Behavioural Theories and the Proof of Behavioural Properties

Michel BIDOIT
Rolf HENNICKER*

Laboratoire d'Informatique, URA 1327 du CNRS
Ecole Normale Supérieure
*Institut für Informatik
Ludwig-Maximilians-Universität München

LIENS - 95 - 5

March 1995

Behavioural Theories and The Proof of Behavioural Properties

Michel Bidoit[1] and Rolf Hennicker[2]

[1] LIENS, C.N.R.S. U.R.A. 1327 & Ecole Normale Supérieure
45, Rue d'Ulm, F-75230 Paris Cedex 05, France
[2] Institut für Informatik, Ludwig-Maximilians-Universität München
Leopoldstr. 11B, D-80802 München, Germany

Abstract

Behavioural theories are a generalization of first-order theories where the equality predicate symbol is interpreted by a behavioural equality of objects (and not by their identity). In this paper we first consider arbitrary behavioural equalities determined by some (partial) congruence relation and we show how to reduce the behavioural theory of any class of Σ-algebras to (a subset of) the standard theory of some corresponding class of algebras. This reduction is the basis of a method for proving behavioural theorems whenever an axiomatization of the behavioural equality is provided. Then we focus on the important special case of (partial) observational equalities where two elements are observationally equal if they cannot be distinguished by observable computations over some set of input values. We provide general conditions under which an obvious infinite axiomatization of the observational equality can be replaced by a finitary one and we provide methodological guidelines for finding such finitary axiomatizations. As a consequence, any proof system for first-order logic can be used to prove the behavioural validity of first-order formulas w.r.t. a given (partial) observational equality.

Keywords: Algebraic specification - Observable behaviour - Theorem proving.

1 Introduction

Behavioural abstraction plays a prominent rôle in formal software development, since it provides a suitable basis for defining adequate correctness concepts (cf. e.g. [9], [17], [15], [19], [2], [16]). For instance, for proving the correctness of a program with respect to a given specification, many examples show that it is essential to abstract from internal implementation details and to rely only on the observable behaviour of the program.

Behavioural correctness concepts can be formalized using a behavioural logic, where the usual satisfaction relation of first-order logic with equality is generalized to a behavioural satisfaction relation determined by a (partial) congruence relation on any Σ-algebra. The most important examples of behavioural equalities are observational equalities relating any two elements of an algebra which cannot be distinguished by observable

computations. In the literature there are different definitions of observational equalities according to different choices of the inputs values that are allowed for observable computations. For instance [17] considers the "total" observational equality imposing no restriction on the input values while in [15] a "partial" observational equality is used where only observable inputs are allowed for observable computations. Since we consider arbitrary partial congruence relations both notions are captured by our general concept of behavioural equality.

For performing correctness proofs in a behavioural framework it is crucial to show that the axioms of a given specification are behaviourally satisfied by an implementation. In other words this means that the axioms of a given specification belong to the *behavioural theory* of the implementing specification where the behavioural theory of a class $\mathbf{C}$ of Σ-algebras is the set of all formulas that are behaviourally satisfied by all algebras of $\mathbf{C}$. Unfortunately it is usually difficult to prove that a formula ϕ belongs to the behavioural theory of a given class $\mathbf{C}$ of Σ-algebras. The behavioural satisfaction relation does not even fulfill the satisfaction condition of institutions. Therefore we are interested in finding "nice" characterizations of behavioural theories which allow us to prove behavioural theorems using standard proof techniques as implemented, for instance, in any available theorem prover for (standard) first-order logic. We split this task into two major parts: First we show how to reduce for any arbitrary behavioural equality and for any class $\mathbf{C}$ of Σ-algebras the behavioural theory of $\mathbf{C}$ to (a subset of) the standard theory of some corresponding class of algebras and we show that this reduction is useful for proving behavioural theorems if an axiomatization of the behavioural equality is provided. Then we focus on (partial) observational equalities and we provide general conditions under which an obvious infinite axiomatization of the observational equality can be replaced by a finitary one.

More precisely we proceed as follows. In a first step (Section 4) we provide a general construction (the so-called "lift operator") which introduces explicit predicate symbols for denoting the behavioural equality. We show that a formula ϕ is behaviourally valid in all algebras of $\mathbf{C}$ if and only if its lifted version $\mathcal{L}(\phi)$ is valid in the standard sense in all lifted algebras of $\mathcal{L}(\mathbf{C})$ (Theorem 24). The usefulness of this characterization of behavioural theories still depends on the possibility to prove standard theorems over $\mathcal{L}(\mathbf{C})$. Therefore we introduce in Section 5 a general notion of (infinitary) axiomatization of the behavioural equality and we show that such an axiomatization allows us to characterize the class $\mathcal{L}(\mathbf{C})$ of lifted algebras (Theorem 36). In particular, we see that given an axiomatizable class $\mathbf{C}$ (i.e. $\mathbf{C}$ is the model class of a flat standard specification) and given an axiomatization of the behavioural equality then $\mathcal{L}(\mathbf{C})$ is axiomatizable as well. However, we still have the problem that the axiomatization of the behavioural equality may be given by a set of *infinitary* formulas since in concrete examples (as in the case of observational equalities) only an infinitary axiomatization may be immediately deduced from the definition of the given behavioural equality. Hence in the next step (Section 6) we consider finitary axiomatizations of the behavioural equality with auxiliary hidden sorts and function symbols and we show that whenever such an axiomatization of the behavioural equality for a class $\mathbf{C}$ of Σ-algebras is provided, then one can prove behavioural

theorems over **C** using standard proof techniques (Theorem 38 and Example 6).

So far we have elaborated a general framework for the reduction of behavioural theories to standard theories. In the remainder of this work (Section 7 and 8) we study how a concrete finitary axiomatization (with hidden part) can be obtained in the case of (partial) observational equalities. For this purpose we set out from an obvious infinitary axiomatization of the partial observational equality which says that two elements a and b are observationally equal if they are denotable by terms which contain only input variables and if all applications of observable contexts to a and b yield the same result. Since there are usually infinitely many terms with input variables and also infinitely many observable contexts both properties are easily axiomatizable by infinitary formulas.

How to find a finitary axiomatization of an observational equality for a given class **C** of Σ-algebras is based on two ideas: On one hand one has to specify the definedness (i.e. denotability by terms with input variables) with the help of new (hidden) predicate symbols. On the other hand we suggest to reduce the set of all observable contexts to a smaller set of contexts which is sufficient for describing the observational equality on all algebras of **C**. We show that any smaller set of observable contexts which induces a congruence relation on the algebras of **C** is appropriate for this purpose (Proposition 41 & Theorem 42). If we can find a finite set of observable contexts with this property we can immediately derive a finitary axiomatization of the observational equality. If we can find only an appropriate infinite subset of the observable contexts then the equality induced by this infinite set has to be axiomatized using contexts with hidden function symbols (Theorem 46). As a last result we show that it is always possible to construct a finitary axiomatization with hidden part of any partial observational equality (Proposition 48). However, since this result relies on a technically complex encoding of the observational equality it is mainly of theoretical interest.

Finally we show in Section 8 how our method for proving the behavioural validity of arbitrary Σ-formulas with respect to a given observational equality can be applied to various concrete examples.

2 Basic Notions

We assume that the reader is familiar with algebraic specifications [7, 22]. The basic concepts and notations that will be used hereafter are briefly summarized in this section.

A (many sorted) *signature* Σ is a pair (S, F) where S is a set of *sorts* and F is a set of *function symbols*.[3] To each function symbol $f \in F$ is associated an *arity* $s_1 \ldots s_n \to s$ with $s, s_1, \ldots, s_n \in S$. If $n = 0$ then f is called *constant* of sort s. A *(total) Σ-algebra* $A = ((A_s)_{s \in S}, (f^A)_{f \in F})$ over a signature $\Sigma = (S, F)$ consists of a family of carrier sets $(A_s)_{s \in S}$ and a family of functions $(f^A)_{f \in F}$ such that, if f has arity $s_1 \ldots s_n \to s$, then f^A is a (total) function from $A_{s_1} \times \ldots \times A_{s_n}$ to A_s (if $n = 0$ then f^A denotes a constant

[3] In this paper we assume that both S and F are finite.

object of A_s). A Σ-algebra A' is a *subalgebra* of a Σ-algebra A if $A'_s \subseteq A_s$ for all $s \in S$ and if for all $f \in F$ the restriction of f^A to A' is the function $f^{A'}$. If $S' \subseteq S$ and $F' \subseteq F$ then $\Sigma' = (S', F')$ is called a *subsignature* of $\Sigma = (S, F)$. If Σ' is a subsignature of Σ, the *restriction* of a Σ-algebra A to the subsignature Σ', denoted by $A|_{\Sigma'}$, is defined by $(A|_{\Sigma'})_s = A_s$ for each $s \in S'$ and $f^{A|_{\Sigma'}} = f^A$ for each $f \in F'$. Throughout this paper we always assume that the carrier sets A_s of a Σ-algebra A are not empty. Σ-morphisms are defined as usual. The category of all Σ-algebras is denoted by $Alg(\Sigma)$.

Given an arbitrary S-sorted family $X = (X_s)_{s \in S}$ of sets X_s, $T_\Sigma(X)$ denotes the Σ-*term algebra freely generated by* X, the carrier sets of which are the sets $T_\Sigma(X)_s$ of *terms* of sort s (and with variables in X). In several occasions we will consider a subset $In \subseteq S$ and we will choose $X_s = \emptyset$ for all $s \in S \setminus In$ (and $X_s \neq \emptyset$ for all $s \in In$). In that case, due to the non empty carrier set requirement of above, we will always assume that the signature Σ is *sensible w.r.t.* In, i.e. that for all $s \in S \setminus In$ (and hence for all $s \in S$), there exists a term t of sort s which is built by function symbols of Σ and by the variables of the non-empty sets X_s with $s \in In$. Given a Σ-algebra A, a *valuation* $\alpha : X \to A$ is a family of mappings $(\alpha_s : X_s \to A_s)_{s \in S}$. Any valuation $\alpha : X \to A$ uniquely extends to a Σ-morphism $I_\alpha : T_\Sigma(X) \to A$, called the *interpretation* associated to α.

A *partial* Σ*-congruence* on a Σ-algebra A is a family $\approx_A = (\approx_{A,s})_{s \in S}$ of partial equivalence relations (i.e. symmetric and transitive relations) $\approx_{A,s}$ on A_s compatible with the signature Σ, i.e. for all $f \in F$ of arity $s_1 \ldots s_n \to s$, for all $a_i, b_i \in A_{s_i}$, if $a_i \approx_{A,s_i} b_i$ then $f^A(a_1, \ldots, a_n) \approx_{A,s} f^A(b_1, \ldots, b_n)$.[4] A Σ*-congruence* $\approx_A$ is *total* if for all a in A, $a \approx_A a$, i.e. all relations $\approx_{A,s}$ are reflexive. The "definition domain" of a partial congruence $\approx_A$, denoted by $\mathrm{Dom}(\approx_A)$, is defined by $\{a \in A \mid a \approx_A a\}$ and is a subalgebra of A (moreover the restriction of $\approx_A$ to $\mathrm{Dom}(\approx_A)$ is a total Σ-congruence on $\mathrm{Dom}(\approx_A)$). In the sequel $A/\approx_A$ denotes the quotient algebra of $\mathrm{Dom}(\approx_A)$ by $\approx_A$.

In the sequel of this paper we assume given an arbitrary but fixed family $X = (X_s)_{s \in S}$ of countably infinite sets X_s of variables of sort $s \in S$. *First-order* Σ*-formulas* are defined as usual, from equations $l = r$, the logical connectives $\neg, \wedge, \vee, \ldots$ and the quantifiers $\forall, \exists$. We will also use *infinitary* Σ*-formulas* of the form $\bigwedge_{i \in I} \phi_i$ and $\bigvee_{i \in I} \phi_i$, where $(\phi_i)_{i \in I}$ is a countable family of Σ-formulas. A Σ*-sentence* is a Σ-formula which contains no free variable. In the sequel we will use the following abbreviations: For any term $t \in T_\Sigma(X)$, $\mathrm{Var}(t)$ denotes the set of variables occurring in t, and similarly $\mathrm{Var}(l, r)$ for a couple of terms l, r. Hence a universally quantified equation will be denoted by $\forall \mathrm{Var}(l, r).\, l = r$. Moreover, $\mathrm{FreeVar}(\phi)$ denotes the set of the free variables of the formula ϕ.

The *(standard) satisfaction* of a Σ-formula ϕ (finitary or not) by a Σ-algebra A, denoted by $A \models \phi$, is defined as usual in the first-order predicate calculus: the predicate symbol $=$ is interpreted in the carrier sets of the algebra by the set-theoretic equality. Due to the requirement of non-empty carrier sets, no pathological situations can occur with respect to the satisfaction relation (cf. [13]).

[4] In the sequel, for sake of clarity, we will often omit the subscript s and write $a \approx_A b$ instead of $a \approx_{A,s} b$.

A *basic (algebraic) specification* SP is a tuple $\langle \Sigma, \mathcal{A}x \rangle$ where $\Sigma = (S, F)$ is a signature and $\mathcal{A}x$ is a set of Σ-sentences, called *axioms* of SP. The *model class* of SP, denoted by $Mod(SP)$, is the class of all Σ-algebras which satisfy the axioms of SP, i.e. $Mod(SP) \stackrel{\text{def}}{=} \{A \in Alg(\Sigma) \mid A \models \phi \text{ for all } \phi \in \mathcal{A}x\}$.

A specification language is called *ASL-like* if to any specification SP is associated a signature, denoted by $Sig(SP)$, and a class of models, denoted by $Mod(SP)$, such that $Mod(SP) \subseteq Alg(Sig(SP))$, and if the language contains basic specifications as defined above and (at least) an operator $+$ for the combination of specifications SP and SP' such that $Sig(SP + SP') = Sig(SP) \cup Sig(SP')$ and $Mod(SP + SP') = \{A \in Alg(Sig(SP + SP')) \mid A|_{Sig(SP)} \in Mod(SP) \text{ and } A|_{Sig(SP')} \in Mod(SP')\}$. In the following, by an ASL-like structured specification SP we always refer to a specification written with such an ASL-like specification language.

The *(standard) theory* of a class $\mathbf{C} \subseteq Alg(\Sigma)$ of Σ-algebras, denoted by $Th(\mathbf{C})$, is defined by $Th(\mathbf{C}) \stackrel{\text{def}}{=} \{\Sigma\text{-formula } \phi \mid A \models \phi \text{ for all } A \in \mathbf{C}\}$. In the following $\mathbf{C} \models \phi$ is an equivalent notation for $\phi \in Th(\mathbf{C})$ and similarly $SP \models \phi$ is an equivalent notation for $\phi \in Th(Mod(SP))$. Note that we will always consider theories including infinitary Σ-formulas. However it is obvious that all our results remain valid if we restrict to first-order theories, i.e. theories consisting only of finitary (first-order) Σ-formulas.

In practice it is often useful to consider, instead of arbitrary Σ-algebras, algebras that are finitely generated by a distinguished subset of the function symbols, called constructors. In these algebras all elements can be denoted by a constructor term (which is built only by constructor symbols and by variables of those sorts for which no constructor is defined). More precisely, such generation principles can be formalized using reachability constraints as follows:

1. A *reachability constraint* over a signature $\Sigma = (S, F)$ is a pair $\mathcal{R} = (S_{\mathcal{R}}, F_{\mathcal{R}})$ such that $S_{\mathcal{R}} \subseteq S$, $F_{\mathcal{R}} \subseteq F$ and for any $f \in F_{\mathcal{R}}$ with arity $s_1 \dots s_n \to s$ the sort s belongs to $S_{\mathcal{R}}$. A sort $s \in S_{\mathcal{R}}$ is called *constrained sort* and a function symbol $f \in F_{\mathcal{R}}$ is called *constructor symbol* (or briefly *constructor*). We assume also that for each constrained sort $s \in S_{\mathcal{R}}$ there exists at least one constructor in $F_{\mathcal{R}}$ with s as codomain. (This ensures that Σ is sensible w.r.t. $S_{\mathcal{R}}$.)

2. A *constructor term* is a term $t \in T_{\Sigma'}(X')_s$ of sort $s \in S_{\mathcal{R}}$, where $\Sigma' = (S, F_{\mathcal{R}})$, $X' = (X'_s)_{s \in S}$ with $X'_s = X_s$ if $s \in S \setminus S_{\mathcal{R}}$ and $X'_s = \emptyset$ if $s \in S_{\mathcal{R}}$. The set of constructor terms is denoted by $T_{\mathcal{R}}$.

3. A Σ-algebra A *satisfies a reachability constraint* $\mathcal{R} = (S_{\mathcal{R}}, F_{\mathcal{R}})$, denoted by $A \models \mathcal{R}$, if for any $s \in S_{\mathcal{R}}$ and any $a \in A_s$, there exists a constructor term $t \in (T_{\mathcal{R}})_s$ and a valuation $\alpha : X' \to A$ such that $I_\alpha(t) = a$. (Note that this definition is independent of X because X_s is countably infinite for all $s \in S$.)

It is important to note that reachability constraints can be expressed by infinitary sentences:

Fact 1. *Let A be a Σ-algebra and $\mathcal{R} = (S_\mathcal{R}, F_\mathcal{R})$ be a reachability constraint over Σ. Then $A \models \mathcal{R}$ if and only if $A \models \bigwedge_{s \in S_\mathcal{R}} \mathrm{GEN}_s^\mathcal{R}$, where $\mathrm{GEN}_s^\mathcal{R}$ is the infinitary Σ-sentence defined by $\mathrm{GEN}_s^\mathcal{R} \stackrel{\mathrm{def}}{=} \forall x{:}s.\, \bigvee_{t \in (T_\mathcal{R})_s} \exists \mathrm{Var}(t).\, x = t$.*

According to this fact, specifications with finitary axioms and reachability constraints can be defined as a particular kind of basic specifications with infinitary axioms as follows. Let Σ be a signature, $\mathcal{R} = (S_\mathcal{R}, F_\mathcal{R})$ be a reachability constraint over Σ and $\mathcal{A}x$ be a set of finitary Σ-sentences. Then the triple $SP = \langle \Sigma, \mathcal{R}, \mathcal{A}x \rangle$ is, by definition, the basic specification $\langle \Sigma, \mathcal{A}x \cup \{\mathrm{GEN}_s^\mathcal{R} \mid s \in S_\mathcal{R}\} \rangle$. In the following a specification will be called *smooth* if it is built using the + specification-building primitive and specifications with finitary axioms and reachability constraints. To put emphasis on smooth specifications is justified by the fact that for such specifications it is easy to obtain sound proof rules by combining a sound (and complete) proof system for many-sorted first order logic with equality and structural induction w.r.t. the defined constructors.

Example 1. Let us consider the following `CONTAINER` specification.

```
spec: CONTAINER
 use: ELEM, NAT, BOOL
 sort: Container
 generated by:
    ∅ : → Container
    insert : Elem Container → Container
 operations:
    _ ∪ _ : Container Container → Container
    remove : Elem Container → Container
    _ ∈ _ : Elem Container → Bool
    card : Container → Nat
    subset : Container Container → Bool
 axioms:
    ∀S,S':Container, e,e':Elem.
    ∅ ∪ S = S
    insert(e,S) ∪ S' = insert(e,S ∪ S')
    remove(e,∅) = ∅
    remove(e,insert(e,S)) = remove(e,S)
    e ≠ e' ⇒ remove(e,insert(e',S)) = insert(e',remove(e,S))
    e ∈ ∅ = false
    [e ∈ insert(e',S) = true] ⇔ [(e = e') ∨ (e ∈ S = true)]
    card(∅) = 0
    [e ∈ S = true] ⇒ card(insert(e,S)) = card(S)
    [e ∈ S = false] ⇒ card(insert(e,S)) = succ(card(S))
    [subset(S,S') = true] ⇔ [∀e:Elem. (e ∈ S = true) ⇒ (e ∈ S' = true)]
end CONTAINER.
```

We do not detail the subspecifications `ELEM`, `NAT` and `BOOL` which are the usual ones. Note that the sort `Container` is constrained by the constructors $\emptyset$ and `insert`. Since the

CONTAINER specification is rather loose, its model class contains, among other algebras, the algebra of finite sets of elements, the algebra of finite multisets of elements, as well as the algebra of finite sequences of elements. It is quite easy to show (by structural induction w.r.t. the constructors $\emptyset$ and insert) that CONTAINER $\models$ S $\cup$ $\emptyset$ = S,[5] or that CONTAINER $\models$ e $\in$ S $\cup$ S' = (e $\in$ S) | (e $\in$ S'), but it is important to note that CONTAINER $\not\models$ insert(x,insert(x,S)) = insert(x,S) and that CONTAINER $\not\models$ insert(x,insert(y,S)) = insert(y,insert(x,S)). As a consequence, the CONTAINER specification cannot be considered as a correct implementation of the usual specification of sets if we require that an implementation satisfies (in the standard sense) all axioms of the given specification. $\diamondsuit$

3 Behavioural and Abstractor Specifications

Behavioural specifications are a generalization of standard specifications which allow to describe the behaviour of data structures and programs with respect to a given (partial) congruence relation. The essential difference to the standard case is that instead of the set-theoretic equality, the given congruence relation is used for the interpretation of the equality predicate symbol. Another approach which also allows to relax the standard semantics of algebraic specifications are abstractor specifications (cf. [19]). In this case an equivalence relation between algebras is used for abstracting from the (standard) model class of a specification. In this section we summarize the basic definitions and results of [6] where the relationships between behavioural and abstractor specifications are studied.

In the sequel we always assume given a family $\approx = (\approx_A)_{A \in Alg(\Sigma)}$ of (possibly partial) Σ-congruences on the algebras of $Alg(\Sigma)$ and an equivalence relation $\equiv$ on $Alg(\Sigma)$ such that $\equiv$ is *factorizable* by $\approx$, i.e. for all $A, B \in Alg(\Sigma)$, $A \equiv B$ if and only if $A/\approx_A$ and $B/\approx_B$ are isomorphic.

3.1 Behavioural and Abstractor Semantics

We start by generalizing the standard satisfaction relation to a behavioural satisfaction relation w.r.t. the given family $\approx$ of partial Σ-congruences. The idea is to interpret the variables occurring in a formula not by all values of an algebra A but only by values in $\mathrm{Dom}(\approx_A)$, and to interpret the equality predicate symbol by the given partial congruence relation $\approx_A$ instead of the set-theoretic equality. A relationship between the standard satisfaction relation and the behavioural satisfaction relation is provided in Theorem 20.

Definition 2 (Behavioural satisfaction relation). Let A be a Σ-algebra. The behavioural satisfaction relation w.r.t. $\approx$, denoted by $\models_\approx$, is defined as follows:
Let $l, r \in T_\Sigma(X)_s$ be two terms of sort s, ϕ, ψ be two Σ-formulas, $\{\phi_i \mid i \in I\}$ be a countable family of Σ-formulas and $\alpha : X \to \mathrm{Dom}(\approx_A)$ be a valuation.

1. $A, \alpha \models_\approx l = r$ holds if and only if $I_\alpha(l) \approx_A I_\alpha(r)$.

[5] For sake of simplicity the variables occurring in the equations used in our examples are implicitly universally quantified.

2. $A, \alpha \models_{\approx} \neg\phi$ holds if and only if $A, \alpha \models_{\approx} \phi$ does not hold, and $A, \alpha \models_{\approx} \phi \wedge \psi$ holds if and only if both $A, \alpha \models_{\approx} \phi$ and $A, \alpha \models_{\approx} \psi$ hold.

3. $A, \alpha \models_{\approx} \forall x{:}s.\phi$ holds if and only if, for all valuations $\beta : X \to \mathrm{Dom}(\approx_A)$ with $\beta(y) = \alpha(y)$ for all $y \neq x$, $A, \beta \models_{\approx} \phi$ holds.

4. $A, \beta \models_{\approx} \bigwedge_{i \in I} \phi_i$ holds if and only if, for all $i \in I$, $A, \beta \models_{\approx} \phi_i$ holds.

5. $A \models_{\approx} \phi$ if and only if $A, \alpha \models_{\approx} \phi$, for all valuations $\alpha : X \to \mathrm{Dom}(\approx_A)$.

Hence Definition 2 is quite similar to the definition of the standard satisfaction relation, the main difference being for (1) where $I_\alpha(l) = I_\alpha(r)$ is replaced by $I_\alpha(l) \approx_A I_\alpha(r)$. Moreover, it is important to note that valuations have their range in $\mathrm{Dom}(\approx_A)$ and not in A, to take into account the fact that $\approx_A$ is a partial congruence.

Behavioural specifications can be built on top of basic specifications as follows, using the behavioural satisfaction relation for the interpretation of the axioms:

Definition 3 (Behavioural specification). Let $SP = \langle \Sigma, \mathcal{A}x \rangle$ be a basic specification. Then:

1. The expression **behaviour** SP **w.r.t.** $\approx$ is a behavioural specification, of signature Σ.

2. The model class of a behavioural specification is defined by:
   $Mod(\textbf{behaviour } SP \textbf{ w.r.t. } \approx) \stackrel{\text{def}}{=} \{A \in Alg(\Sigma) \mid A \models_{\approx} \phi \text{ for all } \phi \in \mathcal{A}x\}$.

The notion of "abstractor" was introduced in [19] for describing a specification-building operation which allows to abstract from the model class of a specification with respect to a given equivalence relation on the class of all Σ-algebras:

Definition 4 (Abstractor operator). For any class $\mathbf{C} \subseteq Alg(\Sigma)$, $Abs_{\equiv}(\mathbf{C})$ denotes the closure of $\mathbf{C}$ under $\equiv$, i.e. $Abs_{\equiv}(\mathbf{C}) \stackrel{\text{def}}{=} \{B \in Alg(\Sigma) \mid B \equiv A \text{ for some } A \in \mathbf{C}\}$.

Abstractor specifications can be built on top of arbitrary specifications using the abstractor operator as follows:

Definition 5 (Abstractor specification). Let SP be an arbitrary specification. Then:

1. The expression **abstract** SP **w.r.t.** $\equiv$ is an abstractor specification, of signature $Sig(SP)$.

2. The model class of an abstractor specification is defined by:
   $Mod(\textbf{abstract } SP \textbf{ w.r.t. } \equiv) \stackrel{\text{def}}{=} Abs_{\equiv}(Mod(SP))$.

Behavioural specifications and abstractor specifications are based on the same intention, namely to allow a more general view of the semantics of specifications. This proves to be especially useful to define implementation relations where implementations may relax (some of) the properties of the given requirement specification (cf. e.g. abstractor

implementations in [19] or behavioural implementations in [10, 5]; for a survey on implementation concepts see [16]). [6] provides an in-depth study of the relationships between both approaches, and in the following of this section we merely recall the central results that will be used in the sequel of this paper.

The factorizability of the equivalence $\equiv$ by $\approx$, which is generally assumed here, is the technical condition under which meaningful relationships can be established. Fully abstract algebras play also an essential rôle for the study of the relationships between behavioural and abstractor specifications. Following Milner's notion (cf. [14]), we define full abstractness with respect to a given family $\approx$ of Σ-congruences as follows:

Definition 6 (Fully abstract algebra).

1. A Σ-algebra A is called fully abstract with respect to $\approx$ (or briefly fully abstract) if $\approx_A$ coincides with the set-theoretic equality over the carrier sets of A. (In particular $\approx_A$ is total.)

2. For any class $\mathbf{C} \subseteq Alg(\Sigma)$ of Σ-algebras, $FA_{\approx}(\mathbf{C})$ denotes the subclass of the fully abstract algebras of $\mathbf{C}$, i.e. $FA_{\approx}(\mathbf{C}) \stackrel{\text{def}}{=} \{A \in \mathbf{C} \mid A \text{ is fully abstract}\}$.

Definition 7 (Regularity). A family $\approx = (\approx_A)_{A \in Alg(\Sigma)}$ of Σ-congruences is called regular if, for any Σ-algebra A, the quotient algebra $A/\approx_A$ is fully abstract.

We will see in the sequel that for all our examples, the considered families of partial Σ-congruences are regular. We still need two technical definitions:

Definition 8 (Behavioural quotient operator). For any class $\mathbf{C} \subseteq Alg(\Sigma)$, $\mathbf{C}/\approx$ denotes the behavioural quotient of $\mathbf{C}$, i.e. $\mathbf{C}/\approx \stackrel{\text{def}}{=} \{A/\approx_A \mid A \in \mathbf{C}\}$.

Definition 9 (Behaviour operator). For any class $\mathbf{C} \subseteq Alg(\Sigma)$, the behaviour of $\mathbf{C}$ is defined by $Beh_{\approx}(\mathbf{C}) \stackrel{\text{def}}{=} Abs_{\equiv}(FA_{\approx}(\mathbf{C}))$.

A central result of [6] is the following characterization of behavioural semantics:

Theorem 10. *If the family $\approx$ is regular, then for any basic specification SP we have:* $Mod(\textbf{behaviour } SP \textbf{ w.r.t. } \approx) = Beh_{\approx}(Mod(SP))$. □

Theorem 10 provides the necessary basis for a detailed comparison between abstractor specifications and behavioural specifications (cf. [6]). Moreover, this result suggests also how behavioural specifications can be built on top of arbitrary specifications:

Definition 11 (Behavioural specification – General case). Let SP be an arbitrary specification. Then:

1. The expression **behaviour** SP **w.r.t.** $\approx$ is a behavioural specification, of signature $Sig(SP)$.

2. The model class of a behavioural specification is defined by:
$Mod(\textbf{behaviour } SP \textbf{ w.r.t. } \approx) \stackrel{\text{def}}{=} Beh_{\approx}(Mod(SP))$.

Theorem 10 ensures that Definitions 3 and 11 are consistent with each other (in the case of a basic specification SP) provided that the considered family $\approx$ is regular. In the sequel we will therefore use the more general Definition 11. According to Theorem 10, it is obvious that in general the model class of **behaviour** SP **w.r.t.** $\approx$ is included in the model class of **abstract** SP **w.r.t.** $\equiv$. In [6] and [5] necessary and sufficient conditions for the equality of the two model classes are studied. If this is the case we will say that the specification SP is behaviourally consistent:

Definition 12 (Behavioural consistency). A specification SP is called behaviourally consistent (w.r.t. $\approx$) if $Mod($**behaviour** SP **w.r.t.** $\approx) = Mod($**abstract** SP **w.r.t.** $\equiv)$.

3.2 The Observational Case

The most important examples of partial congruences are *observational equalities* of objects. The intuition behind observational equalities is the following one: Two objects of an algebra are considered to be observationally equal if they cannot be distinguished by "experiments" with "observable" results. In order to provide the necessary formalization, we will proceed as follows. First, we will define *contexts*, which are a special kind of terms representing the "experiments". Then we will show how a very general notion of *contextual equality* of objects can be associated to any choice of a set of contexts. In a last step we define the observational equality of objects we are interested in as a special case of contextual equalities.

Definition 13 (Context). Let $\Sigma = (S, F)$ be a signature and let $X = (X_s)_{s \in S}$ be the generally assumed family of countably infinite sets of variables of sort s. Let $Z = (\{z_s\})_{s \in S}$ be a disjoint S-sorted family of singleton sets.

1. A Σ-context is a Σ-term $C \in T_\Sigma(X \cup Z)$ which contains, besides variables in X, one or many occurrences of exactly one variable $z_s \in Z$, called the *context variable* of C.

2. By exception, $\mathrm{Var}(C)$ will denote the set of variables occurring in C apart from the context variable of C.

3. $C[t]$ denotes the term obtained by substituting the term $t \in T_\Sigma(X)_s$ for the context variable z_s (of sort s) of C.

4. Given an arbitrary set of Σ-contexts $\mathcal{C}$, we denote by $\mathcal{C}(s)$ the (possibly empty) subset of the contexts of $\mathcal{C}$ with context variable of sort s.

$T_\Sigma(X \cup Z)$ represents the set of all possible experiments (with arbitrary input variables X). In general we may not want to allow experiments on any values in an algebra A, but only on those values that can be denoted by a term t, where some restrictions apply to the variables that may occur in t. More precisely, we will choose a subset of *input sorts* $In \subseteq S$, and consider the smallest subalgebra $A[X_{In}]$ of A generated by Σ and X_{In}, where X_{In} is the S-sorted family of variables defined by $(X_{In})_s = \emptyset$ if $s \notin In$ and $(X_{In})_s = X_s$ if $s \in In$ (where $X = (X_s)_{s \in S}$ is the generally assumed family of countably infinite sets of variables of sort s). This leads to the following definition of the (partial) equality relation associated to a choice of input sorts and a set of Σ-contexts:

Definition 14 (Partial contextual equality). Let $\Sigma = (S, F)$ be a signature, $In \subseteq S$ be a set of input sorts, $\mathcal{C}$ be an arbitrary set of Σ-contexts and A be a Σ-algebra. The partial contextual equality on A induced by $\mathcal{C}$ and In, denoted by $\approx_{\mathcal{C},In,A}$, is defined as follows:
Let $A[X_{In}]$ be the smallest subalgebra of A generated by Σ and X_{In}. The carrier sets of $A[X_{In}]$ are defined by $A[X_{In}]_s = \{a \in A_s \mid$ there exists a term $t \in T_\Sigma(X_{In})_s$ and a valuation $\alpha : X_{In} \to A$ such that $I_\alpha(t) = a\}$.[6]
Two elements $a, b \in A_s$ of sort s are contextually equal (w.r.t. $\mathcal{C}$ and In), denoted by $a \approx_{\mathcal{C},In,A} b$, if and only if both a and b belong to $A[X_{In}]_s$ and, for all contexts $C \in \mathcal{C}$ with context variable z_s of sort s, for all valuations $\alpha : X \to A[X_{In}]$, we have $I_{\alpha_a}(C) = I_{\alpha_b}(C)$, where $\alpha_a, \alpha_b : X \cup \{z_s\} \to A[X_{In}]$ are the unique extensions of α defined by $\alpha_a(z_s) = a$ and $\alpha_b(z_s) = b$.
Note that, if there is no context $C \in \mathcal{C}$ with context variable of sort s, then we have $a \approx_{\mathcal{C},In,A} b$, for all $a, b \in A[X_{In}]_s$ of sort s.

The intuition behind this definition is that two elements a and b are contextually equal w.r.t. a given set $\mathcal{C}$ of Σ-contexts if they belong to the chosen subalgebra $A[X_{In}]$ and if they cannot be distinguished by at least one of the computations represented by the contexts of $\mathcal{C}$. Note that $\approx_{\mathcal{C},In,A}$ is a family of partial equivalence relations (one for each sort $s \in S$). However, $\approx_{\mathcal{C},In,A}$ is not necessarily a partial congruence relation, i.e. $\approx_{\mathcal{C},In,A}$ is not necessarily compatible with the signature Σ.

Even if $\approx_{\mathcal{C},In,A}$ is not a partial Σ-congruence, we can consider its "definition domain", denoted by $\mathrm{Dom}(\approx_{\mathcal{C},In,A})$, and defined by $\{a \in A \mid a \approx_{\mathcal{C},In,A} a\}$. Obviously we have $\mathrm{Dom}(\approx_{\mathcal{C},In,A}) = A[X_{In}]$.

We are now ready to define (partial) observational equalities. For this, we assume given a signature $\Sigma = (S, F)$ and a distinguished set $Obs \subseteq S$ of *observable* sorts (which denote the carrier sets of observable values). Moreover, we assume given a set $In \subseteq S$ of *input* sorts such that Σ is sensible w.r.t. In, and we consider the associated family of set of variables X_{In}. Then two objects of an algebra are considered to be observationally equal if they cannot be distinguished by "experiments" with inputs chosen accordingly to In and with observable results:

Definition 15 (Observable context and observational equality).

1. The set of all observable contexts, denoted by $\mathcal{C}_\Sigma^{Obs}$, is defined as being the set of all Σ-contexts of observable sort $s \in Obs$.

2. Let A be a Σ-algebra. The partial contextual equality on A induced by $\mathcal{C}_\Sigma^{Obs}$ and In (cf. Definition 14) is called the *(partial) observational equality* on A induced by Obs and In and is denoted by $\approx_{Obs,In,A}$.

Remark. In the literature (cf. e.g. [6]), the partial observational equality induced by Obs and In is defined in a slightly different way. Instead of considering the observable contexts $\mathcal{C}_\Sigma^{Obs}$ built from the signature Σ and arbitrary variables X, one restricts to observable

[6] Thereby we assume that Σ is sensible w.r.t. In (cf. Section 2).

contexts built from the signature Σ and variables of an input sort only (i.e. $T_\Sigma(X_{In} \cup Z)$ is used instead of $T_\Sigma(X \cup Z)$ in Definition 13). Then two elements $a, b \in A_s$ of sort s are observationally equal if and only if both a and b belong to $A[X_{In}]_s$ and, for all observable contexts $C \in T_\Sigma(X_{In} \cup Z)$ with context variable z_s of sort s, for all valuations $\alpha : X_{In} \to A$, we have $I_{\alpha_a}(C) = I_{\alpha_b}(C)$, where $\alpha_a, \alpha_b : X_{In} \cup \{z_s\} \to A$ are the unique extensions of α defined by $\alpha_a(z_s) = a$ and $\alpha_b(z_s) = b$. It is fairly obvious that this leads to a definition equivalent to our Definition 15. However Definition 15 will prove to be more convenient in the sequel.

Lemma 16. *The observational equality $\approx_{Obs,In,A}$ on A is a partial Σ-congruence (and $\mathrm{Dom}(\approx_{Obs,In,A}) = A[X_{In}]$).* □

The family $(\approx_{Obs,In,A})_{A \in Alg(\Sigma)}$ of partial observational equalities (which in particular are partial Σ-congruences) will be denoted by $\approx_{Obs,In}$. Most examples studied in the literature are captured by our definition:

- If we choose $In = S$, i.e. the elements of all carrier sets can be used as input for observable computations, then $A[X_{In}] = A$ and $\approx_{Obs,S,A}$ is a total congruence on A which corresponds to the behavioural equality used e.g. in [17] and in [3].
- If we choose $In = Obs$, i.e. only values generated from observable ones can be used as input for observable computations, then $\approx_{Obs,Obs,A}$ is a partial congruence on A and the resulting behavioural satisfaction relation corresponds to the one used in [15] for the behavioural satisfaction of equations. The advantage here is that non-observable junk (i.e. values which are not reachable from the observable ones) will not be considered for the behavioural satisfaction of formulas and hence cannot cause problems, for instance, with respect to the correctness of implementations (cf. e.g. [16]).

We will show in Section 5.2 that the families of observational equalities $\approx_{Obs,In}$ are always regular.

Lemma 17. *Let us consider the total observational equality $\approx_{Obs,S}$ induced by a set Obs of observable sorts. Let A be a Σ-algebra and $\forall \mathrm{Var}(l, r).\, l = r$ be a universally quantified equation. Let s be the common sort of l and r.*
If s is an observable sort, then $A \models_{\approx_{Obs,S}} \forall \mathrm{Var}(l, r).\, l = r$ if and only if $A \models \forall \mathrm{Var}(l, r).\, l = r$.
If s is a non observable sort, then $A \models_{\approx_{Obs,S}} \forall \mathrm{Var}(l, r).\, l = r$ if and only if, for all observable contexts $C \in \mathcal{C}_\Sigma^{Obs}(s)$, $A \models \forall \mathrm{Var}(C) \cup \mathrm{Var}(l, r).\, C[l] = C[r]$. □

Remark. Lemma 17 is often used in the literature to define directly (i.e. without introducing explicitly the total observational equality) the behavioural satisfaction of equations. However the explicit definition we have chosen (cf. Definition 2) is necessary to define the behavioural satisfaction of arbitrary Σ-formulas w.r.t. an arbitrary behavioural equality (this idea is even extended to higher-order logic in [12]). On the other hand, this lemma suggests that, in the total observational framework, to prove the behavioural satisfaction of an equation $l = r$ (between non observable terms), it is equivalent to prove the standard satisfaction of the infinite set of equations $C[l] = C[r]$, for all $C \in \mathcal{C}_\Sigma^{Obs}(s)$. *Context*

Induction (a specialized version of structural induction) was introduced in [10] as a means to prove such infinite sets of equations and has been implemented in the ISAR system (cf. [1]). Unfortunately proofs by context induction are quite complicated and difficult to handle in practice, especially in a first-order framework.

Example 2. Let us consider again our `CONTAINER` specification and assume that the observable sorts are `Elem`, `Nat` and `Bool` and that all sorts are input sorts. Let $\approx_{CONT}$ denote the corresponding total observational equality. Two objects of sort `Container` will be considered as observationally equal if they cannot be distinguished by observable contexts. Here, all observable contexts (with context variable of sort `Container`) must contain either $\in$, `subset` or `card`. If we consider the algebra of finite sequences of elements, it is intuitively clear that two distinct sequences will be observationally equal if they contain the same elements (not necessarily with the same number of occurrences or in the same order), because these sequences cannot be distinguished by any observable context. For the same reasons, it is intuitively clear that the two characteristic equations of sets, `insert(x,insert(x,S)) = insert(x,S)` and `insert(x,insert(y,S)) = insert(y,insert(x,S))`, are behaviourally satisfied by all models of the `CONTAINER` specification. Indeed no observable experiment can distinguish the left and right-hand sides of these equations. The aim of this paper is to provide a general proof technique to formally establish that this intuition is right.
Note that the algebra of finite sequences of elements is not fully abstract (since two distinct sequences may be observationally equal), while the algebra of finite sets of elements is fully abstract. ♢

We can also define *observational abstractions* $\equiv_{Obs,In}$ associated to the input sorts In and the observable sorts Obs as follows:

Definition 18 (Observational abstraction). Two Σ-algebras A and B are called observationally equivalent w.r.t. Obs and In, denoted by $A \equiv_{Obs,In} B$, if there exists an S-sorted family of variables Y_{In} with $(Y_{In})_s = \emptyset$ for all $s \in S \setminus In$ and two valuations $\alpha_A : Y_{In} \to A$ and $\alpha_B : Y_{In} \to B$ with surjective mappings $\alpha_{A,s} : (Y_{In})_s \to A_s$ and $\alpha_{B,s} : (Y_{In})_s \to B_s$ for all $s \in In$ such that for all terms $l, r \in T_\Sigma(Y_{In})_s$ of observable sort $s \in Obs$ the following holds: $I_{\alpha_A}(l) = I_{\alpha_A}(r)$ if and only if $I_{\alpha_B}(l) = I_{\alpha_B}(r)$.

We refer the reader to [6] where a more detailed discussion of observational abstractions is provided as well as a proof of the fact that the observational abstraction $\equiv_{Obs,In}$ is factorizable by the observational equivalence $\approx_{Obs,In}$.

3.3 Behavioural Theories

According to the generalization of the standard satisfaction relation to the satisfaction relation with respect to a family $\approx$ of Σ-congruences we consider the theory with respect to $\approx$ of a given class $\mathbf{C}$ of Σ-algebras.

Definition 19 (Behavioural theory). Let $\mathbf{C} \subseteq Alg(\Sigma)$ be a class of Σ-algebras. The behavioural theory of $\mathbf{C}$, denoted by $Th_\approx(\mathbf{C})$, is defined by:
$Th_\approx(\mathbf{C}) \stackrel{\text{def}}{=} \{\Sigma\text{-formula } \phi \mid A \models_\approx \phi \text{ for all } A \in \mathbf{C}\}$.

In the following $\mathbf{C} \models_{\approx} \phi$ is an equivalent notation for $\phi \in Th_{\approx}(\mathbf{C})$ and similarly $SP \models_{\approx} \phi$ is an equivalent notation for $\phi \in Th_{\approx}(Mod(SP))$.

Another central result of [6] is the following theorem which leads to a characterization of behavioural theories:

Theorem 20. *For all Σ-algebras A, B and Σ-formulas ϕ the following holds:*

1. *$A \models_{\approx} \phi$ if and only if $A/{\approx_A} \models \phi$.*
2. *If $A \equiv B$ then $A \models_{\approx} \phi$ if and only if $B \models_{\approx} \phi$.*
3. *If A is fully abstract then $A \models_{\approx} \phi$ if and only if $A \models \phi$.* □

Corollary 21. *For any class $\mathbf{C}$ of Σ-algebras, we have:*

1. $Th_{\approx}(\mathbf{C}) = Th(\mathbf{C}/{\approx})$.
2. $Th_{\approx}(Abs_{\equiv}(\mathbf{C})) = Th_{\approx}(\mathbf{C})$.
3. $Th_{\approx}(FA_{\approx}(\mathbf{C})) = Th(FA_{\approx}(\mathbf{C}))$.
4. $Th_{\approx}(Beh_{\approx}(\mathbf{C})) = Th_{\approx}(Abs_{\equiv}(FA_{\approx}(\mathbf{C}))) = Th(FA_{\approx}(\mathbf{C}))$. □

In practice it is usually difficult to prove behavioural theorems due to the generalized satisfaction relation. Although Corollary 21(1) shows that in principle behavioural theories can be reduced to standard theories this result is of little practical interest because even if the class $\mathbf{C}$ is axiomatizable we have (in general) no straightforward proof system for the standard theory of the class $\mathbf{C}/{\approx}$ (since the formation of quotients does not preserve the validity of arbitrary Σ-formulas). Therefore we are interested in finding other characterizations of behavioural theories which allow us to prove behavioural theorems using standard proof techniques. For this purpose our general strategy is first to reduce the behavioural theory $Th_{\approx}(\mathbf{C})$ of some class $\mathbf{C}$ of Σ-algebras to (a subset of) the standard theory $Th(\mathbf{D})$ of some other class $\mathbf{D}$ of algebras and then to look for an appropriate axiomatization (or a proof system) for $\mathbf{D}$ (provided that an axiomatization or a proof system for $\mathbf{C}$ is given). According to the previous results we know that this strategy works well in the following case: If $\mathbf{C}$ is a class of fully abstract algebras then we know by Corollary 21(3) that $Th_{\approx}(\mathbf{C}) = Th(\mathbf{C})$ i.e. in this case the behavioural theory of $\mathbf{C}$ can be reduced to the standard theory of $\mathbf{C}$. In the next section we derive a similar result for arbitrary classes $\mathbf{C}$ of Σ-algebras.

4 The Lift Operator

In a first step we introduce a "lift" operator which provides an explicit denotation for the given family ${\approx} = ({\approx_A})_{A \in Alg(\Sigma)}$ of partial Σ-congruences. For this purpose we use predicate symbols to denote the equivalence relations.[7]

[7] We assume the reader to be familiar with the usual notions of predicate symbols and their interpretations.

Definition 22 (Lift operator). Given a signature Σ, a Σ-algebra A, a class $\mathbf{C}$ of Σ-algebras and a Σ-formula ϕ we define their lifted versions as follows:

1. $\mathcal{L}(\Sigma) \stackrel{\text{def}}{=} \Sigma \cup \{\sim_s\colon s\ s\}_{s\in S}$, i.e. $\mathcal{L}(\Sigma)$ is the signature Σ enriched by, for each sort s in S, a binary predicate symbol $\sim_s\colon s\ s$ (to denote the behavioural equality). We will adopt an infix notation for the binary predicate symbols $\sim_s$, i.e. we write $l \sim_s r$ instead of $\sim_s(l, r)$.

2. $\mathcal{L}(A)$ is the unique $\mathcal{L}(\Sigma)$-algebra extension of A defined by:

   (a) $\mathcal{L}(A)\,|_\Sigma \stackrel{\text{def}}{=} A$

   (b) For any s in S, $\sim_s^{\mathcal{L}(A)} \stackrel{\text{def}}{=} \approx_{A,s}$,
   i.e. for any a, b in $\mathcal{L}(A)_s$ $(= A_s)$, $a \sim_s^{\mathcal{L}(A)} b$ if and only if $a \approx_{A,s} b$.

3. $\mathcal{L}(\mathbf{C}) \stackrel{\text{def}}{=} \{\mathcal{L}(A) \mid A \in \mathbf{C}\}$.

4. $\mathcal{L}(\phi) \stackrel{\text{def}}{=} \left[\left(\bigwedge_{y:s\in \text{FreeVar}(\phi)} D_s(y)\right) \Rightarrow \phi^*\right]$, where $D_s(y)$ is an abbreviation for $y \sim_s y$ and ϕ^* is defined by induction on the structure of ϕ as follows:[8]

   (a) If ϕ is an equation $l = r$ between two terms of sort s, then ϕ^* is $l \sim_s r$,

   (b) $(\neg\phi)^* = \neg(\phi^*)$, $(\phi_1 \wedge \phi_2)^* = (\phi_1^*) \wedge (\phi_2^*)$, $(\phi_1 \vee \phi_2)^* = (\phi_1^*) \vee (\phi_2^*)$, and similarly for infinite conjunctions and disjunctions,

   (c) $(\forall x{:}s.\phi)^* = \forall x{:}s.\,[D_s(x) \Rightarrow \phi^*]$.

   Note that if ϕ is a closed Σ-formula then $\mathcal{L}(\phi)$ coincides with ϕ^*.

Definition 23. Given a class $\mathbf{D}$ of $\mathcal{L}(\Sigma)$-algebras, we define:
$Th^{\mathcal{L}}(\mathbf{D}) \stackrel{\text{def}}{=} \{\Sigma\text{-formula } \phi \mid \mathcal{L}(\phi) \in Th(\mathbf{D})\}$.

The following theorem shows that for any class $\mathbf{C}$ of Σ-algebras the behavioural theory of $\mathbf{C}$ consists of all Σ-formulas ϕ whose lifted version $\mathcal{L}(\phi)$ belongs to the standard theory of $\mathcal{L}(\mathbf{C})$.

Theorem 24. *For any Σ-algebra A and Σ-formula ϕ, $A \models_\approx \phi$ if and only if $\mathcal{L}(A) \models \mathcal{L}(\phi)$. Hence, for any class $\mathbf{C}$ of Σ-algebras, $\mathbf{C} \models_\approx \phi$ if and only if $\mathcal{L}(\mathbf{C}) \models \mathcal{L}(\phi)$, i.e. $Th_\approx(\mathbf{C}) = Th^{\mathcal{L}}(\mathcal{L}(\mathbf{C}))$.*

The proof of Theorem 24 relies on the following two lemmas:

Lemma 25. *Let A be a Σ-algebra and ϕ be a Σ-formula.*
For all valuations $\alpha : X \to \text{Dom}(\approx_A)$, the following conditions are equivalent:

1. $A, \alpha \models_\approx \phi$
2. $\mathcal{L}(A), \alpha \models \phi^*$
3. $\mathcal{L}(A), \alpha \models \mathcal{L}(\phi)$

[8] Similar constructions, called relativizations, were used in [23] and recently in [12].

Proof. Since A and $\mathcal{L}(A)$ have the same carrier sets, a valuation from X to $\mathrm{Dom}(\approx_A)$ is a special case of a valuation from X to $\mathcal{L}(A)$. Moreover, for any valuation $\alpha : X \to \mathrm{Dom}(\approx_A)$ and any variable y of sort s, we have $\mathcal{L}(A), \alpha \models D_s(y)$. It is therefore trivial to see that (2) $\Leftrightarrow$ (3). We prove (1) $\Leftrightarrow$ (2) by induction on the form of ϕ:

Case $l = r$**:** Let $\alpha : X \to \mathrm{Dom}(\approx_A)$ be an arbitrary valuation. $A, \alpha \models_{\approx} l = r$ iff $I_\alpha(l) \approx_A I_\alpha(r)$ iff $I_\alpha(l) \sim_s^{\mathcal{L}(A)} I_{\alpha(r)}$ (where s is the sort of l and r) iff $\mathcal{L}(A), \alpha \models l \sim_s r$ iff $\mathcal{L}(A), \alpha \models (l = r)^*$.

Case $\neg\phi$**,** $\phi \wedge \psi$**,** $\ldots$**:** For formulas of the form $\neg\phi$, $\phi \wedge \psi$ and $\phi \vee \psi$ the result follows directly from the induction hypothesis. This is also true for infinite conjunctions and disjunctions.

Case $\forall x{:}s.\phi$**:** Let $\alpha : X \to \mathrm{Dom}(\approx_A)$ be an arbitrary valuation. $A, \alpha \models_{\approx} \forall x{:}s.\phi$ iff, for all valuations $\beta : X \to \mathrm{Dom}(\approx_A)$ with $\beta(y) = \alpha(y)$ if $y \neq x$, $A, \beta \models_{\approx} \phi$ iff (by induction hypothesis) for all valuations $\beta : X \to \mathrm{Dom}(\approx_A)$ with $\beta(y) = \alpha(y)$ if $y \neq x$, $\mathcal{L}(A), \beta \models \phi^*$ iff (justification given below) for all valuations $\gamma : X \to \mathcal{L}(A)$ with $\gamma(y) = \alpha(y)$ if $y \neq x$, $\mathcal{L}(A), \gamma \models D_s(x) \Rightarrow \phi^*$ iff $\mathcal{L}(A), \alpha \models \forall x{:}s.[D_s(x) \Rightarrow \phi^*]$ iff $\mathcal{L}(A), \alpha \models (\forall x{:}s.\phi)^*$.

We still have to justify the central step. Hence we must prove that the two following conditions are equivalent:

(a) For all valuations $\beta : X \to \mathrm{Dom}(\approx_A)$ with $\beta(y) = \alpha(y)$ if $y \neq x$, $\mathcal{L}(A), \beta \models \phi^*$

(b) For all valuations $\gamma : X \to \mathcal{L}(A)$ with $\gamma(y) = \alpha(y)$ if $y \neq x$, $\mathcal{L}(A), \gamma \models D_s(x) \Rightarrow \phi^*$

The direction (b) $\Rightarrow$ (a) is obvious, since $\mathcal{L}(A), \beta \models D_s(x)$. To prove the other direction, assume (a) holds and let $\gamma : X \to \mathcal{L}(A)$ be an arbitrary valuation with $\gamma(y) = \alpha(y)$ if $y \neq x$.

Case $\gamma(x) \in \mathrm{Dom}(\approx_A)$**:** Then in (a) we can take $\beta = \gamma$ and we are done.

Case $\gamma(x) \notin \mathrm{Dom}(\approx_A)$**:** Then $\mathcal{L}(A), \gamma \not\models D_s(x)$, hence $\mathcal{L}(A), \gamma \models D_s(x) \Rightarrow \phi^*$. □

Lemma 26. *Let A be a Σ-algebra and ϕ be a Σ-formula. The following conditions are equivalent:*

1. *For all valuations $\alpha : X \to \mathrm{Dom}(\approx_A)$: $\mathcal{L}(A), \alpha \models \mathcal{L}(\phi)$*
2. *For all valuations $\beta : X \to \mathcal{L}(A)$: $\mathcal{L}(A), \beta \models \mathcal{L}(\phi)$*

Proof. The direction (2) $\Rightarrow$ (1) is obvious. To prove (1) $\Rightarrow$ (2), assume (1) holds and let $\beta : X \to \mathcal{L}(A)$ be an arbitrary valuation.

Case $\beta(y) \in \mathrm{Dom}(\approx_A)$ **for all** $y \in \mathrm{Freevar}(\phi)$**:** Then there exists a valuation $\alpha : X \to \mathrm{Dom}(\approx_A)$ with $\alpha(y) = \beta(y)$ for all $y \in \mathrm{FreeVar}(\phi)$. By assumption $\mathcal{L}(A), \alpha \models \mathcal{L}(\phi)$. But then we conclude that $\mathcal{L}(A), \beta \models \mathcal{L}(\phi)$ since α and β coincide on all free variables of ϕ.

Case $\beta(y) \notin \mathrm{Dom}(\approx_A)$ **for some** $y \in \mathrm{Freevar}(\phi)$**:** Then $\mathcal{L}(A), \beta \not\models \bigwedge_{y:s \in \mathrm{FreeVar}(\phi)} D_s(y)$ and therefore $\mathcal{L}(A), \beta \models \mathcal{L}(\phi)$ (by definition of $\mathcal{L}(\phi)$). □

Proof of Theorem 24. Let A be a Σ-algebra and ϕ a Σ-formula. $A \models_{\approx} \phi$ iff (by definition) for all valuations $\alpha : X \to \mathrm{Dom}(\approx_A)$, $A, \alpha \models_{\approx} \phi$ iff (by Lemma 25) for all valuations $\alpha : X \to \mathrm{Dom}(\approx_A)$, $\mathcal{L}(A), \alpha \models \mathcal{L}(\phi)$ iff (by Lemma 26) for all valuations $\beta : X \to \mathcal{L}(A)$, $\mathcal{L}(A), \beta \models \mathcal{L}(\phi)$ iff (by definition) $\mathcal{L}(A) \models \mathcal{L}(\phi)$. □

Remark. Let $\mathbf{C}$ be a class of fully abstract Σ-algebras. By Theorem 24 we have $Th_{\approx}(\mathbf{C}) = Th^{\mathcal{L}}(\mathcal{L}(\mathbf{C}))$. Moreover, $Th^{\mathcal{L}}(\mathcal{L}(\mathbf{C})) = Th(\mathbf{C})$ because for any fully abstract algebra A, $A \models \phi$ if and only if $\mathcal{L}(A) \models \mathcal{L}(\phi)$. Hence we obtain in this particular case again our previous result $Th_{\approx}(\mathbf{C}) = Th(\mathbf{C})$.

Theorem 24 provides a means for reducing the set of behavioural theorems over an arbitrary class $\mathbf{C}$ of Σ-algebras to a set of standard theorems over a corresponding class $\mathcal{L}(\mathbf{C})$ of $\mathcal{L}(\Sigma)$-algebras. However, the usefulness of this reduction still depends on the possibility to perform proofs of standard theorems over $\mathcal{L}(\mathbf{C})$. Since $\mathcal{L}(\mathbf{C})$ is constructed on top of $\mathbf{C}$ by introducing a denotation for the behavioural equality we claim that for proving standard theorems over $\mathcal{L}(\mathbf{C})$ we need, on one hand, a proof system for proving standard theorems over $\mathbf{C}$ (in the best case $\mathbf{C}$ is axiomatizable) and, on the other hand, we need an axiomatization of the behavioural equality which will be considered in the next sections.

Example 3. We can apply Theorem 24 to various classes of algebras of special interest:

1. Let $\mathbf{C} = \mathit{Mod}(SP)$ be the model class of a basic specification $SP = \langle \Sigma, \mathcal{A}x \rangle$. Then, by Theorem 24, $Th_{\approx}(\mathit{Mod}(SP)) = Th^{\mathcal{L}}(\mathcal{L}(\mathit{Mod}(SP)))$ and we will see in the next section that $\mathcal{L}(\mathit{Mod}(SP))$ can be axiomatized with the help of an axiomatization of the behavioural equality.
   More generally, if SP is an ASL-like structured specification (cf. Section 2) we will see that $\mathcal{L}(\mathit{Mod}(SP))$ can be expressed by a specification of the ASL-like language as soon as an axiomatization of the behavioural equality is provided (cf. Theorem 36 and Theorem 38).

2. Let $\mathbf{C} = \mathit{Abs}_{\equiv}(\mathbf{C}')$ for some class $\mathbf{C}'$ of Σ-algebras. Then an immediate application of Theorem 24 would lead to $Th_{\approx}(\mathit{Abs}_{\equiv}(\mathbf{C}')) = Th^{\mathcal{L}}(\mathcal{L}(\mathit{Abs}_{\equiv}(\mathbf{C}')))$. But even if $\mathbf{C}'$ is axiomatizable there is no simple proof system for $\mathit{Abs}_{\equiv}(\mathbf{C}')$ and hence also not for $\mathcal{L}(\mathit{Abs}_{\equiv}(\mathbf{C}'))$. However, in this case we know, by Corollary 21(2), that $Th_{\approx}(\mathit{Abs}_{\equiv}(\mathbf{C}')) = Th_{\approx}(\mathbf{C}')$ and hence we can apply Theorem 24 to $\mathbf{C}'$ instead of $\mathit{Abs}_{\equiv}(\mathbf{C}')$. Thus we obtain $Th_{\approx}(\mathit{Abs}_{\equiv}(\mathbf{C}')) = Th_{\approx}(\mathbf{C}') = Th^{\mathcal{L}}(\mathcal{L}(\mathbf{C}'))$. Hence for abstractor specifications we have:
   $Th_{\approx}(\mathit{Mod}(\textbf{abstract } SP \textbf{ w.r.t. } \equiv)) = Th_{\approx}(\mathit{Mod}(SP)) = Th^{\mathcal{L}}(\mathcal{L}(\mathit{Mod}(SP)))$, i.e. this case can be reduced to the case considered in Part 1 of the example.

3. Let $\mathbf{C} = \mathit{Beh}_{\approx}(\mathbf{C}')$ for some class $\mathbf{C}'$ of Σ-algebras. Then an application of Theorem 24 would lead to $Th_{\approx}(\mathit{Beh}_{\approx}(\mathbf{C}')) = Th^{\mathcal{L}}(\mathcal{L}(\mathit{Beh}_{\approx}(\mathbf{C}')))$ which again is not a useful reduction. However we know, by Corollary 21(4), that $Th_{\approx}(\mathit{Beh}_{\approx}(\mathbf{C}')) =$

$Th(FA_{\approx}(\mathbf{C}'))$. Hence in this case we are interested in an axiomatization of full abstractness. In particular, for behavioural specifications we have:
$Th_{\approx}(Mod(\textbf{behaviour } SP \textbf{ w.r.t. } \approx)) = Th(FA_{\approx}(Mod(SP)))$.
(Note that in this case we do not actually use Theorem 24 but rather Corollary 21(4).)

4. Let $\mathbf{C}$ be a class of Σ-algebras such that $Abs_{\equiv}(\mathbf{C}) = Beh_{\approx}(\mathbf{C})$ (see [6] for simple necessary and sufficient conditions). Then we have:
$Th_{\approx}(\mathbf{C}) = Th_{\approx}(Abs_{\equiv}(\mathbf{C})) = Th_{\approx}(Beh_{\approx}(\mathbf{C})) = Th(FA_{\approx}(\mathbf{C}))$. ♢

5 Axiomatization of the Behavioural Equality

In the previous section we have shown how to replace the behavioural theory of some given class $\mathbf{C}$ of Σ-algebras by (a subset of) the standard theory of another related class of algebras. The next step is to provide a characterization of this class of algebras in terms of an axiomatization of the behavioural equality.

Definition 27 (Axiomatization of the behavioural equality). An axiomatization of the behavioural equality $\approx$ is an S-sorted family $Beh = (Beh_s(x_s, y_s))_{s \in S}$ of (possibly infinitary) Σ-formulas, where x_s and y_s are the only free variables, of sort s, of $Beh_s(x_s, y_s)$, such that, for any Σ-algebra A, any sort s in S and any valuation $\alpha : \{x_s, y_s\} \to A$, $A, \alpha \models Beh_s(x_s, y_s)$ if and only if $\alpha(x_s) \approx_A \alpha(y_s)$.
Whenever such an axiomatization exists, we say that the behavioural equality $\approx$ is axiomatizable.

In the following, we assume given an axiomatization $(Beh_s(x_s, y_s))_{s \in S}$ of the behavioural equality $\approx$.

Remark. Let A be a Σ-algebra and $a, b \in A_s$ be two arbitrary elements of sort s. Let $\alpha : \{x_s, y_s\} \to A$ be the valuation defined by $\alpha(x_s) = a$ and $\alpha(y_s) = b$. Since A and $\mathcal{L}(A)$ have the same carrier sets, α can be considered as a valuation from $\{x_s, y_s\}$ to $\mathcal{L}(A)$ as well. Then $a \approx_A b$ iff $\mathcal{L}(A), \alpha \models x_s \sim_s y_s$ iff $A, \alpha \models Beh_s(x_s, y_s)$. Moreover, if $a, b \in \mathrm{Dom}(\approx_A)$ then α has its range in $\mathrm{Dom}(\approx_A)$ and in that case the above are further equivalent to $A, \alpha \models_{\approx} x_s = y_s$. This remark is the basis of most proofs of this section.

Example 4 (Observational equality).

1. The total observational equality $\approx_{Obs,S}$ induced by a set Obs of observable sorts (cf. Section 3.2) is axiomatized by the following infinitary formulas:
$Beh_s(x_s, y_s) \stackrel{\text{def}}{=} \bigwedge_{C \in \mathcal{C}_{\Sigma}^{Obs}(s)} \forall \mathrm{Var}(C).\, C[x_s] = C[y_s]$.
Note that if s is an observable sort then $Beh_s(x_s, y_s)$ is equivalent to $x_s = y_s$, since then the trivial context z_s belongs to $\mathcal{C}_{\Sigma}^{Obs}(s)$. Hence for observable elements the observational equality coincides with the set-theoretic equality, as required.

2. The partial observational equality $\approx_{Obs,In}$ induced by a set Obs of observable sorts and a set In of input sorts (cf. Section 3.2) is axiomatized by the following infinitary formulas:

$Beh_s(x_s, y_s) \stackrel{\text{def}}{=} \mathrm{Def}_s(x_s) \wedge \mathrm{Def}_s(y_s) \wedge$
$\bigwedge_{C \in \mathcal{C}_\Sigma^{Obs}(s)} \forall \mathrm{Var}(C).\, \mathrm{DEF}(\mathrm{Var}(C)) \Rightarrow C[x_s] = C[y_s]$,
where $\mathrm{Def}_s(x_s)$ is an abbreviation for $\bigvee_{t \in T_\Sigma(X_{In})_s} \exists \mathrm{Var}(t).\, x_s = t$
and $\mathrm{DEF}(\mathrm{Var}(C))$ stands for $\bigwedge_{v_{s'} \in \mathrm{Var}(C)} \mathrm{Def}_{s'}(v_{s'})$.
Note that if s is an input sort, then $Beh_s(x_s, y_s)$ is equivalent to:
$\bigwedge_{C \in \mathcal{C}_\Sigma^{Obs}(s)} \forall \mathrm{Var}(C).\, \mathrm{DEF}(\mathrm{Var}(C)) \Rightarrow C[x_s] = C[y_s]$.
If s is an observable sort then $Beh_s(x_s, y_s)$ is equivalent to:
$\mathrm{Def}_s(x_s) \wedge \mathrm{Def}_s(y_s) \wedge x_s = y_s$.
Moreover, $Beh_s(x_s, x_s)$ is always equivalent to $\mathrm{Def}_s(x_s)$. ♢

Before we come back to the general case of lifted algebras in Section 5.3 we will consider the usefulness of an axiomatization of the behavioural equality in the case of fully abstract algebras.

5.1 Case of Fully Abstract Algebras

The following proposition provides an obvious characterization of full abstractness in terms of an axiomatization of the behavioural equality.

Proposition 28 (Characterization of fully abstract algebras). *A Σ-algebra A is fully abstract if and only if, for all s in S, $A \models \forall x_s, y_s{:}s.\, [Beh_s(x_s, y_s) \Leftrightarrow x_s = y_s]$.*

Proof.

$\Longrightarrow$: Assume that A is fully abstract and let $\alpha : \{x_s, y_s\} \to A$ be an arbitrary valuation. $A, \alpha \models x_s = y_s$ iff $\alpha(x_s) = \alpha(y_s)$ iff (since A is fully abstract) $\alpha(x_s) \approx_A \alpha(y_s)$ iff (by Definition 27) $A, \alpha \models Beh_s(x_s, y_s)$.
Hence $A \models \forall x_s, y_s{:}s.\, [Beh_s(x_s, y_s) \Leftrightarrow x_s = y_s]$.

$\Longleftarrow$: Assume that $A \models \forall x_s, y_s{:}s.\, [Beh_s(x_s, y_s) \Leftrightarrow x_s = y_s]$. Let $a, b \in A_s$ and $\alpha : \{x_s, y_s\} \to A$ be the valuation defined by $\alpha(x_s) = a$ and $\alpha(y_s) = b$. Then $a = b$ iff $A, \alpha \models x_s = y_s$ iff (by assumption) $A, \alpha \models Beh_s(x_s, y_s)$ iff $a \approx_A b$. Hence A is fully abstract. □

From Proposition 28 we directly deduce the following result which says that for any class $\mathbf{C}$ of Σ-algebras the subclass of the fully abstract algebras of $\mathbf{C}$ can be characterized using an axiomatization of the behavioural equality. This fact will be used when considering the particular case of behavioural theories of behavioural specifications (cf. Example 5(3)).

Theorem 29. *Let FA_{Beh} be the Σ-sentence defined by:*
$\mathrm{FA}_{Beh} \stackrel{\text{def}}{=} \bigwedge_{s \in S} \forall x_s, y_s{:}s.\, [Beh_s(x_s, y_s) \Leftrightarrow x_s = y_s]$.
Then for any class $\mathbf{C}$ of Σ-algebras, $FA_{\approx}(\mathbf{C}) = \mathbf{C} \cap Mod(\langle \Sigma, \mathrm{FA}_{Beh} \rangle)$.
In particular, if we consider the model class of some ASL-like structured specification SP of signature Σ, we have $FA_{\approx}(Mod(SP)) = Mod(SP + \langle \Sigma, \mathrm{FA}_{Beh} \rangle)$. □

5.2 Invariance and Regularity

The results for fully abstract algebras are the basis for the characterization of behavioural theories of behavioural specifications since in this case we have:
$Th_{\approx}(Mod(\textbf{behaviour } SP \textbf{ w.r.t. } \approx)) = Th(FA_{\approx}(Mod(SP)))$.
However remember that the Definitions 3 and 11 of behavioural specifications coincide for basic specifications only if the given family $\approx$ is regular. The aim of this section is to point out that whenever an axiomatization of the behavioural equality is provided one can characterize the regularity of $\approx$ by a certain property (called invariance) of this axiomatization.

Definition 30 (Invariant formula). A formula ϕ is called invariant w.r.t. $\approx$ if, for any Σ-algebra A and valuation $\alpha : X \to \mathrm{Dom}(\approx_A)$, $A, \alpha \models_{\approx} \phi$ if and only if $A, \alpha \models \phi$.

Remark. If ϕ is closed then, due to Theorem 24, ϕ is invariant if and only if ϕ and $\mathcal{L}(\phi)$ are equivalent formulas w.r.t. all lifted algebras $\mathcal{L}(A)$. In particular, in the special case of partial observational equalities induced by a set Obs of observable sorts and a set In of input sorts, formulas built with the logical connectives, quantifiers and equations between observable terms (i.e. terms of an observable sort) with variables of input sorts only are invariant formulas.

The following lemmas will be used for proving the characterization of regularity given in Proposition 33.

Lemma 31. *The axiomatization of the behavioural equality $\approx$ is invariant if and only if, for any Σ-algebra A and any sort s in S, $A \models_{\approx} \forall x_s, y_s{:}s.\, [Beh_s(x_s, y_s) \Leftrightarrow x_s = y_s]$.*

Proof. Let A be a Σ-algebra and $\alpha : X \to \mathrm{Dom}(\approx_A)$ be an arbitrary valuation.

$\Longrightarrow$: $A, \alpha \models_{\approx} Beh_s(x_s, y_s)$ iff (by invariance) $A, \alpha \models Beh_s(x_s, y_s)$ iff $\alpha(x_s) \approx_A \alpha(y_s)$ iff $A, \alpha \models_{\approx} x_s = y_s$.

$\Longleftarrow$: $A, \alpha \models_{\approx} Beh_s(x_s, y_s)$ iff (by assumption) $A, \alpha \models_{\approx} x_s = y_s$ iff $A, \alpha \models Beh_s(x_s, y_s)$. Hence the axiomatization is invariant. □

Lemma 32. *The axiomatization of the behavioural equality $\approx$ is invariant if and only if, for any Σ-algebra A and any sort s in S, $A/{\approx_A} \models \forall x_s, y_s{:}s.\, [Beh_s(x_s, y_s) \Leftrightarrow x_s = y_s]$.*

Proof. Follows from Theorem 20(1) and Lemma 31. □

Proposition 33. *The behavioural equality $\approx$ is regular if and only if its axiomatization is invariant.*

Proof. By definition, the behavioural equality $\approx$ is regular if and only if, for any Σ-algebra A, $A/{\approx_A}$ is fully abstract. According to Proposition 28 and Lemma 32, this is equivalent to the invariance of the axiomatization. □

It is therefore easy to show that the observational equality $\approx_{Obs,In}$ induced by a set Obs of observable sorts and a set In of input sorts is always regular (cf. Example 4). In the following we always assume that the family $\approx$ is regular or equivalently that the axiomatization of the behavioural equality is invariant.

5.3 General Case

We will now generalize the results obtained for fully abstract algebras to the lifting operator. In a first step we obtain an appropriate characterization of lifted algebras.

Proposition 34 (Characterization of lifted algebras). *Let B be an arbitrary $\mathcal{L}(\Sigma)$-algebra. Then $B \in \mathcal{L}(Alg(\Sigma))$ (i.e. B is a lifted algebra) if and only if, for all $s \in S$, $B \models \forall x_s, y_s{:}s.\,[Beh_s(x_s, y_s) \Leftrightarrow x_s \sim_s y_s]$.*

Proof. Let A be a Σ-algebra. Since A and $\mathcal{L}(A)$ have the same carrier sets, it is not necessary to distinguish between valuations from $\{x_s, y_s\}$ to A and valuations from $\{x_s, y_s\}$ to $\mathcal{L}(A)$. In the following we will denote such valuations by $\alpha : \{x_s, y_s\} \to A, \mathcal{L}(A)$.

$\Longrightarrow$: Assume $B \in \mathcal{L}(Alg(\Sigma))$ and let $A \in Alg(\Sigma)$ such that $B = \mathcal{L}(A)$.
Let $\alpha : \{x_s, y_s\} \to A, \mathcal{L}(A)$ be an arbitrary valuation. $\mathcal{L}(A), \alpha \models Beh_s(x_s, y_s)$ iff (since $Beh_s(x_s, y_s)$ is a Σ-formula) $A, \alpha \models Beh_s(x_s, y_s)$ iff (by Definition 27) $\alpha(x_s) \approx_A \alpha(y_s)$ iff (by Definition 22) $\alpha(x_s) \sim_s^{\mathcal{L}(A)} \alpha(y_s)$ iff $\mathcal{L}(A), \alpha \models x_s \sim_s y_s$. Hence $B \models \forall x_s, y_s{:}s.\,[Beh_s(x_s, y_s) \Leftrightarrow x_s \sim_s y_s]$.

$\Longleftarrow$: Assume $B \models \forall x_s, y_s{:}s.\,[Beh_s(x_s, y_s) \Leftrightarrow x_s \sim_s y_s]$. Let $A = B|_\Sigma$. We prove that $B = \mathcal{L}(A)$. For this it is enough to show that for all $s \in S$ and for all a, b in B_s $(= A_s = \mathcal{L}(A)_s)$, $a \sim_s^B b$ iff $a \sim_s^{\mathcal{L}(A)} b$. Let $\alpha : \{x_s, y_s\} \to B, A, \mathcal{L}(A)$ be the valuation defined by $\alpha(x_s) = a$ and $\alpha(y_s) = b$. Then $a \sim_s^B b$ iff $B, \alpha \models x_s \sim_s y_s$ iff $B, \alpha \models Beh_s(x_s, y_s)$ iff $A, \alpha \models Beh_s(x_s, y_s)$ iff $a \approx_A b$ iff $a \sim_s^{\mathcal{L}(A)} b$. Hence $B = \mathcal{L}(A)$. □

This proof suggests that, symmetrically to the semantic lifting induced by a behavioural equality $\approx$, we can define an "axiomatic lifting" of algebras induced by a family of appropriate formulas. Indeed this "axiomatic lifting" can be defined for any Σ_1-algebra, using Σ_1-formulas, where $\Sigma \subseteq \Sigma_1$:[9]

Definition 35 (Axiomatic lifting of algebras). Let Σ_1 be a signature such that $\Sigma \subseteq \Sigma_1$, and let $\Sigma_1^{\mathcal{L}} \stackrel{\text{def}}{=} \mathcal{L}(\Sigma) \cup \Sigma_1$. Let $\phi = (\phi_s)_{s \in S}$ be an S-sorted family of arbitrary Σ_1-formulas, and assume that each ϕ_s has exactly two free variables, say x_s and y_s, of sort $s \in S$. Let BEH_ϕ be the $\Sigma_1^{\mathcal{L}}$-sentence defined by $\mathrm{BEH}_\phi \stackrel{\text{def}}{=} \bigwedge_{s \in S} \forall x_s, y_s{:}s.\,[\phi_s \Leftrightarrow x_s \sim_s y_s]$. Then for any Σ_1-algebra A, there exists a unique $\Sigma_1^{\mathcal{L}}$-algebra, denoted by $\mathcal{E}_\phi^{\mathcal{L}}(A)$, such that:[10]

1. $\mathcal{E}_\phi^{\mathcal{L}}(A)|_{\Sigma_1} = A$.
2. $\mathcal{E}_\phi^{\mathcal{L}}(A) \models \mathrm{BEH}_\phi$.

$\mathcal{E}_\phi^{\mathcal{L}}(A)$ is called the axiomatic lifting of A induced by the family ϕ.
For any class $\mathbf{C}$ of Σ_1-algebras, let $\mathcal{E}_\phi^{\mathcal{L}}(\mathbf{C}) \stackrel{\text{def}}{=} \{\mathcal{E}_\phi^{\mathcal{L}}(A) \mid A \in \mathbf{C}\}$.

[9] The motivations for using a larger signature Σ_1 will become clear in the next section when we consider axiomatizations with hidden parts.

[10] Since the sentence BEH_ϕ defines unambiguously the interpretation of the predicate symbol $\sim_s$, for each $s \in S$.

Remark. Note that $\mathcal{E}^{\mathcal{L}}_{\phi}(\mathbf{C}) = \{\Sigma_1^{\mathcal{L}}\text{-algebra } B \mid B \in Mod(\langle \Sigma_1^{\mathcal{L}}, \mathrm{BEH}_{\phi}\rangle) \text{ and } B|_{\Sigma_1} \in \mathbf{C}\}$. In particular, if we consider the model class of some ASL-like structured specification SP of signature Σ_1, we have $\mathcal{E}^{\mathcal{L}}_{\phi}(Mod(SP)) = Mod(SP + \langle \Sigma_1^{\mathcal{L}}, \mathrm{BEH}_{\phi}\rangle)$.

According to Proposition 34, we can directly conclude that the axiomatic lifting induced by any axiomatization of the behavioural equality coincides with the semantic lifting defined in Definition 22.

Theorem 36. *Let* BEH_{Beh} *be the* $\mathcal{L}(\Sigma)$*-sentence induced by the axiomatization* Beh *of the behavioural equality. For any* Σ*-algebra* A *and any class* $\mathbf{C}$ *of* Σ*-algebras, the following holds:*

1. $\mathcal{L}(A) = \mathcal{E}^{\mathcal{L}}_{Beh}(A)$.
2. $\mathcal{L}(\mathbf{C}) = \mathcal{E}^{\mathcal{L}}_{Beh}(\mathbf{C})$.

In particular, if we consider the model class of some ASL-like structured specification SP *of signature* Σ*, we have* $\mathcal{L}(Mod(SP)) = Mod(SP + \langle \mathcal{L}(\Sigma), \mathrm{BEH}_{Beh}\rangle)$. □

5.4 Application to Various Classes of Algebras

We can apply the previous theorems to various classes of algebras of special interest.

Example 5. Let SP be an arbitrary ASL-like structured specification of signature Σ. Then, according to Theorems 24, 29 and 36, and to Example 3, we have:

1. $SP \models_{\approx} \phi$ if and only if $(SP + \langle \mathcal{L}(\Sigma), \mathrm{BEH}_{Beh}\rangle) \models \mathcal{L}(\phi)$.
2. **abstract** SP **w.r.t.** $\equiv \models_{\approx} \phi$ if and only if $(SP + \langle \mathcal{L}(\Sigma), \mathrm{BEH}_{Beh}\rangle) \models \mathcal{L}(\phi)$.
3. **behaviour** SP **w.r.t.** $\approx \models_{\approx} \phi$ if and only if $(SP + \langle \Sigma, \mathrm{FA}_{Beh}\rangle) \models \phi$.
4. Assume that SP is behaviourally consistent w.r.t. $\approx$. Then $SP \models_{\approx} \phi$ iff **abstract** SP **w.r.t.** $\equiv \models_{\approx} \phi$ iff **behaviour** SP **w.r.t.** $\approx \models_{\approx} \phi$ iff $(SP + \langle \Sigma, \mathrm{FA}_{Beh}\rangle) \models \phi$. ♢

As shown by Example 5, the combination of the results of Sections 4 and 5 allows us to characterize behavioural theories in terms of standard theories using an axiomatization Beh of the behavioural equality. In particular, we have seen how we can axiomatize fully abstract algebras (through FA_{Beh}) and lifted algebras (through BEH_{Beh}). However, in general these axiomatizations are infinitary ones. To perform proofs is therefore still difficult. One possibility is to replace these infinitary sentences by infinitary proof rules (such as *context induction* in the particular case of the total observational equality) and to show that the resulting proof system is sound and complete (cf. [11]). Another possibility, studied in the next sections, is to find special (but general enough) cases where the infinitary axiomatization can be replaced by a finitary one.

6 Finitary Axiomatization of the Behavioural Equality

In practice it is in general simple to find an infinitary axiomatization of the behavioural equality (indeed in the observational framework this axiomatization is directly deduced from the definition of the observational equality). Unfortunately it is usually difficult to find a finitary axiomatization of $\approx$ which for any Σ-algebra A is equivalent to the infinitary one. But since we are always interested in the behavioural theory of some given class $\mathbf{C}$ of Σ-algebras we do not really need a finite axiomatization of the behavioural equality for any arbitrary Σ-algebra but rather just for the algebras in the class $\mathbf{C}$ we are interested in.

In the most simple cases, it turns out that, provided the class $\mathbf{C}$ we are interested in satisfies some simple property expressed by a finitary sentence, we can find a finitary axiomatization of the behavioural equality for the algebras belonging to $\mathbf{C}$ (cf. Section 7, Corollary 43). In general, however, it may be necessary to introduce additional hidden sorts and function symbols that will prove useful in getting rid of the infinitary axiomatization. This idea leads to the notion of finitary axiomatization with hidden part as formalized below. We still assume given a signature $\Sigma = (S, F)$ and a regular family $\approx$ of partial Σ-congruences.

Definition 37 (Finitary axiomatization with hidden part). Let $\mathbf{C}$ be a class of Σ-algebras.

1. Let HID be a specification with **finitary axioms** plus **reachability constraints** of the form $\langle \Sigma_H, \mathcal{R}_H, \mathcal{A}x_H \rangle$, with $\Sigma \subseteq \Sigma_H$, which defines hidden sorts (possibly constrained) and hidden function symbols.

2. Let $HBeh = (HBeh_s(x_s, y_s))_{s \in S}$ be an S-sorted family of **finitary Σ_H-formulas**, where x_s and y_s are the only free variables, of sort s, of $HBeh_s(x_s, y_s)$.

3. Let $Ext_H(\mathbf{C}) \stackrel{\text{def}}{=} \{\Sigma_H\text{-algebra } A_H \mid A_H \in Mod(\text{HID}) \text{ and } A_H|_\Sigma \in \mathbf{C}\}$.

(HID, $HBeh$) is called a finitary axiomatization with hidden part of the behavioural equality $\approx$ with respect to the class $\mathbf{C}$ of Σ-algebras if the following conditions are satisfied:

(i) All Σ-algebras $A \in \mathbf{C}$ can be extended to (at least) an algebra $A_H \in Mod(\text{HID})$, with $A_H|_\Sigma = A$, i.e. $Ext_H(\mathbf{C})|_\Sigma = \mathbf{C}$.

(ii) For any Σ_H-algebra $A_H \in Ext_H(\mathbf{C})$, $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H)|_{\mathcal{L}(\Sigma)} = \mathcal{L}(A_H|_\Sigma)$.
(Remember that $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H)$ denotes the axiomatic lifting of A_H induced by $HBeh$, cf. Definition 35.)

If this is the case, HID will be called the hidden part of the finitary axiomatization.

Remark. Our terminology is consistent with Definition 27 for the following reason. Let $A \in \mathbf{C}$ be a Σ-algebra. Then, according to the condition (i), there exists at least one Σ_H-algebra $A_H \in Mod(\text{HID})$ such that $A_H|_\Sigma = A$ (and $A_H \in Ext_H(\mathbf{C})$). Then the condition (ii) implies that $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H)|_{\mathcal{L}(\Sigma)} = \mathcal{L}(A)$, which means that for any sort $s \in S$ and any valuation $\alpha : \{x_s, y_s\} \to A_H$, $A_H, \alpha \models HBeh_s(x_s, y_s)$ if and only if $\alpha(x_s) \approx_A \alpha(y_s)$.

Some comments may help to provide a better understanding of the definition above:

1. The main goal is to replace the family Beh of infinitary formulas $Beh_s(x_s, y_s)$ by a family $HBeh$ of finitary formulas $HBeh_s(x_s, y_s)$. Then obviously the induced formula BEH_{HBeh} (cf. Definition 35) will be a finitary formula as well (since we assume that S is finite). Intuitively BEH_{HBeh} will characterize some lifted algebras (as the formula BEH_{Beh} does for all algebras, cf. Theorem 36).

2. It is a standard idea to use auxiliary hidden sorts and function symbols to replace formulas expressed in some logic by "equivalent" formulas expressed in a less powerful one (cf. e.g. [21]). It is therefore natural to introduce the hidden part HID in our axiomatization.

3. The finitary axiomatization is "adequate" for the class $\mathbf{C}$ if HID is a conservative extension of $\mathbf{C}$ (cf. condition (i)) and if moreover for all algebras in $\mathbf{C}$ extended accordingly to HID, the finitary axiomatization $HBeh$ is "equivalent" to Beh (cf. condition (ii)).

4. In the most simple cases no hidden part is necessary, i.e. we can choose HID equal to $\langle \Sigma, \emptyset, \emptyset \rangle$. Then $Ext_H(\mathbf{C}) = \mathbf{C}$ and $HBeh$ is simply a family of finitary Σ-formulas. In that case $(\langle \Sigma, \emptyset, \emptyset \rangle, HBeh)$ is a finitary axiomatization of the behavioural equality with respect to a given class $\mathbf{C}$ of Σ-algebras if and only if, for any Σ-algebra $A \in \mathbf{C}$, $\mathcal{E}^{\mathcal{L}}_{HBeh}(A) = \mathcal{L}(A)$.

It is obvious that the results obtained in the previous section carry over to finitary axiomatizations with hidden part. The following theorem provides a summary.

Theorem 38. *Let* $(\mathrm{HID}, HBeh)$ *be a finitary axiomatization with hidden part of the behavioural equality* $\approx$ *with respect to a class* $\mathbf{C}$ *of* Σ*-algebras, where* $\mathrm{HID} = \langle \Sigma_H, \mathcal{R}_H, \mathcal{A}x_H \rangle$. *Then we have:*

1. *Let* FA_{HBeh} *be the following finitary* Σ_H*-sentence:*
   $\mathrm{FA}_{HBeh} \stackrel{\mathrm{def}}{=} \bigwedge_{s \in S} \forall x_s, y_s{:}s.\, [HBeh_s(x_s, y_s) \Leftrightarrow x_s = y_s]$.
   $FA_{\approx}(\mathbf{C}) = (Ext_H(\mathbf{C}) \cap Mod(\langle \Sigma_H, \mathrm{FA}_{HBeh} \rangle))|_{\Sigma}$.

2. $\mathcal{L}(\mathbf{C}) = (\mathcal{E}^{\mathcal{L}}_{HBeh}(Ext_H(\mathbf{C})))|_{\mathcal{L}(\Sigma)}$.

3. *In particular, if we consider the model class of some ASL-like structured specification* SP *of signature* Σ*, then* $Ext_H(Mod(SP)) = Mod(SP + \mathrm{HID})$*, and the condition (i) of Definition 37 means that* HID *is a conservative extension of* SP*, i.e.* $Mod(SP + \mathrm{HID})|_{\Sigma} = Mod(SP)$*. Hence we have:*

   (a) $FA_{\approx}(Mod(SP)) = Mod(SP + \mathrm{HID} + \langle \Sigma_H, \mathrm{FA}_{HBeh} \rangle)|_{\Sigma}$.

   (b) $\mathcal{L}(Mod(SP)) = Mod(SP + \mathrm{HID} + \langle \Sigma^{\mathcal{L}}_H, \mathrm{BEH}_{HBeh} \rangle)|_{\mathcal{L}(\Sigma)}$.

   It is important to note that if SP *is a smooth specification, then the specifications* $SP + \mathrm{HID}$, $SP + \mathrm{HID} + \langle \Sigma_H, \mathrm{FA}_{HBeh} \rangle$, $SP + \mathrm{HID} + \langle \Sigma^{\mathcal{L}}_H, \mathrm{BEH}_{HBeh} \rangle$ *are smooth as well (cf. Section 2).*

4. *In particular, if there is no hidden part, i.e.* $\mathrm{HID} = \langle \Sigma, \emptyset, \emptyset \rangle$, *and if we still consider the model class of some ASL-like structured specification* SP *of signature* Σ, *then the condition (i) of Definition 37 is trivially satisfied, and we have:*

   (a) $FA_{\approx}(Mod(SP)) = Mod(SP + \langle \Sigma, \mathrm{FA}_{HBeh} \rangle)$.

   (b) $\mathcal{L}(Mod(SP)) = Mod(SP + \langle \mathcal{L}(\Sigma), \mathrm{BEH}_{HBeh} \rangle)$. □

Example 6. Let $(\mathrm{HID}, HBeh)$ be a finitary axiomatization with hidden part of the behavioural equality $\approx$ with respect to the model class of some ASL-like structured specification SP of signature Σ. Then the combination of Theorems 24 and 38 shows that, for any Σ-formula ϕ:

1. $SP \models_{\approx} \phi$ iff $(SP + \mathrm{HID} + \langle \Sigma_H^{\mathcal{L}}, \mathrm{BEH}_{HBeh} \rangle) \models \mathcal{L}(\phi)$.
2. **abstract** SP **w.r.t.** $\equiv \models_{\approx} \phi$ iff $(SP + \mathrm{HID} + \langle \Sigma_H^{\mathcal{L}}, \mathrm{BEH}_{HBeh} \rangle) \models \mathcal{L}(\phi)$.
3. **behaviour** SP **w.r.t.** $\approx \models_{\approx} \phi$ iff $(SP + \mathrm{HID} + \langle \Sigma_H, \mathrm{FA}_{HBeh} \rangle) \models \phi$.
4. Assume that SP is behaviourally consistent w.r.t. $\approx$. Then $SP \models_{\approx} \phi$ iff **abstract** SP **w.r.t.** $\equiv \models_{\approx} \phi$ iff **behaviour** SP **w.r.t.** $\approx \models_{\approx} \phi$ iff $(SP + \mathrm{HID} + \langle \Sigma_H, \mathrm{FA}_{HBeh} \rangle) \models \phi$. ♢

As shown in Example 6, once a finitary axiomatization (with hidden part) of the behavioural equality is provided, we can use "standard" proof techniques to prove the behavioural validity of some formula with respect to a given smooth specification (or with respect to the behavioural or abstractor specification built on top of a smooth specification). The aim of the next section is to explain how one can find adequate finitary axiomatizations with hidden part in the particular case of observational equalities.

7 Axiomatization of the Observational Equality

We will now focus on observational equalities, and we assume given a signature $\Sigma = (S, F)$, a set Obs of observable sorts and a set In of input sorts. X_{In} is the S-sorted family of variables defined by $(X_{In})_s = \emptyset$ if $s \notin In$ and $(X_{In})_s = X_s$ if $s \in In$ (where $X = (X_s)_{s \in S}$ is the generally assumed family of countably infinite sets of variables of sort s). The problem to be solved is to find a finitary axiomatization of the observational equality $\approx_{Obs,In}$ (w.r.t. a given class $\mathbf{C}$ of Σ-algebras). Remember that for any Σ-algebra A, the definition domain $\mathrm{Dom}(\approx_{Obs,In,A})$ of $\approx_{Obs,In,A}$ is equal to $A[X_{In}]$, the smallest subalgebra of A generated by Σ and X_{In}.

Let us consider again the axiomatization of $\approx_{Obs,In}$ given in Example 4:
$Beh_s(x_s, y_s) \stackrel{\mathrm{def}}{=} \mathrm{Def}_s(x_s) \wedge \mathrm{Def}_s(y_s) \wedge \bigwedge_{C \in \mathcal{C}_{\Sigma}^{Obs}(s)} \forall \mathrm{Var}(C).\, \mathrm{DEF}(\mathrm{Var}(C)) \Rightarrow C[x_s] = C[y_s]$,
where $\mathrm{Def}_s(x_s)$ is an abbreviation for $\bigvee_{t \in T_{\Sigma}(X_{In})_s} \exists \mathrm{Var}(t).\, x_s = t$
and $\mathrm{DEF}(\mathrm{Var}(C))$ stands for $\bigwedge_{v_{s'} \in \mathrm{Var}(C)} \mathrm{Def}_{s'}(v_{s'})$.

It is clear that we have two distinct reasons for obtaining infinitary formulas: on one hand we must consider an infinite set of observable contexts $\mathcal{C}_{\Sigma}^{Obs}$ which represents the

infinitely many experiments with observable results, and this leads to the infinitary conjunction $\bigwedge_{C \in \mathcal{C}_\Sigma^{Obs}(s)} \cdots$; on the other hand we must only consider values that are denotable by a term $t \in T_\Sigma(X_{In})$, and this leads to the infinitary disjunctions $\bigvee_{t \in T_\Sigma(X_{In})_s} \cdots$. A natural idea to get rid of the first cause of infinitary axiomatization is to check whether, under some conditions, it would be enough to consider some adequate finite set of observable contexts instead of the infinite set of all observable contexts. Indeed, if successful, this idea would be enough to solve our problem in the special case of the total observational equality.

In general, however, it is clear that this idea will not be powerful enough. First, if we consider a partial observational equality, we must still face the infinitary $\mathrm{Def}_s(.)$ parts. Moreover, it is not always possible to consider only a finite subset of observable contexts. In these cases we must find an appropriate hidden part to obtain a finitary axiomatization of the observational equality $\approx_{Obs,In}$.

7.1 Using a Smaller Set of Observable Contexts

Before we consider more concretely how to construct a hidden part in order to obtain a finitary axiomatization of the observational equality, we will first provide a general characterization of finitary axiomatizations of the observational equality. This characterization will point out that under some conditions it is enough to axiomatize a contextual equality (cf. Definition 14) with respect to some (smaller) subset $\mathcal{C}$ of the set $\mathcal{C}_\Sigma^{Obs}$ of all observable contexts.

For this purpose we will assume that Definition 27 is generalized in a straightforward way to axiomatizations of contextual equalities. Then, if $\mathcal{C}$ is an arbitrary set of Σ-contexts, we know (cf. Example 4) that the partial contextual equality $\approx_{\mathcal{C},In}$ induced by $\mathcal{C}$ and In is axiomatized by the S-sorted family of formulas $Beh^{\mathcal{C}} = (Beh^{\mathcal{C}}_s(x_s, y_s))_{s \in S}$, with $Beh^{\mathcal{C}}_s(x_s, y_s) \stackrel{\text{def}}{=} \mathrm{Def}_s(x_s) \wedge \mathrm{Def}_s(y_s) \wedge \bigwedge_{C \in \mathcal{C}(s)} \forall \mathrm{Var}(C).\, \mathrm{DEF}(\mathrm{Var}(C)) \Rightarrow C[x_s] = C[y_s]$,
where $\mathrm{Def}_s(x_s)$ is an abbreviation for $\bigvee_{t \in T_\Sigma(X_{In})_s} \exists \mathrm{Var}(t).\, x_s = t$
and $\mathrm{DEF}(\mathrm{Var}(C))$ stands for $\bigwedge_{v_{s'} \in \mathrm{Var}(C)} \mathrm{Def}_{s'}(v_{s'})$.

In a first step we will study some sufficient conditions under which the contextual equality induced by a set $\mathcal{C}$ of Σ-contexts and a set In of input sorts coincides with the observational equality $\approx_{Obs,In}$.

Lemma 39. *Let A be a Σ-algebra and $\approx_A$ be an arbitrary partial congruence on A. If $\mathrm{Dom}(\approx_A) = \mathrm{Dom}(\approx_{Obs,In,A})(= A[X_{In}])$ and if $\approx_A$ "coincides" with the set-theoretic equality on the carrier sets of all observable sorts, i.e. for all $a, b \in \mathrm{Dom}(\approx_A)_s$, with $s \in Obs$, we have $a \approx_A b$ if and only if $a = b$, then $\approx_A \subseteq \approx_{Obs,In,A}$.*

Proof. Let a, b be two elements of A_s, for some sort $s \in S$, and assume that $a \approx_A b$. Then both a and b belong to $A[X_{In}]$. Since $\approx_A$ is a partial congruence (hence is compatible with the signature Σ), we have, for any observable context $C \in \mathcal{C}_\Sigma^{Obs}(s)$ and for any valuation $\alpha : X \to A[X_{In}]$, $I_{\alpha_a}(C) \approx_A I_{\alpha_b}(C)$, where $\alpha_a, \alpha_b : X \cup \{z_s\} \to A[X_{In}]$ are the unique extensions of α defined by $\alpha_a(z_s) = a$ and $\alpha_b(z_s) = b$, where z_s is the context

variable of C. Since C is an observable context, both $I_{\alpha_a}(C)$ and $I_{\alpha_b}(C)$ belong to the carrier set of some observable sort. But since we have assumed that $\approx_A$ "coincides" with the set-theoretic equality on the carrier sets of the observable sorts, $I_{\alpha_a}(C) \approx_A I_{\alpha_b}(C)$ implies $I_{\alpha_a}(C) = I_{\alpha_b}(C)$, hence we have $a \approx_{Obs,In,A} b$. Therefore $\approx_A \subseteq \approx_{Obs,In,A}$. □

Remark. Indeed it is easy to prove that the set of all partial congruences on A which have $A[X_{In}]$ as definition domain and which "coincide" with the set-theoretic equality on each observable sort is a complete lattice, the smallest element of which is the restriction of the set-theoretic equality to $A[X_{In}]$, the greatest element being the observational equality $\approx_{Obs,In,A}$.

Lemma 40. *Let $\mathcal{C} \subseteq \mathcal{C}_\Sigma^{Obs}$ be an arbitrary subset of observable Σ-contexts such that, for any observable sort $s \in Obs$, $z_s \in \mathcal{C}$ and let A be a Σ-algebra. Then $\approx_{\mathcal{C},In,A} = \approx_{Obs,In,A}$ if and only if $\approx_{\mathcal{C},In,A}$ is a partial Σ-congruence.*

Proof. First note that, according to Definition 14, $\mathrm{Dom}(\approx_{\mathcal{C},In,A}) = A[X_{In}]$. Assume that $\approx_{\mathcal{C},In,A}$ is a partial Σ-congruence. Since $\mathcal{C} \subseteq \mathcal{C}_\Sigma^{Obs}$, obviously we have $\approx_{Obs,In,A} \subseteq \approx_{\mathcal{C},In,A}$. To prove that $\approx_{\mathcal{C},In,A} \subseteq \approx_{Obs,In,A}$, by Lemma 39, it is enough to prove that $\approx_{\mathcal{C},In,A}$ "coincides" with the set-theoretic equality for each carrier set of an observable sort. But this holds since $\mathcal{C}$ contains by assumption all the "trivial" contexts z_s when s is an observable sort. The converse direction is trivial. □

Using the last lemma we can infer necessary and sufficient conditions under which the family of formulas $Beh^{\mathcal{C}}$ is "equivalent" to the axiomatization of the observational equality $\approx_{Obs,In}$.

Proposition 41 (Criteria for using a smaller set of contexts). *Let $\mathcal{C} \subseteq \mathcal{C}_\Sigma^{Obs}$ be an arbitrary subset of observable Σ-contexts such that, for any observable sort $s \in Obs$, $z_s \in \mathcal{C}$. Let Beh and $Beh^{\mathcal{C}}$ be the respective axiomatizations of $\approx_{Obs,In}$ and $\approx_{\mathcal{C},In}$. For any Σ_H-algebra A_H, with $\Sigma \subseteq \Sigma_H$, the following two conditions are equivalent:*

1. $\mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H) = \mathcal{E}^{\mathcal{L}}_{Beh}(A_H)$
2. $\mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H) \models \mathrm{CONG}^{\sim}_{\Sigma}$

where $\mathrm{CONG}^{\sim}_{\Sigma}$ is the finitary $\mathcal{L}(\Sigma)$-sentence defined by $\mathrm{CONG}^{\sim}_{\Sigma} \stackrel{\mathrm{def}}{=} \bigwedge_{f \in F} \mathrm{CONG}^{\sim}_{f}$, where for each function symbol $f \in F$ of arity $s_1 \ldots s_n \to s$, the sentence $\mathrm{CONG}^{\sim}_{f}$ is defined by: $\mathrm{CONG}^{\sim}_{f} \stackrel{\mathrm{def}}{=} \forall x_1, y_1{:}s_1, \ldots, x_n, y_n{:}s_n. \left[\left(\bigwedge_{1 \le i \le n} x_i \sim_{s_i} y_i\right) \Rightarrow f(x_1, \ldots, x_n) \sim_s f(y_1, \ldots, y_n)\right]$.

Proof.

1 ⇒ 2: Is obvious since Beh is the axiomatization of the (partial) Σ-congruence $\approx_{Obs,In}$, hence $\mathcal{E}^{\mathcal{L}}_{Beh}(A_H) \models \mathrm{CONG}^{\sim}_{\Sigma}$.

2 ⇒ 1: Assume that $\mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H) \models \mathrm{CONG}^{\sim}_{\Sigma}$. But then, since $Beh^{\mathcal{C}}$ is an axiomatization of the contextual equality $\approx_{\mathcal{C},In}$, we know that $\approx_{\mathcal{C},In,A_H|_\Sigma}$ is a partial Σ-congruence, and using Lemma 40 we conclude that $\approx_{\mathcal{C},In,A_H|_\Sigma} = \approx_{Obs,In,A_H|_\Sigma}$. Hence $\mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H|_\Sigma) = \mathcal{E}^{\mathcal{L}}_{Beh}(A_H|_\Sigma)$, and therefore $\mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H) = \mathcal{E}^{\mathcal{L}}_{Beh}(A_H)$. □

From Proposition 41 we can infer the following characterization of finitary axiomatizations of the observational equality $\approx_{Obs,In}$. Intuitively this characterization says that for axiomatizing the observational equality $\approx_{Obs,In}$, it is enough to axiomatize some contextual equality $\approx_{\mathcal{C},In}$, provided that $\approx_{\mathcal{C},In,A}$ is a partial Σ-congruence for all algebras A in the class $\mathbf{C}$ of interest.

Theorem 42 (Characterization of finitary axiomatizations of $\approx_{Obs,In}$). *Let $\mathbf{C}$ be a class of Σ-algebras, and let $(\mathrm{HID}, HBeh)$ be as in Definition 37. $(\mathrm{HID}, HBeh)$ is a finitary axiomatization with hidden part of the observational equality $\approx_{Obs,In}$ with respect to the class $\mathbf{C}$ of Σ-algebras if and only if the following conditions are satisfied:*

1. $Ext_H(\mathbf{C})|_\Sigma = \mathbf{C}$.
2. *For any Σ_H-algebra $A_H \in Ext_H(\mathbf{C})$, $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H) \models \mathrm{CONG}^{\sim}_{\Sigma}$.*
3. *There exists a (possibly infinite) set $\mathcal{C} \subseteq \mathcal{C}^{Obs}_{\Sigma}$ of observable Σ-contexts which contains, for any observable sort $s \in Obs$, the trivial context z_s and such that, for any Σ_H-algebra $A_H \in Ext_H(\mathbf{C})$, $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H) = \mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H)$, where $Beh^{\mathcal{C}}$ is the axiomatization of the contextual equality $\approx_{\mathcal{C},In}$.*

Proof. It is obviously enough to show that the condition (ii) of Definition 37 is equivalent to the conditions (2) and (3). Let Beh be the axiomatization of the observational equality $\approx_{Obs,In}$ and let $A_H \in Ext_H(\mathbf{C})$.

$\Longrightarrow$: Assume that $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H)|_{\mathcal{L}(\Sigma)} = \mathcal{L}(A_H|_{\mathcal{L}(\Sigma)})$. Then $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H)|_{\mathcal{L}(\Sigma)} = \mathcal{E}^{\mathcal{L}}_{Beh}(A_H|_{\mathcal{L}(\Sigma)})$ and therefore $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H) = \mathcal{E}^{\mathcal{L}}_{Beh}(A_H)$. Hence Condition (2) is obviously satisfied. On the other hand Condition (3) is trivially satisfied as well by choosing $\mathcal{C} = \mathcal{C}^{Obs}_{\Sigma}$.

$\Longleftarrow$: Assume that the conditions (2) and (3) hold. Then $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H) \models \mathrm{CONG}^{\sim}_{\Sigma}$ and $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H) = \mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H)$, for some $\mathcal{C} \subseteq \mathcal{C}^{Obs}_{\Sigma}$ with $z_s \in \mathcal{C}$ for all $s \in Obs$. Hence $\mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H) \models \mathrm{CONG}^{\sim}_{\Sigma}$ and therefore, by Proposition 41, $\mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H) = \mathcal{E}^{\mathcal{L}}_{Beh}(A_H)$. Hence $\mathcal{E}^{\mathcal{L}}_{HBeh}(A_H)\,|_{\mathcal{L}(\Sigma)} = \mathcal{E}^{\mathcal{L}}_{Beh}(A_H)\,|_{\mathcal{L}(\Sigma)} = \mathcal{E}^{\mathcal{L}}_{Beh}(A_H\,|_{\mathcal{L}(\Sigma)}) = \mathcal{L}(A_H\,|_{\mathcal{L}(\Sigma)})$, i.e. the condition (ii) of Definition 37 holds. □

The following corollary shows that any finite set $\mathcal{C} \subset \mathcal{C}^{Obs}_{\Sigma}$ of observable Σ-contexts (which contains, for any observable sort $s \in Obs$, the trivial context z_s) induces a finitary axiomatization of the total observational equality $\approx_{Obs,S}$ w.r.t. any class $\mathbf{C}$ for which the contextual equality $\approx_{\mathcal{C},S}$ is a Σ-congruence.

Corollary 43. *Let $\mathbf{C}$ be a class of Σ-algebras and let $\mathcal{C} \subset \mathcal{C}^{Obs}_{\Sigma}$ be a* **finite** *set of observable Σ-contexts such that, for any observable sort $s \in Obs$, $z_s \in \mathcal{C}$. Let $TBeh$ be the S-sorted family of finitary Σ-formulas defined by:*
$TBeh_s(x_s, y_s) \stackrel{\mathrm{def}}{=} \bigwedge_{C \in \mathcal{C}(s)} \forall \mathrm{Var}(C).\, C[x_s] = C[y_s]$.
$TBeh$ is a finitary axiomatization (with an empty hidden part) of the total observational equality $\approx_{Obs,S}$ with respect to the class $\mathbf{C}$ of Σ-algebras if and only if we have:
$\mathcal{E}^{\mathcal{L}}_{TBeh}(\mathbf{C}) \models \mathrm{CONG}^{\sim}_{\Sigma}$.

Proof. The family $TBeh$ is obviously an axiomatization of the total contextual equality $\approx_{\mathcal{C},S}$ (cf. Example 4). Hence the corollary is a direct consequence of Theorem 42. □

The above corollary solves our problem in the simplest cases, i.e. when we consider a total observational equality and when, moreover, we can find a finite subset $\mathcal{C}$ of observable Σ-contexts such that the class $\mathbf{C}$ of Σ-algebras we are interested in is such that $\mathcal{E}^{\mathcal{L}}_{TBeh}(\mathbf{C}) \models \mathrm{CONG}^{\sim}_{\Sigma}$: in this case we can immediately apply Theorem 38(4). If it is not possible to find a convenient finite subset $\mathcal{C}$, or if we consider a partial observational equality, then we must use an adequate hidden part to obtain a suitable finitary axiomatization.

7.2 Hidden Part for the Observational Equality

The aim of this subsection is to explain how to find a finitary axiomatization $(HID, HBeh)$ with hidden part of the partial observational equality $\approx_{Obs,In}$ w.r.t. some class $\mathbf{C}$ of Σ-algebras. In a first step we must get rid of the infinitary $\mathrm{Def}_s(.)$ parts. For this we suggest to use auxiliary hidden predicate symbols $D_{H,s}$ axiomatized in such a way that $D_{H,s}(x_s)$ holds if and only if $\mathrm{Def}_s(x_s)$ holds. In a second step we must get rid of the infinitary conjunctions over observable contexts. The previous subsection suggests to replace the set of all observable contexts by some smaller subset $\mathcal{C}$. However, if we cannot find a convenient finite subset $\mathcal{C}$, then we still have infinitary conjunctions over the contexts of $\mathcal{C}$. In that case we suggest to use auxiliary hidden function symbols that will intuitively provide the same observations than $\mathcal{C}$, and to use a finite set $\mathcal{C}_H$ of observable contexts built with the help of these hidden function symbols.

Let us first explain how a family of finitary formulas is induced by the predicate symbols $D_{H,s}$ and by the choice of a finite set of contexts $\mathcal{C}_H$.

Definition 44. Let Σ_H be a signature such that $\Sigma \subseteq \Sigma_H$, and such that for each sort $s \in S$, there exists a unary predicate symbol $D_{H,s}$ of domain s in Σ_H. Let $\mathcal{C}_H$ be an arbitrary finite set of Σ_H-contexts (with variables in X_H, where $X_H = (X_{H,s})_{s \in S_H}$ is the generally assumed family of countably infinite sets of variables of sort $s \in S_H$). The S-sorted family of finitary Σ_H-formulas induced by the predicate symbols $D_{H,s}$ and the contexts $\mathcal{C}_H$, denoted by $HBeh[D_H, \mathcal{C}_H]$, is defined by:
$HBeh[D_H, \mathcal{C}_H]_s(x_s, y_s) \stackrel{\mathrm{def}}{=} D_{H,s}(x_s) \wedge D_{H,s}(y_s) \wedge$
$\bigwedge_{C_H \in \mathcal{C}_H(s)} \forall \mathrm{Var}(C_H).\, \mathrm{DEF}_H(\mathrm{Var}(C_H)) \Rightarrow C_H[x_s] = C_H[y_s]$,
where $\mathrm{DEF}_H(\mathrm{Var}(C_H))$ stands for $\bigwedge_{v_{s'} \in \mathrm{Var}(C_H), s' \in S} D_{H,s'}(v_{s'})$.

Remark. Note that $HBeh[D_H, \mathcal{C}_H]$ is not in general the axiomatization of some contextual equality.

Theorem 42 provides an abstract characterization of finitary axiomatizations of the observational equality $\approx_{Obs,In}$, which is valid for any couple $(HID, HBeh)$. However, if we know that we are going to use the family $HBeh[D_H, \mathcal{C}_H]$ induced by the predicate symbols $D_{H,s}$ and a finite set of contexts $\mathcal{C}_H$, we can derive more precise conditions.

Proposition 45 (Criteria for the hidden part). *Let $\mathcal{C}$ be an arbitrary set of Σ-contexts and let $Beh^{\mathcal{C}}$ be the axiomatization of the contextual equality $\approx_{\mathcal{C},In}$. Let Σ_H be a signature such that $\Sigma \subseteq \Sigma_H$, and such that for each sort $s \in S$, there exists a unary predicate symbol $D_{H,s}$ of domain s in Σ_H. Let $\mathcal{C}_H$ be an arbitrary finite set of Σ_H-contexts (with variables in X_H). Let $HBeh[D_H, \mathcal{C}_H]$ be the S-sorted family of finitary Σ_H-formulas*

induced by the predicate symbols $D_{H,s}$ and the contexts $\mathcal{C}_H$.

For any Σ_H-algebra A_H, $\mathcal{E}^{\mathcal{L}}_{HBeh[D_H,\mathcal{C}_H]}(A_H) = \mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H)$ if the following three conditions are satisfied:

(i) For each sort $s \in S$ and any value $a \in (A_H)_s$, $D^{A_H}_{H,s}(a)$ if and only if there exists a term $t \in T_\Sigma(X_{In})$ and a valuation $\alpha : X_{In} \to A_H$ such that $I_\alpha(t) = a$.

(ii) For each sort $s \in S$, for each Σ_H-context $C_H \in \mathcal{C}_H(s)$, for each valuation $\alpha : X_H \to A_H$ such that $D^{A_H}_{H,s'}(\alpha(v_{s'}))$ for all $s' \in S$ and all $v_{s'} \in X_{H,s'}$, there exists a Σ-context $C \in \mathcal{C}(s)$ and a valuation $\beta : X \to A_H$ with $D^{A_H}_{H,s'}(\beta(v_{s'}))$ for all $s' \in S$ and all $v_{s'} \in X_{s'}$, such that, for any value $a \in (A_H)_s$ with $D^{A_H}_{H,s}(a)$, $I_{\beta_a}(C) = I_{\alpha_a}(C_H)$, where I_{β_a} and I_{α_a} are the unique extensions of I_β and I_α respectively defined by $I_{\beta_a}(z_s) = a$ and $I_{\alpha_a}(z_s) = a$.

(iii) For each sort $s \in S$, for each Σ-context $C \in \mathcal{C}(s)$, for each valuation $\alpha : X \to A_H$ such that $D^{A_H}_{H,s'}(\alpha(v_{s'}))$ for all $s' \in S$ and all $v_{s'} \in X_{s'}$, there exists a Σ_H-context $C_H \in \mathcal{C}_H(s)$ and a valuation $\beta : X_H \to A_H$ with $D^{A_H}_{H,s'}(\beta(v_{s'}))$ for all $s' \in S$ and all $v_{s'} \in X_{H,s'}$, such that, for any value $a \in (A_H)_s$ with $D^{A_H}_{H,s}(a)$, $I_{\beta_a}(C_H) = I_{\alpha_a}(C)$, where I_{β_a} and I_{α_a} are the unique extensions of I_β and I_α respectively defined by $I_{\beta_a}(z_s) = a$ and $I_{\alpha_a}(z_s) = a$.

Proof. Let A_H be an arbitrary Σ_H-algebra. From (i) we can conclude that for any sort $s \in S$ and any valuation $\alpha : \{x_s\} \to A_H$, $A_H, \alpha \models D_{H,s}(x_s)$ if and only if $A_H, \alpha \models \mathrm{Def}_s(x_s)$. Using this fact together with the conditions (ii) and (iii) we can infer that, for any sort $s \in S$ and any valuation $\alpha : \{x_s, y_s\} \to A_H$, $A_H, \alpha \models HBeh[D_H, \mathcal{C}_H]_s(x_s, y_s)$ if and only if $A_H, \alpha \models Beh^{\mathcal{C}}_s(x_s, y_s)$. Hence $\mathcal{E}^{\mathcal{L}}_{HBeh[D_H,\mathcal{C}_H]}(A_H) = \mathcal{E}^{\mathcal{L}}_{Beh^{\mathcal{C}}}(A_H)$. □

The combination of Theorem 42 and Proposition 45 leads to the following criteria for finite axiomatizations of the observational equality $\approx_{Obs,In}$.

Theorem 46 (Criteria for finitary axiomatizations of $\approx_{Obs,In}$). *Let* **C** *be a class of Σ-algebras. Let* HID *be a specification with finitary axioms plus reachability constraints of the form $\langle \Sigma_H, \mathcal{R}_H, \mathcal{A}x_H \rangle$, with $\Sigma \subseteq \Sigma_H$, such that for each sort $s \in S$, there exists a unary predicate symbol $D_{H,s}$ of domain s in Σ_H. Let $\mathcal{C}_H$ be an arbitrary finite set of Σ_H-contexts (with variables in X_H). Let $HBeh[D_H, \mathcal{C}_H]$ be the S-sorted family of finitary Σ_H-formulas induced by the predicate symbols $D_{H,s}$ and the contexts $\mathcal{C}_H$.*

(HID, $HBeh[D_H, \mathcal{C}_H]$) is a finitary axiomatization with hidden part of the observational equality $\approx_{Obs,In}$ with respect to the class **C** *of Σ-algebras if the following conditions are satisfied:*

1. *$Ext_H(\mathbf{C})|_\Sigma = \mathbf{C}$.*
2. *For any Σ_H-algebra $A_H \in Ext_H(\mathbf{C})$, $\mathcal{E}^{\mathcal{L}}_{HBeh[D_H,\mathcal{C}_H]}(A_H) \models \mathrm{CONG}^{\sim}_{\Sigma}$.*

3. *There exists a (possibly infinite) set* $\mathcal{C} \subseteq \mathcal{C}_{\Sigma}^{Obs}$ *of observable* Σ*-contexts which contains, for any observable sort* $s \in Obs$*, the trivial context* z_s *and such that any* Σ_H*-algebra* $A_H \in Ext_H(\mathbf{C})$ *satisfies the three conditions (i), (ii) and (iii) of Proposition 45 (w.r.t.* $\mathcal{C}$*).* □

Theorem 46 provides the key for constructing a finitary axiomatization of the observational equality $\approx_{Obs,In}$ with respect to a given class $\mathbf{C}$ of Σ-algebras. Indeed the conditions (1), (2) and (3) can be considered as proof obligations. Condition (1) requires that all algebras in $\mathbf{C}$ can be extended to a model of the specification HID, i.e. HID does not introduce any "confusion" on the algebras of $\mathbf{C}$. Condition (2) requires that the relation axiomatized by the formulas $HBeh[D_H, \mathcal{C}_H]$ is a Σ-congruence and condition (3) provides a criterion for this relation to coincide with the contextual equality induced by some set $\mathcal{C}$ of Σ-contexts and the input sorts In. It is important to note that if $\mathcal{C}_H$ consists only of Σ-contexts then the condition (3) reduces to the condition (i) of Proposition 45. If, moreover, we consider the total observational equality $\approx_{Obs,S}$ then condition (2) remains the only proof obligation (cf. Corollary 43). In general, however, one has to check the three conditions (i), (ii) and (iii) of Proposition 45. In particular conditions (ii) and (iii) look rather technical; in concrete examples however it turns out that both conditions can be checked quite easily because one has only to relate contexts of $\mathcal{C}_H$ to contexts of $\mathcal{C}$ and vice versa.

In the remainder of this section we will show that, for any partial observational equality $\approx_{Obs,In}$ and any class $\mathbf{C}$ of Σ-algebras, it is always possible to construct a finitary axiomatization with hidden part of $\approx_{Obs,In}$ w.r.t. $\mathbf{C}$. The idea is to encode the observable Σ-contexts and the Σ-terms with input variables in the hidden part, and to specify the application of contexts and the interpretation of terms by corresponding hidden function symbols. However, it should be clear that (at least) the encoding of the contexts is so complex that this result is of purely theoretical interest. In practice it is both more efficient and simpler to define an ad hoc hidden part and to discharge the proof obligations provided in Theorem 46.

Definition 47 (General encoding of definition domains and contexts). Let $\Sigma = (S, F)$ be a signature, $Obs \subseteq S$ be a set of observable sorts and $In \subseteq S$ be a set of input sorts. The specification $\mathrm{HID}[\Sigma, Obs, In] = \langle \Sigma_H, \mathcal{R}_H, \mathcal{A}x_H \rangle$, with $\Sigma_H = (S_H, F_H)$, is defined by:

1. Let $Ar[\Sigma] \stackrel{\text{def}}{=} \{s' \to s \mid s, s' \in S$ and there exists (at least) a Σ-context of sort s with context variable of sort $s'\}$.

2. S_H is equal to S plus:

   (a) for each sort $s \in S \setminus In$, a new sort $T[s]$.

   (b) for each $s' \to s \in Ar[\Sigma]$, a new sort $Ct[s' \to s]$.

3. F_H is equal to F plus:

Function symbols for the encoding of definition domains:

(a) for each function symbol $f \in F$ of arity $s_1 \dots s_n \to s$, with $s \notin In$, a new function symbol $T[f]$ of arity $\zeta_1 s_1 \dots \zeta_n s_n \to T[s]$, with $\zeta_i s_i = s_i$ if $s_i \in In$ and $\zeta_i s_i = T[s_i]$ otherwise.

(b) for each sort $s \in S \setminus In$, a new function symbol $I[s]$ of arity $T[s] \to s$.

(c) for each sort $s \in S$, a new unary predicate symbol $D_{H,s}$ of domain s.

Function symbols for the encoding of Σ-contexts:

(d) for each sort $s \in S$, a new constant $Ct[z_s]$ of sort $Ct[s \to s]$.

(e) for each non constant function symbol $f \in F$ of arity $s_1 \dots s_n \to s$, for each $i \in \{1, \dots, n\}$ and for each sort $s' \in S$ such that $s' \to s_i \in Ar[\Sigma]$, a new function symbol $Ct[f, i, s']$ of arity $\zeta_1 s_1 \dots Ct[s' \to s_i] \dots \zeta_n s_n \to Ct[s' \to s]$, with $\zeta_j s_j = s_j$ if $s_j \in In$ and $\zeta_j s_j = T[s_j]$ otherwise.

(f) for each $s' \to s \in Ar[\Sigma]$, a new function symbol $\mathrm{apply}_{s' \to s}$ of arity $Ct[s' \to s]\ s' \to s$.

4. $\mathcal{R}_H = (S_\mathcal{R}, F_\mathcal{R})$ is defined by:

(a) $S_\mathcal{R} \stackrel{\text{def}}{=} S_H \setminus S$.

(b) $F_\mathcal{R} \stackrel{\text{def}}{=} \{T[f] \mid f \in F\} \cup \{Ct[z_s] \mid s \in S\} \cup$
$\{Ct[f, i, s'] \mid f \in F \text{ of arity } s_1 \dots s_n \to s,\ n > 0,\ s' \to s_i \in Ar[\Sigma]\}$.

5. $\mathcal{A}x_H$ is the union of:

Axioms for the encoding of definition domains:

(a) for each sort $s \in S \setminus In$ and for each $T[f]$, the (implicitly universally quantified) equation: $I[s](T[f](x_1, \dots, x_n)) = f(\zeta_1 x_1, \dots, \zeta_n x_n)$,
with $\zeta_i x_i = x_i$ if $s_i \in In$ and $\zeta_i x_i = I[s_i](x_i)$ otherwise.

(b) for each sort $s \in S \setminus In$, the sentence $\forall x_s{:}s.\, D_{H,s}(x_s) \Leftrightarrow \exists y_s{:}T[s].\, I[s](y_s) = x_s$.

(c) for each sort $s \in In$, the sentence $\forall x_s{:}s.\, D_{H,s}(x_s)$.

Axioms for the encoding of Σ-contexts:

(d) for each sort $s \in S$, the sentence $\forall x_s{:}s.\ \mathrm{apply}_{s \to s}(Ct[z_s], x_s) = x_s$.

(e) for each $s' \to s \in Ar[\Sigma]$ and for each $Ct[f, i, s']$ the (implicitly universally quantified) equation:
$\mathrm{apply}_{s' \to s}(Ct[f, i, s'](x_1, \dots, c_i, \dots, x_n), x_{s'}) =$
$f(\zeta_1 x_1, \dots, \mathrm{apply}_{s' \to s_i}(c_i, x_{s'}), \dots, \zeta_n x_n)$,
where $\zeta_j x_j = x_j$ if $s_j \in In$ and $\zeta_j x_j = I[s_j](x_j)$ otherwise.

Proposition 48. *Let* $\mathrm{HID}[\Sigma, Obs, In]$ *be as in Definition 47. Let* $\mathcal{C}_H$ *be the set of* Σ_H*-contexts defined by* $\mathcal{C}_H \stackrel{\text{def}}{=} \{\mathrm{apply}_{s' \to s}(ctx, z_{s'}) \mid s' \to s \in Ar[\Sigma]$ *such that* $s \in Obs$, $s' \in S$, *ctx is an arbitrary but fixed variable of sort* $Ct[s' \to s]$ *and* $z_{s'}$ *is the context variable of sort* $s'\}$. *Let* $HBeh[D_H, \mathcal{C}_H]$ *be the* S*-sorted family of finitary* Σ_H*-formulas induced by*

the predicate symbols $D_{H,s}$ and the contexts $\mathcal{C}_H$.

Then $(\mathrm{HID}[\Sigma, Obs, In], HBeh[D_H, \mathcal{C}_H])$ *is a finitary axiomatization of the observational equality* $\approx_{Obs,In}$ *with respect to any class* $\mathbf{C}$ *of* Σ*-algebras.*

Proof. We must show that the three conditions of Theorem 46 are satisfied. It is easy to show that any Σ-algebra A can be extended to a model of $\mathrm{HID}[\Sigma, Obs, In]$ by a straightforward construction, i.e. condition (1) is satisfied. Condition (2) is trivially satisfied if we show that condition (3) holds with respect to the set $\mathcal{C}_\Sigma^{Obs}$ of all observable contexts. To prove this, we must check that the conditions (i), (ii) and (iii) of Proposition 45 are satisfied w.r.t. the set of all observable Σ-contexts $\mathcal{C}_\Sigma^{Obs}$ and any Σ_H-algebra $A_H \in Mod(\mathrm{HID}[\Sigma, Obs, In])$. Let $A_H \in Mod(\mathrm{HID}[\Sigma, Obs, In])$.

Proof of (i): W.l.o.g. let $s \in S \setminus In$. We have the following lemmas:

> **Lemma 49.** *For all values $T[a] \in (A_H)_{T[s]}$ there exists a term $t \in T_\Sigma(X_{In})$ and a valuation $\alpha : X_{In} \to A_H$ such that $I[s]^{A_H}(T[a]) = I_\alpha(t)$.*
>
> (Proof by induction on the generators of the sort $T[s]$.)
>
> **Lemma 50.** *For any term $t \in T_\Sigma(X_{In})$ and any valuation $\alpha : X_{In} \to A_H$, there exists a value $T[a] \in (A_H)_{T[s]}$ such that $I[s]^{A_H}(T[a]) = I_\alpha(t)$.*
>
> (Proof by structural induction on $T_\Sigma(X_{In})$.)
>
> From Lemmas 49 and 50 we conclude that for any value $a \in (A_H)_s$ the following holds: $D_{H,s}^{A_H}(a)$ iff (using the axioms defining $D_{H,s}$) there exists $T[a] \in (A_H)_{T[s]}$ such that $I[s]^{A_H}(T[a]) = a$ iff (by Lemmas 49 and 50) there exists a term $t \in T_\Sigma(X_{In})$ and a valuation $\alpha : X_{In} \to A_H$ such that $I_\alpha(t) = a$.

Proof of (ii): Let $s' \in S$ be arbitrary. We have to consider Σ_H-contexts of the form $\mathrm{apply}_{s' \to s}(ctx, z_{s'})$ with $s \in Obs$. Condition (ii) can now be easily derived from the following lemma:

> **Lemma 51.** *For all sorts $Ct[s' \to s] \in S_H$ and for all values $a_H \in (A_H)_{Ct[s' \to s]}$ there exists a Σ-context C of sort s with context variable $z_{s'}$ and a valuation $\beta : X \to A_H$ with $D_{H,s''}^{A_H}(\beta(v_{s''}))$ for all $s'' \in S$ and all $v_{s''} \in X_{s''}$ such that for any value $a \in (A_H)_{s'}$ we have $I_{\beta_a}(C) = I_{\alpha_a}(\mathrm{apply}_{s' \to s}(ctx, z_{s'}))$ where $\alpha_a(ctx) = a_H$, $\alpha_a(z_{s'}) = a$ and $\beta_a(z_{s'}) = a$.*
>
> (Proof by induction on the generators of the sort $Ct[s' \to s]$. Note that Lemma 51 is formulated for arbitrary sorts s, not only for observable ones, which allows to use the necessary induction hypotheses.)

Proof of (iii): Condition (iii) follows from another lemma which is symmetric to Lemma 51 and can be proved by induction on the structure of arbitrary (observable or not) Σ-contexts. □

8 How to Prove Behavioural Properties: Examples

In this section we discuss general guidelines for proving behavioural properties with respect to an observational equality $\approx_{Obs,In}$ and we show how to apply the results obtained in the previous sections to various examples.

In the following we assume given a signature $\Sigma = (S, F)$, a set Obs of observable sorts, a set In of input sorts, and an arbitrary specification SP of signature Σ. We want to prove that $SP \models_{\approx_{Obs,In}} \phi$, for some Σ-formula ϕ.

The results obtained in the previous sections lead to the following method:

1. Select an appropriate subset $\mathcal{C}$ of $\mathcal{C}_\Sigma^{Obs}$ (which contains the trivial contexts z_s for $s \in Obs$).
2. If $In \neq S$ then define an appropriate hidden part with predicate symbols $D_{H,s}$ (for any $s \in S$).
3. If $\mathcal{C}$ is infinite then complete the hidden part as needed and select some finite set of contexts $\mathcal{C}_H$ (built with hidden function symbols). Otherwise let $\mathcal{C}_H = \mathcal{C}$.
4. Consider the S-sorted family of finitary formulas $HBeh[D_H, \mathcal{C}_H]$ induced by the predicate symbols $D_{H,s}$ (if applicable) and the contexts $\mathcal{C}_H$.
5. Check that:
   (a) The resulting hidden part HID is a conservative extension of SP.
   (b) $(SP + \mathrm{HID} + \langle \Sigma_H^{\mathcal{L}}, \mathrm{BEH}_{HBeh[D_H,\mathcal{C}_H]} \rangle) \models \mathrm{CONG}_\Sigma^{\sim}$.
   (c) The criteria for the hidden part given in Proposition 45 are satisfied (w.r.t. the selected set of contexts $\mathcal{C}$).
6. Once these steps are done (once for all), Theorem 38 implies that $SP \models_{\approx_{Obs,In}} \phi$ if and only if $(SP + \mathrm{HID} + \langle \Sigma_H^{\mathcal{L}}, \mathrm{BEH}_{HBeh[D_H,\mathcal{C}_H]} \rangle) \models \mathcal{L}(\phi)$, for any Σ-formula ϕ.

Hence it is clear that the selection of an adequate subset $\mathcal{C}$ of $\mathcal{C}_\Sigma^{Obs}$ is a crucial step in the method. In particular, if we succeed in finding a convenient finite set $\mathcal{C}$, then the definition of the hidden part is considerably simplified (it is enough to introduce the predicate symbols $D_{H,s}$) and the checks to be done are much easier to discharge. But even if the selected set $\mathcal{C}$ is infinite, it is important to understand that the subsequent steps will be much easier if we consider a strict subset of $\mathcal{C}_\Sigma^{Obs}$ instead of the set of all observable contexts. In the next subsection we explain how in general such a set $\mathcal{C}$ can be easily deduced from the specification we are interested in. Then we provide various examples which show how our method can be applied.

8.1 Crucial Contexts

The problem now is to find an adequate subset $\mathcal{C}$ of $\mathcal{C}_\Sigma^{Obs}$, and if possible a finite subset $\mathcal{C}$. Remember that observable contexts represent experiments with observable results. A typical (simple) experiment will start by some computations involving mainly non observable values and providing non observable results, then there will be a computation providing an observable result, possibly followed by more computations over observable values (more complex experiments will be arbitrary combinations of these simple ones). The crucial idea is that intuitively only the step going from non observable values to observable ones is critical. Hence our intuition suggests that, in addition to the trivial observable contexts, it could be enough to consider the "crucial contexts" of the form $f(x_1, \ldots, x_{k-1}, z_{s_k}, x_{k+1}, \ldots, x_n)$, with $f \in F$ of arity $s_1 \ldots s_{k-1}\ s_k\ s_{k+1} \ldots s_n \rightarrow s$, $s \in Obs$ and $s_k \in S \setminus Obs$. If the contextual equality induced by these crucial contexts is a Σ-congruence (for the algebras in the class $\mathbf{C}$ we are interested in), then using Proposition 41 we know that it is fine to replace the infinite set of all observable contexts $\mathcal{C}_\Sigma^{Obs}$ by the finite set of the "crucial contexts".

It should be clear that the selection of the "crucial contexts" is mainly a starting point. As we will see in the examples described in the next subsections, in some cases the set of the "crucial contexts" is not optimal (i.e. an even smaller set of contexts is adequate), while in other cases the set of the "crucial contexts" is not adequate (i.e. we have to select a larger, infinite set of contexts). However, from our experience, the set of the "crucial contexts" is always a very useful starting point and when not adequate it provides nevertheless the "right intuition" about which set of contexts $\mathcal{C}$ must be selected.

8.2 The CONTAINER Example

Let us consider again the CONTAINER specification introduced in Examples 1 and 2. The observable sorts are {Elem, Nat, Bool} and all sorts are input sorts (i.e. we consider the total observational equality).

We are interested in proving behavioural properties of the CONTAINER specification. But before note that we can apply some obvious simplifications. First, the observational equality we consider is a total one. Moreover, on observable sorts the observational equality coincides with the set-theoretic equality, hence the predicate symbols $\sim_s$ are not necessary when s is an observable sort. This means that we can considerably simplify the lifting of formulas, the axiomatizations of the observational equality and the induced formulas BEH by using just one predicate symbol $\sim_{\texttt{Cont}}$ (for sake of clarity we abbreviate the sort Container by Cont).

Here the set of the "crucial contexts" is {e $\in z_{\texttt{Cont}}$, card($z_{\texttt{Cont}}$), subset(S,$z_{\texttt{Cont}}$), subset($z_{\texttt{Cont}}$,S), $z_{\texttt{Bool}}$, $z_{\texttt{Nat}}$, $z_{\texttt{Elem}}$}. However, intuitively here the "crucial context" e $\in z_{\texttt{Cont}}$ is enough to observe containers, and the other observing operations (card and subset) do not observe "more" than the operation $\in$ does. Hence we will select the set $\mathcal{C}$ = {e $\in z_{\texttt{Cont}}$, $z_{\texttt{Bool}}$, $z_{\texttt{Nat}}$, $z_{\texttt{Elem}}$} of observable contexts.

Since we consider the total observational equality, this leads directly to the following formula $TBeh_{\texttt{Cont}}$(S,S') (cf. Corollary 42):
$TBeh_{\texttt{Cont}}$(S,S') $\stackrel{\text{def}}{=}$ [∀e:Elem. e ∈ S = e ∈ S'].
The corresponding axiomatization is the sentence $\text{BEH}_{\texttt{Cont}}$:
∀S,S':Container. ([∀e:Elem. e ∈ S = e ∈ S'] ⇔ S $\sim_{\texttt{Cont}}$ S').

The formula $\text{CONG}^{\sim}_{\Sigma}$ is (after some obvious simplifications):

```
∀e:Elem, S,S',S1,S1':Container.
     [(S ~Cont S') ∧ (S1 ~Cont S1') ⇒
            insert(e,S) ~Cont insert(e,S')  ∧
            (S ∪ S1) ~Cont (S' ∪ S1')       ∧
            remove(e,S) ~Cont remove(e,S')  ∧
            card(S) = card(S')              ∧
            subset(S,S1) = subset(S',S1') ]
```

Let CONTAINER$^{\sim}$ $\stackrel{\text{def}}{=}$ (CONTAINER + $\langle Sig(\texttt{CONTAINER}) \cup \{\sim_{\texttt{Cont}}\}, \text{BEH}_{\texttt{Cont}}\rangle$).

In a first step we must check that CONTAINER$^{\sim}$ $\models \text{CONG}^{\sim}_{\Sigma}$, which is not difficult.

It is then very easy to prove that:
CONTAINER$^{\sim}$ $\models$ insert(e,insert(e,S)) $\sim_{\texttt{Cont}}$ insert(e,S)
and that CONTAINER$^{\sim}$ $\models$ insert(e,insert(e',S)) $\sim_{\texttt{Cont}}$ insert(e',insert(e,S))
which means that the two corresponding equations are behaviourally valid in the model class of the CONTAINER specification. This means as well that this specification can be considered as a correct behavioural implementation of sets.

8.3 The STACK Example

Let us consider the following STACK specification.

```
spec: STACK
  use: ELEM
  sort: Stack
  generated by:
     empty : → Stack
     push : Elem Stack → Stack
  operations:
     pop : Stack → Stack
     top : Stack → Elem
  axioms:
     ∀S:Stack, e:Elem.
     pop(empty) = empty
     pop(push(e,S)) = S
     top(push(e,S)) = e
end STACK.
```

We assume that all sorts are input sorts (i.e. we consider the total observational equality) and that the sort `Elem` is observable. We are interested in proving behavioural properties of stacks. Similarly to the `CONTAINER` example, we can apply obvious simplifications and use just one predicate symbol $\sim_{\texttt{Stack}}$.

Here the set of the "crucial contexts" is reduced to $\{\texttt{top}(z_{\texttt{Stack}}),\ z_{\texttt{Elem}}\}$. It is clear that the context $\texttt{top}(z_{\texttt{Stack}})$ is not enough to observe stacks: two stacks may have the same top element without being observationally equal. Intuitively two stacks must be considered as observationally equal if the elements stored in the stacks are the same (and they are stored in the same order). This can be checked by looking first to the top element, then pop the stacks and look again to the top element, etc. until both stacks are (simultaneously) empty. Hence we will select the set $\mathcal{C} = \{\texttt{top}(\texttt{pop}^n(z_{\texttt{Stack}})) \mid n \in \mathbb{N}\} \cup \{z_{\texttt{Elem}}\}$.

Since we select an infinite set of observable contexts, we must find an adequate hidden part HID and a set of contexts $\mathcal{C}_H$ to be used instead of $\mathcal{C}$ itself. But the selected set $\mathcal{C}$ of observable contexts is simple enough to suggest the following solution. Let HID be the specification:

$\text{HID} \stackrel{\text{def}}{=}$ `NATP` $+ \langle Sig(\texttt{STACK}) \cup Sig(\texttt{NATP}) \cup \{\texttt{topn : Nat Stack} \to \texttt{Elem}\},$
`{∀S:Stack, x:Nat.`
`topn(0,S) = top(S)`
`topn(s(x),S) = topn(x,pop(S))}⟩`

whereby it is assumed that `NATP` is a monomorphic specification of Peano's natural numbers having $\mathbb{N}$ as model. Let $\mathcal{C}_H \stackrel{\text{def}}{=} \{\texttt{topn(x,}z_{\texttt{Stack}}\texttt{)},\ z_{\texttt{Elem}}\}$.

Since we consider the total observational equality, we do not need any hidden predicate symbol, and we obtain as resulting formula $HBeh[\mathcal{C}_H]_{\texttt{Stack}}$(S,S'):
`[∀x:Nat. topn(x,S) = topn(x,S')]`.
The corresponding axiomatization is the sentence $\text{BEH}_{\texttt{Stack}}$:
`∀S,S':Stack. ([∀x:Nat. topn(x,S) = topn(x,S')] ⇔ S ~Stack S')`.

The formula $\text{CONG}^{\sim}_{\Sigma}$ is (after some obvious simplifications):

```
∀e:Elem, S,S':Stack.
    [(S ~Stack S') ⇒
          push(e,S) ~Stack push(e,S')  ∧
          pop(S) ~Stack pop(S')        ∧
          top(S) = top(S') ]
```

Now let `STACK-IMPL` be a specification (of signature $Sig(\texttt{STACK})$) which is supposed to describe a concrete implementation of stacks (for instance by means of arrays and pointers). Let $\texttt{STACK-IMPL-H}^{\sim} \stackrel{\text{def}}{=} (\texttt{STACK-IMPL} + \text{HID} + \langle Sig(\text{HID}) \cup \{\sim_{\texttt{Stack}}\}, \text{BEH}_{\texttt{Stack}}\rangle)$.

Obviously condition (3) of Theorem 46 holds (independently of the choice of the specification `STACK-IMPL`), since to any context $\texttt{topn(x,}z_{\texttt{Stack}}\texttt{)}$ and any valuation $\alpha : \{\texttt{x}\} \to \mathbb{N}$ corresponds the context $\texttt{top}(\texttt{pop}^{\alpha(\texttt{x})}(z_{\texttt{Stack}}))$ (and reciprocally).

Hence, if HID is a conservative extension of the STACK-IMPL specification and if moreover STACK-IMPL-H$^\sim \models \mathrm{CONG}^\sim_\Sigma$, we can apply Theorem 46, and as a consequence we know that, for any formula ϕ:
STACK-IMPL $\models_{\approx_{Obs,S}} \phi$ if and only if STACK-IMPL-H$^\sim \models \mathcal{L}(\phi)$.

This means that for checking that the specification STACK-IMPL is indeed a behavioural implementation of STACK, it is then enough to show that, for all STACK axioms ϕ, STACK-IMPL-H$^\sim \models \mathcal{L}(\phi)$.

8.4 The LIST Example

As a last example we will consider a classical implementation of sets by non redundant lists. Let us first consider the following LIST specification:

spec: LIST
 use: ELEM, BOOL
 sort: List
 generated by:
  empty : → List
  cons : Elem List → List
 operations:
  head : List → Elem
  tail : List → List
  _ ∈ _ : Elem List → Bool
  insert : Elem List → List
  remove : Elem List → List
 axioms:
  ∀L,L':List, e,e':Elem.
  head(cons(e,L)) = e
  tail(empty) = empty
  tail(cons(e,L)) = L
  e ∈ empty = false
  [e ∈ cons(e',L) = true] ⇔ [(e = e') ∨ (e ∈ L = true)]
  insert(e,empty) = cons(e,empty)
  [e ∈ L = true] ⇒ insert(e,L) = L
  [e ∈ L = false] ⇒ insert(e,L) = cons(e,L)
  remove(e,empty) = empty
  remove(e,cons(e,L)) = L
  e ≠ e' ⇒ remove(e,cons(e',L)) = cons(e',remove(e,L))
end LIST.

First note that remove just removes the first occurrence of an element. Hence it is clear that LIST $\not\models$ e ∈ remove(e,L) = false.

We are now interested in studying the behaviour of lists if we forget the operations `cons, head` and `tail`. (Indeed, to use lists for an implementation of sets, we must first forget these operations and in addition perform an appropriate renaming, but we will not take this renaming into account here for sake of simplicity.) Formally, we consider the signature Σ-`Set` defined by Σ−`Set` $\stackrel{\text{def}}{=} Sig(\texttt{LIST}) \setminus \{\texttt{cons}, \texttt{head}, \texttt{tail}\}$, and we consider the specification `LIST-RESTR` defined by `LIST-RESTR` $\stackrel{\text{def}}{=}$ **restrict** `LIST` **to** Σ-`Set`. `LIST-RESTR` has exactly Σ-`Set` as signature, and its model class is defined (as usual) by $Mod(\texttt{LIST-RESTR}) \stackrel{\text{def}}{=} \{A|_{\Sigma\texttt{-Set}} \mid A \in Mod(\texttt{LIST})\}$.

The behavioural theory of `LIST-RESTR` will now be studied w.r.t. the partial observational equality $\approx_{Obs,Obs}$ induced on Σ-`Set`-algebras by $Obs = \{\texttt{Elem, Bool}\}$. This observational equality is a family of partial Σ-`Set`-congruences. Indeed, since our aim is to prove that `LIST-RESTR` is a correct behavioural implementation of `SET`, it is important not to choose `List` as an input sort, since we must only consider lists built with `empty, insert` and `remove` for the behavioural satisfaction of the `SET` axioms (only these lists actually represent sets).

The set of the "crucial contexts" is $\mathcal{C} = \{\texttt{e} \in z_{\texttt{List}}, z_{\texttt{Bool}}, z_{\texttt{Nat}}, z_{\texttt{Elem}}\}$, and looks fine enough to be selected.

Since we consider a partial observational equality $\approx_{Obs,Obs}$, we must find an adequate hidden part HID with a predicate symbol $D_{H,\texttt{List}}$. (Note that as for the previous examples we apply obvious simplifications and use just one predicate symbol $\sim_{\texttt{List}}$ in the lifting; similarly we need predicate symbols $D_{H,s}$ for non input sorts s only, hence here only for the sort `List`.) Since the selected set of contexts is finite there is no need for contexts built with auxiliary hidden function symbols.

The rôle of the predicate symbol $D_{H,\texttt{List}}$ is to denote the lists built with the Σ-`Set` operations (and observable values), i.e. the lists built with `empty`, `insert` and `remove` (and not with `cons` or `tail`). Intuitively it is clear that these lists are exactly the lists with no duplicates. Hence it is quite obvious that the following hidden part HID should be adequate:

HID $\stackrel{\text{def}}{=} \langle \Sigma\texttt{−Set} \cup \{\texttt{nodup : List}\},$
$\{\forall \texttt{L:List. nodup(L)} \Leftrightarrow \texttt{(}\forall \texttt{e:Elem. e} \in \texttt{remove(e,L) = false)}\}\rangle.$

We obtain as resulting formula $HBeh[\texttt{nodup}, \mathcal{C}]_{\texttt{List}}\texttt{(L,L')}$:
`[nodup(L)` $\wedge$ `nodup(L')` $\wedge$ `(`$\forall$`e:Elem. e` $\in$ `L = e` $\in$ `L')].`

The corresponding axiomatization is the sentence $\text{BEH}_{\texttt{List}}$:
$\forall$`L,L':List.`
`([nodup(L)` $\wedge$ `nodup(L')` $\wedge$ `(`$\forall$`e:Elem. e` $\in$ `L = e` $\in$ `L')]` $\Leftrightarrow$ `L` $\sim_{\texttt{List}}$ `L').`

In particular we can infer: $\forall$`L:List. nodup(L)` $\Leftrightarrow$ `L` $\sim_{\texttt{List}}$ `L.`

The formula $\mathrm{CONG}^{\sim}_{\Sigma\text{-}\mathtt{Set}}$ is (after some obvious simplifications):

```
∀e:Elem, L,L':List.
     [(L ∼List L') ⇒
            insert(e,L) ∼List insert(e,L')  ∧
            remove(e,L) ∼List remove(e,L') ]
```

Let $\mathtt{LIST\text{-}RESTR\text{-}H}^{\sim} \stackrel{\text{def}}{=} (\mathtt{LIST\text{-}RESTR} + \mathrm{HID} + \langle Sig(\mathrm{HID}) \cup \{\sim_{\mathtt{List}}\}, \mathrm{BEH}_{\mathtt{List}}\rangle)$
and let $\mathtt{LIST\text{-}H}^{\sim} \stackrel{\text{def}}{=} (\mathtt{LIST} + \mathrm{HID} + \langle Sig(\mathrm{HID}) \cup \{\sim_{\mathtt{List}}\}, \mathrm{BEH}_{\mathtt{List}}\rangle)$.

It is obvious that HID is a conservative extension of the LIST-RESTR specification (indeed it is also a conservative extension of the LIST specification). To check that $\mathtt{LIST\text{-}RESTR\text{-}H}^{\sim} \models \mathrm{CONG}^{\sim}_{\Sigma\text{-}\mathtt{Set}}$, it is equivalent to check that $\mathtt{LIST\text{-}H}^{\sim} \models \mathrm{CONG}^{\sim}_{\Sigma\text{-}\mathtt{Set}}$, which is not difficult. Then we must check that the condition (3) of Theorem 46 holds, i.e. that the condition (i) of Proposition 45 is satisfied. This follows from an easy proof by induction w.r.t. $T_{\Sigma\text{-}\mathtt{Set}}(X_{\{\mathtt{Elem,Bool}\}})_{\mathtt{List}}$.

Hence we can apply Theorem 46, and as a consequence we know that, for any Σ-Set-formula ϕ, $\mathtt{LIST\text{-}RESTR} \models_{\approx_{Obs,Obs}} \phi$ if and only if $\mathtt{LIST\text{-}RESTR\text{-}H}^{\sim} \models \mathcal{L}(\phi)$ if and only if $\mathtt{LIST\text{-}H}^{\sim} \models \mathcal{L}(\phi)$.

For instance, to prove that:
$\mathtt{LIST\text{-}RESTR} \models_{\approx_{Obs,Obs}} \mathtt{insert(e,insert(e',L))} = \mathtt{insert(e',insert(e,L))}$
and that $\mathtt{LIST\text{-}RESTR} \models_{\approx_{Obs,Obs}} \mathtt{e} \in \mathtt{remove(e,L)} = \mathtt{false}$
it is equivalent to prove that (remember that nodup(L) is equivalent to $\mathtt{L} \sim_{\mathtt{List}} \mathtt{L}$):
$\mathtt{LIST\text{-}H}^{\sim} \models \mathtt{nodup(L)} \Rightarrow \mathtt{insert(e,insert(e',L))} \sim_{\mathtt{List}} \mathtt{insert(e',insert(e,L))}$
and that $\mathtt{LIST\text{-}H}^{\sim} \models \mathtt{nodup(L)} \Rightarrow \mathtt{e} \in \mathtt{remove(e,L)} = \mathtt{false}$
which is not difficult. This indeed is the crucial step required to conclude that the LIST specification can be considered as a correct behavioural implementation of sets.

9 Conclusion

In the literature several approaches formalize behavioural correctness concepts by introducing some kind of behavioural semantics (cf. e.g. [9], [17], [15], [18], [2], [16]). The main drawback of these approaches is that they either do not provide a proof-theoretical framework or suggest technically complicated proof techniques which are only of limited interest for practical applications (cf. the context induction principle in [10] or the correspondence relation in [20]). In this paper we have developed a proof theoretical framework for checking the behavioural validity of arbitrary Σ-formulas (finitary or not). We have shown that in the case of partial observational equalities this framework leads to a method that allows us to prove observational theorems using any arbitrary theorem prover for standard first-order logic. For concrete examples we have successfully proved observational theorems with the Larch Prover LP (cf. [8]).

The most important application of our proof technique is the verification of the correctness of behavioural implementations. Thereby a specification *SP-I* is called a be-

havioural implementation of a given basic specification SP w.r.t. a set Obs of observable sorts and a set In of input sorts if the models of SP-I (restricted to the signature of SP) satisfy w.r.t. the observational equality $\approx_{Obs,In}$ all axioms of SP. In [5] we have extended this notion of behavioural implementation to structured specifications and we have investigated proof rules that allow us to establish the correctness of behavioural implementations of structured specifications in a modular way. Since we know that behavioural semantics is in many cases the same but (at most) more restrictive than abstractor semantics (for "factorizable" abstractors) these proof rules are also correct for abstractor implementations in the sense of [19]. As a consequence of our results it is an objective of future work to built an environment for proving behavioural theorems and behavioural implementations on top of some existing theorem prover such as e.g. the Larch Prover.

Acknowledgements: This work is partially supported by the French-German cooperation programme PROCOPE and by the ESPRIT Working Group COMPASS II. We would like to thank for stimulating discussions and helpful comments Hubert Baumeister, Fernando Orejas, Don Sannella, Andrzej Tarlecki and Martin Wirsing. Preliminary results of parts of this work were developed in [3] and [4].

References

[1] B. Bauer and R. Hennicker. Proving the correctness of algebraic implementations by the ISAR system. In *Proc. of DISCO'93*, pages 2–16. Springer-Verlag L.N.C.S. 722, 1993.

[2] G. Bernot and M. Bidoit. Proving the correctness of algebraically specified software: modularity and observability issues. In *Proc. of AMAST'91*, pages 216–242. Springer-Verlag Workshops in Computing Series, 1992.

[3] M. Bidoit and R. Hennicker. Proving behavioural theorems with standard first-order logic. In *Proc. of ALP'94*, pages 41–58. Springer-Verlag L.N.C.S. 850, 1994.

[4] M. Bidoit and R. Hennicker. Behavioural theories. In *Recent Trends in Data Type Specification*. Springer-Verlag L.N.C.S. 906, 1995. To appear.

[5] M. Bidoit and R. Hennicker. Proving the correctness of behavioural implementations. In *Proc. of AMAST'95*. Springer-Verlag L.N.C.S., 1995. To appear.

[6] M. Bidoit, R. Hennicker, and M. Wirsing. Behavioural and abstractor specifications. Technical Report LIENS–94–10 *(To appear in Science of Computer Programming)*, 1994. A short version appeared as: Characterizing behavioural semantics and abstractor semantics, in *Proc. of ESOP'94*, Springer-Verlag L.N.C.S. 788, pages 105–119, 1994.

[7] H. Ehrig and B. Mahr. *Fundamentals of algebraic specification 1. Equations and initial semantics*, volume 6 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1985.

[8] S. Garland and J. Guttag. An overview of LP, the Larch Prover. In *Proc. of the Third International Conference on Rewriting Techniques and Applications*, pages 137–151. Springer-Verlag L.N.C.S. 355, 1989.

[9] J. Goguen and J. Meseguer. Universal realization, persistent interconnection and implementation of abstract modules. In *Proc. of 9th ICALP*, pages 265–281. Springer-Verlag L.N.C.S. 140, 1982.

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

[11] R. Hennicker and M. Wirsing. *Behavioural specifications.* Proof and Computation, International Summer School Marktoberdorf 1993. Springer-Verlag, 1995.

[12] M. Hofmann and D. Sannella. On behavioural abstraction and behavioural satisfaction in higher-order logic. In *Proc. of TAPSOFT'95.* Springer-Verlag L.N.C.S., 1995. To appear.

[13] G. Kreisel and J.L. Krivine. *Eléments de Logique Mathématique.* Dunod (Paris), 1967.

[14] R. Milner. Fully abstract models of typed λ-calculi. *Theoretical Computer Science*, 4:1–22, 1977.

[15] P. Nivela and F. Orejas. Initial behaviour semantics for algebraic specification. In *Recent Trends in Data Type Specification*, pages 184–207. Springer-Verlag L.N.C.S. 332, 1988.

[16] F. Orejas, M. Navarro, and A. Sànches. Implementation and behavioural equivalence: A survey. In *Recent Trends in Data Type Specification*, pages 93–125. Springer-Verlag L.N.C.S. 655, 1993.

[17] H. Reichel. Initial restrictions of behaviour. In *Proc. of IFIP Working Conference, The Role of Abstract Models in Information Processing*, 1985.

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

[19] D. Sannella and A. Tarlecki. Toward formal development of programs from algebraic specification: implementation revisited. *Acta Informatica*, 25:233–281, 1988.

[20] O. Schoett. Behavioural correctness of data representation. *Science of Computer Programming*, 14:43–57, 1990.

[21] J.W. Thatcher, E.G. Wagner, and J.B. Wright. Data type specification: parameterization and the power of specification techniques. *ACM Transactions on Programming Languages and Systems*, 4:711–732, 1982.

[22] M. Wirsing. *Algebraic specification.* Handbook of Theoretical Computer Science. Elsevier Science Publishers B. V., 1990.

[23] M. Wirsing. *Structured specifications: syntax, semantics and proof calculus.* Logic and Algebra of Specification, International Summer School Marktoberdorf 1991. Springer-Verlag, 1993.