Behavioral equivalence of hidden k-logics: an abstract algebraic approach[☆]

Sergey Babenyshev[1], Manuel A. Martins[2,*]

Abstract

This work advances a research agenda which has as its main aim the application of Abstract Algebraic Logic (AAL) methods and tools to the specification and verification of software systems. It uses a generalization of the notion of an abstract deductive system to handle multi-sorted deductive systems which differentiate visible and hidden sorts. Two main results of the paper are obtained by generalizing properties of the Leibniz congruence — the central notion in AAL.

In this paper we discuss a question we posed in [2] about the relationship between the behavioral equivalences of equivalent hidden logics. We also present a necessary and sufficient intrinsic condition for two hidden logics to be equivalent.

Keywords: Behavioral Equivalence, Hidden Logic, Leibniz Congruence.

1. Introduction

Behavioral abstraction plays an important role in modern specification theory by providing a more satisfactory way to prove correctness of a pro-

[☆]Accepted authors' manuscript published as: Sergey Babenyshev, Manuel A. Martins, *Behavioral equivalence of hidden k-logics: An abstract algebraic approach*, Journal of Applied Logic, Volume 16, pag. 72-91, 2016. [DOI:10.1016/j.jal.2016.03.002]. The final publication is available at Elsevier via http://www.sciencedirect.com/science/article/pii/S157086831630009X .

[*]Corresponding author

Email address: martins@ua.pt (Manuel A. Martins)

[1]Siberian Fire-Rescue Academy, Severnaya Str. 1, Zheleznogorsk, 662972, Russia

[2]CIDMA - Center for R&D in Mathematics and Applications, Dep. of Mathematics, U. Aveiro, Portugal

gram with respect to a given specification. Computational systems often have interfaces that encapsulate the local states of program objects and focus instead on the operations that modify the local states and some distinguished set of specific properties — *attributes* (in particular, these features are inherent to the *object oriented* (OO) *paradigm*). In order to implement this approach, many programming languages use techniques that hide internal data types, providing abstraction from unimportant details and protection for internal data.

Like a state of a transition system, a state of an OO program can be viewed as encapsulating all pertinent information about the abstract machine when it reaches a certain stage while executing a sequence of methods and procedures. The information about the local state conceptually is partitioned into a *visible* and a *hidden* part, with the former representing visible data, like attributes, and the latter representing the hidden states of objects in the OO paradigm. Methods can modify the hidden state of the object (giving as a result a new hidden state). Hidden states (data) can only be indirectly compared by considering the visible outputs of the same programs, applied to this hidden data. Using this approach, software can be designed according to a *behavioral* specification, the latter specifying certain *visible* behavior, instead of providing detailed requirements for all internal aspects of execution. In this respect, the above-mentioned approach turned out to be related to the *coalgebraic* approach (cf. [38]).

Behaviorally, two terms are said to be equivalent if and only if they cannot be distinguished in any visible context. This basic notion of behavioral equivalence is due to Reichel [51]. The idea of using the satisfaction relation on hidden terms for determining behavioral equivalence was also introduced by Reichel in the 80's [51] and it seems to be a useful way of defining equivalence between hidden terms. In fact, in applied settings, there are some well designed pieces of software that may fail to satisfy their requirements strictly, but do satisfy them behaviorally, i.e., under any program executed on the system (see [6, 3, 9]). More formally, in such approach, the standard equality predicate is augmented by *behavioral equivalence* (two data elements representing states are said to be *behaviorally equivalent* if every function returns the same visible value when executed on the same visible input). Behavioral equivalence has been adopted and generalized by many researches. The most significant contributions have been provided by Goguen, Bidoit, Bouhoula and their associates.

Hidden algebras were introduced by Goguen in [26] and further developed

in [25, 27], in order to generalize many-sorted algebras to give an algebraic semantics for the object oriented paradigm. In fact, the behavioral aspects of modern software make hidden algebras in practice more suitable than standard algebras as abstract machine implementations. Consequently, there has been an increasing interest in this field. Goguen and his collaborators have been improving their theory and applying it in more general settings. Now almost all of the results may be established for the general setting of *polyadic loose-data semantics*. Polyadic loose-data semantics allow any kind of operation symbols and, in order to have more freedom in choosing an adequate implementation, the visible part of the algebras is no longer fixed: it may be any sorted algebra in which the requirements (axioms) of the given specification are valid. However, some authors are interested in applying coalgebraic methods, and therefore they have to restrict their signatures to the monadic fixed-data semantics. Malcolm [32] had shown that behavioral equivalence may be formulated in the context of coalgebra (see also [31] and [52]).

Several generalizations of the notion of behavioral equivalence have been considered. Goguen et al. [25, 55] consider Γ-*behavioral equivalence*, where Γ is a subset of the set of all operation symbols in the signature. Γ-behavioral equivalence is defined analogously to ordinary behavioral equivalence, but only makes use of the contexts built from the operation symbols in Γ. It can be proven that the Γ-behavioral equivalence is the largest Γ-congruence with the identity as the visible part. Based on this fact, the coinduction methods may still be formulated for this more general notion. Other interesting questions concerning Γ-behavioral equivalence also arise, such as the study of the compatibility of some operation symbols outside of Γ with respect to Γ-behavioral equivalence. This problem has been studied by Diaconescu et al. [19] and Bidoit et al. [4]. On the other hand, Bidoit and Hennicker [5] generalized the notion by endowing the hidden algebras with a binary relation, that may be partial. In particular, one can apply the algebraic approach to the behavioral setting by considering the algebras together with the Γ-behavioral equivalence.

Various notions of behavioral logics have been considered for reasoning about behavioral equivalence. The most relevant to the topic of this paper are the variations developed by Goguen et al. (*hidden logics* [24, 25]) and by Bidoit and Hennicker (*observational logics* [28, 3]). These approaches formalize behavioral validity (correctness) as follows: hidden logic is a variant of the equational logic in which some part of the specification is visible and

another is hidden. The basic syntactic structures for hidden logics are equations and the behavioral satisfaction relation is defined by interpreting the equality symbol as the *behavioral equality*. Observational logics are different from hidden logics in that respect that instead of dealing with equations and quasi-equations, they work with first-order formulas. Accordingly, Tarski's satisfaction relation of first-order formulas (with equality) is reinterpreted as a "behavioral satisfaction relation" which is determined, in a natural way, by the family of congruence relations (partial, in general) with which each algebra is provided. Thus both approaches are based on behavioral equivalence, i.e., indistinguishability in all pertinent contexts, with differently understood notions of contexts. There is also another kind of observational logic due to Padawitz, called *swinging types logic*[3], but it is similar to the observational logic of Bidoit and Hennicker (see [46, 45]).

Several semi-automatic methods were suggested for behavioral reasoning in hidden logics. In particular, Diaconescu and Futatsugi [18] have developed the **CafeOBJ** language. It implements behavioral rewriting to make behaviorally sound reductions of terms and is based on a behavioral version of the well-known method of rewriting for automated theorem proving[4]. Goguen et al. have been developing algorithms for automated behavioral reasoning based on their techniques of coinduction and have been making use of cobases. Coinduction in its pure form requires human intervention in the choice of the cobasis. A cobasis is just a set of operation symbols that generates a relation on the set of terms which is a subset of the behavioral equivalence. A good choice of a cobasis can simplify the proof enormously. Those algorithms have been improved in order to be applied to more general situations and have been implemented in **BOBJ** language. In [53] Goguen and Roşu present a new technique which combines behavioral rewriting coinduction (see also [30] and [54]). The most recent version is CCCRW, called *conditional circular coinductive rewriting with case analysis*. The authors claim that it was in fact the most powerful automated proof technique available at that moment [22] (see also [23, 44]). Besides the fact that this new algorithm uses conditional circular coinductive rewriting to prove behavioral validity, it also allows case analysis. This theoretical results supports the

[3] `http://ls1-www.cs.uni-dortmund.de/padawitz/Swinging.html`.

[4] See `http://cafeobj.org/` for details.

automated behavioral prover **Circ** based on the circularity principle[5]. This principle generalizes both circular coinduction and structural induction.

Our approach to behavioral equivalence differs substantially from the standard approach in Computer Science (CS) described above, in the sense that it is greatly influenced by Abstract Algebraic Logic (AAL) (see [33] and [40]). This is, in fact, the main novelty of the presented research agenda. AAL is an area of algebraic logic that focuses on the study of the relationship between logical equivalence and logical truth (see [7, 13, 21]). More precisely, AAL is centered on the process of associating a class of algebras to a logical system. This approach contrasts with the usual treatment given in Algebraic Logic where the emphasis is on the study of the class of algebras obtained in this process.

In order to apply AAL to the theory of specification of abstract data types, we have to look at the specification logic as a deductive system, i.e., as a substitution-invariant closure relation on an appropriate set of formulas, and at the behavioral equivalence as some generalized notion of the *Leibniz congruence* (cf. [7, 40]). The notion of Leibniz congruence has to be considered in the context of the dichotomy of visible vs hidden data: namely, the formulas used in the characterization of the Leibniz congruence also have to be restricted to an appropriate proper subset of all formulas, namely the formulas (called sometimes *contexts*) that, when computed as algebraic terms, give results of visible type. In addition, the traditional AAL notion of a deductive system has to be generalized to situations where data is considered to be heterogeneous, that is, the data elements may be of different sorts. For instance, the basic data is usually split into several sorts, like integers, reals and Booleans, whose properties are well-known and for which well-defined and easily manipulated representations are available; and there are auxiliary data types, of various hidden sorts, such as arrays, lists, stacks, maps, sets, channels, sorted lists, associative arrays and so on, whose properties will be specified by their behavior in execution runs ending with visible output, and hence ultimately in terms of the basic data. Thus we separate the basic data and the auxiliary data by splitting the data elements in two types: the ones to which the user has direct access to (*visible data*) and those (*hidden data*) that the user only has access to by observing the visible output of programs that were applied to these data.

[5] `http://fsl.cs.illinois.edu/index.php/Circ`

In order to specify and reason about the properties of OO systems, we use in our approach hidden k-logics that accommodate the dichotomy hidden vs. visible by considering hidden and visible k-terms and k-formulas. These systems are a natural generalization of k-deductive systems which allow for the use of tools and results from abstract algebraic logic, more specifically the use of the central notion of Leibniz congruence.

The adequacy of using the Leibniz congruence, and consequently its properties, to deal with behavioral equivalence in the OO paradigm is expressed in Theorem 2.1. In fact, the second-order definability principle due to Leibniz is directly related to the behavioral equivalence concept. Moreover, both can be characterized as the largest congruence with a concrete property: compatibility with a filter, for the Leibniz congruence (cf. [33]), and having identity on the visible part, for the behavioral equivalence [25]. Moreover, these notions coincide for hidden equational logics — the Rosu's hidden logics (cf. [40]).

The most common notion of equivalence between specifications (cf. [29]) asserts that two specifications are *semantically equivalent* if they have the same class of models. From the point of view of implementation, this definition is too restrictive. There are other ways of defining equivalence between logics (see for instance [11]). In this paper, we discuss a notion of equivalence that generalizes to the hidden setting the well-known concept of equivalence between deductive systems in the field of AAL (cf. [8]). A paradigmatic example is the equivalence of the equational logic of Boolean algebras with classical propositional calculus.

We would like to point out that the behavioral specification theory has also influenced AAL. Caleiro et al. discussed in [10] the new notion of behavioral algebraization based on the notion of behavioral equivalence. They showed that this notion can capture some algebraic properties of well-known logical systems.

1.1. Outline of the paper.

In Section 2 we start by presenting the basic concepts of sorted set theory and the basic theory of hidden k-logics (for more details see [56] and [33]). In order to show the diversity of logical systems which can be captured by this notion, we present two examples, namely a logic for reasoning about intervals in linearly ordered sets and another one for sorted quasi-orders. The theory of hidden logics unifies several approaches to behavioral specification systems, as well as many other examples, under the umbrella of a general notion of

hidden logic that generalizes the well studied notion of a deductive system. In Section 3, we introduce the notion of equivalence between hidden k-logics via *interpretations*, based on *display translations*. Section 4 contains some technical observations about interpretations and ends with the main result of the paper, Theorem 4.6, that provides a necessary condition for two hidden logics to be behaviorally equivalent.

2. Hidden sorted algebra

In this section we review some basic concepts of the sorted Universal algebra in order to set our notation (for more details see [56]).

2.1. Sorted set theory

Let SORT be a nonempty set whose elements are called *sorts*. A SORT-*sorted set* A (or simply a *sorted set* if SORT is clear from the context) is a family of sets indexed by SORT, which we denote by $A = \langle A_S \mid S \in \text{SORT} \rangle$. Sometimes it is worth to consider a sorted set A as an ordinary set, namely as $\bigcup_{S \in \text{SORT}} A_S$; consequently we may write $a \in A$ to mean $a \in A_S$ for some $S \in \text{SORT}$.

A sorted set A is called *nonempty* if $A_S \neq \varnothing$ for every $S \in \text{SORT}$. A is *locally countably infinite* (*locally finite*) if, for every $S \in \text{SORT}$, A_S is a countably infinite (finite) set; and A is said to be *globally finite* if A is locally finite and A_S is empty except for a finite number of sorts. Note that, if SORT is finite, then "global" and "local" finiteness are equivalent.

A family of SORT-sorted sets, indexed by a set of sorts SORT$'$, is called a (SORT$'$ − SORT)-*complex sorted set* and we write in this case $C = \langle C_R \mid R \in \text{SORT}' \rangle$, where C_R is a SORT-sorted set for each $R \in \text{SORT}'$. In the case of SORT$'$ = SORT a (SORT$'$ − SORT)-complex sorted set is simply called a *double* SORT-*sorted set*. We say that C is *globally finite* if C_R is globally finite for every $R \in \text{SORT}'$. If sets of sorts are clear from context, we refer to such families as a sorted set, a complex sorted set, and a double sorted set, without explicit reference to the set of sorts. Similar notational simplifications will be used below without further warning. For any sorted set A and $\Lambda \subseteq \text{SORT}$, A_Λ denotes the Λ-sorted set $\langle A_S \mid S \in \Lambda \rangle$.

The standard notions of set theory extend in a natural way to multi-sorted case. Given sorted sets A and B, A is a *sorted subset* of B, in symbols $A \subseteq B$, if $A_S \subseteq B_S$ for all $S \in \text{SORT}$. Operations, such as union, intersection, direct product, etc, are defined componentwise.

For each nonzero natural k, we define the *sorted (cartesian) k^{th}-power of* A, denoted A^k, as the sorted set $\langle A_S^k \mid S \in \mathrm{SORT}\rangle$. By a *$k$-element* of A of sort S we mean simply an element $\bar{a} = \langle a_0, \ldots, a_{k-1}\rangle \in A_S^k$, i.e., a k-tuple of elements of A_S. The k-elements are represented by putting bars over the symbols, that we use to represent elements of A. A k-subset is a sorted subset of the sorted k^{th}-power of A.

The *sorted power set of* A, denoted by $\mathcal{P}(A)$, is the sorted set $\langle B \mid B \subseteq A\rangle$. We write $B \subseteq_{\mathcal{GF}} A$ if B is a sorted subset of A and B is globally finite. The set of all globally finite sorted subsets of A is denoted by $\mathcal{P}_{\mathcal{GF}}(A)$, i.e., $\mathcal{P}_{\mathcal{GF}}(A) = \{B \mid B \subseteq_{\mathcal{GF}} A\}$.

A *sorted function or mapping* h from A to B, in symbols $h : A \to B$, is a sorted set $h = \langle h_S \mid S \in \mathrm{SORT}\rangle$ of functions such that $h_S : A_S \to B_S$ for each $S \in \mathrm{SORT}$. A sorted mapping is called injective, surjective or bijective if all its components have the corresponding property. We will use the standard set-theoretical notation without explicit use of indices.

Let us define for every unsorted mapping $h : A \to B$ and every k-element $\bar{a} = \langle a_0, \ldots, a_{k-1}\rangle \in A^k$, $h(\bar{a}) := \langle h(a_0), \ldots, h(a_{k-1})\rangle \in B^k$ and for any $\bar{b} = \langle b_0, \ldots, b_{k-1}\rangle \in B^k$, $h^{-1}(\bar{b}) := \{\bar{a} \in A^k : h(\bar{a}) = \bar{b}\}$. This notation extends naturally to sets in the following way: let $F \subseteq A^k$, we define the *image of* F, in symbols $h(F)$, to be the set $h(F) = \big\{\langle h(a_0), \ldots, h(a_{k-1})\rangle \mid \langle a_0, \ldots, a_{k-1}\rangle \in F\big\}$ and we define the *pre-image of* $G \subseteq B^k$, in symbols $h^{-1}(G)$, to be the set $h^{-1}(G) = \bigcup_{\bar{b}\in G} h^{-1}(\bar{b})$.

The *kernel of* h, denoted by $\ker(h)$, is defined by $\ker(h) = \{\langle a, b\rangle \in A^2 \mid h(a) = h(b)\}$. For sorted maps these notions are defined componentwise, if $h : A \to B$ is a sorted mapping, then we define the *sorted image of* $F \subseteq A^k$, in symbols $h(F)$, to be the set $h(F) = \langle h_S(F_S) \mid S \in \mathrm{SORT}\rangle$ and we define the *sorted pre-image* of $G \subseteq B^k$, in symbols $h^{-1}(G)$, to be the set $h^{-1}(G) = \langle h_S^{-1}(G_S) \mid S \in \mathrm{SORT}\rangle$. The *sorted kernel* of h is the sorted set $\ker(h) = \langle \ker(h_S) \mid S \in \mathrm{SORT}\rangle$.

A *sorted equivalence relation* on A is a sorted subset Θ of A^2 such that Θ_S is an equivalence relation on A_S for each $S \in \mathrm{SORT}$. The sorted identity equivalence relation defined for each $S \in \mathrm{SORT}$ by $\{\langle a, a\rangle \mid a \in A_S\}$ is denoted by $\triangle_A$, i.e., $\triangle_A = \langle \triangle_{A_S} \mid S \in \mathrm{SORT}\rangle$. Given a sorted equivalence relation Θ on A, we define the *sorted quotient* of A by Θ, denoted by A/Θ, as the sorted set $\langle A_S/\Theta_S \mid S \in \mathrm{SORT}\rangle$, where $A_S/\Theta_S = \{[a]_{\Theta_S} \mid a \in A_S\}$. The mapping $nat : A \to A/\Theta$, defined by $nat_S(a) = [a]_{\Theta_S}$ for each $a \in A_S$, is called the *natural mapping*.

A *hidden* (*sorted*) *signature* is a triple

$$\Sigma = \big\langle \mathrm{SORT}, \mathrm{VIS}, \langle \mathrm{OP}_\tau \mid \tau \in \mathrm{TYPE} \rangle \big\rangle,$$

where: SORT is a nonempty, countable set whose elements are called *sorts*; VIS is a subset of SORT, called the set of *visible sorts*; TYPE is a set of nonempty sequences $S_0, \ldots, S_n$ of sorts, called *types* and usually written as $S_0, \ldots, S_{n-1} \to S_n$; and, for each $\tau \in \mathrm{TYPE}$, OP_τ is a countable set. We will denote $\langle \mathrm{OP}_\tau \mid \tau \in \mathrm{TYPE} \rangle$ by OP.

The sorts in $\mathrm{SORT} \setminus \mathrm{VIS}$, that are not visible, are called *hidden sorts*. The set of hidden sorts is denoted by HID. The elements of OP_τ are called *operation symbols of type* τ. Operation symbols of type $\to S$ are said to be *constants*. For simplicity, we require the sets of operation symbols to be pairwise disjoint in order to avoid overloading of names (i.e., for any distinct $\tau, \tau' \in \mathrm{TYPE}$, $\mathrm{OP}_\tau \cap \mathrm{OP}_{\tau'} = \varnothing$).

From each hidden signature Σ we obtain the associated *un-hidden* signature Σ^{UH} by making all sorts of Σ visible.

Two hidden signatures Σ, Σ' are said to be *algebraically indistinguishable* (a.i., for short) if $\Sigma^{\mathrm{UH}} = (\Sigma')^{\mathrm{UH}}$, i.e., if they have the same un-hidden signature.

A Σ-*algebra* is a pair $\big\langle A, \langle O^A \mid \tau \in \mathrm{TYPE}, O \in \mathrm{OP}_\tau \rangle \big\rangle$, where A is a SORT-sorted set, such that $A_S \neq \varnothing$, for all $S \in \mathrm{SORT}$, and for any $\tau \in \mathrm{TYPE}$ and $O \in \mathrm{OP}_\tau$, O^A is an operation on A of type τ (i.e., if $\tau = S_0, \ldots, S_{n-1} \to S_n$ then $O^A : A_{S_0} \times \cdots \times A_{S_{n-1}} \to A_{S_n}$). As usual, we use the same symbol to denote an algebra and the carrier of the algebra.

We assume for carrier sets A of data structures that $A_S \neq \varnothing$, for all $S \in \mathrm{SORT}$, a condition similar to the one used to define regular universal algebras. With this assumption we exclude some data structures of practical interest. However, the mathematics is simpler in this case and most results of universal algebra hold in their usual form.

A (*sorted*) *congruence on* a Σ-algebra A is a sorted binary relation $\Theta \subseteq A^2$ such that: (i) for each $S \in \mathrm{SORT}$, Θ_S is an equivalence relation on A_S and (ii) Θ satisfies the congruence condition: for every operation symbol $O \in \mathrm{OP}_\tau$ with $\tau = S_0, \ldots, S_{n-1} \to S_n$, and all $a_0, a'_0 \in A_{S_0}, \ldots, a_{n-1}, a'_{n-1} \in A_{S_{n-1}}$ such that $a_i \Theta_{S_i} a'_i$, $O^A(a_0, \ldots, a_{n-1}) \Theta_{S_n} O^A(a'_0, \ldots, a'_{n-1})$ holds. The set of all congruences over A is denoted by $\mathrm{Con}(A)$.

The sorted notions of subalgebra, homomorphism, isomorphism, etc. are defined in a natural way (see [41] for the formal definitions).

For each set of sorts SORT, we fix a locally countably infinite sorted set $X = \langle X_S \mid S \in \mathrm{SORT}\rangle$ of *(propositional sorted) variables*. We assume that the components of the sorted set of variables are pairwise disjoint. The elements in X_S are called *S-variables*. To denote that a variable x is of sort S (i.e., that $x \in X_S$) we write $x{:}S$.

We say that a term $O(t_0, \ldots, t_{n-1})$ has type S_n if $O \in \mathrm{OP}_\tau$, with $\tau = S_0, \ldots, S_{n-1} \to S_n$. Given a signature Σ, we define the SORT-sorted set $\mathrm{Te}_\Sigma(X)$ of terms over the signature Σ with variables in X as usual. Note that, since the components of the family $\mathrm{Te}_\Sigma(X)$ are pairwise disjoint, a SORT-sorted subset Γ of $\mathrm{Te}_\Sigma(X)$ can be identified with the unsorted set $\bigcup_{S\in\mathrm{SORT}} \Gamma_S$. A hidden signature Σ is said to be *standard* if there is a ground term (i.e., a term without variables) of every sort. We use the lower case Greek letters $\varphi, \psi, \vartheta, \ldots$ to represent terms, possibly with annotations to indicate sorts of terms and variables. Specifically, writing φ in the form

$$\varphi(x_0{:}S_0, \ldots, x_{n-1}{:}S_{n-1}){:}S \tag{1}$$

indicates that φ is of sort S and that the variables that actually occur in φ are included in the list $x_0, \ldots, x_{n-1}$ of variables of sorts $S_0, \ldots, S_{n-1}$, respectively.

We define, in the usual way, operations over $\mathrm{Te}_\Sigma(X)$ to obtain the *term algebra* over the signature Σ. It is well known that $\mathrm{Te}_\Sigma(X)$ has the *universal mapping property over X* in the sense that, for every Σ-algebra A and every sorted map $h : X \to A$, called an *assignment*, there is a unique sorted homomorphism $h^* : \mathrm{Te}_\Sigma(X) \to A$ that extends h. In the sequel, we will not distinguish between these two maps. If φ is the term (1), and $a_i \in A_{S_i}$, $i < n$, we write $\varphi^A(a_0, \ldots, a_{n-1})$ for the image $h(\varphi)$ under any homomorphism $h : \mathrm{Te}_\Sigma(X) \to A$ such that $h(x_i) = a_i$ for all $i < n$. A map from X to the set of terms, and its unique extension to an endomorphism of $\mathrm{Te}_\Sigma(X)$, is called a *substitution*.

To provide a context that allows us to deal simultaneously with specification logics that are assertional (for example the ones with a Boolean sort but no equality) and equational, we introduce the notion of a k-term for any nonzero natural number k. In the sequel k denotes a fixed nonzero natural number. A *k-variable of sort S* is a sequence of k-variables all of the same sort S. A *k-term (k-*formula *in logical context) of sort S over Σ* is a sequence of k Σ-terms all of the same sort S. We indicate k-terms by overlining, so $\bar\varphi(\bar x){:}S = \langle \varphi_0(\bar x){:}S, \ldots, \varphi_{k-1}(\bar x){:}S\rangle$. When we do not need to make the common sort S of each term of $\bar\varphi{:}S$ explicit,

we simply write it as $\bar{\varphi}$. $\mathrm{Te}^k_\Sigma(X)$ is the sorted set of all k-terms over Σ. Thus $\mathrm{Te}^k_\Sigma(X) = \langle (\mathrm{Te}_\Sigma(X))^k_S : S \in \mathrm{SORT}\rangle$. The set of all *visible k-terms* $(\mathrm{Te}^k_\Sigma(X))_{\mathrm{VIS}}$ is the VIS-sorted set $\langle (\mathrm{Te}_\Sigma(X))^k_V : V \in \mathrm{VIS}\rangle$.

2.2. Data structures

Let Σ be a hidden signature. A *visible k-data structure* (a *k-data structure* for short) *over* Σ is a pair $\mathcal{A} = \langle A, F\rangle$, where A is a Σ-algebra and $F \subseteq A^k_{\mathrm{VIS}}$; A is called the *underlying algebra* and F the *designated filter of* $\mathcal{A}$ (see [40] for examples in the hidden equational case).

Let $\langle A, F\rangle$ be a k-data structure over a hidden signature Σ. A congruence relation θ on A is *compatible with* F if, for all $V \in \mathrm{VIS}$ and for all $\bar{a}, \bar{a}' \in A^k_V$, $a_i \theta_V a'_i$, for all $i < k$, implies: $\bar{a} \in F_V$ iff $\bar{a}' \in F_V$. It is not difficult to see that the largest congruence relation on A compatible with F always exists (see [40]). Hence, given a k-data structure $\langle A, F\rangle$, we define the *Leibniz congruence* of F on A as the largest congruence relation on A compatible with F, which we denote by $\Omega_A(F)$, or simply $\Omega(F)$ when A is clear from the context. One of the main properties of the Leibniz congruence is its preservation under inverse images of surjective homomorphisms, i.e., given a k-data structure $\mathcal{A} = \langle A, F\rangle$ over Σ, a Σ-algebra B and a surjective homomorphism $h : B \to A$, we have that $h^{-1}(\Omega_A(F)) = \Omega_B(h^{-1}(F))$.

A systematic study of the properties of the Leibniz congruence in the hidden k-logics setting can be found in [33]. An interesting result that justifies the use of the term "Leibniz congruence" is the analogy with the second order definability principle due to Leibniz:

Theorem 2.1. *Given a k-data structure $\mathcal{A} = \langle A, F\rangle$ and $a, a' \in A_S$, $a \equiv a'$ $(\Omega(F)_S)$ iff, for every $V \in \mathrm{VIS}$ and every k-term*

$$\bar{\varphi}(z{:}S, u_0{:}Q_0, \ldots, u_{m-1}{:}Q_{m-1}){:}V$$

and all $b_0 \in A_{Q_0}, \ldots, b_{m-1} \in A_{Q_{m-1}}$,

$$\bar{\varphi}^A(a, b_0, \ldots, b_{m-1}) \in F_V \text{ iff } \bar{\varphi}^A(a', b_0, \ldots, b_{m-1}) \in F_V.$$

This result also points out the adequacy of using the Leibniz congruence to present the behavioral equivalence in the OO paradigm (cf. [33]). Moreover, it can be seen as a generalization of the corresponding well-known result in Hidden Algebra (cf. [25]).

2.3. Hidden k-logic

For each nonzero natural number k, a hidden k-logic is considered to be a consequence relation on the set of visible k-terms of some hidden signature, independently of any specific choice of axioms and rules of inference. More precisely, it is defined as a substitution invariant consequence relation on the set of visible k-terms.

Definition 2.1. A *hidden k-logical system*[6] (*hidden k-logic* for short) is a pair $\mathcal{L} = \langle \Sigma, \vdash_{\mathcal{L}} \rangle$, where Σ is a hidden signature with VIS as its set of visible sorts, and $\vdash_{\mathcal{L}} \subseteq \mathcal{P}((\mathrm{Te}^k_\Sigma(X))_{\mathrm{VIS}}) \times (\mathrm{Te}^k_\Sigma(X))_{\mathrm{VIS}}$ is an (unsorted) relation that satisfies for all $\Gamma \cup \Delta \cup \{\bar\gamma, \bar\varphi\} \subseteq (\mathrm{Te}^k_\Sigma(X))_{\mathrm{VIS}}$ the following conditions:

(i) $\Gamma \vdash_{\mathcal{L}} \bar\gamma$ for each $\bar\gamma \in \Gamma$;

(ii) if $\Gamma \vdash_{\mathcal{L}} \bar\varphi$, and $\Delta \vdash_{\mathcal{L}} \bar\gamma$ for each $\bar\gamma \in \Gamma$, then $\Delta \vdash_{\mathcal{L}} \bar\varphi$;

(iii) if $\Gamma \vdash_{\mathcal{L}} \bar\varphi$, then $\sigma(\Gamma) \vdash_{\mathcal{L}} \sigma(\bar\varphi)$ for every substitution σ.

Note that, being unsorted, $\vdash_{\mathcal{L}}$ can relate premises and consequences of different visible sorts.

A hidden k-logic is *specifiable* if $\vdash_{\mathcal{L}}$ is *finitary* (or compact), i.e., if $\Gamma \vdash_{\mathcal{L}} \bar\varphi$ implies $\Delta \vdash_{\mathcal{L}} \bar\varphi$ for some globally finite subset Δ of Γ. The relation $\vdash_{\mathcal{L}}$ is called *the consequence relation of* $\mathcal{L}$; when $\mathcal{L}$ is clear from the context we simply write $\vdash$. A hidden k-logic with VIS = SORT will be called a *visible k-logic*, or simply a *k-logic*.

As common in a sentential logic framework, we treat formulas (k-formulas) as synonymous to terms (k-terms, respectively). Moreover, for a given hidden k-logic $\mathcal{L} = \langle \Sigma, \vdash_{\mathcal{L}} \rangle$ we denote $\mathrm{Te}^k_\Sigma(X)$ and $(\mathrm{Te}^k_\Sigma(X))_{\mathrm{VIS}}$ by $\mathrm{Fm}(\mathcal{L})$ and $\mathrm{Fm}_{\mathrm{VIS}}(\mathcal{L})$, respectively.

Hidden k-logics were introduced by Martins and Pigozzi (cf. [40]) in the context of the algebraic specification and verification of software systems. The basic theory of hidden k-logics was presented in [33]. The class of hidden k-logics includes such well-known logical systems as the 2-dimensional

[6]We use the term "hidden" for our notion of a logical system since, as an equational hidden logic, it uses the dichotomy hidden vs. visible. A similar notion of a *general logic*, defined as a closure relation as well, is due to Meseguer [42]. This system is called an entailment system and combines the closure relation with the notion of institution (see also [20] for a related notion of π-institutions). In the context of hidden paradigm, in [54], Roşu and Lucanu introduced the notion of a contextual entailment system.

hidden and standard equational logics, as well as the Boolean logic (for more examples see [33]).

Every consequence relation $\vdash$ has a natural extension to a relation, also denoted by $\vdash$, between sets of visible k-terms; it is defined by $\Gamma \vdash \Delta$ if $\Gamma \vdash \bar{\varphi}$ for each $\bar{\varphi} \in \Delta$. We define the relation of *interderivability* between sorted sets in the following way: $\Gamma \dashv\vdash \Delta$ if, $\Gamma \vdash \Delta$ and $\Delta \vdash \Gamma$. We will abbreviate $\{\bar{\psi}\} \vdash \bar{\varphi}$, $\Gamma \cup \{\bar{\varphi}_0, \ldots, \bar{\varphi}_{n-1}\} \vdash \bar{\varphi}$ and $\Gamma_0 \cup \cdots \cup \Gamma_{n-1} \vdash \bar{\varphi}$ by $\bar{\psi} \vdash \bar{\varphi}$, $\Gamma, \bar{\varphi}_0, \ldots, \bar{\varphi}_{n-1} \vdash \bar{\varphi}$ and $\Gamma_0, \ldots, \Gamma_{n-1} \vdash \bar{\varphi}$, respectively.

Let $\mathcal{L}$ be a (not necessarily specifiable) hidden k-logic. By a *theorem* of $\mathcal{L}$ we mean a visible k-term $\bar{\varphi}$ such that $\vdash_{\mathcal{L}} \bar{\varphi}$, i.e., $\varnothing \vdash_{\mathcal{L}} \bar{\varphi}$. The set of all theorems is denoted by $\mathrm{Thm}(\mathcal{L})$. A rule such as

$$\frac{\bar{\varphi}_0 : V_0, \ldots, \bar{\varphi}_{n-1} : V_{n-1}}{\bar{\varphi}_n : V_n}, \tag{2}$$

where $\bar{\varphi}_0, \ldots, \bar{\varphi}_n$ are all visible k-terms, is said to be a *derivable rule* of $\mathcal{L}$ if $\{\bar{\varphi}_0, \ldots, \bar{\varphi}_{n-1}\} \vdash_{\mathcal{L}} \bar{\varphi}_n$. A set of visible k-terms T closed under the consequence relation, i.e., $T \vdash_{\mathcal{L}} \bar{\varphi}$ implies $\bar{\varphi} \in T$, is called a *theory* of $\mathcal{L}$ or $\mathcal{L}$-*theory*. The set of all theories is denoted by $\mathrm{Th}(\mathcal{L})$; it forms a complete lattice under set-theoretic inclusion, which is algebraic if $\mathcal{L}$ is specifiable. Let $T_i \in \mathrm{Th}(\mathcal{L})$, for $i \in I$. Their meet is $\bigcap_{i \in I} T_i$ and their join is the intersection of all theories that contain each T_i, i.e., $\bigvee^{\mathcal{L}}_{i \in I} T_i = \bigcap \{T \in \mathrm{Th}(\mathcal{L}) : T_i \subseteq T \text{ for all } i \in I\}$. Given any set Γ of visible k-terms, the set $\mathrm{Cn}_{\mathcal{L}}(\Gamma)$ is the smallest $\mathcal{L}$-theory containing Γ. It is easy to see that $\mathrm{Cn}_{\mathcal{L}}(\Gamma) = \{\, \bar{\varphi} \in (\mathrm{Te}^k_\Sigma(X))_{\mathrm{VIS}} : \Gamma \vdash_{\mathcal{L}} \bar{\varphi}\}$, i.e. is the set of all consequences of Γ.

Very often, a specifiable hidden k-logic is presented in the so-called Hilbert style, i.e., by a set of axioms (visible k-terms) and inference rules of the general form (2). We say that a visible k-term $\bar{\psi}$ is *directly derivable* from a set Γ of visible k-terms by a rule such as (2) if there is a substitution $h : X \to \mathrm{Te}_\Sigma(X)$ such that $h(\bar{\varphi}_n) = \bar{\psi}$ and $h(\bar{\varphi}_0), \ldots, h(\bar{\varphi}_{n-1}) \in \Gamma$.

Given a set AX of visible k-terms and a set IR of inference rules, we say that $\bar{\psi}$ is *derivable* from Γ by the set AX and the set IR, in symbols $\Gamma \vdash_{\mathrm{AX,IR}} \bar{\psi}$, if there is a finite sequence of k-terms, $\bar{\psi}_0, \ldots, \bar{\psi}_{n-1}$ such that $\bar{\psi}_{n-1} = \bar{\psi}$, and for each $i < n$ one of the following conditions hold:

(a) $\bar{\psi}_i \in \Gamma$,
(b) $\bar{\psi}_i$ is a substitution instance of a k-term in AX
(c) $\bar{\psi}_i$ is directly derivable from $\{\bar{\psi}_j\}_{j<i}$ by one of the inference rules in IR.

It is clear that $\langle \Sigma, \vdash_{\mathrm{AX,IR}} \rangle$ is a specifiable hidden k-logic. Moreover, a hidden k-logic $\mathcal{L}$ is specifiable iff there exist (possibly infinite) sets AX and IR, of axioms and inference rules, respectively, such that, for any visible k-terms $\bar{\psi}$ and any set Γ of visible k-terms, $\Gamma \vdash_{\mathcal{L}} \bar{\psi}$ iff $\Gamma \vdash_{\mathrm{AX,IR}} \bar{\psi}$. The pair $\langle \mathrm{AX}, \mathrm{IR} \rangle$ is called a presentation of $\mathcal{L}$ by axioms and inference rules. Hence we can present our examples of specifiable logics by exhibiting their sets of axioms and of inference rules. If $\mathcal{L} = \langle \Sigma, \vdash_{\mathrm{AX,IR}} \rangle$, for some AX and IR with $|\mathrm{AX} \cup \mathrm{IR}| < \omega$, we say that $\mathcal{L}$ is *finitely axiomatizable*.

2.3.1. Semantics.

Let $\mathcal{A} = \langle A, F \rangle$ be a k-data structure. A visible k-term $\bar{\varphi}{:}V$ is said to be a *semantic consequence* of a set of visible k-terms Γ in $\mathcal{A}$, in symbols $\Gamma \models_{\mathcal{A}} \bar{\varphi}$, if, for every assignment $h : X \to A$, $h(\bar{\varphi}) \in F_V$ whenever $h(\bar{\psi}) \in F_W$ for every $\bar{\psi}{:}W \in \Gamma$. A visible k-term $\bar{\varphi}$ is a *validity* of $\mathcal{A}$, and conversely $\mathcal{A}$ is a *model* of $\bar{\varphi}$, if $\varnothing \models_{\mathcal{A}} \bar{\varphi}$. A rule such as (2) is a *validity*, or a *valid rule*, of $\mathcal{A}$, and conversely $\mathcal{A}$ is a *model* of the rule, if $\{\bar{\varphi}_0, \ldots, \bar{\varphi}_{n-1}\} \models_{\mathcal{A}} \bar{\varphi}_n$. A visible formula $\bar{\varphi}$ is a *semantic consequence* of a set of visible k-terms Γ for an arbitrary class $\mathcal{M}$ of k-data structures over Σ, in symbols $\Gamma \models_{\mathcal{M}} \bar{\varphi}$, if $\Gamma \models_{\mathcal{A}} \bar{\varphi}$ for each $\mathcal{A} \in \mathcal{M}$. It can be proved that $\models_{\mathcal{M}}$ is always a logic, however it may not be specifiable. A visible k-term or rule as (2) is a *validity of* $\mathcal{M}$ if it is a validity of each member of $\mathcal{M}$.

A k-data structure $\mathcal{A}$ is a *model* of a hidden k-logic $\mathcal{L}$ if every consequence of $\mathcal{L}$ is a semantic consequence of $\mathcal{A}$, i.e., $\Gamma \vdash_{\mathcal{L}} \bar{\varphi}$ always implies $\Gamma \models_{\mathcal{A}} \bar{\varphi}$, for every $\Gamma \cup \{\bar{\varphi}\} \subseteq (\mathrm{Te}^k_\Sigma(X))_{\mathrm{VIS}}$. When $\mathcal{A}$ is a model of $\mathcal{L}$, the designated filter of $\mathcal{A}$ is called an $\mathcal{L}$*-filter over* A. The set of all $\mathcal{L}$-filters over an algebra A is denoted by $\mathrm{Fi}_{\mathcal{L}}(A)$. The special models whose underlying algebra is the formula algebra, i.e., of the form $\langle \mathrm{Te}_\Sigma(X), T \rangle$, with $T \in \mathrm{Th}(\mathcal{L})$, are called *Lindenbaum-Tarski models*. The class of all models of $\mathcal{L}$ is denoted by $\mathrm{Mod}(\mathcal{L})$. If $\mathcal{L}$ is a specifiable hidden k-logic, then $\mathcal{A}$ is a model of $\mathcal{L}$ iff every axiom and rule of inference is a validity of $\mathcal{A}$. The class of all *reduced models of* $\mathcal{L}$, i.e., all models $\langle A, F \rangle$ such that $\Omega(F) = id_A$, is denoted by $\mathrm{Mod}^*(\mathcal{L})$. A class of k-data structures $\mathcal{M}$ is a *data structure semantics* for $\mathcal{L}$ if $\vdash_{\mathcal{L}} = \models_{\mathcal{M}}$. The Completeness Theorem holds for hidden k-logics (cf. [40]), i.e., for every $\Gamma \cup \{\bar{\varphi}\} \subseteq (\mathrm{Te}^k_\Sigma(X))_{\mathrm{VIS}}$,

$$\Gamma \vdash_{\mathcal{L}} \bar{\varphi} \quad \text{iff} \quad \Gamma \models_{\mathrm{Mod}(\mathcal{L})} \bar{\varphi} \quad \text{iff} \quad \Gamma \models_{\mathrm{Mod}^*(\mathcal{L})} \bar{\varphi}.$$

Note that the mentioned notion of completeness is not the one concerning

complexity theory, it is rather the notion of semantic completeness, namely a logic is complete with respect to some semantics if any semantic consequence is a syntactic consequence.

An important class of hidden 2-logics is the class of *hidden equational logics*, where the notion of equality is only considered for visible data. It is defined (using the reflexivity, symmetry, transitivity and congruence rules) as a sorted equational logic, restricted to the visible part (cf. [40]).

In an equational framework, a pair of terms of the same sort $\langle s, t\rangle$ is called *an equation* and it is denoted by $s \approx t$.

Definition 2.2 (Free hidden equational logic, cf. [33])**.** Let Σ be a hidden signature and VIS its set of visible sorts.

1. The *free hidden equational logic over* Σ (*free* HEL_Σ for short) is the specifiable hidden 2-logic presented as follows:

   Axioms: for all $V \in \mathrm{VIS}$
   $$x{:}V \approx x{:}V$$

   Inference rules: for each $V, W \in \mathrm{VIS}$,

   (IR_1) $\dfrac{x{:}V \approx y{:}V}{y{:}V \approx x{:}V}$;

   (IR_2) $\dfrac{x{:}V \approx y{:}V, y{:}V \approx z{:}V}{x{:}V \approx z{:}V}$;

   (IR_3) $\dfrac{\varphi{:}V \approx \psi{:}V}{\vartheta(x/\varphi){:}W \approx \vartheta(x/\psi){:}W}$ for each $\vartheta \in \mathrm{Te}_W$ and each $x \in X_V$.

2. The *free un-hidden equational logic over* Σ (*free* UHEL_Σ, for short) contains an equality predicate for each sort, visible and hidden. The axioms and inference rules are the same as those of the free HEL_Σ, except that V and W are now allowed to range over all sorts. Thus $\mathrm{UHEL}_\Sigma = \mathrm{HEL}_{\Sigma^{\mathrm{UH}}}$.

An *applied hidden equational logic over* Σ (or simply a HEL_Σ) is any hidden 2-logic $\mathcal{L}$ over Σ that satisfies all axioms and rules of inference of the free HEL_Σ. An *applied un-hidden equational logic over* Σ (UHEL_Σ) is defined similarly; the subscript Σ may be omitted if it is clear from the context. The basic notions and results about hidden k-logics, as well as many examples of HELs may be found in [33] and [40]. In addition, the authors in [40] discussed in more detail this application of AAL tools, namely its advantages in the special case of HELs in comparison with the theory developed by Goguen

and his collaborators on behaviorally reasoning over behavioral conditional equations.

2.4. More examples

Example 2.1 (Nested intervals, [40])**.** As illustration we now give an example of a hidden 3-logic that formalizes some algebraic properties of the non-empty closed nested intervals of an abstract linearly ordered set. The 3-formula $\langle x, y, z\rangle$ represents the ternary ordering relation $x \leq y \leq z$, although there is no formal representation of the binary relation $\leq$. A *set* s is an interval $[n, m] = \{\, x : n \leq x \leq m \,\}$ of elements in the linear ordering, where n, m are respectively the *lower bound* ($lb(s)$), and the *upper bound* ($ub(s)$) of the interval.

The set of sorts SORT $= \{set, num\}$, where *num* is the only visible sort.

Operations.

$lub, glb : num, num \to num,$

$ub, lb : set \to num,$

$elt\text{-}of : set \to num,$

$\cup, \& : set, set \to set.$

Axioms.

$\langle x, x, x\rangle,$

$\langle glb(x, y),\ x,\ lub(x, y)\rangle,$

$\langle glb(x, y),\ y,\ lub(x, y)\rangle,$

$\langle lb(s),\ elt\text{-}of(s),\ ub(s)\rangle,$

$\langle glb(lb(s),\ lb(t)),\ elt\text{-}of(\cup(s, t)),\ lub(ub(s),\ ub(t)\rangle,$

$\langle lub(lb(s),\ lb(t)),\ elt\text{-}of(\&(s, t)),\ glb(ub(s),\ ub(t)\rangle.$

Rules of Inference.

$$\frac{\langle x,\ y,\ w\rangle,\ \langle y,\ z,\ w'\rangle}{\langle x,\ y,\ z\rangle},$$

$$\frac{\langle w,\ x,\ y\rangle,\ \langle w',\ y,\ z\rangle}{\langle x,\ y,\ z\rangle},$$

$$\frac{\langle x,\ z,\ x'\rangle,\ \langle y,\ z,\ y'\rangle}{\langle lub(x,\ y),\ z,\ glb(x', y')\rangle}.$$

♢

In the following example we consider a sorted version of a deductive system with quasi-ordering (for the extensive treatment of non-sorted logics see

[47, 50]). We assume that the signatures are endowed with a function that gives, for each operation and each argument position, its *polarity*. Formally, *a polarity for a signature* Σ is a TYPE-sorted map ρ such that for each type $\tau = S_0, \ldots, S_n \in \mathrm{TYPE}$, $\rho_\tau : \mathrm{OP}_\tau \to \{+, -\}^{n-1}$. Also a pair of terms of the same sort $\langle s, t\rangle$ will be denoted by $s \preceq t$.

Example 2.2 (Sorted quasi-ordering logic)**.** Let Σ be a sorted hidden signature with VIS = SORT. The *(sorted) quasi-ordering logic over* Σ (QOL_Σ for short) is the specifiable 2-logic presented as follows:

Axioms: for all $S \in \mathrm{SORT}$

$$x{:}S \preceq x{:}S$$

Inference rules: For each $S, S_i \in \mathrm{SORT}$, $i = 0, \ldots n-1$,

(IR_1) $$\frac{x{:}S \preceq y{:}S,\ y{:}S \preceq z{:}S}{x{:}S \preceq z{:}S}.$$

(IR_{2^+}) For each $\sigma(\ldots, x_i : S_i, \ldots) \in \mathrm{OP}_\tau$ and $\rho_\tau(\sigma)_i = +$ and $i < n$

$$\frac{\varphi{:}S_i \preceq \psi{:}S_i}{\sigma(x_0, \ldots, x_{i-1}, \varphi, x_{i+1}, \ldots, x_{n-1}){:}S \preceq \sigma(x_0, \ldots, x_{i-1}, \psi, x_{i+1}, \ldots, x_{n-1}){:}S}.$$

(IR_{2^-}) For each $\sigma(\ldots, x_i : S_i, \ldots) \in \mathrm{OP}_\tau$ and $\rho_\tau(\sigma)_i = -$ and $i < n$

$$\frac{\varphi{:}S_i \preceq \psi{:}S_i}{\sigma(x_0, \ldots, x_{i-1}, \psi, x_{i+1}, \ldots, x_{n-1}){:}S \preceq \sigma(x_0, \ldots, x_{i-1}, \varphi, x_{i+1}, \ldots, x_{n-1}){:}S}.$$

Let Σ be a signature with polarity ρ and A be a Σ-algebra. The models of the (sorted) quasi-ordering logic are pairs $\langle A, F\rangle$ where A is a Σ-algebra and F is a quasi-ordering in A compatible with the polarity of Σ. We can use infix notation $a \preceq_F b$ instead of $(a, b) \in F$.

For QOL_Σ the following holds:

Lemma 2.2. *Let $\langle A, F\rangle$ be a model of a quasi-ordering logic. Then a congruence θ is compatible with F iff $\theta \subseteq F$.*

Proof. Suppose that θ is compatible with F. Let $a\theta a'$. Since $a \preceq_F a$ and $a\theta a$, then we have by compatibility $a \preceq_F a'$.

Conversely, let $a, a', b, b' \in A$ such that $a\theta b$, $a'\theta b'$ and $a \preceq_F a'$. Since $\theta \subseteq F$ and θ is symmetric, we have $b \preceq_F a$, $a \preceq_F a'$, $a' \preceq_F b'$. Therefore, by (IR_1), $b \preceq_F b'$. □

Theorem 2.3. *Let $\langle A, F\rangle$ be a model of a quasi-ordering logic. Then the Leibniz congruence of F is exactly $F \cap F^{-1}$.*

Proof. $F \cap F^{-1}$ is a congruence on A, since it is reflexive, symmetric and closed under the rules (IR_1), (IR_{2^+}), (IR_{2^-}). Moreover, by the previous lemma, it is compatible with F.

Suppose now that θ is another congruence compatible with F. Let $a\theta b$. Since $a \preceq_F a$ and $a\theta a$, by compatibility we have that $\langle b, a\rangle \in F$, so $\langle a, b\rangle \in F^{-1}$. Since θ is symmetric, $b\theta a$, which by a similar argument implies that $\langle a, b\rangle \in F$. Hence $\langle a, b\rangle \in F \cap F^{-1}$.

Therefore $F \cap F^{-1}$ is the largest congruence compatible with F. □

It can be shown that QOL_Σ is equivalential in the sense that there are 2-formulas ($x \preceq y$ and $y \preceq x$) that define the Leibniz congruence for each model. We also would like to emphasize that there is no equality symbol in the language. ♢

3. Equivalent Hidden *k*-Logics

In this section we discuss a notion of equivalence between hidden logics (not necessarily of the same dimension). There are many ways of defining equivalence between logics (see for instance [11]). Our notion is syntactic, in the sense that it does not involve models of logics. It generalizes to the hidden setting the well-known concept of equivalence between deductive systems in the field of AAL (cf. [8]).

Now we introduce a notion of translation between hidden k-logics that may possibly differ on the sets of visible sorts. Let Σ and Σ' be two algebraically indistinguishable signatures with sets of visible sorts VIS and VIS$'$ respectively. A (k, l)-*translation* from Σ to Σ' is a (VIS$-$VIS$'$)-complex sorted set $\tau = \langle \tau_R(\bar{x}{:}R) \mid R \in \mathrm{VIS}\rangle$, where $\tau_R(\bar{x}{:}R) = \langle \bar{\tau}_{R,V}(\bar{x}{:}R) \mid V \in \mathrm{VIS}'\rangle$ is a VIS$'$-sorted set of visible terms, with $\bar{\tau}_{R,V}(\bar{x}{:}R)$ a finite set of l-terms over Σ' of sort V and whose variables range among $\bar{x} = \langle x_0{:}R, \ldots, x_{k-1}{:}R\rangle$. If VIS$'$ = SORT, τ is called a *display* (k, l)-*translation* of Σ. If, in addition, $l = 2$ then τ is called an *equational display* k-*translation* of Σ.

For each $\bar{\varphi} \in (\mathrm{Te}^k_\Sigma(X))_R$, we will write $\tau_R(\bar{\varphi})$ to denote the VIS$'$-sorted set $\langle \bar{\tau}_{R,V}(\bar{\varphi}) \,|\, V \in \mathrm{VIS}'\rangle$. Given a set Γ of visible k-terms and τ a translation, $\tau(\Gamma)$ is the VIS$'$-sorted set $\bigcup\{\tau_S(\bar{\varphi}) \mid \bar{\varphi} \in \Gamma_S, S \in \mathrm{VIS}\}$. When Γ is a singleton, say $\Gamma = \{\bar{\gamma}\}$, we write $\tau(\bar{\gamma})$ instead of $\tau(\{\bar{\gamma}\})$.

We say that the translation τ is an *interpretation from* $\mathcal{L}$ *to* $\mathcal{L}'$, over hidden signatures Σ and Σ', respectively, if

$$\Gamma \vdash_{\mathcal{L}} \bar{\varphi} \quad \text{iff} \quad \tau(\Gamma) \vdash_{\mathcal{L}'} \tau(\bar{\varphi}).$$

This notion of interpretation has been successfully used in the abstract development of software systems (by providing a broad concept of refinement that accommodates several crucial transformation approaches in software development (see [37, 38, 39])), as well as in a variety of other contexts (see for example [43, 12]).

The following notion of equivalence is defined using translations and extends the notion of equivalence defined in [8] for one-sorted logics, used to study the algebraic counterparts of deductive systems.

Definition 3.1 (Equivalence of hidden logics)**.** Let Σ and Σ' be two a.i. hidden signatures, $\mathcal{L}$ a hidden k-logic over Σ, and $\mathcal{L}'$ a hidden l-logic over Σ'. We say that $\mathcal{L}$ is (syntactically) *equivalent* to $\mathcal{L}'$ if there are globally finite interpretations τ and ρ, from $\mathcal{L}$ to $\mathcal{L}'$ and from $\mathcal{L}'$ to $\mathcal{L}$, respectively, which are inverse to one another, i.e.,

$$\bar{x}{:}V \dashv\vdash_{\mathcal{L}} \rho(\tau(\bar{x}{:}V)) \text{ and } \bar{y}{:}V' \dashv\vdash_{\mathcal{L}'} \tau(\rho(\bar{y}{:}V')),$$

where $\bar{x}{:}V$ and $\bar{y}{:}V'$ are a k-variable and an l-variable respectively, for all $V \in \mathrm{VIS}$ and $V' \in \mathrm{VIS}'$.

It is straightforward to show that equivalence of logics is an equivalence relation. It can also be proved that a presentation of a hidden logic $\mathcal{L}$ can be automatically transformed into a presentation of any system $\mathcal{L}'$ equivalent to $\mathcal{L}$ (see [8] for the one-sorted case). Moreover, if $\mathcal{L}$ admits a finite presentation by axioms and inference rules, so does $\mathcal{L}'$. There are also some other interesting properties that are preserved under such equivalence. In particular, the Craig interpolation property and the deduction-detachment theorem were previously discussed in [2].

Example 3.1 (One-sorted case)**.** It is well known that the classical propositional calculus (CPC) is equivalent to the equational logic of Boolean algebras by the translations $\tau(p) = \langle p, \top \rangle$ and $\rho(\langle p, q \rangle) = \{p \leftrightarrow q\}$ (see [8]). $\diamondsuit$

Example 3.2 (One-sorted case)**.** Semilattices can be viewed both as an algebraic and as a relational structure, namely as a partially ordered structure. This duality can be expressed by means of an equivalence witnessed by the translations $\tau(\langle p, q \rangle) = \{\langle p, p \wedge q \rangle\}$ and $\rho(\langle p, q \rangle) = \{\langle p, q \rangle, \langle q, p \rangle\}$ (see [8]).$\diamondsuit$

The following example of **Flags** relies on a well-known encoding of predicates as functions to Boolean (e.g. [16]). In programming, *flag* refers to a

binary value that has an assigned meaning, like, e.g., "Has a previous operation resulted in overflow or not?". Accordingly, a flag value can be set directly to two values, say *up* and *down*, the values can be *reversed*, and there is a way to retrieve the information about the state of the *flag* through a request *up?*.

Example 3.3 (Flags)**.** We present two different, but equivalent specifications of flags. The first is an 1-logic and the second is a HEL.

Flags *as a Boolean 1-logic.* Consider the hidden signature Σ_{flag}:

SORT $= \{flag, bool\}$, with "*bool*" the unique visible sort, and the following operation symbols:

$$up : flag \to flag; \qquad rev : flag \to flag;$$
$$dn : flag \to flag; \qquad up? : flag \to bool,$$

and the operation symbols for the Boolean part: $\neg, \wedge, \vee, true, false$. The Boolean biconditonal $\varphi \leftrightarrow \psi$ is an abbreviation for the compound operation $(\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi)$.

The Boolean logic of flags, $\mathcal{L}_{bflag}$, is the 1-logic with the following axioms:

$$up?(up(x)) \qquad up?(rev(x)) \leftrightarrow \neg(up?(x))$$
$$\neg up?(dn(x))$$

and includes the usual logical axioms for the classical propositional logic. There are no extra-logical rules of inference.

Flags *as a* HEL. Consider now the Flags logic as a HEL. The signature is the same as above. The equational logic of flags, $\mathcal{L}_{eflag}$, is the $\mathrm{HEL}_{\Sigma_{flag}}$ with the following extra-logical axioms:

$$up?(up(x)) \approx true \qquad up?(rev(x)) \approx \neg(up?(x))$$
$$up?(dn(x)) \approx false$$

and includes the usual logical axioms for Boolean algebra. There are no extra-logical rules of inference.

We claim that $\mathcal{L}_{bflag}$ and $\mathcal{L}_{eflag}$ are equivalent. In fact, it is easy to see that

$$\frac{\varphi_0 \leftrightarrow \varphi'_0, \cdots, \varphi_{n-1} \leftrightarrow \varphi'_{n-1}}{\psi \leftrightarrow \psi'} \text{ is a derived rule of } \mathcal{L}_{bflag}$$

iff

$$\frac{\varphi_0 \approx \varphi'_0, \cdots, \varphi_{n-1} \approx \varphi'_{n-1}}{\psi \approx \psi'} \text{ is a derived rule of } \mathcal{L}_{eflag}.$$

That is, $\mathcal{L}_{bflag}$ and $\mathcal{L}_{eflag}$ are equivalent with $\rho(x,y) = \{x \leftrightarrow y\}$ and $\tau(x) = \{\langle x, true\rangle\}$ the associated translations. $\diamondsuit$

4. Behavioral equivalence of Hidden logics

The results presented in this section involve some notions from lattice theory. So we start by recalling some concepts that will be needed in the sequel.

A *lattice* is a partially ordered set (poset) L in which any pair of elements has an infimum and a supremum. It is *complete* if an infimum and a supremum exist for any subset of L. An element $a \in L$ is compact if for all $A \subseteq L$, $a \leq \bigvee A$ implies that there is a finite $B \subseteq A$ such that $a \leq \bigvee B$. A lattice L is *algebraic* if it is complete and every element in L is a supremum of compact elements.

A map $\alpha : L_1 \to L_2$ between two lattices L_1 and L_2 is an *isomorphism* if it is a bijection such that α and α^{-1} are monotonic with respect to the underlying lattice orders.

Given a hidden logic $\mathcal{L}$, the set of theories over $\mathcal{L}$ and the set of $\mathcal{L}$-filters over an appropriate algebra with the sorted set inclusion are complete lattices; moreover, they are algebraic if the logic is finitary. The compact elements are the finitely generated theories, that is, the theories of the form $\mathrm{Cn}_{\mathcal{L}}(\Gamma)$, with Γ a globally finite sorted set.

Let $\mathrm{Fi}_{\mathcal{L}}(A)$ be the lattice of $\mathcal{L}$-filters over A. For $G \subseteq A^k$, $\mathrm{Fg}^A_{\mathcal{L}}(G)$ denotes the $\mathcal{L}$-filter generated by G. Note that since $\mathrm{Fi}_{\mathcal{L}}(A)$ is a complete lattice, $\mathrm{Fg}^A_{\mathcal{L}}(G)$ always exists and equals to $\bigcap\{F \in \mathrm{Fi}_{\mathcal{L}}(A) : G \subseteq F\}$.

Let τ be a (k,l)-translation from Σ to Σ'. We define the map $\widetilde{\tau}^A : \mathrm{Fi}_{\mathcal{L}}(A) \to \mathrm{Fi}_{\mathcal{L}'}(A)$ by $\widetilde{\tau}^A(F) = \mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F))$. When A is the term algebra we will omit explicit reference to it. In that case ($G \subseteq \mathrm{Fm}_{\mathrm{VIS}}(\mathcal{L})$), $\mathrm{Fg}^{\mathrm{Fm}_{\mathrm{VIS}}(\mathcal{L})}_{\mathcal{L}}(G) = \mathrm{Cn}_{\mathcal{L}}(G)$ — the $\mathcal{L}$-theory generated by G.

Let Σ and Σ' be two algebraically indistinguishable hidden signatures and $\mathcal{L}$ and $\mathcal{L}'$ be two hidden logics over Σ and Σ', respectively. A map $\alpha : \mathrm{Th}(\mathcal{L}) \to \mathrm{Th}(\mathcal{L}')$ is said to *commute with substitutions* if, for every substitution σ and every $T \in \mathrm{Th}(\mathcal{L})$

$$\mathrm{Cn}_{\mathcal{L}'}(\sigma(\alpha(T))) = \alpha(\mathrm{Cn}_{\mathcal{L}}(\sigma(T))).$$

Let $\tau = \langle \tau_R(\bar{x}\!:\!R) \mid R \in \mathrm{VIS}\rangle$ be a (k,l)-*translation* and A be a Σ-algebra. For any sorted set $G \subseteq A$, $\tau^A(G)$ stands for the VIS$'$-sorted set defined

for each $V \in \mathrm{VIS}'$ by $\tau^A(G)_V = \bigcup\{\bar{\tau}^A_{R,V}(\bar{a}) : R \in \mathrm{VIS}, \bar{a} \in G_R\}$, where $\bar{\tau}^A_{R,V}(\bar{a}) = \{\bar{\delta}^A(\bar{a}) : \bar{\delta} \in \bar{\tau}_{R,V}\}$.

Lemma 4.1. *Let $\mathcal{L}$, $\mathcal{L}'$ be a k- and an l-hidden logic, respectively, which are equivalent under the interpretations $\tau : \mathcal{L} \to \mathcal{L}'$ and $\rho : \mathcal{L}' \to \mathcal{L}$ and let A be an appropriate algebra. Then, for every $G \subseteq A^k_{\mathrm{VIS}}$, we have*

$$\mathrm{Fg}^A_{\mathcal{L}}(\rho^A(\tau^A(G))) = \mathrm{Fg}^A_{\mathcal{L}}(G) \tag{3}$$

Proof. From the definition of equivalence, (3) holds for singleton sets.

Let now G be any sorted subset of A^k_{VIS}. Then we have

$$\begin{aligned}
\mathrm{Fg}^A_{\mathcal{L}}(G) &= \bigvee^{\mathcal{L}}\{\mathrm{Fg}^A_{\mathcal{L}}(\{\bar{a}\}) : \bar{a} \in G\} \\
&= \bigvee^{\mathcal{L}}\{\mathrm{Fg}^A_{\mathcal{L}}(\rho^A(\tau^A(\{\bar{a}\}))) : \bar{a} \in G\} \\
&= \mathrm{Fg}^A_{\mathcal{L}}(\bigcup\{\rho^A(\tau^A(\{\bar{a}\})) : \bar{a} \in G\}) \\
&= \mathrm{Fg}^A_{\mathcal{L}}(\rho^A(\tau^A(G))),
\end{aligned}$$

as needed. □

As a consequence we have that $\mathrm{Fg}^A_{\mathcal{L}}(\rho^A(\tau^A(F))) = F$ whenever $F \in \mathrm{Fi}_{\mathcal{L}}(A)$. Symmetrically, for all $H \in \mathrm{Fi}_{\mathcal{L}'}(A)$: $\mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(\rho^A(H))) = H$. When instantiated for the term algebra, these results can be stated as:

- $\mathrm{Cn}_{\mathcal{L}}(\Gamma) = \mathrm{Cn}_{\mathcal{L}}(\rho(\tau(\Gamma)))$ for any $\Gamma \subseteq \mathrm{Fm}_{\mathrm{VIS}}(\mathcal{L})$ and
- $\mathrm{Cn}_{\mathcal{L}'}(\Gamma') = \mathrm{Cn}_{\mathcal{L}'}(\tau(\rho(\Gamma')))$ for any $\Gamma' \subseteq \mathrm{Fm}_{\mathrm{VIS}'}(\mathcal{L}')$.

The next two theorems can be formulated for the general case of Σ-algebras, however we just formulate and prove them for the special case of the formula algebra.

Lemma 4.2. *Let $\mathcal{L}$ be a k-hidden logic. For every substitution σ and any $\Gamma \subseteq \mathrm{Fm}_{\mathrm{VIS}}(\mathcal{L})$, we have*

$$\mathrm{Cn}_{\mathcal{L}}(\sigma(\mathrm{Cn}_{\mathcal{L}}(\Gamma))) = \mathrm{Cn}_{\mathcal{L}}(\sigma(\Gamma)).$$

Proof. Clearly $\Gamma \subseteq \sigma^{-1}(\mathrm{Cn}_{\mathcal{L}}(\sigma(\Gamma))$. Since $\mathrm{Th}(\mathcal{L})$ is closed under inverse substitutions[7], $\sigma^{-1}(\mathrm{Cn}_{\mathcal{L}}(\sigma(\Gamma))) \in \mathrm{Th}(\mathcal{L})$, hence $\mathrm{Cn}_{\mathcal{L}}(\Gamma) \subseteq \sigma^{-1}(\mathrm{Cn}_{\mathcal{L}}(\sigma(\Gamma)))$, i.e., $\sigma(\mathrm{Cn}_{\mathcal{L}}(\Gamma)) \subseteq \mathrm{Cn}_{\mathcal{L}}(\sigma(\Gamma))$. Therefore $\mathrm{Cn}_{\mathcal{L}}(\sigma(\mathrm{Cn}_{\mathcal{L}}(\Gamma))) \subseteq \mathrm{Cn}_{\mathcal{L}}(\sigma(\Gamma))$.

[7]Let $T \in \mathrm{Th}(\mathcal{L})$. Suppose that $\sigma^{-1}(T) \vdash \bar{\varphi}$. Since $\sigma(\sigma^{-1}(T)) \subseteq T$, $T \vdash \sigma(\bar{\varphi})$. Thus $\sigma(\bar{\varphi}) \in T$, i.e., $\bar{\varphi} \in \sigma^{-1}(T)$.

Conversely, $\sigma(\Gamma) \subseteq \sigma(\mathrm{Cn}_{\mathcal{L}}(\Gamma)) \subseteq \mathrm{Cn}_{\mathcal{L}}(\sigma(\mathrm{Cn}_{\mathcal{L}}(\Gamma)))$. Thus $\mathrm{Cn}_{\mathcal{L}}(\sigma(\Gamma)) \subseteq \mathrm{Cn}_{\mathcal{L}}(\sigma(\mathrm{Cn}_{\mathcal{L}}(\Gamma)))$. □

Lemma 4.3. *Let $\mathcal{L}$ and $\mathcal{L}'$ be a k- and an l-hidden logic, respectively, such that $\mathcal{L}$ is specifiable. For every interpretation τ from $\mathcal{L}$ to $\mathcal{L}'$ and any $\Gamma \subseteq \mathrm{Fm}_{\mathrm{VIS}}(\mathcal{L})$, we have*

$$\mathrm{Cn}_{\mathcal{L}'}(\tau(\mathrm{Cn}_{\mathcal{L}}(\Gamma))) = \mathrm{Cn}_{\mathcal{L}'}(\tau(\Gamma)).$$

Proof. For $R \in \mathrm{VIS}$, let $T_R = \tau_R^{-1}(\mathrm{Cn}_{\mathcal{L}'}(\tau(\Gamma))) := \{\bar{\varphi} \in \mathrm{Fm}_{\mathrm{VIS}}(\mathcal{L})_R : \tau_R(\bar{\varphi}) \subseteq \mathrm{Cn}_{\mathcal{L}'}(\tau(\Gamma))\}$.

Claim. $T = \langle T_R : R \in \mathrm{VIS}\rangle \in \mathrm{Th}(\mathcal{L})$.

> **Proof of the claim:** Let $T \vdash \bar{\varphi}$. Since $\mathcal{L}$ is specifiable, there are $\bar{\psi}_0, \ldots, \bar{\psi}_{n-1} \in T$ such that $\bar{\psi}_0, \ldots, \bar{\psi}_{n-1} \vdash \bar{\varphi}$. But τ is an interpretation, thus $\tau(\bar{\psi}_0), \ldots, \tau(\bar{\psi}_{n-1}) \vdash \tau(\bar{\varphi})$. As $\tau(\bar{\psi}_i) \subseteq \mathrm{Cn}_{\mathcal{L}'}(\tau(\Gamma))$, $i = 0, \ldots, n-1$, $\tau(\bar{\varphi}) \subseteq \mathrm{Cn}_{\mathcal{L}'}(\tau(\Gamma))$. Hence, $\bar{\varphi} \in T$.

Moreover, it is clear that $\Gamma \subseteq T$. Then, from the claim, $\mathrm{Cn}_{\mathcal{L}}(\Gamma) \subseteq T$, and thus $\tau(\mathrm{Cn}_{\mathcal{L}}(\Gamma)) \subseteq \mathrm{Cn}_{\mathcal{L}'}(\tau(\Gamma))$. Therefore $\mathrm{Cn}_{\mathcal{L}'}(\tau(\mathrm{Cn}_{\mathcal{L}}(\Gamma))) \subseteq \mathrm{Cn}_{\mathcal{L}'}(\tau(\Gamma))$.

To see the converse inclusion, note that

$$\tau(\Gamma) \subseteq \tau(\mathrm{Cn}_{\mathcal{L}}(\Gamma)) \subseteq \mathrm{Cn}_{\mathcal{L}'}(\tau(\mathrm{Cn}_{\mathcal{L}}(\Gamma))).$$

Therefore $\mathrm{Cn}_{\mathcal{L}'}(\tau(\Gamma)) \subseteq \mathrm{Cn}_{\mathcal{L}'}(\tau(\mathrm{Cn}_{\mathcal{L}}(\Gamma)))$. □

As a corollary we have:

Corollary 4.4. *For any subset $\{T_i : i \in I\} \subseteq \mathrm{Th}(\mathcal{L})$, $\widetilde{\tau}(\bigvee^{\mathcal{L}}_{i\in I} T_i) = \bigvee^{\mathcal{L}'}_{i\in I} \widetilde{\tau}(T_i)$.*

Next we will present the main theorem of the paper. The one-sorted variant of it, for k-deductive systems, was proved in [8]. In the multi-sorted setting, the implication from (ii) to (i) requires more attention when we need to define a substitution and for that we have to impose the extra condition that the logics have to be standard.

Theorem 4.5. *Let $\mathcal{L}$ and $\mathcal{L}'$ be a k- and an l-hidden standard, specifiable logic over algebraically indistinguishable hidden signatures Σ and Σ', respectively. The following conditions are equivalent:*

(i) *$\mathcal{L}$ and $\mathcal{L}'$ are equivalent;*

(ii) *there is an isomorphism from* $\mathrm{Th}(\mathcal{L})$ *to* $\mathrm{Th}(\mathcal{L}')$ *that commutes with substitutions.*

Proof. Let $\mathcal{L}$, $\mathcal{L}'$ be a k- and an l-hidden logic, respectively, which are equivalent under the interpretations $\tau : \mathcal{L} \to \mathcal{L}'$ and $\rho : \mathcal{L}' \to \mathcal{L}$. Recall that, by definition,

$$\widetilde{\tau}(T) = \mathrm{Cn}_{\mathcal{L}'}(\tau(T)), \quad \widetilde{\rho}(T) = \mathrm{Cn}_{\mathcal{L}}(\rho(T)).$$

Claim. $\widetilde{\tau} : \mathrm{Th}(\mathcal{L}) \to \mathrm{Th}(\mathcal{L}')$ and $\widetilde{\rho} : \mathrm{Th}(\mathcal{L}') \to \mathrm{Th}(\mathcal{L})$ both commute with substitutions.

Proof of the claim: We give the proof for $\widetilde{\tau}$, for the other map it is completely similar.

Let σ be a substitution. Then for each $R \in \mathrm{VIS}$ and any $V \in \mathrm{VIS}'$

$$\begin{aligned}
\tau_{R,V}(\sigma(\bar{x}{:}R)) &= \tau_{R,V}(\langle \sigma(x_0{:}R), \ldots, \sigma(x_{k-1}{:}R)\rangle) \\
&= \{\delta(\sigma(x_0{:}R), \ldots, \sigma(x_{k-1}{:}R)) : \delta \in \tau_{R,V}(\bar{x})\} \\
&= \{\sigma(\delta(x_0{:}R, \ldots, x_{k-1}{:}R)) : \delta \in \tau_{R,V}(\bar{x})\} \\
&= \sigma(\tau_{R,V}(\bar{x})).
\end{aligned}$$

Thus $\tau_R(\sigma(\bar{x} : R)) = \sigma(\tau_R(\bar{x}))$.

For a $T \in \mathrm{Th}(\mathcal{L})$ we have

$$\begin{aligned}
\widetilde{\tau}(\mathrm{Cn}_{\mathcal{L}}(\sigma(T))) &= \mathrm{Cn}_{\mathcal{L}'}(\tau(\mathrm{Cn}_{\mathcal{L}}(\sigma(T)))) \\
&= \mathrm{Cn}_{\mathcal{L}'}(\tau(\sigma(T))) && \text{by Lemma 4.3} \\
&= \mathrm{Cn}_{\mathcal{L}'}(\sigma(\tau(T))) \\
&= \mathrm{Cn}_{\mathcal{L}'}(\sigma(\mathrm{Cn}_{\mathcal{L}'}(\tau(T)))) && \text{by Lemma 4.2} \\
&= \mathrm{Cn}_{\mathcal{L}'}(\sigma(\widetilde{\tau}(T)))
\end{aligned}$$

Finally we show that $\widetilde{\tau}$ and $\widetilde{\rho}$ are mutually inverse lattice isomorphisms.

On the one hand, from Corollary 4.4, $\widetilde{\tau}$ and $\widetilde{\rho}$ are joint-semilattice homomorphisms. On the other hand, as it was pointed out after Lemma 4.1, we have

- $\mathrm{Cn}_{\mathcal{L}}(\Gamma) = \mathrm{Cn}_{\mathcal{L}}(\rho(\tau(\Gamma))$ for any $\Gamma \subseteq \mathrm{Fm}_{\mathrm{VIS}}(\mathcal{L})$ and
- $\mathrm{Cn}_{\mathcal{L}'}(\Gamma') = \mathrm{Cn}_{\mathcal{L}'}(\tau(\rho(\Gamma'))$ for any $\Gamma' \subseteq \mathrm{Fm}_{\mathrm{VIS}'}(\mathcal{L}')$

These two equalities can be used to show that the two maps are inverses of each other. Actually, we have, for all $T \in \mathrm{Th}(\mathcal{L})$,

$$\begin{array}{lll} \widetilde{\rho}(\widetilde{\tau}(T)) & = \widetilde{\rho}(\mathrm{Cn}_{\mathcal{L}'}(\tau(T)) & \\ & = \mathrm{Cn}_{\mathcal{L}}(\rho(\tau(T)) & \text{by Lemma 4.3} \\ & = \mathrm{Cn}_{\mathcal{L}}(T) & \text{by Lemma 4.1} \\ & = T. & \end{array}$$

Similarly, we can show that $\widetilde{\tau}(\widetilde{\rho}(T)) = T$, for all $T \in \mathrm{Th}(\mathcal{L}')$.

To prove the implication from (ii) to (i), let $f : \mathrm{Th}(\mathcal{L}) \to \mathrm{Th}(\mathcal{L}')$ be a lattice isomorphism that commutes with substitutions.

For each visible sort $V \in \mathrm{VIS}$, let $\bar{x} = \langle x_0{:}V, \ldots, x_{k-1}{:}V \rangle$ be a fixed k-variable of a visible sort V, with x_i's distinct variables. Let T' be the image of the theory generated by $\bar{x}$, i.e, $T' = f(\mathrm{Cn}_{\mathcal{L}}(\{\bar{x}\}))$. Since $\mathrm{Cn}_{\mathcal{L}}(\{\bar{x}\})$ is finitely generated, it is compact in $\mathrm{Th}(\mathcal{L})$. Thus T' is compact in $\mathrm{Th}(\mathcal{L}')$. Therefore there is a finite number of formulas

$$\eta^i(x_0{:}V, \ldots, x_{k-1}{:}V, y_0{:}Q_0, \ldots, y_{m-1}{:}Q_{m-1}){:}R_i,$$

where $i = 0, \ldots, n-1$ and $R_i \in \mathrm{VIS}'$, such that

$$T' = \mathrm{Cn}_{\mathcal{L}'}\{\eta^i(x_0{:}V, \ldots, x_{k-1}{:}V, y_0{:}Q_0, \ldots, y_{m-1}{:}Q_{m-1}) : i < n\},$$

where $y_0{:}Q_0, \ldots, y_{m-1}{:}Q_{m-1}$ is the list of all variables, distinct from the $x_i{:}V$, that occur in at least one of the η^i. Let σ be any substitution such that $\sigma(x_i) = x_i$ and $\sigma(y_j) = t_j$, where t_j is a ground term of sort Q_j, for all $i < k$ and $j < m$ (note that such t_j exists since $\mathcal{L}'$ is standard). Then

$$\begin{array}{lll} T' = f(\mathrm{Cn}_{\mathcal{L}}\{\sigma(\bar{x})\}) & & \sigma(\bar{x}) = \bar{x} \\ = f(\mathrm{Cn}_{\mathcal{L}}(\sigma(\mathrm{Cn}_{\mathcal{L}}\{\bar{x}\})) & & \text{by Lemma 4.2} \\ = \mathrm{Cn}_{\mathcal{L}'}(\sigma(f(\mathrm{Cn}_{\mathcal{L}}\{\bar{x}\}))) & & \text{f commutes with substitutions} \\ = \mathrm{Cn}_{\mathcal{L}'}(\sigma((\mathrm{Cn}_{\mathcal{L}'}\{\eta^i(x_0{:}V, \ldots, x_{k-1}{:}V, y_0{:}Q_0, \ldots, y_{m-1}{:}Q_{m-1}){:}i < n\}))) & & \\ = \mathrm{Cn}_{\mathcal{L}'}(\sigma(\{\eta^i(x_0{:}V, \ldots, x_{k-1}{:}V, y_0{:}Q_0, \ldots, y_{m-1}{:}Q_{m-1}) : i < n\})) & & \\ & & \text{by Lemma 4.2} \\ = \mathrm{Cn}_{\mathcal{L}'}\{\eta^i(x_0{:}V, \ldots, x_{k-1}{:}V, t_0{:}Q_0, \ldots, t_{m-1}{:}Q_{m-1}) : i < n\}. & & \end{array}$$

Let $\tau^i = \eta^i(x_0{:}V, \ldots, x_{k-1}{:}V, t_0{:}Q_0, \ldots, t_{m-1}{:}Q_{m-1}){:}R_i, i < n$. Let us define $\tau_{V,R_i}(\bar{x}{:}V) = \{\tau^i(\bar{x}{:}V) : i < n\}$ and $\tau_{V,R}(\bar{x}{:}V) = \varnothing$, for $R \neq R_i$. Finally, we define $\tau = \langle \tau_V(\bar{x}{:}V) : V \in \mathrm{VIS} \rangle$.

Let $\bar{\varphi}{:}V \in \mathrm{Fm}^k_{\mathrm{VIS}}(\mathcal{L})$ and σ' be a substitution such that $\sigma'(\bar{x}) = \bar{\varphi}$.

$$\begin{aligned}
f(\mathrm{Cn}_{\mathcal{L}}(\{\bar{\varphi}\})) &= f(\mathrm{Cn}_{\mathcal{L}}(\sigma'(\bar{x})) && \text{since } \sigma'(\bar{x}) = \bar{\varphi} \\
&= f(\mathrm{Cn}_{\mathcal{L}}(\sigma'(\mathrm{Cn}_{\mathcal{L}}\{\bar{x}\})) && \text{by Lemma 4.2} \\
&= \mathrm{Cn}_{\mathcal{L}'}\sigma'(f(\mathrm{Cn}_{\mathcal{L}}\{\bar{x}\})) && f \text{ commutes with substitutions} \\
&= \mathrm{Cn}_{\mathcal{L}'}\sigma'(\{\tau^i(\bar{x}) : i < n\}) \\
&= \mathrm{Cn}_{\mathcal{L}'}\{\tau^i(\bar{\varphi}) : i < n\} && \text{by Lemma 4.2} \\
&= \mathrm{Cn}_{\mathcal{L}'}(\tau(\bar{\varphi})).
\end{aligned}$$

This can be extended to sets of formulas. Let $\Gamma \subseteq \mathrm{Fm}_{\mathrm{VIS}}(\mathcal{L})$,

$$\begin{aligned}
f(\mathrm{Cn}_{\mathcal{L}}(\Gamma)) &= f(\bigvee^{\mathcal{L}}\{\mathrm{Cn}_{\mathcal{L}}(\{\bar{\varphi}\}) : \bar{\varphi} \in \Gamma\}) \\
&= \bigvee^{\mathcal{L}'}\{f(\mathrm{Cn}_{\mathcal{L}}(\{\bar{\varphi}\})) : \bar{\varphi} \in \Gamma\} \\
&= \bigvee^{\mathcal{L}'}\{\mathrm{Cn}_{\mathcal{L}'}(\tau(\bar{\varphi})) : \bar{\varphi} \in \Gamma\} \\
&= \mathrm{Cn}_{\mathcal{L}'}(\tau(\Gamma)).
\end{aligned}$$

Now, we claim that τ is an interpretation from $\mathcal{L}$ to $\mathcal{L}'$. Indeed, for any $\Gamma \cup \{\bar{\varphi}\} \subseteq \mathrm{Fm}^k_{\mathrm{VIS}}(\mathcal{L})$, we have

$$\begin{aligned}
\Gamma \vdash_{\mathcal{L}} \bar{\varphi} &\iff \mathrm{Cn}_{\mathcal{L}}\{\bar{\varphi}\} \subseteq \mathrm{Cn}_{\mathcal{L}}(\Gamma) \\
&\iff f(\mathrm{Cn}_{\mathcal{L}}\{\bar{\varphi}\}) \subseteq f(\mathrm{Cn}_{\mathcal{L}}(\Gamma)) \\
&\iff \mathrm{Cn}_{\mathcal{L}'}(\tau(\bar{\varphi})) \subseteq \mathrm{Cn}_{\mathcal{L}'}(\tau(\Gamma)) \\
&\iff \tau(\Gamma) \vdash_{\mathcal{L}'} \tau(\bar{\varphi})
\end{aligned}$$

Similarly we can show that there is an interpretation ρ from $\mathcal{L}'$ to $\mathcal{L}$ such that $\mathrm{Cn}_{\mathcal{L}}(\rho(\Gamma')) = f^{-1}(\mathrm{Cn}_{\mathcal{L}'}(\Gamma'))$, for all $\Gamma' \subseteq \mathrm{Fm}_{\mathrm{VIS}'}(\mathcal{L}')$.

Finally, we show that τ and ρ are inverses of each other. Let $\bar{x}$ be a k-variable of sort $V \in \mathrm{VIS}$. Then $\mathrm{Cn}_{\mathcal{L}'}(\tau(\rho(\bar{x}))) = f(\mathrm{Cn}_{\mathcal{L}}(\rho(\bar{x}))) = f(f^{-1}(\mathrm{Cn}_{\mathcal{L}'}(\bar{x}))) = \mathrm{Cn}_{\mathcal{L}'}(\bar{x})$, i.e., $\bar{y}{:}V' \dashv\vdash_{\mathcal{L}'} \tau(\rho(\bar{y}{:}V'))$. Similarly, we can show that $\bar{x}{:}V \dashv\vdash_{\mathcal{L}} \rho(\tau(\bar{x}{:}V))$. □

Similar to condition (i) of this theorem, a semantic necessary and sufficient condition for two hidden logics to be behaviorally equivalent can be formulated. Instead of lattice of theories, it uses the lattice of logical filters of an arbitrary algebra over the underlying signature (see [8] for the one-sorted case).

The *Leibniz operator* over $\mathcal{L}$ is the map $\Omega^{\mathcal{L}}_A : \mathrm{Fi}_{\mathcal{L}}(A) \to \mathrm{Con}(A)$ defined by $\Omega^{\mathcal{L}}_A(F) = \Omega_A(F)$ (explicit reference to the hidden k-logic $\mathcal{L}$ is usually omitted, and if the algebra A is clear from the context we simply denote the Leibniz operator by Ω).

The following theorem provides a necessary condition for the equivalence of two hidden logics. The version for k-deductive systems can be found in [8].

Theorem 4.6. *Let $\mathcal{L}$, $\mathcal{L}'$ be a k- and an l-hidden logic, respectively, which are equivalent under the interpretations $\tau : \mathcal{L} \to \mathcal{L}'$ and $\rho : \mathcal{L}' \to \mathcal{L}$. Then, for any k-data structure $\langle A, F\rangle \in \mathrm{Mod}(\mathcal{L})$, we have*

$$\Omega(F) = \Omega(\mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F))). \tag{4}$$

Proof. First we show that for every $F \in \mathrm{Fi}_{\mathcal{L}}(A)$ and $\bar{a} \in A^l_V$,

$$\bar{a} \in \mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F))_V \text{ iff } \rho^A(\{\bar{a}\}) \subseteq F \tag{5}$$

Indeed, let $\bar{a} \in \mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F))_V$, then $\rho^A(\{\bar{a}\}) \subseteq \rho^A(\mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F))) \subseteq F$. The last inclusion holds since, similarly to Lemma 4.3, $\mathrm{Fg}^A_{\mathcal{L}}(\rho^A(\mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F)))) = \mathrm{Fg}^A_{\mathcal{L}}(\rho^A(\tau^A(F))) = F$.

The reverse implication is a consequence of Lemma 4.1. Suppose that $\rho^A(\{\bar{a}\}) \subseteq F$. Hence $\tau^A(\rho^A(\{\bar{a}\})) \subseteq \tau^A(F) \subseteq \mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F))$. Finally by Lemma 4.1, (5) holds.

Furthermore, we have:

Claim. For any $\theta \in \mathrm{Con}(A)$ and any $F \subseteq A$, θ is compatible with F iff θ is compatible with $\mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F))$.

Indeed, let $\bar{a} \in \mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F))_V$ and $\bar{b} \in A^l_V$ such that $a_i \theta_V b_i$, $i = 1, \ldots, l$, for some $V \in \mathrm{VIS}'$. Then

$$\begin{aligned}
&\bar{a} \in \mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F))_V \\
&\iff \rho^A(\{\bar{a}\}) \subseteq F \\
&\iff (\bar{\rho}^i)^A(\bar{a}) \in F_R, \text{ for all } R \in \mathrm{VIS}, \bar{\rho}^i \in \rho_{V,R} \\
&\iff (\bar{\rho}^i)^A(\bar{b}) \in F_R, \text{ for all } R \in \mathrm{VIS}, \bar{\rho}^i \in \rho_{V,R} \\
&\iff \rho^A(\bar{b}) \subseteq F \\
&\iff \bar{b} \in \mathrm{Fg}^A_{\mathcal{L}'}(\tau^A(F))_V,
\end{aligned}$$

For the reverse implication, the argument is similar. □

Conclusion

This work is part of a wider approach of applying abstract algebraic logic methods and tools to the study of the specification and semantics of object-oriented software systems [33, 40, 34, 35, 36, 2]. The general idea consists in considering a generalization of the concept of deductive system to specify heterogeneous computer systems and to supply a meaningful and useful behavioral equivalence relation in this new framework. This generalization combines the multi-sorted setting with the dichotomy visible vs. hidden data types. It is achieved by means of the central notion of Leibniz congruence taken from Abstract Algebraic Logic. As a side benefit, we obtain a *matrix* semantics as a well behaved semantics for OO systems.

In this paper we discuss a new relation of equivalence between specifications, introduced in [2]. Besides some illustrative examples, we discuss the relationship between the behavior of equivalent hidden logics. This new approach opens some interesting questions, such as: Is it possible to characterize the logics that are equivalent to applied equational logics, which in the context of ordinary deductive systems are called algebraizable logics.

We have also presented two sorted calculi in order to show the diversity of situations that can be captured by this notion. In fact, although our theory unifies several approaches to behavioral specification systems, many other examples outside of the CS can be studied in the context of a general notion of hidden logic with the tools that are based on properties of the Leibniz congruence. The example concerning quasi-orderings can be of interest in CS. It seems that it can be modified to specify state transition machines, with structure on states, in the same lines as Diaconescu did in [17]. More specifically, we may split the set of sorts in two: one for data elements and the other for transitions.

Acknowledgments. This work was supported in part by the Portuguese Foundation for Science and Technology FCT through CIDMA, within project `UID/MAT/04106/2013`. The second author also acknowledges the financial assistance by EU FP7 Marie Curie `PIRSES- GA-2012-318986` project GeTFun and the project `FFI2013-47126-P` by the Spanish Ministry of Research.

References

[1] S. Babenyshev and M. A. Martins. Admissible equivalence systems. *Bull. Sect. Logic Univ. Łódź*, 39(1–2): 17–33, 2010.

[2] S. Babenyshev and M. A. Martins. Deduction detachment theorem in hidden k-logics. *J. Log. Comput.* 24(1): 233-255, 2014.

[3] M. Bidoit and R. Hennicker. *Behavioural theories and the proof of behavioural properties*, Theor. Comput. Sci. 165(1): 3–55, 1996.

[4] M. Bidoit and R. Hennicker. Observer complete definitions are behaviourally coherent. In *Proc. OBJ/CafeOBJ/Maude Workshop at Formal Methods'99, Toulouse, France, Sep.*, pages 83–94. 1999. Preliminary long version available as Report LSV 99-4.

[5] M. Bidoit, R. Hennicker, and M. Wirsing. Behavioural and abstractor specifications. *Sci. Comput. Program.*, 25(2-3):149–186, 1995.

[6] M. Bidoit, D. Sannella and A. Tarlecki. *Observational interpretation of CASL specifications*, Mathematical Structures in Computer Science, Cambridge University Press, 18(2): 325–371, 2008.

[7] W. J. Blok and D. Pigozzi. *Algebraizable Logics*, Memoirs of the American Mathematical Society, Vol. 77, No. 396, 1989.

[8] W. J. Blok and D. Pigozzi. Abstract algebraic logic and the deduction theorem. Preprint. Available at `http://www.math.iastate.edu/dpigozzi/papers/ aaldedth.pdf`, 2001.

[9] A. Bouhoula and M. Rusinowitch. *Observational proofs by rewriting*, Theor. Comput. Sci., 275(1–2): 675–698, 2002.

[10] C. Caleiro, R. Gonçalves and M. A. Martins. *Behavioral Algebraization of Logics*, Studia Logica, 91(1): 63–111, 2009.

[11] C. Caleiro, R. Gonçalves. *Equipollent logical systems*, In Beziau, Jean-Yves (ed.), Logica universalis. Towards a general theory of logic. Basel: Birkhäuser. (2007) 97–109.

[12] W.A. Carnielli, M.E. Coniglio and I.M.L. D'Ottaviano. New dimensions on translations between logics. *Logica Universalis*, 3(1): 1–18, 2009.

[13] J. Czelakowski. *Protoalgebraic logics*, Trends in Logic–Studia Logica Library. 10. Dordrecht: Kluwer Academic Publishers, 2001.

[14] J. Czelakowski and D. Pigozzi. Fregean logics. *Ann. Pure Appl. Logic*, 127(1–3): 17–76, 2004.

[15] L. Descalço and M. A. Martins. On the injectivity of the Leibniz operator. *Bull. Sect. of Logic*, 34(4): 203–211, 2005.

[16] R. Diaconescu. Quasi-Boolean encodings and conditionals in algebraic specifications. *J. of Logic and Algebraic Programming*, 79(2): 174–188, 2010.

[17] R. Diaconescu. Coinduction for preordered algebra. *Inf. Comput.*, 209(2): 108–117, 2011.

[18] R. Diaconescu and K. Futatsugi. CafeOBJ report: The language, proof techniques, and methodologies for object-oriented algebraic specification. In AMAST series in Computing, editor, *World Scientific*, volume 6, 1998.

[19] R. Diaconescu and K. Futatsugi. Behavioural coherence in object-oriented algebraic specification. *J. UCS*, 6(1):74–96, 2000.

[20] J. Fiadeiro and A. Sernadas. Structuring theories on consequence. In *Recent trends in data type specification. Specification of abstract data types, Sel. Pap. 5th Workshop, Gullane/UK 1987, Lect. Notes Comput. Sci. 332*, pages 44–72. 1988.

[21] J. M. Font, R. Jansana and D. Pigozzi. A survey of abstract algebraic logic. *Stud. Log.* 74(1–2): 13–97, 2003.

[22] J. Goguen, K. Lin, and G. Roşu. Conditional circular coinductive rewriting with case analysis. In *16th International Workshop, WADT 2002*, volume 2755 of *Lecture Notes in Computer Science*, pages 216–232, Frauenchiemsee, Germany, September 2002.

[23] E. Goriac, D. Lucanu, and G. Roşu. Automating coinduction with case analysis. In *Formal Methods and Software Engineering - 12th International Conference on Formal Engineering Methods, ICFEM 2010, Shanghai, China, November 17-19, 2010. Proceedings*, pages 220–236, 2010.

[24] J. Goguen and G. Malcolm. *Hidden coinduction: Behavioural correctness proofs for objects*, Math. Struct. Comput. Sci. 9(3): 287–319, 1999.

[25] J. Goguen and G. Malcolm. *A hidden agenda*, Theor. Comput. Sci. 245(1): 55–101, 2000.

[26] J. Goguen. Types as theories. In *Topology and category theory in computer science*, Oxford Sci. Publ., pages 357–390. Oxford Univ. Press, New York, 1989.

[27] J. Goguen and G. Roşu. Hiding more of hidden algebra. In *Wing, Jeannette M. et al. (ed.), FM '99. Formal methods. World congress on Formal methods in the development of computing systems. Toulouse, France, September 20-24. Proceedings.* 1999.

[28] R. Hennicker and M. Bidoit. Observational logic. In *Proc. AMAST '98, 7th International Conference on Algebraic Methodology and Software Technology. Lecture Notes in Computer Science, Berlin: Springer*, pages 263–277. 1999.

[29] R. Hennicker. Structural specifications with behavioural operators: semantics, proof methods and applications, Habilitationsschrift, 1997.

[30] K. Lin J. Goguen and G. Roşu. Circular coinductive rewriting. In *Proceedings, Automated Software Engineering '00 (Grenoble France), IEEE Press*, pages 123–131. September 2000.

[31] D. Lucanu, O. Gheorghies, and A. Apetrei. Bisimulation and hidden algebra. In *Jacobs, Bart (ed.) et al., CMCS '99. Proceedings of the 2nd workshop on coalgebraic methods in computer science. A satellite event to the European joint conferences on Theory and practice of software, ETAPS '99. Amsterdam, the Netherlands, March 20-21, 1999. Amsterdam: Elsevier, Electronic Notes in Theoretical Computer Science. 19, electronic paper No.13* . 1999.

[32] G. Malcolm. Behavioural equivalence, bisimulation, and minimal realisation. In *Magne Haveraaen and Olaf Owe and Ole-Johan Dahl (eds.), Recent Trends in Data Type Specifications. 11th Workshop on Specification of Abstract Data Types*. Springer-Verlag Lecture Notes in Computer Science, 1996.

[33] M. A. Martins. *Behavioral reasoning in generalized hidden logics*. PhD thesis, University of Lisbon, 2004.

[34] M. A. Martins. Behavioral institutions and refinements in generalized hidden logics, *J. Univers. Comput. Sci.*, 12(8): 1020–1049, 2006.

[35] M. A. Martins. Closure properties for the class of behavioral models. *Theor. Comput. Sci.*, 379(1–2): 53–83, 2007.

[36] M. A. Martins. On the behavioral equivalence between k-data structures. *Computer Journal*, 51(2): 181–191, 2008.

[37] M. A. Martins, A. Madeira, and L. S. Barbosa. Refinement by interpretation. In *7th IEEE International Conference on Software Engineering and Formal Methods (SEFM'09)*. IEEE Computer Society Press, 2009.

[38] M. A. Martins, A. Madeira, and L. S. Barbosa. A coalgebraic perspective on logical interpretations, *Studia Logica*, 101(4): 783–825, 2013.

[39] M. A. Martins, A. Madeira, and L. S. Barbosa. The role of logical interpretations in program development, *Logical Methods in Computer Science*, 10(1): 1–30, 2014

[40] M. A. Martins and D. Pigozzi. Behavioural reasoning for conditonal equations. *Math. Struct. Comput. Sci.*, 17(5): 1075–1113, 2007.

[41] K. Meinke and J. V. Tucker. *Universal algebra*, Handb. Log. Comput. Sci., vol. 1, Oxford Univ. Press, New York, 1992, pp. 189–411.

[42] J. Meseguer. General logics. In *Logic colloq. '87, Proc. Colloq., Granada/Spain 1987, Stud. Logic Found. Math. 129*, pages 275–329. 1989.

[43] T. Mossakowski, R. Diaconescu, and A. Tarlecki. What is a logic translation? *Logica Universalis*, 3(1): 95–124, 2009.

[44] B. Moore and G. Roşu. Program verification by coinduction. Technical Report http://hdl.handle.net/2142/73177, University of Illinois, February 2015.

[45] P. Padawitz, *Swinging types=functions+relations+transition systems.*, Theor. Comput. Sci. **243** (2000), no. 1-2, 93–165.

[46] P.Padawitz, *Swinging data types: The dialectic between actions and constructors*, Available at `https://fldit-www.cs.uni-dortmund.de/~peter/`.

[47] K. Pałasińska and D. Pigozzi. Partially Ordered Varieties and Quasivarieties. Revised notes of lectures on joint work with Katarzyna Palasinska given at the CAUL, Lisbon in September of 2003, and at the Universidad Catolica, Santiago in November of 2003. *Manuscript*. Available at `http://orion.math.iastate.edu/dpigozzi/notes/santiago_notes.pdf`.

[48] D. Pigozzi. Equality-test and if-then-else algebras: Axiomatization and specification. *SIAM J. Comput*, 20(4): 766–805, 1991.

[49] D. Pigozzi. Abstract algebraic logic. *Encyclopedia of Mathematics, Supplement III* (*M. Hazewinkel, ed.*), *Kluwer Academic Publishers, Dordrecht*, pages 2–13, 2001.

[50] J.G. Raftery. Order algebraizable logics. *Annals of Pure and Applied Logic*, 164(3): 251–283, 2013.

[51] H. Reichel. Behavioural validity of conditional equations in abstract data types. In *Contributions to general algebra 3, Proc. Conf., Vienna 1984*, pages 301–324. 1985.

[52] H. Reichel. An approach to object semantics based on terminal coalgebras. *Math. Struct. Comput. Sci.*, 5(2):129–152, 1995.

[53] G. Roşu and J. Goguen. Circular coinduction. Technical report. Available at `http://www-cse.ucsd.edu/users/goguen/pps/ccoind.ps`.

[54] G. Roşu and D. Lucanu. Circular coinduction: A proof theoretical foundation. In *Proceedings of the 3rd International Conference on Algebra and Coalgebra in Computer Science (CALCO'09)*, volume 5728 of *Lecture Notes in Computer Science*, pages 127–144, 2009.

[55] G. Roşu. *Hidden Logic*. PhD thesis, University of California, San Diego, 2000.

[56] D. Sannella and A. Tarlecki. Foundations of Algebraic Specification and Formal Software Development. EATCS Monographs in Theoretical Computer Science. Springer, 2012.