Constructor-based Observational Logic [⋆]

Michel Bidoit [a] and Rolf Hennicker [b]

[a] *Laboratoire Spécification et Vérification (LSV), CNRS & ENS de Cachan, France*

[b] *Institut für Informatik, Ludwig-Maximilians-Universität München, Germany*

Abstract

This paper focuses on the integration of reachability and observability concepts within an algebraic, institution-based framework. In the first part of this work, we develop the essential ingredients that are needed to define the constructor-based observational logic institution, called COL, which takes into account both the generation- and observation-oriented aspects of software systems. The underlying paradigm of our approach is that the semantics of a specification should be as loose as possible to capture all its correct realizations. We also consider the "black box" semantics of a specification which is useful to study the behavioral properties a user can observe when he/she is experimenting with the system.

In the second part of this work, we develop proof techniques for structured COL-specifications. For this purpose we introduce an institution encoding from the COL institution to the institution of many-sorted first-order logic with equality and sort-generation constraints. Using this institution encoding, we can then reduce proofs of consequences of structured specifications built over COL to proofs of consequences of structured specifications written in a simple subset of the algebraic specification language Casl. This means, in particular, that any inductive theorem prover, such as e.g. the Larch Prover or PVS, can be used to prove theorems over structured COL-specifications.

Key words: Algebraic specification, observability, reachability, institution, proof techniques.

[⋆] This work was partially supported by the German DFG-project InopSys and by the German BMBF-project GLOWA-Danube. Preliminary partial results of this study have been published in [6].

Email addresses: `bidoit@lsv.ens-cachan.fr` (Michel Bidoit), `hennicke@informatik.uni-muenchen.de` (Rolf Hennicker).

URLs: `www.lsv.ens-cachan.fr/~bidoit/` (Michel Bidoit), `www.pst.informatik.uni-muenchen.de/~hennicke/` (Rolf Hennicker).

Contents

1 Introduction 3

1.1 The COL Institution 4

1.2 Black Box Semantics of COL-Specifications 5

1.3 Proving Consequences of Structured COL-specifications 6

1.4 Related Frameworks 8

2 Basic Notions 9

2.1 Algebraic Preliminaries 9

2.2 Institutions 9

2.3 Structured Specifications 11

PART I — The Constructor-based Observational Logic COL 12

3 COL-Signatures and COL-Algebras 12

3.1 COL-Signatures, Generated Parts and Observational Equalities 12

3.2 COL-Algebras and Black Box Functors 17

3.3 COL-Satisfaction Relation and Basic COL-Specifications 22

4 The Constructor-based Observational Logic Institution COL 26

4.1 COL-Signature Morphisms 27

4.2 The COL Institution 28

4.3 Structured COL-Specifications 32

4.4 Amalgamation and Interpolation 33

PART II — Proving Consequences of Structured COL-Specifications 39

5 Institution Encodings 41

6 Encoding COL into IBB 45

7 Encoding IBB into CFOLEq 47

8 Conclusion and Future Work 56

Acknowledgements 57

References 57

1 Introduction

The purpose of any specification formalism is to specify programs or, more generally, software systems. Each specification should therefore determine a class of programs that correctly realize the specified requirements. This idea is directly reflected by the loose semantics approach to specifications which, in the spirit of Hoare [22], is based on the following general assumption:

The semantics of a specification SP consists
of all correct realizations of SP.

In the algebraic approach to software development, programs are modelled by (many-sorted) algebras and the properties of these algebras are specified by logical axioms provided by some specification SP. Then a program is a correct realization if it is a model of SP. In practice, additional concepts are needed to model reachability and observability aspects which play an important role in software development.

Reachability concepts focus on the specification of generation principles usually presented by a set of constructors. Several algebraic specification languages incorporate features to express reachability like, for instance, the Larch shared language [19] or CASL [1,10]. The standard interpretation of reachability is to admit as models of a specification only those algebras which are reachable w.r.t. the given constructors. Many examples show, however, that this view is too restrictive since a correct realization of a specification may contain non-reachable (junk) elements (e.g., one may want to realize natural numbers by integers). Hence, according to the general assumption from above, we are interested in a more loose interpretation of reachability which will be provided by the COL framework following the ideas of the constructor-based logic institution introduced in [8].

Observability concepts are used to specify the desired observable properties of a program (see, e.g., [33,35,29,31,13]). Particular institutions which formalize the syntactic and semantic aspects of observability were introduced in [15] (hidden algebra) and in [21] (observational logic). These approaches take into account our general assumption from above in the sense that any program which satisfies the observable behavior prescribed by a specification SP is considered as a correct realization of SP. Thus even realizations which do not literally satisfy the axioms of a given specification SP are captured by the semantics of SP as long as they have the desired observable behavior.

The aim of this paper is, first, to *integrate* our treatments of reachability and observability (so far only provided separately in constructor-based logic and in observational logic) in a common, powerful institution, called COL (constructor-based observational logic), such that the semantics of a structured COL-specification $\mathrm{SP}_{\mathrm{COL}}$ contains all correct realizations of $\mathrm{SP}_{\mathrm{COL}}$, both from the reachability *and* from the observability point of view. Then, in the second part of this paper, we will develop proof techniques for reasoning about behavioral consequences of structured COL-specifications.

1.1 The COL *Institution*

The fundamental assumption underlying the development of the COL institution is that a COL-signature Σ_{COL} contains a distinguished set of constructors and a distinguished set of observers. Intuitively the constructors specify those elements which are of interest from the user's point of view. They determine the Σ_{COL}-generated part of an algebra. The observers determine a set of observable contexts which represent the observable experiments a user can perform to examine hidden states. Two states are considered to be observationally equal if they cannot be distinguished by these observable experiments. For the precise definition of the observational equality we also take into account the presence of constructors since the only relevant experiments are those where the parameters (if any) are values of interest, i.e., belong to the Σ_{COL}-generated part of the algebra under consideration.

Using the notion of a COL-signature, we then define COL-algebras as those structures whose operations are compatible with the Σ_{COL}-generated part and with the observational equality. These conditions are formally expressed by so-called reachability and observability constraints. In contrast to standard approaches to reachability, the reachability constraint does not require that all elements of an algebra are generated by the constructors (i.e., that the algebra is reachable), because this would be too restrictive to capture all correct realizations (as discussed above). We rather require that, up to observational equality, the Σ_{COL}-generated part of an algebra is preserved by the non-constructor operations, i.e., non-constructor operations may lead out of the generated part as long as they produce an element which is indistinguishable from an element inside the generated part. The closure of the Σ_{COL}-generated part of an algebra A is called the Σ_{COL}-generated subalgebra of A. The observability constraint requires that, on the Σ_{COL}-generated subalgebra, the observational equality is preserved by the non-observer operations, i.e., the observational equality is a congruence relation on the Σ_{COL}-generated subalgebra (but not necessarily on the whole algebra). In this way we obtain a category of COL-algebras where the corresponding morphism notion expresses the behavioral relationships between algebras. (In particular, isomorphic COL-algebras are considered to be behaviorally equivalent.) We will illustrate by examples that our concept of COL-algebras allows flexible constructions of realizations.

To specify properties of such realizations, we use ordinary first-order formulas together with a powerful satisfaction relation $\models_{\Sigma_{\mathrm{COL}}}$, called COL-satisfaction, which takes into account the reachability and the observability points of view in the sense that the universal quantifier "$\forall x : s$" is interpreted by considering only constructor-generated elements for the values of x and the equality symbol "=" is interpreted by the observational equality. Then the model class of a basic COL-specification $\mathrm{SP}_{\mathrm{COL}} = \langle \Sigma_{\mathrm{COL}}, \mathrm{Ax} \rangle$ (with signature Σ_{COL} and a set Ax of first-order sentences as axioms) consists of all COL-algebras which satisfy w.r.t. $\models_{\Sigma_{\mathrm{COL}}}$ the axioms Ax.

Many technicalities that are used in this paper to define the observational equality and the COL-satisfaction relation are borrowed from our previous work in [9,4] where we have used partial congruence relations. However, the COL framework is very different in spirit from these earlier approaches since the crucial idea is now to introduce distinguished sets of observer and constructor operations and not to start from observable sorts and from input sorts as done in [9,4]. This has many consequences from the specification methodological point of view (because this leads naturally to co-inductive and/or inductive definitions), from the proof theoretic point of view (since now there are less observable contexts and less constructor terms) and also from the modularity point of view since this is crucial to obtain a flexible notion of signature morphism such that the satisfaction condition of institutions (see [16]) is satisfied.

Indeed the declaration of distinguished sets of constructors and observers is essential to define COL-signature morphisms. We will require that constructors and observers are preserved by the morphisms and that no "new" constructors and no "new" observers are introduced in the target signature for "old" sorts (i.e., for sorts being in the image of the source signature). Thus for those sorts neither new elements can be generated nor new observations can be made, which is enough to guarantee encapsulation of properties w.r.t. the COL-satisfaction relation.

For structuring specifications we use the specification-building operators introduced in [37] which are applicable in the context of an arbitrary institution and hence also for COL. Thus we obtain a basic language for structured COL-specifications which includes operators for basic specifications, union of specifications, translation and hiding.

1.2 Black Box Semantics of COL*-Specifications*

The semantics of a COL-specification describes all its correct realizations and hence can be considered as its glass box semantics which is important from the implementor's point of view. From the user's point of view it is equally important to reason about the logical consequences of structured COL-specifications, whereby a first-order sentence φ is a consequence of a specification $\mathrm{SP}_{\mathrm{COL}}$, denoted by $\mathrm{SP}_{\mathrm{COL}} \models_{\Sigma_{\mathrm{COL}}} \varphi$, if all models of $\mathrm{SP}_{\mathrm{COL}}$ satisfy φ w.r.t. $\models_{\Sigma_{\mathrm{COL}}}$. For this purpose it is convenient to *abstract* the models of a specification into "idealized" models, such that the consequences w.r.t. $\models_{\Sigma_{\mathrm{COL}}}$ of the actual models of the specification are exactly the consequences of the idealized models in standard first-order logic. Technically, an idealized model is constructed from a COL-algebra A by first restricting to the Σ_{COL}-generated subalgebra of A and then by identifying all elements which are observationally equal. The resulting algebra is reachable and fully abstract (w.r.t. the given constructors and observers). The class of the idealized models of a specification $\mathrm{SP}_{\mathrm{COL}}$ is called the black box semantics of $\mathrm{SP}_{\mathrm{COL}}$ and denoted by $[\![\mathrm{SP}_{\mathrm{COL}}]\!]$.

Indeed the black box semantics allows us to characterize behavioral consequences of COL-specifications by means of standard satisfaction since, for any Σ-sentence φ,

$$\mathrm{SP}_{\mathrm{COL}} \models_{\Sigma_{\mathrm{COL}}} \varphi \text{ if and only if } [\![\mathrm{SP}_{\mathrm{COL}}]\!] \models \varphi\,.$$

1.3 Proving Consequences of Structured COL*-specifications*

The above characterization provides the crucial idea to develop powerful and practically applicable proof techniques for the verification of logical consequences of COL-specifications by reasoning in terms of standard satisfaction. The above characterization shows that proofs of behavioral consequences of a COL-specification $\mathrm{SP}_{\mathrm{COL}}$ can be reduced to proofs of standard consequences of the black box semantics $[\![\mathrm{SP}_{\mathrm{COL}}]\!]$ of $\mathrm{SP}_{\mathrm{COL}}$. But we still have to cope with the problem that the black box semantics of $\mathrm{SP}_{\mathrm{COL}}$ results from a semantic construction and has no direct syntactic representation (if we want to avoid infinitary logic). Therefore the next idea is to provide a suitable transformation $\hat{\varepsilon}(\mathrm{SP}_{\mathrm{COL}})$ of $\mathrm{SP}_{\mathrm{COL}}$ and to characterize consequences of $[\![\mathrm{SP}_{\mathrm{COL}}]\!]$ by means of consequences of $\hat{\varepsilon}(\mathrm{SP}_{\mathrm{COL}})$:

$$[\![\mathrm{SP}_{\mathrm{COL}}]\!] \models \varphi \;\text{ if and only if } \hat{\varepsilon}(\mathrm{SP}_{\mathrm{COL}}) \models \varphi$$

where $\hat{\varepsilon}(\mathrm{SP}_{\mathrm{COL}})$ is built over a standard specification formalism which in our case will be finitary first-order logic with sort-generation constraints.

To obtain our results we use the concept of an institution encoding introduced in [38]. The essential idea is that, given an institution encoding $\varepsilon : I \to I'$ between an institution I and a target institution I', one obtains a proof rule:

$$\frac{\hat{\varepsilon}(SP_I) \vdash^{I'} \varepsilon^{\mathrm{Sen}}(\varphi)}{SP_I \vdash^{I} \varphi}$$

for proving consequences φ of structured specifications SP_I (built over I). Thereby $\varepsilon^{\mathrm{Sen}}$ is a translation of sentences (given by ε), $\hat{\varepsilon}$ is a (structure preserving) translation of specifications (derived from ε) and $\vdash^{I'}$ denotes an existing proof system for structured specifications built over the target institution I'. We will show that the above proof rule is sound (and complete) if $\vdash^{I'}$ is sound (and complete), provided that satisfaction is closed under isomorphisms in both I and I′ and that $\varepsilon : I \to I'$ is an iso-reflecting logical institution encoding.

As mentioned above, in our case the target institution I' will be the institution of first-order logic with equality and sort-generation constraints used in (a sublanguage of) Casl. The crucial idea of our approach is to internalize observable contexts by an adequate syntactic encoding such that observable contexts are represented by generated values of auxiliary "context sorts". The application of an observable context to a (non-observable) element can then be inductively defined by using so-called "apply operations". Moreover, to reflect adequately the black box semantics of a specification, appropriate axioms of the form $\forall x, y : s.\ [(\forall c : \mathit{contextsort}.\ \mathit{apply}(c, x) = \mathit{apply}(c, y)) \Rightarrow x = y]$ are introduced which characterize full abstractness and, besides sort-generation constraints for the auxiliary context sorts, we also introduce sort-generation constraints which express reachability w.r.t. the given constructors.

A syntactic encoding of observable contexts was already described in [4], but there it was considered to be of no practical interest. The same idea is now fruitful because of the following two reasons:

(1) Since the introduction of observational logic in [21], we use a distinguished set of observer operations which leads to a smaller set of observable contexts and hence to much simpler proofs when the contexts are encoded.
(2) Since [7], we define the set of observable contexts using a coinductive style, and the corresponding encoding of observable contexts is much more adequate for behavioral proofs.

In summary our approach provides proof techniques that:

- integrate constructors (in particular, for datatypes) *and* observers (in particular, for states or infinite objects),
- support full first-order logic for axioms and for consequences (in particular, conditional equations),
- are applicable to arbitrary structured specifications built with the usual institution-independent specification-building operators,
- make no use of infinitary rules or infinitary sentences (in contrast with some of our earlier work, e.g., [21,6]), and hence
- are easily implemented with existing (inductive) theorem provers.

1.4 Related Frameworks

Many approaches in the literature already cover in some way reachability and/or observability. However, most of them either are not based on a loose semantics (like [31]) or are too restrictive w.r.t. the interpretation of reachability in the sense that only reachable models are admitted. Thus standard implementations which simply contain junk (like the realization of natural numbers by integers) are ruled out from the models of a specification. For instance CoCasl [26] provides a framework that integrates also reachability and observability concepts, but CoCasl does not provide a glass box semantics (while a proper combination of the generated and cogenerated constructs of CoCasl correspond to our black box semantics). The ultra-loose approach of [40], the notion of behavioral specification w.r.t. a partial observational equality in [9,23] and the hidden algebra approach are closely related to our framework. The main difference to [40] is that there no explicit notion of observer or constructor operation is used while in our approach they are the basic ingredients of a signature which lead to a specification methodology and to an institution tailored to observability and reachability. The partial observational equality of [9] does not take into account distinguished sets of observer and constructor operations which in our case are essential to obtain efficient proof techniques and lead to an adequate notion of signature morphism. The main difference to the presentation of hidden algebra in [15] is that there the reachable values are given by a fixed data universe while in our approach constructors can be defined for arbitrary sorts and hence also for hidden state sorts which we believe is important to deal with reachable states.

For hidden algebra different kinds of proof techniques have been proposed. Most closely related to our proof strategy is the approach in [34] which is also based on a context encoding but is restricted to flat equational specifications. Another proof technique for hidden algebra is circular coinductive rewriting [17] which was recently extended to conditional equations with observable premises. Circular coinduction describes a procedure where, in our opinion, the circularity corresponds to the application of an induction hypothesis in the sense of our proof method. A survey of inference rules for algebraic and coalgebraic specifications with explicit induction and coinduction rules is given in [32].

2 Basic Notions

In this section we briefly summarize the basic technical ingredients that are needed in our framework.

2.1 Algebraic Preliminaries

We assume that the reader is familiar with the basic notions of algebraic specifications (see, e.g., [39,25,2]), like the notions of (many-sorted) *signature* $\Sigma = (S, \mathrm{OP})$ (where S is a set of *sorts* and OP is a set of *operation symbols* $op : s_1, \dots, s_n \to s$), *signature morphism* $\sigma : \Sigma \to \Sigma'$, *(total)* Σ*-algebra* $A = ((A_s)_{s \in S}, (op^A)_{op \in \mathrm{OP}})$, Σ*-term algebra* $T_\Sigma(X)$ over a family $X = (X_s)_{s \in S}$ of pairwise disjoint sets X_s of variables of sort s and *interpretation* $I_\alpha : T_\Sigma(X) \to A$ w.r.t. a *valuation* $\alpha : X \to A$. The class of all Σ-algebras is denoted by $\mathrm{Alg}(\Sigma)$. Together with Σ-morphisms this class forms a category which, for simplicity, is also denoted by $\mathrm{Alg}(\Sigma)$.

For any signature morphism $\sigma : \Sigma \to \Sigma'$, the *reduct functor* $\_\_|_\sigma : \mathrm{Alg}(\Sigma') \to \mathrm{Alg}(\Sigma)$ is defined as usual. For $\Sigma' = (S', \mathrm{OP}')$, an S'-sorted n-ary relation is a family $R' = (R'_{s'})_{s' \in S'}$ of n-ary relations $R'_{s'}$ and the reduct of R' w.r.t. a signature morphism $\sigma : \Sigma \to \Sigma'$ is the S-sorted relation $R'|_\sigma = ((R'|_\sigma)_s)_{s \in S}$ where $(R'|_\sigma)_s \overset{\text{def}}{=} R'_{\sigma(s)}$ for all $s \in S$.

2.2 Institutions

The notion of an institution was introduced by Goguen and Burstall [16] to formalize the general concept of a logical system from a model-theoretic point of view. [1] An *institution* I consists of:

- a category Sign of *signatures*;
- a functor $\mathrm{Sen} : \mathrm{Sign} \to \mathbf{Set}$, giving a set $\mathrm{Sen}(\Sigma)$ of Σ-*sentences* for each signature $\Sigma \in \mathrm{Sign}$;
- a functor $\mathrm{Mod} : \mathrm{Sign}^{\mathrm{op}} \to \mathbf{Cat}$, giving a category $\mathrm{Mod}(\Sigma)$ of Σ-*models* for each signature $\Sigma \in \mathrm{Sign}$; and
- for each signature $\Sigma \in \mathrm{Sign}$, a *satisfaction relation* $\models_\Sigma \subseteq \mathrm{Mod}(\Sigma) \times \mathrm{Sen}(\Sigma)$

such that the so-called *satisfaction condition* is fulfilled. The satisfaction condition requires that for any signature morphism $\sigma : \Sigma \to \Sigma'$, Σ-sentence

[1] See [37] for an overview on the theory of institutions.

$\varphi \in \mathrm{Sen}(\Sigma)$ and Σ'-model $M' \in \mathrm{Mod}(\Sigma')$:

$$M' \models_{\Sigma'} \sigma(\varphi) \text{ if and only if } M'|_\sigma \models_\Sigma \varphi\,.$$

Here and in the following we write $M'|_\sigma$ for $\mathrm{Mod}(\sigma)(M')$, and similarly $\sigma(\varphi)$ for $\mathrm{Sen}(\sigma)(\varphi)$.

A Σ-sentence φ is a *logical consequence* of a Σ-sentence ψ, denoted by $\psi \models_\Sigma \varphi$, if for each Σ-model M we have: If $M \models_\Sigma \psi$ then $M \models_\Sigma \varphi$.

Satisfaction in the institution I is said to be *closed under isomorphisms* when isomorphic models satisfy exactly the same sentences.

An important example is the institution FOLEq of many-sorted first-order logic with equality as detailed, e.g., in [3]. In FOLEq signatures are many-sorted signatures, models are Σ-algebras and sentences are arbitray first-order Σ-formulas which are built from equations $t = r$ (with terms $t, r \in T_\Sigma(X)$ of the same sort), from logical connectives $\neg$, $\wedge$, $\vee$, $\Rightarrow$, and from the quantifiers $\forall$ and $\exists$. The satisfaction relation is the usual satisfaction relation of first-order logic with equality (which is closed under isomorphisms). Similarly the institution IFOLEq of infinitary first-order logic with equality is defined where sentences may contain conjunctions and disjunctions of countably many sentences.

The institution CFOLEq will be used in the second part of this paper. It is an extension of the FOLEq institution where, in addition to the usual (finitary) first-order sentences, we consider also as extra sentences *sort-generation constraints* of the form $\mathrm{SGC}(S_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Cons}})$ such that for a given signature $\Sigma = (S, \mathrm{OP})$, $S_{\mathrm{Cons}} \subseteq S$ and $\mathrm{OP}_{\mathrm{Cons}} \subseteq OP$. The sorts in S_{Cons} are called constrained sorts and the operation symbols in $\mathrm{OP}_{\mathrm{Cons}}$ are called constructors. [2] A Σ-algebra A satisfies a sort-generation constraint $\mathrm{SGC}(S_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Cons}})$ if for any sort $s \in S_{\mathrm{Cons}}$ and any element $a \in A_s$ there exists a constructor term t (built only from constructors and from variables of non-constrained sorts) and a valuation α such that $I_\alpha(t) = a$. Sort-generation constraints are used e.g. in the CASL language [1,10]. Note that satisfaction in CFOLEq is closed under isomorphisms.

[2] From a technical point of view, to ensure that the satisfaction condition of institutions will hold, a signature morphism is needed as a third component of a sort-generation constraint. Thus, given a signature $\Sigma = (S, \mathrm{OP})$, a Σ-sort-generation constraint is a triple $(S_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Cons}}, \theta)$, where $\theta : \Sigma_0 \to \Sigma$ is a signature morphism and $S_{\mathrm{Cons}} \subseteq S_0$, $\mathrm{OP}_{\mathrm{Cons}} \subseteq \mathrm{OP}_0$, with $\Sigma_0 = (S_0, \mathrm{OP}_0)$. We use the abbreviation $\mathrm{SGC}(S_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Cons}})$ for Σ-constraints $(S_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Cons}}, \theta)$ where θ is either the identity or a signature inclusion, and only sort-generation constraints of this form will be needed in this paper. See e.g. [27,28] for more details. Moreover, w.l.o.g. we always assume that S_{Cons} is exactly the set of the range sorts of the constructors in $\mathrm{OP}_{\mathrm{Cons}}$, to ensure consistency with the forthcoming definitions and notations.

Any institution provides a suitable framework for defining a set of specification-building operators which are independent from the concrete form of the institution. We will use the following four fundamental operators introduced in [36] for constructing structured specifications over an institution I. The semantics of a specification SP is always determined by its signature, denoted by $\mathcal{S}ig[\mathrm{SP}]$, and by its class of models, denoted by $\mathcal{M}od[\mathrm{SP}]$. Note that our semantics of hiding slightly deviates from [36]. We prefer to follow [39] (see also the discussion in [36, pp. 189–190]), and we will ensure that the model class of a specification is always closed under isomorphisms (provided satisfaction is so in I).

***presentation*:** Any pair $\langle\Sigma, \Phi\rangle$ consisting of a signature $\Sigma \in \mathrm{Sign}$ and of a set Φ of Σ-sentences is a specification with semantics:
$\mathcal{S}ig[\langle\Sigma, \Phi\rangle] \overset{\mathrm{def}}{=} \Sigma$
$\mathcal{M}od[\langle\Sigma, \Phi\rangle] \overset{\mathrm{def}}{=} \{M \in \mathrm{Mod}(\Sigma) \mid M \models_\Sigma \Phi\}$

***union*:** For any two specifications SP_1 and SP_2 with the same signature $\mathcal{S}ig[\mathrm{SP}_1] = \mathcal{S}ig[\mathrm{SP}_2] = \Sigma$, the expression $\mathrm{SP}_1 \cup \mathrm{SP}_2$ is a specification with semantics:
$\mathcal{S}ig[\mathrm{SP}_1 \cup \mathrm{SP}_2] \overset{\mathrm{def}}{=} \Sigma$
$\mathcal{M}od[\mathrm{SP}_1 \cup \mathrm{SP}_2] \overset{\mathrm{def}}{=} \mathcal{M}od[\mathrm{SP}_1] \cap \mathcal{M}od[\mathrm{SP}_2]$

***translation*:** For any specification SP and signature morphism $\sigma : \mathcal{S}ig[\mathrm{SP}] \to \Sigma$, the expression **translate** SP **by** σ is a specification with semantics:
$\mathcal{S}ig[\textbf{translate}\ \mathrm{SP}\ \textbf{by}\ \sigma] \overset{\mathrm{def}}{=} \Sigma$
$\mathcal{M}od[\textbf{translate}\ \mathrm{SP}\ \textbf{by}\ \sigma] \overset{\mathrm{def}}{=} \{M \in \mathrm{Mod}(\Sigma) \mid M|_\sigma \in \mathcal{M}od[\mathrm{SP}]\}$

***hiding*:** For any specification SP and signature morphism $\sigma : \Sigma \to \mathcal{S}ig[\mathrm{SP}]$, the expression **derive from** SP **by** σ is a specification with semantics:
$\mathcal{S}ig[\textbf{derive from}\ \mathrm{SP}\ \textbf{by}\ \sigma] \overset{\mathrm{def}}{=} \Sigma$
$\mathcal{M}od[\textbf{derive from}\ \mathrm{SP}\ \textbf{by}\ \sigma] \overset{\mathrm{def}}{=} \mathrm{Iso}_\Sigma(\{M|_\sigma \mid M \in \mathcal{M}od[\mathrm{SP}]\})$,
where $\mathrm{Iso}_\Sigma(\_)$ denotes the closure under Σ-isomorphisms in $\mathrm{Mod}(\Sigma)$.

Fact 1. *If satisfaction is closed under isomorphisms, then the model class of any structured specification* SP *over* I *built with the above specification-building primitives is closed under isomorphisms.*

PART I — The Constructor-based Observational Logic COL

In the first part of this paper we develop, step by step, the syntactic and semantic notions which lead to the constructor-based observational logic institution, called COL. The COL institution has evolved as a synthesis of the constructor-based logic institution presented in [8] and of the observational logic institution originally introduced in [21]. While the duality of these frameworks has been studied in [8], in this paper we focus on their integration.

3 COL-Signatures and COL-Algebras

3.1 COL-Signatures, Generated Parts and Observational Equalities

We start by considering the syntactic concept of a COL-signature which consists of a standard algebraic signature together with a distinguished set of constructor operations and a distinguished set of observer operations. Intuitively, the constructors determine those elements which are of interest from the user's point of view while the observers determine a set of observable experiments that a user can perform to examine hidden states. Thus we can abstract from junk elements and also from concrete state representations (whereby two states are considered to be "observationally equal" if they cannot be distinguished by observable experiments).

Definition 2 (COL**-signature).** *A* constructor *is an operation symbol cons* : $s_1, \ldots, s_n \to s$ *with* $n \geq 0$*. The result sort s of cons is called a* constrained sort*. An* observer *is a pair* (obs, i) *where obs is an operation symbol obs* : $s_1, \ldots, s_n \to s$ *with* $n \geq 1$ *and* $1 \leq i \leq n$*. The distinguished argument sort* s_i *of obs is called a* state sort *(or* hidden sort*). If obs* : $s_1 \to s$ *is a unary observer we will simply write obs instead of* $(obs, 1)$*.*

A COL-signature $\Sigma_{\text{COL}} = (\Sigma, \text{OP}_{\text{Cons}}, \text{OP}_{\text{Obs}})$ *consists of a signature* $\Sigma = (S, \text{OP})$*, a set* $\text{OP}_{\text{Cons}} \subseteq \text{OP}$ *of constructors and a set* OP_{Obs} *of observers* (obs, i) *with* $obs \in \text{OP}$*.*

The set $S_{\text{Cons}} \subseteq S$ *of* constrained sorts *(w.r.t.* OP_{Cons}*) consists of all sorts s such that there exists at least one constructor in* OP_{Cons} *with range s. The set* $S_{\text{Loose}} \subseteq S$ *of* loose sorts *consists of all sorts which are not constrained, i.e.* $S_{\text{Loose}} = S \setminus S_{\text{Cons}}$ *.*

The set $S_{\text{State}} \subseteq S$ *of* state sorts *(or* hidden sorts*, w.r.t.* OP_{Obs}*) consists of all sorts* s_i *such that there exists at least one observer* (obs, i) *in* OP_{Obs} *,*

obs : $s_1, \ldots, s_i, \ldots, s_n \to s$. *The set* $S_{\text{Obs}} \subseteq S$ *of* observable sorts *consists of all sorts which are not a state sort, i.e.* $S_{\text{Obs}} = S \setminus S_{\text{State}}$ *.*

An observer $(obs, i) \in \mathrm{OP}_{\text{Obs}}$ *with profile* $obs : s_1, \ldots, s_i, \ldots, s_n \to s$ *is called a* direct observer *of* s_i *if* $s \in S_{\text{Obs}}$ *, otherwise it is an* indirect observer*.*

Note that in many examples state sorts are also constrained sorts which allows us to deal with reachable states. We implicitly assume in the following that whenever we consider a COL-signature Σ_{COL}, then $\Sigma_{\text{COL}} = (\Sigma, \mathrm{OP}_{\text{Cons}}, \mathrm{OP}_{\text{Obs}})$ with $\Sigma = (S, \mathrm{OP})$ and similarly for Σ'_{COL} etc.

Remark 3. Let $\Sigma_{\text{COL}} = (\Sigma, \mathrm{OP}_{\text{Cons}}, \mathrm{OP}_{\text{Obs}})$ be a COL-signature. If the set OP_{Obs} of observers is empty, then all sorts are observable sorts. This is the special case considered in the constructor-based logic institution [8]. On the other hand, if the set $\mathrm{OP}_{\text{Cons}}$ of constructors is empty, then all sorts are loose sorts which is the special case considered in the observational logic institution [21]. If the sets of observers and of constructors are both empty, then we are in the standard framework of universal algebra.

Example 4. As a running example we consider the following COL-signature $\Sigma_{\text{COL}} = (\Sigma, \mathrm{OP}_{\text{Cons}}, \mathrm{OP}_{\text{Obs}})$ for containers of natural numbers where:

$\Sigma = (S, \mathrm{OP})$, $S = \{\ \mathit{bool},\ \mathit{nat},\ \mathit{container}\ \}$
$\mathrm{OP} = \{\ \mathit{true} : \to \mathit{bool};\ \ \mathit{false} : \to \mathit{bool};$
$\quad 0 : \to \mathit{nat};\ \ \mathit{succ} : \mathit{nat} \to \mathit{nat};\ \ \mathit{add} : \mathit{nat}, \mathit{nat} \to \mathit{nat};$
$\quad \mathit{empty} : \to \mathit{container};\ \ \mathit{insert} : \mathit{container}, \mathit{nat} \to \mathit{container};$
$\quad \mathit{remove} : \mathit{container}, \mathit{nat} \to \mathit{container};$
$\quad \mathit{isin} : \mathit{container}, \mathit{nat} \to \mathit{bool}\ \}$
$\mathrm{OP}_{\text{Cons}} = \{\ \mathit{true},\ \mathit{false},\ 0,\ \mathit{succ},\ \mathit{empty},\ \mathit{insert}\ \}$
$\mathrm{OP}_{\text{Obs}} = \{\ (\mathit{isin}, 1)\ \}$

Hence, in this example, all sorts are constrained, *container* is the only state sort and the observable sorts are *bool* and *nat*. ◇

Any COL-signature determines a set of constructor-terms which are inductively defined starting from constants in $\mathrm{OP}_{\text{Cons}}$. The interpretation of a constructor term denotes always a value of a constrained sort.

Definition 5 (Constructor term). *Let* Σ_{COL} *be a* COL*-signature, and let* $X = (X_s)_{s \in S}$ *be a family of pairwise disjoint, countably infinite sets* X_s *of variables of sort* s*. The sets* $\mathcal{T}(\Sigma_{\text{COL}})_s$*,* $s \in S_{\text{Cons}}$ *, of* constructor terms *with "constrained result sort"* s *are inductively defined as follows:*

(1) Each constant $cons : \to s \in \mathrm{OP}_{\text{Cons}}$ *belongs to* $\mathcal{T}(\Sigma_{\text{COL}})_s$ *.*
(2) For each constructor $cons : s_1, \ldots, s_n \to s \in \mathrm{OP}_{\text{Cons}}$ *with* $n \geq 1$ *and terms* $t_1, \ldots, t_n$ *such that* t_i *is a variable* $x_i{:}s_i$ *if* $s_i \in S_{\text{Loose}}$ *and* $t_i \in \mathcal{T}(\Sigma_{\text{COL}})_{s_i}$ *if* $s_i \in S_{\text{Cons}}$ *,* $cons(t_1, \ldots, t_n) \in \mathcal{T}(\Sigma_{\text{COL}})_s$ *.*

The set of all constructor terms is denoted by $\mathcal{T}(\Sigma_{\mathrm{COL}})$. We implicitly assume in the following that for any constrained sort $s \in S_{\mathrm{Cons}}$ there exists a constructor term of sort s.

Note that only constructor symbols and variables of loose sorts are used to build constructor terms. In particular, if all sorts are constrained, i.e., $S_{\mathrm{Cons}} = S$, the constructor terms are exactly the $(S, \mathrm{OP}_{\mathrm{Cons}})$-ground terms which are built by the constructor symbols. This is the case, for instance, in the above example.

The syntactic notion of a constructor term induces, for any Σ-algebra A, the definition of a family of subsets of the carrier sets of A, called the Σ_{COL}-generated part, which consists of those elements which can be constructed by the interpretations of the given constructors (starting from constants and from arbitrary elements of loose sort, if any). In the following considerations the Σ_{COL}-generated part plays a crucial role since it represents those elements which are of interest from the user's point of view.

Definition 6 (Σ_{COL}-generated part). *Let Σ_{COL} be a* COL*-signature. For any Σ-algebra $A \in \mathrm{Alg}(\Sigma)$, the Σ_{COL}-*generated part *of A is an S-sorted family of sets $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A) = (\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_s)_{s\in S}$ defined as follows.*

***Case** $s \in S_{\mathrm{Loose}}$:* $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_s = A_s$
***Case** $s \in S_{\mathrm{Cons}}$:* $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_s = \{a \in A_s \mid$ *there exists a term* $t \in \mathcal{T}(\Sigma_{\mathrm{COL}})_s$ *and a valuation* $\alpha : X \to A$ *such that* $I_\alpha(t) = a\}$.

Definition 7 (Reachable algebra). *Let Σ_{COL} be a* COL*-signature. A Σ-algebra A is called* reachable *(w.r.t. Σ_{COL}) if its carrier sets coincide with the carrier sets of its Σ_{COL}-generated part.*

Remark 8. The Σ_{COL}-generated part of a Σ-algebra A is uniquely determined by the constructors $\mathrm{OP}_{\mathrm{Cons}}$ distinguished by $\Sigma_{\mathrm{COL}} = (\Sigma, \mathrm{OP}_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Obs}})$. The observers $\mathrm{OP}_{\mathrm{Obs}}$ are irrelevant here. Hence the notion of reachability also depends only on the given constructors $\mathrm{OP}_{\mathrm{Cons}}$.

Example 9. Consider the signature Σ_{COL} of Example 4 and the following Σ-algebra A with carriers:
$A_{bool} = \{T, F\}$, $A_{nat} = \mathbb{Z}$ (set of the integers),
$A_{container} = \mathbb{Z}^* \times \mathbb{Z}^*$ (pairs of finite lists of integers),
and with operations:
$true^A = T$, $false^A = F$, $0^A = 0$, $succ^A(a) = a + 1$, $add^A(a, b) = a + b$,
$empty^A = (<>, <>)$,
$insert^A((< a_1, \ldots, a_n >, < b_1, \ldots, b_m >), a) =$
$(< a, a_1, \ldots, a_n >, < b_1, \ldots, b_m >)$ if $a \neq a_i$ for $i = 1, \ldots, n$,
$insert^A((s, t), a) = (s, t)$ otherwise,
$remove^A((< a_1, \ldots, a_n >, < b_1, \ldots, b_m >), a) =$

$(< a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n >, < a, b_1, \ldots, b_m >)$ if $a_i = a$ and $a_j \neq a$ for $j = 1, \ldots, i-1$,
$\mathit{remove}^A((s,t), a) = (s,t)$ otherwise,
$\mathit{isin}^A((< a_1, \ldots, a_n >, t), a) = F$ if $a \neq a_i$ for $i = 1, \ldots, n$,
$\mathit{isin}^A((s,t), a) = T$ otherwise.
The above Σ-algebra A can be considered as an implementation of containers of natural numbers whereby the natural numbers are implemented by the integers and containers are implemented by two finite lists s and t such that s stores the elements which are actually in the container and t is a "trash" which stores those elements that have been removed from the container. The *remove* operation is defined in an efficient way: only one occurrence of a given element is deleted from the actual elements of a container. This is sufficient since the *insert* operation only stores an element if it does not already occur in the container. The Σ_{COL}-generated part $\text{Gen}_{\Sigma_{\text{COL}}}(A)$ of A consists of the following sets:
$\text{Gen}_{\Sigma_{\text{COL}}}(A)_{\mathit{bool}} = \{T, F\}$,
$\text{Gen}_{\Sigma_{\text{COL}}}(A)_{\mathit{nat}} = \mathbb{N}$ (set of the natural numbers),
$\text{Gen}_{\Sigma_{\text{COL}}}(A)_{\mathit{container}} = \{(s, <>) \mid s \in \mathbb{N}^*$ and each element of s occurs only once in $s\}$. $\Diamond$

Let us now focus on the set OP_{Obs} of observers declared by a COL-signature Σ_{COL}. The observers determine a set of observable contexts which represent the observable experiments. In contrast to the inductive definition of constructor terms, observable contexts are defined in a coinductive style.

Definition 10 (Observable context). *Let* Σ_{COL} *be a* COL*-signature, let* $X = (X_s)_{s \in S}$ *be a family of pairwise disjoint, countably infinite sets* X_s *of variables of sort* s *and let* $Z = (\{z_s\})_{s \in S_{\text{State}}}$ *be a disjoint family of singleton sets (one for each state sort). The sets* $\mathcal{C}(\Sigma_{\text{COL}})_{s \to s'}$ *of* observable Σ_{COL}-contexts *with "application sort"* s *and "observable result sort"* s'*, with* $s \in S_{\text{State}}$ *and* $s' \in S_{\text{Obs}}$*, are inductively defined as follows:*

(1) For each direct observer (obs, i) *with* $\mathit{obs} : s_1, \ldots, s_i, \ldots, s_n \to s'$ *and pairwise disjoint variables* $x_1{:}s_1, \ldots, x_n{:}s_n$*,*
$\mathit{obs}(x_1, \ldots, x_{i-1}, z_{s_i}, x_{i+1}, \ldots, x_n) \in \mathcal{C}(\Sigma_{\text{COL}})_{s_i \to s'}$ *.*

(2) For each observable context $c \in \mathcal{C}(\Sigma_{\text{COL}})_{s \to s'}$ *, for each indirect observer* (obs, i) *with* $\mathit{obs} : s_1, \ldots, s_i, \ldots, s_n \to s$*, and pairwise disjoint variables* $x_1{:}s_1, \ldots, x_n{:}s_n$ *not occurring in* c*,*
$c[\mathit{obs}(x_1, \ldots, x_{i-1}, z_{s_i}, x_{i+1}, \ldots, x_n)/z_s] \in \mathcal{C}(\Sigma_{\text{COL}})_{s_i \to s'}$
where $c[\mathit{obs}(x_1, \ldots, x_{i-1}, z_{s_i}, x_{i+1}, \ldots, x_n)/z_s]$ *denotes the term obtained from* c *by substituting the term* $\mathit{obs}(x_1, \ldots, x_{i-1}, z_{s_i}, x_{i+1}, \ldots, x_n)$ *for* z_s *.*

The set of all observable contexts is denoted by $\mathcal{C}(\Sigma_{\text{COL}})$*. We implicitly assume in the following that for any state sort* $s \in S_{\text{State}}$ *there exists an observable context with application sort* s*.*

Note that only the observer operations are used to build observable contexts For instance, the context $isin(z_{container}, x)$ is (up to renaming of the variable x) the only observable context in the container example.

The syntactic notion of an observable context will be used to define, for any Σ-algebra A, a semantic relation, called observational equality, which expresses indistinguishability of states. As already pointed out, the observable contexts represent observable experiments which can be applied to examine states. Then two states are observationally equal if they cannot be distinguished by these experiments.

If there is no constructor symbol, this intuitive idea can easily be formalized as done in the observational logic framework, see [21] and [8, Section 2]. However, if we integrate observability and reachability concepts, we have to be careful with respect to the role of constructors in observable experiments. For instance, in the container example, the observable context $isin(z_{container}, x)$ represents a set of observable experiments on containers which depend on the actual values of the variable x of sort nat. Since nat is a constrained sort, from the user's point of view the only relevant values are representable by a constructor term (and hence belong to the Σ_{COL}-generated part). This leads to the following definition of the observational equality which depends, in contrast to the pure observational approach in [21,8], not only on the observers but also on the chosen constructors.

Definition 11 (Observational Σ_{COL}-equality). *Let Σ_{COL} be a* COL*-signature. For any Σ-algebra $A \in \mathrm{Alg}(\Sigma)$, the* observational Σ_{COL}-equality *on A is an S-sorted binary relation $\approx_{\Sigma_{\mathrm{COL}},A} = (\approx_{\Sigma_{\mathrm{COL}},A,s})_{s\in S}$ defined as follows.*
For all $s \in S$, two elements $a, b \in A_s$ are observationally equal w.r.t. Σ_{COL}, i.e., $a \approx_{\Sigma_{\mathrm{COL}},A,s} b$ (or, for short, $a \approx_{\Sigma_{\mathrm{COL}},A} b$), if and only if

***Case** $s \in S_{\mathrm{Obs}}$: $a = b$*

***Case** $s \in S_{\mathrm{State}}$: for all observable sorts $s' \in S_{\mathrm{Obs}}$, for all observable contexts $c \in \mathcal{C}(\Sigma_{\mathrm{COL}})_{s\to s'}$, and for all valuations $\alpha, \beta : X \cup \{z_s\} \to A$ with $\alpha(x) = \beta(x) \in \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)$ if $x \in X$, $\alpha(z_s) = a$ and $\beta(z_s) = b$, we have $I_\alpha(c) = I_\beta(c)$.*

Definition 12 (Fully-abstract algebra). *Let Σ_{COL} be a* COL*-signature. A Σ-algebra A is called* fully abstract *(w.r.t. Σ_{COL}) if the observational Σ_{COL}-equality $\approx_{\Sigma_{\mathrm{COL}},A}$ on A coincides with the set-theoretic equality.*

Example 13. Consider the signature Σ_{COL} of Example 4 and the algebra of containers defined in Example 9 where a container is represented by a pair (s, t) of finite lists of integers. Two containers $(s1, t1)$ and $(s2, t2)$ are observationally equal, $(s1, t1) \approx_{\Sigma_{\mathrm{COL}},A} (s2, t2)$, if for all natural numbers n, $isin^A((s1, t1), n) = isin^A((s2, t2), n)$ holds. By definition of $isin^A$, this means that the same natural numbers occur in both $s1$ and $s2$. Thus the observational equality abstracts not only from the ordering and multiple occurrences of elements (and from the content of both "trashes" $t1$ and $t2$), but also from the occurrences of negative integers. This expresses exactly our intuition according to the given constructors and observers. For instance, the following container representations are observationally equal: $(< 1, 2 >, <>) \approx_{\Sigma_{\mathrm{COL}}, A} (< 2, -7, 2, -3, 1 >, < 6, -4 >)$. $\diamond$

3.2 COL-Algebras and Black Box Functors

Up to now the syntactic notion of a COL-signature Σ_{COL} has lead to the semantic concepts of a Σ_{COL}-generated part (determined by the constructors) and of an observational equality (determined by the observers but with an impact of the constructors) which both have been defined for an arbitrary algebra over the underlying signature Σ. As we will see in the following discussion, the constructors and the observers induce also certain constraints on algebras which lead to the notion of a COL-algebra.

In traditional approaches to reachability, constructor symbols are used to restrict the admissible models of a specification to those algebras which are reachable with respect to the given constructors (i.e. to reachable algebras, see Definition 7). We do not adopt this interpretation since, as many examples show, it is too restrictive if the semantics of a specification is expected to capture all correct realizations. For instance, the container algebra of Example 9 is not reachable w.r.t. the given constructors but should be usable as a correct realization of containers. As a consequence, we are interested in a more flexible framework where the constructor symbols are still essential, but nevertheless non-reachable algebras can be accepted as models if they satisfy certain conditions. Since the Σ_{COL}-generated part represents the elements of interest, one could simply require that no further elements should be constructible by the non-constructor operations (i.e. the Σ_{COL}-generated part is a Σ-subalgebra). Indeed, if we are working in a pure constructor-based framework, this condition fits perfectly to our intuition (see [8], Section 3). However, if we deal simultaneously with observability, this requirement is still too strong because from the user's point of view it doesn't matter if a non-constructor operation yields an element *outside* the Σ_{COL}-generated part as long as this element is observationally equal to some other element *inside* the Σ_{COL}-generated part. Technically this means that we first consider the smallest Σ-subalgebra containing the Σ_{COL}-generated part of a given Σ-algebra A and then require that each element of this subalgebra is observationally equal to some element of the Σ_{COL}-generated part of A. This condition is expressed by the reachability constraint given below which is based on the notion of a Σ_{COL}-generated subalgebra.

Definition 14 (Σ_{COL}-generated subalgebra). *Let Σ_{COL} be a* COL-*signature. For any Σ-algebra $A \in \mathrm{Alg}(\Sigma)$, the* Σ_{COL}-generated subalgebra *of A, denoted by $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_\Sigma = (\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s})_{s\in S}$, is the smallest Σ-subalgebra of A which contains the Σ_{COL}-generated part* $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)$.

The Σ_{COL}-generated subalgebra represents the only elements a user can compute (over the loose carrier sets) by invoking operations of Σ. Indeed, given a COL-signature $\Sigma_{\mathrm{COL}} = (\Sigma, \mathrm{OP}_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Obs}})$ with underlying signature $\Sigma = (S, \mathrm{OP})$, $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_\Sigma$ is the Σ-subalgebra of A generated by the (interpretations of the) operations OP over the carrier sets A_s with loose sort $s \in S_{\mathrm{Loose}}$.

Fact 15. *For any Σ-algebra A, we have:*

(1) $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_s = \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s} = A_s$ *for each loose sort $s \in S_{\mathrm{Loose}}$.*
(2) $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_s \subseteq \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s} \subseteq A_s$ *for each constrained sort $s \in S_{\mathrm{Cons}}$.*

Definition 16 (Reachability constraint). *Let Σ_{COL} be a* COL-*signature. A Σ-algebra A* satisfies the reachability constraint induced by Σ_{COL}, *if for any $a \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_\Sigma$ there exists $b \in \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)$ such that $a \approx_{\Sigma_{\mathrm{COL}},A} b$.*

Since for observable sorts the observational equality is the set-theoretic equality, it is obvious that for any Σ-algebra A which satisfies the reachability constraint induced by Σ_{COL} we have $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_s = \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s}$ for each observable sort $s \in S_{\mathrm{Obs}}$. Hence, for Σ-algebras which satisfy the given reachability constraint, Fact 15 can be refined in the following way.

Fact 17. *Let Σ_{COL} be a* COL-*signature. For any Σ-algebra A which satisfies the reachability constraint induced by Σ_{COL}, we have:*

(1) $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_s = \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s} = A_s$ *for each loose sort $s \in S_{\mathrm{Loose}}$.*
(2) $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_s = \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s} \subseteq A_s$ *for each observable sort $s \in S_{\mathrm{Obs}}$.*
(3) $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_s \subseteq \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s} \subseteq A_s$ *for each constrained state sort $s \in S_{\mathrm{Cons}} \cap S_{\mathrm{State}}$.*

Example 18. Let A be the container algebra of Example 9. It is obvious that the Σ_{COL}-generated part of A is not closed under the operation remove^A. For instance, $\mathit{remove}^A((<1,2>,<>),1) = (<2>,<1>) \notin \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_{\mathit{container}}$. In fact, for the constrained state sort *container*, we have $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_{\mathit{container}} \subsetneq \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,\mathit{container}} \subsetneq A_{\mathit{container}}$ where:
$\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,\mathit{container}} = \{(s,t) \mid s,t \in \mathbb{N}^*$ and each element of s occurs only once in $s\}$.
However, any element $(s,t) \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,\mathit{container}}$ is observationally equal to $(s,<>)$ (see Example 13) which is an element of the Σ_{COL}-generated part. Considering the observable sort *nat*, the Σ_{COL}-generated part is preserved under add^A, i.e., $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_{\mathit{nat}} = \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,\mathit{nat}} \subsetneq A_{\mathit{nat}}$.

Moreover, for the observable sort *bool*, obviously $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)_{bool} = \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,bool} = A_{bool}$.
Thus A satisfies the reachability constraint induced by Σ_{COL}. $\diamond$

Let us now discuss the constraints on a Σ-algebra A that are induced by the observers $\mathrm{OP}_{\mathrm{Obs}}$ of a COL-signature Σ_{COL}. Since the declaration of observers determines a particular observational equality on any Σ-algebra A, the (interpretations of the) non-observer operations should respect this observational equality, i.e. a non-observer operation should not contribute to distinguish states. For this purpose one could simply require that the observational equality is a Σ-congruence on A. Indeed, if we are working in a pure observational framework, this condition fits perfectly to our intuition (see [21,8]). However, if we deal simultaneously with reachability, this requirement is too strong because computations performed by a user can only lead to elements in the Σ-subalgebra $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}$. As a consequence, it is sufficient to require the congruence property on this subalgebra which is expressed by the following observability constraint.

Definition 19 (Observability constraint). *Let* Σ_{COL} *be a* COL*-signature. A* Σ*-algebra* A satisfies the observability constraint induced by Σ_{COL}*, if* $\approx_{\Sigma_{\mathrm{COL}},A}$ *is a* Σ*-congruence on* $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}$.

Example 20. The container algebra A of Example 9 satisfies the observability constraint of the given COL-signature for containers. Note, however, that $\approx_{\Sigma_{\mathrm{COL}},A}$ is only a Σ-congruence on $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}$ but not on the whole algebra A since $remove^A$ does not respect the observational equality for *all* elements of A.
Consider, for instance, the element $(<1,1>,<>) \notin \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,container}$. $(<1,1>,<>) \approx_{\Sigma_{\mathrm{COL}},A} (<1>,<>)$ but $remove^A((<1,1>,<>),1) = (<1>,<1>)$ is *not* observationally equal to $remove^A((<1>,<>),1) = (<>,<1>)$. $\diamond$

Definition 21 (COL-algebra). *Let* Σ_{COL} *be a* COL*-signature. A* Σ_{COL}-algebra *(also called* COL-algebra*) is a* Σ*-algebra* A *which satisfies the reachability and the observability constraints induced by* Σ_{COL}*. The class of all* Σ_{COL}*-algebras is denoted by* $\mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$.

Fact 22. *Let* Σ_{COL} *be a* COL*-signature. Any* Σ*-algebra* A *which is reachable and fully abstract w.r.t.* Σ_{COL} *is a* Σ_{COL}*-algebra.*

Remark 23. Compared with the partial observational equality used in [9], the important difference here is the declaration of the constructor and observer operations which provide much more flexibility than declaring just observable sorts and input sorts as done in [9]. The input sorts correspond to the loose sorts and, for any Σ-algebra A, the domain of the partial observational equality

$\approx_{S_{\mathrm{Obs}},S_{\mathrm{Loose}},A}$ is just the generated subalgebra $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}$. Moreover, if A is a COL-algebra, then the observational equality $\approx_{\Sigma_{\mathrm{COL}},A}$ coincides, on $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}$, with the partial observational equality $\approx_{S_{\mathrm{Obs}},S_{\mathrm{Loose}},A}$.

The satisfaction of the reachability and observability constraints allows us to construct for each COL-algebra A its *black box view* which is a reachable and fully abstract algebra representing the behavior of A from the user's point of view. The black box view of a Σ_{COL}-algebra A is constructed in two steps. First, we *restrict* to the Σ_{COL}-generated subalgebra $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}$ of A thus forgetting junk values that a user can never compute (over the carrier sets of the loose sorts) by invoking operations of Σ. Since, by assumption, A satisfies the observability constraint induced by Σ_{COL}, the observational Σ_{COL}-equality $\approx_{\Sigma_{\mathrm{COL}},A}$ is a Σ-congruence on $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}$. Therefore, in the next step, we can construct the quotient algebra $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}/\!\approx_{\Sigma_{\mathrm{COL}},A}$ which *identifies* all elements of $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}$ which are indistinguishable "from the outside". $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}/\!\approx_{\Sigma_{\mathrm{COL}},A}$ is considered as the black box view of A.

Definition 24 (Black box view). *Let A be a Σ_{COL}-algebra. The quotient algebra $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}/\!\approx_{\Sigma_{\mathrm{COL}},A}$ is called the* black box view *of A.*

Fact 25. *The black box view of any Σ_{COL}-algebra A is reachable and fully abstract w.r.t. Σ_{COL}. Moreover, if A is both reachable and fully abstract, then it is isomorphic to its black box view.*

To obtain a category of COL-algebras we define the following morphism notion which is a generalization of standard Σ-homomorphisms.

Definition 26 (COL-morphism). *Let $A, B \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$ be two Σ_{COL}-algebras. A* Σ_{COL}-morphism *(also called* COL-morphism*) $h : A \to B$ is an S-sorted family $(h_s)_{s\in S}$ of relations*

$$h_s \subseteq \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s} \times \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_{\Sigma,s}$$

with the following properties, for all $s \in S$:

(1) For all $a \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s}$, there exists $b \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_{\Sigma,s}$ such that $a\ h_s\ b$.
(2) For all $a \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s}$, b, $b' \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_{\Sigma,s}$, if $a\ h_s\ b$, then ($a\ h_s\ b'$ if and only if $b \approx_{\Sigma_{\mathrm{COL}},B} b'$).
(3) For all a, $a' \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s}$, $b \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_{\Sigma,s}$, if $a\ h_s\ b$ and $a \approx_{\Sigma_{\mathrm{COL}},A} a'$, then $a'\ h_s\ b$.
(4) For all $op : s_1, \dots, s_n \to s \in \mathrm{OP}$, for all $a_i \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s_i}$, and $b_i \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_{\Sigma,s_i}$, if $a_i\ h_{s_i}\ b_i$ for $i = 1, \dots, n$, then $op^A(a_1, \dots, a_n)\ h_s\ op^B(b_1, \dots, b_n)$.

The following lemma shows that there is a one to one correspondence between

COL-morphisms $h : A \to B$ and standard morphisms between the black box views of A and B.[3]

Lemma 27. *Let $A, B \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$ be two Σ_{COL}-algebras and $h : A \to B$ be a Σ_{COL}-morphism.*
Then $h/{\approx_{\Sigma_{\mathrm{COL}}}} : \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_\Sigma/{\approx_{\Sigma_{\mathrm{COL}},A}} \to \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_\Sigma/{\approx_{\Sigma_{\mathrm{COL}},B}}$, defined by $h/{\approx_{\Sigma_{\mathrm{COL}}}}([a]) = [b]$ if $a\ h\ b$, is a Σ-morphism. Moreover, for each Σ-morphism $k : \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_\Sigma/{\approx_{\Sigma_{\mathrm{COL}},A}} \to \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_\Sigma/{\approx_{\Sigma_{\mathrm{COL}},B}}$, there exists a unique Σ_{COL}-morphism $h : A \to B$ such that $h/{\approx_{\Sigma_{\mathrm{COL}}}} = k$.

Proof. The properties of COL-morphisms imply that $h/{\approx_{\Sigma_{\mathrm{COL}}}}$ is a well-defined Σ-morphism. For proving the second part of the lemma assume that $k : \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_\Sigma/{\approx_{\Sigma_{\mathrm{COL}},A}} \to \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_\Sigma/{\approx_{\Sigma_{\mathrm{COL}},B}}$ is a Σ-morphism. Then k induces a family of relations $h_s \subseteq \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s} \times \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_{\Sigma,s}$ such that for all $a \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s}$, $b \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_{\Sigma,s}$ we have $a\ h_s\ b$ if and only if $k_s([a]) = [b]$. It is straightforward to show that h is indeed a Σ_{COL}-morphism between A and B such that $h/{\approx_{\Sigma_{\mathrm{COL}}}} = k$. For proving the uniqueness of h let $h' : A \to B$ be a Σ_{COL}-morphism with $h'/{\approx_{\Sigma_{\mathrm{COL}}}} = k$. Then, for any $a \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma,s}$, $b \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_{\Sigma,s}$, $a\ h_s\ b$ iff $k_s([a]) = [b]$ iff $h'/{\approx_{\Sigma_{\mathrm{COL}}}}([a]) = [b]$ iff $a\ h'_s\ b$. □

Definition 28 (Category of COL-algebras). *For any COL-signature Σ_{COL}, the class $\mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$ together with the Σ_{COL}-morphisms defines a category which, by abuse of notation, will also be denoted by $\mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$. The composition of Σ_{COL}-morphisms is the usual composition of relations and for each $A \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$, the identity $id_A : A \to A$ is the reduct ${\approx_{\Sigma_{\mathrm{COL}},A}}|_{\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_\Sigma}$ of the observational equality $\approx_{\Sigma_{\mathrm{COL}},A}$ to the subalgebra $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_\Sigma$.*[4]

Using the black box construction of Definition 24 one can relate, for any COL-signature Σ_{COL}, the category $\mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$ of Σ_{COL}-algebras and the category $\mathrm{Alg}(\Sigma)$ of (standard) Σ-algebras by a functor which associates to any COL-algebra its black box view. According to Lemma 27 this functor establishes a one to one correspondence between COL-morphisms and standard morphisms, i.e., it is full and faithful.

Definition 29 (Black box functor). *For any COL-signature Σ_{COL}, the* black box functor $\mathcal{BB}_{\Sigma_{\mathrm{COL}}} : \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}}) \to \mathrm{Alg}(\Sigma)$ *is the full and faithful functor defined by:*

[3] Hence COL-morphisms could have been defined also directly as standard morphisms between the black box views of two COL-algebras A and B. We prefer, however, an explicit definition on the carriers of A and B and to distinguish clearly between the category of COL-algebras and the one of standard algebras.
[4] It is easy to prove that all required properties of a category are indeed satisfied.

(1) For each $A \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$, $\mathcal{BB}_{\Sigma_{\mathrm{COL}}}(A) \overset{\mathrm{def}}{=} \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_\Sigma / \approx_{\Sigma_{\mathrm{COL}},A}$.

(2) For each Σ_{COL}*-morphism* $h : A \to B$, $\mathcal{BB}_{\Sigma_{\mathrm{COL}}}(h) \overset{\mathrm{def}}{=} h/\approx_{\Sigma_{\mathrm{COL}}}$ *where* $h/\approx_{\Sigma_{\mathrm{COL}}} : \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_\Sigma / \approx_{\Sigma_{\mathrm{COL}},A} \to \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(B)\rangle_\Sigma / \approx_{\Sigma_{\mathrm{COL}},B}$ *is defined in Lemma 27.*

Remark 30. Two isomorphic COL-algebras can be considered to be behaviorally equivalent since they have (up to standard isomorphism) the same black box view. Indeed two COL-algebras are COL-isomorphic if and only if they are observationally equivalent in the sense of [9] with respect to the observational equivalence relation $\equiv_{S_{\mathrm{Obs}},S_{\mathrm{Loose}}}$ between algebras (see Example 4.4 in [9]).

Fact 31. *The black box view of any* COL*-algebra* A *(which is a reachable and fully abstract algebra, and hence is also a* COL*-algebra, see Facts 25 and 22), is* COL*-isomorphic to* A*.*

3.3 COL-Satisfaction Relation and Basic COL-Specifications

In the next step we generalize the standard satisfaction relation of first-order logic by abstracting with respect to reachability and observability. First, from the reachability point of view, the valuations of variables are restricted to the elements of the Σ_{COL}-generated part only.[5] From the observability point of view, the idea is to interpret the equality symbol "=" occurring in a first-order formula φ not by the set-theoretic equality but by the observational equality of elements.

Definition 32 (COL-satisfaction relation). *For any* COL*-signature* Σ_{COL}*, the* COL-satisfaction relation *between* Σ*-algebras and (finitary) first-order* Σ*-formulas (with variables in* X*) is denoted by* $\models_{\Sigma_{\mathrm{COL}}}$ *and defined as follows. Let* $A \in \mathrm{Alg}(\Sigma)$.

(1) For any two terms $t, r \in T_\Sigma(X)_s$ *of the same sort* s *and for any valuation* $\alpha : X \to \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)$, $A, \alpha \models_{\Sigma_{\mathrm{COL}}} t = r$ *holds if* $I_\alpha(t) \approx_{\Sigma_{\mathrm{COL}},A} I_\alpha(r)$.

(2) For any arbitrary Σ*-formula* φ *and for any valuation* $\alpha : X \to \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)$, $A, \alpha \models_{\Sigma_{\mathrm{COL}}} \varphi$ *is defined by induction over the structure of the formula* φ *in the usual way. In particular,* $A, \alpha \models_{\Sigma_{\mathrm{COL}}} \forall x{:}s.\, \varphi$ *if for all valuations* $\beta : X \to \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)$ *with* $\beta(y) = \alpha(y)$ *for all* $y \neq x$, $A, \beta \models_{\Sigma_{\mathrm{COL}}} \varphi$.

(3) For any arbitrary Σ*-formula* φ, $A \models_{\Sigma_{\mathrm{COL}}} \varphi$ *holds if for all valuations* $\alpha : X \to \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)$, $A, \alpha \models_{\Sigma_{\mathrm{COL}}} \varphi$ *holds.*

The notation $A \models_{\Sigma_{\mathrm{COL}}} \varphi$ is extended in the usual way to classes of algebras and sets of formulas.

[5] This idea is related to the ultra-loose approach of [40] where the same effect is achieved by using formulas with relativized quantification.

Remark 33. The COL-satisfaction relation is defined for arbitrary Σ-algebras and hence is also defined for Σ_{COL}-algebras.[6] In the case of Σ_{COL}-algebras the COL-satisfaction relation would be the same if we would have used in the above definition valuations "$\alpha : X \to \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}$" (with values in the Σ_{COL}-generated subalgebra) instead of valuations "$\alpha : X \to \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)$" (with values in the Σ_{COL}-generated part).

The next theorem shows that the black box functor is compatible with the COL-satisfaction relation and the standard satisfaction relation.

Theorem 34. *Let Σ_{COL} be a* COL*-signature, let φ be a Σ-formula and let A be a Σ_{COL}-algebra. Then:*

$$A \models_{\Sigma_{\mathrm{COL}}} \varphi \textit{ if and only if } \mathcal{BB}_{\Sigma_{\mathrm{COL}}}(A) \models_{\Sigma} \varphi\,.$$[7]

Proof. Let A be a Σ_{COL}-algebra. The restriction of the Σ_{COL}-equality $\approx_{\Sigma_{\mathrm{COL}},A}$ to the COL-generated subalgebra $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A)\rangle_{\Sigma}$ of A is, trivially, a partial Σ-congruence on A in the sense of [9]. Hence, taking into account Remark 33, we can apply Theorem 3.11 of [9] and obtain the desired result.[8] □

Fact 35. *Let Σ_{COL} be a* COL*-signature, let φ be a Σ-formula and let A be a Σ-algebra which is reachable and fully abstract w.r.t. Σ_{COL}. Then:*

$$A \models_{\Sigma_{\mathrm{COL}}} \varphi \textit{ if and only if } A \models \varphi\,.$$

Definition 36 (Basic COL-specification). *A basic* COL*-specification $\mathrm{SP}_{\mathrm{COL}} = \langle \Sigma_{\mathrm{COL}}, \mathrm{Ax}\rangle$ consists of a* COL*-signature Σ_{COL} and a set* Ax *of Σ-sentences, called axioms. The semantics of $\mathrm{SP}_{\mathrm{COL}}$ is given by its signature $\mathcal{S}ig[\mathrm{SP}_{\mathrm{COL}}]$ and by its class of models $\mathcal{M}od[\mathrm{SP}_{\mathrm{COL}}]$ which are defined by:*

$$\mathcal{S}ig[\mathrm{SP}_{\mathrm{COL}}] \overset{\mathrm{def}}{=} \Sigma_{\mathrm{COL}}$$
$$\mathcal{M}od[\mathrm{SP}_{\mathrm{COL}}] \overset{\mathrm{def}}{=} \{A \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}}) \mid A \models_{\Sigma_{\mathrm{COL}}} \mathrm{Ax}\}$$

According to the flexible COL-satisfaction relation, the model class of a COL-specification $\mathrm{SP}_{\mathrm{COL}}$ describes all algebras which can be considered as correct realizations of $\mathrm{SP}_{\mathrm{COL}}$.

Example 37. The following specification extends the COL-signature of Example 4 by appropriate axioms for containers of natural numbers.[9]

[6] The more general definition for arbitrary Σ-algebras is useful when considering refinement relations which are beyond the scope of this paper.
[7] When it is clear from the context we often write $\models$ instead of $\models_{\Sigma}$ to denote the standard satisfaction relation.
[8] Similar results are provided e.g. in [23].
[9] We use here a syntactic sugar similar to the one of Casl [1].

spec CONTAINER =

  sorts *bool*, *nat*, *container*

  ops $true$, $false : bool$;

   $0 : nat$; $succ : nat \to nat$; $add : nat \times nat \to nat$;

   $empty : container$; $insert : container \times nat \to container$;

   $remove : container \times nat \to container$;

   $isin : container \times nat \to bool$;

  constructors *true*, *false*, *0*, *succ*, *empty*, *insert*

  observer $(isin, 1)$

  axioms

  $\forall x, y : nat;\ c : container$

  %% standard axioms for booleans and natural numbers, plus

- $isin(empty, x) = false$ (1)
- $isin(insert(c, x), x) = true$ (2)
- $x \neq y \Rightarrow isin(insert(c, y), x) = isin(c, x)$ (3)
- $remove(empty, x) = empty$ (4)
- $remove(insert(c, x), x) = remove(c, x)$ (5)
- $x \neq y \Rightarrow remove(insert(c, y), x) = insert(remove(c, x), y)$ (6)

end

It is important to note that the declaration of constructors and observers leads to corresponding specification methods. As usual, non-constructor operations can be defined by a complete case distinction w.r.t. the given constructors. For instance, the axioms (1) - (3) define the non-constructor *isin* by a complete case analysis w.r.t. *empty* and *insert* and, similarly, *remove* is specified by a constructor complete definition according to the axioms (4) - (6).

On the other hand, also the observers give rise to a specification method whereby the observable effect of the non-observer operations can be defined by a complete case distinction w.r.t. the given observers. For instance, axiom (1) can be considered as an observer complete definition of *empty* and axioms (2) and (3) can be considered as an observer complete definition of *insert* (see [5] for a general schema of observer complete definitions). Thus the axioms (1) - (3) can be seen from both sides, from the observational or from the reachability point of view, the result is the same.

However, this is not the case for the axioms (4) - (6) that specify *remove* (which is neither a constructor nor an observer). In this case we have chosen a constructor style, but we can ask whether we couldn't use just as well an observer style with the same semantic result. Indeed it is simple to provide an observer complete definition of *remove* by the following two formulas:

- $isin(remove(c, x), x) = false$ (7)
- $x \neq y \Rightarrow isin(remove(c, x), y) = isin(c, y)$ (8)

Obviously, with a standard interpretation, the formulas (7) and (8) are quite different from the axioms (4) - (6). However, in the COL framework developed in this paper it turns out that indeed the axioms (4) - (6) could be replaced by the formulas (7) and (8) without changing the semantics of the container specification. A formal proof of this fact will be provided in Section 7, Example 85.

Let us still point out that the container algebra A of Example 9 is a model of CONTAINER. Thereby it is essential that the COL-satisfaction relation interprets the equality symbol by the observational equality. Otherwise, axiom (5) would not be satisfied by A. For instance, if we interpret c by the empty container $(<>,<>)$ and x by 1, we have:
$\mathit{remove}^A(\mathit{insert}^A((<>,<>),1),1) = \mathit{remove}^A((<1>,<>),1) = (<>,<1>)$
and $\mathit{remove}^A((<>,<>),1) = (<>,<>)$
where the results $(<>,<1>)$ and $(<>,<>)$ are not the same but are observationally equal.

On the other hand, if we would use (7) and (8) for specifying *remove* then it is essential that the COL-satisfaction relation interprets variables by values in the Σ_{COL}-generated part.[10] Otherwise, axiom (7) would not be satisfied by the container algebra A. For instance, if we would interpret c by the non reachable container $(<1,1>,<>)$ and x by 1, we would obtain:
$\mathit{isin}^A(\mathit{remove}^A((<1,1>,<>),1),1) = \mathit{isin}^A((<1>,<1>),1) = \mathit{true}$. $\diamond$

The model class $\mathcal{M}od[\mathrm{SP}_{\mathrm{COL}}]$ of a COL-specification $\mathrm{SP}_{\mathrm{COL}}$ reflects all its correct realizations. In the following we will refer to $\mathcal{M}od[\mathrm{SP}_{\mathrm{COL}}]$ as the *glass box semantics* of the specification $\mathrm{SP}_{\mathrm{COL}}$. Glass box semantics is appropriate from an implementor's point of view. Of equal importance, from a user's point of view, are the logical consequences of a given specification.

Definition 38 (COL-theorem). *Let* $\mathrm{SP}_{\mathrm{COL}} = \langle \Sigma_{\mathrm{COL}}, \mathrm{Ax} \rangle$ *be a basic* COL-*specification. A* Σ*-sentence* φ *is called a* COL-theorem *of* $\mathrm{SP}_{\mathrm{COL}}$*, denoted by* $\mathrm{SP}_{\mathrm{COL}} \models_{\Sigma_{\mathrm{COL}}} \varphi$*, if* $\mathcal{M}od[\mathrm{SP}_{\mathrm{COL}}] \models_{\Sigma_{\mathrm{COL}}} \varphi$.

For the consideration of COL-theorems it is convenient to *abstract* the models of a specification into "idealized" models, such that the consequences of the actual models of a COL-specification are exactly the consequences of its idealized models, in *standard* first-order logic. An appropriate representation of the idealized models is provided by the class of all black box views of the models of a given COL-specification. This class will be called the *black box semantics* of the specification. Black box semantics is appropriate from a client's point of view.

Definition 39 (Black box semantics). *Let* $\mathrm{SP}_{\mathrm{COL}} = \langle \Sigma_{\mathrm{COL}}, \mathrm{Ax} \rangle$ *be a basic* COL-*specification. Its* black box semantics *is defined by* $[\![\mathrm{SP}_{\mathrm{COL}}]\!] \overset{\mathrm{def}}{=} \mathrm{Iso}_\Sigma(\mathcal{BB}_{\Sigma_{\mathrm{COL}}}(\mathcal{M}od[\mathrm{SP}_{\mathrm{COL}}]))$*, where* $\mathrm{Iso}_\Sigma(\_)$ *denotes the closure under* Σ*-isomorphisms in* $\mathrm{Alg}(\Sigma)$.

[10] To our knowledge, the only approaches which allow this kind of relativization are the ultra-loose approach [40] and the constructor-based institution [8].

As an obvious consequence of Theorem 34 we obtain the following characterization of COL-theorems which shows the adequacy of the black box semantics.

Theorem 40 (COL-theorems). *Let* $\mathrm{SP}_{\mathrm{COL}} = \langle \Sigma_{\mathrm{COL}}, \mathrm{Ax}\rangle$ *be a basic* COL-*specification and let* φ *be a* Σ*-sentence. Then:*

$$\mathrm{SP}_{\mathrm{COL}} \models_{\Sigma_{\mathrm{COL}}} \varphi \textit{ if and only if } [\![\mathrm{SP}_{\mathrm{COL}}]\!] \models \varphi\,.$$

The next theorem provides a characterization of the black box semantics of basic COL-specifications.

Theorem 41 (Black box semantics relies on reachable fully abstract models). *Let* $\mathrm{SP}_{\mathrm{COL}} = \langle \Sigma_{\mathrm{COL}}, \mathrm{Ax}\rangle$ *be a basic* COL*-specification.*
$[\![\mathrm{SP}_{\mathrm{COL}}]\!] = \{\Sigma{-}\textit{algebra } A \mid A \models \mathrm{Ax} \textit{ and } A \textit{ is both reachable and fully abstract w.r.t. } \Sigma_{\mathrm{COL}}\}$*.*[11]

Proof. Let A be a Σ-algebra.
$\subseteq$: Assume $A \in [\![\mathrm{SP}_{\mathrm{COL}}]\!]$. Then A is isomorphic to $\mathcal{BB}_{\Sigma_{\mathrm{COL}}}(B)$ for some $B \in \mathcal{M}od[\mathrm{SP}_{\mathrm{COL}}]$. $\mathcal{BB}_{\Sigma_{\mathrm{COL}}}(B)$ is reachable and fully abstract w.r.t. Σ_{COL} (see Fact 25), hence so is A. Moreover, since $B \models_{\Sigma_{\mathrm{COL}}} \mathrm{Ax}$, by Theorem 34, $\mathcal{BB}_{\Sigma_{\mathrm{COL}}}(B) \models \mathrm{Ax}$, and so does the isomorphic algebra A.
$\supseteq$: Assume $A \models \mathrm{Ax}$ and A is both reachable and fully abstract w.r.t. Σ_{COL}. Then $A \models_{\Sigma_{\mathrm{COL}}} \mathrm{Ax}$ (see Fact 35) and A is a Σ_{COL}-algebra (see Fact 22). Hence $A \in \mathcal{M}od[\mathrm{SP}_{\mathrm{COL}}]$. Since A is both reachable and fully abstract w.r.t. Σ_{COL}, A is isomorphic to $\mathcal{BB}_{\Sigma_{\mathrm{COL}}}(A)$ (see Fact 25), hence $A \in [\![\mathrm{SP}_{\mathrm{COL}}]\!]$. □

For instance, the black box semantics of the container specification given in Example 37 is (up to isomorphism) the algebra of finite sets of natural numbers.

4 The Constructor-based Observational Logic Institution COL

The definitions stated in the last section provide the basic ingredients for defining the *constructor-based observational logic institution*, called COL.

[11] An infinitary axiomatic characterization of this class could be given by using the infinitary axiomatizations of reachability and full abstractness considered in Section 4.4.

For the definition of the COL institution it is particularly important to use an appropriate morphism notion for COL-signatures which guarantees encapsulation of properties with respect to the COL-satisfaction relation (formally expressed by the satisfaction condition of institutions, see [16]). To ensure that the satisfaction condition holds, the crucial idea is to require that neither "new" constructors nor "new" observers are introduced for "old" sorts when composing systems via signature morphisms. Then, on the one hand, the set of constructor terms for constructing elements of "old" sorts remains unchanged (up to renaming) and so does the Σ_{COL}-generated part. On the other hand, also the set of observable contexts for observing "old" sorts remains unchanged (up to renaming) and so does the observational equality. These facts are formally stated in Lemma 45 and 47 below.

Definition 42 (COL-signature morphism). *Let* $\Sigma_{\text{COL}} = (\Sigma, \text{OP}_{\text{Cons}}, \text{OP}_{\text{Obs}})$ *and* $\Sigma'_{\text{COL}} = (\Sigma', \text{OP}'_{\text{Cons}}, \text{OP}'_{\text{Obs}})$ *be two* COL*-signatures with* $\Sigma = (S, \text{OP})$ *and* $\Sigma' = (S', \text{OP}')$*. A* COL-signature morphism $\sigma_{\text{COL}} : \Sigma_{\text{COL}} \to \Sigma'_{\text{COL}}$ *is a signature morphism* $\sigma : \Sigma \to \Sigma'$ *such that:*

(1) If $op \in \text{OP}_{\text{Cons}}$*, then* $\sigma(op) \in \text{OP}'_{\text{Cons}}$.
(2) If $op' \in \text{OP}'_{\text{Cons}}$ *with* $op' : s'_1, \ldots, s'_n \to s'$ *and* $s' \in \sigma(S)$*, then for all* $s \in S$ *such that* $\sigma(s) = s'$*, there exists* $op \in \text{OP}_{\text{Cons}}$ *with* $op : s_1, \ldots, s_n \to s$ *such that* $op' = \sigma(op)$*.*
(3) If $(op, i) \in \text{OP}_{\text{Obs}}$*, then* $(\sigma(op), i) \in \text{OP}'_{\text{Obs}}$.
(4) If $(op', i) \in \text{OP}'_{\text{Obs}}$ *with* $op' : s'_1, \ldots, s'_i, \ldots, s'_n \to s'$ *and* $s'_i \in \sigma(S)$*, then for all* $s_i \in S$ *such that* $\sigma(s_i) = s'_i$*, there exists* $op \in \text{OP}$ *with* $op : s_1, \ldots, s_i, \ldots, s_n \to s$ *such that* $(op, i) \in \text{OP}_{\text{Obs}}$ *and* $op' = \sigma(op)$*.*

As a consequence of the definition, for all $s \in S$, the following holds:
$s \in S_{\text{Cons}}$ if and only if $\sigma(s) \in S'_{\text{Cons}}$, $s \in S_{\text{Loose}}$ if and only if $\sigma(s) \in S'_{\text{Loose}}$,
$s \in S_{\text{State}}$ if and only if $\sigma(s) \in S'_{\text{State}}$, $s \in S_{\text{Obs}}$ if and only if $\sigma(s) \in S'_{\text{Obs}}$.

We implicitly assume in the following that whenever we consider a COL-signature morphism $\sigma_{\text{COL}} : \Sigma_{\text{COL}} \to \Sigma'_{\text{COL}}$, then the underlying signature morphism is $\sigma : \Sigma \to \Sigma'$.

Definition 43 (Category of COL-signatures). COL*-signatures together with* COL*-signature morphisms define a category which will be denoted by* Sign_{COL}.

Lemma 44. *The category* Sign_{COL} *has pushouts.*

Proof. Obviously the properties of a category are satisfied. To show the existence of pushouts let $\sigma_{1,\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma_{1,\mathrm{COL}}$ and $\sigma_{2,\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma_{2,\mathrm{COL}}$ be COL-signature morphisms with underlying signature morphisms $\sigma_1 : \Sigma \to \Sigma_1$ and $\sigma_2 : \Sigma \to \Sigma_2$. It is well-known that in the category of algebraic signatures there exists a pushout as shown in the following diagram.

$$\begin{tikzcd}
\Sigma \arrow[r, "\sigma_1"] \arrow[d, "\sigma_2"'] & \Sigma_1 \arrow[d, "\sigma_1'"] \\
\Sigma_2 \arrow[r, "\sigma_2'"] & \Sigma'
\end{tikzcd}$$

Now let $\mathrm{OP}'_{\mathrm{Cons}} = \{\sigma_1'(op_1) \mid op_1 \in \mathrm{OP}_{1,\mathrm{Cons}}\} \cup \{\sigma_2'(op_2) \mid op_2 \in \mathrm{OP}_{2,\mathrm{Cons}}\}$, let $\mathrm{OP}'_{\mathrm{Obs}} = \{(\sigma_1'(op_1), i) \mid (op_1, i) \in \mathrm{OP}_{1,\mathrm{Obs}}\} \cup \{(\sigma_2'(op_2), i) \mid (op_2, i) \in \mathrm{OP}_{2,\mathrm{Obs}}\}$, and let $\Sigma'_{\mathrm{COL}} = (\Sigma', \mathrm{OP}'_{\mathrm{Cons}}, \mathrm{OP}'_{\mathrm{Obs}})$. It is straightforward to prove that σ_1' and σ_2' give rise to COL-signature morphisms $\sigma'_{1,\mathrm{COL}}$ and $\sigma'_{2,\mathrm{COL}}$ such that the following diagram is a pushout in the category of COL-signatures.

$$\begin{tikzcd}
\Sigma_{\mathrm{COL}} \arrow[r, "\sigma_{1,\mathrm{COL}}"] \arrow[d, "\sigma_{2,\mathrm{COL}}"'] & \Sigma_{1,\mathrm{COL}} \arrow[d, "\sigma'_{1,\mathrm{COL}}"] \\
\Sigma_{2,\mathrm{COL}} \arrow[r, "\sigma'_{2,\mathrm{COL}}"] & \Sigma'_{\mathrm{COL}}
\end{tikzcd}$$

□

4.2 The COL Institution

The next three lemmas are crucial for defining the reduct functor on classes of COL-algebras and for proving the COL-satisfaction condition. The first lemma shows that Σ_{COL}-generated parts of algebras are compatible with reducts along COL-signature morphisms.

Lemma 45. *For any* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$ *and for any* Σ'*-algebra* $A' \in \mathrm{Alg}(\Sigma')$*, we have* $\mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')|_\sigma = \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A'|_\sigma)$.

In the above lemma the Σ'_{COL}-generated part $\mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')$ of A' is considered as an S'-sorted unary relation, $\mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')|_\sigma$ is the reduct of this relation w.r.t. σ (see Section 2.1) and $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A'|_\sigma)$ is the Σ_{COL}-generated part of the reduct $A'|_\sigma$. Thus the lemma states an equation between S-sorted sets.

Proof. If $s \in S_{\mathrm{Loose}}$ then $\sigma(s) \in S'_{\mathrm{Loose}}$ and conversely. Hence, in this case, $(\mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')|_\sigma)_s = \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')_{\sigma(s)} = A'_{\sigma(s)} = (A'|_\sigma)_s = \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A'|_\sigma)_s$.

If $s \in S_{\mathrm{Cons}}$ then $\sigma(s) \in S'_{\mathrm{Cons}}$ and conversely. In this case, the conditions *(1)* and *(2)* of Definition 42 imply that for any constructor term $t' \in \mathcal{T}(\Sigma'_{\mathrm{COL}})_{\sigma(s)}$, one can construct a corresponding constructor term $t \in \mathcal{T}(\Sigma_{\mathrm{COL}})_s$ and vice versa. Hence one can conclude that $(\mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')|_\sigma)_s = \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')_{\sigma(s)} = \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A'|_\sigma)_s$. □

Lemma 45 cannot be generalized to Σ_{COL}-generated subalgebras. The reason is that, given a COL-signature morphism $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$, it may be the case that for some constrained sort $s \in S_{\mathrm{Cons}}$ there exists an operation symbol $op' \in \mathrm{OP}'$ with $op' : s'_1, \ldots, s'_n \to \sigma(s)$ such that op' is not in the image of the underlying signature morphism σ. (Of course, due to the properties of COL-signature morphisms, this can only be the case if op' is neither a constructor nor an observer.) In this case the interpretation of op' in a Σ'-algebra A' may lead to elements which belong to the Σ'_{COL}-generated subalgebra of A' but not to the Σ_{COL}-generated subalgebra of its reduct $A'|_\sigma$. Thus only the direction "$\supseteq$" of Lemma 45 can be (trivially) propagated to Σ_{COL}-generated subalgebras.

Lemma 46. *For any* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$ *and for any* Σ'*-algebra* $A' \in \mathrm{Alg}(\Sigma')$*, we have* $\langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A') \rangle_{\Sigma'}|_\sigma \supseteq \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A'|_\sigma) \rangle_\Sigma$.

The next lemma shows that observational Σ_{COL}-equalities are compatible with reducts along COL-signature morphisms.

Lemma 47. *For any* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$ *and for any* Σ'*-algebra* $A' \in \mathrm{Alg}(\Sigma')$*, we have* $(\approx_{\Sigma'_{\mathrm{COL}},A'})|_\sigma = \approx_{\Sigma_{\mathrm{COL}},(A'|_\sigma)}$.

In this lemma the observational Σ'_{COL}-equality $\approx_{\Sigma'_{\mathrm{COL}},A'}$ on A' is an S'-sorted binary relation, $(\approx_{\Sigma'_{\mathrm{COL}},A'})|_\sigma$ is the reduct of this relation w.r.t. σ (see Section 2.1) and $\approx_{\Sigma_{\mathrm{COL}},(A'|_\sigma)}$ is the observational Σ_{COL}-equality on the reduct $A'|_\sigma$. Thus the lemma states an equation between S-sorted binary relations.

Proof. For any $s \in S$, $((\approx_{\Sigma'_{\mathrm{COL}},A'})|_\sigma)_s = (\approx_{\Sigma'_{\mathrm{COL}},A'})_{\sigma(s)}$ and $(A'|_\sigma)_s = A'_{\sigma(s)}$. Hence it is sufficient to prove that for all $a, b \in A'_{\sigma(s)}$, $a \approx_{\Sigma'_{\mathrm{COL}},A'} b$ iff $a \approx_{\Sigma_{\mathrm{COL}},(A'|_\sigma)} b$.
If $s \in S_{\mathrm{Obs}}$ then $\sigma(s) \in S'_{\mathrm{Obs}}$ and conversely. Hence, in this case, $a \approx_{\Sigma'_{\mathrm{COL}},A'} b$ iff $a = b$ iff $a \approx_{\Sigma_{\mathrm{COL}},(A'|_\sigma)} b$. If $s \in S_{\mathrm{State}}$ then $\sigma(s) \in S'_{\mathrm{State}}$ and conversely. In this case, the conditions *(3)* and *(4)* of Definition 42 imply that for any observable context $c' \in \mathcal{C}(\Sigma'_{\mathrm{COL}})$ with application sort $\sigma(s)$ one can construct a corresponding observable context $c \in \mathcal{C}(\Sigma_{\mathrm{COL}})$ with application sort s and vice versa. All variables occurring in c' (different from $z_{\sigma(s)}$) are interpreted by values in the Σ'_{COL}-generated part $\mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')$ and all variables occurring in c (different from z_s) are interpreted by values in the Σ_{COL}-generated part $\mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A'|_\sigma)$. Since, by Lemma 45, the generated parts are compatible with the reduct along σ_{COL}, one can conclude $a \approx_{\Sigma'_{\mathrm{COL}},A'} b$ iff $a \approx_{\Sigma_{\mathrm{COL}},(A'|_\sigma)} b$. □

As an obvious consequence of Lemma 45, COL-reduct functors preserve reachability and, as an obvious consequence of Lemma 47, COL-reduct functors preserve full abstractness of algebras.

Corollary 48. *For any* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$ *and for any* Σ'*-algebra* $A' \in \mathrm{Alg}(\Sigma')$*, we have:*

(1) If A' *is reachable w.r.t.* Σ'_{COL} *then* $A'|_\sigma$ *is reachable w.r.t.* Σ_{COL} *.*
(2) If A' *is fully abstract w.r.t.* Σ'_{COL} *then* $A'|_\sigma$ *is fully abstract w.r.t.* Σ_{COL} *.*

As a consequence of Lemmas 45, 46 and 47, we obtain the following theorem which directly leads to the definition of the COL-reduct functor.

Theorem 49. *For any* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$ *and for any* Σ'_{COL}*-algebra* $A' \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}})$*,* $A'|_\sigma$ *satisfies the reachability and observability constraints w.r.t.* Σ_{COL} *, i.e.,* $A'|_\sigma \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$*. Moreover, for any* Σ'_{COL}*-morphism* $h' : A' \to B'$ *the reduct* $h'|_\sigma : A'|_\sigma \to B'|_\sigma$ *is a* Σ_{COL}*-morphism.*

Proof. The proof of the second part of the theorem is straightforward. For the first part, assume that $A' \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}})$. We have to show that $A'|_\sigma$ satisfies the reachability and observability constraints w.r.t. Σ_{COL} . Let us first consider the reachability constraint. Let $a \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A'|_\sigma)\rangle_\Sigma$. Then, by Lemma 46, $a \in \langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')\rangle_{\Sigma'}|_\sigma$ and hence $a \in \langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')\rangle_{\Sigma'}$. Since A' satisfies the reachability constraint w.r.t. Σ'_{COL} , there exists $b \in \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')$ (and hence $b \in \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')|_\sigma$) such that $a \approx_{\Sigma'_{\mathrm{COL}},A'} b$ (and hence $a\ (\approx_{\Sigma'_{\mathrm{COL}},A'})|_\sigma\ b$). By Lemma 45, $b \in \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A'|_\sigma)$ and, by Lemma 47, $a \approx_{\Sigma_{\mathrm{COL}},(A'|_\sigma)} b$. Thus $A'|_\sigma$ satisfies the reachability constraint w.r.t. Σ_{COL} .
For proving the observability constraint, let $a, b \in \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A'|_\sigma)\rangle_\Sigma$. Then, by Lemma 46, $a, b \in \langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')\rangle_{\Sigma'}|_\sigma$ and hence $a, b \in \langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')\rangle_{\Sigma'}$. Since A' satisfies the observability constraint w.r.t. Σ'_{COL} , $a \approx_{\Sigma'_{\mathrm{COL}},A'} b$ (and hence $a\ (\approx_{\Sigma'_{\mathrm{COL}},A'})|_\sigma\ b$). Then, by Lemma 47, $a \approx_{\Sigma_{\mathrm{COL}},(A'|_\sigma)} b$. Thus $A'|_\sigma$ satisfies the observability constraint w.r.t. Σ_{COL} . □

Definition 50 (COL-reduct functor). *For any* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$ *, the* COL-reduct functor $\_|_{\sigma_{\mathrm{COL}}} : \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}}) \to \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$ *is defined as follows.*

(1) For each $A' \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}})$*,* $A'|_{\sigma_{\mathrm{COL}}} \overset{\mathrm{def}}{=} A'|_\sigma$ *.*
(2) For each Σ'_{COL}*-morphism* $h' : A' \to B'$*,* $h'|_{\sigma_{\mathrm{COL}}} \overset{\mathrm{def}}{=} h'|_\sigma$ *.*

As another important consequence of the above lemmas we obtain that the black box functor introduced in Definition 29 commutes with the reduct functor.

Theorem 51 (Black box commutes with reduct). *For any* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$ *and for any* Σ'_{COL}*-algebra* $A' \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}})$, $\mathcal{BB}_{\Sigma'_{\mathrm{COL}}}(A')|_\sigma \;=\; \mathcal{BB}_{\Sigma_{\mathrm{COL}}}(A'|_{\sigma_{\mathrm{COL}}})$.

Proof. Under the given assumptions we have:
$\mathcal{BB}_{\Sigma'_{\mathrm{COL}}}(A')|_\sigma \;=\; (\langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')\rangle_{\Sigma'}/{\approx_{\Sigma'_{\mathrm{COL}},A'}})|_\sigma \;=$
$(\langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')\rangle_{\Sigma'})|_\sigma/({\approx_{\Sigma'_{\mathrm{COL}},A'}})|_\sigma) \;=\;$ (due to Lemma 45 and 47 and due to the fact that A' and $A'|_\sigma$ satisfy their reachability constraints)
$=\; \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A'|_\sigma)\rangle_\Sigma/{\approx_{\Sigma_{\mathrm{COL}},(A'|_\sigma)}} \;=\; \mathcal{BB}_{\Sigma_{\mathrm{COL}}}(A'|_{\sigma_{\mathrm{COL}}})$. □

Theorems 51 and 34 are the essential facts needed to prove the COL-satisfaction condition.

Theorem 52 (COL**-satisfaction condition).** *For any* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$*,* Σ'_{COL}*-algebra* $A' \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}})$ *and* Σ*-sentence* φ*:*

$$A' \models_{\Sigma'_{\mathrm{COL}}} \sigma(\varphi) \textit{ if and only if } A'|_{\sigma_{\mathrm{COL}}} \models_{\Sigma_{\mathrm{COL}}} \varphi\,.$$

Proof. $A' \models_{\Sigma'_{\mathrm{COL}}} \sigma(\varphi)$ iff, by Theorem 34, $\mathcal{BB}_{\Sigma'_{\mathrm{COL}}}(A') \models_{\Sigma'} \sigma(\varphi)$ iff (since the satisfaction condition holds in the standard first-order logic institution) $\mathcal{BB}_{\Sigma'_{\mathrm{COL}}}(A')|_\sigma \models_\Sigma \varphi$ iff, by Theorem 51, $\mathcal{BB}_{\Sigma_{\mathrm{COL}}}(A'|_{\sigma_{\mathrm{COL}}}) \models_\Sigma \varphi$ iff, by Theorem 34, $A'|_{\sigma_{\mathrm{COL}}} \models_{\Sigma_{\mathrm{COL}}} \varphi$. □

We have now defined all the ingredients that constitute the constructor-based observational logic institution COL.

Definition 53 (The COL **institution).** *The institution* COL *is defined as follows:*

- *The category of signatures is the category* $\mathrm{Sign}_{\mathrm{COL}}$ *of* COL*-signatures with* COL*-signature morphisms.*
- *The functor* $\mathrm{Sen}_{\mathrm{COL}} : \mathrm{Sign}_{\mathrm{COL}} \to \mathbf{Set}$ *maps each* COL*-signature* $\Sigma_{\mathrm{COL}} = (\Sigma, \mathrm{OP}_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Obs}})$ *to the set of (finitary) first-order* Σ*-sentences and each* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$ *to the obvious translation function which transforms* Σ*-sentences into* Σ'*-sentences.*
- *The functor* $\mathrm{Mod}_{\mathrm{COL}} : \mathrm{Sign}^{\mathrm{op}}_{COL} \to \mathbf{Cat}$ *maps:*
  $\star$ *each* COL*-signature* Σ_{COL} *to the category* $\mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$ *of* Σ_{COL}*-algebras with* Σ_{COL}*-morphisms;*
  $\star$ *each* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$ *to the* COL*-reduct functor* $\_|_{\sigma_{\mathrm{COL}}} : \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}}) \to \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$*.*
- *For each* COL*-signature* Σ_{COL}*, the satisfaction relation is the* COL*-satisfaction relation* $\models_{\Sigma_{\mathrm{COL}}}$ *between* Σ_{COL}*-algebras and* Σ*-sentences.*

The COL institution provides a suitable framework for instantiating the institution-independent specification-building operators introduced in Section 2.3. Thus we obtain the following set of operations for constructing structured COL-specifications. The class of all COL-specifications is denoted by $\mathrm{SPEC}_{\mathrm{COL}}$ and the semantics of a COL-specification is determined by its COL-signature and by its model class.

***presentation*:** Any basic specification $\langle \Sigma_{\mathrm{COL}}, \mathrm{Ax} \rangle$ in the sense of Definition 36 is a COL-specification with semantics:
$\mathcal{S}ig[\langle \Sigma_{\mathrm{COL}}, \mathrm{Ax} \rangle] \overset{\mathrm{def}}{=} \Sigma_{\mathrm{COL}}$
$\mathcal{M}od[\langle \Sigma_{\mathrm{COL}}, \mathrm{Ax} \rangle] \overset{\mathrm{def}}{=} \{A \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}}) \mid A \models_{\Sigma_{\mathrm{COL}}} \mathrm{Ax}\}$

***union*:** For any two COL-specifications $\mathrm{SP}_{1,\mathrm{COL}}$ and $\mathrm{SP}_{2,\mathrm{COL}}$ with the same signature $\mathcal{S}ig[\mathrm{SP}_{1,\mathrm{COL}}] = \mathcal{S}ig[\mathrm{SP}_{2,\mathrm{COL}}] = \Sigma_{\mathrm{COL}}$, the expression $\mathrm{SP}_{1,\mathrm{COL}} \cup \mathrm{SP}_{2,\mathrm{COL}}$ is a COL-specification with semantics:
$\mathcal{S}ig[\mathrm{SP}_{1,\mathrm{COL}} \cup \mathrm{SP}_{2,\mathrm{COL}}] \overset{\mathrm{def}}{=} \Sigma_{\mathrm{COL}}$
$\mathcal{M}od[\mathrm{SP}_{1,\mathrm{COL}} \cup \mathrm{SP}_{2,\mathrm{COL}}] \overset{\mathrm{def}}{=} \mathcal{M}od[\mathrm{SP}_{1,\mathrm{COL}}] \cap \mathcal{M}od[\mathrm{SP}_{2,\mathrm{COL}}]$

***translation*:** For any COL-specification $\mathrm{SP}_{\mathrm{COL}}$ and COL-signature morphism $\sigma_{\mathrm{COL}} : \mathcal{S}ig[\mathrm{SP}_{\mathrm{COL}}] \to \Sigma_{\mathrm{COL}}$, the expression **translate** $\mathrm{SP}_{\mathrm{COL}}$ **by** σ_{COL} is a COL-specification with semantics:
$\mathcal{S}ig[\textbf{translate } \mathrm{SP}_{\mathrm{COL}} \textbf{ by } \sigma_{\mathrm{COL}}] \overset{\mathrm{def}}{=} \Sigma_{\mathrm{COL}}$
$\mathcal{M}od[\textbf{translate } \mathrm{SP}_{\mathrm{COL}} \textbf{ by } \sigma_{\mathrm{COL}}] \overset{\mathrm{def}}{=}$
$\{A \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}}) \mid A|_{\sigma_{\mathrm{COL}}} \in \mathcal{M}od[\mathrm{SP}_{\mathrm{COL}}]\}$

***hiding*:** For any COL-specification $\mathrm{SP}_{\mathrm{COL}}$ and any COL-signature morphism $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \mathcal{S}ig[\mathrm{SP}_{\mathrm{COL}}]$, the expression **derive from** $\mathrm{SP}_{\mathrm{COL}}$ **by** σ_{COL} is a COL-specification with semantics:
$\mathcal{S}ig[\textbf{derive from } \mathrm{SP}_{\mathrm{COL}} \textbf{ by } \sigma_{\mathrm{COL}}] \overset{\mathrm{def}}{=} \Sigma_{\mathrm{COL}}$
$\mathcal{M}od[\textbf{derive from } \mathrm{SP}_{\mathrm{COL}} \textbf{ by } \sigma_{\mathrm{COL}}] \overset{\mathrm{def}}{=}$
$\mathrm{Iso}_{\Sigma_{\mathrm{COL}}}(\{A|_{\sigma_{\mathrm{COL}}} \mid A \in \mathcal{M}od[\mathrm{SP}_{\mathrm{COL}}]\})$.

Theorem 34 implies that COL-satisfaction is closed under COL-isomorphisms, and hence the model class of any structured specification over COL built with the above specification-building primitives is closed under COL-isomorphisms (see Fact 1).

The notions of a COL-theorem and of the black box semantics of a basic COL-specification (see Definitions 38 and 39) and the characterization of COL-theorems (see Theorem 40) can be generalized to structured COL-specifications in a straightforward way. How to prove COL-theorems of structured COL-specifications will be studied in the second part of this paper.

4.4 Amalgamation and Interpolation

In this section we discuss the amalgamation and interpolation properties in the context of the COL institution. Let us first focus on the amalgamation property as defined, for instance, in [38]. For this purpose we assume given a pushout in the category $\mathrm{Sign}_{\mathrm{COL}}$ of COL-signatures:

$$\begin{tikzcd}
\Sigma_{\mathrm{COL}} \arrow[r, "\sigma_{1,\mathrm{COL}}"] \arrow[d, "\sigma_{2,\mathrm{COL}}"'] & \Sigma_{1,\mathrm{COL}} \arrow[d, "\sigma'_{1,\mathrm{COL}}"] \\
\Sigma_{2,\mathrm{COL}} \arrow[r, "\sigma'_{2,\mathrm{COL}}"] & \Sigma'_{\mathrm{COL}}
\end{tikzcd}$$

The amalgamation property requires that for any two COL-algebras $A_1 \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{1,\mathrm{COL}})$, $A_2 \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{2,\mathrm{COL}})$ such that $A_1|_{\sigma_{1,\mathrm{COL}}} = A_2|_{\sigma_{2,\mathrm{COL}}}$ (i.e., $A_1|_{\sigma_1} = A_2|_{\sigma_2}$), there exists a unique COL-algebra $A' \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}})$ such that $A'|_{\sigma'_{1,\mathrm{COL}}} = A_1$ and $A'|_{\sigma'_{2,\mathrm{COL}}} = A_2$ (i.e., $A'|_{\sigma'_1} = A_1$ and $A'|_{\sigma'_2} = A_2$). The following diagram shows the corresponding reduct functors.

$$\begin{tikzcd}
\mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}}) & \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{1,\mathrm{COL}}) \arrow[l, "\text{--}|_{\sigma_{1,\mathrm{COL}}}"'] \\
\mathrm{Alg}_{\mathrm{COL}}(\Sigma_{2,\mathrm{COL}}) \arrow[u, "\text{--}|_{\sigma_{2,\mathrm{COL}}}"] & \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}}) \arrow[l, "\text{--}|_{\sigma'_{2,\mathrm{COL}}}"'] \arrow[u, "\text{--}|_{\sigma'_{1,\mathrm{COL}}}"']
\end{tikzcd}$$

Since COL-algebras are standard Σ-algebras there is only one choice for A' which is the amalgamated union of A_1 and A_2 considered as Σ_1- and Σ_2-algebras respectively. Then the question is whether A' is a Σ'_{COL}-algebra, i.e., whether A' satisfies the reachability and observability constraints induced by Σ'_{COL}. The following example shows that this is in general not the case.

Example 54. Let $\Sigma_{\mathrm{COL}} = (\Sigma, \mathrm{OP}_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Obs}})$ be the COL-signature of Example 4 without the *remove* operation:

$\Sigma = (S, \mathrm{OP})$, $S = \{\ \mathit{bool},\ \mathit{nat},\ \mathit{container}\ \}$
$\mathrm{OP} = \{\ \mathit{true} : \to \mathit{bool};\ \ \mathit{false} : \to \mathit{bool};$
$\quad 0 : \to \mathit{nat};\ \ \mathit{succ} : \mathit{nat} \to \mathit{nat};\ \ \mathit{add} : \mathit{nat}, \mathit{nat} \to \mathit{nat};$
$\quad \mathit{empty} : \to \mathit{container};\ \ \mathit{insert} : \mathit{container}, \mathit{nat} \to \mathit{container};$
$\quad \mathit{isin} : \mathit{container}, \mathit{nat} \to \mathit{bool}\ \}$

$\mathrm{OP}_{\mathrm{Cons}} = \{\ \mathit{true},\ \mathit{false},\ 0,\ \mathit{succ},\ \mathit{empty},\ \mathit{insert}\ \}$
$\mathrm{OP}_{\mathrm{Obs}} = \{\ (\mathit{isin}, 1)\ \}$

Let $\Sigma_{1,\mathrm{COL}}$ be the COL-signature obtained by adding the operation symbol $\mathit{append} : \mathit{container}, \mathit{nat} \to \mathit{container}$ to Σ_{COL} and let $\Sigma_{2,\mathrm{COL}}$ be the COL-signature obtained by adding the operation symbol $\mathit{remove} : \mathit{container}, \mathit{nat} \to \mathit{container}$ to Σ_{COL}. Both operations, *append* and *remove*, are neither constructors nor observers. Let $\sigma_{1,\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma_{1,\mathrm{COL}}$ and $\sigma_{2,\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma_{2,\mathrm{COL}}$ be the inclusion morphisms.

Let A_2 be the container algebra of Example 9 (considered as a $\Sigma_{2,\mathrm{COL}}$-algebra), and let A_1 be obtained from A_2 by replacing (the interpretation of) *remove* by the following interpretation of *append* (which, in contrast to *insert*, always adds an element to the container):
$\mathit{append}^{A_1}((< a_1, \ldots, a_n >, t), a) = (< a, a_1, \ldots, a_n >, t)$
Obviously, $A_1|_{\sigma_1} = A_2|_{\sigma_2}$, and, for $i = 1, 2$, the Σ_{COL}-generated subalgebra of $A_i|_{\sigma_i}$ coincides with its Σ_{COL}-generated part. In particular, for the sort *container*, $\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A_i|_{\sigma_i}) \rangle_{\Sigma,\mathit{container}} = \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A_i|_{\sigma_i})_{\mathit{container}} = \{(s, <>) \mid s \in \mathbb{N}^*$ and each element of s occurs only once in $s\}$.

Considering the generated subalgebras of A_1 and A_2, we see that the operations *append* and *remove* can produce elements which belong to the generated subalgebras of A_1 and A_2 respectively but not to the Σ_{COL}-generated subalgebra of their reduct $A_i|_{\sigma_i}$. More precisely, we have
$\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A_1|_{\sigma_1}) \rangle_{\Sigma,\mathit{container}} \subsetneq \langle \mathrm{Gen}_{\Sigma_{1,\mathrm{COL}}}(A_1) \rangle_{\Sigma_1,\mathit{container}} = \{(s, <>) \mid s \in \mathbb{N}^*\}$
and, see Example 18,
$\langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A_2|_{\sigma_2}) \rangle_{\Sigma,\mathit{container}} \subsetneq \langle \mathrm{Gen}_{\Sigma_{2,\mathrm{COL}}}(A_2) \rangle_{\Sigma_2,\mathit{container}} =$
$\{(s, t) \mid s, t \in \mathbb{N}^*$ and each element of s occurs only once in $s\}$.

It is easy to check that A_1 is a $\Sigma_{1,\mathrm{COL}}$-algebra. We also know, from Examples 18 and 20, that A_2 is a $\Sigma_{2,\mathrm{COL}}$-algebra. Let us now construct the amalgamated union A' of A_1 and A_2 considered as standard algebras. A' has the same carrier sets as A_1 and A_2 and has interpretations $\mathit{append}^{A'}$ and $\mathit{remove}^{A'}$ for both operations *append* and *remove* whereby $\mathit{append}^{A'} = \mathit{append}^{A_1}$ and $\mathit{remove}^{A'} = \mathit{remove}^{A_2}$. If we construct the Σ'_{COL}-generated subalgebra of A', then we obtain $\langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A') \rangle_{\Sigma',\mathit{container}} = \{(s, t) \mid s, t \in \mathbb{N}^*\}$. Hence, $\langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A') \rangle_{\Sigma',\mathit{container}} \supsetneq \langle \mathrm{Gen}_{\Sigma_{2,\mathrm{COL}}}(A_2) \rangle_{\Sigma_2,\mathit{container}}$ and therefore the observability constraint satisfied by A_2 can not be propagated to A'. Indeed, $\approx_{\Sigma'_{\mathrm{COL}},A'}$ is not a Σ'-congruence on $\langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A') \rangle_{\Sigma'}$ because $\mathit{remove}^{A'}$ does not respect the observational equality for *all* elements of $\langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A') \rangle_{\Sigma'}$. For instance, $(< 1, 1 >, <>)$ and $(< 1 >, <>)$ are observationally equivalent elements of $\langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A') \rangle_{\Sigma'}$ but, see Example 20, $\mathit{remove}^{A'}((< 1, 1 >, <>), 1)$ $= (< 1 >, < 1 >)$ is *not* observationally equal to $\mathit{remove}^{A'}((< 1 >, <>), 1)$ $= (<>, < 1 >)$. $\diamond$

The problem exhibited in the above example is that, in general, reducts along

COL-signature morphisms are not compatible with generated subalgebras (see also Lemma 46). In practice, however, when building large systems from smaller ones, one would require persistent constructions (as for the semantics of specifications of generic units in CASL architectural specifications [11]) which, in the context of the COL institution, should be compatible with generated subalgebras. Then the amalgamated union of two COL-algebras exists as shown in the next theorem.

Theorem 55 (COL-Amalgamation). *Let be given a pushout diagram as depicted above. Let* $A_1 \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{1,\mathrm{COL}})$ *and* $A_2 \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{2,\mathrm{COL}})$ *such that* $A_1|_{\sigma_{1,\mathrm{COL}}} = A_2|_{\sigma_{2,\mathrm{COL}}}$. *If* $\langle \mathrm{Gen}_{\Sigma_{1,\mathrm{COL}}}(A_1)\rangle_{\Sigma_1}|_{\sigma_1} = \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A_1|_{\sigma_1})\rangle_{\Sigma}$ *and* $\langle \mathrm{Gen}_{\Sigma_{2,\mathrm{COL}}}(A_2)\rangle_{\Sigma_2}|_{\sigma_2} = \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A_2|_{\sigma_2})\rangle_{\Sigma}$ *then there exists a unique amalgamated union* $A' \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}})$ *of* A_1 *and* A_2.

Proof. Let $A_1 \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{1,\mathrm{COL}})$, $A_2 \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{2,\mathrm{COL}})$ such that $A_1|_{\sigma_1} = A_2|_{\sigma_2}$ and let $A' \in \mathrm{Alg}(\Sigma')$ be the amalgamated union of A_1 and A_2 in the sense of standard algebras. Then $A'|_{\sigma'_1} = A_1$ and $A'|_{\sigma'_2} = A_2$. Using the assumption, we have $\langle \mathrm{Gen}_{\Sigma_{1,\mathrm{COL}}}(A_1)\rangle_{\Sigma_1}|_{\sigma_1} = \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A_1|_{\sigma_1})\rangle_{\Sigma} = \langle \mathrm{Gen}_{\Sigma_{\mathrm{COL}}}(A_2|_{\sigma_2})\rangle_{\Sigma} = \langle \mathrm{Gen}_{\Sigma_{2,\mathrm{COL}}}(A_2)\rangle_{\Sigma_2}|_{\sigma_2}$. Hence the reducts of the generated subalgebras of A_1 and A_2 coincide. From this we want to conclude:
(1) $\langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')\rangle_{\Sigma'}|_{\sigma'_1} = \langle \mathrm{Gen}_{\Sigma_{1,\mathrm{COL}}}(A_1)\rangle_{\Sigma_1}$ and
(2) $\langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')\rangle_{\Sigma'}|_{\sigma'_2} = \langle \mathrm{Gen}_{\Sigma_{2,\mathrm{COL}}}(A_2)\rangle_{\Sigma_2}$.
W.l.o.g. we prove (1): According to Lemma 46 and taking into account $A'|_{\sigma'_1} = A_1$ only the direction "$\subseteq$" is interesting.
Let $s_1 \in S_1$ and let $a_1 \in \langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')\rangle_{\Sigma'}|_{\sigma'_1}$ be an element of sort s_1. Then $a_1 \in \langle \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')\rangle_{\Sigma'}$ (considered as an element of sort $\sigma'_1(s_1)$). If $a_1 \in \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')$ then, by Lemma 45, $a_1 \in \mathrm{Gen}_{\Sigma_{1,\mathrm{COL}}}(A_1)$ and therefore $a_1 \in \langle \mathrm{Gen}_{\Sigma_{1,\mathrm{COL}}}(A_1)\rangle_{\Sigma_1}$. Otherwise, let $op : s \to \sigma'_1(s_1) \in \mathrm{OP}'$ be, w.l.o.g., a unary operation such that $a_1 = op^{A'}(a)$ for some $a \in \mathrm{Gen}_{\Sigma'_{\mathrm{COL}}}(A')$. If $op \in \sigma'_1(\mathrm{OP}_1)$ then $s \in \sigma'_1(S_1)$ and therefore, by Lemma 45, $a \in \mathrm{Gen}_{\Sigma_{1,\mathrm{COL}}}(A_1)$. Hence $a_1 \in \langle \mathrm{Gen}_{\Sigma_{1,\mathrm{COL}}}(A_1)\rangle_{\Sigma_1}$. If $op \in \sigma'_2(\mathrm{OP}_2)$ then $s \in \sigma'_2(S_2)$ and therefore, by Lemma 45, $a \in \mathrm{Gen}_{\Sigma_{2,\mathrm{COL}}}(A_2)$. Hence $a_1 \in \langle \mathrm{Gen}_{\Sigma_{2,\mathrm{COL}}}(A_2)\rangle_{\Sigma_2}$. Since in this case s_1 is a shared sort, we have $a_1 \in \langle \mathrm{Gen}_{\Sigma_{2,\mathrm{COL}}}(A_2)\rangle_{\Sigma_2}|_{\sigma_2}$ and thus, under the given assumption, also $a_1 \in \langle \mathrm{Gen}_{\Sigma_{1,\mathrm{COL}}}(A_1)\rangle_{\Sigma_1}|_{\sigma_1}$. Hence $a_1 \in \langle \mathrm{Gen}_{\Sigma_{1,\mathrm{COL}}}(A_1)\rangle_{\Sigma_1}$, i.e. (1) is proved.

Since A_1 and A_2 satisfy the reachability and observability constraints induced by $\Sigma_{1,\mathrm{COL}}$ and $\Sigma_{2,\mathrm{COL}}$ respectively, one can easily derive from (1) and (2), using Lemmas 45 and 47, that A' satisfies the reachability and observability constraints induced by Σ'_{COL}, i.e., $A' \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma'_{\mathrm{COL}})$. Obviously, A' is unique since amalgamated unions of standard algebras are unique. □

Let us now focus on the interpolation property as defined, for instance, in [38].

The interpolation property requires that for any pushout in $\mathrm{Sign}_{\mathrm{COL}}$

$$\begin{tikzcd}
\Sigma_{\mathrm{COL}} \arrow[r, "\sigma_{1,\mathrm{COL}}"] \arrow[d, "\sigma_{2,\mathrm{COL}}"'] & \Sigma_{1,\mathrm{COL}} \arrow[d, "\sigma'_{1,\mathrm{COL}}"] \\
\Sigma_{2,\mathrm{COL}} \arrow[r, "\sigma'_{2,\mathrm{COL}}"] & \Sigma'_{\mathrm{COL}}
\end{tikzcd}$$

and sentences $\varphi_1 \in \mathrm{Sen}_{\mathrm{COL}}(\Sigma_{1,\mathrm{COL}})$ and $\varphi_2 \in \mathrm{Sen}_{\mathrm{COL}}(\Sigma_{2,\mathrm{COL}})$ such that $\sigma'_{1,\mathrm{COL}}(\varphi_1) \models_{\Sigma'_{\mathrm{COL}}} \sigma'_{2,\mathrm{COL}}(\varphi_2)$ there exists a sentence $\varphi \in \mathrm{Sen}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$ such that $\varphi_1 \models_{\Sigma_{1,\mathrm{COL}}} \sigma_{1,\mathrm{COL}}(\varphi)$ and $\sigma_{2,\mathrm{COL}}(\varphi) \models_{\Sigma_{2,\mathrm{COL}}} \varphi_2$.

The idea to check interpolation is to reduce the required property to a logic where interpolation holds. Theorems 40 and 41 give the hint that logical consequences in COL can be reduced to standard consequences in first-order logic if axiomatizations of reachability and full abstractness are provided. Since the given constructors and observers, in general, lead to infinitely many constructor terms and observable contexts such axiomatizations can only be defined if we switch to infinitary first-order logic (where sentences may contain countably infinite conjunctions and disjunctions). Then the following formulas $\mathrm{REACH}(\Sigma_{\mathrm{COL}})$ and $\mathrm{FA}(\Sigma_{\mathrm{COL}})$ provide the required axiomatizations for reachability and full abstractness.

- The infinitary sentence $\mathrm{REACH}(\Sigma_{\mathrm{COL}})$ is defined by:
  $\mathrm{REACH}(\Sigma_{\mathrm{COL}}) \overset{\mathrm{def}}{=} \bigwedge_{s \in S_{\mathrm{Cons}}} \mathrm{REACH}(\Sigma_{\mathrm{COL}})_s$
  where for each constrained sort $s \in S_{\mathrm{Cons}}$, $\mathrm{REACH}(\Sigma_{\mathrm{COL}})_s$ is defined by:
  $\mathrm{REACH}(\Sigma_{\mathrm{COL}})_s \overset{\mathrm{def}}{=} \forall x{:}s. \bigvee_{t \in \mathcal{T}(\Sigma_{\mathrm{COL}})_s} \exists \mathrm{Var}(t).\ x = t\,.$ [12]
- The infinitary sentence $\mathrm{FA}(\Sigma_{\mathrm{COL}})$ is defined by:
  $\mathrm{FA}(\Sigma_{\mathrm{COL}}) \overset{\mathrm{def}}{=} \bigwedge_{s \in S_{\mathrm{State}}} \mathrm{FA}(\Sigma_{\mathrm{COL}})_s$
  where for each state sort $s \in S_{\mathrm{State}}$, $\mathrm{FA}(\Sigma_{\mathrm{COL}})_s$ is defined by:
  $$\mathrm{FA}(\Sigma_{\mathrm{COL}})_s \overset{\mathrm{def}}{=} \forall x, y{:}s. \left(\bigwedge_{s' \in S_{\mathrm{Obs}}, c \in \mathcal{C}(\Sigma_{\mathrm{COL}})_{s \to s'}} \forall \mathrm{Var}(c).\ c[x] = c[y] \right) \Rightarrow x = y\,.$$ [13]

Fact 56. *Let Σ_{COL} be a COL-signature with underlying signature Σ and let A be a Σ-algebra. A is reachable and fully abstract w.r.t. Σ_{COL} if and only if $A \models \mathrm{REACH}(\Sigma_{\mathrm{COL}}) \wedge \mathrm{FA}(\Sigma_{\mathrm{COL}})$.*

[12] $\exists \mathrm{Var}(t)$ is an abbreviation for $\exists x_1{:}s_1.\ldots\exists x_n{:}s_n$ where $x_1, \ldots, x_n$ are the variables (of sort $s_1, \ldots, s_n$) of the constructor term t.
[13] $\forall \mathrm{Var}(c)$ is an abbreviation for $\forall x_1{:}s_1.\ldots\forall x_n{:}s_n$ where $x_1, \ldots, x_n$ are the variables (of sort $s_1, \ldots, s_n$) of the context c, apart from its context variable z_s.

In the remainder of this section we assume that Σ-sentences are finitary or infinitary Σ-sentences of the institution IFOLEq (see Section 2.2) and that we consider the infinitary variant of the COL institution where sentences include infinitary sentences and where the COL-satisfaction relation is extended in the straightforward way to infinitary sentences. All theorems related to COL-satisfaction (in particular Theorems 34, 40, and 41) carry over to the infinitary case. Then logical consequence w.r.t. COL-satisfaction can be characterized by logical consequence in IFOLEq in the following way.[14]

Lemma 57. *Let Σ_{COL} be a* COL*-signature with underlying signature Σ, let ψ and φ be Σ-sentences. Then:*

$$\psi \models_{\Sigma_{\mathrm{COL}}} \varphi \textit{ if and only if } \psi \wedge \mathrm{REACH}(\Sigma_{\mathrm{COL}}) \wedge \mathrm{FA}(\Sigma_{\mathrm{COL}}) \models \varphi$$

Proof. $\psi \models_{\Sigma_{\mathrm{COL}}} \varphi$ iff (by definition) $\langle \Sigma_{\mathrm{COL}}, \{\psi\}\rangle \models_{\Sigma_{\mathrm{COL}}} \varphi$ iff (by Theorem 40) $[\![\langle \Sigma_{\mathrm{COL}}, \{\psi\}\rangle]\!] \models \varphi$ iff (by Theorem 41) $\{\Sigma-$algebra $A \mid A \models \psi$ and A is both reachable and fully abstract w.r.t. $\Sigma_{\mathrm{COL}}\} \models \varphi$ iff (by Fact 56)
$\langle \Sigma, \{\psi \wedge \mathrm{REACH}(\Sigma_{\mathrm{COL}}) \wedge \mathrm{FA}(\Sigma_{\mathrm{COL}})\}\rangle \models \varphi$ iff (by definition)
$\psi \wedge \mathrm{REACH}(\Sigma_{\mathrm{COL}}) \wedge \mathrm{FA}(\Sigma_{\mathrm{COL}}) \models \varphi$. □

Lemma 57 together with the fact that interpolation holds in the infinitary logic $\mathcal{L}_{\omega_1,\omega}$ (see [24]) and hence, if we restrict to injective signature morphisms, also in IFOLEq, are the key for getting the interpolation property for infinitary COL with injective signature morphisms.

Theorem 58 (COL-Interpolation). *Let be given a pushout diagram as depicted above such that the signature morphisms σ_1 and σ_2 underlying $\sigma_{1,\mathrm{COL}}$ and $\sigma_{2,\mathrm{COL}}$ are injective. Then the interpolation property holds in infinitary* COL*.*

Proof. We have to show that for any Σ_1-sentence φ_1 and Σ_2-sentence φ_2 with $\sigma'_1(\varphi_1) \models_{\Sigma'_{\mathrm{COL}}} \sigma'_2(\varphi_2)$ there exists a Σ-sentence φ such that $\varphi_1 \models_{\Sigma_{1,\mathrm{COL}}} \sigma_1(\varphi)$ and $\sigma_2(\varphi) \models_{\Sigma_{2,\mathrm{COL}}} \varphi_2$.
In the following of this proof let $\mathrm{RFA}(\Sigma'_{\mathrm{COL}})$ be a shorthand notation for $\mathrm{REACH}(\Sigma'_{\mathrm{COL}}) \wedge \mathrm{FA}(\Sigma'_{\mathrm{COL}})$ and similarly for $\Sigma_{1,\mathrm{COL}}$ and $\Sigma_{2,\mathrm{COL}}$.
Now, let φ_1 be a Σ_1-sentence and φ_2 be a Σ_2-sentence with $\sigma'_1(\varphi_1) \models_{\Sigma'_{\mathrm{COL}}} \sigma'_2(\varphi_2)$. Then, by Lemma 57, $\sigma'_1(\varphi_1) \wedge \mathrm{RFA}(\Sigma'_{\mathrm{COL}}) \models \sigma'_2(\varphi_2)$. Since $\sigma'_{1,\mathrm{COL}}$ and $\sigma'_{2,\mathrm{COL}}$ are COL-signature morphisms and since Σ'_{COL} is the pushout signature,

$$\mathrm{RFA}(\Sigma'_{\mathrm{COL}}) = \sigma'_1(\mathrm{RFA}(\Sigma_{1,\mathrm{COL}})) \wedge \sigma'_2(\mathrm{RFA}(\Sigma_{2,\mathrm{COL}})).$$ [15]

[14] See Section 2.3 for the definition of logical consequence.
[15] Strictly speaking, both sides are just equivalent formulas; the equation holds literally only if conjunctions are interpreted as sets of conjuncts.

Hence $\sigma_1'(\varphi_1) \wedge \sigma_1'(\mathrm{RFA}(\Sigma_{1,\mathrm{COL}})) \wedge \sigma_2'(\mathrm{RFA}(\Sigma_{2,\mathrm{COL}})) \models \sigma_2'(\varphi_2)$ and therefore, by a simple syntactic and logical transformation,

$$\sigma_1'(\varphi_1 \wedge \mathrm{RFA}(\Sigma_{1,\mathrm{COL}})) \models \sigma_2'(\mathrm{RFA}(\Sigma_{2,\mathrm{COL}}) \Rightarrow \varphi_2).$$

Since the given signature morphisms are injective we can now apply the interpolation theorem for infinitary first-order logic (see [24]) and obtain that there exists a Σ-sentence φ such that

$$\varphi_1 \wedge \mathrm{RFA}(\Sigma_{1,\mathrm{COL}}) \models \sigma_1(\varphi) \text{ and } \sigma_2(\varphi) \models (\mathrm{RFA}(\Sigma_{2,\mathrm{COL}}) \Rightarrow \varphi_2).$$

Since the latter formula is equivalent to $\sigma_2(\varphi) \wedge \mathrm{RFA}(\Sigma_{2,\mathrm{COL}}) \models \varphi_2$ we then obtain, by Lemma 57, $\varphi_1 \models_{\Sigma_{1,\mathrm{COL}}} \sigma_1(\varphi)$ and $\sigma_2(\varphi) \models_{\Sigma_{2,\mathrm{COL}}} \varphi_2$. $\square$

PART II — Proving Consequences of Structured COL-Specifications

In the first part of this paper, we have defined the constructor-based observational logic COL, which as an institution provides a suitable framework for defining structured COL-specifications. We are now interested in the proof of consequences of structured COL-specifications, and in particular in efficient and practicable proof techniques that would easily be implemented in available theorem provers for ordinary specifications.

Note that we unfortunately cannot reuse our previous work on proof systems for structured specifications with observability operators (see [3]). From a technical point of view, since the COL institution lacks the amalgamation property in general, there is no obvious way to compute the *normal form* of a structured COL-specification. For the same reason, even if we were able to define a sound and complete proof system for the institution COL, we would not know how to lift it to a compositional sound and complete proof system for structured COL-specifications (see [12]). Moreover, our previous work was relying either on infinitary sentences or on infinitary proof rules, which are not so appropriate from a practical point of view.

We will therefore follow a different approach, based on institution encodings à la Tarlecki [38]. Thereby a crucial step is an adequate syntactic encoding of observable contexts as term-generated values of auxiliary "context-sorts". This syntactic encoding idea (of observable contexts into generated values of context-sorts) was already described in [4, Section 7.3], but there it was claimed to be of purely theoretical interest. The reasons why the same idea now becomes fruitful are twofold: First, in our COL setting, we distinguish a subset of operations as observers, which leads to a smaller set of observable contexts; then, since [7], we use a coinductive definition of the observable contexts, which leads to an adequate syntactic encoding, in contrast to the more usual inductive definition of observable contexts.

We start by an intuitive illustration of our syntactic encoding technique.

Example 59. To illustrate the constructions and proof techniques developed in the second part of this paper, we will use the following running example of a specification of infinite streams.[16] This specification contains, in particular, a coinductive definition of the *zip* function for an alternating merge of two streams (see also e.g. [17]).

spec Stream =
  sorts *elem*, *stream*
  ops $head : stream \rightarrow elem$;
    $tail : stream \rightarrow stream$;

[16] We use here again a syntactic sugar similar to the one of Casl [1].

$odd : stream \rightarrow stream;$
$even : stream \rightarrow stream;$
$\_\_.\_\_ : elem \times stream \rightarrow stream;$
$zip : stream \times stream \rightarrow stream;$

observers *head*, *tail*

axioms

$\forall e : elem;\ S, S' : stream$

- $head(e.S) = e$
- $tail(e.S) = S$
- $head(odd(S)) = head(S)$
- $tail(odd(S)) = odd(tail(tail(S)))$
- $even(S) = odd(tail(S))$
- $head(zip(S, S')) = head(S)$
- $tail(zip(S, S')) = zip(S', tail(S))$

end

For instance, we would like to prove that:[17]
$\textsc{Stream} \models_{\mathrm{COL}} zip(odd(S), even(S)) = S$.

The observable contexts are of the form $head(tail^n(z_{stream}))$. According to Definition 10, these observable contexts are coinductively defined as follows: $head(z_{stream})$ is an observable context, and if $head(c)$ is an observable context (with context variable z_{stream}), then $head(c[tail(z_{stream})/z_{stream}])$ is also an observable context.

The main idea underlying our encoding technique is to introduce a new sort $Cont[stream{\rightarrow}elem]$ whose values are expected to reflect observable contexts; this new sort will have two constructors $head^* : \ \rightarrow Cont[stream{\rightarrow}elem]$ and $tail^* : Cont[stream{\rightarrow}elem] \rightarrow Cont[stream{\rightarrow}elem]$ which correspond to the coinductive definition of observable contexts. Then we introduce also an operation $apply : Cont[stream{\rightarrow}elem], stream \rightarrow elem$ inductively defined by the axioms $apply(head^*, S) = head(S)$ and $apply(tail^*(c), S) = apply(c, tail(S))$. Intuitively, two stream values S and S' are observationally equal if and only if, for all observable context values c in $Cont[stream{\rightarrow}elem]$, $apply(c, S) = apply(c, S')$. Thus, to prove that an equation $L = R$ is an observational theorem of Stream, it is equivalent to prove that $apply(c, L) = apply(c, R)$ is an inductive theorem of Stream enriched by the above declarations and axioms. Now, this can be proved by an induction w.r.t. the constructors $head^*$ and $tail^*$ of the sort $Cont[stream{\rightarrow}elem]$, which leads to the basic case:
$apply(head^*, L) = apply(head^*, R)$, i.e., $head(L) = head(R)$;
and to the induction step:

[17] In our examples, for the sake of simplicity, we just use $\models_{\mathrm{COL}}$ to indicate that we use the satisfaction relation of the COL institution, omitting the COL-signature; moreover, the variables occurring in the formulas are implicitly universally quantified.

$apply(tail^*(c0), L) = apply(tail^*(c0), R)$, i.e.,
$apply(c0, tail(L)) = apply(c0, tail(R))$,
with $apply(c0, L) = apply(c0, R)$ as induction hypothesis, and where $c0$ denotes a fresh constant of sort $Cont[stream \to elem]$. To proceed further, one will then apply axioms of STREAM to rewrite the terms $head(L)$, $head(R)$, $tail(L)$, $tail(R)$, as usual for proofs of consequences of specifications.

Thereby it is essential to understand the benefit of our coinductive definition of observable contexts. Assume for a moment that we would have defined observable contexts in the usual inductive style. Following the same ideas as above, this would lead to the definition of two new sorts: $Cont[stream \to elem]$, plus an auxiliary new sort $Cont[stream \to stream]$, with constructors $head^\dagger : Cont[stream \to stream] \to Cont[stream \to elem]$, $Z : \to Cont[stream \to stream]$, and $tail^\dagger : Cont[stream \to stream] \to Cont[stream \to stream]$. Now we need an operation $apply : Cont[stream \to elem], stream \to elem$ as above plus an auxiliary operation $apply : Cont[stream \to stream], stream \to stream$ which are inductively defined by the axioms: $apply(head^\dagger(c), S) = head(apply(c, S))$, $apply(Z, S) = S$, and $apply(tail^\dagger(c), S) = tail(apply(c, S))$. This means that to prove an equation $L = R$, a proof by induction leads to the following basic case:
$apply(head^\dagger(Z), L) = apply(head^\dagger(Z), R)$, which reduces to $head(L) = head(R)$; and to the induction step:
$apply(head^\dagger(tail^\dagger(c0)), L) = apply(head^\dagger(tail^\dagger(c0)), R)$, i.e.,
$head(tail(apply(c0, L))) = head(tail(apply(c0, R)))$,
with $apply(head^\dagger(c0), L) = apply(head^\dagger(c0), R)$, i.e., $head(apply(c0, L)) = head(apply(c0, R))$ as induction hypothesis. How to proceed to conclude the induction step is therefore problematic in general. This is the reason why context induction [20] is not an appropriate proof method in a framework with distinguished observer operations and coinductive axiomatizations. $\diamond$

The aim of the second part of this paper is to formalize the above ideas in a general setting, with arbitrary structured COL-specifications on the one hand, and arbitrary first-order sentences on the other.

5 Institution Encodings

Institution encodings are the main technical tool that will be used in the second part of this paper, and we detail in this section the main definitions and results needed for our purposes. Our presentation is strongly inspired by the original work presented by Andrzej Tarlecki in [38].

Throughout this section we assume given two arbitrary institutions $\mathrm{I} = (\mathrm{Sign}, \mathrm{Sen}, \mathrm{Mod}, \models)$ and $\mathrm{I}' = (\mathrm{Sign}', \mathrm{Sen}', \mathrm{Mod}', \models')$. Moreover, we assume that satisfaction in both I and I′ is closed under isomorphisms, hence the model class of any structured specification over I (I′, resp.) is closed under isomorphisms.

In contrast with institution morphisms and representations, in an institution encoding both models and sentences are translated covariantly with respect to each other. (For a survey on different notions of morphisms between institutions, see [18], where institution encodings are called forward institution morphisms).

Definition 60 (Institution encoding). *An* institution encoding $\varepsilon : \mathrm{I} \rightarrow \mathrm{I}'$ *between two institutions* I *and* I′ *consists of:*

- *a functor* $\varepsilon^{\mathrm{Sig}} : \mathrm{Sign} \rightarrow \mathrm{Sign}'$*,*
- *a natural transformation* $\varepsilon^{\mathrm{Sen}} : \mathrm{Sen} \rightarrow \varepsilon^{\mathrm{Sig}};\ \mathrm{Sen}'$*, and*
- *a natural transformation* $\varepsilon^{\mathrm{Mod}} : \mathrm{Mod} \rightarrow (\varepsilon^{\mathrm{Sig}})^{\mathrm{op}};\ \mathrm{Mod}'$

such that for any $\Sigma \in \mathrm{Sign}$*, the translations* $\varepsilon^{\mathrm{Sen}}_{\Sigma} : \mathrm{Sen}(\Sigma) \rightarrow \mathrm{Sen}'(\varepsilon^{\mathrm{Sig}}(\Sigma))$ *and* $\varepsilon^{\mathrm{Mod}}_{\Sigma} : \mathrm{Mod}(\Sigma) \rightarrow \mathrm{Mod}'(\varepsilon^{\mathrm{Sig}}(\Sigma))$ *preserve the satisfaction relation, that is, for any* $\varphi \in \mathrm{Sen}(\Sigma)$ *and* $M \in \mathrm{Mod}(\Sigma)$ *the following* encoding condition *holds:*

$$M \models_{\Sigma} \varphi \textit{ if and only if } \varepsilon^{\mathrm{Mod}}_{\Sigma}(M) \models'_{\varepsilon^{\mathrm{Sig}}(\Sigma)} \varepsilon^{\mathrm{Sen}}_{\Sigma}(\varphi)\,.$$

Definition 61 (Iso-reflecting encoding). *An institution encoding* $\varepsilon : \mathrm{I} \rightarrow \mathrm{I}'$ *is* iso-reflecting *if, for any signature* Σ *and models* $M, M' \in \mathrm{Mod}(\Sigma)$*,* $\varepsilon^{\mathrm{Mod}}_{\Sigma}(M)$ *isomorphic to* $\varepsilon^{\mathrm{Mod}}_{\Sigma}(M')$ *(in* I′*) implies* M *isomorphic to* M' *(in* I*).*

Of particular interest are logical encodings.

Definition 62 (Logical institution encoding). *An institution encoding* $\varepsilon : \mathrm{I} \rightarrow \mathrm{I}'$ *is a* logical institution encoding characterized by a family $\langle \Gamma'_{\Sigma} \subseteq \mathrm{Sen}'(\varepsilon^{\mathrm{Sig}}(\Sigma))\rangle_{\Sigma \in \mathrm{Sign}}$ *if for each* $\Sigma \in \mathrm{Sign}$*,* Γ'_{Σ} *characterizes the image of the* Σ*-model encoding, that is:*

$$\mathrm{Iso}_{\varepsilon^{\mathrm{Sig}}(\Sigma)}(\varepsilon^{\mathrm{Mod}}_{\Sigma}(\mathrm{Mod}(\Sigma))) = \mathcal{M}od[\langle \varepsilon^{\mathrm{Sig}}(\Sigma), \Gamma'_{\Sigma}\rangle]$$

where $\mathcal{M}od[\langle \varepsilon^{\mathrm{Sig}}(\Sigma), \Gamma'_{\Sigma}\rangle]$ *is the model class of the presentation* $\langle \varepsilon^{\mathrm{Sig}}(\Sigma), \Gamma'_{\Sigma}\rangle$ *defined over the institution* I′ *(see Section 2.3).*

This definition leads to a rather obvious syntactic translation of structured specifications (in the sense of Section 2.3) under a logical institution encoding.

Definition 63 (Structured specification encoding). *Let* $\varepsilon : \mathrm{I} \rightarrow \mathrm{I}'$ *be a logical institution encoding characterized by a family* $\langle \Gamma'_{\Sigma}\rangle_{\Sigma \in \mathrm{Sign}}$*. Given a structured specification* SP *over the institution* I*, its* encoding under ε*, written* $\hat{\varepsilon}(\mathrm{SP})$*, is defined as follows:*

- $\hat{\varepsilon}(\langle \Sigma, \mathrm{Ax}\rangle) \overset{\mathrm{def}}{=} \langle \varepsilon^{\mathrm{Sig}}(\Sigma), \varepsilon^{\mathrm{Sen}}_{\Sigma}(\mathrm{Ax}) \cup \Gamma'_{\Sigma}\rangle$*,*
- $\hat{\varepsilon}(\mathrm{SP}_1 \cup \mathrm{SP}_2) \overset{\mathrm{def}}{=} \hat{\varepsilon}(\mathrm{SP}_1) \cup \hat{\varepsilon}(\mathrm{SP}_2)$*,*

- *let* $\sigma : \mathcal{S}ig[\mathrm{SP}] \to \Sigma$ *be a signature morphism in* I*;*
  $\hat{\varepsilon}(\textbf{translate}\ \mathrm{SP}\ \textbf{by}\ \sigma) \stackrel{\text{def}}{=} (\textbf{translate}\ \hat{\varepsilon}(\mathrm{SP})\ \textbf{by}\ \varepsilon^{\mathrm{Sig}}(\sigma)) \cup \langle \varepsilon^{\mathrm{Sig}}(\Sigma), \Gamma'_{\Sigma}\rangle$*,*
- *let* $\sigma : \Sigma \to \mathcal{S}ig[\mathrm{SP}]$ *be a signature morphism in* I*;*
  $\hat{\varepsilon}(\textbf{derive from}\ \mathrm{SP}\ \textbf{by}\ \sigma) \stackrel{\text{def}}{=} \textbf{derive from}\ \hat{\varepsilon}(\mathrm{SP})\ \textbf{by}\ \varepsilon^{\mathrm{Sig}}(\sigma)$*.*

This translation is well-defined and under suitable assumptions, the model class of the encoding of a structured specification SP is (the closure under isomorphisms of) the encoding of the model class of SP, as stated in the following theorem.

Theorem 64. *Let* I *and* I' *be two institutions such that satisfaction is closed under isomorphisms both in* I *and in* I'*, and let* $\varepsilon : \mathrm{I} \to \mathrm{I}'$ *be an iso-reflecting logical institution encoding. Then for any structured specification* SP *over* I *with signature* $\mathcal{S}ig[\mathrm{SP}] = \Sigma$*, we have* $\mathcal{M}od[\hat{\varepsilon}(\mathrm{SP})] = \mathrm{Iso}_{\varepsilon^{\mathrm{Sig}}(\Sigma)}(\varepsilon^{\mathrm{Mod}}_{\Sigma}(\mathcal{M}od[\mathrm{SP}]))$*.*

Proof. The proof is done by induction on the structure of SP. The fact that $\varepsilon^{\mathrm{Mod}}_{\Sigma}(\mathcal{M}od[\mathrm{SP}]) \subseteq \mathcal{M}od[\hat{\varepsilon}(\mathrm{SP})]$ follows directly from Definitions 60 and 63, and from the assumption that ε is a logical institution encoding, so we do not detail its proof here. Then the assumption that satisfaction is closed under isomorphisms in I' entails that $\mathcal{M}od[\hat{\varepsilon}(\mathrm{SP})]$ is closed under isomorphisms, and it is therefore obvious that $\mathrm{Iso}_{\varepsilon^{\mathrm{Sig}}(\Sigma)}(\varepsilon^{\mathrm{Mod}}_{\Sigma}(\mathcal{M}od[\mathrm{SP}])) \subseteq \mathcal{M}od[\hat{\varepsilon}(\mathrm{SP})]$. We focus now on the proof that $\mathcal{M}od[\hat{\varepsilon}(\mathrm{SP})] \subseteq \mathrm{Iso}_{\varepsilon^{\mathrm{Sig}}(\Sigma)}(\varepsilon^{\mathrm{Mod}}_{\Sigma}(\mathcal{M}od[\mathrm{SP}]))$, which is the non-trivial part.

Case $\mathrm{SP} = \langle \Sigma, \mathrm{Ax}\rangle$**:** By definition $\hat{\varepsilon}(\mathrm{SP}) = \langle \varepsilon^{\mathrm{Sig}}(\Sigma), \varepsilon^{\mathrm{Sen}}_{\Sigma}(\mathrm{Ax}) \cup \Gamma'_{\Sigma}\rangle$.
Let $B \in \mathcal{M}od[\hat{\varepsilon}(\mathrm{SP})]$. Hence $B \models'_{\varepsilon^{\mathrm{Sig}}(\Sigma)} \Gamma'_{\Sigma}$. From the assumption that ε is logical it follows there exists $B' \in \varepsilon^{\mathrm{Mod}}_{\Sigma}(\mathrm{Mod}(\Sigma))$ isomorphic to B, i.e., there exists $A \in \mathrm{Mod}(\Sigma)$ such that B is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A)$. By hypothesis $B \models'_{\varepsilon^{\mathrm{Sig}}(\Sigma)} \varepsilon^{\mathrm{Sen}}_{\Sigma}(\mathrm{Ax})$, and since satisfaction is closed under isomorphisms in I', $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A) \models'_{\varepsilon^{\mathrm{Sig}}(\Sigma)} \varepsilon^{\mathrm{Sen}}_{\Sigma}(\mathrm{Ax})$. Now from the encoding condition we conclude $A \models_{\Sigma} \mathrm{Ax}$, which implies $A \in \mathcal{M}od[SP]$. Thus B is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A)$, with $A \in \mathcal{M}od[SP]$.

Case $\mathrm{SP} = \mathrm{SP}_1 \cup \mathrm{SP}_2$**:** By definition $\hat{\varepsilon}(\mathrm{SP}) = \hat{\varepsilon}(\mathrm{SP}_1) \cup \hat{\varepsilon}(\mathrm{SP}_2)$.
Let $B \in \mathcal{M}od[\hat{\varepsilon}(\mathrm{SP})]$. Hence $B \in \mathcal{M}od[\hat{\varepsilon}(\mathrm{SP}_1)]$ and $B \in \mathcal{M}od[\hat{\varepsilon}(\mathrm{SP}_2)]$. From the induction hypothesis there exists $A_1 \in \mathcal{M}od[\mathrm{SP}_1]$ such that B is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A_1)$, and similarly there exists $A_2 \in \mathcal{M}od[\mathrm{SP}_2]$ such that B is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A_2)$. By transitivity $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A_1)$ is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A_2)$, and using the fact that ε is iso-reflecting we conclude that A_1 is isomorphic to A_2. Now, since satisfaction is closed under isomorphisms in I, in particular the model class of SP_2 is also closed under isomorphisms, hence $A_1 \in \mathcal{M}od[\mathrm{SP}_2]$, which ensures $A_1 \in \mathcal{M}od[SP]$. Thus B is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A_1)$, with $A_1 \in \mathcal{M}od[SP]$.

Case $\mathrm{SP} = \textbf{translate}\ \mathrm{SP}_1\ \textbf{by}\ \sigma$**,** with $\sigma : \mathcal{S}ig[\mathrm{SP}_1] \to \Sigma$**:**
By definition $\hat{\varepsilon}(\mathrm{SP}) = (\textbf{translate}\ \hat{\varepsilon}(\mathrm{SP}_1)\ \textbf{by}\ \varepsilon^{\mathrm{Sig}}(\sigma)) \cup \langle \varepsilon^{\mathrm{Sig}}(\Sigma), \Gamma'_{\Sigma}\rangle$.

Let $B \in \mathcal{M}od[\hat{\varepsilon}(\mathrm{SP})]$. Hence $B \in \mathcal{M}od[\textbf{translate } \hat{\varepsilon}(\mathrm{SP}_1) \textbf{ by } \varepsilon^{\mathrm{Sig}}(\sigma)]$, i.e., $B|_{\varepsilon^{\mathrm{Sig}}(\sigma)} \in \mathcal{M}od[\hat{\varepsilon}(\mathrm{SP}_1)]$. From the induction hypothesis, there exists $A \in \mathcal{M}od[\mathrm{SP}_1]$ such that:

(a) $B|_{\varepsilon^{\mathrm{Sig}}(\sigma)}$ is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A)$.

We know also $B \models'_{\varepsilon^{\mathrm{Sig}}(\Sigma)} \Gamma'_{\Sigma}$, which entails that there exists $C \in \mathrm{Mod}(\Sigma)$ such that B is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(C)$, which implies that:

(b) $B|_{\varepsilon^{\mathrm{Sig}}(\sigma)}$ is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(C)|_{\varepsilon^{\mathrm{Sig}}(\sigma)} = \varepsilon^{\mathrm{Mod}}_{\Sigma_1}(C|_{\sigma})$,

where $\Sigma_1 = \mathcal{S}ig[\mathrm{SP}_1]$. From (a) and (b) we conclude that, since ε is iso-reflecting, A is isomorphic to $C|_{\sigma}$. Now, since satisfaction is closed under isomorphisms in I, the model class of SP_1 is closed under isomorphisms, and thus $C|_{\sigma} \in \mathcal{M}od[\mathrm{SP}_1]$, which implies that $C \in \mathcal{M}od[\textbf{translate } \mathrm{SP}_1 \textbf{ by } \sigma]$. Thus B is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(C)$, with $C \in \mathcal{M}od[SP]$.

Case SP = **derive from** SP_1 **by** σ**,** with $\sigma : \Sigma \to \mathcal{S}ig[\mathrm{SP}_1]$ **:**

By definition $\hat{\varepsilon}(\mathrm{SP}) = \textbf{derive from } \hat{\varepsilon}(\mathrm{SP}_1) \textbf{ by } \varepsilon^{\mathrm{Sig}}(\sigma)$.

Let $B \in \mathcal{M}od[\hat{\varepsilon}(\mathrm{SP})]$. Hence there exists $B' \in \mathcal{M}od[\hat{\varepsilon}(\mathrm{SP}_1)]$ such that B is isomorphic to $B'|_{\varepsilon^{\mathrm{Sig}}(\sigma)}$. From the induction hypothesis, there exists $A \in \mathcal{M}od[\mathrm{SP}_1]$ such that:

(a) $\varepsilon^{\mathrm{Mod}}_{\Sigma_1}(A)$ is isomorphic to B',

where $\Sigma_1 = \mathcal{S}ig[\mathrm{SP}_1]$. Note that $A|_{\sigma} \in \mathcal{M}od[\textbf{derive from } \mathrm{SP}_1 \textbf{ by } \sigma]$, and that:

(b) $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A|_{\sigma}) = \varepsilon^{\mathrm{Mod}}_{\Sigma_1}(A)|_{\varepsilon^{\mathrm{Sig}}(\sigma)}$.

From (a) we derive that $\varepsilon^{\mathrm{Mod}}_{\Sigma_1}(A)|_{\varepsilon^{\mathrm{Sig}}(\sigma)}$ is isomorphic to $B'|_{\varepsilon^{\mathrm{Sig}}(\sigma)}$, which itself is isomorphic to B. Thus, according to (b), B is isomorphic to $\varepsilon^{\mathrm{Mod}}_{\Sigma}(A|_{\sigma})$, with $A|_{\sigma} \in \mathcal{M}od[SP]$.

This concludes the proof that $\mathcal{M}od[\hat{\varepsilon}(\mathrm{SP})] \subseteq \mathrm{Iso}_{\varepsilon^{\mathrm{Sig}}(\Sigma)}(\varepsilon^{\mathrm{Mod}}_{\Sigma}(\mathcal{M}od[\mathrm{SP}]))$. □

Corollary 65. *Under the hypotheses of Theorem 64, for any structured specification* SP *over* I *with signature* $\mathcal{S}ig[\mathrm{SP}] = \Sigma$ *and for any* Σ*-sentence* $\varphi \in \mathrm{Sen}(\Sigma)$*:*

$$\mathrm{SP} \models_{\Sigma} \varphi \textit{ if and only if } \hat{\varepsilon}(\mathrm{SP}) \models'_{\varepsilon^{\mathrm{Sig}}(\Sigma)} \varepsilon^{\mathrm{Sen}}_{\Sigma}(\varphi).$$

An important consequence is that under the hypotheses of the above theorem, the following proof rule is correct:

$$\frac{\hat{\varepsilon}(\mathrm{SP}) \vdash^{\mathrm{I}'} \varepsilon^{\mathrm{Sen}}_{\Sigma}(\varphi)}{\mathrm{SP} \vdash^{\mathrm{I}} \varphi}$$

which means that, given a sound (resp. complete) proof system for consequences of structured specifications over I′, we obtain a sound (resp. complete) proof system for consequences of structured specifications over I.

The aim of the second part of this paper is therefore to define an iso-reflecting logical institution encoding from the COL institution to the CFOLEq institution of first-order logic with equality and sort-generation constraints (see

Section 2.2). Then standard proof systems for first-order logic together with induction can be applied to prove COL-theorems.

For technical reasons, we will proceed in two steps. In the first one we will encode the COL institution into an intermediate institution IBB, and in the second one we will encode IBB into CFOLEq. The motivations for proceeding in this way are that this splitting leads to much easier and clearer proofs of the fact that each of the two encodings is both logical and iso-reflecting. Then we obtain the desired result by applying twice Corollary 65.

6 Encoding COL into IBB

As explained above, the intermediate institution IBB is introduced for technical reasons only.

Definition 66 (The IBB institution). *The institution* IBB *is defined as follows:*

- *The category of signatures* $\mathrm{Sign}_{\mathrm{IBB}}$ *is exactly the category of signatures* $\mathrm{Sign}_{\mathrm{COL}}$ *of the* COL *institution.*
- *The functor* $\mathrm{Sen}_{\mathrm{IBB}}$ *is exactly the functor* $\mathrm{Sen}_{\mathrm{COL}}$ *of the* COL *institution. (So, sentences in* IBB *are usual (finitary) first-order sentences.)*
- *For any* COL*-signature* $\Sigma_{\mathrm{COL}} = (\Sigma, \mathrm{OP}_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Obs}})$*, let* $\mathrm{Alg}_{\mathrm{rfa}(\Sigma_{\mathrm{COL}})}(\Sigma)$ *denote the full subcategory of* $\mathrm{Alg}(\Sigma)$ *of all* Σ*-algebras which are both reachable and fully abstract w.r.t.* Σ_{COL}*. The functor* $\mathrm{Mod}_{\mathrm{IBB}} : \mathrm{Sign}_{\mathrm{IBB}}^{\mathrm{op}} \to \mathbf{Cat}$ *maps:*
  ⋆ *each* COL*-signature* Σ_{COL} *to* $\mathrm{Alg}_{\mathrm{rfa}(\Sigma_{\mathrm{COL}})}(\Sigma)$*;*
  ⋆ *each* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \Sigma'_{\mathrm{COL}}$ *to the (standard) reduct functor associated to the underlying (standard) signature morphism* σ*, restricted to reachable and fully abstract algebras.*

$$\begin{tikzcd}
\mathrm{Mod}_{\mathrm{IBB}} : & \Sigma'_{\mathrm{COL}} \arrow[r, mapsto] & \mathrm{Alg}_{\mathrm{rfa}(\Sigma'_{\mathrm{COL}})}(\Sigma') \arrow[d, "{--|_\sigma}"] \\
& \Sigma_{\mathrm{COL}} \arrow[u, "\sigma_{\mathrm{COL}}"] \arrow[r, mapsto] & \mathrm{Alg}_{\mathrm{rfa}(\Sigma_{\mathrm{COL}})}(\Sigma)
\end{tikzcd}$$

- *In* IBB*, the satisfaction relation is inherited from the usual satisfaction relation between* Σ*-algebras and* Σ*-sentences of* FOLEq.

The fact that the above defines an institution is a direct consequence of Corollary 48, which ensures that $\mathrm{Mod}_{\mathrm{IBB}}$ is indeed a well-defined functor. Moreover, obviously satisfaction is closed under isomorphisms in IBB.

The encoding $\varepsilon 1$ from COL to IBB is now defined as follows.

Definition 67 (Institution encoding $\varepsilon 1$ from COL to IBB). *The institution encoding* $\varepsilon 1 : \mathrm{COL} \to \mathrm{IBB}$ *between the institutions* COL *and* IBB *consists of:*

- *the identity functor* $\varepsilon 1^{\mathrm{Sig}} : \mathrm{Sign}_{\mathrm{COL}} \to \mathrm{Sign}_{\mathrm{IBB}}$ *,*
- *the identity natural transformation* $\varepsilon 1^{\mathrm{Sen}} : \mathrm{Sen}_{\mathrm{COL}} \to \varepsilon 1^{\mathrm{Sig}}; \mathrm{Sen}_{\mathrm{IBB}}$ *,*
- *the natural transformation* $\varepsilon 1^{\mathrm{Mod}} : \mathrm{Mod}_{\mathrm{COL}} \to (\varepsilon 1^{\mathrm{Sig}})^{\mathrm{op}}; \mathrm{Mod}_{\mathrm{IBB}}$ *induced by the black-box functors* $\mathcal{BB}_{\Sigma_{\mathrm{COL}}}$ *(see Definition 29).*

The fact that $\varepsilon 1^{\mathrm{Mod}}$ is a natural transformation results from Theorem 51 which ensures that the following diagram is commutative: [18]

$$\begin{tikzcd}
\Sigma_{2,\mathrm{COL}} & \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{2,\mathrm{COL}}) \arrow[r, "\mathcal{BB}_{\Sigma_{2,\mathrm{COL}}}"] \arrow[d, "-{-}|_{\sigma_{\mathrm{COL}}}"'] \arrow[dr, phantom, "="] & \mathrm{Alg}_{\mathrm{rfa}(\Sigma_{2,\mathrm{COL}})}(\Sigma_2) \arrow[d, "-{-}|_{\sigma}"] \\
\Sigma_{1,\mathrm{COL}} \arrow[u, "\sigma_{\mathrm{COL}}"] & \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{1,\mathrm{COL}}) \arrow[r, "\mathcal{BB}_{\Sigma_{1,\mathrm{COL}}}"] & \mathrm{Alg}_{\mathrm{rfa}(\Sigma_{1,\mathrm{COL}})}(\Sigma_1)
\end{tikzcd}$$

The fact that the encoding condition holds results from Theorem 34, since for $\varepsilon 1$ the encoding condition reads as follows. For any COL-signature Σ_{COL}, sentence $\varphi \in \mathrm{Sen}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}}) = \mathrm{Sen}_{\mathrm{IBB}}(\Sigma_{\mathrm{COL}})$, and model $A \in \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$:

$$\underbrace{A \models_{\Sigma_{\mathrm{COL}}} \varphi}_{\text{in COL}} \text{ if and only if } \underbrace{\mathcal{BB}_{\Sigma_{\mathrm{COL}}}(A) \models_{\Sigma} \varphi}_{\text{in FOLEq}} .$$

Since the black box functors $\mathcal{BB}_{\Sigma_{\mathrm{COL}}}$ are full and faithful, the institution encoding $\varepsilon 1$ is iso-reflecting. Moreover, $\varepsilon 1$ is a logical encoding in a trivial way.

Lemma 68. *The institution encoding* $\varepsilon 1 : \mathrm{COL} \to \mathrm{IBB}$ *is a logical institution encoding, trivially characterized by a family of empty sets of sentences, i.e. we have, for each* COL*-signature* Σ_{COL}*:*

$$\mathrm{Iso}_{\Sigma}(\varepsilon 1^{\mathrm{Mod}}_{\Sigma_{\mathrm{COL}}}(\mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}}))) = \underbrace{\mathcal{M}od[\langle \Sigma_{\mathrm{COL}}, \emptyset \rangle]}_{\text{in IBB}} .$$

Proof. The above is a special case of Theorem 41, with $\mathrm{Ax} = \emptyset$. □

Since signatures and sentences are the same in COL and IBB, and due to the fact that $\varepsilon 1 : \mathrm{COL} \to \mathrm{IBB}$ is a logical institution encoding in a trivial way,

[18] Remember that for any COL-signature Σ_{COL}, $\mathrm{Mod}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}}) \overset{\mathrm{def}}{=} \mathrm{Alg}_{\mathrm{COL}}(\Sigma_{\mathrm{COL}})$.

$\varepsilon 1$ induces a trivial encoding $\widehat{\varepsilon 1}$ for structured specifications. Thus, according to Definition 63, for any structured specification $\mathrm{SP}_{\mathrm{COL}}$ over the institution COL, we have $\widehat{\varepsilon 1}(\mathrm{SP}_{\mathrm{COL}}) = \mathrm{SP}_{\mathrm{COL}}$, now viewed as a structured specification over the institution IBB. Thereby the model class of the specification $\mathrm{SP}_{\mathrm{COL}}$ interpreted as a specification over COL corresponds to the *glass box semantics* of $\mathrm{SP}_{\mathrm{COL}}$, while the model class of $\mathrm{SP}_{\mathrm{COL}}$ interpreted as a specification over IBB corresponds to its *black box semantics.*

In summary, the iso-reflecting logical encoding $\varepsilon 1$ satisfies the hypotheses of Theorem 64, and hence Corollary 65 provides the following result.

Theorem 69. *For any structured specification* $\mathrm{SP}_{\mathrm{COL}}$ *over the* COL *institution with signature* Σ_{COL}*, and any* Σ*-sentence* φ*, we have:*

$$\underbrace{\mathrm{SP}_{\mathrm{COL}} \models_{\Sigma_{\mathrm{COL}}} \varphi}_{\text{in COL}} \text{ if and only if } \underbrace{\mathrm{SP}_{\mathrm{COL}} \models_{\Sigma} \varphi}_{\text{in IBB}} .$$

Note that the above theorem generalizes Theorem 40 to structured specifications.

Remark 70. It is easy to see that the institution IBB has composable signatures, i.e., amalgamations. Therefore, it is easy to define the *normal form* nf(SP) of a structured specification SP over IBB. If we would have a sound and complete proof system Π_{IBB} for the institution IBB, this would then lead to a sound and complete proof system for structured specifications over IBB (see [3]), and therefore also for structured specifications over COL. This is unfortunately not the case. A possible way out is then to compose the institution encoding $\varepsilon 1$ with another institution encoding $\varepsilon : \mathrm{IBB} \to \mathrm{IFOLEq}$, where IFOLEq is the institution of infinitary first-order logic with equality, and to show that ε is an iso-reflecting logical institution encoding. This solution would work from a technical point of view and, as expected (see Section 4.4), the family of sentences to characterize the logical institution encoding ε is $\langle\{\mathrm{REACH}(\Sigma_{\mathrm{COL}}),\ \mathrm{FA}(\Sigma_{\mathrm{COL}})\}\rangle_{\Sigma_{\mathrm{COL}} \in \mathrm{Sign}_{\mathrm{COL}}}$. However, from a practical point of view this solution is not so relevant, since the resulting institution encoding from COL to IFOLEq involves infinitary sentences. This is why we will in the sequel follow a different approach and choose the institution CFOLEq, which provides a standard logical framework, as target institution.

7 Encoding IBB into CFOLEq

The institution CFOLEq is an extension of the FOLEq institution, where in addition to the usual (finitary) first-order sentences, we consider also as extra sentences *sort-generation constraints* of the form $\mathrm{SGC}(S_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Cons}})$, see Section 2.2. Note that a Σ-algebra A satisfies a sort-generation constraint $\mathrm{SGC}(S_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Cons}})$ if it is reachable w.r.t. $\mathrm{OP}_{\mathrm{Cons}}$.

It is well-known that a *free sort-generation constraint* is just an abbreviation for the corresponding sort-generation constraint plus a finite set of first-order sentences to state that all distinct constructor terms (up to variable renaming) denote distinct values. Therefore, in the following, we will also assume that the CFOLEq institution is equipped with free sort-generation constraints of the form $\mathrm{FSGC}(S_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Cons}})$, with the meaning described above (see [28, pp. 152–153]).

As explained at the beginning of this second part, the encoding from IBB to CFOLEq will rely on a syntactic counterpart of observable contexts. Therefore we need a few preliminary definitions. Remember that given a COL-signature Σ_{COL}, for each state sort s and observable sort s', $\mathcal{C}(\Sigma_{\mathrm{COL}})_{s\to s'}$ denotes the set of the observable Σ_{COL}-contexts with application sort s and result sort s'.

Definition 71 (Signature Σ^+ associated to a COL-signature Σ_{COL}). *Let $\Sigma_{\mathrm{COL}} = (\Sigma, \mathrm{OP}_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Obs}})$ be a COL-signature. The associated signature Σ^+ is defined as follows.*
$\Sigma^+ \overset{\mathrm{def}}{=} \Sigma \cup \Delta(\mathrm{OP}_{\mathrm{Obs}}) \cup \Lambda(\mathrm{OP}_{\mathrm{Obs}})$
where $\Delta(\mathrm{OP}_{\mathrm{Obs}})$ is the signature fragment containing:

- *For each state sort $s \in S_{\mathrm{State}}$ and observable sort $s' \in S_{\mathrm{Obs}}$, if $\mathcal{C}(\Sigma_{\mathrm{COL}})_{s\to s'}$ is not empty,* a new sort *$Cont[s{\to}s']$;*[19]
- *For each direct observer $(obs, i) \in \mathrm{OP}_{\mathrm{Obs}}$ with $obs : s_1, \dots, s_i, \dots, s_n \to s'$,* a new operation *$obs_i^* : s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_n \to Cont[s_i{\to}s']$.*[20]
- *For each indirect observer $(obs, i) \in \mathrm{OP}_{\mathrm{Obs}}$ with $obs : s_1, \dots, s_i, \dots, s_n \to s$, and for all observable sorts $s' \in S_{\mathrm{Obs}}$ such that $\mathcal{C}(\Sigma_{\mathrm{COL}})_{s\to s'}$ is not empty,*[21] new (overloaded) operations *$obs_i^* : Cont[s{\to}s'], s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_n \to Cont[s_i{\to}s']$.*

and $\Lambda(\mathrm{OP}_{\mathrm{Obs}})$ is the signature fragment containing:

- *For each new sort $Cont[s{\to}s']$,* a new operation *$apply : Cont[s{\to}s'], s \to s'$ (for the sake of clarity, the operations apply are overloaded also).*

The new sorts $Cont[s{\to}s']$ are expected to reflect the observable contexts in $\mathcal{C}(\Sigma_{\mathrm{COL}})_{s\to s'}$ and to be generated by the constructors obs_i^*. Note that the above definition follows the coinductive definition of observable contexts given in Definition 10.

[19] Otherwise, i.e. if $\mathcal{C}(\Sigma_{\mathrm{COL}})_{s\to s'}$ is empty, no new sort is added.
[20] The existence of the direct observer (obs, i) entails the non emptiness of $\mathcal{C}(\Sigma_{\mathrm{COL}})_{s_i\to s'}$, hence the existence of the new sort $Cont[s_i{\to}s']$.
[21] Hence, the new sort $Cont[s{\to}s']$ exists, and so does the new sort $Cont[s_i{\to}s']$.

Example 72. For instance, in the case of our STREAM example (see Example 59 above), we introduce a new sort *Cont*[*stream*→*elem*] and three new operations:

$$\begin{array}{lll} \mathit{head}^* & : & \to \mathit{Cont}[\mathit{stream}{\to}\mathit{elem}] \\ \mathit{tail}^* & : & \mathit{Cont}[\mathit{stream}{\to}\mathit{elem}] \to \mathit{Cont}[\mathit{stream}{\to}\mathit{elem}] \\ \mathit{apply} & : & \mathit{Cont}[\mathit{stream}{\to}\mathit{elem}],\ \mathit{stream} \to \mathit{elem} \end{array}$$

◇

Definition 73 (The functor $\varepsilon2^{\mathrm{Sig}}$). *The functor* $\varepsilon2^{\mathrm{Sig}} : \mathrm{Sign}_{\mathrm{IBB}} \to \mathrm{Sign}_{\mathrm{CFOLEq}}$ *is defined by (remember that* $\mathrm{Sign}_{\mathrm{IBB}} = \mathrm{Sign}_{\mathrm{COL}}$ *and* $\mathrm{Sign}_{\mathrm{CFOLEq}} = \mathrm{Sign}_{\mathrm{FOLEq}}$*):*

$$\begin{tikzcd} \Sigma_{1,\mathrm{COL}} \arrow[r, mapsto, "\varepsilon2^{\mathrm{Sig}}"] \arrow[d, "\sigma_{\mathrm{COL}}"'] & \Sigma_1^+ \arrow[d, "\sigma^+"] \\ \Sigma_{2,\mathrm{COL}} \arrow[r, mapsto, "\varepsilon2^{\mathrm{Sig}}"] & \Sigma_2^+ \end{tikzcd}$$

where $\sigma^+ : \Sigma_1^+ \to \Sigma_2^+$ *is defined as the following extension of the signature morphism* $\sigma : \Sigma_1 \to \Sigma_2$ *underlying the* COL*-signature morphism* σ_{COL}*:*

- *Each new sort* $\mathit{Cont}[s{\to}s']$ *is mapped by* σ^+ *to the sort* $\mathit{Cont}[\sigma(s){\to}\sigma(s')]$*;*
- *Each new operation* obs_i^* *is mapped by* σ^+ *to the operation* $\sigma(\mathit{obs})_i^*$ *;*
- *Each new operation* $\mathit{apply} : \mathit{Cont}[s{\to}s'], s \to s'$ *is mapped by* σ^+ *to the operation* $\mathit{apply} : \mathit{Cont}[\sigma(s){\to}\sigma(s')], \sigma(s) \to \sigma(s')$*.*

The above definition makes sense since the definition of COL-signature morphisms (see Definition 42) ensures that if the sort $\mathit{Cont}[s{\to}s']$ exists in Σ_1^+, then so does $\mathit{Cont}[\sigma(s){\to}\sigma(s')]$ in Σ_2^+. Similarly, if (obs, i) is an observer with profile $\mathit{obs} : s_1, \ldots, s_i, \ldots, s_n \to s$, then $(\sigma(\mathit{obs}), i)$ is an observer with profile $\sigma(s_1), \ldots, \sigma(s_i), \ldots, \sigma(s_n) \to \sigma(s')$.

Definition 74 (Functor $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}$ associated to a COL-signature). *For each* COL*-signature* Σ_{COL}*,* $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}} : \mathrm{Alg}(\Sigma) \to \mathrm{Alg}(\Sigma^+)$ *is defined as the functor which associates to any* Σ*-algebra* A *the* Σ^+*-free extension of* A *satisfying the following set of equations, hereafter denoted by* $\mathrm{Ax}_{\Sigma_{\mathrm{COL}}}[\mathit{apply}]$*:*

- *For each direct observer* $(\mathit{obs}, i) \in \mathrm{OP}_{\mathrm{Obs}}$ *with* $\mathit{obs} : s_1, \ldots, s_i, \ldots, s_n \to s'$*, the equation:*
  $\forall x_1{:}s_1, \ldots, x_n{:}s_n.$
  $$\mathit{apply}(\mathit{obs}_i^*(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n), x_i) = \mathit{obs}(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n).$$
- *For each indirect observer* $(\mathit{obs}, i) \in \mathrm{OP}_{\mathrm{Obs}}$ *with* $\mathit{obs} : s_1, \ldots, s_i, \ldots, s_n \to s$*, and for all observable sorts* $s' \in S_{\mathrm{Obs}}$ *such that the new sort* $\mathit{Cont}[s{\to}s']$ *exists, the equations:*
  $\forall c{:}\,\mathit{Cont}[s{\to}s'], x_1{:}s_1, \ldots, x_n{:}s_n.$
  $$\mathit{apply}(\mathit{obs}_i^*(c, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n), x_i) = \mathit{apply}(c, \mathit{obs}(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n)).$$

In general free extensions are defined up to isomorphisms only. Here we assume that the free extension $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}(A)$ of a Σ-algebra A is obtained by a canonical construction (basically, taking the quotient of the Σ^+-term algebra with variables in A by the smallest congruence generated by the equations defining *apply* and by the identities that hold in A, see [14, p. 189]).

Lemma 75 (Properties of $\mathrm{FreeExt}$**).** *For any* COL*-signature* Σ_{COL}*, any* Σ*-algebras* A *and* A'*, any* COL*-signature morphism* $\sigma_{\mathrm{COL}} : \Sigma_{1,\mathrm{COL}} \to \Sigma_{2,\mathrm{COL}}$ *and for any* Σ_2*-algebra* A_2 *:*

(1) The functor $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}$ *is strongly persistent, i.e.,* $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}(A)|_{\Sigma} = A$.
(2) If $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}(A)$ *is isomorphic to* $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}(A')$*, then* A *is isomorphic to* A'*.*
(3) $\mathrm{FreeExt}_{\Sigma_{2,\mathrm{COL}}}(A_2)|_{\sigma^+} = \mathrm{FreeExt}_{\Sigma_{1,\mathrm{COL}}}(A_2|_{\sigma})$.

Proof. *(1)* results from the fact that the operations *apply* are defined in a sufficiently complete inductive way w.r.t. the constructors of the sorts $\mathit{Cont}[s{\to}s']$. *(2)* follows from *(1)* and the functoriality of the forgetful functor. *(3)* follows from the fact that our free extensions are obtained by canonical constructions and that, due to the definition of COL-signature morphisms and to Definition 74, we have $\sigma^+(\mathrm{Ax}_{\Sigma_{1,\mathrm{COL}}}[\mathit{apply}]) \subseteq \mathrm{Ax}_{\Sigma_{2,\mathrm{COL}}}[\mathit{apply}]$. □

Example 76. In the case of our STREAM example (see Example 59), $\mathrm{Ax}_{\Sigma_{\mathrm{STREAM}}}$ contains two equations:

$\forall S{:}\mathit{stream}.\ \mathit{apply}(\mathit{head}^*, S) = \mathit{head}(S)$
$\forall c{:}\mathit{Cont}[\mathit{stream}{\to}\mathit{elem}], S{:}\mathit{stream}.\ \mathit{apply}(\mathit{tail}^*(c), S) = \mathit{apply}(c, \mathit{tail}(S)).$ ◇

The encoding $\varepsilon 2$ from IBB to CFOLEq can now be defined as follows.

Definition 77 (Institution encoding $\varepsilon 2$ **from** IBB **to** CFOLEq**).** *The institution encoding* $\varepsilon 2 : \mathrm{IBB} \to \mathrm{CFOLEq}$ *between the institutions* IBB *and* CFOLEq *is defined by:*

- *the functor* $\varepsilon 2^{\mathrm{Sig}} : \mathrm{Sign}_{\mathrm{IBB}} \to \mathrm{Sign}_{\mathrm{CFOLEq}}$ *(see Definition 73),*
- *the natural transformation* $\varepsilon 2^{\mathrm{Sen}} : \mathrm{Sen}_{\mathrm{IBB}} \to \varepsilon 2^{\mathrm{Sig}}; \mathrm{Sen}_{\mathrm{CFOLEq}}$ *induced by the inclusions* $\mathrm{Sen}_{\mathrm{IBB}}(\Sigma_{\mathrm{COL}}) \subseteq \mathrm{Sen}_{\mathrm{CFOLEq}}(\Sigma^+)$*, for any* COL*-signature* Σ_{COL}*,* [22]
- *the natural transformation* $\varepsilon 2^{\mathrm{Mod}} : \mathrm{Mod}_{\mathrm{IBB}} \to (\varepsilon 2^{\mathrm{Sig}})^{\mathrm{op}}; \mathrm{Mod}_{\mathrm{CFOLEq}}$ *induced by the (restriction to* $\mathrm{Mod}_{\mathrm{IBB}}(\Sigma_{\mathrm{COL}}) = \mathrm{Alg}_{\mathrm{rfa}(\Sigma_{\mathrm{COL}})}(\Sigma) \subseteq \mathrm{Alg}(\Sigma)$ *of the) functors* $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}$ *(see Definition 74).*

The fact that $\varepsilon 2^{\mathrm{Mod}}$ is a natural transformation results from Lemma 75(3),

[22] Remember that $\mathrm{Sen}_{\mathrm{IBB}}(\Sigma_{\mathrm{COL}}) = \mathrm{Sen}_{\mathrm{FOLEq}}(\Sigma)$ and that $\Sigma \subseteq \Sigma^+$.

which ensures that the following diagram is commutative:[23]

$$\begin{tikzcd}
\Sigma_{2,\mathrm{COL}} & \mathrm{Alg}_{\mathrm{rfa}(\Sigma_{2,\mathrm{COL}})}(\Sigma_2) \arrow[r, "\mathrm{FreeExt}_{\Sigma_{2,\mathrm{COL}}}"] \arrow[d, "--|_{\sigma}"'] \arrow[dr, phantom, "="] & \mathrm{Alg}(\Sigma_2^+) \arrow[d, "--|_{\sigma^+}"] \\
\Sigma_{1,\mathrm{COL}} \arrow[u, "\sigma_{\mathrm{COL}}"] & \mathrm{Alg}_{\mathrm{rfa}(\Sigma_{1,\mathrm{COL}})}(\Sigma_1) \arrow[r, "\mathrm{FreeExt}_{\Sigma_{1,\mathrm{COL}}}"] & \mathrm{Alg}(\Sigma_1^+)
\end{tikzcd}$$

Lemma 78. *The encoding condition holds for $\varepsilon 2$.*

Proof. Let Σ_{COL} be an arbitrary COL-signature, φ be an arbitrary first-order Σ-sentence, and $A \in \mathrm{Alg}_{\mathrm{rfa}(\Sigma_{\mathrm{COL}})}(\Sigma)$. Since, for first-order sentences, the satisfaction relation of both IBB and CFOLEq is just the standard satisfaction relation of FOLEq, the encoding condition reads as follows:

$$\underbrace{A \models_\Sigma \varphi}_{\text{in FOLEq}} \text{ if and only if } \underbrace{\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}(A) \models_{\Sigma^+} \varphi}_{\text{in FOLEq}} .$$

But this is obvious, since $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}(A) \models_{\Sigma^+} \varphi$ iff (by the satisfaction condition for the institution FOLEq) $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}(A)|_\Sigma \models_\Sigma \varphi$ iff (since the free functor $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}$ is strongly persistent) $A \models_\Sigma \varphi$. □

Remark 79. Note that the definition of the free functors $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}$ used in the institution encoding $\varepsilon 2$ relies only on the observers and not on the constructors. When Σ_{COL} contains only constructors, i.e., $\Sigma_{\mathrm{COL}} = (\Sigma, \mathrm{OP}_{\mathrm{Cons}}, \emptyset)$, then the free functor $\mathrm{FreeExt}_{\Sigma_{\mathrm{COL}}}$ is trivial.

It follows directly from Lemma 75(2) that $\varepsilon 2$ is iso-reflecting. We now prove that $\varepsilon 2$ is a logical institution encoding.

Lemma 80. *The institution encoding* $\varepsilon 2 : \mathrm{IBB} \to \mathrm{CFOLEq}$ *is a logical institution encoding characterized by the family* $\langle \mathrm{ENC}(\Sigma_{\mathrm{COL}}) \rangle_{\Sigma_{\mathrm{COL}} \in \mathrm{Sign}_{\mathrm{COL}}}$, *where for each* COL-*signature* $\Sigma_{\mathrm{COL}} = (\Sigma, \mathrm{OP}_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Obs}})$, $\mathrm{ENC}(\Sigma_{\mathrm{COL}})$ *is the set of* Σ^+*-sentences in* $\mathrm{Sen}_{\mathrm{CFOLEq}}(\Sigma^+)$ *containing:*

- *the sort-generation constraint* $\mathrm{SGC}(S_{\mathrm{Cons}}, \mathrm{OP}_{\mathrm{Cons}})$ *induced by the declared constructors* $\mathrm{OP}_{\mathrm{Cons}}$ *, and*
- *the free sort-generation constraint* $\mathrm{FSGC}(\Delta(\mathrm{OP}_{\mathrm{Obs}}))$ *induced by the signature fragment* $\Delta(\mathrm{OP}_{\mathrm{Obs}})$ *defined in Definition 71, and*
- *the set of equations* $\mathrm{Ax}_{\Sigma_{\mathrm{COL}}}[\mathit{apply}]$ *defined in Definition 74, and*

[23] Remember that for any COL-signature Σ_{COL}, $\mathrm{Mod}_{\mathrm{IBB}}(\Sigma_{\mathrm{COL}}) = \mathrm{Alg}_{\mathrm{rfa}(\Sigma_{\mathrm{COL}})}(\Sigma)$ and for any signature Σ, $\mathrm{Mod}_{\mathrm{CFOLEq}}(\Sigma) = \mathrm{Alg}(\Sigma)$.

- *for each state sort $s \in S_{\text{State}}$, the first-order sentence* $\text{fa}(s)$*:*

$$\forall x, y : s. \left(\bigwedge_{Cont[s \to s']} \forall c : Cont[s \to s'].\ apply(c, x) = apply(c, y) \right) \Rightarrow x = y\,.$$[24]

Proof. According to Definition 62, we have to prove that, for each COL-signature Σ_{COL}, $\text{Iso}_{\Sigma^+}(\varepsilon 2^{\text{Mod}}_{\Sigma_{\text{COL}}}(\text{Mod}_{\text{IBB}}(\Sigma_{\text{COL}}))) = \mathcal{M}od[\langle \Sigma^+, \text{ENC}(\Sigma_{\text{COL}})\rangle]$. We have $\varepsilon 2^{\text{Mod}}_{\Sigma_{\text{COL}}}(\text{Mod}_{\text{IBB}}(\Sigma_{\text{COL}})) = \{\text{FreeExt}_{\Sigma_{\text{COL}}}(A) \mid A \in \text{Alg}_{\text{rfa}(\Sigma_{\text{COL}})}(\Sigma)\}$, according to Definition 77.
$\subseteq$: Let $B \in \text{Iso}_{\Sigma^+}(\varepsilon 2^{\text{Mod}}_{\Sigma_{\text{COL}}}(\text{Mod}_{\text{IBB}}(\Sigma_{\text{COL}})))$. Then B is isomorphic to $B' = \text{FreeExt}_{\Sigma_{\text{COL}}}(A)$, for some Σ-algebra A reachable and fully abstract w.r.t. Σ_{COL}. Since A is reachable, $A \models \text{SGC}(S_{\text{Cons}}, \text{OP}_{\text{Cons}})$, thus $B' \models \text{SGC}(S_{\text{Cons}}, \text{OP}_{\text{Cons}})$. Definition 74 entails that $B' \models \text{FSGC}(\Delta(\text{OP}_{\text{Obs}}))$ (due to the freeness of $\text{FreeExt}_{\Sigma_{\text{COL}}}$) and that $B' \models \text{Ax}_{\Sigma_{\text{COL}}}[apply]$. Now, since A is fully abstract, for any state sort $s \in S_{\text{State}}$ and values $a, b \in A_s$, $a \approx_{\Sigma_{\text{COL}}, A} b$ iff $a = b$. But Definition 71 ensures that there is (up to variable renaming) a one to one correspondence between contexts $c \in \mathcal{C}(\Sigma_{\text{COL}})_{s \to s'}$ and constructor terms of sort $Cont[s \to s']$. Hence, since A is reachable and $B' \models \text{FSGC}(\Delta(\text{OP}_{\text{Obs}}))$, there is a one to one correspondence between a context $c \in \mathcal{C}(\Sigma_{\text{COL}})_{s \to s'}$ and a valuation $\alpha : \text{Var}(c) \to \text{Gen}_{\Sigma_{\text{COL}}}(A)$ on the one hand, and a value in $B'_{Cont[s \to s']}$ on the other. This is enough to conclude that since A is fully abstract, $B' \models \text{fa}(s)$ for all state sorts $s \in S_{\text{State}}$. Thus $B' \in \mathcal{M}od[\langle \Sigma^+, \text{ENC}(\Sigma_{\text{COL}})\rangle]$, and the same holds for the isomorphic algebra B.
$\supseteq$: Let $B \in \mathcal{M}od[\langle \Sigma^+, \text{ENC}(\Sigma_{\text{COL}})\rangle]$ and let $A = B|_{\Sigma}$. Since by hypothesis $B \models \text{FSGC}(\Delta(\text{OP}_{\text{Obs}}))$ and $B \models \text{Ax}_{\Sigma_{\text{COL}}}[apply]$, we can conclude that B is isomorphic to $B' = \text{FreeExt}_{\Sigma_{\text{COL}}}(A)$. Since $B \models \text{SGC}(S_{\text{Cons}}, \text{OP}_{\text{Cons}})$, so does B', hence so does also A, which means that A is reachable. Now, a reasoning similar to the one above for ($\subseteq$) shows that since $B \models \text{fa}(s)$, so does the isomorphic algebra B', and therefore for all state sorts $s \in S_{\text{State}}$, A must be fully abstract. Hence $A \in \text{Alg}_{\text{rfa}(\Sigma_{\text{COL}})}(\Sigma)$. □

Corollary 81. *The institution encoding $\varepsilon 2 : \text{IBB} \to \text{CFOLEq}$ induces the following structured specification encoding $\widehat{\varepsilon 2}$ of structured specifications over* IBB *into structured specifications over* CFOLEq*:*

- $\widehat{\varepsilon 2}(\langle \Sigma_{\text{COL}}, \text{Ax}\rangle) \stackrel{\text{def}}{=} \langle \Sigma^+, \text{Ax} \cup \text{ENC}(\Sigma_{\text{COL}})\rangle$,
- $\widehat{\varepsilon 2}(\text{SP}_1 \cup \text{SP}_2) \stackrel{\text{def}}{=} \widehat{\varepsilon 2}(\text{SP}_1) \cup \widehat{\varepsilon 2}(\text{SP}_2)$,
- *let $\sigma_{\text{COL}} : \mathcal{S}ig[\text{SP}] \to \Sigma_{\text{COL}}$ be a* COL*-signature morphism (and hence an* IBB*-signature morphism);*
  $\widehat{\varepsilon 2}(\textbf{translate}\ \text{SP}\ \textbf{by}\ \sigma_{\text{COL}}) \stackrel{\text{def}}{=}$
  $$(\textbf{translate}\ \widehat{\varepsilon 2}(\text{SP})\ \textbf{by}\ \sigma^+) \cup \langle \Sigma^+, \text{ENC}(\Sigma_{\text{COL}})\rangle,$$

[24] The sentence $\text{fa}(s)$ is finite, since for any state sort $s \in S_{\text{State}}$, there is only a finite number of sorts $Cont[s \to s']$, where $s' \in S_{\text{Obs}}$ is an observable sort. Note also the obvious correspondence between the finitary sentence $\text{fa}(s)$ and the infinitary sentence $\text{FA}(\Sigma_{\text{COL}})_s$ introduced at the end of Section 6.

- *let* $\sigma_{\mathrm{COL}} : \Sigma_{\mathrm{COL}} \to \mathcal{S}ig[\mathrm{SP}]$ *be a* COL*-signature morphism (and hence an* IBB*-signature morphism);*
  $\widehat{\varepsilon 2}($**derive from** SP **by** $\sigma_{\mathrm{COL}}) \overset{\mathrm{def}}{=}$ **derive from** $\widehat{\varepsilon 2}(\mathrm{SP})$ **by** σ^{+}.

In summary, the iso-reflecting logical encoding $\varepsilon 2$ satisfies the hypotheses of Theorem 64, and hence Corollary 65 provides the following result.

Corollary 82. *For any structured specification* $\mathrm{SP}_{\mathrm{COL}}$ *over the* IBB *institution with signature* Σ_{COL}*, and any* Σ*-sentence* φ*, we have:*

$$\underbrace{\mathrm{SP}_{\mathrm{COL}} \models_{\Sigma_{\mathrm{COL}}} \varphi}_{\text{in IBB}} \quad \textit{if and only if} \quad \underbrace{\widehat{\varepsilon 2}(\mathrm{SP}_{\mathrm{COL}}) \models_{\Sigma^{+}} \varphi}_{\text{in CFOLEq}} .$$

The above corollary, together with Theorem 69, leads to our final main result.

Theorem 83. *For any structured specification* $\mathrm{SP}_{\mathrm{COL}}$ *over the* COL *institution with signature* Σ_{COL}*, and any first-order* Σ*-sentence* φ*, we have:*

$$\underbrace{\mathrm{SP}_{\mathrm{COL}} \models_{\Sigma_{\mathrm{COL}}} \varphi}_{\text{in COL}} \quad \textit{if and only if} \quad \underbrace{\widehat{\varepsilon 2}(\mathrm{SP}_{\mathrm{COL}}) \models_{\Sigma^{+}} \varphi}_{\text{in CFOLEq}} .$$

Theorem 83 means that we can reuse any available theorem prover for proving consequences of structured CFOLEq-specifications to prove consequences of structured COL-specifications. Even if we know that, due to the presence of sort-generation constraints, there is no formal complete proof system for CFOLEq, there are plenty of theorem provers for first-order logic with equality and sort-generation constraints, where various proof by induction techniques are available, and can therefore be reused for free for COL-specifications, for instance PVS [30] or the Larch Prover [19].

It is also important to note that the encoded specification $\widehat{\varepsilon 2}(\mathrm{SP}_{\mathrm{COL}})$ is still a structured specification, with a structure very close to the one of $\mathrm{SP}_{\mathrm{COL}}$. Thus proof techniques guided by the structure of the specification can be applied as well. Moreover, our encoding is "efficient" in the sense that it needs few extra symbols and axioms (mainly the equations $\mathrm{Ax}_{\Sigma_{\mathrm{COL}}}[\mathit{apply}]$ and the sentences $\mathrm{fa}(s)$).

Example 84. The encoding of our running STREAM example (see Example 59) is the following CFOLEq-specification (using the syntactic abbreviations provided by CASL):

spec STREAM-ENC = STREAM[25] **then**
  free type $\mathit{Cont}[\mathit{stream}\to\mathit{elem}] ::= \mathit{head}^{*} \mid \mathit{tail}^{*}(\mathit{Cont}[\mathit{stream}\to\mathit{elem}])$;
  op $\mathit{apply} : \mathit{Cont}[\mathit{stream}\to\mathit{elem}] \times \mathit{stream} \to \mathit{elem}$;

[25] Without the **observers** *head*, *tail* clause.

axioms
$\forall S, S' : \mathit{stream};\ c : \mathit{Cont}[\mathit{stream} \rightarrow \mathit{elem}]$

- $\mathit{apply}(\mathit{head}^*, S) = \mathit{head}(S)$ (apply-1)
- $\mathit{apply}(\mathit{tail}^*(c), S) = \mathit{apply}(c, \mathit{tail}(S))$ (apply-2)
- $(\forall c : \mathit{Cont}[\mathit{stream} \rightarrow \mathit{elem}].\ \mathit{apply}(c, S) = \mathit{apply}(c, S')\ )$
  $\Rightarrow S = S'$ (fa)

end
For instance, according to Theorem 83, to prove that:
$\textsc{Stream} \models_{\mathrm{COL}} \mathit{zip}(\mathit{odd}(S), \mathit{even}(S)) = S$
is equivalent to prove that:
$\textsc{Stream-Enc} \models_{\mathrm{CFOLEq}} \mathit{zip}(\mathit{odd}(S), \mathit{even}(S)) = S$.
To prove this, we use (fa), which leads to the new goal (where all variables are implicitly universally quantified):
$\mathit{apply}(c,\ \mathit{zip}(\mathit{odd}(S), \mathit{even}(S))) = \mathit{apply}(c,\ S)$.
We start an induction on c w.r.t. the constructors head^* and tail^* of the sort $\mathit{Cont}[\mathit{stream} \rightarrow \mathit{elem}]$.

Basic case head^***:**
We have to prove: $\mathit{apply}(\mathit{head}^*,\ \mathit{zip}(\mathit{odd}(S), \mathit{even}(S))) = \mathit{apply}(\mathit{head}^*,\ S)$
which reduces, using (apply-1), to: $\mathit{head}(\mathit{zip}(\mathit{odd}(S), \mathit{even}(S))) = \mathit{head}(S)$
which reduces to a trivial equality using the axioms of Stream.
Induction step tail^***:**
The induction hypothesis is:
$\mathit{apply}(\mathit{c0},\ \mathit{zip}(\mathit{odd}(S), \mathit{even}(S))) = \mathit{apply}(\mathit{c0},\ S)$
where $\mathit{c0}$ is a fresh constant of sort $\mathit{Cont}[\mathit{stream} \rightarrow \mathit{elem}]$ and S is a universally quantified variable of sort *container*.
The new goal is:
$\mathit{apply}(\mathit{tail}^*(\mathit{c0}),\ \mathit{zip}(\mathit{odd}(S), \mathit{even}(S))) = \mathit{apply}(\mathit{tail}^*(\mathit{c0}),\ S)$
which reduces, using (apply-2), to:
$\mathit{apply}(\mathit{c0},\ \mathit{tail}(\mathit{zip}(\mathit{odd}(S), \mathit{even}(S)))) = \mathit{apply}(\mathit{c0},\ \mathit{tail}(S))$
Using the axioms of Stream, we obtain:
$\mathit{apply}(\mathit{c0},\ \mathit{zip}(\mathit{even}(S), \mathit{odd}(\mathit{tail}(\mathit{tail}(S))))) = \mathit{apply}(\mathit{c0},\ \mathit{tail}(S))$
Since the axiom defining *even* provides the rewrite rule
(R) $\mathit{odd}(\mathit{tail}(S)) \rightarrow \mathit{even}(S)$, we further obtain:
$\mathit{apply}(\mathit{c0},\ \mathit{zip}(\mathit{even}(S), \mathit{even}(\mathit{tail}(S)))) = \mathit{apply}(\mathit{c0},\ \mathit{tail}(S))$
Now we conclude since the current goal is an instance of the induction hypothesis, with S instantiated by $\mathit{tail}(S)$, again rewritten with (R).

Indeed the full proof is easily automated, as shown by the following Larch Prover proof script.

```
set immunity ancestor
set name zip
prove zip(odd(S),even(S)) = S
  instantiate S by S, S’ by zip(odd(S),even(S)) in fa
  resume by lemma \A c (apply(c,S)=apply(c,zip(odd(S),even(S))))
    resume by induction on c
      instantiate S by tail(S) in *InductHyp
qed
```

As illustrated by this example, the key step is the induction on the variable c of sort $\mathit{Cont}[\mathit{stream}{\to}\mathit{elem}]$. This standard constructor induction mimics an induction on the observable contexts, hence some kind of *context induction.* The main reason for our proofs to remain simple is that we have chosen an adequate coinductive definition of observable contexts which then leads to an adequate constructor induction scheme when working with the encoded specification. Moreover, since we only encode the observable contexts induced by the chosen observers, we obtain a fairly simple and efficient encoding. As a last remark it is interesting to note that a careful comparison of our proof steps above with the similar proof reported in [17] shows that they are very similar, which convinces us that so-called *circular coinduction* corresponds to context induction with an appropriate context induction scheme. ◇

Example 85. Let us consider again the CONTAINER example discussed in Section 3.3, Example 37. Its encoding is the following CFOLEq-specification:

spec CONTAINER-ENC = CONTAINER[26] **then**
  generated types $\mathit{bool} ::= \mathit{true} \mid \mathit{false};\ \mathit{nat} ::= \mathit{0} \mid \mathit{succ}(\mathit{nat});$
    $\mathit{container} ::= \mathit{empty} \mid \mathit{insert}(\mathit{container};\ \mathit{elem});$
  free type $\mathit{Cont}[\mathit{container}{\to}\mathit{bool}] ::= \mathit{isin}^*(\mathit{nat});$
  op $\mathit{apply} : \mathit{Cont}[\mathit{container}{\to}\mathit{bool}] \times \mathit{container} \to \mathit{bool};$
  axioms
  $\forall x : \mathit{nat};\ c, c' : \mathit{container};\ \mathit{ctx} : \mathit{Cont}[\mathit{container}{\to}\mathit{bool}]$
  • $\mathit{apply}(\mathit{isin}^*(x), c) = \mathit{isin}(c, x)$ (apply)
  • $(\forall \mathit{ctx} : \mathit{Cont}[\mathit{stream}{\to}\mathit{elem}].\ \mathit{apply}(\mathit{ctx}, c) = \mathit{apply}(\mathit{ctx}, c')\ )$
    $\Rightarrow c = c'$ (fa)

end

In Example 37 we have claimed that the constructor complete definition of *remove* given by the axioms (4) - (6) can be replaced by the observer complete definition given by the formulas (7) and (8) without changing the semantics of the specification CONTAINER. Now, let CONTAINER′ be the specification obtained from CONTAINER by replacing the axioms (4) - (6) by the formulas (7) and (8). Then we have to show:
(A) CONTAINER $\models_{\mathrm{COL}}$ (7) $\wedge$ (8) and
(B) CONTAINER′ $\models_{\mathrm{COL}}$ (4) $\wedge$ (5) $\wedge$ (6).
According to Theorem 83, proving (A) is equivalent to prove:
(A-ENC) CONTAINER-ENC $\models_{\mathrm{CFOLEq}}$ (7) $\wedge$ (8), which follows from an easy proof by induction w.r.t. the constructors *empty* and *insert* of the sort *container*, not detailed here. Similarly, proving (B) is equivalent to prove:

[26] Without the **constructors** and **observer** clause.

(B-ENC) $\text{Container}'\text{-Enc} \models_{\mathrm{CFOLEq}} (4) \wedge (5) \wedge (6)$, where Container′-Enc is similar to Container-Enc but extends Container′ instead of Container. Let us for instance consider the proof of (5). As in the previous example, we use (fa) to derive the new goal:
$\mathit{apply}(\mathit{ctx},\ \mathit{remove}(\mathit{insert}(c,x),x)) = \mathit{apply}(\mathit{ctx},\ \mathit{remove}(c,x))$.
A (trivial) induction on *ctx* leads to the new goal:
$\mathit{apply}(\mathit{isin}^*(y),\ \mathit{remove}(\mathit{insert}(c,x),x)) = \mathit{apply}(\mathit{isin}^*(y),\ \mathit{remove}(c,x))$.
Using (apply), we obtain:
$\mathit{isin}(\mathit{remove}(\mathit{insert}(c,x),x),\ y) = \mathit{isin}(\mathit{remove}(c,x),\ y)$.
Case $x = y$:
$\mathit{isin}(\mathit{remove}(\mathit{insert}(c,x),x),y) \overset{\text{by (7)}}{=} \mathit{false} \overset{\text{by (7)}}{=} \mathit{isin}(\mathit{remove}(c,x),y)$.
Case $x \neq y$:
$\mathit{isin}(\mathit{remove}(\mathit{insert}(c,x),x),y) \overset{\text{by (8)}}{=} \mathit{isin}(\mathit{insert}(c,x),y) \overset{\text{by (3)}}{=} \mathit{isin}(c,y) \overset{\text{by (8)}}{=} \mathit{isin}(\mathit{remove}(c,x),y)$.
which concludes the proof of (5). The proofs of (4) and (6) are similar. $\diamond$

8 Conclusion and Future Work

We have presented a logical, institution-based framework, called the COL institution, which formalizes and integrates notions of reachability and observability that are useful in software development. While observability concepts provide a means to specify the observable behavior of a software system in an abstract, implementation independent way, reachability concepts focus on those data which are relevant from the user's point of view. Our approach is fairly general (supporting structured specifications and full first-order logic) and results in practically useful proof techniques for proving behavioral properties of specifications.

The essential assumptions of this work are that a signature contains distinguished sets of constructors and of observers and that the semantics of a specification describes all correct realizations (formally represented by the class of the models of the specification). The declaration of constructor and observer operations induces several advantages:

(1) It supports a clear specification methodology where, from the reachability point of view, operations can be inductively defined by a (standard) case distinction w.r.t. the given constructors and, from the observability point of view, operations can be coinductively defined by specifying their observable effects w.r.t. the given observers. We have seen that in the case of constrained state sorts both specification styles are equivalent w.r.t. the given semantics.
(2) In contrast to earlier approaches which were based on observable sorts and on input sorts only (see [4]), the introduction of observers and con-

structors leads to a powerful notion of signature morphism which ensures that the satisfaction condition of institutions is satisfied. Thus we obtain the formal basis for defining structured specifications which guarantee the encapsulation of behavioral properties.

(3) Moreover, the distinguished observer and constructor operations lead to a smaller subset of observable contexts and constructors terms and thus to simpler proof methods than those considered in [4]. Indeed we have shown that using the "right" coinductive definition of the observable contexts we obtain a practically useful syntactic encoding principle which embeds the fact that context induction is the same as structural induction on context sorts. Thus any inductive theorem prover can be used to prove behavioral consequences of a specification.

A main topic of future work is to consider refinement relations between structured COL-specifications and to study proof methods for refinements. In contrast to the horizontal structuring mechanisms expressed by the specification-building operators, vertical structuring in the sense of refinements cannot be based on COL-signature morphisms since usually a more concrete specification uses a different set of constructors and/or observers than a given (abstract) specification does. For a proper solution we are interested in a component-based framework which will incorporate ideas of CASL architectural specifications [11].

Acknowledgements

We are very grateful to Michal Misiak and the anonymous referees of a previous version of this paper for their detailed remarks and suggestions for improvements.

References

[1] Egidio Astesiano, Michel Bidoit, Hélène Kirchner, Bernd Krieg-Brückner, Peter D. Mosses, Donald Sannella, and Andrzej Tarlecki. CASL: The Common Algebraic Specification Language. *Theoretical Computer Science*, 286(2):153–196, 2002.

[2] Egidio Astesiano, Hans-Jörg Kreowski, and Bernd Krieg-Brückner, editors. *Algebraic Foundations of Systems Specification.* Springer, 1999.

[3] Michel Bidoit, María Victoria Cengarle, and Rolf Hennicker. Proof systems for structured specifications and their refinements. In *[2]*, chapter 11, pages 385–433. Springer, 1999.

[4] Michel Bidoit and Rolf Hennicker. Behavioural theories and the proof of behavioural properties. *Theoretical Computer Science*, 165(1):3–55, 1996.

[5] Michel Bidoit and Rolf Hennicker. Observer complete definitions are behaviourally coherent. In *Proc. OBJ/CafeOBJ/Maude Workshop at Formal Methods'99, Toulouse, France, Sep. 1999*, pages 83–94. THETA, 1999. Preliminary long version available as Report LSV–99–4 at `www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-1999-4.rr.ps`.

[6] Michel Bidoit and Rolf Hennicker. On the integration of observability and reachability concepts. In M. Nielsen and U. Engberg, editors, *Proc. 5th Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS'2002), Grenoble, France, Apr. 2002*, volume 2303 of *Lecture Notes in Computer Science*, pages 21–36. Springer, 2002.

[7] Michel Bidoit, Rolf Hennicker, and Alexander Kurz. On the duality between observability and reachability. In F. Honsell and M. Miculan, editors, *Proc. 4th Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS'01), Genova, Italy*, volume 2030 of *Lecture Notes in Computer Science*, pages 72–87. Springer, 2001.

[8] Michel Bidoit, Rolf Hennicker, and Alexander Kurz. Observational logic, constructor-based logic, and their duality. *Theoretical Computer Science*, 298(3):471–510, 2003.

[9] Michel Bidoit, Rolf Hennicker, and Martin Wirsing. Behavioural and abstractor specifications. *Science of Computer Programming*, 25(2–3):149–186, 1995.

[10] Michel Bidoit and Peter D. Mosses. CASL *User Manual – Introduction to Using the Common Algebraic Specification Language*, volume 2900 of *Lecture Notes in Computer Science*. Springer, 2004.

[11] Michel Bidoit, Donald Sannella, and Andrzej Tarlecki. Architectural specifications in CASL. *Formal Aspects of Computing*, 13(3–5):252–273, 2002.

[12] Tomasz Borzyszkowski. Logical systems for structured specifications. *Theoretical Computer Science*, 286(2):197–245, 2002.

[13] R. Diaconescu and K. Futatsugi. *CafeOBJ Report: The Language, Proof Techniques, and Methodologies for Object-Oriented Algebraic Specification*, volume 6 of *AMAST Series in Computing*. World Scientific, 1998.

[14] H. Ehrig and B. Mahr. *Algebraic Specification I*. Springer, 1985.

[15] J. Goguen and G. Roşu. Hiding more of hidden algebra. In J.M. Wing, J. Woodcock, and J. Davies, editors, *Proc. Formal Methods (FM'99)*, volume 1709 of *Lecture Notes in Computer Science*, pages 1704–1719. Springer, 1999.

[16] Joseph Goguen and Rod Burstall. Institutions: abstract model theory for specification and programming. *Journal of the ACM*, 39(1):95–146, 1992.

[17] Joseph Goguen, Kai Lin, and Grigore Roşu. Circular coinductive rewriting. In *Proc. 15th International Conference on Automated Software Engineering (ASE'00)*, pages 123–132. IEEE Computer Society, 2000.

[18] Joseph Goguen and Grigore Roşu. Institution morphisms. *Formal Aspects of Computing*, 13(3–5):274–307, 2002.

[19] John V. Guttag and James J. Horning. *Larch: Languages and Tools for Formal Specification*. Springer, 1993.

[20] Rolf Hennicker. Context induction: a proof principle for behavioural abstraction and algebraic implementations. *Formal Aspects of Computing*, 3(4):326–345, 1991.

[21] Rolf Hennicker and Michel Bidoit. Observational logic. In *Proc. 7th Int. Conf. Algebraic Methodology and Software Technology (AMAST'98), Amazonia, Brazil, Jan. 1999*, volume 1548 of *Lecture Notes in Computer Science*, pages 263–277. Springer, 1999.

[22] C.A.R. Hoare. Proofs of correctness of data representations. *Acta Informatica*, 1:271–281, 1972.

[23] Martin Hofmann and Donald Sannella. On behavioural abstraction and behavioural satisfaction in higher-order logic. *Theoretical Computer Science*, 167:3–45, 1996.

[24] H.J. Keisler. *Model Theory for Infinitary Logic*. North-Holland, 1971.

[25] J. Loeckx, H.-D. Ehrich, and M. Wolf. *Specification of Abstract Data Types*. Wiley and Teubner, 1996.

[26] T. Mossakowski, H. Reichel, M. Roggenbach, and L. Schröder. Algebraic-coalgebraic specification in CoCasl. In Martin Wirsing, Dirk Pattinson, and Rolf Hennicker, editors, *Recent Trends in Algebraic Development Techniques (Selected papers of the 16th WADT, September 2002)*, pages 376–392. Springer, 2003.

[27] Till Mossakowski. Relating Casl with other specification languages: the institution level. *Theoretical Computer Science*, 286(2):367–475, 2002.

[28] Peter D. Mosses, editor. Casl *Reference Manual*, volume 2960 of *Lecture Notes in Computer Science*. Springer, 2004.

[29] M.P. Nivela and F. Orejas. Initial behaviour semantics for algebraic specifications. In *Recent Trends in Data Type Specification*, volume 332 of *Lecture Notes in Computer Science*, pages 184–207. Springer, 1988.

[30] Sam Owre, John Rushby, Natarajan Shankar, and David Stringer-Calvert. PVS: An Experience Report. In D. Hutter, W. Stephan, P. Traverso, and M. Ullman, editors, *Applied Formal Methods—FM-Trends 98*, volume 1641 of *Lecture Notes in Computer Science*, pages 338–345. Springer, 1998.

[31] Peter Padawitz. Swinging data types: syntax, semantics, and theory. In *Recent Trends in Data Type Specification*, volume 1130 of *Lecture Notes in Computer Science*, pages 409–435. Springer, 1996.

[32] Peter Padawitz. Basic inference rules for algebraic and coalgebraic specifications. Technical report, September 2002. Available at `http://ls5-www.cs.uni-dortmund.de/~peter/WADT01.ps.gz`.

[33] Horst Reichel. Behavioural validity of conditional equations in abstract data types. In *Proc. of the Vienna Conference, June 1984, Contributions to General Algebra 3*, pages 301–324. Verlag B.G. Teubner, 1985.

[34] Grigore Rosu. Inductive behavioral proofs by unhiding. In H. Peter Gumm, editor, *Electronic Notes in Theoretical Computer Science*, volume 82. Elsevier, 2003.

[35] D. Sannella and A. Tarlecki. On observational equivalence and algebraic specification. *Journal of Computer and System Sciences*, 34:150–178, 1987.

[36] Donald Sannella and Andrzej Tarlecki. Specifications in an arbitrary institution. *Information and Computation*, 76:165–210, 1988.

[37] Andrzej Tarlecki. Institutions: An Abstract Framework for Formal Specification. In *[2]*, chapter 4, pages 105–130. Springer, 1999.

[38] Andrzej Tarlecki. Towards heterogeneous specifications. In D.M. Gabbay and M. de Rijke, editors, *Frontiers of Combining Systems 2 (Proc. FroCos'98)*, volume 7 of *Studies in Logic and Computation*, pages 337–360. Research Studies Press/Wiley, 2000.

[39] Martin Wirsing. Algebraic Specification. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, chapter 13, pages 676–788. Elsevier Science Publishers B.V., 1990.

[40] Martin Wirsing and Manfred Broy. A modular framework for specification and information. In J. Diaz and F. Orejas, editors, *Proc. TAPSOFT'89*, volume 351 of *Lecture Notes in Computer Science*, pages 42–73. Springer, 1989.