A higher-order behavioural algebraic institution for ASL

Nikos Mylonakis
LSI Department
Universitat Politècnica de Catalunya

March 20, 2000

Abstract

In this paper, we generalise the semantics of ASL including the three behavioural operators presented in [3] for a fixed but arbitrary algebraic institution. After that, we define a behavioural algebraic institution which is used to give an alternative semantics of the behavioural operators, to define the normal forms of the both semantics of behavioural operators and to relate both semantics. Finally, we present a higher-order behavioural algebraic institution.

1 Introduction

ASL is a specification language which was originally defined by a set of specification operators which determined the set of specification expressions of the language. The most important operators of the original definition were operators to build structured specifications from smaller specifications or to make some modifications from a given specification, like for example the renaming of a given specification. This language was not originally designed to be used directly but as a basis to define the semantics of higher level specification languages. Two specification languages which used ASL to define their semantics were EML and PLUSS. A specific kind of operator which appeared in ASL was an operator which was used to *behaviourally abstract* a given specification closing its model-theoretical semantics by an equivalence relation between algebras. Later on, different operators related to the one just described were developed. We will refer to these operators as behavioural operators.

In this paper we give a general semantic framework for behavioural operators. In a general setting, these operators are parameterized by fixed but arbitrary equivalence relations. There have been defined three different kinds of behavioural operators in ASL . We will refer to them as the **abstract** operator, the **behaviour** operator and the **quotient** operator. All of them have a specification as an argument and they transform the model-theoretic semantics of the argument specification.

Intuitively, the **abstract** operator extends the class of models of the argument specification with those models which are equivalent (by an equivalence relation between models) to some model belonging to the model-theoretical semantics of the argument specification.

The class of models of the **behaviour** operator is defined by those models whose behaviour (denoted also as a model and defined via a congruence relation on values within a model) belongs to the class of models of the argument specification of the operator.

Finally, the class of models of the **quotient** operator is defined by the closure under isomorphism of the quotient of the models associated to the semantics of the argument specification of the operator.

A formal semantics of these operators is given in [3] by defining their signature and their model-theoretical semantics. They use a first-order logic with equality to define the sentences of specifications. Apart from giving the semantics of the operators, a theory which establishes different equivalences between the semantics of these operators is presented.

In [7], an alternative semantics for the behavioural operators is given using just flat specifications as argument specifications. They use higher-order logic as specification logic and a similar theory as in [3] to relate the semantics of the behaviour and abstract operator is developed.

Since both semantics are quite independent of the specification logic of the specification language, it seems reasonable to make a generalisation of the semantics of these operators for an arbitrary but fixed institution. These institutions have to satisfy specific properties in order to include these operators in a version of ASL with structuring operators and they are a restricted version of semiexact institutions presented in [9]. We refer to these institutions as algebraic institutions ($AINS$) and the two main restrictions are that the category of signatures is the category of first-order relational signatures and the model functor of these institutions assigns to every first-order relational signature Σ the category of Σ-algebras which we will denote as $Alg(\Sigma)$ instead of an arbitrary category of models $Mod(\Sigma)$. This is necessary because these institutions are used to define the semantics of different set of ASL operators including behavioural operators which we will denote as $BASLker$ languages. The signatures of the specification expressions of $BASLker$ languages are first-order signatures since the semantics of some of the behavioural operators are defined using a fixed but arbitrary partial Σ-congruence and therefore using the internal structure of first-order signatures. These languages also include the common operators of another set of operators of ASL which we will denote as $ASLker$ languages. These common operators are base specifications (with syntax $< \Sigma, \phi >$), a sum operator to define structured specifications and an export operator. These restrictions are not needed to define the semantics of the common operators of $ASLker$ languages and see [2] for the semantics of these operators in a fixed but arbitrary semiexact institution. In [2], the semantics of the behavioural operators of $BASLker$ languages is given just for an institution of infinitary first-order logic and for concrete observational equivalences. See also [1] for an abstract categorical framework to relate the semantics of the behavioural operators which is not required for our purposes.

In order to define a certain kind of proof systems for the deduction of sentences from $ASLker$ languages, it is required additionally a normalisation function on specification expressions where the normal forms of specifications are defined in terms of the export operator (with syntax $< \Sigma', \phi > |_{\Sigma}$). This normalisation function is also useful to relate the semantics of [3] and [7]. We call any set of operators defined with a normalisation function and including at least the common operators of $ASLker$ languages as $ASLnf$ language. To generalise the semantics of [3] and [7], we define a new institution which we will refer as behavioural algebraic institution ($BAINS$) which incorporates additional components to a fixed but arbitrary algebraic institution ($AINS$) in order to define the semantics of the behaviour operator of [7] and the normalisation function of the behavioural operators with the semantics of [3].

The structure of the paper is as follows: first we introduce the abstract concept of algebraic institution and then we present a concrete higher-order algebraic institution. Then, we give the semantics of the behavioural operators and how to relate them in an arbitrary but fixed algebraic institution following the ideas of [3] and [2]. Next, we present behavioural algebraic institutions, a normalisation function for the behavioural operators presented previously and a relationship between the semantics of [3] and [7]. Finally, we present concrete equivalence relations and a concrete behavioural algebraic institution using the concrete equivalences and the higher-order algebraic institution presented in the first section.

2 Semiexact algebraic institutions

In this section, we will present the abstract semantic framework to define the semantics of different operators of ASL including the behavioural operators. We will assume predefined basic concepts of institutions, which can be found in [4], [5], or in [8].

Definition 2.1 *An algebraic institution ($AINS$) is an institution which consists of:*

- *The category of first-order relational signatures $AlgSig$ whose objects are first-order relational signatures and morphisms are signature morphisms.*
- *a functor $Sen_{AINS} : AlgSig \to Set$*
- *the functor $Alg : AlgSig^{op} \to Cat$ where:*
  - *for any $\Sigma \in |AlgSig|$, $Alg(\Sigma)$ is the category of Σ-algebras*
  - *for any morphism $\sigma : \Sigma \to \Sigma'$ in $AlgSig$, $Alg(\sigma)$ is the reduct functor $\_|_{\sigma} : Alg(\Sigma') \to Alg(\Sigma)$.*

- *for each* $\Sigma \in |AlgSig|$ *a satisfaction relation*

$$\models_{AINS,\Sigma}: |Alg(\Sigma)| \times Sen_{AINS}(\Sigma)$$

such that

  - *the satisfaction condition holds for any signature morphism* $\sigma : \Sigma \rightarrow \Sigma'$ *and for any formula* $\phi \in Sen_{AINS}(\Sigma)$*.This condition is formally defined as:*

$$\forall A \in |Alg(\Sigma')|.\ A \models_{AINS,\Sigma'} Sen_{AINS}(\sigma)(\phi) \Leftrightarrow A|_\sigma \models_{AINS,\Sigma} \phi$$

  - *an abstract satisfaction condition holds for any formula* ϕ *in* Sen_{AINS} (Σ)*:*

$$\forall A, B \in |Alg(\Sigma)|.\ A \cong B \ \Rightarrow\ (A \models_{AINS,\Sigma} \phi \Leftrightarrow B \models_{AINS,\Sigma} \phi)$$

Notation and comments:

- *The main differences between algebraic institutions and the original definition of institutions is that the category of signatures is not an arbitrary category but the category of first-order relational signatures and that it is added the abstract satisfaction condition which almost all institutions satisfy.*

- *We will normally refer to first-order relational signatures just as relational signatures. For any relational signature* $\Sigma = (S, Op, Pr) \in |AlgSig|$*, the functions* $Sorts(\Sigma)$*,* $Ops(\Sigma)$ *and* $Prs(\Sigma)$ *will return* S*,* Op *and* Pr *respectively.*

  For any relational signature $\Sigma = (S, Op, Pr)$*, if* $Pr = \emptyset$ *we will define it just by* $\Sigma = (S, Op)$ *and it will be normally referred just as signature.*

  For any relational signature $\Sigma = (S, Op, Pr) \in |AlgSig|$ *and for a fixed but arbitrary* S*-sorted infinite denumerable set of variables* X $T_\Sigma(X)$ *will denote the* Σ*-term algebra freely generated by* X *and* $P_\Sigma(X)$ *will denote the set of terms of the form* $p(t_1, \ldots, tn)$ *where* $p : s_1 \times \ldots \times s_n \in Prs(\Sigma)$ *and* $t_1 \in T_{\Sigma,s_1}(X), \ldots, t_n \in T_{\Sigma,s_n}(X)$

  For any algebra $A \in Alg_{AINS}(\Sigma)$ *and a* S*-sorted valuation* $\alpha : X \rightarrow A$*,* $I_\alpha : T_\Sigma(X) \rightarrow A$ *will denote the unique extension to a* Σ*-morphism of the valuation* α*, and for the case of* $p(t_1, \ldots, t_n) \in P_\Sigma(X)$*,* $I_\alpha(p(t_1, \ldots, t_n))$ *will hold if and only if* $(I_\alpha\ t_1, \ldots, I_\alpha\ t_n) \in p_A$ *. We will refer to them as the interpretations of terms and predicates associated to* α*.*

  We will assume predefined the function $\alpha \bigcup \{(x_1, v_1), \ldots, (x_n, v_n)\}$ *which given a valuation* $\alpha : X \rightarrow A$ *and a set of pairs of the form* $\{(x_1, v_1), \ldots, (x_n, v_n)\}$ *such that* $x_i \in X_{s_i}$ *and* $v_i \in A_{s_i}$ *for any* $i \in [1..n]$*, it will return the usual update of the valuation* α *with the given set of pairs.*

- *If* $\Sigma \subseteq \Sigma'$*, we will normally denote by* $\sigma : \Sigma \hookrightarrow \Sigma'$ *the obvious embedding morphism and we will normally refer to it as inclusion.*

- *Since it is well known that AlgSig has pushouts, the pushout object of any pair of morphisms* $\sigma : \Sigma_0 \to \Sigma_1$, $\sigma' : \Sigma_0 \to \Sigma_2$ *in AlgSig (where* $\Sigma_0, \Sigma_1, \Sigma_2 \in |AlgSig|$*) will be denoted in general as* $PO(\sigma : \Sigma_0 \to \Sigma', \sigma' : \Sigma_0 \to \Sigma'')$ *and if the pair of morphisms are both inclusions the pushout object will be normally denoted as* $\Sigma_1 +_{\Sigma_0} \Sigma_2$ *and the pushout morphisms as* $(inl : \Sigma_1 \to \Sigma_1 +_{\Sigma_0} \Sigma_2, inr : \Sigma_2 \to \Sigma_1 +_{\Sigma_0} \Sigma_2)$. *In this last case, we can assume in general that either inl or inr are inclusions but not both.*

- *We will also drop usually the subscript of the functor* Sen_{AINS} *and the subscripts of* $\models_{AINS,\Sigma}$ *if it can be inferred from the context.*

  We will also refer as

  $$\models_{AINS,\Sigma}: |Alg(\Sigma)| \times \mathcal{P}(Sen_{AINS}(\Sigma))$$

  the obvious extension of the satisfaction relation to a set of sentences.

- *For any signatures* $\Sigma, \Sigma' \in |AlgSig|$ *and for any signature morphism* $\sigma : \Sigma \to \Sigma'$*, the morphism*

  $$Sen_{AINS}(\sigma) : Sen_{AINS}(\Sigma) \to Sen_{AINS}(\Sigma')$$

  will be normally denoted just by $\sigma : Sen_{AINS}(\Sigma) \to Sen_{AINS}(\Sigma')$.

Definition 2.2 *An institution* $INS = (Sign_{INS}, Sen_{INS} : Sign_{INS} \to Set, Mod_{INS} : Sign^{op}_{INS} \to Cat, <\models_{INS,\Sigma}>_{\Sigma \in Sign_{INS}})$ *is semiexact if for any pushout in* $Sign_{INS}$ $(inl : \Sigma_1 \to \Sigma', inr : \Sigma_2 \to \Sigma')$ *of any pair of morphisms* $(\sigma : \Sigma_0 \to \Sigma_1, \sigma' : \Sigma_0 \to \Sigma_2)$ *and for any models* $M_1 \in Mod_{INS}(\Sigma_1), M_2 \in Mod_{INS}(\Sigma_2)$ *such that* $M_1|_\sigma = M_2|_{\sigma'}$ *there exists an unique model* $M \in Mod_{INS}(\Sigma')$ *such that* $M|_{inl} = M_1$ *and* $M|_{inr} = M2$.

Notation: *This definition is equivalent to the definition of semiexact institution presented in [4].*

Proposition 2.3 *Any fixed but arbitrary algebraic institution AINS is semiexact.*

Now we present a concrete higher-order algebraic institution HOL. Before presenting it, we give some basic definitions which will be used in its definition.

Definition 2.4 *For each* $\Sigma = (S, Op, Pr) \in |AlgSig|$*, the set* $Types_{HOL}(\Sigma)$ *is inductively defined by the following set of rules:*

- *If* $s \in S$ *then* $s \in Types_{HOL}(\Sigma)$.

- *If* $\tau_1 \in Types_{HOL}(\Sigma), \ldots, \tau_n \in Types_{HOL}(\Sigma)$ *and* $n \geq 0$ *then* $[\tau_1, \ldots, \tau_n] \in Types_{HOL}(\Sigma)$.

Notation: *The type* $[]$ *will be normally denoted by* **Prop**.

For any signature morphism $\sigma : \Sigma \to \Sigma'$, *we will also denote by* σ *the usual renaming function between types* $\sigma : Types_{HOL}(\Sigma) \to Types_{HOL}(\Sigma')$.

Definition 2.5 *The semantic function* $[\![\tau]\!]_A$ *is inductively defined for any type* $\tau \in Types_{HOL}(\Sigma)$ *and for any* Σ*-algebra* A *as follows:*

$$[\![s]\!]_A = A_s$$

$$[\![[\tau_1, \ldots, \tau_n]]\!]_A = \mathcal{P}([\![\tau_1]\!]_A \times \ldots \times [\![\tau_n]\!]_A)$$

Notation: *The semantics of* **Prop** *is a set of two elements: the empty set and the set with the empty tuple. These two elements will be denoted as* **ff** *and* **tt** *respectively.*

Definition 2.6 *The set* $Sen_{HOL}(\Sigma, X_{HOL}, \tau)$ *for a fixed but arbitrary* $Types_{HOL}$ (Σ)*-sorted infinite denumerable set of variables* X_{HOL} *and for every* $\tau \in Types_{HOL}(\Sigma)$ *is inductively defined by the following set of rules:*

- *If* $x \in X_{HOL,\tau}$ *then* $x_\tau \in Sen_{HOL}(\Sigma, X_{HOL}, \tau)$.

- *If* $f : s_1 \times \ldots \times s_n \to s \in Ops(\Sigma)$, $t_1 \in T_{\Sigma,s_1}(< X_{HOL,s} >_{s \in S}), \ldots,$

  $t_n \in T_{\Sigma,s_n}(< X_{HOL,s} >_{s \in S})$

  then $f(t_1, \ldots, t_n) \in Sen_{HOL}(\Sigma, X_{HOL}, s)$.

- *If* $p : s_1 \times \ldots \times s_n \in Prs(\Sigma)$, $t_1 \in T_{\Sigma,s_1}(< X_{HOL,s} >_{s \in S}), \ldots,$

  $t_n \in T_{\Sigma,s_n}(< X_{HOL,s} >_{s \in S})$

  then $p(t_1, \ldots, t_n) \in Sen_{HOL}(\Sigma, X_{HOL}, \mathbf{Prop})$

- *If* $\tau_1, \ldots, \tau_n \in Types_{HOL}(\Sigma)$, $x_1 \in X_{HOL,\tau_1}, \ldots, x_n \in X_{HOL,\tau_n}$

  and $\phi \in Sen_{HOL}(\Sigma, X_{HOL}, \mathbf{Prop})$ *then*

  $\lambda(x_1 : \tau_1, \ldots, x_n : \tau_n).\phi \in Sen_{HOL}(\Sigma, X_{HOL}, [\tau_1, \ldots, \tau_n])$.

- *if* $\tau_1, \ldots, \tau_n \in Types_{HOL}(\Sigma)$, $t \in Sen_{HOL}(\Sigma, X_{HOL}, [\tau_1, \ldots, \tau_n])$,

  $t_1 \in Sen_{HOL}(\Sigma, X_{HOL}, \tau_1), \ldots, t_n \in Sen_{HOL}(\Sigma, X_{HOL}, \tau_n)$

then $t(t_1, \ldots, t_n) \in Sen_{HOL}(\Sigma, X_{HOL}, \mathbf{Prop})$.

- *if* $\tau \in Types_{HOL}(\Sigma)$, $x \in X_{HOL,\tau}$ *and*

  $\phi \in Sen_{HOL}(\Sigma, X_{HOL}, \mathbf{Prop})$ *then*

  $\forall x : \tau.\phi \in Sen_{HOL}(\Sigma, X_{HOL}, \mathbf{Prop})$.

- *if* $\phi, \phi' \in Sen_{HOL}(\Sigma, X, \mathbf{Prop})$ *then* $\phi \supset \phi' \in Sen_{HOL}(\Sigma, X, \mathbf{Prop})$.

Notation: *We will denote by* $Terms_{HOL}(\Sigma, X_{HOL})$ *the set*

$$\bigcup_{\tau \in Types_{HOL}(\Sigma)} Sen_{HOL}(\Sigma, X_{HOL}, \tau)$$

For any signature morphism $\sigma : \Sigma \rightarrow \Sigma'$, *we will also denote by* σ *the usual renaming function between terms*

$$\sigma : Terms_{HOL}(\Sigma, X_{HOL}) \rightarrow Terms_{HOL}(\Sigma', X_{HOL})$$

such that for any type $\tau \in Types_{HOL}(\Sigma)$, *for any higher-order sentence* $\phi \in Sen_{HOL}(\Sigma, X_{HOL}, \tau)$, $\sigma(\phi) \in Sen_{HOL}(\Sigma', X_{HOL}, \sigma(\tau))$.

The usual definition of $\beta-$*equality between terms in* $Terms_{HOL}(\Sigma, X_{HOL})$ *identifying also* α*-convertible terms will be denoted by* $=_\beta$ *and the usual substitution operation avoiding name clashes will be denoted by* $t\{t'/x\}$ *for any* $t \in Terms_{HOL}(\Sigma, X_{HOL})$, $t' \in Sen_{HOL}(\Sigma, X_{HOL}, \tau)$ *and* $x \in X_{HOL,\tau}$.

Definition 2.7 *The function* $[\![t]\!]_{\rho,A}$ *for any term* $t \in Terms_{HOL}(\Sigma, X_{HOL})$, *for any algebra* $A \in Alg(\Sigma)$, *for any* $Types_{HOL}(\Sigma)$*-sorted valuation* ρ *which for every* $\tau \in Types_{HOL}(\Sigma)$, ρ_τ *has arity* $\rho_\tau : X_{HOL,\tau} \rightarrow [\![\tau]\!]_A$ *is inductively* *defined by the structure of t as follows:*

$[\![x_\tau]\!]_{\rho,A} = \rho_\tau(x)$

$[\![f(t_1,\ldots,t_n)]\!]_{\rho,A} = f_A([\![t_1]\!]_{\rho,A},\ldots,[\![t_n]\!]_{\rho,A})$

$[\![p(t_1,\ldots,t_n)]\!]_{\rho,A} = if\ ([\![t_1]\!]_{\rho,A},\ldots,[\![t_n]\!]_{\rho,A}) \in p_A\ then\ \mathbf{tt}\ else\ \mathbf{ff}$

$[\![\lambda(x_1:\tau_1,\ldots,x_n:\tau_n).\phi]\!]_{\rho,A} =$

$$\{(v_1,\ldots,v_n)|v_1 \in [\![\tau_1]\!]_{\rho,A},\ldots,v_n \in [\![\tau_n]\!]_{\rho,A}, [\![\phi]\!]_{\rho\cup\{(x_1,v_1),\ldots,(x_n,v_n)\}} = \mathbf{tt}\}$$

$[\![\lambda(x_1:\tau_1,\ldots,x_n:\tau_n).\phi\ (t_1,\ldots,t_n)]\!]_{\rho,A} = [\![\phi]\!]_{\rho\cup\{(x_1,[\![t_1]\!]_{\rho,A}),\ldots,(x_n,[\![t_n]\!]_{\rho,A})\},A}$

$[\![\phi \supset \phi']\!]_{\rho,A} = if\ \ [\![\phi]\!]_{\rho,A}\ = \mathbf{tt}\ then\ \ [\![\phi]\!]_{\rho,A}\ else\ \ \mathbf{tt}$

$[\![\forall x:\tau.\phi]\!]_{\rho,A} = if\ \ \forall v \in [\![\tau]\!]_A.[\![\phi]\!]_{\rho\cup\{(x,v)\},A} = \mathbf{tt}\ \ then\ \mathbf{tt}\ \ else\ \mathbf{ff}$

Proposition 2.8 *The tuple*

$$HOL = (AlgSig, Sen_{HOL}, Alg, <\models_{HOL,\Sigma}>_{\Sigma\in|AlgSig|})$$

such that:

- $Sen_{HOL} : AlgSig \to Set$ *is a functor defined in the following way:*
  - *For each* $\Sigma \in |AlgSig|$, *the set* $Sen_{HOL}(\Sigma)$ *is defined as*

    $$Sen_{HOL}(\Sigma)\ =\ Sen_{HOL}(\Sigma, X_{HOL}, \mathbf{Prop})$$
  - *For each signature morphism* $\sigma : \Sigma \to \Sigma'$ *the morphism*

    $$Sen_{HOL}(\sigma) : Sen_{HOL}(\Sigma) \to Sen_{HOL}(\Sigma')$$

    is the usual renaming function between sentences using $\sigma : \Sigma \to \Sigma'$ *and it will also be denoted just by* σ.
- *for each* $\Sigma \in |AlgSig|$, *for all* $A \in |Alg(\Sigma)|$, *for all* $\phi \in Sen_{HOL}(\Sigma)$, *the satisfaction relation* $A \models_\Sigma \phi$ *holds if and only if for any* $Types_{HOL}(\Sigma)$*-sorted valuation* ρ *which for every* $\tau \in Types_{HOL}(\Sigma)$, ρ_τ *has arity* $X_{HOL,\tau} \to [\![\tau]\!]_A$, $[\![\phi]\!]_{\rho,A}\ =\ \mathbf{tt}$

is an algebraic institution.

Proof sketch:
We have to prove that Sen_{HOL} is a functor (which is straightforward) and we have to prove that the relation $<\models_{HOL,\Sigma}>_{\Sigma\in|AlgSig|}$ satisfies:

- the satisfaction condition which follows by induction on $Terms_{HOL}$ in a similar way as in the first-order case.
- the abstract satisfaction condition holds extending the isomorphism between algebras to higher-order types in the obvious way. See [7] for details of the proofs.

3 Abstract semantics of different operators of ASL

In this section, we will present the semantics of different operators of ASL including the behavioural operators presented in [3] as we briefly explained in the introduction.

Definition 3.1 *An ASLker specification language with a fixed but arbitrary algebraic institution AINS is a specification language defined with a set of operators including basic specifications,and export operator and an operator for structuring specifications which we will refer as the sum operator. The syntax of these operators is the following:*

$$SP_0 ::= \quad < \Sigma, \Phi >$$

$$SP_1|_{\Sigma}$$

$$SP_1 +_{\Sigma} SP_2$$

where the signature $\Sigma = (S, Op) \in |AlgSig|$ and $\Phi \subseteq Sen_{AINS}(\Sigma)$. Let ASLK be an ASLker specification language and let $SPEX(ASLK)$ be the set of specification expressions of this language. The semantics of an ASLker language ASLK is inductively defined by the functions $Signature : SPEX(ASLK) \rightarrow |AlgSig|$, $Symbols : SPEX(ASLK) \rightarrow |AlgSig|$ and $Models : SPEX(ASLK) \rightarrow Alg(Signature(SP))$.

The function Signature must return the signature with just the visible symbols of the given specification, whereas the function Symbols must return the signature with the visible and hidden symbols of the given specification. These functions must be inductively defined by specification expressions, and for the cases of the common operators the definition is as follows:

$$Signature(< \Sigma, \Phi >) = \Sigma$$

$$Symbols(< \Sigma, \Phi >) = \Sigma$$

$$Models(< \Sigma, \Phi >) = \{A \mid A \models_{AINS,\Sigma} \Phi\}$$

where the signature $\Sigma = (S, Op) \in |AlgSig|$ and $\Phi \subseteq |Sen_{AINS}(\Sigma)|$.

$Signature(SP|_\Sigma) = \Sigma$

$Symbols(SP|_\Sigma) = Symbols(SP)$

$Models(SP|_\Sigma) = \{A|_\Sigma \mid A \in Models(SP)\}$

where SP ranges over specification expressions, the signature $\Sigma = (S, Op) \in |AlgSig|$ and $\Sigma \subseteq Signature(SP)$

$Signature(SP_1 +_\Sigma SP_2) = Signature(SP_1) +_\Sigma Signature(SP_2)$

$Symbols(SP_1 +_\Sigma SP_2) = Symbols(SP_1) +_\Sigma Symbols(SP_2)$

$Models(SP_1 +_\Sigma SP_2) =$

$$\{A \mid A \in Alg(Signature(SP_1) +_\Sigma Signature(SP_2)),$$

$$A|_{inl} \in Models(SP_1), A|_{inr} \in Models(SP_2)\}$$

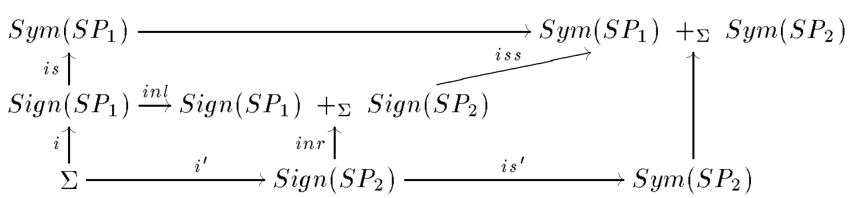
where the signature $\Sigma = (S, Op) \in |AlgSig|$, SP_1, SP_2 ranges over specification expressions in $SPEX(ASLK)$, $\Sigma \subseteq Signature(SP_1)$, $\Sigma \subseteq Signature(SP_2)$ and the pushouts

$$Signature(SP_1) +_\Sigma Signature(SP_2)$$

and

$$Symbols(SP_1) +_\Sigma Symbols(SP_2)$$

are the pushouts of the following diagram:



Since $Signature(SP_1) +_\Sigma Signature(SP_2)$ is a pushout iss is the unique morphism with arity

$$iss : Signature(SP_1) +_\Sigma Signature(SP_2) \hookrightarrow$$

$$Symbols(SP_1) +_\Sigma Symbols(SP_2)$$

and the pushouts can be chosen in such a way that iss is an inclusion.

Comment: The functions *Signature* and *Symbols* are needed to define different proof systems for *ASLker* languages. See [?] and [6] for the definition of these proof systems.

Definition 3.2 *Let ASL be an ASLker specification language and* $Symbols_{nf}$ *a function with arity* $Symbols_{nf} : SPEX(ASLN) \rightarrow |AlgSig|$ *. A function nf with arity* $nf : SPEX(ASLN) \rightarrow SPEX(ASLN)$ *is a normalisation function if the function nf and the function* $Symbols_{nf}$ *satisfy the following conditions which we will refer as normalisation conditions:*

- *For all* $SP \in SPEX(ASLN)$,

$$nf(SP) =< Symbols_{nf}(SP), \Phi > |_{Signature(SP)}$$

  for some $\Phi \subseteq |Sen_{AINS}(Symbols_{nf}(SP))|$.

- *For all* $SP \in SPEX(ASLN)$, $Symbols(SP) \subseteq Symbols_{nf}(SP)$.

- $A \in Models(nf(SP)) \Leftrightarrow A \in Models(SP)$

Comment: If the *ASLnf* language just contains the common operators of these languages, the function $Symbols_{nf}$ coincides with the functions *Symbols*, but this will not be the case for example for the common operators of *BASLnf* specification languages presented in later sections. The function $Symbols_{nf}$ is also needed to define certain kind of proof systems associated to the *ASLnf* specifications languages presented in next definition.

Definition 3.3 *An ASLnf specification language is an ASLker specification language whose semantic definition also requires the definition of a normalisation function nf together with the function* $Symbols_{nf}$*. The functions nf and* $Symbols_{nf}$ *must also be defined by induction on specification expressions and* *the definition for the common operators is as follows:*

$nf(<\Sigma, \Phi>) =<\Sigma, \Phi> |_{\Sigma}$

$Symbols_{nf}(<\Sigma, \Phi>) = \Sigma$

where the signature $\Sigma = (S, Op) \in |AlgSig|$ *and* $\Phi \subseteq |Sen_{AINS}(\Sigma)|$

$nf(SP_1 +_{\Sigma} SP_2) =<\Sigma'_1 +_{\Sigma} \Sigma'_2, inl(\Phi_1) \cup inr(\Phi_2)> |_{\Sigma_1 +_{\Sigma} \Sigma_2}$

$Symbols_{nf}(SP_1 +_{\Sigma} SP_2) = \Sigma'_1 +_{\Sigma} \Sigma'_2$

where $nf(SP_1) =<\Sigma'_1, \Phi_1> |_{\Sigma_1}$, $nf(SP_2) =<\Sigma'_2, \Phi_2> |_{\Sigma_2}$

and inl *and* inr *are fixed but arbitrary pushouts of*
$i_1 : \Sigma \hookrightarrow Signature(SP_1)$ *and* $i_2 : \Sigma \hookrightarrow Signature(SP_2)$

$nf(SP|_{\Sigma}) =<\Sigma', \Phi> |_{\Sigma}$

$Symbols_{nf}(SP) = \Sigma'$

where $nf(SP) =<\Sigma', \Phi> |_{\Sigma''}$.

Notation: In the following, ASL will range over an $ASLker$ or an $ASLnf$ specification language.

Proposition 3.4 *The* nf *function of the previous definition satisfies the normalisation conditions.*

Proof sketch:
The proof is by an easy induction on specification expressions.

Definition 3.5 *Let* ASL *be an* $ASLker$ *or an* $ASLnf$ *specification language with an arbitrary but fixed algebraic institution* $AINS$*. For any specification expression* $SP \in SPEX(ASL)$ *and for any sentence* $\phi \in Sen(Signature(SP))$ *the satisfaction relation* $\models_{AINS,Signature(SP)}$ *is defined as follows:*

$$SP \models_{AINS,Signature(SP)} \phi \Leftrightarrow \forall A \in Models(SP).A \models_{AINS,Signature(SP)} \phi$$

Definition 3.6 *Assume that* $\Sigma \in |AlgSig|$*. A partial* Σ*-congruence on a* Σ*-algebra* A *(normally denoted by* $\approx_A$*) is defined for every sort* s *in* S *as a relation which given any* Σ*-algebra* A*, returns a symmetric and transitive relation (normally denoted as* $\approx_{s,A}$*) with domain* $\subseteq s_A \times s_A$*.This relation is compatible with every operation* $f_A : s_{1A} \times \ldots \times s_{nA} \rightarrow s_A$ *where* $f \in Ops(\Sigma)$*. This means that* *for every* $v_1, w_1 \in s_{1A}$, ..., $v_n, w_n \in s_{nA}$, *if* $v_1 \approx_{s_1,A} w_1$, ... , $v_n \approx_{s_n,A} w_n$ *then* $f_A(v_1, \ldots, v_n) \approx_{s,A} f_A(w_1, \ldots, w_n)$*, and it is also compatible withe every predicate* $p_A : s_{1A} \times \ldots \times s_{nA}$ *which means that for every* $v_1, w_1 \in s_{1A}$, ..., $v_n, w_n \in s_{nA}$, *if* $v_n, w_n \in s_{nA}$, *if* $v_1 \approx_{s_1,A} w_1$, ... , $v_n \approx_{s_n,A} w_n$ *then* $p_A(v_1, \ldots, v_n) \Leftrightarrow p_A(w_1, \ldots, w_n)$

Notation: *A family of partial* Σ*-congruences* $<\approx_A>_{A \in Alg(\Sigma)}$ *will be denoted just by* $\approx$.

Definition 3.7 *Assume that* $\Sigma \in |AlgSig|$*, let* A *be a* Σ*-algebra and let* $\approx_A$ *be a partial* Σ*-congruence. The domain of* $\approx_A$ *is defined for any sort* $s \in S$ *as follows:*

$$Dom_s(\approx_A) \ = \ \{v \mid v \ \approx_{A,s} \ v\}$$

Definition 3.8 *Let* $\Sigma = (S, Op)$ *be a signature, let* A *be a* Σ*-algebra and let* $\approx_A$ *be a partial* Σ*-congruences. For any* $s \in S$ *and for any* $v \in Dom_s(\approx_A)$*, the class* $[v]_{\approx_A}$ *is defined as follows:*

$$[v]_{\approx_A} = \{v' \mid v' \approx_A v\}$$

Definition 3.9 *The quotient of an algebra* $A \in |Alg(\Sigma)|$ *by a partial* Σ*-congruence* $\approx_A$ *is defined as:*

$$s_{A/\approx_A} = \{[v]_{\approx_A} \mid v \in s_A \textit{ for every sort } s \textit{ in } \Sigma\}$$

$$f_{A/\approx_A}([v_1]_{\approx_A}, \ldots, [v_n]_{\approx_A}) = [f_A(v_1, \ldots, v_n)]_{\approx_A}$$

$$p_{A/\approx_A}([v_1]_{\approx_A}, \ldots, [v_n]_{\approx_A}) \ \Leftrightarrow \ p_{A/\approx_A}(v_1, \ldots, v_n)$$

Definition 3.10 *The behaviour of a* Σ*-algebra* A *with respect to a partial congruence is defined by its quotient and denoted by* $Beh_{\approx_A}(A) = Dom(A/\approx_A)$.

Definition 3.11 *An equivalence relation between algebras of signature* Σ *is a relation with domain included in* $|Alg(\Sigma)| \times |Alg(\Sigma)|$ *which is reflexive, symmetric and transitive, and it will be normally denoted by the symbol* $\equiv$.

In the following, we define a list of semantic operators which are used for the definition the semantics of different operators for ASL.

Definition 3.12 *The operators on classes of models* Iso, $\_/\approx$, $Abs_\equiv$, $Beh_\approx$ *are formally defined as follows:*

$$Iso(C) = \{A \mid \exists B \in C.A \cong B\}$$

$$C/\approx = \{A/\approx_A \ \mid A \in C\}$$

$$Abs_\equiv(C) = \{A \mid \exists B \in C.A \equiv B\}$$

$$Beh_\approx(C) = \{A \mid A/\approx_A \in C\}$$

Definition 3.13 *A BASLker specification language is an ASLker specification language including the* **behaviour**, **abstract** *and* **quotient** *operators with syntax:*

$$SP_0 ::= \textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx \qquad (\textbf{behaviour})$$

$$\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv \qquad (\textbf{abstract})$$

$$SP/\approx \qquad (\textbf{quotient})$$

and the following semantics:

$$Signature(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx) = Signature(SP)$$

$$Symbols(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx) = Symbols(SP)$$

$$Models(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx) = Beh_{\approx}(Models(SP))$$

$$Signature(\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv) = Signature(SP)$$

$$Symbols(\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv) = Symbols(SP)$$

$$Models(\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv) = Abs_{\equiv}(Models(SP))$$

$$Signature(SP/\approx) = Signature(SP)$$

$$Symbols(SP/\approx) = Symbols(SP)$$

$$Models(SP/\approx) = Iso(Models(SP)/\approx)$$

where $\approx$ *and* $\equiv$ *denote fixed but arbitrary family of partial* $Signature(SP)$*-congruences and equivalence relations respectively.*

Finally, we relate the semantics of the behavioural operators of BASLker languages, giving first some properties on classes of Σ-algebras and specifications:

Definition 3.14 *A family of partial* Σ*-congruences is isomorphism compatible if for all* Σ*-algebras* A *and* B*, if* $A \cong B$ *then* $A/\approx_A \cong B/\approx_B$

Definition 3.15 *An equivalence relation between* Σ*-algebras is isomorphism protecting if for all* Σ*-algebras* A *and* B*,* $A \cong B$ *implies* $A \equiv B$*.*

Definition 3.16 *A family of partial* Σ*-congruences is weakly regular if for all* $A \in |Alg(\Sigma)$*,* $A/\approx_A$ *is isomorphic to* $(A/\approx_A)/(\approx_{A/\approx_A})$*.*

Definition 3.17 *An equivalence relation between Σ-algebras ($\equiv$) is factorizable by a family of partial Σ-congruences if for all Σ-algebras A and B, $A \equiv B$ if and only if $A/\approx_A \cong B/\approx_B$.*

Proposition 3.18 *If SP is closed under isomorphism, $\approx$ is isomorphism compatible and $\equiv$ is isomorphism protecting then* **behaviour** SP **wrt** $\approx$ *and* **abstract** SP **by** $\equiv$ *and $SP/\approx$ are closed under isomorphism.*

Proof:

Assume that SP is closed under isomorphism, $\approx$ is isomorphism compatible and $\equiv$ is isomorphism protecting.

What we have to prove for the above operators Op(SP) is that

$$\forall A, B \in Alg(Signature(Op(SP))). \quad A \in Models(Op(SP)) \quad \wedge \quad A \cong B \\ \Rightarrow \quad B \in Models(Op(SP))$$

where the case of the quotient operator is obvious by definition.

1. $Op(SP) =$ **behaviour** SP **wrt** $\approx$
   Assume that $A, B \in Alg(Signature(SP))$.
   By the definition of $Models($**behaviour** SP **wrt** $\approx)$ and since $A \in Models($**behaviour** SP **wrt** $\approx)$ we know that $A/\approx \in Models(SP)$ and our goal is equivalent to $B/\approx \in Models(SP)$. Since $\approx$ is isomorphism compatible and instantiating it with A and B, we have that

   $$A \cong B \quad \Rightarrow \quad A/{\approx_A} \cong B/{\approx_B}$$

   Since SP is closed under isomorphism and instantiating it with $A/{\approx_A}$ and $B/{\approx_B}$, we've got that

   $$A/\approx_A \in Models(SP) \ \wedge \ A/\approx_A \cong B/\approx_B \ \Rightarrow \ B/\approx_B \in Models(SP)$$

   Using that $A \cong B$, the instantiation of $\approx$ is isomorphism compatible and the instantiation of SP is closed under isomorphism , we trivially prove our goal.

2. $Op(SP) =$ **abstract** SP **by** $\equiv$
   Assume that $A, B \in Alg(Signature(SP))$.
   Since $\equiv$ is isomorphism protecting, instantiating this proposition with A and B and using that $A \cong B$, we have that $A \equiv B$.
   Since $A \equiv B$ and using $A \in Models($**abstract** SP **by** $\equiv)$, we've got that

   $$B \in Models(\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv)$$

Definition 3.19 *Let ASL be an $ASLker$ or $ASLnf$ language. Two specifications $SP_1, SP_2 \in SPEX(ASL)$ are equal if the following holds:*

$$Signature(SP_1) \ = \ Signature(SP_2)$$

$$\forall A \in Alg(Signature(SP_1).A \in Models(SP_1) \ \Leftrightarrow \ A \in Models(SP_2)$$

Theorem 3.20 *For any specification expression SP with signature (S, Op) which is closed under isomorphism, for any family of partial Σ-congruences $(\approx)$ which is weakly regular and for any equivalence relation between Σ-algebras $(\equiv)$ which is factorizable by $\approx$, the following equivalence between specifications holds:*

$$\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv \ = \ \textbf{behaviour} \quad (SP/\approx) \quad \textbf{wrt} \quad \approx$$

Proof:
Let SP be a specification expression with signature Σ which is closed under isomorphism, let $\approx$ be a weakly regular family of partial Σ-congruences and let $\equiv$ be an equivalence relation which is factorizable by $\approx$.

By the definition of equality of specifications, we have to prove that:

- $Signature(\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv) =$
  $Signature(\textbf{behaviour} \quad SP/\approx \quad \textbf{wrt} \quad \approx)$ which holds since
  $Signature(\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv) =$
  $Signature(\textbf{behaviour} \quad SP/\approx \quad \textbf{wrt} \quad \approx) = Signature(SP)$

- $$\forall A \in Alg(Signature(SP)).A \in Models(\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv) \quad \Leftrightarrow$$

  $$A \in Models(\textbf{behaviour} \quad SP/\approx \quad \textbf{wrt} \quad \approx)$$

  Assume that $A \in Alg(Signature(SP))$ and assume that
  $A \in Models(\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv)$.
  By the definition of $Models(\textbf{abstract} \quad SP \quad \textbf{by} \quad \equiv)$, we have that $\exists B \in Models(SP).A \equiv B$. Since $\equiv$ is factorizable by $\approx$ we have that last proposition is equivalent to

  $$\exists B \in Models(SP).A/\!\approx \cong B/\!\approx$$

  By the definition of $Models(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx)$ and the definition of $Models(SP/\approx)$ our goal is equivalent to

  $$A/\approx \in Iso(Models(SP)/\approx)$$

  which is true because by the definition of $Iso(Models(SP)/\approx)$ our goal can be trivially proven equivalent to

  $$\exists B \in Models(SP).A/\!\approx \cong B/\!\approx$$

4 BASLnf specification languages

In this section, we present the inductive definition of the normalisation function of the behavioural operators of *BASLker* specification languages using a fixed but arbitrary institution which extends algebraic institutions with several components such as a fixed but arbitrary family of partial congruences, a behavioural satisfaction relation and different functions which are used to define the normal forms of the behavioural operators. We will refer to these institutions as behavioural algebraic institutions. We also generalise the semantics of the behaviour operator with higher-order logic as specification logic of [7] using also behavioural algebraic institutions and we relate this semantics with the semantics of the behaviour operator of $BASLker$ specification languages.

In [6], the normalisation function of the behavioural operators with infinitary first-order logic as specification logic is presented. The equivalences of the behaviour and abstract operator are the observational and behavioural equality which are presented in the last section of this chapter. The normalisation functions of the behavioural and quotient operator are defined as the normalisation functions of structured specifications, the semantics of which are equivalent to the semantics of the behaviour and quotient operator.

In [7], the semantics of the behavioural operator is given in terms of a behavioural satisfaction relation which is denoted as $\models^{\approx}$ where $\approx$ is a fixed but arbitrary family of partial congruences and they use higher-order logic as specification logic.

They also define a relativization function which we will refer as $brel$ relating standard and behavioural satisfaction in the following way:

$$\forall \Sigma \in |AlgSig|.\forall A \in |Alg(\Sigma)|.\forall \phi \in Sen_{HOL}(\Sigma).$$

$$A \models^{\approx} \phi \;\Leftrightarrow\; A \models brel(\phi)$$

The semantics of the behavioural operator is defined using the behavioural satisfaction relation in the following way:

$$Models(\textbf{behaviour} \quad < \Sigma, \Phi > \quad \textbf{wrt} \quad \approx) \;=\; \{A \in Alg(\Sigma) \mid A \models^{\approx} \Phi\}$$

This operator can only be applied to base specifications and because of the relation between behavioural and standard satisfaction, the semantics of the behavioural operator can also be defined as:

$$Models(\textbf{behaviour} \quad < \Sigma, \Phi > \quad \textbf{wrt} \quad \approx) \;=\; \{A \in Alg(\Sigma) \mid A \models brel(\Phi)\}$$

Since they also prove that

$$\forall \Sigma \in |AlgSig|.\forall A \in |Alg(\Sigma)|.\forall \phi \in Sen_{HOL}(\Sigma).$$

$$A/\!\approx\, \models \phi \;\Leftrightarrow\; A \models^{\approx} \phi$$

the relativization function $brel$ also satisfies the following condition:

$$\forall \Sigma \in |AlgSig|.\forall A \in |Alg(\Sigma)|.\forall \phi \in Sen_{HOL}(\Sigma).$$

$$A/\!\approx\, \models \phi \;\Leftrightarrow\; A \models brel(\phi)$$

In order to define the normalisation function of the behavioural operators of $BASLker$ languages for an arbitrary algebraic institution and arbitrary equivalence relations, we have decided to define them in terms of the normal form of the argument specification of the behavioural operators. The definition of the normal form will use functions on sentences based on the idea of relativization function presented in [7] and above for the behaviour and quotient operator. The normal form of the abstract operator will be defined as the normal form of the behaviour of the quotient of the argument specification of the abstract operator. It follows that this is the normal form of the abstract operator using theorem 3.20 as in [6].

The definition of the relativization functions for the normalisation functions of $BASLker$ languages will need in general to extend the original signature of the specification with extra symbols. For example, an alternative definition of the normalisation function of the behaviour operator for the institution HOL and for an observational equality is based on [6], which, as we mentioned above, proves the equivalence between the semantics of the behaviour operator with the semantics of a structured specification. The symbols of the structured specification extends the symbols of the behavioural operator with, for example, a disjoint copy of the signature of the behaviour operator and symbols to denote the observational equality. A possible definition of the relativization functions uses the same symbols as the symbols of the structured specifications. The conditions which must satisfy these relativization functions are more complicated than the condition which satisfies the relativization function of [7] presented below.

Thus, we present in this section an institution which extends algebraic institutions with a behavioural satisfaction relation and two different functions on sentences for any inclusion $i : \Sigma \hookrightarrow \Sigma'$ in $AlgSig$. These institutions will be referred as behavioural algebraic institution. One of this kind of functions will be denoted by $brel[i, bi[\Sigma]]$ where i is an inclusion with arity $i : \Sigma \hookrightarrow \Sigma'$ and $bi[\Sigma]$ is an inclusion with arity $bi[\Sigma] : \Sigma' \hookrightarrow \Sigma''$ and these functions will be used to define the normalisation functions of the behaviour operator. These functions can also be used to define the normal form of the generalized semantics of the behaviour operator presented in [7] extended to structured specifications. See subsection 5.4 for a concrete example of a definition of the inclusion $bi[\Sigma]$ and the function $brel[i, bi[\Sigma]]$ in a behavioural algebraic institution with higher-order logic as algebraic institution.

The inclusion i of the function $brel[i, bi[\Sigma]]$ used in the normalisation function of the behaviour operator **behaviour** SP **wrt** $\approx$ of $BASLker$ specification languages will have arity $i : Signature(SP) \hookrightarrow Symbols_{nf}(SP)$ whereas the inclusion of the function $brel$ used in the behaviour operator of [7] will be the identity $i : Symbols_{nf}(SP) \hookrightarrow Symbols_{nf}(SP)$.

The other kind of functions will be used to define the normalisation function of the semantics of the quotient operator and it will be denoted as $qrel[i, qi[\Sigma]]$ where i is an inclusion with arity $i : \Sigma \hookrightarrow \Sigma'$ and $qi[\Sigma]$ is an inclusion with arity $qi[\Sigma] : \Sigma' \rightarrow \Sigma''$.

The conditions which must satisfy these two kind of functions are presented in the general definition of behavioural algebraic institutions ($BAINS$). See next section for proofs that these concrete functions satisfy the general conditions which are defined in $BAINS$. In this section, we present the definition of $BAINS$, the alternative and generalised version of the semantics of the behaviour operator of [7] and how to relate it with $BASLker$ languages.

Definition 4.1 *A behavioural algebraic institution (BAINS) consists of an algebraic institution and additionally the following 6 components:*

- *For any signature* $\Sigma \in |AlgSig|$*, a fixed but arbitrary family of partial* Σ*-congruences* $<\approx_A>_{A \in |Alg(\Sigma)|}$.
- *For each signature* $\Sigma \in |AlgSig|$ *a behavioural satisfaction relation* $\models^{\approx}_{BIASL,\Sigma}$: $Alg(\Sigma) \times Sen_{BIASL}(\Sigma)$*, which is related to the standard satisfaction by the behavioural satisfaction condition. This condition is defined as:*

$$\forall A \in Alg(\Sigma).\forall \phi \in Sen_{BIASL}(\Sigma).A/\approx \models_{BIASL,\Sigma} \phi \Leftrightarrow A \models^{\approx}_{BIASL,\Sigma} \phi$$

- *For each inclusion* $i : \Sigma \hookrightarrow \Sigma'$ *in AlgSig, an inclusion* $bi[\Sigma] : \Sigma' \hookrightarrow \Sigma''$ *and a function*

$$brel[\Sigma, bi[\Sigma]] : \mathcal{P}(Sen_{BAINS}(\Sigma')) \to \mathcal{P}(Sen_{BAINS}(bi[\Sigma](\Sigma')))$$

  which satisfies the following conditions which we will refer as the behavioural relativization conditions:

$$(1)\forall A' \in |Alg(\Sigma')|.\forall \Phi \in \mathcal{P}(Sen_{BAINS}(\Sigma')).\forall A \in |Alg(\Sigma)|.$$

$$A'|_{\Sigma} = A/\approx \wedge A' \models \Phi \Rightarrow$$

$$\exists A'' \in |Alg(bi[\Sigma](\Sigma'))|.\, A''|_{\Sigma'} = A''' \wedge A'' \models brel[i, bi[\Sigma]](\Phi)$$

  for a given Σ'*-algebra* A''' *is defined as follows:*

$$\begin{array}{l} A'''|_{\Sigma} = A \\ A'''_s = A'_s \quad \text{for any sort } s \in Sorts(\Sigma') - Sorts(\Sigma) \\ f_{A'''} = f_{A'} \quad \text{for any sort } f \in Ops(\Sigma') - Ops(\Sigma) \\ P_{A'''} = P_{A'} \quad \text{for any sort } P \in Pr(\Sigma') - Pr(\Sigma) \end{array}$$

  and

$$(2)\forall A'' \in Alg(bi[\Sigma](\Sigma')).\forall \Phi \in \mathcal{P}(Sen_{BAINS}(\Sigma')).$$

$$A'' \models brel[i, bi[\Sigma]](\Phi) \Rightarrow \exists A' \in |Alg(\Sigma')|.A'|_{\Sigma} = A''|_{\Sigma}/\approx \wedge A' \models \Phi$$

- *For each inclusion* $i : \Sigma \hookrightarrow \Sigma'$ *in AlgSig, an inclusion* $qi[\Sigma] : \Sigma' \hookrightarrow \Sigma''$ *and a function*

$$qrel[i, qi[\Sigma]] : \mathcal{P}(Sen_{BAINS}(\Sigma')) \to \mathcal{P}(Sen_{BAINS}(qi[\Sigma](\Sigma')))$$

which satisfies the following conditions which we will refer as the quotient relativization conditions:

$(1)\forall A' \in |Alg(\Sigma')|.\forall \Phi \in \mathcal{P}(Sen_{BAINS}(\Sigma')).\ A' \models_{\Sigma'} \Phi \Rightarrow$

$$\exists A'' \in Alg(qi[\Sigma](\Sigma')).\ A''|_{\Sigma} \cong A'|_{\Sigma}/\!\approx\ \wedge A'' \models_{qi[\Sigma](\Sigma')} qrel[i, qi[\Sigma]](\Phi)$$

$(2)\forall A'' \in Alg(qi[\Sigma](\Sigma')).\forall \Phi \in \mathcal{P}(Sen_{BAINS}(\Sigma')).$

$$A'' \models_{qi[\Sigma](\Sigma')} qrel[i, qi[\Sigma]](\Phi) \Rightarrow$$

$$\exists A' \in |Alg(\Sigma')|.A''|_{\Sigma} \cong A'|_{\Sigma}/\!\approx\ \wedge A' \models_{\Sigma', BAINS} \Phi$$

Remark: *If the inclusion i is an identity ($i : \Sigma \to \Sigma$) then the first behavioural relativization condition can be rewritten to:*

$\forall A \in |Alg(\Sigma)|.\forall \Phi \in \mathcal{P}(Sen_{BAINS}(\Sigma)).$

$$A/\!\approx\,\models\ \Phi\ \Leftrightarrow \exists A' \in |Alg(bi[\Sigma](\Sigma))|.\ A'|_{\Sigma} = A\ \wedge\ A' \models brel[i, bi[\Sigma]](\Phi)$$

Definition 4.2 *A $BASLnf$ specification language is an $ASLnf$ specification language over a behavioural algebraic institution $BAINS$ with additionally the* **behaviour**, **abstract** *and* **quotient** *operator with the same additional semantic conditions as in $BASLker$ and additionally the following conditions:*

Let $nf(SP)$ *be* $< \Sigma', \Phi > |_{\Sigma}$. *Then:*

$nf(\textbf{behaviour}\quad SP\quad \textbf{wrt}\quad \approx)\ =$

$$< Symbols_{nf}(\textbf{behaviour}\quad SP\quad \textbf{wrt}\quad \approx), brel[i, bi[\Sigma]](\Phi) > |_{\Sigma}$$

$Symbols_{nf}(\textbf{behaviour}\quad SP\quad \textbf{wrt}\quad \approx)\ =\ bi[\Sigma](\Sigma')$

$nf(\textbf{abstract}\quad SP\quad \textbf{by}\quad \equiv)\ =\ nf(\textbf{behaviour}\quad SP/\approx\quad \textbf{wrt}\quad \approx)$

$Symbols_{nf}(\textbf{abstract}\quad SP\quad \textbf{by}\quad \equiv)\ =$

$$Symbols_{nf}(\textbf{behaviour}\quad SP/\approx\quad \textbf{wrt}\quad \approx)$$

$nf(SP/\approx)\ = < Symbols_{nf}(SP/\approx), qrel[i, qi[\Sigma]](\Phi) > |_{\Sigma}$

$Symbols_{nf}(nf(SP/\approx))\ =\ qi[\Sigma](\Sigma')$

where $brel[i, bi[\Sigma]] : Sen_{BAINS}(\Sigma') \to Sen_{BAINS}(bi[\Sigma](\Sigma'))$ *is a function which satisfies the behavioural relativization conditions where i has arity* $i : \Sigma \hookrightarrow \Sigma'$

and $qrel[i, qi[\Sigma]] : Sen_{BAINS}(\Sigma') \rightarrow Sen_{BAINS}(qi[\Sigma](\Sigma'))$ *is a function which satisfies the quotient relativization conditions.*

Theorem 4.3 *BASLnf is an ASLnf specification language.*

Proof:
The proof is by induction on specification expressions. Let *behop* denote any of the three behavioural operators. It is trivial to show that

$$Signature(behop(SP)) = Signature(nf(behop(SP)))$$

and that there exists an inclusion between Σ' and $Symbols_{nf}(SP)$ for all the behavioural operators. Besides, we have to show that

$$A \in behop(SP) \Leftrightarrow A \in nf(behop(SP))$$

for every behavioural operator.
We assume that $nf(SP) =< \Sigma', \Phi > |_{\Sigma}$ where $\Sigma' = Symbols_{nf}(SP)$ and $\Sigma = Signature(SP)$ for every argument specification SP of any behavioural operator $behop(SP)$.

- $behaviour_{nf}$ SP **wrt** $\approx$:
  $\Rightarrow$)
  By the definition of the behaviour operator we know that:

  $$A \in Models(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx) \Leftrightarrow A/\approx \in Models(SP)$$

  By the induction hypotheses, $A/\approx \in Models(SP)$ can be rewritten to:

  $$\exists A' \in |Alg(\Sigma')|. A'|_{\Sigma} = A/\approx \ \wedge A' \models \Phi$$

  Let A' be the Σ'-algebra such that

  $$A'|_{\Sigma} = A/\approx \ \wedge A' \models \Phi$$

  By the first behavioural relativization condition of the function $brel[i, bi[\Sigma]]$, we can deduce that

  $$\exists A'' \in Alg(bi[\Sigma](\Sigma')). A''|_{\Sigma} = A''' \ \wedge \ A'' \models brel[i, bi[\Sigma]](\Phi)$$

  where A''' is defined as:

  $$\begin{array}{l} A'''|_{\Sigma} = A \\ A'''_s = A'_s \quad \text{for any sort } s \in Sorts(\Sigma') - Sorts(\Sigma) \\ f_{A'''} = f_{A'} \quad \text{for any sort } f \in Ops(\Sigma') - Ops(\Sigma) \\ P_{A'''} = P_{A'} \text{for any sort } P \in Pr(\Sigma') - Pr(\Sigma) \end{array}$$

  And therefore we have that:

  $$A \in Models(< bi[\Sigma](\Sigma'), brel[i, bi[\Sigma']](\Phi) > |_{\Sigma})$$

and therefore $A \in Models(nf(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx))$.

$\Leftarrow$)
By the definition of $Models(nf(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx))$ we know that

$$\exists A' \in Alg(bi[\Sigma](\Sigma')).\, A'|_\Sigma = A \ \wedge \ A' \models_{\Sigma'} brel[i, qi[\Sigma]](\Phi)$$

Let A' be a Σ'-algebra such that $A'|_\Sigma = A$ and $A' \models_{\Sigma'} brel[i, qi[\Sigma]](\Phi)$. By the second condition of the behavioural relativization function we can deduce that

$$\exists A'' \in |Alg(\Sigma')|.A''|_\Sigma \, = \, A/\!\approx \ \wedge A'' \models \Phi$$

and by the induction hypotheses we have that

$$A \in Models(\textbf{behaviour} \quad SP \quad \textbf{wrt} \quad \approx)$$

- $SP/_{nf} \approx$:
  $\Rightarrow$)
  By the definition of $Models(SP/\approx)$, we know that

  $$A \in Models(SP/\approx) \Leftrightarrow \exists A' \in |Alg(\Sigma)|.A \cong A'/\!\approx \ \wedge A' \in Models(SP)$$

  By the induction hypotheses, we can transform the right hand side part of the previous proposition to:

  $$\exists A'' \in |Alg(\Sigma')|.A''|_\Sigma = A' \ \wedge \ A \cong A'/\!\approx \ \wedge A'' \models \Phi$$

  Let A'' be a Σ'-algebra such that

  $$A''|_\Sigma = A' \ \wedge \ A \cong A'/\approx \ \wedge A'' \models \Phi$$

  By the condition of quotient relativization function we can deduce that

  $$\exists A''' \in Alg(qi[\Sigma](\Sigma')).\, A'''|_\Sigma \, \cong \, A'/\!\approx \ \wedge$$

  $$A''' \models_{\Sigma', BAINS} qrel[i, qi[\Sigma]](\Phi)$$

  and therefore $A \in Models(nf(SP/\approx))$

  $\Leftarrow$):

  By the definition of $Models(nf(SP/\approx))$ we know that

  $$\exists A' \in Alg(qi[\Sigma](\Sigma')).\, A'|_\Sigma = A \ \wedge \ A' \models_{\Sigma'} qrel[i, qi[\Sigma]](\Phi)$$

  Let A' be a Σ'-algebra such that $A'|_\Sigma = A$ and $A' \models_{\Sigma'} qrel[i, qi[\Sigma]](\Phi)$. By the second condition of the quotient relativization function we can deduce that

  $$\exists A'' \in |Alg(\Sigma')|.A'|_\Sigma \, \cong \, A''|_\Sigma/\!\approx \ \wedge A'' \models_{\Sigma', BAINS} \Phi$$

  and by the induction hypotheses we have that $A \in Models(SP/\approx)$.

- **abstract** SP **by** $\equiv$: We know by theorem 4.3.24 that
  abstract SP **by** $\equiv =$ **behaviour** $SP/\approx$ **wrt** $\equiv$.
  Since we know by induction hypotheses that
  $A \in Models(nf(SP)) \Leftrightarrow A \in Models(SP)$,
  by the induction condition of the quotient operator we know that $SP/\approx$ $= nf(SP/\approx)$ and by the induction condition of the behavioural operator we know that
  behaviour $SP/\approx$ **wrt** $\approx = nf($**behaviour** $SP/\approx$ **wrt** $\equiv)$.
  Thus **abstract** SP **by** $\equiv = nf($**abstract** SP **by** $\equiv)$.

Finally, we relate the semantics of the behaviour operator of $BASLnf$ languages with the generalisation of the semantics of this operator given in [7].

In the section of further work of [7], general lines are given to define the semantics of this operator for structured specifications. The $Models$ function is defined using an auxiliar function as follows:

$$Models(\textbf{behaviour}\ \ SP\ \ \textbf{wrt}\ \ \approx)\ =\ Mod_{\approx}(SP)$$

and the auxiliar function $Mod_{\approx}$, following the underlying ideas of [7], can be defined for the common operators of $ASLnf$ languges and for an arbitrary but fixed behavioural institution $BAINS$ as follows:

$$Mod_{\approx}(<\Sigma,\Phi>)\ =\ \{A \in Alg(\Sigma) \mid A \models^{\approx} \Phi\}$$

$$Mod_{\approx}(SP_1 +_{\Sigma} SP_2)\ =$$

$$\{A \mid A \in Alg(Signature(SP_1) +_{\Sigma} Signature(SP_2)),$$

$$A|_{inl} \in Mod_{\approx}(SP_1), A|_{inr} \in Mod_{\approx}(SP_2)\}$$

$$Mod_{\approx}(SP|_{\Sigma})\ =\ \{A|_{\Sigma} \mid A \in Mod_{\approx}(SP)\}$$

Note that the $Mod_{\approx}$ is not defined for the behavioural operator.

An alternative possible way to give semantics to the behaviour operator in $ASLnf$ languages which is equivalent to the previous extension under the same syntactic restrictions is as follows:

$$Signature(behaviour_{nf} \quad SP \quad \mathbf{wrt} \quad \approx) = Signature(SP)$$

$$Symbols(behaviour_{nf} \quad SP \quad \mathbf{wrt} \quad \approx) = Symbols(SP)$$

$$Models(behaviour_{nf} \quad SP \quad \mathbf{wrt} \quad \approx) =$$

$$\{A \in Alg(\Sigma') \mid A \models^{\approx} \Phi\}$$

$$nf(behaviour_{nf} \quad SP \quad \mathbf{wrt} \quad \approx) =$$

$$< Symbols_{nf}(behaviour_{nf} \quad SP \quad \mathbf{wrt} \quad \approx),$$

$$brel[i, bi[\Sigma']](\Phi) > |_{\Sigma}$$

$$Symbols_{nf}(behaviour_{nf} \quad SP \quad \mathbf{wrt} \quad \approx) = bi[\Sigma'](\Sigma')$$

where $nf(SP) =< \Sigma', \Phi > |_{\Sigma}$,

$$brel[i, bi[\Sigma']] : \mathcal{P}(Sen_{BAISS}(\Sigma')) \to \mathcal{P}(Sen_{BAISS}(bi[\Sigma'](\Sigma')))$$

is a function which satisfies the behavioural relativization condition,
and i has arity $i : \Sigma' \hookrightarrow \Sigma'$.

One way to see that this alternative semantics is equivalent to the one presented in [7] is to define the $ASLnf$ language $ASLN$ with just the common operators of $ASLnf$ languages and prove the following proposition for any $SP \in SPEX(ASLN)$:

$$\forall A \in Models(SP).\ A \in Mod_{\approx}(SP) \Leftrightarrow$$

$$A \in Models(behaviour_{nf} \quad SP \quad \mathbf{wrt} \quad \approx)$$

which follows trivially by the behavioural relativization conditions. Finally, we can relate our alternative semantics with the semantics of the behaviour operator of $BASLker$ languages with the following theorem:

Theorem 4.4 *Let $BAINS$ be a behavioural algebraic institution. Let $BSPL_1$ be a $BASLnf$ specification language over $BAINS$ and let $BSPL_2$ be the $ASLnf$ language with additionally the behaviour operator with the alternative semantics presented below and the same institution $BAINS$. Let SP_1 be a specification expression of $BSPL_1$ and let SP_2 be a specification expression of $BSPL_2$ such that*

$$A \in Models(SP_1) \Leftrightarrow A \in Models(SP_2).$$

$$nf(SP_1) = nf(SP_2) =< \Sigma', \Phi > |_{\Sigma}$$

Under these assumptions the following holds:

$$A \in Models(behaviour_{nf} \quad SP_2 \quad \mathbf{wrt} \quad \approx) \Rightarrow$$

$$A \in Models(\mathbf{behaviour} \quad SP_1 \quad \mathbf{wrt} \quad \approx)$$

Proof:

Assume that $A \in Models(behaviour_{nf} \quad SP_2 \quad \mathbf{wrt} \quad \approx)$. By the definition of the semantics of this behavioural operator, the previous proposition is equivalent to

$$\exists A' \in Alg(bi[\Sigma'](\Sigma')).A'|_{\Sigma} = A \ \wedge \ A' \models brel[i, bi[\Sigma']](\Phi)$$

By the behavioural relativization condition we can deduce that $A'/{\approx} \models \Phi$ and by the behavioural satisfaction condition we can deduce that $A' \models^{\approx} \Phi$. Since $A'|_{\Sigma} = A$ we have that $A \in Models(\mathbf{behaviour} \quad SP_1 \quad \mathbf{wrt} \quad \approx)$.

Note that the left implication of the propositions which relate the class of models of the behavioural operators doesn't seem to hold since from our point of view it is necessary that an equivalent or more general condition to the following one

$$\forall A'' \in |Alg(\Sigma'')|.\forall \phi \in Sen_{BAINS}(\Sigma'').$$

$$\exists A \in |Alg(\Sigma)|.A''|_{\Sigma} = A/{\approx} \quad \wedge \quad A'' \models_{\Sigma'',BAINS} \phi \Rightarrow$$

$$\exists A' \in |Alg(\Sigma')|.A''|_{\Sigma'} = A'/{\approx} \quad \wedge \quad A'' \models_{\Sigma'',BAINS} \phi$$

must be satisfied for any pair of inclusions $i : \Sigma \hookrightarrow \Sigma'$, $i : \Sigma' \hookrightarrow \Sigma''$ of a fixed but arbitrary behavioural algebraic institutions and we have not succeeded to find it for example for the institution of this kind with a higher-order logic presented in the next section.

5 BHOL: A behavioural algebraic institution

In this section, we present a behavioural algebraic institution for HOL with a concrete family of partial congruence: the observational equality.

First, we present the definition of the behavioural satisfaction relation for an arbitrary but fixed family of partial congruences as in [7]. Then, we define the observational equality for first-order relational signatures which will be referred to as relational observational equality and for first-order signatures which will be referred to as observational equality, but in both cases they will be denoted by the same symbol.The formulation of the observational equality is the same as in [3] or [6].

Next, we define the relativisation functions for the institution HOL and finally we present the behavioural algebraic institution $BHOL$.

5.1 The behavioural satisfaction relation

In order to define the behavioural satisfaction relation for the algebraic institution HOL it is necessary to extend the given fixed but arbitrary family of partial Σ-congruences to higher-order types. We present this extension just for the general case because the instantiation to the relational observational equality is obvious.

Definition 5.1 *Let $\approx$ be a family of partial Σ-congruences. The relation $\approx_{A,[\tau_1,\ldots,\tau_n]}$: $[\![[\tau_1,\ldots,\tau_n]]\!]_A \times [\![[\tau_1,\ldots,\tau_n]]\!]_A$ for any $\tau_1 \in Types_{HOL}(\Sigma),\ldots,\ \tau_n \in Types_{HOL}(\Sigma)$ and for any $A \in |Alg(\Sigma)|$ is defined as follows:*

$$p \approx_{A,[\tau_1,\ldots,\tau_n]} p' \Leftrightarrow \forall v_1, v_1' \in [\![\tau_1]\!]_A. \ldots .\forall v_n, v_n' \in [\![\tau_n]\!]_A.$$

$$(v_1,\ldots,v_n) \in p \Leftrightarrow (v_1',\ldots,v_n') \in p'$$

Definition 5.2 *For any relational signature $\Sigma \in |AlgSig|$, for any sort $s \in Sorts(\Sigma)$, for any Σ-algebra A, a value $v \in A_s$ respects a partial Σ-congruence if $v \approx_A v$. A predicate $p \in [\![[\tau_1,\ldots,\tau_n]]\!]_A$ respects $\approx_{A,[\tau_1,\ldots,\tau_n]}$ for any $\tau_1 \in Types_{HOL}(\Sigma),\ldots,\ \tau_n \in Types_{HOL}(\Sigma)$ if the following condition holds:*

$$\forall v_1, v_1' \in [\![\tau_1]\!]_A. \ldots .\forall v_n, v_n' \in [\![\tau_n]\!]_A.$$

$$(v_1,\ldots,v_n) \in p \Leftrightarrow (v_1',\ldots,v_n') \in p$$

Proposition 5.3 *$\approx_{A,\tau}$ is a partial equivalence relation for any $\tau \in Types_{HOL}(\Sigma)$*

Proof:
See [7].

Definition 5.4 *The semantic function $[\![\tau]\!]^{\approx}_A$ is inductively defined for any type $\tau \in Types_{HOL}(\Sigma)$ and for any Σ-algebra A as follows:*

$$[\![s]\!]^{\approx}_A = \{\, v \in A_s \mid v \textit{ respects } \approx \}$$

$$[\![[\tau_1,\ldots,\tau_n]]\!]^{\approx}_A = \{p \in \mathcal{P}([\![\tau_1]\!]^{\approx}_A \times \ldots \times [\![\tau_n]\!]^{\approx}_A \mid p \textit{ respects } \approx \}$$

Notation: *The semantics of* **Prop** *is a set of two elements: the empty set and the set with the empty tuple. These two elements will be denoted as* **ff** *and* **tt** *respectively.*

Definition 5.5 *The function $[\![t]\!]^{\approx}_{\rho,A}$ for any term $t \in Terms_{HOL}(\Sigma, X_{HOL})$, for any algebra $A \in Alg(\Sigma)$, for any $Types_{HOL}(\Sigma)$-sorted valuation ρ which for every $\tau \in Types_{HOL}(\Sigma)$, ρ_τ has arity $\rho_\tau : X_{HOL,\tau} \to [\![\tau]\!]^{\approx}_A$ is inductively* *defined by the structure of* $t \in Terms(\Sigma, X)$ *as follows:*

$$[\![x_\tau]\!]^{\approx}_{\rho,A} = \rho_\tau(x)$$

$$[\![f(t_1,\ldots,t_n)]\!]^{\approx}_{\rho,A} = f_A([\![t_1]\!]^{\approx}_{\rho,A},\ldots,[\![t_n]\!]^{\approx}_{\rho,A})$$

$$[\![p(t_1,\ldots,t_n)]\!]^{\approx}_{\rho,A} = if\ ([\![t_1]\!]^{\approx}_{\rho,A},\ldots,[\![t_n]\!]^{\approx}_{\rho,A}) \in p_A\ then\ \mathbf{tt}\ else\ \mathbf{ff}$$

$$[\![\lambda(x_1:\tau_1,\ldots,x_n:\tau_n).\phi]\!]^{\approx}_{\rho,A} =$$

$$\{(v_1,\ldots,v_n) | v_1 \in [\![\tau_1]\!]^{\approx}_{\rho,A},\ldots,v_n \in [\![\tau_n]\!]^{\approx}_{\rho,A}, [\![\phi]\!]^{\approx}_{\rho\cup\{(x_1,v_1),\ldots,(x_n,v_n)\}} = \mathbf{tt}\}$$

$$[\![\lambda(x_1:\tau_1,\ldots,x_n:\tau_n).\phi\ (t_1,\ldots,t_n)]\!]^{\approx}_{\rho,A} = [\![\phi]\!]^{\approx}_{\rho\cup\{(x_1,[\![t_1]\!]^{\approx}_{\rho,A}),\ldots,(x_n,[\![t_n]\!]^{\approx}_{\rho,A})\},A}$$

$$[\![\phi \supset \phi']\!]^{\approx}_{\rho,A} = if\ [\![\phi]\!]^{\approx}_{\rho,A} = \mathbf{tt}\ then\ [\![\phi]\!]^{\approx}_{\rho,A}\ else\ \mathbf{tt}$$

$$[\![\forall x:\tau.\phi]\!]^{\approx}_{\rho,A} = if\ \forall v \in [\![\tau]\!]^{\approx}_{A}.[\![\phi]\!]^{\approx}_{\rho\cup\{(x,v)\},A} = \mathbf{tt}\ \ then\ \mathbf{tt}\ else\ \mathbf{ff}$$

Definition 5.6 *For each* $\Sigma \in |AlgSig|$, *for all* $A \in |Alg(\Sigma)|$, *for all* $\phi \in Sen_{HOL}(\Sigma)$, *the satisfaction relation* $A \models^{\approx}_{\Sigma} \phi$ *holds if and only if for any* $Types_{HOL}(\Sigma)$*-sorted valuation* ρ *which for every* $\tau \in Types_{HOL}(\Sigma)$, ρ_τ *has arity* $\rho_\tau : X_{HOL,\tau} \to [\![\tau]\!]_A$, $[\![\phi]\!]^{\approx}_{\rho,A} = \mathbf{tt}$

5.2 The relational observational equality

In this subsection, we define the relational observational equality with relational signatures giving also a relational behavioural equality which is factorizable by the relational observational equality.

Definition 5.7 *Let* Σ *be a relational signature in* $|AlgSig|$, *let* In *and* Obs *be two set of sorts s.t.* $In, Obs \subseteq Sorts(\Sigma)$ *and let* X_{In} *be an* In*-sorted set of variables. The* $Sorts(\Sigma)$*-sorted set of contexts* $PC_{\Sigma,Obs}(X_{In})$ *is defined for each sort* s *as the set of terms* $P_\Sigma(X_{In} \cup z_s)$ *such that* z_s *is a free variable which satisfies the condition* $\{z_s\} \cap X_s = \emptyset$. *This set is also denoted as* $PC_{\Sigma,Obs}(X_{In}, z_s)$ *for every sort* $s \in S$.

Definition 5.8 *Let* Σ *be a relational signature in* $|AlgSig|$, *let* Obs *and* In *be two set of sorts s.t.* $Obs, In \subseteq Sorts(\Sigma)$ *and* In *is sensible wrt* Σ*.Let* A *be a* Σ*-algebra. The relational observational equality* ($\approx^{Obs,In}_{A}$) *is formally defined* *for each sort s and for each $v, w \in A[X_{In}]_s$ as follows:*

$$v \approx^{Obs,In}_{s,A} w \Leftrightarrow$$

$$\begin{cases} \forall c \in C_{\Sigma,Obs}(X_{In}, z_s).\forall \alpha \in X_{In} \to A[X_{In}]. \\ \qquad I_{\alpha \cup \{(z_s,v)\}}(c) = I_{\alpha \cup \{(z_s,w)\}}(c) \wedge \\ \forall pc \in PC_{\Sigma,Obs}(X_{In}, z_s).\forall \alpha \in X_{In} \to A[X_{In}]. \\ \qquad I_{\alpha \cup \{(z_s,v)\}}(pc) \Leftrightarrow I_{\alpha \cup \{(z_s,w)\}}(pc) \\ \qquad\qquad , if s \in S - Obs \\ v = w \qquad\qquad , if s \in Obs \end{cases}$$

Notation: *To denote an observational equality $\approx^{Obs,In}_{A}$ we will normally drop the subscript denoting an algebra A if it can be inferred from the context.*

Proposition 5.9 *Let Σ be a relational signature in $|AlgSig|$, let Obs and In be two set of sorts s.t. $Obs, In \subseteq Sorts(\Sigma)$. The relational observational equality ($\approx^{Obs,In}$) is a family of partial Σ-congruences.*

Proof:
In the same way as in [3].

Proposition 5.10 *Let Σ be a relational signature in $|AlgSig|$, let Obs and In be two set of sorts s.t. $Obs, In \subseteq Sorts(\Sigma)$ and let A be a Σ-algebra. The relational observational equality ($\approx^{Obs,In}_{A}$) is weakly regular.*

Proof:
In a similar way as in [3].

Definition 5.11 *Let Σ be a signature,let In and Obs be two sets of sorts s.t. $In, Obs \subseteq Sorts(\Sigma)$ and let X_{In} be an In-sorted set of variables. The relational behavioural equality between Σ-algebras $\equiv_{Obs,In}$ is formally defined as:*

$$A \equiv_{Obs,In} B \quad \Leftrightarrow \quad \forall t, r \in T_\Sigma(X_{In}).\forall \alpha \in X_{In} \to A.\forall \beta \in X_{In} \to B.$$

$$I_\alpha(t) = I_\alpha(r) \Leftrightarrow I_\beta(t) = I_\beta(r) \wedge$$

$$\forall p, p' \in P_\Sigma(X_{In}).\forall \alpha \in X_{In} \to A.\forall \beta \in X_{In} \to B.$$

$$I_\alpha(p) \Leftrightarrow I_\alpha(p') \Leftrightarrow I_\beta(p) \Leftrightarrow I_\beta(p')$$

Proposition 5.12 *Let Σ be a signature and let In and Obs be two sets of sorts s.t. $In, Obs \subseteq Sorts(\Sigma)$. $\equiv_{Obs,In}$ is an equivalence relation.*

Proof:
In a very similar way as in [3].

Proposition 5.13 *Let Σ be a signature, let In and Obs be two sets of sorts s.t. $In, Obs \subseteq Sorts(\Sigma)$ and let A be a Σ-algebra. $\equiv_{Obs,In}$ is factorizable by $\approx_A^{Obs,In}$.*

Proof:
In a similar way as in [3].

5.3 The relativization functions

One possible way to define the functions which have to satisfy the behavioural relativization conditions is to define for any inclusion $i : \Sigma \hookrightarrow \Sigma'$, the inclusion $bihol[\Sigma] : \Sigma' \hookrightarrow bihol[\Sigma](\Sigma')$ with a disjoint copy of Σ' which we will denote as $Copy'(\Sigma')$ and to define the function

$$brelhol[i, bihol[\Sigma]] : \mathcal{P}(Sen_{BHOL}(\Sigma')) \rightarrow \mathcal{P}(Sen_{BHOL}(bihol[\Sigma](\Sigma')))$$

in such a way that if a $bihol[\Sigma](\Sigma')$-algebra A'' satisfies the set of sentences $brelhol[i, bihol[\Sigma]](\Phi)$, then A'' must also satisfy the following condition:

$$A''|_{Copy'(\Sigma)} \cong A''|_{\Sigma} / \approx^{Obs,In} \wedge A''|_{Copy'(\Sigma')} \models Copy'(\Phi)$$

This can be achieved by defining the set of sentences $brelhol[i, bihol[\Sigma]](\Phi)$ as the union of $Copy'(\Phi)$ with an axiomatization in higher-order logic of the observational equality $\approx^{Obs,In}$ and an axiomatization of a pseudo epimorphism between $A''|_{\Sigma}$ and $A''|_{Copy'(\Sigma)}$. This axiomatization requires extra symbols to denote the observational equality and the pseudo epimorphism. The axiomatization of the observational equality is based on [7] and the axiomatization of the pseudo epimorphism is based on [6].

For example, for the signature of a base specification $Container$ with sorts $Container, Elem, Nat$ and $Bool$, operations $\emptyset : Container$, $insert : Elem \times Container \rightarrow Container$, $union : Container \times Container \rightarrow Container, \ldots$ which will be denoted by $Contsign$ and set of sentences $Contax$ including

$$\forall s : Container.union\ \emptyset\ s\ =\ s$$

$$\forall s, s' : Container.(union\ (insert\ e\ s)\ s')(insert\ e\ (union\ s\ s'))$$

bihol[Contsign] would be defined as:

$$bihol[Contsign](Contsign)\ =\ Contsign \cup Copy(Contsign) \cup$$

$$\{\sim_s : s \times s \mid s \in Sorts(Contsign)\} \cup \{\pi_s : s \rightarrow Copy(s) \mid s \in Sorts(Contsign)\}$$

(where Copy(Contsign) is a disjoint copy of the signature $Contsign$, the symbols $\sim_s$ are used to denote the observational equality and the symbols π_s to denote a pseudoepimorphism), and the set of sentences $brelhol[i, bihol[Contsign]](Contax)$ will include the axioms $Contax$ appropiately renamed for the signature $Copy(Consign)$, axioms to define the indistinguishability relation $(Indist\_rel[Contsign[\sim], Obs, In])$ and axioms to establish a pseudo-epimorphism $(pEpi[bihol[Contsign], Obs, In])$ for given sorts Obs, In which determine the indistinguishability relation. These set of axioms are detailed in this section.

To define the functions which have to satisfy the quotient relativization conditions we proceed in a similar way defining for any inclusion $i : \Sigma \hookrightarrow \Sigma'$, the inclusion $qihol[\Sigma] : \Sigma' \hookrightarrow bihol[\Sigma](\Sigma')$ with also a disjoint copy of Σ' which we will also denote as $Copy'(\Sigma')$, and defining the function

$$qrelhol[i, qihol[\Sigma]] : \mathcal{P}(Sen_{BHOL}(\Sigma')) \to \mathcal{P}(Sen_{BHOL}(qihol[\Sigma](\Sigma')))$$

in such a way that if a $qihol[\Sigma](\Sigma')$-algebra A'' satisfies a set of sentences $qrelhol[i, qihol[\Sigma]](\Phi)$, then A'' must also satisfy the following condition:

$$A''|_\Sigma \cong A''|_{Copy'(\Sigma)} / \approx^{Copy'(Obs), Copy'(In)} \wedge A''|_{Copy'(\Sigma')} \models Copy'(\Phi)$$

This can be achieved in a symmetrical way as in the definition of the set of sentences $brelhol[i, bihol[\Sigma]](\Phi)$ by defining the set of sentences $qrelhol[i, qihol[\Sigma]](\Phi)$ as the union of $Copy'(\Phi)$ with an axiomatization in higher-order logic of the observational equality $\approx^{Copy(Obs), Copy(In)}$ (instead of $\approx^{Obs, In}$) and an axiomatization of a pseudo epimorphism between $A''|_{Copy'(\Sigma)}$ and $A''|_\Sigma$ (instead of a pseudo epimorphism between $A''|_\Sigma$ and $A''|_{Copy'(\Sigma)}$).

For the same signature $Contsign$ and the set of axioms $Contax$ presented above, qihol[Contsign] would be defined as

$$\begin{aligned} qihol[Contsign](Contsign) \; &= \; Contsign \cup Copy(Contsign) \cup \\ &\{\sim_{Copy(s)} : Copy(s) \times Copy(s) \mid s \in Sorts(Contsign)\} \cup \\ &\{\pi_{Copy(s)} : Copy(s) \to s \mid s \in Sorts(Contsign)\} \end{aligned}$$

and the set of sentences $qrelhol[i, qihol[Contsign]](Contax)$ will also include the axioms $Copy(Contax)$, axioms to define the indistinguishability relation $(Indist\_rel[Copy(Contsign[\sim]), Obs, In])$ and axioms to establish a pseudoepimorphism $(pEpi[qihol[Contsign], Obs, In])$.

See the rest of this subsection for a complete formal definition of the functions

$$brelhol[i, bihol[\Sigma]] : \mathcal{P}(Sen_{BHOL}(\Sigma')) \to \mathcal{P}(Sen_{BHOL}(bihol[\Sigma](\Sigma')))$$

and the functions

$$qrelhol[i, qihol[\Sigma]] : \mathcal{P}(Sen_{BHOL}(\Sigma')) \to \mathcal{P}(Sen_{BHOL}(qihol[\Sigma](\Sigma')))$$

for any inclusion $i : \Sigma \hookrightarrow \Sigma'$ and the next subsection for proofs that these functions satisfy the behavioural and quotient relativization conditions respectively.

Definition 5.14 *The relational signature $\Sigma[\sim]$ is defined for any signature $\Sigma \in |AlgSig|$ and for any $Sorts(\Sigma)$-set of new symbols $\sim$ as:*

$$\Sigma[\sim] = \Sigma \cup \{\sim_s : s \times s \mid s \in S\}$$

Definition 5.15 *The relational signature $\Sigma[\sim, \pi_{Copy}]$ for any signature $\Sigma \in |AlgSig|$, for any bijective signature morphism $Copy : \Sigma \to Copy(\Sigma)$ such that $\Sigma \cap Copy(\Sigma) = \emptyset$ and for any $Sorts(\Sigma)-$set of new symbols $\sim$ and π is defined as:*

$$\Sigma[\sim, \pi_{Copy}] = \Sigma[\sim] \cup Copy(\Sigma) \cup \{\pi_s : s \to Copy(s) | s \in S\}$$

Remark: *The relational signature $Copy(\Sigma)[\sim, \pi_{Copy-1}]$ stands for the following signature:*

$$Copy(\Sigma)[\sim, \pi_{Copy-1}] = Copy(\Sigma)[\sim] \cup \Sigma \cup \{\pi_s : Copy(s) \to s | s \in S\}$$

Definition 5.16 *The $Sorts(\Sigma)$-set of sentences $D[\Sigma, In]$ for any $In \subseteq Sorts(\Sigma)$ is defined as follows:*

$$D[\Sigma, In] = \{D_s[\Sigma, In] \mid s \in Sorts(\Sigma)\}$$

where

$$D_s[\Sigma, In] = \lambda x : s.(\forall P : [s].(\bigwedge_{f:s_1 \times \ldots \times s_n \to s \in Ops(\Sigma)} (\forall x_1 : s_1 . \ldots . \forall x_n : s_n .$$

$$(\bigwedge_{i \in [1..n], s_i = s} P(x_i) \supset P(f(x_1, \ldots, x_n))))) \supset P\ x) \qquad , if\ s \in S - In$$

$$D_s[\Sigma, In] = \lambda x : s.\mathbf{true} \qquad , if\ s \in In$$

Definition 5.17 *The set of sentences $Indist\_rel[\Sigma[\sim], Obs, In]$ is defined as follows:*

$$Indist\_rel[\Sigma[\sim], Obs, In] =$$

$$\{Indist\_rel[\Sigma[\sim], Obs, In, s] \mid s \in Sorts(\Sigma)\}$$

where

$$Indist\_rel[\Sigma[\sim], Obs, In, s] = \lambda x : s.\lambda y : s.\exists_{s' \in Sorts(\Sigma)} R_{s'} : [s', s'].$$

$$R_s(x, y) \;\wedge OBSEQ[(S, Op), R, Obs, In] \;\wedge$$

$$CONG[\Sigma, R, Obs, In] \;\wedge\; D_s[\Sigma, In](x) \;\wedge\; D_s[\Sigma, In](y)$$

$$OBSEQ[\Sigma, R, Obs, In] = \bigwedge_{obs \in Obs} \forall v : obs.\forall w : obs.$$

$$D_{obs}[\Sigma, In](v) \;\wedge D_{obs}[\Sigma, In](w) \;\Rightarrow (R_{obs}(v, w) \Leftrightarrow v = w)$$

$$CONG[\Sigma, R, Obs, In] =$$

$$\bigwedge_{f : s_1 \times \ldots \times s_n \rightarrow s \in Ops(\Sigma)} \forall x_1 : s_1.\forall y_1 : s_1. \;\ldots\; .\forall x_n : s_n.\forall y_n : s_n.$$

$$(R_{s1}(x_1, y_1) \;\wedge\; \ldots \;\wedge\; R_{sn}(x_n, y_n) \;\Rightarrow$$

$$R_s(f(x_1, \ldots, x_n), f(y_1, \ldots, y_n))) \wedge$$

$$\bigwedge_{p : s_1 \times \ldots \times s_n \in Prs(\Sigma)} \forall x_1 : s_1.\forall y_1 : s_1. \;\ldots\; .\forall x_n : s_n.\forall y_n : s_n.$$

$$(R_{s1}(x_1, y_1) \;\wedge\; \ldots \;\wedge\; R_{sn}(x_n, y_n) \;\Rightarrow$$

$$(p(x_1, \ldots, x_n) \;\Leftrightarrow\; p(y_1, \ldots, y_n))$$

Definition 5.18 *The set of sentences* $pEpi_{BHOL}[\Sigma[\sim, \pi_{Copy}], Obs, In]$ *is defined as:*

$$pEpi_{BHOL}[\Sigma[\sim, \pi_{Copy}], Obs, In] \;=\; Hom[\Sigma[\sim, \pi_{Copy}], Obs, In] \cup$$

$$Epihom[\Sigma[\sim, \pi_{Copy}], Obs, In] \;\cup\; \sim -comp[\Sigma[\sim, \pi_{Copy}], Obs, In]$$

where

$$Hom[\Sigma[\sim, \pi_{Copy}], Obs, In] = \bigcup_{f:s_1\times\ldots\times s_n\to s\in Ops(\Sigma)} \{\forall t_1 : s_1. \ldots .\forall t_n : s_n.$$

$$\pi_{Copy,s}(f(t_1,\ldots,t_n)) = Copy(f)(\pi_{Copy,s_1}(t_1),\ldots,\pi_{Copy,s_n}(t_n))\} \cup$$

$$\bigcup_{p:s_1\times\ldots\times s_n\to s\in Prs(\Sigma)} \{\forall t_1 : s_1. \ldots .\forall t_n : s_n.$$

$$\bigwedge_{i\in[1..n]} D_{s_i}[\Sigma, In](t_i) \;\Rightarrow$$

$$p(t_1,\ldots,t_n)) \;\Leftrightarrow\; Copy(p)(\pi_{Copy,s_1}(t_1),\ldots,\pi_{Copy,s_n}(t_n))\}$$

$$Epihom[\Sigma[\sim, \pi_{Copy}], Obs, In] =$$

$$\bigcup_{s\in S}\{\forall y : Copy(s).\exists x : s.D_s[\Sigma, In](x) \;\wedge\; \pi_{Copy,s}(x) = y)\}$$

$$\sim -comp[\Sigma[\sim, \pi_{Copy}], Obs, In] =$$

$$\bigcup_{s\in S}\{\forall x : s.\forall y : s.D_s[\Sigma, In](x) \;\wedge\; D_s[\Sigma, In](y) \Rightarrow$$

$$x \sim_s y \Leftrightarrow \pi_{Copy,s}(x) \;=\; \pi_{Copy,s}(y)\}$$

Definition 5.19 *For any inclusion* $i : \Sigma \hookrightarrow \Sigma'$ *in AlgSig, the inclusion* $bihol[\Sigma] : \Sigma' \hookrightarrow bihol[\Sigma](\Sigma')$ *is defined as follows:*

$$bihol[\Sigma](\Sigma') \;=\; Copy'(\Sigma') \,\cup\, \Sigma'[\sim, \pi_{Copy}]$$

where $Copy'$ *is a pushout morphism of the bijective signature morphism Copy and the inclusion* $i : \Sigma \hookrightarrow \Sigma'$ *such that*

$$Copy'(\Sigma') \cap \Sigma'[\sim, \pi_{Copy}] = Copy'(\Sigma')$$

Definition 5.20 *For any inclusion* $i : \Sigma \hookrightarrow \Sigma'$ *in AlgSig, the function*

$$brelhol[i, bihol[\Sigma]] : \mathcal{P}(Sen_{BHOL}(\Sigma')) \to \mathcal{P}(Sen_{BHOL}(bihol[\Sigma](\Sigma')))$$

is defined as follows:

$$brelhol[i, bihol[\Sigma]](\Phi) \ =$$

$$Copy'(\Phi) \ \cup \ pEpi_{BHOL}[\Sigma[\sim, \pi_{Copy}], Obs, In] \cup$$

$$\{\forall x : s.\forall y : s.\, x \sim_s y \ \Leftrightarrow \ Indist\_rel[\Sigma[\sim], Obs, In, s](x, y) \mid$$

$$s \in Sorts(\Sigma)\} \cup$$

$$\{\forall x : s.\forall y : s.\, x \sim_s y \ \Leftrightarrow \ x \ = \ y \mid s \in Sorts(\Sigma') - Sorts(\Sigma)\} \cup$$

$$\{\forall x : s.\, \pi_s(x) \ = \ x \mid s \in Sorts(\Sigma') - Sorts(\Sigma)\}$$

Definition 5.21 *For any inclusion* $i : \Sigma \hookrightarrow \Sigma'$ *in AlgSig, the inclusion* $qihol[\Sigma] : \Sigma' \hookrightarrow qihol[\Sigma](\Sigma')$ *is defined as follows:*

$$qihol[\Sigma](\Sigma') \ = \ Copy'(\Sigma') \ \cup \ Copy'(\Sigma')[\sim, \pi_{Copy-1}]$$

where $Copy'$ *is a pushout morphism of the bijective signature morphism Copy and the inclusion* $i : \Sigma \hookrightarrow \Sigma'$ *such that*

$$Copy'(\Sigma') \cap Copy'(\Sigma')[\sim, \pi_{Copy-1}] = Copy'(\Sigma')$$

Definition 5.22 *For any inclusion* $i : \Sigma \hookrightarrow \Sigma'$ *in AlgSig, the function*

$$qrelhol[i, qihol[\Sigma]] : \mathcal{P}(Sen_{BHOL}(\Sigma')) \rightarrow \mathcal{P}(Sen_{BHOL}(qihol[\Sigma](\Sigma')))$$

is defined as follows:

$$qrelhol[i, qihol[\Sigma]](\Phi) \ =$$

$$Copy'(\Phi) \ \cup \ pEpi_{BHOL}[Copy(\Sigma)[\sim, \pi_{Copy-1}], Copy(Obs), Copy(In)] \cup$$

$$\{\forall x : Copy(s).\forall y : Copy(s).\, x \sim_{Copy(s)} y \ \Leftrightarrow$$

$$Indist\_rel[Copy(\Sigma)[\sim], Copy(Obs), Copy(In)](x, y) \mid s \in Sorts(\Sigma)\} \cup$$

$$\{\forall x : Copy'(s).\forall y : Copy'(s).\, x \sim_{Copy'(s)} y \ \Leftrightarrow$$

$$x \ = \ y \mid s \in Sorts(\Sigma') - Sorts(\Sigma)\} \cup$$

$$\{\forall x : Copy'(s).\, \pi_{Copy'(s)}(x) \ = \ x \mid s \in Sorts(\Sigma') - Sorts(\Sigma)\}$$

Lemma 5.23 *If* $R_A^{Obs,In}$ *is a partial* Σ*-congruence which satisfies the following condition:*

$$\forall obs \in Obs.\forall v, w \in A_{obs}[X_{In}].(v \ R_{A,obs}^{Obs,In} \ w \ \ \Leftrightarrow \ \ v \ = \ w)$$

then

$$\forall s \in S.\forall v, w \in A[X_{In}].v\ R^{Obs,In}_{A,s}\ w \ \Rightarrow\ v \approx^{Obs,In}_{A,s} w$$

Proof:
It follows by context induction.The proof of the general case uses that R is af partial Σ-congruence which coincides with the set theoretical equality for observable sorts.

Lemma 5.24 *The sentence $Indist\_rel[\Sigma[\sim], Obs, In, s]$ for any sort $s \in Sorts(\Sigma)$ and for any free variables $x, y \in X_s$ satisfies the following condition which we will refer as the indistinguishability condition:*

$$\forall s \in S.\forall A \in |Alg(\Sigma)|.\forall \rho \in \{x, y\} \rightarrow A.$$

$$[\![Indist\_rel[\Sigma[\sim], Obs, In](x, y)]\!]_{\rho,A} \Leftrightarrow \rho(x) \approx^{Obs,In}_{A,s} \rho(y)$$

Proof:
See [7].

5.4 The institution $BHOL$

Theorem 5.25 *The algebraic institution BHOL extended with the following components:*

- *For each $\Sigma \in |AlgSig|$ and for a fixed but arbitrary sets $Obs, In \subseteq Sorts(\Sigma)$, the relational observational equality.*
- *For each inclusion $i : \Sigma \hookrightarrow \Sigma'$ in AlgSig, the inclusion $bihol[\Sigma] : \Sigma' \hookrightarrow bihol[\Sigma](\Sigma')$ and the function $brelhol[i, bihol[\Sigma]]$*
- *For each inclusion $i : \Sigma \hookrightarrow \Sigma'$ in AlgSig, the inclusion $qihol[\Sigma] : \Sigma' \hookrightarrow qihol[\Sigma](\Sigma')$ and the function $qrelhol[i, qihol[\Sigma]]$*

is a behavioural algebraic institution.

Proof:

We have to prove that

- for each $\Sigma \in |AlgSig|$ and for a fixed but arbitrary sets $Obs, In \subseteq Sorts(\Sigma)$, the relational observational equality $\approx^{Obs,In}_{A}$ is a family of partial Σ-congruences which follows by proposition 6.17.
- The behavioural satisfaction relation satisfies the behavioural satisfaction condition. See theorem 3.35 of [7].
- For each inclusion $i : \Sigma \hookrightarrow \Sigma'$ in $AlgSig$, the function $brelhol[i, bihol[\Sigma]]$ satisfies the behavioural relativization conditions.

– (1)): Assume that $A' \in |Alg(\Sigma')|$ and $\Phi \in \mathcal{P}(Sen_{BHOL}(\Sigma'))$, and let A be a Σ-algebra such that

$$A'|_{\Sigma} = A/\approx \wedge A' \models \Phi$$

To prove that

$$\exists A'' \in Alg(bi[\Sigma](\Sigma')).\, A''|_{\Sigma'} = A''' \ \wedge\ A'' \models brel[i, bi[\Sigma]](\Phi)$$

where the Σ'-algebra A''' is defined as follows:

$$\begin{array}{l} A'''|_{\Sigma} = A \\ A'''_s = A'_s \quad \text{for any sort } s \in Sorts(\Sigma') - Sorts(\Sigma) \\ f_{A'''} = f_{A'} \quad \text{for any sort } f \in Ops(\Sigma') - Ops(\Sigma) \\ P_{A'''} = P_{A'} \text{ for any sort } P \in Pr(\Sigma') - Pr(\Sigma) \end{array}$$

we will define a $bi[\Sigma](\Sigma')$-algebra A'' such that

$$A''|_{\Sigma'} = A''' \ \wedge\ A'' \models brel[i, bi[\Sigma]](\Phi)$$

The $bi[\Sigma](\Sigma')$-algebra A'' is defined as follows:

* $A''|_{\Sigma} = A$.
* $A''_s = A'_s$ for any sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$
* $f_{A''} = f_{A'}$ for any sort $f \in Ops(\Sigma') - Ops(\Sigma)$
* $P_{A''} = P_{A'}$ for any sort $P \in Pr(\Sigma') - Pr(\Sigma)$
* $A''|_{Copy(\Sigma)} = A/\!\approx_A^{Obs,In}|_{Copy-1}$.
* $A''_{Copy'(s)} = A'_s$ for any sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$
* $Copy'(f)_{A''} = f_{A'}$ for any sort $f \in Ops(\Sigma') - Ops(\Sigma)$
* $Copy'(P)_{A''} = P_{A'}$ for any sort $P \in Pr(\Sigma') - Pr(\Sigma)$
* $A''_{\pi_s} = \epsilon_{A,s}$ for every sort $s \in S$.
  where
  $\epsilon_{A,s}(v) = [v]_{\approx_{A,s}^{Obs,In}}$ for all $s \in Sorts(\Sigma)$.
* $A''_{\pi_s} = id_{A',s}$ for every sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$.
  where
  $id_{A,s}(v) = v$

* $A''_{\sim_s} = \approx^{Obs,In}_{A,s}$ for every sort $s \in Sorts(\Sigma)$.

* $A''_{\sim_{Copy'(s)}} = =$ for every sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$.

It is obvious that $A''|_{\Sigma'} = A'''$ and to prove that

$$A'' \models brelhol[i, bihol[\Sigma]](\Phi)$$

we have to prove that

* $A'' \models_{bihol[\Sigma](\Sigma'),BHOL} Copy'(\Phi)$ which holds by the satisfaction lemma since $\Phi \in \mathcal{P}(Sen_{BHOL}(\Sigma'))$, $A''|_{Copy'(\Sigma')} = A'|_{Copy'-1}$ and $A' \models_{\Sigma',BHOL} \Phi$.
* $A'' \models Indist\_rel[\Sigma[\sim], Obs, In]$ holds since for every sort $s \in Sorts(\Sigma)$ $A''_{\sim_s} = \approx^{In,Obs}_{A,s}$ and $Indist\_rel[\Sigma[\sim], Obs, In]$ satisfies the indistinguishability condition.
*

$$A'' \models_{bihol[\Sigma](\Sigma')} \{\forall x : Copy'(s). \forall y : Copy'(s).$$

$$x \sim_{Copy'(s)} y \;\Leftrightarrow\; x = y \mid s \in Sorts(\Sigma') - Sorts(\Sigma)\}$$

  since for every sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$ $A''_{\sim_s}$ is the set theoretical equality.
* $A'' \models pEpi[\Sigma[\sim, \pi_{Copy}]]$ holds since for every sort $s \in Sorts(\Sigma)$, A''_{π_s} can be seen as an epimorphism between A and $A/\approx^{Obs,In}_A$ and therefore A'' satisfies $Hom[\Sigma[\sim, \pi_{Copy}], Obs, In]$, $Epihom[\Sigma[\pi_{Copy}], Obs, In]$ and also $\sim -comp[\Sigma[\sim, \pi_{Copy}], Obs, In]$.
* $A'' \models \{\forall x : s.\, \pi_s(x) = x \mid s \in Sorts(\Sigma') - Sorts(\Sigma)\}$ since for every sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$ A''_{π_s} is the identity function.

– (2)): Assume that $A'' \in Alg(bihol[\Sigma](\Sigma'))$, $\Phi \in \mathcal{P}(Sen_{BHOL}(\Sigma))$ and $A'' \models brel[i, bi[\Sigma]](\Phi)$.
We have to show that

$$\exists A' \in |Alg(\Sigma')|.A'|_{\Sigma} = A''|_{\Sigma}/\approx \;\wedge A' \models \Phi$$

Since $A''|_{Copy'(\Sigma')} \models Copy'(\Phi)$ and because of the indistinguishability condition together with the definition of $pEpi[\Sigma[\sim, \pi_{Copy}]]$ we can deduce that

$$A''|_{Copy(\Sigma)} \cong A''|_{\Sigma}/\approx^{Obs,In}_A|_{Copy-1}$$

Finally, by proposition 5.2, the abstract satisfaction condition and the satisfaction condition of the institution we can prove our goal.

• For each inclusion $i : \Sigma \hookrightarrow \Sigma'$ in $AlgSig$, the function $qrelhol[i, qihol[\Sigma]]$ satisfies the quotient relativization conditions.

–

$$(1)\forall A' \in |Alg(\Sigma')|.\forall \Phi \in \mathcal{P}(Sen_{BHOL}(\Sigma')).\, A' \models \Phi \Rightarrow$$

$$\exists A'' \in Alg(qihol[\Sigma](\Sigma')).\, A''|_\Sigma \,\cong\, A'|_\Sigma/\approx \,\wedge A'' \models qrelhol[\Sigma](\Sigma')(\Phi)$$

Assume that $A' \in |Alg(\Sigma')|$, $\Phi \in \mathcal{P}\,(Sen_{BHOL}(\Sigma'))$ and $A' \models_{\Sigma',BHOL}$ Φ. To prove that

$$\exists A'' \in Alg(qihol[\Sigma](\Sigma')).\, A''|_\Sigma \,\cong\, A'|_\Sigma/\approx \,\wedge$$

$$A'' \models_{\Sigma',BHOL} qrelhol[i, qihol[\Sigma]](\Phi)$$

we will define an algebra A'' such that

$$A''|_\Sigma \,\cong\, A'|_\Sigma/\approx \,\wedge A'' \models qrelhol[i, qihol[\Sigma]](\Phi)$$

The algebra A'' is defined as follows:

* $A''|_\Sigma = A'|_\Sigma/\approx_A^{Obs,In}$.
* $A''_s = A'_s$ for any sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$
* $f_{A''} = f_{A'}$ for any sort $f \in Ops(\Sigma') - Ops(\Sigma)$
* $P_{A''} = P_{A'}$ for any sort $P \in Pr(\Sigma') - Pr(\Sigma)$
* $A''|_{Copy'(\Sigma')} = A'|_{Copy'-1}$.
* $A''_{\pi_{Copy'(s)}} = \epsilon_{A'|_{Copy'-1},Copy'(s)}$ for every sort $s \in Sorts(\Sigma)$.
  where
  $\epsilon_{A,Copy'(s)}(v) = [v]_{\approx_{A,Copy'(s)}^{Copy'(Obs),Copy'(In)}}$
* $A''_{\pi_{Copy'(s)}} = id_{A'|_{Copy'-1},Copy'(s)}$
  for every sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$.
  where
  $id_{A,s}(v) = v$
* $A''_{\sim_{Copy'(s)}} = \approx_{A'|_{Copy'-1}|_{Copy(\Sigma)},Copy'(s)}^{Copy'(In),Copy'(Obs)}$ for every sort $s \in Sorts(\Sigma)$.
* $A''_{\sim_{Copy'(s)}} = =$ for every sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$.

It is obvious that $A''|_\Sigma \,\cong\, A'|_\Sigma/\approx$ since $A''|_\Sigma = A'|_\Sigma/\approx$.
To prove that

$$A'' \models_{\Sigma',BHOL} qrelhol[i, qihol[\Sigma]](\Phi)$$

we have to prove that

* $A'' \models_{qi[\Sigma](\Sigma'),BHOL} Copy'(\Phi)$ which holds by the satisfaction lemma since $\Phi \in \mathcal{P}(Sen_{BHOL}(\Sigma'))$, $A''|_{Copy'(\Sigma')} = A'|_{Copy'-1}$ and $A' \models_{\Sigma',BHOL} \Phi$.
* $A'' \models Indist\_rel[Copy(\Sigma)[\sim], Copy(Obs), Copy(In)]$ holds since for every sort $s \in Sorts(\Sigma)$ $A''_{\sim_{Copy(s)}} = \approx^{Copy(In),Copy(Obs)}_{A'|_{Copy'-1},Copy(s)}$ and $Indist\_rel[Copy(\Sigma)[\sim], Copy(Obs), Copy(In)]$ satisfies the indistinguishability condition.
*
$$A'' \models_{qihol[\Sigma](\Sigma'),BHOL} \{\forall x : Copy'(s).\forall y : Copy'(s).$$
$$x \sim_{Copy'(s)} y \Leftrightarrow x = y \mid s \in Sorts(\Sigma') - Sorts(\Sigma)\}$$
since for every sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$ $A''_{\sim_{Copy(s)}}$ is the set theoretical equality.
* $A'' \models pEpi[Copy(\Sigma)[\sim, \pi_{Copy-1}]]$ holds since for every sort $s \in Sorts(\Sigma)$, $A''_{\pi_{Copy(s)}}$ can be seen as an epimorphism between $A'|_\Sigma$ and $A'|_\Sigma/\approx^{Obs,In}_A$ and therefore A'' satisfies $Hom[\Sigma[\sim, \pi_{Copy}], Obs, In]$, $Epihom[\Sigma[\pi_{Copy}], Obs, In]$ and also $\sim -comp[\Sigma[\sim, \pi_{Copy}], Obs, In]$.
* $A'' \models \{\forall x : Copy'(s).\, \pi_{Copy'(s)}(x) = x \mid s \in Sorts(\Sigma') - Sorts(\Sigma)\}$ since for every sort $s \in Sorts(\Sigma') - Sorts(\Sigma)$ $A''_{\pi_{Copy'(s)}}$ is the identity function.

–

$$(2)\forall A'' \in Alg(qihol[\Sigma](\Sigma')).\forall \Phi \in \mathcal{P}(Sen_{BHOL}(\Sigma')).$$
$$A'' \models_{\Sigma,BHOL} qrelhol[i, qihol[\Sigma]](\Phi) \Rightarrow$$
$$\exists A' \in |Alg(\Sigma')|.A''|_\Sigma \cong A'|_\Sigma/\approx \wedge A' \models_{\Sigma',BHOL} \Phi$$

Assume that $A'' \in Alg(qihol[\Sigma](\Sigma'))$, $\Phi \in \mathcal{P}(Sen_{BHOL}(\Sigma'))$, and
$$A'' \models_{\Sigma,BHOL} qrelhol[i, qihol[\Sigma]](\Phi)$$
By the definition of $qrelhol[\Sigma](\Sigma')(\Phi)$ we know that
$A''|_{Copy'(\Sigma')}|_{Copy'} \models_{\Sigma',BHOL} \Phi$
By the definitions of $Indist\_rel[Copy(\Sigma)[\sim], Copy(Obs), Copy(In)]$ and $pEpi[Copy(\Sigma)[\sim, \pi_{Copy-1}], Copy(Obs), Copy(In)]$
we have that $A''|_{Copy(\Sigma)}|_{Copy}/\approx^{Obs,In}_A \cong A''|_\Sigma$ and therefore $A''|_\Sigma \in Models(SP/\approx^{Obs,In}_A)$.

6 Conclusions

In this paper, we have generalised the semantics of the operators of ASL presented in [?] including three behavioural operators for a fixed but arbitrary algebraic institution. Then, we have defined behavioural algebraic institutions which are useful to define an alternative semantics of the behavioural operator which is equivalent to the one presented in [7] and to define the normal form of the behavioural operators for both semantics. Normal forms can be used to define a certain kind of non-structured proof systems which are defined for example in [6].

Finally, we present a concrete higher-order behavioural algebraic institution using the higher-order institution presented in [7]

References

[1] M. Bidoit and A. Tarlecki. Behavioural satisfaction and equivalence in concrete model categories. In *Proc. CAAP'96 Trees in Algebra and Programming (LNCS 1059)*, 1996.

[2] Michel Bidoit, María Victoria Cengarle, and Rolf Hennicker. Proof systems for structured specifications and their refinements. Chapter 10 of the book Algebraic Foundations of Systems Specification.(To appear).

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

[4] R. Diaconescu, J. Goguen, and P. Stefaneas. Logical support for modularisation. In G. Huet and G. Plotkin, editors, *Logical Environments*. Cambridge University Press, 1991. Workshop on Logical Frameworks (Edinburgh).

[5] Joseph A Goguen and Rod Burstall. INSTITUTIONS: Abstract model theory for specification and programming. *Journal of the Assoc. for Computing Machinery*, 39(1):95–146, 1992.

[6] Rolf Hennicker. *Structured Specifications with Behavioural Operators: Semantics, Proof Methods and Applications.* Habilitationsschrift, Institut für Informatik, Ludwig-Maximilians-Universität München, June 1997.

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

[8] Andrzej Tarlecki. Institutions:an abstract framework for formal specifications. Chapter 3 of the book Algebraic Foundations of Systems Specification (Springer).(To appear).

[9] Andrzej Tarlecki. On the existence of free models in abstract algebraic institutions. *Theoretical Computer Science*, 37, 1985.