Baltic J. Modern Computing, Vol. 4 (2016), No. 4, pp. 789–810
DOI: http://dx.doi.org/10.22364/bjmc.2016.4.4.13

An Algebraic Approach to Functional-Dependency Semantics for First-Order Languages

Jānis CĪRULIS

Institute of Mathematics and Computer Science
University of Latvia
Raiņa b., 29, Riga LV-1459, Latvia

janis.cirulis@lu.lv

Abstract. Motivated by applications to relational databases, N. Alechina proposed in 2000 a many-sorted first-order logic with specialized models (called fd-models), where an ordinary first-order structure was equipped with a system of functions realizing functional dependences between variables (interpreted as attributes). Due to these constraints, only a restricted set of assignments is available in such a model. This factor may rouse some undesirable effects when formulas of the language are evaluated. We develop for the relevant first-order languages another, assignment-free algebraic semantics based on improved fd-models.

Keywords: first-order language, functional dependency, fd-frame, independence, semantics

1 Introduction

In 80-ies, various first-order logics were widely used for understanding relational databases and turned out, in particular, to be a useful tool of analysis of the concept of functional dependency between attributes in them. A natural mode of using a logical language to speak about a database is to interpret the formal variables of the language as attributes in its scheme. In this way functional dependences are introduced in the semantics of the language, and they may also been reflected into its syntax. Let us illustrate this point by two examples from the literature.

Probably, the first serious logical system dealing with dependencies between variables was the 'Independence friendly logic' (shortly, 'IF logic') proposed by Hintikka and Sandu (1989). A further development of IF-logic and its compositional semantics, introduced in Hodge (1997), is 'Dependence logic' of Väänänen (2007). See also Grädel and Väänänen (2012) and (2013).

The language of dependence logic is the ordinary first-order language expanded by new atomic formulas $=(t_1, t_2, \ldots, t_n)$ (called dependence formulas), where $t_1, \ldots, t_n$ are arbitrary terms; such a formula is informally read as 'the value of t_n is functionally dependent on the values of $t_1, \ldots, t_{n-1}$'. The intended semantics is the so called *team semantics*. A team for a first-order structure $\mathcal{A} = (A, \ldots)$ is a set of assignments ϕ for the variables of the language in A. By definition, a dependence formula $=(t_1, t_2, \ldots, t_n, t)$ is satisfied by a team Θ if, for all $\phi, \phi' \in \Theta$, $\phi(t) = \phi'(t)$ whenever $\phi(t_i) = \phi'(t_i)$ for $i = 1, 2, \ldots, n$. This truth condition resembles the well-known definition of a functional dependency of attributes in a database relation; see also Example 2.2 below. We leave out here truth conditions for compound formulas, and only note that formula counts as valid in a model $\mathcal{A}$ if it is satisfied in it by all teams.

Functional dependences between variables dealt with in the dependence logic are predetermined by the "content" of a concrete team and may vary when changing teams. Another, less popular kind of logic of dependence, also suggested by relational databases, was discussed in Alechina (2000), and deals with "built-in" functional dependences. It is based on a many-sorted first-order language without function symbols and dependence formulas of any kind, but not all assignments of values to variables are available in its models. In more detail, there is a set I of sorts (interpreted as attribute names), and each variable (attribute) v_i belongs to a different sort $i \in I$. A model of such a language $L(I)$ is a first order structure $\mathcal{A} := ((A_i)_{i \in I}, \ldots)$, where A_i are nonempty sets (domains of attributes), not necessarily disjoint or even different. An *fd-model* of $L(I)$ is then a pair $(\mathcal{A}, F)$, where F is a (possibly, empty) set of functions $f_{ij} \colon A_i \to A_j$ (*functional dependences*). The set Φ of assignments admissible in an fd-model is given by

$$\Phi := \{\phi \in \prod_{i \in I} A_i \colon \text{ for every } f_{ij} \in F,\ \phi(v_j) = f_{ij}(\phi(v_i))\}.$$

Imitating the traditional definitions, a formula is said to be true in the fd-model if it is satisfied by all assignments from Φ, and valid if it is true in all models. The logic given rise to by this semantics is called L_{fd}.

A logical consequence relation relatively to a given fd-model (not discussed by Alechina) also could be defined in the same way: a formula α logically implies β if every assignment from Φ that satisfies α satisfies also β. However, the restricted set of admissible assignments may turn out to be too small to get adequate characterizations of validity and logical consequence: as the following example shows, Φ may be even empty.

Example 1.1. Consider four attributes x_1, x_2, x_3, x_4 with domains

$$A_1 := \{a_{11}, a_{12}\},\ A_2 := \{a_{21}, a_{22}\},\ A_3 := \{a_{31}, a_{32}\},\ A_4 := \{a_{41}, a_{42}\}$$

and dependences

$$f_{31} \colon a_{31} \mapsto a_{11}, a_{32} \mapsto a_{12},\ f_{32} \colon a_{31} \mapsto a_{21}, a_{32} \mapsto a_{22},$$
$$f_{41} \colon a_{41} \mapsto a_{11}, a_{42} \mapsto a_{12},\ f_{42} \colon a_{41} \mapsto a_{22}, a_{42} \mapsto a_{21}.$$

If $\phi(x_3) = a_{31}$, then $\phi(x_1) = a_{11}$, $\phi(x_2) = a_{21}$, and there is no choice for $\phi(x_4)$. Likewise with $\phi(x_3) = a_{32}$. One may conclude that the given set of dependences forces at least some of the four attributes to conflict with each other. ◁

Therefore, with the described semantics, some semantic properties of formulas and relations between them may become inadequate; for example, it may happen that every formula logically implies any other formula (when $\Phi = \varnothing$).

A possible way to avoid such difficulties could be admitting partial assignments defined only on compatible subsets of variables. We choose another strategy, and develop in this paper an algebraic, assignment-free semantics for a certain first-order logical language. The primary object in this kind of semantics will be the so called *fd-frame* (this concept is borrowed from Cīrulis (2004)), which consists of a set of variables, a family of value sets for them, and a (closed in a sense) set of unary functional dependencies between variables; as we shall see in the next section, admitting dependences between single variables only is not a real restriction, but simplifies matters. A subset of variables is recognized as compatible in a frame if there is a variable they all depend on. We associate with an fd-frame F a certain set S elements of which are informally interpreted as statements about variables (and their values) in F, and turn it into an algebraic structure with operations interpreted as negation, conjunction, disjunction (the two latter being partial unless all variables are pairwise consistent) and existential and universal quantification. Moreover, we introduce on S a relation interpreted as entailment (consequence relation). Finally, we show by an example how the statement algebra can be used to interpret an appropriate first order language (similar to that of Alechina's L_{fd}): with any formula, a statement from S is associated as its meaning, but no formula with incompatible free variables becomes meaningful in a given frame. On this ground, the semantic notions of validity and logical consequence are then introduced in the semantics. However, we do not consider any formal system for this logic, and have not aimed to introduce statement algebras as a class of structures respect to which some predefined formal logical system should be sound or even complete.

It is worth to note already here that variables in an fd-frame are still rather formal: as shown by examples in the next section, they may, but not necessary have to, be thought of as attributes related with some database or an information system of any other kind. Several ideas developed in this paper go back to Cīrulis (1987), where they are realized in another form.

2 Functional-dependency frames

We begin with some preliminaries. A *preorder* on some set P is a reflexive and transitive relation on P. Upper and lower bounds for subsets of a preordered set are defined just as in ordered sets; however a least upper bound (l.u.b.) and a greatest lower bound (g.l.b.) of a subset P, if defined, may be not unique. P is said to be *bounded complete* if every subset bounded from above has a l.u.b. (equivalently, if every its non-empty subset has a g.l.b.), and *finitely bounded complete* if a l.u.b. exists (at least) for every finite bounded set. However, greatest lower bounds need not exist in a finitely bounded preordered set. Every (finitely) bounded preordered set P has least or *initial* elements—the l.u.b-s of the empty subset. P is finitely bounded complete if and only if it has initial elements and every two-element bounded subset has a l.u.b. If the preorder under consideration is antisymmetric, i.e., P is actually a poset, l.u.b.-s and g.l.b.-s are unique and are called *joins* and *meets*, respectively.

2.1 Functional dependences between variables

A *system of functional dependences* is determined by the following data:

- a nonempty set Var of *variables*,
- a family $Val := (Val_x)_{x \in Var}$ of nonempty sets of *values* for each variable,
- a family $F := (d_x^y\colon Val_y \to Val_x)_{(x,y) \in D}$ of functions (*dependences*), where D is some subset of Var^2.

The set D may be interpreted as a binary relation on Var. We shall usually write $x \leftarrow y$ for $(x, y) \in D$. Therefore,

$$x \leftarrow y \text{ iff } d_x^y \text{ exists in } F.$$

We call the relation $\leftarrow$ (i.e., D) a *dependency relation*, read '$x \leftarrow y$' as 'x *(functionally) depends on* y', and consider each function d_x^y as the respective dependence. Therefore, x here is a function of y: if $x \leftarrow y$ and y has a value v, then x has the value $d_x^y(v)$ (of course, there is also a feedback: if x has a value u, then y must have a value in $(d_x^y)^{-1}(u)$).

We assume that F satisfies the following conditions:

(F1) F is closed under composition: if d_y^z and d_x^y are in F, then $d_x^y d_y^z$ belongs to F,
(F2) the identity map id_{Val_x} on Val_x belongs to F for every x.

Then

$$d_x^y d_y^z = d_x^z, \quad \mathsf{id}_{Val_x} = d_x^x, \tag{1}$$

and the relation $\leftarrow$ is evidently a preorder. The described system (Var, Val, F) satisfying (F1) and (F2) will be called an *fd-structure* ('fd' for 'functional dependency'), and the preordered set Var, its *scheme*. If $\leftarrow$ is antisymmetric and, therefore, an order relation, it is natural to consider dependeces in F as inclusion dependences.

The family F is said to be *surjective* if the following condition is fulfilled:

(F3) every dependence d_x^y is surjective.

Dependences may be required to be surjective because x, in the case when it depends on y, is supposed to have only the possible values determined by y. Therefore, (F3) actually requires the value sets Val_x to be non-redundant.

Variables x and y in an fd-structure are *equivalent* (in symbols, $x \leftrightarrow y$) if $x \leftarrow y$ and $y \leftarrow x$. As it should be in this case, the dependences d_y^x and d_x^y are mutually inverse bijections. Indeed, for every possible value v of y, the corresponding value of x is $d_x^y(v)$, and this, in its turn, forces y to have the value $d_y^x(d_x^y(v))$, which therefore must equal to v. Thus $d_y^x d_x^y$ is the identity map on Val_y. Likewise, $d_x^y d_y^x$ is the identity map on Val_x; this proves that the above claim is valid. The system F may be called *reflexive* if it satisfies a condition which provides the converse:

(F4) if d_x^y is in F and is bijective, then its inverse belongs to F; therefore $(d_x^y)^{-1} = d_y^x$.

The following natural extension of condition (F4):

> (F5) if $x, y \leftarrow z$ and there is a function $d\colon Val_y \to Val_x$ such that $d^z_x(w) = d(d^z_y(w))$ for all $w \in Val_z$, then d belongs to F; therefore $d = d^y_x$,

will not be needed in this paper. Below, some other natural additional conditions on fd-structures will be discussed, but first we consider several examples of such structures.

2.2 Examples

The first two examples are related to information systems in the sense of Pawlak (1981) (more recent sources are, e.g., Khan and Banerjee (2009), Pancerz (2014)), known also as knowledge representation systems, attribute-value systems and information tables. The notion of information system essentially coincides with that of relation (with unordered columns labeled by attributes) in a relational database, of many-valued context (Granter and Wille (1999)) and of Chu space (e.g., Pratt (1994),(2005), also Wolski and Gomolińska (2013)).

Example 2.1. An information system is a quadruple (Ob, At, V, I), where
– Ob is a nonempty set of objects,
– At is a set of their attributes,
– V is a family $(V_a, a \in At)$ of sets; each V_a is considered as the set of possible values (domain) of a,
– I is a function $Ob \times At \to \bigcup(V_a\colon a \in At)$ such that $I(o, a) \in V_a$; it is called an information function.

Normally, attributes in an information system are thought of as formally independent in the sense that the information function may be quite arbitrary (unlike database relations, where some constraints may be built in its scheme). Accordingly, let F consist of all identity maps on the sets V_a; thus, the set At is trivially ordered: $a \leftarrow b$ in At iff $a = b$. Then the triple (At, V, F) is an fd-structure; of course, (F3), (F4) and (F5) also are trivially fulfilled. $\triangleleft$

A version of information systems with built-in dependencies (and, hence, with a non-trivially ordered scheme) have been discussed by the present author in Cīrulis (2002) and later in Cīrulis (2004).

Example 2.2. Real dependencies between attributes of an information system are supposed to be caused by the behavior of the information function on the object set as a whole. Namely, an attribute a is considered as dependent on a set B of attributes if the value of a for every object turns out to be uniquely determined by values of attributes in B for this object:

$$a \leftarrow B \;:\equiv\; \text{for all } o_1, o_2 \in Ob,\ f(o_1, a) = f(o_2, a) \text{ whenever } \\ f(o_1, b) = f(o_2, b) \text{ for every } b \in B.$$

Subsets of At may be treated as complex attributes. Let At^+ be the collection of all finite such attributes. For $A, B \in At^+$, put

$$A \leftarrow B :\equiv a \leftarrow B \text{ for all } a \in A.$$

Now, for every $o \in Ob$ let ϕ_o be a function on At such that $\phi_o(a) = I(o, a)$ for all a. For every $X \in At^+$, let V_X be the set of all restrictions $\{\phi_o|X \colon o \in Ob\}$. Then $A \leftarrow B$ if and only if there is a mapping $d_A^B \colon V_B \to V_A$ such that $\phi_o|A = d_A^B(\phi_o|B)$ for every $o \in Ob$. With F^+ the set of all such functions and $V^+ := (V_X \colon X \in At^+)$, the triple (At^+, V^+, F^+) is an fd-structure satisfying (F3), (F4) and, of course, (F5). Notice that the dependency relation on its scheme is governed by the so called Armstrong axioms (with $A \subseteq B$, d_A^B is a so called inclusion dependence). $\lhd$

A variant of such an fd-structure occurs when some *conflict* (or concurrence) relation—a symmetric and antireflexive relation $\sharp$—lives on the set of attributes and only conflict-free subsets of At are allowed to be included in At^+.

In the next example, the role of variables is played by entities which usually are not thought of as such.

Example 2.3. Suppose that I and O are the input set and, respectively, the output set of some automaton, possibly, nondeterministic. Let I^* be the set of all input words over I, and write $x \leq y$ to mean that an input word x is a prefix of an input word y. Let, furthermore, O_x^* be the set of all words over O of length equal to the length of x. For $x \leq y$, define a function $d_x^y \colon O_y^* \to O_x^*$ by the rule that $d_x^y(w)$ is the prefix of w in O_x^*; let F be the set of all such functions Then the triple $(I^*, (O_x^*)_{x \in I^*}, F)$ is an fd-structure. Its scheme I^* is even tree-ordered; the conditions (F3), (F4) and (F5) are obviously fulfilled.

Thus, input words are treated here as variables, and output words of an appropriate length, as their values. Of course, one may choose for variables some set X of more conventional entities (together with a fixed bijective correspondence between I^* and X), and redefine the above fd-structure accordingly. $\lhd$

Our last example of an fd-structure is more abstract. It is suggested by some ideas in quantum logic; see Section 4 in Cīrulis (2015) for another realization of them.

Recall that an orthoposet is a bounded poset $(P, \leq, 0, 1)$ equipped with an orthocomplementation—a unary operation $^\perp$ such that

$$\text{if } p \leq q, \text{ then } q^\perp \leq p^\perp, \quad p^{\perp\perp} = p, \quad p \wedge p^\perp = 0.$$

Any Boolean algebra is an example of such a poset. Elements p, q of P are said to be orthogonal (in symbols, $p \perp q$) if $p \leq q^\perp$, and a subset M of P is said to be orthogonal (or an orthosubset) if it does not contain 0 and its elements are mutually orthogonal.

Example 2.4. Given an orthoposet P, we may perceive it as (possibly, an approximation to) the set of inner events of some system, device or a similar object S, where $\leq$ serves as a part-of (or inclusion) relation for events, while the orthocomplement of an event p means the complementary event of p. A maximal orthosubset may then be thought of as an exhaustive collection of mutually exclusive events.

We consider any injective function defined on a maximal orthosubset M as an observable parameter of S and consider such parameters as variables. Let M_x stand for the domain of a variable x, and Val_x, for its range (which may be a proper subset of its codomain); then every element of M_x can be interpreted as an event that x has a concrete value. Conversely, any observed value of x signalizes that a certain event has occurred. If to every event q from M_y there is an event p in M_x such that $q \leq p$, then to any $v \in Val_y$ there is $u \in Val_x$ such that x has the value u whenever y has the value v (again, also conversely). In such a situation, we say that x functionally depends on y, and write $x \leftarrow y$; of course, then there is a function $d_x^y\colon Val_y \to Val_x$ such that always $u = d_x^y(v)$. Let Var be some set of variables on P, and let F be the corresponding set of dependence functions; then $(Var, (Val_x)_{x \in Var}, F)$ is an fd-structure, and (F3), (F4), (F5) also are satisfied. Notice that two variables are equivalent in this model if and only if they have a common domain in P. $\lhd$

Ending this subsection, we state a problem:

> Characterize those fd-structures that can, up to isomorphism, be realized on some orthoposet in the way described in the last example.

2.3 Frames

Suppose we are given some fd-structure (Var, Val, F). Informally, a subset X of Var may be regarded as compatible if values of its elements are “coexistent” or “available simultaneously“. For instance, this is certainly the case if there is a variable which all variables in X depend on. More formally, X is compatible whenever it has an upper bound in the preordered set Var.

What about the converse? A compatible subset itself may be thought of as a complex, or compound, variable. However, it is, in a sense, only a virtual one and should be somehow represented by an element of Var. Then we, for example, could implicitly take into account also dependency on several variables. A good candidate for an “actual” complex variable representing X in Var is a least upper bound of X.

Notice that a compatible subset of Var represented by its l.u.b. necessarily has an upper bound; this implies that the compatible subsets of variables may be identified with the bounded ones. Thus, we state the following condition:

(V0) the set of variables Var is bounded complete.

This condition is not an essential restriction on Var: it may be considered as a principle which says that we always can, if necessary, “define” or “construct” a new variable which serves as a l.u.b. of that or other compatible set of variables and add it to Var. However, we are forced to accept the fact that different compatible subsets may be represented by the same actual attribute (or equivalent attributes).

Next, it is a plausible intuitive idea that the value set of an actual complex variable has to be built up from the value sets of the component variables in a regular way. Let again X be a compatible subset of Var with a l.u.b. y. Then any element $v \in Val_y$ should be completely determined by its components $d_x^y(v)$ with $x \in X$. Put in precise terms, this condition reads as follows:

(V1) for all $v_1, v_2 \in Val_y$, if $d_x^y(v_1) = d_x^y(v_2)$ whenever $x \in X$, then $v_1 = v_2$.

In other words, elements of Val_y should be separable by the dependences d_x^y with $x \in X$. If the condition is satisfied, we say that the fd-structure *respects least upper bounds* (in Var). In particular, if X is empty (recall that then its l.u.b. is a least element of Var), then the *if* part of the above condition is trivially fulfilled, and Val_y must have only one element. Thus, the condition (V1) implies that a variable depends of all variables if and only if it is constant. Another consequence is that the variables (their values) belonging to any compatible subset correlate with those in another one having the same (least) upper bounds.

In fact, X itself can be provided with an appropriate set Val_X of "complex" values. For $x_1, x_2 \in X$, let us call values $u \in Val_{x_1}$ and $v \in Val_{x_2}$ *consistent* (in symbols, $u \sim_{x_1 x_2} v$, or just $u \sim v$ if the context is clear), if $d_z^{x_1}(u) = d_z^{x_2}(v)$ for every variable z that depends on both x_1 and x_2. We further say that an assignment ϕ for X (i.e., a function which assigns a value from Val_x to every variable $x \in X$) is a *consistent* if $\phi(x_1)$ and $\phi(x_2)$ are consistent for all $x_1, x_2 \in X$; in particular, then $\phi(x_1) = d_{x_1}^{x_2}(\phi(x_2))$ whenever $x_1 \leftarrow x_2$. A consistent assignment may be thought of as a record of "simultaneously possible" values of variables from X. Now, the subset $Val_X \subseteq \prod(Val_x \colon x \in X)$ of all consistent assignments may be considered as the domain of the complex attribute X. For example, if there is some $v \in Val_y$ such that $\phi_v(x) = d_x^y(v)$ whenever $x \in X$, then ϕ_v necessarily belongs to Val_X. On the other hand, every value of X should correspond in this sense to an appropriate value of y, which determines the former one. This can be put in precise terms as follows:

(V2) for every $\phi \in Val_X$, there is $v \in Val_y$ such that $\phi = \phi_v$.

We say that the family Val, or the fd-structure under consideration itself, is *saturated*, if this condition is fulfilled.

If both discussed conditions, (V1) and (V2), are fulfilled, then the family of mappings $(d_x^y \colon x \in X)$ embeds Val_y into the direct product of all domains Val_x with $x \in X$; in fact, maps Val_y even onto Val_X. We may consider also the "inverse" of this embedding—a mapping $\bowtie\colon Val_X \to Val_y$ defined by

for every $\phi \in Val_X$ and $v \in V_y$, $\bowtie(\phi) = v$ if and only if $\phi = \phi_v$.

With one exception in the next subsection, the full strength of the condition (V0) will not be needed in this paper: some finitary consequences of it are sufficient. Namely, the set Var is assumed to be finitely bounded complete, and every finite but nonempty its subset, to have a g.l.b.. These conditions can be equivalently reworded in terms of two binary operations as in the following definition, which goes back to Cīrulis (2004).

Definition 2.5. A (finitary) *fd-frame* F is an fd-structure where
– every pair of variables bounded above has a l.u.b.,
– every pair of variables has a g.l.b.,
– there are initial (i.e., least) variables,
– F respects least upper bounds and is saturated.

For convenience of notation, we also assume that (i) for any two variables x and y having least upper bounds one of these bounds is selected and denoted by $x \vee y$, (ii) for any x and y selected and denoted by $x \wedge y$ is one of their greatest lower bounds, (iii) one of the initial variables is selected and denoted by o. Therefore, after identifying equivalent elements, the preordered set of all variables becomes a nearsemilattice in the sense of Cīrulis (2004).

Now the definition of consistent values can be simplified: by (F1),

$$u \sim_{x,y} v \text{ iff } d^x_{x\wedge y}(u) = d^y_{x\wedge y}(v).$$

Let us turn back to the examples of the preceding subsection. The fd-structure of Example 2.1 is not a frame (for distinct variables do not have any g.l.b.), but it becomes a frame if one adds a new initial attribute (thus, making Atr what is called a flat domain) with an one-element domain and corresponding dependences. On the other hand, any attributes in a Pawlak-style information system are normally thought of as compatible; this requires the attribute set to be extended as in Example 2.2. The fd-structure of Example 2.2 is an fd-frame, and so is its variant mentioned just after the example. The fd-structure of Example 2.3 is a frame if the empty word is included in I^*. Observe that the frame has essentially incompatible sets of variables, and that this feature cannot be avoided in any natural way. As to the last example, we note without proof that, in the case when every maximal orthosubset of P is a domain of a variable, (i) the set of variables here proves to be bounded complete if the underlaying orthoposet P is orthocomplete, i.e., every orthosubset of P has the join, and (ii) then the fd-structure under question is an fd-frame.

2.4 On independency of variables

Let F be any frame satisfying (F3) and (F4). In terms of functional dependences, also independency of variables in F can be characterized. We begin with the following informal description of independency:

> y is independent from x if, at any given value of x, y can take every of its possible values.

In a more technical language, the independence relation $\perp$ is characterized as follows:

> $y \perp x :\equiv$ all values from Val_y are consistent with every value from Val_x.

Put another way, this means that $y \perp x$ iff every assignment to variables x and y is consistent.

Lemma 2.6. *If $x, y \leftarrow z$, then $y \perp x$ if and only if to every $u \in Val_x$ and $v \in Val_y$ there is $w \in Val_z$ such that $u = d^z_x(w)$ and $v = d^z_y(w)$.*

Proof. Assume that $x, y \leftarrow z$; then x and y have a l.u.b. Next, $y \perp x$ iff any value $v \in Val_y$ is consistent with any value in $u \in Val_x$. As the frame is saturated, this is the case iff $u = d^{x\vee y}_x(w_0)$ and $v = d^{x\vee y}_y(w_0)$ for an appropriate $w_0 \in Val_{x\vee y}$. As $d^z_{x\vee y}$ is surjective, there is $w \in Val_z$ such that $w_0 = d^z_{x\vee y}(w)$. Thus $u = d^z_x(w)$ and $v = d^z_y(w)$ by (F1). □

The following properties of $\perp$ are easily verified.

Lemma 2.7. *In Var,*

(a) *If $y \perp x$, then $x \perp y$,*
(b) *if $x \leftarrow y$ and $y \perp z$, then $x \perp z$,*
(c) *$0 \perp x$,*
(d) *$x \perp x$ iff $x \leftrightarrow o$,*
(e) *if $y \leftarrow z$, then $y \perp x$ iff $y \perp z \wedge x$.*

Proof. (a) Evident.

(b) Suppose that $x \leftarrow y$, $y \perp z$, and take $u \in Val_x$, $w \in Val_z$. Then $u = d^y_x v$ for some v by (F3); moreover, $d^y_{y\wedge z}(v) = d^z_{y\wedge z}(w)$ by (1). Using (F1) several times, now

$$d^x_{x\wedge z}(u) = d^x_{x\wedge z} d^y_x(v) = d^y_{x\wedge z}(v) = d^{y\wedge z}_{x\wedge z} d^y_{y\wedge z}(v) = d^{y\wedge z}_{x\wedge z} d^z_{y\wedge z}(w) = d^z_{x\wedge z}(w).$$

Thus, $x \perp z$.

(c) Evident.

(d) We know that always $o \leftarrow x$. If $x \perp x$, then the set Val_x is a singleton; hence, d^x_o is a bijection and, by (F4), $x \leftarrow o$; thus, $x \leftrightarrow o$. The converse follows from (c).

(e) By (b), $y \perp z \wedge x$ whenever $y \perp x$. Conversely, suppose that $y \leftarrow z$ and $y \perp z \wedge x$. If $v \in Val_y$ and $u \in Val_x$, then

$$d^y_{y\wedge x}(v) = d^y_{y\wedge z\wedge x}(v) = d^{z\wedge x}_{y\wedge z\wedge x}(d^x_{z\wedge x}(u)) = d^x_{y\wedge x}(u),$$

see (F1). Thus, $y \perp x$. □

We can prove more: the relation $\perp$ is also additive.

Lemma 2.8. *Suppose that $x, y \leftarrow z$ and that y is a l.u.b. of $Y := \{y_i : i \in I\}$. Then $y \perp x$ if and only if $y_i \perp x$ for all i.*

Proof. Necessity follows from item (b) of the previous lemma. Sufficiency: assume that both suppositions of the lemma are fulfilled and that $y_i \perp x$ for all i. Without restriction of generality, we may also assume that z is a l.u.b. of x and y. Choose any $u \in Val_x$ and $v \in Val_y$, and put $v_i := d^y_{y_i}(v)$ for all $i \in I$; then $v_i \sim v_j$ for all $i, j \in I$. Moreover, $v_i \sim u$ by the last assumption. Thus, a function ϕ on $\{x\} \cup Y$ defined by $\phi(x) = u$ and $\phi(y_i) = v_i$ for $i \in I$ is a consistent assignment, and there is an element $w \in Val_z$ such that $u = d^z_x(w)$ and, for all $i \in I$, $v_i = d^z_{y_i}(w)$—see (V2). By virtue of (F1), now $d^y_{y_i}(d^z_y(w)) = d^z_{y_i}(w) = v_i = d^y_{y_i}(v)$ for all i. As the frame respects least upper bounds, it follows that $v = d^z_y(w)$. Therefore,

$$d^y_{x\wedge y}(v) = d^y_{x\wedge y}(d^z_y(w)) = d^z_{x\wedge y}(w) = d^x_{x\wedge y}(d^z_x(w)) = d^x_{x\wedge y}(u),$$

i.e., $v \sim u$. Since both v and u are arbitrary, it follows that $y \perp x$. □

One more useful connection between $\perp$ and $\leftarrow$,

(F6) if, for all z, $z \perp x$ whenever $z \perp y$, then $x \leftarrow y$,

generally does not hold in F though its converse easily follows from Lemma 2.7(b). The independence relation can be characterized purely in terms of the relation $\leftarrow$ (i.e., not involving values of variables).

Theorem 2.9. *For all x, y, $y \perp x$ if and only if $x \wedge y \leftrightarrow o$.*

Proof. By (1), $y \perp x$ if and only if $d^y_{x\wedge y}(v) = d^x_{x\wedge y}(u)$ for all $u \in Val_x$ and $v \in Val_y$. Sufficiency of the condition now follows from the fact that Val_o is a singleton set (as F respects upper bounds; see the preceding subsection). Necessity: the dependences $d^x_{x\wedge y}$ and $d^y_{x\wedge y}$ are surjective; so independence of y from x implies that $Val_{x\wedge y}$ must be a singleton and, consequently, $x \wedge y$ is an initial variable (see (F4)). □

Let us return to examples of frames from subsection 2.2. In Example 2.2, complex attributes A and B are independent if and only if, for every $o_1, o_2 \in Ob$, there is $o \in Ob$ such that $\phi_o|A = \phi_{o_1}|A$ and $\phi_o|B = \phi_{o_2}|B$. In Example 2.3, two input sequences x and y are independent if and only if they have only the trivial prefix in common. In Example 2.4, it turns out that variables x and y are independent if and only if $\{1\}$ is the single orthosubset of which both M_x and M_y are refinements. (An orthosubset R of P is a refinement of an orthosubset Q if every element of Q is the join of a subset of R.)

We now introduce a new, independency-based operation on Var. Assume for a moment that the set of variables is bounded complete.

Consider the set $[x]_z := \{y \leftarrow z \colon y \perp x\}$. Being bounded by z, it has a l.u.b.; we denote one of these by $z - x$. By virtue of Lemma 2.8, $z - x$ belongs to $[x]_z$ and, hence, is greatest in this set:

$$z - x \leftarrow z, \quad z - x \perp x, \quad \text{if } y \leftarrow z \text{ and } y \perp x, \text{ then } y \leftarrow z - x. \tag{2}$$

Therefore, $z - x$ is a greatest in the sense of $\leftarrow$ variable that depends on z and is independent from x. Let us call the operation $-$ *subtraction*. This name is suggested by a set-theoretic interpretation of axioms (2): they are satisfied for x, y, z arbitrary sets if $\leftarrow$ stands for set inclusion, $\perp$, for their disjointness, and $-$, for set subtraction.

It is easily seen that $x - x \leftrightarrow o,\ z - o \leftrightarrow z,\ o - x \leftrightarrow o$. By Lemma 2.7(e),

$$z - x \leftrightarrow z - (x \wedge z).$$

Note also that subtraction is stable w.r.t. $\leftrightarrow$:

$$\text{if } z_1 \leftrightarrow z_2 \text{ and } x_1 \leftrightarrow x_2, \text{ then } z_1 - x_1 \leftrightarrow z_2 - x_2.$$

Some of the following properties of subtraction will be referred to in subsection 3.4.

Theorem 2.10. *For all x, y, z,*

(a) *if $y \leftarrow z$, then $y \perp x$ iff $y \leftarrow z - x$,*
(b) *$x \wedge (z - x) \leftrightarrow o$,*
(c) *if $x \leftarrow y$, then $z - y \leftarrow z - x$,*
(d) *$x \perp z$ iff $z - x \leftrightarrow z$.*

Proof. (a) By the definition of $z - x$ and, in the opposite direction, Lemma 2.7(b), as $z - x \perp x$.

(b) By (2) and Theorem 2.9.

(c) Suppose that $x \leftarrow y$. By (2), $z - y \leftarrow z$, $z - y \perp y$ and also $z - y \leftarrow z - x$, since $z - y \perp x$ by Lemma 2.7(b).

(d) By definition of subtraction, always $z - x \leftarrow z$. Further, $z \leftarrow z - x$ if and only if $z \perp x$: see (a). □

We saw that subtraction can be introduced on Var if this set is bounded complete. However, it is explicitly assumed in Definition 2.5 that Var is finitely bounded complete; consequently, existence of substraction should be considered as an additional condition on the scheme of F. So, we shall say that the scheme of an fd-frame is *subtractive* if subtraction (i.e., an operation $-$ satisfying (2)) on it exists.

3 Statement algebra of an fd-frame

Let $\mathsf{F} := (Var, (Val_x)_{x \in Var}, F)$ be any frame which satisfies (F3) and (F4).

3.1 Statements

We consider that the basic sentences about variables of an fd-frame that are of interest for logic are those of the form

the value of a variable x belongs to a subset U of Val_x.

Such a sentence may be codified as a pair (x, U). We therefore call a *statement* any pair (x, U) with $U \subseteq Val_x$, let S_x stand for the set $\{x\} \times B_x$ of statements with x fixed (here, B_x is the powerset of Val_x), and denote by S the set $\bigcup(\mathsf{S}_x \colon x \in Var)$ of all statements associated with a frame. Each set S_x inherits the structure of a Boolean algebra from B_x:

$$(x,U) \cup (x;V) := (x, U \cup V), \quad (x,U) \cap (x,V) := (x, U \cap V),$$
$$-(x,U) := (x, -U), \quad (x,U) \subseteq (x,V) :\equiv U \subseteq V,$$

where $-U$ stands for $Val_x \setminus U$, i.e., $-$ is here the complementation in the Boolean algebra B_x. Of course, the Boolean algebras S_x are interconnected: whenever $x \leftarrow y$, there is a natural pair of mappings

$$\varepsilon^x_y \colon B_x \to B_y, \;\; \pi^y_x \colon B_y \to B_x$$

defined by

$$\varepsilon^x_y U := \{v \in Val_y \colon d^y_x(v) \in U\},$$
$$\pi^y_x V := \{u \in Val_x \colon u = d^y_x(v) \text{ for some } v \in V\} = \{d^y_x(v) \colon v \in V\}.$$

The many-sorted algebra $B(\mathsf{F}) := (B_x, \varepsilon^x_y, \pi^y_x)_{x \leftarrow y \in Var}$ will be our starting point in describing the structure of S.

So, ε^x_y is the preimage map, and π^y_x is the image map of d^y_x:

$$\varepsilon^x_y U = (d^y_x)^{-1}(U), \quad \pi^y_x V = d^y_x(V).$$

It is well-known, and can be easily checked, that then they form a so called *adjunction* (known also as a (contravariant) Galois connection):

$$\text{if } x \leftarrow y, \text{ then } V \subseteq \varepsilon^x_y(U) \text{ if and only if } \pi^y_x(V) \subseteq U; \tag{3}$$

here, ε_y^x is said to be the right adjoint of π_x^y, and π_x^y, the left adjoint of ε_y^x. For further references, we give a list of basic properties of ε_y^x and π_x^y. Actually, only the properties (a)–(d), (g),(h),(m) and (p) in the subsequent proposition are independent: the remaining ones can be derived from them. On the other hand, the condition (3) is equivalent to the conjunction of (c) and (g)–(i).

Proposition 3.1. *The mappings ε_y^x and π_x^y have the following properties:*

(a) $\varepsilon_x^x = \mathsf{id}_{B_x}$,
(b) $\varepsilon_z^y \varepsilon_y^x = \varepsilon_z^x$,
(c) ε_y^x *is isotonic,*
(d) $-\varepsilon_y^x = \varepsilon_y^x -$,
(e) $\pi_x^x = \mathsf{id}_{B_x}$,
(f) $\pi_x^y \pi_y^z = \pi_x^z$,
(g) π_x^y *is isotonic,*
(h) $\mathsf{id}_{B_y} \subseteq \varepsilon_y^x \pi_x^y$,
(i) $\pi_x^y \varepsilon_y^x \subseteq \mathsf{id}_{B_x}$,
(j) ε_y^x *preserves* $\cap$,
(k) π_x^y *preserves* $\cup$,
(l) ε_y^x *is a Boolean homomorphism,*
(m) $\pi_x^y \varepsilon_y^x = \mathsf{id}_{B_x}$,
(n) ε_y^x *is injective,*
(o) π_x^y *is surjective,*
(p) *if* $x, y \leftarrow z$, *then* $\pi_y^z \varepsilon_z^x = \varepsilon_y^{x\wedge y} \pi_{x\wedge y}^x$,
(q) *if* $x \leftarrow y \leftarrow z$, *then* $\pi_x^z \varepsilon_z^y = \pi_x^y$ *and* $\pi_y^z \varepsilon_z^x = \varepsilon_y^x$.

Proof. We shall check here only the properties (j),(k),(m) and (p). Notice that (n) and (o) are consequences of (m), and (q) is a consequence of (p), (a) and (e). Also, (l) is a conjunction of (d) and (j).

(j) By (c), $\varepsilon_y^x(U_1 \cap U_2) \subseteq \varepsilon_y^x U_1 \cap \varepsilon_y^x U_2$. On the other hand, suppose that $v \in \varepsilon_y^x U_1 \cap \varepsilon_y^x U_2$. By the definition of ε_y^x, then $d_x^y(v) \in U_1 \cap U_2$ and, further, $v \in \varepsilon_y^x(U_1 \cap U_2)$.

(k) By (g), $\pi_x^y V_1 \cup \pi_x^y V_2 \subseteq \pi_x^y(V_1 \cup V_2)$. On the other hand, suppose that $u \in \pi_x^y(V_1 \cup V_2)$. Then $u = d_x^y(v)$ for some $v \in V_1 \cup V_2$ and, further, u belongs either to $\pi_x^y V_1$ or $\pi_x^y V_2$.

(m) Clearly, $\pi_x^y \varepsilon_y^x U \subseteq U$ by (i). On the other hand, let $u \in U$. Since d_x^y is surjective, $u = d_x^y(v)$ for some $v \in Val_y$. Then $v \in \varepsilon_y^x U$ and $u \in \pi_x^y \varepsilon_y^x U$.

(p) Let $x, y \leftarrow z$. Then by (h,c,g) and (b,f,m),

$$\pi_y^z \varepsilon_z^x U \subseteq \varepsilon_y^{x\wedge y} \pi_{x\wedge y}^y (\pi_y^z \varepsilon_z^x (\varepsilon_x^{x\wedge y} \pi_{x\wedge y}^x U)) = \varepsilon_y^{x\wedge y} \pi_{x\wedge y}^z \varepsilon_z^{x\wedge y} \pi_{x\wedge y}^x U = \varepsilon_y^{x\wedge y} \pi_{x\wedge y}^x U.$$

To prove the reverse inclusion, recall that the frame is saturated, and suppose that $v \in \varepsilon_y^{x\wedge y} \pi_{x\wedge y}^x U$. This means that $d_{x\wedge y}^y(v) = d_{x\wedge y}^x(u)$ for some $u \in U$. Then there is $w_0 \in Val_{x\vee y}$ such that $u = d_x^{x\vee y}(w_0)$ and $v = d_y^{x\vee y}(w_0)$. By (F3), $w_0 = d_z^{x\vee y}(w)$ for some $w \in Val_z$; hence, $u = d_x^z(w)$ and $v = d_y^z(w)$, see (F1). It follows that $w \in \varepsilon_z^x U$ and $v \in \pi_y^z \varepsilon_z^x U$. Therefore, $\varepsilon_y^{x\wedge y} \pi_{x\wedge y}^x U \subseteq \pi_y^z \varepsilon_z^x U$, as needed. □

Due to (j), the pair $(\varepsilon^x_y, \pi^y_x)$ becomes a kind of what is known as an embedding-projection pair: by (l), ε^x_y is a Boolean embedding, while (k) and (o) show that π^y_x can be considered as a projection.

3.2 Entailment

Our main goal in this subsection is to find an appropriate consequence relation, or entailment, $\preceq$ for statements.

The intuitive (and vague) idea behind the assertion ("metastatement" about two statements from S) '(x, U) entails (y, V)' is that it should be considered as a constraint

> "every time" when x takes a value in U, y necessary has a value in V.

For x and y compatible, the constraint can be formalized as follows:

> for every $\phi \in Val_{\{x,y\}}$, $\phi(y) \in V$ whenever $\phi(x) \in U$,

and amounts, due to (V2), (F3) and (F1), to the condition

$$\text{if } x, y \leftarrow z, \text{ then}$$
$$(x, U) \preceq (v, V) \text{ iff, for every } w \in Val_z,\ d^z_y(w) \in V \text{ whenever } d^z_x(w) \in U.$$

Rewritten in a more compact form, this condition reads as

$$\text{if } x, y \leftarrow z, \text{ then } (x, U) \preceq (v, V) \text{ iff } \varepsilon^x_z U \subseteq \varepsilon^y_z V. \tag{4}$$

In particular, if also $x \leftarrow y$, then

$$(x, U) \preceq (y, V) \text{ iff } \varepsilon^x_y(U) \subseteq V, \quad (y, V) \preceq (x, U) \text{ iff } V \subseteq \varepsilon^x_y(U). \tag{5}$$

However, we a looking for an entailment applicable to arbitrary statements. Notice that the defining condition $\varepsilon^x_z U \subseteq \varepsilon^y_z V$ in the prospective definition (4) is, by (3), equivalent to the condition $\pi^z_y \varepsilon^x_z U \subseteq V$ which, due to Proposition 3.1(p), is equivalent to the inclusion $\varepsilon^{x \wedge y}_x \pi^x_{x \wedge y} U \subseteq V$ not involving z anymore. Eventually, we assume the following general definition.

Definition 3.2. A statement (x, U) *entails* (y, V), in symbols, $(x, U) \preceq (y, V)$, if $\varepsilon^{x \wedge y}_y \pi^x_{x \wedge y} U \subseteq V$.

Two other forms of the defining condition are useful. Recall that values $u \in Val_x$ and $v \in Val_y$ are consistent (in symbols, $u \sim v$) if $d^x_{x \wedge y}(u) = d^y_{x \wedge y}(v)$ or, equivalently, $v \in (d^y_{x \wedge y})^{-1}(d^x_{x \wedge y}(u))$. Thus, if $x \leftarrow y$, then $u \sim v$ iff $u = d^y_x(v)$.

Lemma 3.3. *For all statements (x, U) and (y, V),*

(a) $(x, U) \preceq (y, V)$ *if and only if $U \subseteq \varepsilon^{x \wedge y}_x W$ and $\varepsilon^{x \wedge y}_y W \subseteq V$ for some $W \in B_{x \wedge y}$,*

(b) $(x, U) \preceq (y, V)$ *if and only every element of V_y that is consistent with an element of U belongs to V.*

Proof. (a) Suppose that the left-side condition is fulfilled. Put $W := \pi^x_{x\wedge y}U$; then $\pi^x_{x\wedge y}U \subseteq W$, i.e., $U \subseteq \varepsilon^{x\wedge y}_x W$ by (3). Further, $\varepsilon^{x\wedge y}_y(W) \subseteq V$ by the supposition. Conversely, suppose that $U \subseteq \varepsilon^{x\wedge y}_x W$ and $\varepsilon^{x\wedge y}_y W \subseteq V$. Then by Proposition 3.1(i,m,c) $\pi^x_{x\wedge y}U \subseteq W$ and $\varepsilon^{x\wedge y}_y \pi^x_{x\wedge y}U \subseteq \varepsilon^{x\wedge y}_y W \subseteq V$, i.e., $(x, U) \preceq (y, V)$, as needed.

(b) By the definition of consistency, the right-side condition says that

$$(\forall v \in Val_y)(\forall u \in U)\ (\text{if } v \in (d^y_{x\wedge y})^{-1} d^x_{x\wedge y}(u),\ \text{then } v \in V).$$

The left-hand assertion $(x, U) \preceq (y, V)$ can be rewritten in an expanded form as

$$(\forall u \in U)\ (d^y_{x\wedge y})^{-1}(d^x_{x\wedge y}(u)) \subseteq V.$$

Clearly, the two assertions are equivalent. □

We now easily obtain several natural properties of entailment.

Theorem 3.4. *In* S*,*

(a) $\preceq$ *is reflexive and transitive,*
(b) $(x, \varnothing) \preceq (y, V) \preceq (z, Val_z)$*,*
(c) $(x, U_1) \preceq (x, U_2)$ *if and only if* $U_1 \subseteq U_2$*,*
(d) $(x, U) \preceq --(x, U) \preceq (x, U)$*,*
(e) $(x, U) \preceq (y, V)$ *if and only if* $-(y, V) \preceq -(x, U)$*,*
(f) $(x, U) \preceq (y, V_1), (y, V_2)$ *if and only if* $(x, U) \preceq (y, V_1) \cap (y, V_2)$*,*
(g) $(x, U_1), (x, U_2) \preceq (y, V)$ *if and only* $(x, U_1) \cup (x, U_2) \subseteq (y, V)$*.*

Proof. (a) Clearly, $\preceq$ is reflexive. It is also transitive: if $U \preceq V$ and $V \preceq W$, then $\varepsilon^{x\wedge y}_y \pi^x_{x\wedge y}U \subseteq V$, and in virtue of Proposition 3.1(g,c),

$$\varepsilon^{y\wedge z}_z \pi^y_{y\wedge z}(\varepsilon^{x\wedge y}_y \pi^x_{x\wedge y}U) \subseteq \varepsilon^{y\wedge z}_z \pi^y_{y\wedge z}V \subseteq W.$$

On the other hand, by Proposition 3.1(h,c,g),(b,f) and (p),

$$\begin{aligned}\varepsilon^{x\wedge z}_z \pi^x_{x\wedge z}U &\subseteq \varepsilon^{x\wedge z}_z(\varepsilon^{x\wedge y\wedge z}_{x\wedge z}\pi^{x\wedge z}_{x\wedge y\wedge z}(\pi^x_{x\wedge z}U))\\ &= \varepsilon^{x\wedge y\wedge z}_z \pi^x_{x\wedge y\wedge z}U\\ &= \varepsilon^{y\wedge z}_z \varepsilon^{x\wedge y\wedge z}_{y\wedge z}\pi^{x\wedge y}_{x\wedge y\wedge z}\pi^x_{x\wedge y}U\\ &= \varepsilon^{y\wedge z}_z \pi^y_{y\wedge z}\varepsilon^{x\wedge y}_y \pi^x_{x\wedge y}U,\end{aligned}$$

whence, together with the previous inclusion, $\varepsilon^{x\wedge z}_z \pi^x_{x\wedge z}U \subseteq W$, i.e., $(x, U) \preceq (z, W)$.

Among the remaining properties, only (e) requires a comment. Assume that $(x, U) \preceq (y, V)$, and choose some $v \in -V$ and $u \in Val_x$. If $v \sim u$, i.e., $d^x_{x\wedge y}(u) = d^y_{x\wedge y}(v)$, then $u \notin U$, as otherwise $v \in V$ by the assumption (see Lemma 3.3(b)). Thus, $(y, -V) \preceq (x, -U)$ and $-(y, V) \preceq -(x, U)$. The converse now follows due to (d). □

At last, logical equivalence of statements is defined in terms of entailment in the usual way:

$$(x, U) \simeq (y, V)\ :\equiv\ (x, U) \preceq (y, V) \text{ and } (y, V) \preceq (x, U).$$

We do not consider this relation in more detail (but see Section 4).

3.3 Logical operations

Items (d) and (e) of the previous theorem show that we already have a reasonable negation-like operation $-$ on S, while items (f) and (g) show that the operations $\cap$ and $\cup$, if considered as partial operations on S, are rather restricted forms of conjunction, resp. disjunction in the preordered set of all statements. We are now going to demonstrate how they can be extended for an arbitrary pair of statements (x, U) and (y, V) with x and y compatible.

Assume that $x \vee y$ exists for some x and y, and let

$$\begin{aligned}(x, U) \curlywedge (y, V) &:= (x \vee y, \varepsilon^x_{x\vee y} U \cap \varepsilon^y_{x\vee y} V) = (x \vee y, \varepsilon^x_{x\vee y} U) \cap (x \vee y, \varepsilon^y_{x\vee y} V),\\ (x, U) \curlyvee (y, V) &:= (x \vee y, \varepsilon^x_{x\vee y} U \cup \varepsilon^y_{x\vee y} V) = (x \vee y, \varepsilon^x_{x\vee y} U) \cup (x \vee y, \varepsilon^y_{x\vee y} V).\end{aligned}$$

Both operations $\curlywedge$ and $\curlyvee$ are, evidently, idempotent, commutative and associative Therefore, these operations are partial semilattice operations extending $\cap$, resp., $\cup$; due to Proposition 3.1(d), they even prove to be connected by de Morgan laws. For instance,

$$\begin{aligned}-((x, U) \curlywedge (y, V)) &= -(x \vee y, \varepsilon^x_{x\vee y} U) \cup -(x \vee y, \varepsilon^y_{x\vee y} V)\\ = (x \vee y, \varepsilon^x_{x\vee y}(-U)) \cup (x \vee y, \varepsilon^y_{x\vee y}(-V)) &= -(x, U) \curlyvee -(y, V).\end{aligned}$$

However, $(\mathsf{S}, \curlywedge, \curlyvee)$ is not a partial lattice, because the operations induce different order relations (still related to $\preceq$; cf. (5)):

$$\begin{aligned}(x, U) \curlywedge (y, V) = (x, U) &\text{ iff } y \leftarrow x \text{ and } U \subseteq \varepsilon^y_x V,\\ (x, U) \curlyvee (y, V) = (y, V) &\text{ iff } x \leftarrow y \text{ and } \varepsilon^x_y U \subseteq V.\end{aligned}$$

Nevertheless, the subsequent theorem, which generalizes items (f) and (g) of Theorem 3.4, shows that the operations $\curlywedge$ and $\curlyvee$ can be used for characterizing some greatest lower bounds and least upper bounds in S w.r.t. the preorder $\preceq$.

Theorem 3.5. *In* $(\mathsf{S}, \preceq)$,

(a) *provided that y and z are compatible,*

$$(x, U) \preceq (y, V), (z, W) \textit{ iff } (x, U) \preceq (y, V) \curlywedge (z, W),$$

(b) *provided that x and y are compatible,*

$$(x, U), (y, V) \preceq (z, W) \textit{ iff } (x, U) \curlyvee (y, V) \preceq (z, W).$$

Proof. (a) Assume that y and z are compatible. To prove the ‘if’ part, it suffices to show that

$$(y, V) \curlywedge (z, W) \preceq (y, V), (z, W).$$

We check the first of the two entailments: by Proposition 3.1(a,g,i),

$$\varepsilon^{y\wedge(y\vee z)}_y \pi^{y\vee z}_{y\wedge(y\vee z)} (\varepsilon^y_{y\vee z} V \cap \varepsilon^z_{y\vee z} W) = \varepsilon^y_y \pi^{y\vee z}_y (\varepsilon^y_{y\vee z} V \cap \varepsilon^z_{y\vee z} W) \subseteq \pi^{y\vee z}_y \varepsilon^y_{y\vee z} V \subseteq V;$$

the other one is proved similarly.

To prove the ‘only if’ part, assume that $\varepsilon^{x\wedge y}_y \pi^x_{x\wedge y} U \subseteq V$ and $\varepsilon^{x\wedge z}_z \pi^x_{x\wedge z} U \subseteq W$. We have to show that

$$\varepsilon^{x\wedge(y\vee z)}_{y\vee z}\pi^{x}_{x\wedge(y\vee z)}U \subseteq \varepsilon^{y}_{y\vee z}V, \varepsilon^{z}_{y\vee z}W.$$

The first assumption implies that $\varepsilon^{x\wedge y}_{y\vee z}\pi^{x}_{x\wedge y}U = \varepsilon^{y}_{y\vee z}\varepsilon^{x\wedge y}_{y}\pi^{x}_{x\wedge y}U \subseteq \varepsilon^{z}_{y\vee z}V$ (Proposition 3.1(b,c)); therefore, to prove the first of needed inclusions, it suffices to note that

$$\varepsilon^{x\wedge(y\vee z)}_{y\vee z}\pi^{x}_{x\wedge(y\vee z)}U \subseteq \varepsilon^{x\wedge(y\vee z)}_{y\vee z}\varepsilon^{x\wedge y}_{x\wedge(y\vee z)}\pi^{x\wedge(y\vee z)}_{x\wedge y}\pi^{x}_{x\wedge(y\vee z)}U = \varepsilon^{x\wedge y}_{y\vee z}\pi^{x}_{x\wedge y}U.$$

by Proposition 3.1(h,c,g) and (b,f). The other one is proved similarly.

The proof of (b) is similar; however, some details are different. Assume that x and y are compatible.

To prove the 'if' part, it suffices to show that $(x,U),(y,V) \preceq (x,U) \curlyvee (y,V)$. Evidently, $\varepsilon^{x}_{x\vee y}U \subseteq \varepsilon^{x}_{x\vee y}U \cup \varepsilon^{y}_{x\vee y}V$, whence (see Proposition 3.1(e)) $(x,U) \preceq (x \vee y, \varepsilon^{x}_{x\vee y}U \cup \varepsilon^{y}_{x\vee y}V)$. The other inclusion is demonstrated similarly.

To prove the 'only if' part, suppose that $\varepsilon^{x\wedge z}_{z}\pi^{x}_{x\wedge z}U \subseteq W$ and $\varepsilon^{y\wedge z}_{z}\pi^{y}_{y\wedge z}V \subseteq W$. We have to show that

$$\varepsilon^{(x\vee y)\wedge z}_{z}\pi^{x\vee y}_{(x\vee y)\wedge z}\varepsilon^{x}_{x\vee y}U,\ \varepsilon^{(x\vee y)\wedge z}_{z}\pi^{x\vee y}_{(x\vee y)\wedge z}\varepsilon^{y}_{x\vee y}V \subseteq W.$$

But

$$\varepsilon^{(x\vee y)\wedge z}_{z}\pi^{x\vee y}_{(x\vee y)\wedge z}\varepsilon^{x}_{x\vee y}U = \varepsilon^{(x\vee y)\wedge z}_{z}\varepsilon^{x\wedge z}_{(x\vee y)\wedge z}\pi^{x}_{x\wedge z}U = \varepsilon^{x\wedge z}_{z}\pi^{x}_{x\wedge z}U \subseteq W$$

by Theorem 3.1(p,b) and the first assumption. The other inclusion is proved similarly.

□

We mention an alternative version of disjunction and conjunction on S:

$$(x,U) \wedge (y,V) := (x \wedge y, \pi^{x}_{x\wedge y}U \cap \pi^{y}_{x\wedge y}V) = (x \wedge y, \pi^{x}_{x\wedge y}U) \cap (x \wedge y, \pi^{y}_{x\wedge y}V),$$
$$(x,U) \vee (y,V) := (x \wedge y, \pi^{x}_{x\wedge y}U \cup \pi^{y}_{x\wedge y}V) = (x \wedge y, \pi^{x}_{x\wedge y}U) \cup (x \wedge y, \pi^{y}_{x\wedge y}V).$$

These are total operations; moreover, they both are idempotent commutative, and associative. However, none of them correlates well with entailment: the analogue of Theorem 3.5 for these operations does not hold true. This is the main reason why we do not make use of them.

3.4 Quantifiers

Thus, the operations $\curlywedge$, $\curlyvee$ and $-$ on S may be considered as logical operations with statements, namely, as conjunction, disjunction and negation, respectively. We still need some algebraic facilities for treating quantifiers over statements. An appropriate tool is provided by algebraic logic, where a unary operation Q on a Boolean algebra B is said to be an (existential) quantifier (also, cylindrification) if it satisfies three axioms

$$Q(0) = 0, \quad p \leq Q(p), \quad Q(p \wedge Q(q)) = Q(p) \wedge Q(q);$$

see, e.g., Halmos (1955), (1956). A quantifier is always a closure operator on B; moreover, its range is a subalgebra of B. These two conditions jointly are characteristic for quantifiers (Theorem 3 in Halmos (1956)), and we take them for a definition of Boolean quantifier.

Example 3.6. For $x \leftarrow y$, the function Q_x on B_y defined by $Q_x(V) := \varepsilon^x_y \pi^y_x V$ is a quantifier. Indeed, by Proposition 3.1(h), (c,g) and (m),

$$V \subseteq Q_x(V), \quad \text{if } V_1 \subseteq V_2 \in B_y, \text{ then } Q_x(V_1) \subseteq Q_x(V_2), \quad Q_x(Q_x(V)) = Q_x(V),$$

i.e., Q_x is a closure operator. By Proposition 3.1(l,k), Q_x preserves $\cup$, so that the range of Q_x is closed under unions. It is closed also under complementation: if $V \in Q_x(B_y)$, then $V = Q_x(V')$ for some $V' \in B_y$ and, using Proposition 3.1(d,m,d),

$$-V = -\varepsilon^x_y \pi^y_x V' = \varepsilon^x_y(-\pi^y_x V') = \varepsilon^x_y \pi^y_x \varepsilon^x_y(-\pi^y_x V') = \varepsilon^x_y \pi^y_x(-\varepsilon^x_y \pi^y_x V') = Q_x(-V),$$

i.e., $-V \in Q_x(B_y)$. Thus, the range of Q_x is a Boolean subalgebra of B_y, and Q_x itself is a quantifier. ◁

Therefore, the statement $(y, Q_x(V))$ may be regarded as resulting from (y, V) by existential quantification. The following observation allows us to replace it by an equivalent simpler one:

$$(y, Q_x(V)) \simeq (x, \pi^y_x(V)).$$

One direction here follows from Theorem 3.1(m), the other one is a tautology. (The idea that an existential quantifier is expressible by the left adjoint of the preimage map ε^x_y is known well; in an abstract form, it is common in categorical logic; see, e.g. Pitts (2000)).

To introduce the usual notation for quantifiers, it is necessary to make use of variable subtraction; therefore, in the rest of the section the underlaying frame F is supposed to be subtractive. We write $\exists_x(y, V)$ for $Q_{y-x}(y, V)$, i.e., assume the following definition of the quantifier $\exists_x$ in S:

$$\exists_x(y, V) := (y - x, \pi^y_{y-x} V).$$

Dually, universal quantifiers on a Boolean algebra is characterized as an interior operators whose range is a subalgebra. To give an example, we have to introduce one more family of mappings $\mu^y_x \colon B_y \to B_x$, where $x \leftarrow y$:

$$\mu^y_x(V) := \{u \in \mathit{Val}_x \colon (d^y_x)^{-1}(u) \subseteq V\}.$$

Each μ^y_x is the right adjoint of ε^x_y:

$$\varepsilon^x_y U \subseteq V \ \text{ iff } \ U \subseteq \mu^y_x V.$$

Now, the operation Q'_x on B_y defined by $Q'_x := \varepsilon^x_y \mu^y_x$ is an universal quantifier, and

$$\text{if } x \leftarrow y, \text{ then } (y, Q'_x(V)) \simeq (x, \mu^y_x V).$$

We omit proofs of these claims: they are similar to those related with the operation Q_x, and assume the definition

$$\forall_x(y, V) := (y - x, \mu^y_{y-x} V).$$

For illustration, we list a few natural and easily verified properties of existential quantifiers.

Theorem 3.7. *In* S*,*

(a) *if* $y \leftarrow z$*, then* $\exists_y(x, U) \preceq \exists_z(x, U)$*,*
(b) *if* $(x, U) \preceq (y, V)$ *and* $x \perp y$*, then* $\exists_x(x, U) \preceq (y, V)$*,*
(c) *if* $x \perp y$*, then* $\exists_x(y, V) \simeq (y, V)$*.*

Proof. (a) We have to prove that $\varepsilon_{x-z}^{(x-y)\wedge(x-z)}\pi_{(x-y)\wedge(x-z)}^{x-y}\pi_{x-y}^{x}U \subseteq \pi_{x-z}^{x}U$. But if $y \leftarrow z$, then $x - z \leftarrow x - y$ (Lemma 2.10(c)), and the inclusion reduces, by Proposition 3.1(a,f), to the tautology $\pi_{x-z}^{x}U \subseteq \pi_{x-z}^{x}U$.

(b) Let $(x, U) \preceq (y, V)$, i.e., $\varepsilon_y^{x\wedge y}\pi_{x\wedge y}^{x}U \subseteq V$. But if $x \perp y$, then $x \wedge y = o$ (Theorem 2.9) and $\varepsilon_y^{x\wedge y}\pi_{x\wedge y}^{x}U = \varepsilon_y^{o}\pi_o^{x}U$, whence $\exists_x(x, U) \preceq (y, V)$.

(c) Assume that $x \perp y$; then $y - x = y$ by Lemma 2.10(d). Hence, $(y, V) = (y - x, \pi_{y-x}^{y}V) = \exists_x(y, V)$. □

An inspection of this proof shows that the considered properties of existential quantifiers are connected with certain properties of independency and subtraction of variables. Other desirable properties of quantification my call for additional specific assumptions on the structure of Var and, consequently, of F. For example, if we want statements $\exists_{y-x}(y, V)$ and $(x, \pi_x^y V)$ to be logically equivalent, then the equivalence $y - (y - x) \leftrightarrow x$ should be fulfilled in Var. This latter equivalence proves to be a consequence of (F6), but, seemingly, cannot be derived from the definition (2) without any additional assumption on the relations $\leftarrow$ or $\perp$.

Properties of universal quantifiers on S will not be considered here in detail, for they can be expressed in terms of existential quantifiers in the usual way.

Proposition 3.8. *For all* $x, y \in Var$ *and* $V \in B_y$*,* $\forall_x(y, V) = -\exists_x(y, -V)$*.*

Proof. In view of Theorem 3.4(d), it suffices to show that $-\pi_x^y(V) = \mu_x^y(-V)$. For every $u \in Val_x$,

$$\begin{aligned} u \in \mu_x^y(-V) &\text{ iff } (d_x^y)^{-1}u \subseteq -V \\ &\text{ iff for all } v \in Val_y,\ \text{if } u = d_x^y v \text{ then } v \notin V \\ &\text{ iff for no } v \in V,\ d_x^y v = u \\ &\text{ iff } u \notin \pi_x^y V. \end{aligned}$$ □

This ends our construction of the statement algebra of an fd-frame.

4 Conclusion

We have noted in Introduction that some semantic anomalies may appear for certain logical languages taking account of functional dependency between variables, when the set of total assignments accessible in a model is restricted. In the paper, another, purely algebraic and assignment-free semantics for such languages is presented. The basic concept is that of fd-frame, which incorporates variables, their value sets and functional dependences between them. Associated with a frame is an algebraic structure S (algebra of statements), elements of which are interpreted as statements 'a variable x has a value in a subset U'; this algebra provides an entailment relation $\preceq$ on S (a kind of consequence relation), negation, partial conjunction and disjunction, and both quantifiers over statements. However, as noticed in subsection 3.4, properties of quantifiers depend of the structure of the set of functional dependences of the frame; this point requires further investigation.

The equivalence relation $\simeq$ corresponding to the preorder $\preceq$ may be regarded as logical equivalence of statements in a frame. It can be verified that $\simeq$ is even a congruence relation of S; we thus could build up the quotient algebra of S modulo $\simeq$. Recall analogical constructions in the traditional propositional and first-order calculi, where identifying logically equivalent logical formulas gives rise to the so called Lindenbaum-Tarski algebra of abstract propositions, which is, in the case of classical logics, always a Boolean algebra. We postpone to another paper this construction of Lindenbaum-Tarski algebra for S, and note here only, that it, in particular, turns out to be an orthoposet. It could be possible to approach in this way the representation problem stated just after Example 2.4.

We now describe a logical language for which fd-frames provide a semantics; so, it is a language appropriate to speak about this semantic framework. Let L be a first-order language with

- a denumerable set Var of variables,
- for each n-tuple $\mathbf{x} := (x_1, x_2, \cdots, x_n)$ of mutually distinct variables, a set $\mathsf{P}_{\mathbf{x}}$ (possibly, empty) of n-place predicate symbols,
- logical connectives $\neg$, $\wedge$, $\vee$ and quantifier symbols $\exists$, $\forall$.

Atomic formulas of L are those of the form $P\mathbf{x}$ with $P \in \mathsf{P}_{\mathbf{x}}$ (therefore, L may be treated, like Alechina's L_{fd}, as a many-sorted language with just one variable of each sort); other formulas of L are formed from these applying connectives and quantifiers in a usual way. Let Frm stand for the set of all formulas.

If Var is really the scheme of an fd-frame, then it may have incompatible subsets of variables, and then not every formula in Frm can be recognized as meaningful. We define recursively the set $MFrm$ of formulas meaningful relatively to the scheme, and the type $\tau(\alpha)$ of every meaningful formula α. Roughly, the type of a formula is the join of those variables which it depends on.

- if $\alpha = P(x_1, x_2, \cdots, x_n)$ and the subset $\{x_1, x_2, \cdots, x_n\}$ of Var is compatible, then $\alpha \in MFrm$ and $\tau(\alpha) := x_1 \vee x_2 \vee \cdots \vee x_n$,
- if $\alpha \in MFrm$, then $\neg\alpha \in MFrm$ and $\tau(\neg\alpha) := \tau\alpha$,
- if $\alpha, \beta \in MFrm$ and $\tau(\alpha)$ is compatible with $\tau(\beta)$, then $\tau(\alpha \wedge \beta) = \tau(\alpha \vee \beta) := \tau(\alpha) \vee \tau(\beta)$,
- if $\alpha \in MFrm$, then $(\exists x)\alpha, (\forall x)\alpha \in MFrm$ and $\tau((\exists x)\alpha) = \tau((\forall x)\alpha) =: \tau(\alpha) - x$.

Therefore, all formulas are meaningful in the case when every finite subset of variables is compatible. Now, an *fd-model* of L is a system A consisting of a subtractive frame F (with Var its set of variables) and a subset $|P|$ of $Val_{\tau(\alpha)}$ for every atomic formula $\alpha := P(x_1, x_2, \ldots, x_n)$ from $MFrm$.

We further recursively define the *valuation* V_{A} of formulas in a given fd-model A as a function $MFrm \to \mathsf{S}$:

– if $Px_1x_2\dots x_n \in MFrm$, then
  $\mathsf{V}_\mathsf{A}(Px_1x_2\dots x_n) := (x_1 \vee x_2 \vee \cdots \vee x_n, |Px_1x_2\dots x_n|)$,
– if $\alpha \in MFrm$, then $\mathsf{V}_\mathsf{A}(\neg\alpha) := -\mathsf{V}_\mathsf{A}(\alpha)$,
– if $\alpha, \beta \in MFrm$, then $\mathsf{V}_\mathsf{A}(\alpha \wedge \beta) := \mathsf{V}_\mathsf{A}(\alpha) \curlywedge \mathsf{V}_\mathsf{A}(\beta)$,
– if $\alpha, \beta \in MFrm$, then $\mathsf{V}_\mathsf{A}(\alpha \vee \beta) := \mathsf{V}_\mathsf{A}(\alpha) \curlyvee \mathsf{V}_\mathsf{A}(\beta)$,
– if $\alpha \in MFrm$, then $\mathsf{V}_\mathsf{A}((\exists x)\alpha) := \exists_x(\mathsf{V}_\mathsf{A}(\alpha))$,
– if $\alpha \in MFrm$, then $\mathsf{V}_\mathsf{A}((\forall x)\alpha) := \forall_x(\mathsf{V}_\mathsf{A}(\alpha))$.

Finally, we can introduce the following basic semantical notions:
– a formula α is *true* in a fd-model A if it is meaningful and $\mathsf{V}_\mathsf{A}(\alpha) = Val_{\tau(\alpha)}$,
– α *entails* β in A if both α and β are meaningful and $\mathsf{V}_\mathsf{A}(\alpha) \preceq \mathsf{V}_\mathsf{A}(\beta)$,
– a formula α is *valid* if it is true in every fd-model where it is meaningful,
– α *logically implies* β if α entails β in every fd-model where both α and β are meaningful.

This ends the description of semantics of L. The language together with this semantics give rise to a logic, which may be called a *logic of functional dependency*. Investigation of metalogical properties of this logic **L**, including development of an appropriate deductive system, and its comparison with Alechina's logic mentioned in Introduction is a natural task for further work.

Acknowledgments

This work was supported by Latvian Council of Science, Grant Nr. 271/2012. The author is grateful also to the anonymous referee for suggestion that help to improve the presentation.

References

Alechina, N. (2000). Functional dependencies between variables. *Studia Logica* **66**, 273–283.

Cirulis, J. (1987). Logic of indeterminancy, *Abstracts of Eighth Internat. Congress of Logic, Methodology and Philosophy of Science* (Moscow, Aug. 17–22, 1987) Vol. 5, Part 1, Moscow, 246–248.

Cīrulis, J. (2002) Knowledge representation in extended Pawlak's information systems: algebraic aspects. Eds.: Eiter, Th. et al., *Foundations of information and knowledge systems*, Proc. 2nd international symposium FoIKS 2002 (Salzau Castle, Germany, February 20–23, 2002). Berlin: Springer, Lect. Notes Comput. Sci. **2284**, 250–267.

Cīrulis, J. (2004) Knowledge representation systems and skew nearlattices. Eds.: Chajda, I. et al., *Contrib. Gen. Algebra* **14**, Proc. 64th workshop on general algebra AAA64 (Olomouc, Czech Republic, May 30–June 2, 2002) and of the 65th workshop on general algebra AAA65 (Potsdam, Germany, March 21–23, 2003). Klagenfurt: Verlag Johannes Heyn, 43–51.

Cīrulis, J. (2015). Dependency ordering of quantum observables. *Int. J. Theor. Phys.* **54**, 4247–4259.

Frink, O. (1963). Pseudo-complements in semi-lattices. *Duke Math. J.* **29**, 505–514.

Ganter, B., Wille, R. (1999). *Formal Concept Analysis. Mathematical foundations*. Springer, Berlin.

Grädel, E., Väänänen, J. (2012). Dependence, independence, and incomplete information. Ed.: A. Deutsch, *Database Theory, ICDT*, Proc. 15th Int. Conf. on Database Techn.(Berlin, Germany, March 26–29, 2012), New York: ACM, 1–7.

Grädel, E, Väänänen, J. (2013). Dependence and independence. *Studia Logica* **101**, 399–410.

Halmos, P.R. (1955). Algebraic logic, I. Monadic Boolean algebras. *Compositio Math.* **12**, 217-249. Reprinted in Halmos (1962), pp. 37–72.

Halmos, P.R. (1956). The basic concepts of algebraic logic. *Amer. Math. Monthly* **53**, 363–387. Reprinted in Halmos (1962), pp. 9–33.

Halmos, P.R. (1962). *Algebraic Logic*, Chelsea Publ. Co., New York.

Hintikka, J., Sandu, G. (1989). Informational independence as a semantical phenomenon. Eds.: Fenstadt, J.E. et al., *Logic, Methodology and Philosophy of Science VIII*, Proc. Eighth Internat. Congress of Logic, Methodology and Philosophy of Science (Moscow, Aug. 17–22, 1987), North-Holland, Amsterdam, 571–589.

Hodges, W. (1997). Compositional semantics for a language of imperfect information. Logic *Journal of IGPL* **5**, 539–563.

Khan, Md.A., Banerjee, M. (2009). A logic for complete information systems. Sossai, C. (ed.) et al., *Symbolic and quantitative approaches to reasoning with uncertainty*, Proc. 10th European conference, ECSQARU (Verona, Italy, July 1–3, 2009). Berlin: Springer. Lect. Notes Comput. Sci. **5590**, 829–840.

López, A.F., Barroso, M.I.T. (2001). *Pseudocomplemented semilattices, Boolean algebras, and compatible products*. J. Algebra **242**, 60–91.

Pancerz, K. (2014). Some Remarks on Complex Information Systems over Ontological Graphs. Gruca Al. (ed.) et al, *Man-Machine Interactions 3*. Berlin: Springer, Advances in Intelligent Systems and Computing, **242**, 377–384.

Pawlak, Z. (1981). Information systems—theoretical foundations. *Inform. systems* **6**, 205–218.

Pitts, A.M. (2000) Categorical logic. Abramsky S. (ed.) et al, *Handbook of Logic in Computer Science, vol. 5. Algebraic and Logical Structures*, Oxford University Press 2000 (Chapter 2).

Pratt, V.R. (1994). Chu Spaces: Automata with quantum aspects *PhysComp '94*, Proc. Workshop on Physics and Computation, Boston, Nov. 1994. IEEE, Dallas, 186–195.

Pratt, V.R. (2003). Chu spaces as a semantic bridge between linear logic and mathematics *Theor. Comput. Sci.* **294**, 439-471.

Väänänen, J. (2007). *Dependence Logic: A New Approach to Independence Friendly Logic*. London Mathematical Society Student Texts, **70**, Cambridge UP.

Wolski, M., Gomolińska, A. (2013). Concept formation: rough sets and Scott systems. *Fundamenta Informaticae*, **127**, 17–33.

Received October 10, 2016 , accepted October 24, 2016