \Rightarrow c \vdash a \Rightarrow c$ (Tran)

Generalization: $a \Rightarrow b \vdash c \Rightarrow d$ *when* $a \leqslant c$ *and* $d \leqslant b$ (Gen)

Generalized Composition: $a \Rightarrow b, c \Rightarrow d \vdash a \vee c \Rightarrow b \oplus d$ (GComp)

Augmentation: $a \Rightarrow b \vdash a \vee c \Rightarrow b \oplus c$ (Augm)

Generalized Transitivity: $a \Rightarrow b, b \vee c \Rightarrow d \vdash a \vee c \Rightarrow d$ (GTran)

Proof. Here, we provide a derivation chain for each of these derived rules of inference. Generalized Reflexivity (GRef): $\vdash a \Rightarrow b$ if $b \leqslant a$

(i) $a \Rightarrow a$ .... by (Ref).
(ii) $b \Rightarrow b$ .... by (Ref).
(iii) $a \Rightarrow b$ .... by i), ii) and (Simp).

In the last step, we have considered that, since $b \leqslant a$, the following equality holds $a \oplus (b \ominus a) = a \oplus 0 = a$

Transitivity (Tran): $a \Rightarrow b, b \Rightarrow c \vdash a \Rightarrow c$

(i) $a \Rightarrow b$ .... by hypothesis.
(ii) $b \Rightarrow c$ .... by hypothesis.
(iii) $a \Rightarrow c$ .... by i), ii) and (Simp).

In the last step, we have considered the following equality $a \oplus (b \ominus b) = a \oplus 0 = a$.

Generalization (Gen): $a \Rightarrow b \vdash c \Rightarrow d$ when $a \leqslant c$ and $d \leqslant b$

(i) $c \Rightarrow a$ .... by (GRef).
(ii) $a \Rightarrow b$ .... by hypothesis.
(iii) $c \Rightarrow b$ .... by (i), (ii) and (Tran).
(iv) $b \Rightarrow d$ .... by (GRef).
(v) $c \Rightarrow d$ .... by (iii), (iv) and (Tran).

Generalized Composition (GComp): $a \Rightarrow b, c \Rightarrow d \vdash a \vee c \Rightarrow b \oplus d$

(i) $a \Rightarrow b$ .... by hypothesis.
(ii) $a \vee c \Rightarrow b$ .... by (i) and (Gen).
(iii) $c \Rightarrow d$ .... by hypothesis.
(iv) $a \vee c \Rightarrow d$ .... by (iii) and (Gen).
(v) $a \vee c \Rightarrow b \oplus d$ .... by (ii), (iv) and (Comp).

Augmentation (Augm): $a \Rightarrow b \vdash a \vee c \Rightarrow b \oplus c$

(i) $a \Rightarrow b$ .... by hypothesis.
(ii) $c \Rightarrow c$ .... by (Ref).
(iii) $a \vee c \Rightarrow b \oplus c$ .... by (i), (ii) and (GComp).

Generalized Transitivity (GTran): $a \Rightarrow b, b \vee c \Rightarrow d \vdash a \vee c \Rightarrow d$

(i) $a \Rightarrow b$ .... by hypothesis.
(ii) $a \vee c \Rightarrow b \oplus c$ .... by (i) and (Augm).
(iii) $a \vee c \Rightarrow b \vee c$ .... by (ii) and (Gen).

(iv) $b \vee c \Rightarrow d$ .... by hypothesis.
(v) $a \vee c \Rightarrow d$ .... by (iii), (iv) and (`Tran`).

□

Remark 5. Let us note that one can introduce an inference system that is equivalent to the one of Simplification logic and generalizes the original Armstrong system (1). Indeed, by Proposition 3.4, it follows that (`GRef`), (`Augm`), and (`Tran`) are derived rules in Simplification logic. Conversely, it can be shown that (`Ref`), (`Comp`), and (`Simp`) can be derived from (`GRef`), (`Augm`), and (`Tran`). Trivially, (`Ref`) is a particular case of (`GRef`). Moreover, (`Comp`) can be derived as follows:

(i) $a \Rightarrow b$ .... by hypothesis.
(ii) $a \Rightarrow a \oplus b$ .... by (i) and (`Augm`).
(iii) $a \oplus b \Rightarrow a \vee b$ .... by (`GRef`).
(iv) $a \Rightarrow a \vee b$ .... by (ii), (iii), and (`Tran`).
(v) $a \Rightarrow c$ .... by hypothesis.
(vi) $a \vee b \Rightarrow b \oplus c$ .... by (v) and (`Augm`).
(vii) $a \Rightarrow b \oplus c$ .... by (iv), (vi), and (`Tran`).

Finally, (`Simp`) can be derived as follows:

(i) $c \Rightarrow d$ .... by hypothesis.
(ii) $b \oplus (c \ominus b) \Rightarrow c$ .... by (`GRef`).
(iii) $b \oplus (c \ominus b) \Rightarrow d$ .... by (ii), (i), and (`Tran`).
(iv) $a \Rightarrow b$ .... by hypothesis.
(v) $a \vee (c \ominus b) \Rightarrow b \oplus (c \ominus b)$ .... (iv) and (`Augm`).
(vi) $a \vee (c \ominus b) \Rightarrow d$ .... by (v), (iii), and (`Tran`).
(vii) $a \oplus (c \ominus b) \Rightarrow a \vee (c \ominus b)$ .... by (`GRef`).
(viii) $a \oplus (c \ominus b) \Rightarrow d$ .... by (vii), (vi), and (`Tran`).

As a consequence, the inference system of Simplification logic can be equivalently introduced using (`GRef`), (`Augm`), (`Tran`), and (`Ext`) as the basic inference rules, however, we would lose the benefit of having (`Simp`) as the fundamental inference rule whose hypotheses do not have any constraint on their form.

We also utilize a standard notion of a syntactic equivalence:

Definition 3.5. Two theories $\Sigma_1, \Sigma_2 \subseteq \mathcal{L}$ are said to be *equivalent*, denoted by $\Sigma_1 \equiv \Sigma_2$, whenever the following condition holds:

$$\Sigma_1 \vdash a \Rightarrow b \text{ iff } \Sigma_2 \vdash a \Rightarrow b, \text{ for all } a \Rightarrow b \in \mathcal{L}. \tag{28}$$

This condition is equivalent to the following one: $\Sigma_1 \vdash a \Rightarrow b$ for all $a \Rightarrow b \in \Sigma_2$ and $\Sigma_2 \vdash a \Rightarrow b$ for all $a \Rightarrow b \in \Sigma_1$.

One remarkable characteristic of Simplification logic, that justifies its name, is that its inference rules (Definition 3.1) induce a set of equivalences allowing the design of transformations to more precise specifications, with a lower level of redundancy. In the following proposition we present these equivalences.

Proposition 3.6. *For all $\langle \boldsymbol{f}, \boldsymbol{g} \rangle \in S$, the following equivalences hold:*

$$\textit{Decomposition}: \{a \Rightarrow b\} \equiv \{a \Rightarrow b \ominus a\} \qquad (\texttt{DeEq})$$

$$\textit{Composition}: \{a \Rightarrow b, a \Rightarrow c\} \equiv \{a \Rightarrow b \oplus c\} \qquad (\texttt{CoEq})$$

$$\textit{Simplification}: \text{if } \boldsymbol{f}(a) \leqslant c, \quad \{a \Rightarrow b, c \Rightarrow d\} \equiv \{a \Rightarrow b, c \ominus \boldsymbol{f}(b) \Rightarrow d \ominus \boldsymbol{f}(b)\} \qquad (\texttt{SiEq})$$

Proof. These equivalences, read from left to right are trivial from the (`GRef`), (`Comp`), (`Simp`) and (`Ext`) rules of inference. We have to prove the other side.
Decomposition Equivalence (`DeEq`):
We have to prove that $\{a \Rightarrow b \ominus a\} \vdash a \Rightarrow b \ominus a$

(i) $a \Rightarrow b \ominus a$ .... by Hypothesis.
(ii) $a \Rightarrow a$ .... by (`Ref`).
(iii) $a \Rightarrow a \oplus (b \ominus a)$ .... by (i), (ii) and (`Augm`).
(iv) $a \Rightarrow b$ .... by (iii) and (`Gen`).

In the last step, we have considered that the following inequality holds $b \leqslant a \oplus b \leqslant a \oplus (b \ominus a)$

Composition Equivalence (`CoEq`):
We have to prove that $\{a \Rightarrow b \oplus c\} \vdash a \Rightarrow b$ and $\{a \Rightarrow b \oplus c\} \vdash a \Rightarrow c$

(i) $a \Rightarrow b \oplus c$ .... by Hypothesis.
(ii) $a \Rightarrow b$ .... by (i) and (`Gen`).

The same schema is used to prove the second inference.

Simplification Equivalence (`SiEq`):
We have to prove that $\{a \Rightarrow b, c \ominus \boldsymbol{f}(b) \Rightarrow d \ominus \boldsymbol{f}(b)\} \vdash c \Rightarrow d$ when $\boldsymbol{f}(a) \leqslant c$

(i) $c \ominus \boldsymbol{f}(b) \Rightarrow d \ominus \boldsymbol{f}(b)$ .... by Hypothesis.
(ii) $a \Rightarrow b$ .... by Hypothesis.
(iii) $\boldsymbol{f}(a) \Rightarrow \boldsymbol{f}(b)$ .... by (`Ext`).
(iv) $\boldsymbol{f}(a) \vee (c \ominus \boldsymbol{f}(b)) \Rightarrow \boldsymbol{f}(b) \oplus (d \ominus \boldsymbol{f}(b))$ .... by (i), (iii) and (`GComp`).
(v) $\boldsymbol{f}(a) \oplus (c \ominus \boldsymbol{f}(b)) \Rightarrow \boldsymbol{f}(b) \oplus (d \ominus \boldsymbol{f}(b))$ .... by (iv) and (`Gen`).
(vi) $\boldsymbol{f}(b) \oplus (d \ominus \boldsymbol{f}(b)) \Rightarrow d$ .... by (`GRef`).
(vii) $\boldsymbol{f}(a) \oplus (c \ominus \boldsymbol{f}(b)) \Rightarrow d$ .... by (v), (vi) and (`Tran`).
(viii) $c \oplus (c \ominus \boldsymbol{f}(b)) \Rightarrow d$ .... by (9), (vii) and (`Gen`), since $\boldsymbol{f}(a) \leqslant c$.
(ix) $c \Rightarrow \boldsymbol{f}(a)$ .... by (`GRef`), since $\boldsymbol{f}(a) \leqslant c$.
(x) $c \Rightarrow \boldsymbol{f}(b)$ .... by (ix), (iii) and (`Tran`).
(xi) $c \Rightarrow c$ .... by (`Ref`).
(xii) $c \Rightarrow c \ominus \boldsymbol{f}(b)$ .... by (vii) and (`Gen`).
(xiii) $c \vee c \Rightarrow c \oplus (c \ominus \boldsymbol{f}(b))$ .... by (xi) , (xii) and (`Comp`).
(xiv) $c \Rightarrow c \oplus (c \ominus \boldsymbol{f}(b))$ .... by idempotenceness.
(xv) $c \Rightarrow d$ .... by (xiv), (viii) and (`Tran`).

□

Example 3.7. Let $\Sigma = \big\{\{1/_{0.3}, 3/_{0.6}\}\Rightarrow\{2/_{0.7}\},\ \{3/_{0.7}\}\Rightarrow\{1/_{0.6}\}\big\}$ be the theory introduced in Example 3.3. By (`SiEq`), it is equivalent to

$$\big\{\{3/_{0.6}\}\Rightarrow\{2/_{0.7}\},\ \{3/_{0.7}\}\Rightarrow\{1/_{0.6}\}\big\}.$$

As we mentioned above, the equivalence returns a new theory that is a simplification of the first one.

4. Semantics

In this section, we introduce the interpretation of formulas and define their semantic derivation. We provide the semantics of formulas on an abstract level using particular type of closure operators. Later, we show how the semantics of the classic if-then dependencies, including attribute implications and functional dependencies, fit in this framework. We start by introducing the notion of additivity of a closure operator.

Definition 4.1. A closure operator $\boldsymbol{c}$ in $\mathbb{L}$ is called additive whenever

$$a \oplus b \leqslant \boldsymbol{c}(a \vee b) \tag{29}$$

holds for all $a, b \in L$.

Clearly, if $\oplus$ coincides with $\vee$, then any closure operator in $\mathbb{L}$ is additive. In general, non-trivial additive closure operators exist as it is shown in the next example.

Example 4.2. The mapping $\boldsymbol{c}_1 \colon L^{\mathbb{N}} \to L^{\mathbb{N}}$ defined in Example 2.7 is an additive $\mathbb{S}$-closure operator.

Proposition 4.3. *Let $\boldsymbol{c}$ be a closure operator in $\mathbb{L}$. If $\boldsymbol{c}$ is additive, for all $a, b \in L$, one has $\boldsymbol{c}(\boldsymbol{c}(a) \oplus b) = \boldsymbol{c}(a \oplus b)$.*

Proof. Since $\boldsymbol{c}$ is isotone, one has $\boldsymbol{c}(a) \vee b \leqslant \boldsymbol{c}(a \oplus b)$. Now, since $\boldsymbol{c}$ is additive and idempotent, one has $\boldsymbol{c}(a) \oplus b \leqslant \boldsymbol{c}(\boldsymbol{c}(a) \vee b) \leqslant \boldsymbol{c}(\boldsymbol{c}(a \oplus b)) = \boldsymbol{c}(a \oplus b)$. Finally, $\boldsymbol{c}(\boldsymbol{c}(a) \oplus b) \leqslant \boldsymbol{c}(\boldsymbol{c}(a \oplus b)) = \boldsymbol{c}(a \oplus b)$. Conversely, $\boldsymbol{c}(a \oplus b) \leq \boldsymbol{c}(\boldsymbol{c}(a) \oplus b)$ because $\boldsymbol{c}$ is isotone and (9) holds. □

We now introduce the notions of a model and of a semantic derivation.

Definition 4.4. Let $a \Rightarrow b \in \mathcal{L}$. A mapping $\boldsymbol{c} \colon L \to L$ is said to be a model for $a \Rightarrow b$, written $\boldsymbol{c} \models a \Rightarrow b$, whenever $\boldsymbol{c}$ is an additive S-closure operator in $\mathbb{L}$ and $b \leqslant \boldsymbol{c}(a)$. Furthermore, the set of all models of $a \Rightarrow b$ is denoted by $\mathcal{M}od(a \Rightarrow b)$.

As usual, a mapping $\boldsymbol{c}$ is a model for a theory $\Sigma \subseteq \mathcal{L}$ if it is a model for all the implications $a \Rightarrow b \in \Sigma$. That is, we may put

$$Mod(\Sigma) = \textstyle\bigcap_{a \Rightarrow b \in \Sigma} \mathcal{M}od(a \Rightarrow b). \tag{30}$$

Example 4.5. The $\mathbb{S}$-closure operator $\boldsymbol{c}_1$ introduced in Example 2.7, which is additive (see Example 4.2), is a model for the following theory:

$$\Sigma = \big\{ \{2/_{0.9}\} \Rightarrow \{2/_{1}, 3/_{0.5}\}, \{3/_{0.9}, 4/_{0.7}\} \Rightarrow \{3/_{1}, 4/_{0.6}\} \big\}.$$

Similarly as the syntactic derivation, we introduce the notion of the derivation based on the semantics.

Definition 4.6. Let $a \Rightarrow b \in \mathcal{L}$ and $\Sigma \subseteq \mathcal{L}$. The implication $a \Rightarrow b$ is said to be semantically derived from the theory Σ whenever $\mathcal{M}od(\Sigma) \subseteq \mathcal{M}od(a \Rightarrow b)$. This fact is denoted by $\Sigma \models a \Rightarrow b$.

In the rest of this section, we outline how the general interpretation of our formulas related to the one of the classic attribute implictions and functional dependencies.

Example 4.7. Consider a finite non-empty set Y of elements called attributes (or features). An ordinary attribute implication is a formula $A \Rightarrow B$ where $A, B \subseteq Y$. In order to express attribute implications as formulas in our setting, it suffices to consider $\mathbb{L}$ as the Boolean algebra of all subsets of Y. That is, we consider $\mathbb{L} = \langle 2^Y, \subseteq, \cup, \setminus, \emptyset, Y \rangle$ where $\setminus$ is the usual set difference. Obviously, $\mathbb{L}$ is a dual residuated lattice. Hence, any $A \Rightarrow B$ where $A, B \subseteq Y$ is a well-formed formula.

The interpretation and semantic derivation for classic attribute implications is defined using finite formal contexts Ganter and Wille (1997) as models. Recall that a formal context is a structure $\langle X, Y, I \rangle$ where X is a finite non-empty set of objects, Y is the set of attributes used so far, and $I \subseteq X \times Y$ is an object-attribute incidence relation, $\langle x, y \rangle \in I$ is interpreted so that "object x has attribute/feature y". Each formal context $\langle X, Y, I \rangle$ induces a couple of concept-forming operators ${}^{\uparrow}\colon 2^X \to 2^Y$ and ${}^{\downarrow}\colon 2^Y \to 2^X$ defined by

$$A^{\uparrow} = \{y \in Y \mid \langle x, y \rangle \in I \text{ for all } x \in A\}, \tag{31}$$

$$B^{\downarrow} = \{x \in X \mid \langle x, y \rangle \in I \text{ for all } y \in B\}. \tag{32}$$

It is well-known that ${}^{\uparrow}$ and ${}^{\downarrow}$ form an antitone Galois connection and their composition ${}^{\downarrow\uparrow}$ is a closure operator in 2^Y. In this setting, an attribute implication $A \Rightarrow B$ holds in a context $\langle X, Y, I \rangle$ whenever $A \subseteq \{x\}^{\uparrow}$ implies $B \subseteq \{x\}^{\uparrow}$ for any object $x \in X$. In words, $A \Rightarrow B$ holds in $\langle X, Y, I \rangle$ if for any of the objects $x \in X$, the following condition holds: If x has all the attributes from A, then it has all the attributes from B. It is easily seen that $A \Rightarrow B$ holds in $\langle X, Y, I \rangle$ iff $B \subseteq A^{\downarrow\uparrow}$.

Now, assume that we have an $\mathbb{L}$-parameterization $S = \{\langle \mathtt{I}, \mathtt{I} \rangle\}$. Recall that by Definition 4.4, an S-closure operator $\boldsymbol{c}\colon 2^Y \to 2^Y$ is a model of $A \Rightarrow B$ whenever $B \subseteq \boldsymbol{c}(A)$. Hence, if $A \Rightarrow B$ holds in $\langle X, Y, I \rangle$, then $\langle X, Y, I \rangle$ induces a model in our sense, in particular, the closure operator $\boldsymbol{c}_{\langle X,Y,I \rangle}\colon 2^Y \to 2^Y$ such that $\boldsymbol{c}_{\langle X,Y,I \rangle}(C) = C^{\downarrow\uparrow}$ for all $C \subseteq Y$. That is, $\boldsymbol{c}_{\langle X,Y,I \rangle} \in \mathcal{M}od(A \Rightarrow B)$.

Conversely, each $\boldsymbol{c} \in \mathcal{M}od(A \Rightarrow B)$ can be seen as $\boldsymbol{c}_{\langle X,Y,I \rangle}\colon 2^Y \to 2^Y$ for some $\langle X, Y, I \rangle$. Indeed, given $\boldsymbol{c} \in \mathcal{M}od(A \Rightarrow B)$, we let X be the set of all fixed points of $\boldsymbol{c}$, i.e., $X = \{\boldsymbol{c}(F) \mid F \subseteq Y\}$. Furthermore, we introduce $I \subseteq X \times Y$ by putting $I = \{\langle F, y \rangle \in X \times Y \mid y \in F\}$. It can be easily seen that the fixed points of the composed operator ${}^{\downarrow\uparrow}$ coincide with those of $\boldsymbol{c}$ and so $\boldsymbol{c}$ is identical to $\boldsymbol{c}_{\langle X,Y,I \rangle}$. As a consequence, the semantic derivation in sense of attribute implications is the same as the semantic derivation in sense of our general logic in case of these particular $\mathbb{L}$ and S. Hence, the entailment of attribute implications can indeed be studied within our framework as one of its particular cases and all the general observations made in our paper hold in this particular case as well.

Example 4.8. In relational databases Maier (1983), a functional dependency is a formula of the form $A \Rightarrow B$, where A and B are subsets of attributes of a (finite) relation schema. For our purposes, a relation schema can be seen as a finite set R of attributes. Syntactically, functional dependencies are the same formulas as attribute implications from Example 4.7, however, they are interpreted in relations on relation schemes. In formalization of functional dependencies and their semantics in our logic, we can proceed in much the same way as in Example 4.7. In fact, using the result from Fagin (1977); Sagiv et al. (1981), the relationship comes almost immediately using our observations from Example 4.7.

Functional dependencies are typically interpreted in relation on relation schemes Maier (1983). Recall that a relation $\mathcal{R}$ on a relation scheme R is a finite subset of a direct product $\prod_{y \in R} D_y$, where each D_y is called a domain (a type) of the attribute $y \in R$. Each $r \in \mathcal{R}$ is called a tuple. A functional dependency holds in $\mathcal{R}$ whenever for any two tuples $r_1, r_2 \in \mathcal{R}$, the following condition is satisfied: If r_1 and r_2 are equal on all the attributes from A, written $r_1(A) = r_2(A)$, then r_1 and r_2 are equal on all the attributes from B, written $r_1(B) = r_2(B)$.

As in Example 4.7, each relation $\mathcal{R}$ on a scheme R induces a closure operator. Namely, for any $A \subseteq R$, we can define

$$\boldsymbol{e}_{\mathcal{R}}(A) = \{\langle r_1, r_2 \rangle \in \mathcal{R} \times \mathcal{R} \mid r_1(A) = r_2(A)\}, \tag{33}$$

$$\boldsymbol{c}_{\mathcal{R}}(A) = \{y \in R \mid \boldsymbol{e}_{\mathcal{R}}(A) \subseteq \boldsymbol{e}_{\mathcal{R}}(\{y\})\} \tag{34}$$

for any $A \subseteq R$. Note that $\boldsymbol{e}_{\mathcal{R}}(A)$ is an equivalence relation on $\mathcal{R}$ for any $A \subseteq R$ and $\boldsymbol{c}_{\mathcal{R}}$ is a closure operator in 2^R. It can be easily seen that a functional dependency $A \Rightarrow B$ holds in $\mathcal{R}$ iff $B \subseteq \boldsymbol{c}_{\mathcal{R}}(A)$. In addition, for $\mathbb{L}$ and S from Example 4.7, we obtain the desired correspondence between relations on relation schemes and models in our sense. Indeed, if $A \Rightarrow B$ holds in $\mathcal{R}$, then $\boldsymbol{c}_{\mathcal{R}} \in \mathcal{M}od(A \Rightarrow B)$. In addition, an arbitrary $\boldsymbol{c} \in \mathcal{M}od(A \Rightarrow B)$ coincides with $\boldsymbol{c}_{\mathcal{R}} \in \mathcal{M}od(A \Rightarrow B)$ for some $\mathcal{R}$. In particular, given $\boldsymbol{c}$, we can construct $\mathcal{R}$ that for each fixed point of $\boldsymbol{c}$ contains a pair of tuples that agree exactly on all the attributes from the fixed point of $\boldsymbol{c}$. As a consequence, the general remarks at the end of Example 4.7 apply in this case as well.

5. Soundness and completeness

In this section, we elaborate the soundness and completeness of our logic. We start by showing that the axiomatic system of the parameterized simplification logic proposed in Section 3 is sound with respect to the general semantics proposed in Section 4.

Theorem 5.1 (Soundness). *For any implication $a \Rightarrow b \in \mathcal{L}$ and any theory $\Sigma \subseteq \mathcal{L}$, it follows that $\Sigma \vdash a \Rightarrow b$ implies $\Sigma \models a \Rightarrow b$.*

Proof. The proof uses a standard analysis by cases combined with induction on the length of a derivation.

(i) For any model $\boldsymbol{c}$, it holds that $a \leqslant \boldsymbol{c}(a)$ and, therefore, $\Sigma \models a \Rightarrow a$.

(ii) Suppose that $\boldsymbol{c} \models a \Rightarrow b$ and $\boldsymbol{c} \models a \Rightarrow c$. That is, $b \leqslant \boldsymbol{c}(a)$ and $c \leqslant \boldsymbol{c}(a)$, then $b \vee c \leqslant \boldsymbol{c}(a)$ and $b \oplus c \leqslant \boldsymbol{c}(b \vee c) \leqslant \boldsymbol{c}(\boldsymbol{c}(a)) = \boldsymbol{c}(a)$.

(iii) Assume $\boldsymbol{c} \models a \Rightarrow b$ and $\boldsymbol{c} \models c \Rightarrow d$, i.e., $b \leqslant \boldsymbol{c}(a)$ and $d \leqslant \boldsymbol{c}(c)$.

From (12), $c \leqslant b \oplus (c \ominus b)$ and $d \leqslant \boldsymbol{c}(c) \leqslant \boldsymbol{c}(b \oplus (c \ominus b))$. Then, from $b \leqslant \boldsymbol{c}(a)$ and Proposition 4.3, one has

$$d \leqslant \boldsymbol{c}(b \oplus (c \ominus b)) \leqslant \boldsymbol{c}(\boldsymbol{c}(a) \oplus (c \ominus b)) = \boldsymbol{c}(a \oplus (c \ominus b)).$$

And, therefore, $\boldsymbol{c} \models a \oplus (c \ominus b) \Rightarrow d$.

(iv) If $\boldsymbol{c} \models a \Rightarrow b$, then $b \leqslant \boldsymbol{c}(a)$ and $\langle \boldsymbol{f}, \boldsymbol{g} \rangle \in S$.

Since f is monotone, $\boldsymbol{f}(b) \leqslant \boldsymbol{f}(\boldsymbol{c}(a))$, and using the properties of S-closure operators (Vychodil 2017, Theorem 17 (a)), we get $\boldsymbol{f}(\boldsymbol{c}(a)) \leqslant \boldsymbol{c}(\boldsymbol{f}(a))$ and thus $\boldsymbol{f}(b) \leqslant \boldsymbol{c}(\boldsymbol{f}(a))$, meaning that $\boldsymbol{c} \models \boldsymbol{f}(a) \Rightarrow \boldsymbol{f}(b)$

To conclude, the proof is finished by induction. ☐

Now, we deal with the completeness issue. In order to prove the completeness of the axiomatic system, we introduce several notions. First, we consider, in this framework, the definition of the syntactic closure of an element of L.

Definition 5.2 (Syntactic closure)**.** Given a theory $\Sigma \subseteq \mathcal{L}$, for each $a \in L$, the syntactic closure of a (w.r.t. Σ) is defined as $\boldsymbol{c}_\Sigma(a) = \bigvee \mathcal{C}_\Sigma(a)$ where

$$\mathcal{C}_\Sigma(a) = \{b \in K \mid \Sigma \vdash c \Rightarrow b \text{ for some } c \in K \text{ such that } c \leqslant a\}.$$

The following lemma leads to the result ensuring that the mapping $\boldsymbol{c}_\Sigma \colon L \to L$ is, in fact, a closure operator in $\mathbb{L}$.

Lemma 5.3. *For any $\Sigma \subseteq \mathcal{L}$, if $a \in K$ then $\boldsymbol{c}_\Sigma(a) = \bigvee\{b \in K \mid \Sigma \vdash a \Rightarrow b\}$.*

Proof. We prove that $\mathcal{C}_\Sigma(a) = \{b \in K \mid \Sigma \vdash a \Rightarrow b\}$ or, equivalently, that $\Sigma \vdash a \Rightarrow b$ if and only if there exists $c \in K$ such that $c \leqslant a$ and $\Sigma \vdash c \Rightarrow b$. One implication is trivial and the other one is consequence of (`GRef`) and (`Tran`). ☐

Theorem 5.4. *For any $\Sigma \subseteq \mathcal{L}$ and $a \Rightarrow b \in \mathcal{L}$, we have:*

$$\Sigma \vdash a \Rightarrow b \textit{ if and only if } b \leqslant \boldsymbol{c}_\Sigma(a)$$

Proof. The direct implication is a consequence of Lemma 5.3. Conversely, assume $b \leqslant \boldsymbol{c}_\Sigma(a)$ and consider $J = \{x \in K \mid \Sigma \vdash a \Rightarrow x\}$. From Lemma 5.3, one has $b \leqslant \bigvee J$ and, since $b \in K$, there exists a finite set $X \subseteq J$ such that $b \leqslant \bigvee X$. Notice that $\bigvee X \in K$ and, by (`GRef`), $\Sigma \vdash \bigvee X \Rightarrow b$. Moreover, for all $x \in X$, one has $\Sigma \vdash a \Rightarrow x$ and, by applying (`GComp`) and (`Gen`) a finite number of times, $\Sigma \vdash a \Rightarrow \bigvee X$. Finally, by applying (`Tran`) to $a \Rightarrow \bigvee X$ and $\bigvee X \Rightarrow b$, one has $\Sigma \vdash a \Rightarrow b$. ☐

The following assertion show an important property of $\boldsymbol{c}_\Sigma$ provided that S is an $\mathbb{L}$-parameterization. Recall that an L-parameterization S is called an $\mathbb{L}$-parameterization whenever S is a monoid.

Theorem 5.5. *If S is an $\mathbb{L}$-parameterization, then for any theory $\Sigma \subseteq \mathcal{L}$, the mapping $\boldsymbol{c}_\Sigma \colon L \to L$ is an additive $\mathbb{S}$-closure operator. Moreover, it is* algebraic*, i.e., for all $a \in L$,*

$$\boldsymbol{c}_\Sigma(a) = \bigvee\{\boldsymbol{c}_\Sigma(x) \mid x \in K \textit{ and } x \leqslant a\}.$$

Proof. First we prove that $\boldsymbol{c}_\Sigma$ is an $\mathbb{S}$-closure operator.

- Extensiveness: For all $a \in L$, since $\mathbb{L}$ is algebraic, there exists $B \subseteq K$ such that $a = \bigvee B$. For all $b \in B \subseteq K$, $b \in \mathcal{C}_\Sigma(a)$ because, by (`Ref`), $\Sigma \vdash b \Rightarrow b$ and $b \leqslant a$. Therefore, $B \subseteq \mathcal{C}_\Sigma(a)$ and

$$a = \bigvee B \leqslant \bigvee \mathcal{C}_\Sigma(a) = c_\Sigma(a).$$

- Isotonicity: Straightforwardly, $a_1 \leqslant a_2$ implies $\mathcal{C}_\Sigma(a_1) \subseteq \mathcal{C}_\Sigma(a_2)$ and, therefore, $\boldsymbol{c}_\Sigma(a_1) \leqslant \boldsymbol{c}_\Sigma(a_2)$.
- Consider $\langle \boldsymbol{f}, \boldsymbol{g} \rangle \in S$ and $a \in L$. To prove $\boldsymbol{c}_\Sigma(\boldsymbol{g}(\boldsymbol{c}_\Sigma(a))) \leqslant \boldsymbol{g}(\boldsymbol{c}_\Sigma(a))$ it is enough to see that $b \leqslant \boldsymbol{g}(\boldsymbol{c}_\Sigma(a))$ for all $b \leqslant \mathcal{C}_\Sigma(\boldsymbol{g}(\boldsymbol{c}_\Sigma(a)))$.

  If $b \in \mathcal{C}_\Sigma(\boldsymbol{g}(\boldsymbol{c}_\Sigma(a)))$, there exists $c \in K$ such that $\Sigma \vdash c \Rightarrow b$ and $c \leqslant \boldsymbol{g}(\boldsymbol{c}_\Sigma(a))$. Then, since $\langle \boldsymbol{f}, \boldsymbol{g} \rangle$ is a Galois connection, one has $\boldsymbol{f}(c) \leqslant \boldsymbol{c}_\Sigma(a) = \bigvee \mathcal{C}_\Sigma(a)$.

  Now, as $c \in K$, $\boldsymbol{f}(c) \in K$ and there exists $\{b_1, \ldots, b_n\} \subseteq \mathcal{C}_\Sigma(a)$ such that $\boldsymbol{f}(c) \leqslant \bigvee_{i=1}^n b_i$, and there also exists $\{c_1, \ldots, c_n\} \subseteq K$ with $c_i \leqslant a$ and $\Sigma \vdash c_i \Rightarrow b_i$ for all $1 \leqslant i \leqslant n$.

  Now, $\Sigma \vdash \bigvee_{i=1}^n c_i \Rightarrow \bigvee_{i=1}^n b_i$ is obtained by applying (`GComp`) and (`Gen`) n times. By the other side, from (`GRef`), one has $\Sigma \vdash \bigvee_{i=1}^n b_i \Rightarrow \boldsymbol{f}(c)$, and by (`Tran`), $\Sigma \vdash \bigvee_{i=1}^n c_i \Rightarrow \boldsymbol{f}(c)$.

  From $\Sigma \vdash c \Rightarrow b$ and (`Ext`), we obtain $\Sigma \vdash \boldsymbol{f}(c) \Rightarrow \boldsymbol{f}(b)$, and by (`Tran`), $\Sigma \vdash \bigvee_{i=1}^n c_i \Rightarrow \boldsymbol{f}(b)$.

  As $\bigvee_{i=1}^n c_i \leqslant a$ and $\bigvee_{i=1}^n c_i \in K$, we have that $\boldsymbol{f}(b) \in \mathcal{C}_\Sigma(a)$ and, therefore, $\boldsymbol{f}(b) \leqslant \boldsymbol{c}_\Sigma(a)$. Finally, this implies that $b \leqslant \boldsymbol{g}(\boldsymbol{c}_\Sigma(a))$.
- Finally, we prove $\boldsymbol{c}_\Sigma(a) = \bigvee\{\boldsymbol{c}_\Sigma(x) \mid x \in K, x \leqslant a\}$ for all $a \in L$.

  From isotonicity of $\boldsymbol{c}_\Sigma$, we have $\bigvee\{\boldsymbol{c}_\Sigma(x) \mid x \in K, x \leqslant a\} \leqslant \boldsymbol{c}_\Sigma(a)$. The other inequality is a consequence of the fact that $b \in \mathcal{C}_\Sigma(a)$ implies the existence of $c \in K$ such that $c \leqslant a$ and $\Sigma \vdash c \Rightarrow b$, and therefore $b \leqslant \boldsymbol{c}_\Sigma(c) \in \{\boldsymbol{c}_\Sigma(x) \mid x \in K, x \leqslant a\}$.

□

Theorem 5.6 (Completeness)**.** *If S is an $\mathbb{L}$-parametrization, then $\Sigma \models a \Rightarrow b$ imples $\Sigma \vdash a \Rightarrow b$ for any $\Sigma \subseteq \mathcal{L}$ and $a \Rightarrow b \in \mathcal{L}$.*

Proof. Assume $\Sigma \nvdash a \Rightarrow b$ and prove $\Sigma \not\models a \Rightarrow b$. First, $\boldsymbol{c}_\Sigma \in \mathcal{M}od(\Sigma)$ because $c \Rightarrow b \in \Sigma$ implies $d \in \mathcal{C}_\Sigma(c)$ and $d \leqslant \boldsymbol{c}_\Sigma(c)$. Second, from Theorem 5.4, we have that $b \not\leqslant \boldsymbol{c}_\Sigma(a)$ and, then, $\boldsymbol{c}_\Sigma \notin \mathcal{M}od(a \Rightarrow b)$. Finally, as Theorem 5.5 ensures, the mapping $\boldsymbol{c}_\Sigma$ is an additive S-closure operator. Therefore, $\mathcal{M}od(\Sigma) \not\subseteq \mathcal{M}od(a \Rightarrow b)$. □

Proposition 5.7. *For all $\Sigma_1, \Sigma_2 \subseteq \mathcal{L}$, we have that $\Sigma_1 \equiv \Sigma_2$ if and only if $\mathcal{M}od(\Sigma_1) = \mathcal{M}od(\Sigma_2)$.*

Proof. The assertion is a consequence of Theorem 5.1 and Theorem 5.6. □

6. Complete theories for additive S-closure operators

In this section, we focus on the issues of characterizing properties of theories that derive exactly all formulas whose model is a given S-closure operator. This issue is interesting from the data-analytical viewpoint. Indeed, in many logics of if-then rules, there exist characterizations of so-called complete sets of rules that entail exactly the same if-then dependencies that are valid in a given dataset. The results in this section may be viewed as a general counterpart to these observations made in our abstract framework.

Definition 6.1. Let $\boldsymbol{c}\colon L \to L$ be an additive S-closure operator. A theory $\Sigma \subseteq \mathcal{L}$ is said to be *complete* for $\boldsymbol{c}$ whenever, for all $a \Rightarrow b \in \mathcal{L}$,

$$\Sigma \vdash a \Rightarrow b \text{ if and only if } b \leqslant \boldsymbol{c}(a).$$

Obviously, the greatest theory (with respect to the set inclusion) that is complete for an additive S-closure operator $\boldsymbol{c}$ is

$$\Sigma_{\boldsymbol{c}} = \{a \Rightarrow b \in \mathcal{L} \mid b \leqslant \boldsymbol{c}(a)\}.$$

and, for any other complete theory Σ, we have that $\Sigma \equiv \Sigma_{\boldsymbol{c}}$.

Example 6.2. For the $\mathbb{S}$-closure operator $\boldsymbol{c}_1$ introduced in Example 2.7, which is additive (see Example 4.2), for all $A \Rightarrow B \in \mathcal{L}$, we have that $A \Rightarrow B \in \Sigma_{\boldsymbol{c}_1}$ if and only if, for all $n \in \mathbb{N}$, one of the following conditions holds:

(1) $A(n) \leqslant 0.6$ and $B(n) \leqslant 0.6$.
(2) $0.6 < A(n) \leqslant 0.8$ and $B(n) \leqslant 0.8$.
(3) $0.8 < A(n)$.

Notice that it is not the unique complete theory for $\boldsymbol{c}_1$. For instance, the following one is also complete for $\boldsymbol{c}_1$:

$$\Sigma = \big\{\{n/_{0}\} \Rightarrow \{n/_{0.6}\},\ \{n/_{0.7}\} \Rightarrow \{n/_{0.8}\},\ \{n/_{0.9}\} \Rightarrow \{n/_{1}\} \mid n \in \mathbb{N}\big\}.$$

Theorem 6.3. *If the L-parametrization S is a monoid and Σ is complete for $\boldsymbol{c}$, then $\boldsymbol{c}_\Sigma(a) = \boldsymbol{c}(a)$ for all $a \in K$.*

In addition, $\boldsymbol{c}_\Sigma$ is the greatest algebraic additive S-closure operator such that $\boldsymbol{c}_\Sigma \leqslant \boldsymbol{c}$.

Proof. First, since $\mathbb{L}$ is algebraic and Σ is complete for $\boldsymbol{c}$, we have that for all $a \in K$

$$\begin{aligned}\boldsymbol{c}(a) &= \bigvee\{x \in K \mid x \leqslant \boldsymbol{c}(a)\}\\ &= \bigvee\{b \in K \mid b \leqslant \boldsymbol{c}(x) \text{ for some } x \in K \text{ with } x \leqslant \boldsymbol{c}(a)\}\\ &= \bigvee\{b \in K \mid \Sigma \vdash x \Rightarrow b \text{ for some } x \in K \text{ with } x \leqslant \boldsymbol{c}(a)\} = \boldsymbol{c}_\Sigma(a).\end{aligned}$$

Second, from Theorem 5.5, we have that, for all $a \in L$,

$$\boldsymbol{c}_\Sigma(a) = \bigvee\{\boldsymbol{c}_\Sigma(x) \mid x \in K, x \leqslant a\} = \bigvee\{\boldsymbol{c}(x) \mid x \in K, x \leqslant a\} \leqslant \boldsymbol{c}(a).$$

Finally, assume that there exists an algebraic additive S-closure operator $\boldsymbol{c}'\colon L \to L$ such that $\boldsymbol{c}'(a) \leq \boldsymbol{c}(a)$ for all $a \in L$. Then

$$\boldsymbol{c}'(a) = \bigvee\{\boldsymbol{c}'(x) \mid x \in K, x \leqslant a\} \leqslant \bigvee\{\boldsymbol{c}(x) \mid x \in K, x \leqslant a\} = \boldsymbol{c}_\Sigma(a).$$

□

Corollary 6.4. *If $\boldsymbol{c}$ is an algebraic additive S-closure operator and Σ is complete for $\boldsymbol{c}$, then $\boldsymbol{c} = \boldsymbol{c}_\Sigma$.*

The following theorem ensures that the pair of mappings $\boldsymbol{c} \rightsquigarrow \Sigma_{\boldsymbol{c}}$ and $\Sigma \rightsquigarrow \boldsymbol{c}_\Sigma$ is a Galois connection between the set of additive S-closure operators with the induced relation $\leqslant$ (which is an order relation) and the set of theories with the preorder relation given by $\vdash$.

Theorem 6.5. *Let $\boldsymbol{c}$ be an additive S-closure operator and $\Sigma \subseteq \mathcal{L}$ be a theory. The* *following epuivalences hold:*

$$\boldsymbol{c}_\Sigma \leqslant \boldsymbol{c} \text{ if and only if } \Sigma \subseteq \Sigma_{\boldsymbol{c}} \text{ (or, equivalently, iff } \Sigma_{\boldsymbol{c}} \vdash \Sigma).$$

Proof. On the one hand, we prove that, if $\boldsymbol{c}_\Sigma(x) \leqslant c(x)$ for all $x \in L$, then $\Sigma \subseteq \Sigma_c$. For all $a \Rightarrow b \in \Sigma$, we have that $\Sigma \vdash a \Rightarrow b$ and, by Theorem 5.4, $b \leqslant \boldsymbol{c}_\Sigma(a) \leqslant \boldsymbol{c}(a)$. Therefore, $a \Rightarrow b \in \Sigma_{\boldsymbol{c}}$.

On the other hand, if $\Sigma \subseteq \Sigma_c$ then $\Sigma_{\boldsymbol{c}} \vdash \Sigma$ and, for all $a \in L$,

$$\begin{aligned}\boldsymbol{c}_\Sigma(a) &= \bigvee\{b \in K \mid \Sigma \vdash c \Rightarrow b, c \in K, c \leqslant a\}\\ &\leqslant \bigvee\{b \in K \mid \Sigma_{\boldsymbol{c}} \vdash c \Rightarrow b, c \in K, c \leqslant a\}\\ &= \bigvee\{b \in K \mid b \leqslant \boldsymbol{c}(c), c \in K, c \leqslant a\} \leq \boldsymbol{c}(a),\end{aligned}$$

finishing the proof. □

The following corollary follows by standard arguments as a consequence of the previous observations.

Corollary 6.6. *For any algebraic additive S-closure operator $\boldsymbol{c}$, one has $\boldsymbol{c} = \boldsymbol{c}_{\Sigma_{\boldsymbol{c}}}$. For any theory $\Sigma \subseteq \mathcal{L}$, one has $\Sigma \equiv \Sigma_{\boldsymbol{c}_\Sigma}$.* □

Let us conclude this section with a note of two different meaning of the notion of completeness we have used in this paper. In the first sense, we have introduced a logic with complete axiomatization in sense that all formulas that are provable from a given theory are exactly the formulas which are semantically entailed by the theory. This notion of completeness is important from the point of view of further development of algorithms that are based directly on the inference system: An algorithm that decide whether a formulas is provable from a system of other formulas is, owing to the completeness obtained in Theorem 5.6, an algorithm that decides whether a formula is a semantic consequence the second sense, we have investigated complete theories that have been defined as sets of all dependencies that hold for given additive S-closure operators. This second notion of completeness is important from the viewpoint of algorithms computing bases of dependencies from data. Examples of algorithms utilizing the notions of completeness can be found in the sequel Cordero et al. (2019) to the present paper.

7. Conclusion

In this first part of a series of two papers, we have introduced a simplification logic which is parameterized by systems of isotone Galois connections and its formulas are define using compact elements of dual residuated lattices. The semantics of the logic is defined using particular closure operators satisfying a condition of additivity. This general setting allows us to consider several existing logics of if-then rules, like the logics of attribute implications and functional dependencies, as special cases of the proposed logic. We have shown sound and complete axiomatization which is based on deduction rules of composition, simplification, and extension. In addition, we have provided a characterization of theories that are complete in data. In the second part Cordero et al. (2019), we present further details on the properties of the parameterized simplification logic, we study algorithms for transforming sets of formulas into other sets of formulas with distinctive properties, and show several instances of the algorithms depending on the choices of parameterizations and illustrate the impact on concrete logics of if-then rules covered by our approach.

Acknowledgment

Supported by Grant TIN2017-89023-P of the Science and Innovation Ministry of Spain, which is co-financed by the European Regional Development Fund (ERDF).

V. Vychodil was also supported the ECOP (Education for Competitiveness Operational Programme) project no. CZ.1.07/2.3.00/20.0059, which was co-financed by the European Social Fund and the state budget of the Czech Republic during 2011–2014.

References

Agrawal, R., Imieliński, T., Swami, A., Jun. 1993. Mining association rules between sets of items in large databases. SIGMOD Rec. 22 (2), 207–216.

Armstrong, W. W., 1974. Dependency structures of data base relationships. In: Rosenfeld, J. L., Freeman, H. (Eds.), Information Processing 74: Proceedings of IFIP Congress. North Holland, Amsterdam, pp. 580–583.

Beeri, C., Bernstein, P. A., March 1979. Computational problems related to the design of normal form relational schemas. ACM Trans. Database Syst. 4, 30–59.

Belohlavek, R., Dauben, J. W., Klir, G. J., 2017. Fuzzy Logic and Mathematics: A Historical Perspective. Oxford University Press, New York.

Belohlavek, R., Vychodil, V., 2016. Attribute dependencies for data with grades I. International Journal of General Systems 45 (7–8), 864–888.

Belohlavek, R., Vychodil, V., 2017a. Attribute dependencies for data with grades II. International Journal of General Systems 46 (1), 66–92.

Belohlavek, R., Vychodil, V., 2017b. Relational similarity-based model of data 1: Foundations and query systems. International Journal of General Systems 46 (7), 671–751.

Belohlavek, R., Vychodil, V., 2019. Relational similarity-based model of data 2: Dependencies in data. International Journal of General Systems 47 (1), 1–50.

Benito-Picazo, F., Cordero, P., Enciso, M., Angel, 2018. Minimal generators, an affordable approach by means of massive computation. The Journal of Supercomputing.

Birkhoff, G., 1940. Lattice theory, 1st Edition. American Mathematical Society, Providence.

Cintula, P., Hájek, P., Noguera, C. (Eds.), 2011a. Handbook of Mathematical Fuzzy Logic, Volume 1. Vol. 37 of Studies in Logic, Mathematical Logic and Foundations. College Publications.

Cintula, P., Hájek, P., Noguera, C. (Eds.), 2011b. Handbook of Mathematical Fuzzy Logic, Volume 2. Vol. 38 of Studies in Logic, Mathematical Logic and Foundations. College Publications.

Codd, E. F., 1983. A relational model of data for large shared data banks. Communications of the ACM 26, 64–69.

Cordero, P., Enciso, M., Mora, A., de Guzmán, I. P., 2014. A tableaux-like method to infer all minimal keys. Logic Journal of the IGPL 22 (6), 1019–1044.

Cordero, P., Enciso, M., Mora, A., Ojeda-Aciego, M., Rossi, C., 2015. Knowledge discovery in social networks by using a logic-based treatment of implications. Knowl.-Based Syst. 87, 16–25.

Cordero, P., Enciso, M., Mora, A., Vychodil, V., 2019. Parameterized simplification logic II: Algorithms. CoRR abs/1402.2071.

Esteva, F., Godo, L., 2001. Monoidal t-norm based logic: Towards a logic for left-continuous t-norms. Fuzzy Sets and Systems 124 (3), 271–288.

Fagin, R., Nov. 1977. Functional dependencies in a relational database and propositional logic. IBM Journal of Research and Development 21 (6), 534–544.

Galatos, N., Jipsen, P., Kowalski, T., Ono, H., 2007. Residuated Lattices: An Algebraic Glimpse at Substructural Logics, Volume 151, 1st Edition. Elsevier Science, San Diego, USA.

Ganter, B., 2010. Two basic algorithms in concept analysis. In: Proceedings of the 8th International Conference on Formal Concept Analysis. ICFCA'10. Springer-Verlag, Berlin, Heidelberg, pp. 312–340.

Ganter, B., Wille, R., 1997. Formal Concept Analysis: Mathematical Foundations, 1st Edition. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Goguen, J. A., 1969. The logic of inexact concepts. Synthese 19, 325–373.

Gottwald, S., 2008. Mathematical fuzzy logics. Bulletin of Symbolic Logic 14 (2), 210–239.

Guigues, J.-L., Duquenne, V., 1986. Familles minimales d'implications informatives resultant d'un tableau de données binaires. Math. Sci. Humaines 95, 5–18.

Hájek, P., 1998. Metamathematics of Fuzzy Logic. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Ježková, L., Cordero, P., Enciso, M., 2017. Fuzzy functional dependencies: A comparative survey. Fuzzy Sets and Systems 317, 88–120.

Klement, E. P., Mesiar, R., Pap, E., 2000. Triangular Norms, 1st Edition. Springer.

Lloyd, J. W., 1984. Foundations of Logic Programming. Springer-Verlag New York, Inc., New York, NY, USA.

Maier, D., Oct. 1980. Minimum covers in relational database model. J. ACM 27 (4), 664–674.

Maier, D., 1983. Theory of Relational Databases. Computer Science Pr, Rockville, MD, USA.

Mora, A., Cordero, P., Enciso, M., Fortes, I., Aguilera, G., 2012. Closure via functional dependence simplification. Int. J. Comput. Math. 89 (4), 510–526.

Naumov, P., Tao, J., 2017. Information flow under budget constraints. ACM Transactions on Computational Logic 18 (4).

Raju, K. V. S. V. N., Majumdar, A. K., 1988. Fuzzy functional dependencies and lossless join decomposition of fuzzy relational database systems. ACM Transactions on Database Systems 13 (2), 129–166.

Sagiv, Y., Delobel, C., Parker, Jr., D. S. t., Fagin, R., 1981. An equivalence between relational database dependencies and a fragment of propositional logic. J. ACM 28 (3), 435–453.

Triska, J., Vychodil, V., 2017. Logic of temporal attribute implications. Annals of Mathematics and Artificial Intelligence 79 (4), 307–335.

Triska, J., Vychodil, V., 2018. Minimal bases of temporal attribute implications. Annals of Mathematics and Artificial Intelligence 83 (1), 73–97.

Vychodil, V., 2016. Parameterizing the semantics of fuzzy attribute implications by systems of isotone Galois connections. IEEE Trans. on Fuzzy Systems 24, 645–660.

Vychodil, V., 2017. Closure structures parameterized by systems of isotone Galois connections. International Journal of Approximate Reasoning 91, 1–21.

Ward, M., Dilworth, R. P., 1939. Residuated lattices. Trans. Amer. Math. Soc. 45, 335–354.

Wille, R., 1982. Restructuring lattice theory: An approach based on hierarchies of concepts. In: Rival, I. (Ed.), Ordered Sets. Vol. 83 of NATO Advanced Study Institutes Series. Springer Netherlands, pp. 445–470.

Zaki, M. J., 2004. Mining non-redundant association rules. Data Mining and Knowledge Discovery 9, 223–248.