Stability of termination and sufficient-completeness under pushouts via amalgamation

Daniel Găină[a,b], Masaki Nakamura[c], Kazuhiro Ogata[d], Kokichi Futatsugi[e,f]

[a] *Kyushu University*
[b] *La Trobe University*
[c] *Toyama Prefectural University*
[d] *Japan Advanced Institute of Science and Technology*
[e] *National Institute of Advanced Industrial Science and Technology*
[f] *National Institute of Informatics*

Abstract

In the present study, we provide conditions for the existence of pushouts of signature morphisms in constructor-based order-sorted algebra, and then we prove that reducibility and termination of term rewriting systems are closed under pushouts. Under the termination assumption, reducibility is equivalent to sufficient-completeness, which is crucial for proving several important properties in computing for constructor-based logics such as completeness, existence of initial models and interpolation. In logic frameworks that are not based on constructors, sufficient-completeness is essential to establish the soundness of the induction schemes which are based on some methodological constructor operators. We discuss the application of our results to the instantiation of parameterized specifications.

Keywords: algebraic specification, sufficient-completeness, reducibility, pushout, parameterized specification, term rewriting

1. Introduction

Pushouts. The basic mathematical mechanism that supports the instantiation of parameterized specifications and the composition of module specifications is the pushout construction in the category of signature morphisms. This approach was introduced for many-sorted algebraic specifications (see e.g. [1]) and then it was generalized to order-sorted algebra (see e.g. [2] and [3]). The logic framework of order-sorted algebra (**OSA**) allows the definition of subsorts, which in turn, gives the specifiers the possibility to treat partial functions in a clean way. **OSA** is also suitable for dealing with overloading which allows a single symbol to be used for several different operations. Another important practical aspect covered by **OSA** is error handling, which roughly means that some expressions that do not type check but may have a meaningful value, can be used in reasonings without any danger of obtaining inconsistent results. One of the main results of this paper consists of the pushout construction of signature morphisms in constructor-based order-sorted algebra (**COSA**).

Constructor-based logics. Constructor-based logics [4, 5] are obtained from a base logic by enhancing the syntax with a subsignature of constructor operators and restricting the semantics to models that are reachable by constructors. The logics with constructors are more expressive than their base versions but the expressivity comes with a cost. Some logic properties useful in computing such as completeness, compactness, the existence of initial models and interpolation, are lost or hold under additional requirements. Notice that the completeness property is studied in connection with some system of proof rules that break the initial goal into a finite number of subgoals. For this reason, completeness usually implies compactness. In [6], a subset of the authors proved a quasi completeness result for constructor-based logics. The result in [6] uses a proof rule scheme with an infinite number of premises. This completeness result does not imply compactness but it provides a starting point for defining systems of (finitary) proof rules that can be used in formal verification of hardware and software systems [7, 8]. The existence of initial models was proven in [9] and it holds by adding additional hypotheses to the result for the base logic. Interpolation

in logic with constructors was studied in [10] and [11]. The results enumerated above are based on the sufficient-completeness property. A theory is sufficient-complete if all terms of constrained sorts can be 'reduced' to terms that contain constructors and elements (i.e. operators and/or variables) of loose sorts. [1]

Amalgamation. The second main contribution of the present work is the amalgamation property for the pushouts of (constructor-based) order-sorted signatures. Amalgamation is defined as a limit preservation property of the model functor, and it is required by almost all model-theoretic developments in algebraic specifications [12]. In this paper, amalgamation is used to prove the closure of termination under pushouts of signatures. Pushouts of order-sorted signatures do not have the amalgamation property, in general [13]. In this paper, we provide sufficient conditions for a pushout of order-sorted signatures to enjoy amalgamation. This result is straightforwardly extended to constructor-based order-sorted signatures.

Reducibility, termination and sufficient-completeness. In the present work, we consider only basic specifications which are pairs $\mathsf{SP} = \langle \Sigma_{\mathsf{SP}}, \Gamma_{\mathsf{SP}} \rangle$ consisting of a signature Σ_{SP} and a set of sentences Γ_{SP} over Σ_{SP}. The set of sentences Γ_{SP} can be used to reason formally about the class of models satisfying Γ_{SP}. Therefore, Γ_{SP} provides an operational semantics for the specification SP since the universally quantified equations can be regarded as oriented term rewrite rules and may be used to compute. We define a notion of reducibility that is equivalent with sufficient-completeness assuming that the underlying specification is terminating. The advantage is that reducibility is decidable for unconditional term rewriting systems and some classes of conditional term rewriting systems (see, for example, [14, 15]) unlike sufficient-completeness, which is known to be undecidable, in general. On the other hand, there are several classes of term rewriting systems for which termination is decidable such as those whose termination can be proved by the recursive path ordering and the dependency pair method [14], the well-founded order-sorted recursive theories [16] and so on. In this paper, we interpret terms into well-founded domains to show termination.

Little seems to be known about the modularity of sufficient-completeness, and the modularity results we are aware of do not deal with sorts and subsorts except for [16] and [15]. The third main contribution of the present work is the closure of reducibility and termination under pushouts of **COSA** signatures. Since reducibility is equivalent to sufficient-completeness provided that termination holds, we obtain the closure of sufficient-completeness under pushouts, too. This result is applicable to the instantiation of parameterized specifications. If both the parameterized specification and the instance of the parameter are reducible/terminating, the specification obtained by instantiation is reducible/terminating. It is worth noting that we prove the closure of reducibility and termination under pushouts for conditional term rewriting systems modulo a set of equations which include: associativity, commutativity, identity axioms, as well as some *rotative* and *reversible* laws necessary for describing ring-shaped networks used in mobile robot algorithms [17]. This means that our results are applicable to order-sorted specifications with universally quantified conditional equations for which some of the binary operations can be associative and/or commutative.

Related Work. Colimits of signature morphisms and theories have been playing a very important role in algebraic specification [18, 19]. By contrast, theory limits seem to be much less important in applications. Colimits of algebraic signature morphisms are a straightforward generalization of colimits in the category of sets (see e.g. [12]). The pushout construction in the category of **OSA** signatures is much more difficult than for ordinary algebraic signatures [2]. In [9], a subset of the authors proved the existence of pushouts in the category of constructor-based algebraic signatures. In this paper, we generalize the result to constructor-based order-sorted signatures. The definition of **COSA** signature morphism presented in this paper is more general than the one used in [6] in the sense that it captures a wider class of arrows. Choosing the appropriate level of generalization for the notion of signature morphisms in **COSA** is essential for proving the existence of pushouts.

The classical notion of sufficient-completeness in term rewriting (e.g. [20, 21, 22, 16]) states that any ground term can be reduced to a term built from constructors. Our notion of sufficient-completeness generalizes the standard definition by replacing ground terms with terms that have variables of loose sorts. This is necessary since in applications it often happens to have both, sorts with constructors and sorts with no constructors. Therefore, in such scenarios the standard definition is not applicable, while our theoretical infrastructure can be successfully used.

[1] In constructor-based logics, the sorts of constructors are called constrained and a sort with no constructors is called loose.

An incremental proof of sufficient-completeness for *fair extensions* of theories is given in [16] in the context of order-sorted unconditional rewriting modulo associativity and commutativity. The basic idea there is that extensions do not interfere with the base theory. An incremental proof of sufficient-completeness can be found also in [15] in the context of order-sorted conditional rewriting. This approach is closer to the present contribution in the sense that it is based on the reducibility of *basic terms* [22]. The aforementioned results are applicable to specification imports. The present contribution is applicable to parameterization, which is one of the most important features to make specifications modular and reusable, and our best hope for scaling up formal verification process to complex systems. Another distinctive feature of our modular approach is the fact that the results are somehow independent of the internal structure of the term rewriting systems that are composed by instantiation, i.e. no restrictions on the term rewriting rules are needed. A model-theoretic approach to the closure of sufficient-completeness under pushouts of signature morphisms can be found in [10] in the context of constructor-based many-sorted algebra. In contrast, the interest here is not about the denotational semantics of parameterized order-sorted specifications, but it is about the operational semantics by term rewriting.

2. Preliminaries

There are several non-equivalent axiomatizations of **OSA** in the literature including the ones proposed by Goguen and Meseguer [23] and by Poigné [24]. The version we consider here originates in [3], and it enjoys better mathematical properties than the ones enumerated before (see [13] for a comparison).

2.1. Order-sorted algebra

In this section we recall the basic definitions of **OSA**.

Definition 1 (**OSA** signatures)**.** *An order-sorted (algebraic) signature is a triple* $(S, \leq, F)$ *with* $(S, \leq)$ *a poset and* (S, F) *a many-sorted algebraic signature.*

Notice that $F = \{F_{w\to s}\}_{(w,s)\in S^*\times S}$ is a $(S^* \times S)$-indexed set of function symbols. F can be understood also as a set of elements of the form $\sigma\colon w \to s$, where σ is the name of a function, w is its arity and s is its sort. Therefore, we may write $\sigma \in F_{w\to s}$ or $\sigma\colon w \to s \in F$; both have the same meaning. Generally, w ranges over arities, which are understood here as strings of sorts; in other words an arity gives the number of arguments together with their sorts. The set $S/_{\equiv_\leq}$ of connected components of $(S, \leq)$ is the quotient of S under the equivalence relation $\equiv_\leq$ generated by $\leq$. For the sake of simplicity, we let $[S]$ denote $S/_{\equiv_\leq}$, and $[s]$ denote $s/_{\equiv_\leq}$ for all sorts $s \in S$. The equivalence $\equiv_\leq$ can be extended to strings of elements from S in the usual way.

Definition 2. *An order-sorted signature is* sensible *if for any function symbols* $\sigma\colon w \to s$ *and* $\sigma\colon w' \to s'$ *we have* $w \equiv_\leq w'$ *implies* $s \equiv_\leq s'$.

The notion of sensible signature is less restrictive than the notion of pre-regularity [23], and it provides sufficient conditions to avoid excessive ambiguities. We recall two types of overloading supported by sensible signatures:

1. *subsort overloading*, where the same function symbol is used with related typings in the subsort ordering, which means there exist $\sigma\colon w \to s$ and $\sigma\colon w' \to s'$ such that $w \equiv_\leq w'$, and

2. *ad-hoc overloading*, where the same function symbol is used with typings that are unrelated in the subsort ordering, which means there exist $\sigma\colon w \to s$ and $\sigma\colon w' \to s'$ such that w and w' have same length but $w \not\equiv_\leq w'$.

Lemma 3. *Any term over a sensible order-sorted signature has a unique connected component.*

Proof. Straightforward, by induction. □

Notation 1. *Let* $(S, \leq, F)$ *be an order-sorted signature.*

- *For any connected components* $[s_0], [s_1], \ldots, [s_n] \in [S]$*, we let* $[\sigma]\colon [s_1] \ldots [s_n] \to [s_0]$ *denote the set* $\{\sigma\colon s'_1 \ldots s'_n \to s'_0 \mid s'_i \in [s_i], 0 \leq i \leq n\}$ *of 'subsort polymorphic' operators with name* σ *for those components;*

- *Let* $[F] = \{[\sigma]\colon [w] \to [s] \mid (w, s) \in S^* \times S$ *and* $(\sigma\colon w \to s) \in F\}$.

The order-sorted signature morphisms are algebraic signature morphisms that preserve the subsort relation and the subsort overloading.

Definition 4 (**OSA** signature morphisms)**.** *The category* $\mathtt{Sig}^{\mathbf{OSA}}$ *of order-sorted signatures is defined as follows:*

1. *The objects are order-sorted signatures introduced in Definition 1.*
2. *The arrows are order-sorted signature morphisms* $\varphi\colon (S, \leq, F) \to (S', \leq', F')$*, i.e. many-sorted algebraic signature morphisms* $\varphi\colon (S, F) \to (S', F')$ *such that the function* $\varphi\colon (S, \leq) \to (S', \leq')$ *is monotonic and* φ *maps the operator symbols in each subsort polymorphic family* $[\sigma]\colon [s_1] \ldots [s_n] \to [s_0]$ *to a subset of the set* $[\varphi(\sigma)]\colon [\varphi(s_1)] \ldots [\varphi(s_n)] \to [\varphi(s_0)]$.

Notice that for each order-sorted signature there exists an isomorphic order-sorted signature which is sensible. Therefore, throughout this paper we consider only order-sorted signatures which are sensible.

In the following we define some notions related to signature morphisms necessary to prove our results.

Definition 5 (Reflection & encapsulation)**.** *Let* $\varphi\colon (S, \leq, F) \to (S', \leq', F')$ *be an order-sorted signature morphism.*

1. φ reflects the preorder over sorts *iff for all sorts* $s' \in S'$ *and* $sr \in S$ *such that* $s' \leq' \varphi(sr)$ *there exists* $s \in S$ *such that* $\varphi(s) = s'$ *and* $s \leq sr$.
2. φ encapsulates *operators iff for all sorts* $s \in S$*, arities* $w' \in S'^*$ *and function symbols* $\sigma'\colon w' \to \varphi(s) \in F'$ *there exists* $\sigma\colon w \to s \in F$ *such that* $\varphi(\sigma\colon w \to s) = \sigma'\colon w' \to \varphi(s)$.

The notion of encapsulation is used extensively throughout this paper especially in regard to *constructors*.

Definition 6 (**OSA** sentences)**.** *Let* $(S, \leq, F)$ *be an order-sorted signature.*

1. *The set of sentences* $\mathtt{Sen}^{\mathbf{OSA}}(S, \leq, F)$ *consists of universally quantified conditional equations* $\forall X \cdot t_0 = t_0'$ *if* $t_1 = t_1' \wedge \ldots \wedge t_n = t_n'$*, where* X *is a finite set of variables for* $(S, \leq, F)$ *and* t_i *and* t_i' *are terms with variables from* X *such that the sorts of* t_i *and* t_i' *are in the same connected component.*
2. *Given a signature morphism* $\varphi\colon (S, \leq, F) \to (S', \leq', F')$*, the sentence translation* $\mathtt{Sen}^{\mathbf{OSA}}(\varphi)\colon \mathtt{Sen}^{\mathbf{OSA}}(S, \leq, F) \to \mathtt{Sen}^{\mathbf{OSA}}(S', \leq', F')$ *along* φ *is symbol-wise.*

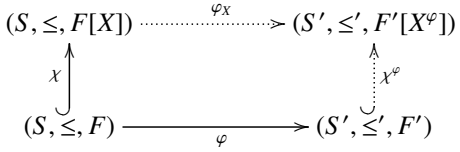


Figure 1: Quantification pushout

One subtle topic is the translation of quantified sentences along signature morphisms. The idea is to avoid clashes of variables translated along the signature morphisms with the symbols from the target signature. Consider a signature morphism $\varphi\colon (S, \leq, F) \to (S', \leq', F')$ and a set of variables X for $(S, \leq, F)$. If $x \in X$ is a constant in F' then the translation of a sentence $\forall x \cdot \gamma \in \mathtt{Sen}^{\mathbf{OSA}}(S, \leq, F)$ along φ may be problematic. To avoid inconsistencies, a variable for $(S, \leq, F)$ is a triple $(x, s, (S, \leq, F))$ consisting of a name x, a sort s and a signature $(S, \leq, F)$. One can easily define a pushout as shown in Figure 1 such that

- $X^\varphi = \{(x, \varphi(s), (S', \leq' F')) \mid (x, s, (S, \leq, F)) \in X\}$ is a set of variables for $(S', \leq', F')$;
- $F[X]$ and $F'[X^\varphi]$ are obtained by adding the variables from X and X^φ as constants to F and F', respectively;
- χ and χ^φ are inclusions of signatures, and

- $\varphi_X\colon (S, \leq, F[X]) \to (S', \leq', F'[X^\varphi])$ is the canonical extension of φ mapping each $(x, s, (S, \leq, F)) \in X$ to $(x, \varphi(s), (S', \leq', F')) \in X^\varphi$.

Notice that Definition 6 defines a functor $\mathtt{Sen}^{\mathbf{OSA}}\colon \mathtt{Sig}^{\mathbf{OSA}} \to \mathbb{S}et$ such that $\mathtt{Sen}^{\mathbf{OSA}}(\varphi)(\forall X \cdot \gamma) = \forall X^\varphi \cdot \mathtt{Sen}^{\mathbf{OSA}}(\varphi_X)(\gamma)$ for all $\gamma \in \mathtt{Sen}^{\mathbf{OSA}}(S, \leq, F[X])$. When there is no danger of confusion, we may identify a variable $(x, s, (S, \leq, F)) \in X$ only by its name and sort (x, s), or, simply, by its name x.

As expected, order-sorted algebras are generalizations of many-sorted algebras that give a coherent semantics to subsort polymorphism.

Definition 7 (**OSA** models)**.** *For any* **OSA** *signature* $(S, \leq, F)$*, the category of order-sorted algebras* $\mathtt{Mod}^{\mathbf{OSA}}(S, \leq, F)$ *is defined as follows:*

1. *The objects are order-sorted* $(S, \leq, F)$*-algebras* A*, i.e. many-sorted* (S, F)*-algebras* A *such that*
   - $A_s \subseteq A_{s'}$ *if* $s \leq s'$*, and*
   - $A_{(\sigma\colon w \to s)}(a) = A_{(\sigma\colon w' \to s')}(a)$ *for all function symbols* $(\sigma\colon w \to s) \in F$ *and* $(\sigma\colon w' \to s') \in ([\sigma]\colon [w] \to [s])$ *and any element* $a \in A_w \cap A_{w'}$*.* [2]
2. *The arrows are order-sorted homomorphisms* $h\colon A \to B$*, i.e. many-sorted algebraic homomorphisms* $h\colon A \to B$ *such that* $h_s(a) = h_{s'}(a)$ *for all sorts* $s, s' \in S$ *with* $[s] = [s']$ *and all elements* $a \in A_s \cap A_{s'}$*.*

For any signature morphism $\varphi\colon (S, \leq, F) \to (S', \leq', F')$, the reduct model functor $\_\restriction_\varphi\colon \mathtt{Mod}^{\mathbf{OSA}}(S', \leq', F') \to \mathtt{Mod}^{\mathbf{OSA}}(S, \leq, F)$ is defined in the usual way.

- on objects $A' \in |\mathtt{Mod}^{\mathbf{OSA}}(S', \leq', F')|$: $(A'\restriction_\varphi)_s = A'_{\varphi(s)}$ for all sorts $s \in S$ and $(A'\restriction_\varphi)_{\sigma\colon w \to s} = A'_{\varphi(\sigma)\colon \varphi(w) \to \varphi(s)}$ for all operators $\sigma\colon w \to s \in F$.
- on homomorphisms $h'\colon A' \to B' \in \mathtt{Mod}^{\mathbf{OSA}}(S', \leq', F')$: $(h'\restriction_\varphi)_s = h'_{\varphi(s)}$ for all sorts $s \in S$.

We overload the notation, and for all signature morphisms $\varphi\colon (S, \leq, F) \to (S', \leq', F')$, we let φ denote also the extension of $\varphi\colon (S, \leq, F) \to (S', \leq', F')$ to

- terms $\varphi^{term}\colon T_{(S, \leq, F)} \to T_{(S', \leq', F')}\restriction_\varphi$, and
- sentences $\mathtt{Sen}^{\mathbf{OSA}}(\varphi)\colon \mathtt{Sen}^{\mathbf{OSA}}(S, \leq, F) \to \mathtt{Sen}^{\mathbf{OSA}}(S', \leq', F')$.

2.2. *Constructor-based order-sorted algebra*

Variants of **COSA** can be found in [6] and [25]. The definition of constructor-based signature morphism given in the present contribution is more general than in [6] or [25]. **COSA** signatures are obtained from **OSA** signatures by adding constructor operators.

Definition 8 (**COSA** signatures)**.** *A constructor-based order-sorted signature* $(S, \leq, F, F^c)$ *consists of an order-sorted signature* $(S, \leq, F)$ *and a subfamily of operator symbols* $F^c \subseteq F$ *called constructors.*

Throughout this paper we let Σ, Σ' and Σ_i to range over constructor-based order-sorted signatures of the form $(S, \leq, F, F^c)$, $(S', \leq', F', F'^c)$ and $(S_i, \leq_i, F_i, F_i^c)$, respectively, where i is a natural number.

Notation 2. *Given a signature* $\Sigma = (S, \leq, F, F^c)$*, we denote by*

- $S^{cs} = \{s \in S \mid$ *there exist* $w \in S^*$ *and* $\sigma \in F^c_{w \to s}\}$ *the set of* constrained *sorts,*
- $S^{ls} = S \setminus S^{cs}$ *the set of* loose *sorts,*
- $F^{cs} = \{\sigma \in F_{w \to s} \mid w \in S^*$ *and* $s \in S^{cs}\}$ *the family of operators on constrained sorts,*

[2] We denote by $A_{(\sigma\colon w \to s)}$ or σ^A the interpretation of a function symbol $\sigma\colon w \to s$ into an order-sorted algebra A.

- $F^{ls} = \{\sigma \in F_{w\to s} \mid w \in S^* \text{ and } s \in S^{ls}\}$ *the family of operators on loose sorts,*
- $\Sigma^c = (S, \leq, F^c \cup F^{ls})$ *the signature of constructors.*

In applications, it often happens to have both sorts with constructors and sorts without constructors as illustrated by the following simple example.

Example 9. *Let* LIST *be the signature defined in the following diagram.*

```
signature LIST
sorts Elt NeList List
subsort NeList < List
op empty :  -> List {constr}
op _;_ :  Elt List -> NeList {constr}
op map :  List -> List
op f :  Elt -> Elt
```

Then:

- $S^{cs}_{\texttt{LIST}} = \{\texttt{NeList}, \texttt{List}\}$, $S^{ls}_{\texttt{LIST}} = \{\texttt{Elt}\}$,
- $F^c = \{(\texttt{empty} :\to \texttt{List}), (\_;\_ : \texttt{Elt List} \to \texttt{NeList})\}$,
- $F^{cs}_{\texttt{LIST}} = \{(\texttt{empty} :\to \texttt{List}), (\_;\_ : \texttt{Elt List} \to \texttt{NeList}), (\texttt{map}\colon \texttt{List} \to \texttt{List})\}$,
- $F^{ls}_{\texttt{LIST}} = \{\texttt{f} : \texttt{Elt} \to \texttt{Elt}\}$.

The role played by each operator can be understood only in the presence of the following two equations:

- $\texttt{map}(\texttt{empty}) = \texttt{empty}$, *and*
- $\forall \texttt{E} : \texttt{Elt}, \texttt{L} : \texttt{List} \cdot \texttt{map}(\texttt{E} : \texttt{Elt}; \texttt{L} : \texttt{List}) = \texttt{f}(\texttt{E} : \texttt{Elt}); \texttt{map}(\texttt{L} : \texttt{List})$.

An application of `map` to a list has the result of applying the function `f` to each element of the given list. Intuitively, the constrained operator `map` is fully defined with respect to the constructors empty list and semicolon regardless of the definition of the loose operator `f`. Therefore the signature of constructors contains not only constructors but also loose operators.

Given a signature $\Sigma = (S, \leq, F, F^c)$ and a set of variables X for Σ, we let T_Σ and $T_\Sigma(X)$ denote the order-sorted algebras of terms $T_{(S,\leq,F)}$ and $T_{(S,\leq,F)}(X)$, respectively. We introduce terminology to describe some classes of terms.

Definition 10 (Classes of terms)**.** *Let* Σ *be a* **COSA** *signature.*

- *A constrained term* t *over* Σ *is a* $(S, \leq, F)$*-term free of variables of constrained sorts, that is* $t \in T_\Sigma(X)$ *for some set of variables of loose sorts* X*.*
- *A constructor term* t *over* Σ *is a constrained term over* Σ *free of operators from* $F^{cs} \setminus F^c$*, that is* $t \in T_{\Sigma^c}(X)$ *for some set of variables of loose sorts* X*.*
- *A basic term [22] is a term of the form* $t = \sigma(t_1, \ldots, t_n)$*, where* $\sigma \in F^{cs} \setminus F^c$ *and* t_i *are constructor terms.*

The main ingredients for defining **COSA** signature morphisms have been defined.

Definition 11 (**COSA** signature morphisms)**.** *The category* $\texttt{Sig}^{\textbf{COSA}}$ *of constructor-based order-sorted signature morphisms is defined as follows:*

1. *the objects are constructor-based order-sorted signatures introduced in Definition 8;*
2. *the arrows* $\varphi\colon \Sigma \to \Sigma'$ *are* **OSA** *signature morphisms* $\varphi\colon (S, \leq, F) \to (S', \leq', F')$ *such that*
   (a) *constructors are preserved, i.e.* $\varphi(F^c) \subseteq F'^c$*,*

(b) *constructors are quasi encapsulated, i.e. for all* $s_c \in S^{cs}$ *and* $\sigma' : w' \to \varphi(s_c) \in F'^c$ *there exists* $\sigma : w \to s_c \in F^c$ *such that* $\varphi(\sigma : w \to s_c) = \sigma' : w' \to \varphi(s_c)$,

(c) *the preorder over sorts is quasi reflected, i.e. for all sorts* $s_c \in S^{cs}$ *and* $s' \in S'$ *such that* $s' \leq' \varphi(s_c)$, *one of the following conditions holds:*

i. *there exists* $s \in S$ *such that* $\varphi(s) = s'$ *and* $s \leq s_c$, *or*

ii. *there exists* $s_l \in S^{ls}$ *such that* $s' \leq' \varphi(s_l)$ *and* $s_l \leq s_c$.

Given a **COSA** signature morphism $\varphi : \Sigma \to \Sigma'$, we denote by $\varphi^c : \Sigma^c \to \Sigma'^c$ the restriction of φ to constructors.

Remark 12. *Let* $\varphi : \Sigma \to \Sigma'$ *be a* **COSA** *signature morphism which encapsulates all constructors, i.e. for all* $s \in S$ *and* $\sigma' : w' \to \varphi(s) \in F'^c$ *there exists* $\sigma : w \to s \in F^c$ *such that* $\varphi(\sigma : w \to s) = \sigma' : w' \to \varphi(s)$. *Then* s *is constrained iff* $\varphi(s)$ *is constrained, for all sorts* $s \in S$.

The conditions 2a, 2b and 2(c)i from Definition 11 are sufficient to ensure that the reducts of reachable models are again reachable, and they can be found in [6], too. However, the condition 2(c)i or 2(c)ii allows us to (a) capture a larger class of signature morphisms for applications, and (b) prove that pushouts of **COSA** signature morphisms exist (see Section 3).

Definition 13 (**COSA** sentences)**.**

1. $\mathtt{Sen}^{\mathbf{COSA}}(\Sigma) = \mathtt{Sen}^{\mathbf{OSA}}(S, \leq, F)$ *for all signatures* $\Sigma = (S, \leq, F, F^c)$, *and*

2. $\mathtt{Sen}^{\mathbf{COSA}}(\Sigma \xrightarrow{\varphi} \Sigma') = \mathtt{Sen}^{\mathbf{OSA}}((S, \leq, F) \xrightarrow{\varphi} (S', \leq', F'))$ *for all signature morphisms* $\varphi : \Sigma \to \Sigma'$.

In **COSA**, the semantics is restricted to *reachable models*.

Definition 14 (**COSA** models)**.** *Given a constructor-based order-sorted signature* Σ, *the category of constructor-based order-sorted algebras* $\mathtt{Mod}^{\mathbf{COSA}}(\Sigma)$ *is the full subcategory* [3] *of* $\mathtt{Mod}^{\mathbf{OSA}}(S, \leq, F)$ *such that the objects are the order-sorted algebras that are reachable by the constructors in* F^c, *i.e. for all* $A \in |\mathtt{Mod}^{\mathbf{OSA}}(S, \leq, F)|$, *we have* $A \in |\mathtt{Mod}^{\mathbf{COSA}}(\Sigma)|$ *iff there exist a set of variables* X *of loose sorts and an evaluation* $f : X \to A$ *such that its unique extension* $f^\# : T_{\Sigma^c}(X) \to A\restriction_{\Sigma^c}$ *to a* Σ^c*-homomorphism is surjective on the constrained sorts.*

The following result shows that the reducts of reachable models along **COSA** signature morphisms are also reachable.

Proposition 15. *Let* $\varphi : \Sigma \to \Sigma'$ *be a* **COSA** *signature morphism.*

1. *Given a set* X' *of loose variables for* Σ', *there exist a set of loose variables* X *for* Σ *and a substitution* $\theta : X \to T_{\Sigma'^c}(X')\restriction_{\varphi^c}$ *such that the unique extension* $\theta : T_{\Sigma^c}(X) \to T_{\Sigma'^c}(X')\restriction_{\varphi^c}$ *of* θ *to a* Σ^c*-homomorphism is surjective.*

2. *If* $A' \in |\mathtt{Mod}^{\mathbf{COSA}}(\Sigma')|$ *then* $A'\restriction_{\varphi} \in |\mathtt{Mod}^{\mathbf{COSA}}(\Sigma)|$.

Proof. We focus on the first statement, as the second statement is a direct consequence of the first. We define $\theta : X \to T_{\Sigma'^c}(X')\restriction_{\varphi^c}$ as follows:

1. For all constrained sorts $s \in S^{cs}$, $X_s = \emptyset$.

2. For all loose sorts $s \in S^{ls}$, $X_s = \{(x_{t'}, s, \Sigma) \mid t' \in T_{\Sigma'^c}(X')_{\varphi(s)}\}$ and $\theta_s(x_{t'}, s, \Sigma) = t'$ for all $t' \in T_{\Sigma'^c}(X')_{\varphi(s)}$.

We show that for all constructor terms $t' \in T_{\Sigma'^c}(X')$ and sorts $s \in S$ if $\varphi(s)$ is the sort of t' then there exists $t \in T_{\Sigma^c}(X)_s$ such that $\theta(t) = t'$. We proceed by induction on the structure of the terms in $T_{\Sigma'^c}(X')$.

1. If t' is a variable in X' then we have $\theta(x_{t'}, s, \Sigma) = t'$.

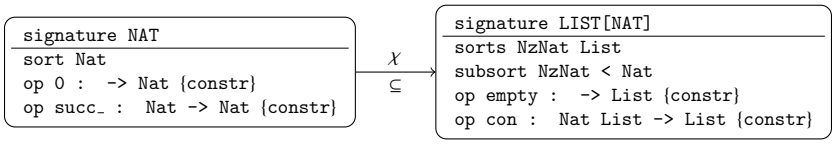
[3] $\mathbb{C}$ is a full subcategory of $\mathbb{C}'$ if $\mathbb{C}$ is a subcategory of $\mathbb{C}'$ such that for all objects $A, B \in |\mathbb{C}|$ we have $\mathbb{C}(A, B) = \mathbb{C}'(A, B)$, where $\mathbb{C}(A, B)$ denotes the class of arrows from A to B.

2. If t' is of the form $\sigma'(\tau'_1, \ldots, \tau'_n) \in T_{\Sigma'^c}(Y')_{\varphi(s)}$ then there are two subcases to consider.
   (a) $s \in S^{ls}$: We have $\theta(x_{t'}, s, \Sigma) = t'$.
   (b) $s \in S^{cs}$: Since the constructors are quasi encapsulated, there exist $s_1 \ldots s_n \in S^*$ and $\sigma \in F^c_{s_1 \ldots s_n \to s}$ such that $\varphi(\sigma) = \sigma'$. By the induction hypothesis, there exists $\tau_i \in T_{\Sigma^c}(X)_{s_i}$ such that $\theta(\tau_i) = \tau'_i$ for all $i \in \{1, \ldots, n\}$. We have $\theta(\sigma(\tau_1, \ldots, \tau_n)) = \varphi(\sigma)(\theta(\tau_1), \ldots, \theta(\tau_n)) = \sigma'(\tau'_1, \ldots, \tau'_n)$.
3. If $s' \leq' \varphi(s_r)$ and $t' \in T_{\Sigma'^c}(X')_{s'}$ then we have two subcases to consider.
   (a) If $s_r \in S^{ls}$ then $\theta_{s_r}(x_{t'}, s_r, \Sigma) = t'$.
   (b) If $s_r \in S^{cs}$ then there are two subcases to consider.
      i. There exists $s \in S$ such that $\varphi(s) = s'$ and $s \leq s_r$. By the induction hypothesis, there exists $t \in T_{\Sigma^c}(X)_s$ such that $\theta_s(t) = t'$. Since $s \leq s_r$ we have $\theta_{s_r}(t) = \theta_s(t) = t'$.
      ii. There exists $s_l \in S^{ls}$ such that $s' \leq' \varphi(s_l)$ and $s_l \leq s_r$. We have $\theta_{s_l}(x_{t'}, s_l, \Sigma) = t'$. Since $s_l \leq s_r$, we have that $\theta_{s_r}(x_{t'}, s_l, \Sigma) = \theta_{s_l}(x_{t'}, s_l, \Sigma) = t'$. □

By Proposition 15, for all **COSA** signature morphisms $\varphi\colon \Sigma \to \Sigma'$, one can define the reduct functor $\_\restriction_\varphi\colon \mathtt{Mod}^{\mathbf{COSA}}(\Sigma') \to \mathtt{Mod}^{\mathbf{COSA}}(\Sigma)$ as the restriction of $\_\restriction_\varphi\colon \mathtt{Mod}^{\mathbf{OSA}}(S', \leq', F') \to \mathtt{Mod}^{\mathbf{OSA}}(S, \leq, F)$ to reachable models. This is crucial for showing that the properties of parameterized specifications are preserved by their instances.

The following example shows that the condition 2(c)i or 2(c)ii from Definition 11 imposed to **COSA** signature morphisms is necessary to prove Proposition 15.

Example 16. *The following diagram defines a signature morphism in* **OSA**.



If we bring the constructors into play then (a) the specification `LIST[NAT]` has two constrained sorts, `Nat` and `List`, one loose sort, `NzNat`, and (b) the **OSA** signature morphism $\chi\colon$ `NAT` $\hookrightarrow$ `LIST[NAT]` is not a **COSA** signature morphism as it introduces the subsort `NzNat` to the 'old' constrained sort `Nat`. We construct a `LIST[NAT]`-model such that its reduct along χ is not reachable. Consider the following non-standard model of natural numbers A:

> $A_{\mathtt{Nat}} = V_1 \cup V_2$, where $V_1 = \{0, 1, 2, 3, \ldots\}$ and $V_2 = \{0', 1', 2', 3', \ldots\}$, $A_{\mathtt{NzNat}} = V_2$, the constant $\mathtt{0} :\to \mathtt{Nat}$ is interpreted as 0, and $\mathtt{succ}_\_ : \mathtt{Nat} \to \mathtt{Nat}$ is interpreted as the successor function: $\mathtt{succ}^A n = (n + 1)$ and $\mathtt{succ}^A n' = (n + 1)'$ for all natural numbers $n \in \mathbb{N}$.

Let L be the model consisting of lists of elements from $A_{\mathtt{Nat}}$. Since the elements of L are reachable by the constructors in `LIST[NAT]`, we have $L \in |\mathtt{Mod}^{\mathbf{COSA}}(\Sigma_{\mathtt{LIST[NAT]}})|$:

> Consider a set of variables $X = \{x_n \mid n \in \mathbb{N}\}$ of sort `NzNat` (which is loose), and the evaluation $f\colon X \to L$ defined by $f(x_n) = n'$ for all natural numbers $n \in \mathbb{N}$. The extension $f^\#\colon T_{\Sigma_{\mathtt{LIST[NAT]}}}(X) \to L$ of $f\colon X \to L$ to a $\Sigma_{\mathtt{LIST[NAT]}}$-homomorphism is surjective.

The reduct of L along χ is A. Notice that $A \notin |\mathtt{Mod}^{\mathbf{COSA}}(\Sigma_{\mathtt{NAT}})|$ as the elements of V_2 are not reachable by the constructors in `NAT`.

3. Pushouts

The pushout construction is the basic mechanism which supports the instantiation of parameterized specifications. It provides the mathematical foundations for building complex and large systems by combining basic specifications. The existence of pushouts in **COSA** is based on some established results in **OSA**. The variant of **OSA** considered here has pushouts of signature morphisms.

Theorem 17 (Pushouts of **OSA** signatures [3])**.** *The category of order-sorted signatures has pushouts.*

Proof sketch. Let $S_1 \xrightarrow{\varphi_1'} S \xleftarrow{\varphi_2'} S_2$ be the pushout of $S_1 \xleftarrow{\varphi_1} S_0 \xrightarrow{\varphi_2} S_2$ in the category of sets. We define the preorder $\leq$ on S as the reflexive and transitive closure of the relation $\varphi_1'(\leq_1) \cup \varphi_2'(\leq_2)$. Notice that $([S_i], [F_i])$ is an ordinary algebraic signature, for all indexes $i \in \{0, 1, 2\}$. Let $([S_1], [F_1]) \xrightarrow{\varphi_1'} ([S], [F]) \xleftarrow{\varphi_2'} ([S_2], [F_2])$ be the pushout of $([S_1], [F_1]) \xleftarrow{\varphi_1} ([S_0], [F_0]) \xrightarrow{\varphi_2} ([S_2], [F_2])$ in the category of algebraic signatures. Without loss of generality, we assume that (a) $([S], [F])$ has no ad-hoc overloaded function symbols, and (b) $[F]$ consists of elements of the form $[\sigma]\colon [w] \to [s]$, where $w \in S^*$ and $s \in S$. This assumption is based on the fact that pushouts are unique up to isomorphism. For each $w \in S^*$ and $s \in S$, we define the set of function symbols $F_{w\to s} = \{\sigma\colon w \to s \mid \varphi_i'([\sigma_i]\colon [w_i] \to [s_i]) = [\sigma]\colon [w] \to [s] \text{ for some } i \in \{1, 2\} \text{ and } \sigma_i\colon w_i \to s_i \in F_i \text{ such that } \varphi_i'(w_i, s_i) = (w, s)\}$. For all $i \in \{1, 2\}$, we let $\varphi_i'\colon (S_i, \leq_i, F_i) \to (S, \leq, F)$ be the signature morphism which maps each $\sigma_i\colon w_i \to s_i$ to $\sigma\colon w \to s$, where $w = \varphi_i(w)$, $s = \varphi_i(s)$ and $[\sigma]\colon [w] \to [s] = \varphi_i([\sigma_i]\colon [w_i] \to [s_i])$. Since $([S], [F])$ has no ad-hoc overloaded function symbols, the signature $(S, \leq, F)$ is sensible. □

The category of constructor-based order-sorted signatures is not co-complete, but if certain conditions are satisfied pushouts do exist.

$$\begin{array}{ccc} (S_1, \leq_1, F_1) & \overset{\varphi'}{\dashrightarrow} & (S, \leq, F) \\ {\scriptstyle\chi}\uparrow & & \uparrow{\scriptstyle\chi'} \\ (S_0, \leq_0, F_0) & \xrightarrow[\varphi]{} & (S_2, \leq_2, F_2) \end{array} \qquad \begin{array}{ccc} (S_1, \leq_1, F_1, F_1^c) & \overset{\varphi'}{\dashrightarrow} & (S, \leq, F, F^c) \\ {\scriptstyle\chi}\uparrow & & \uparrow{\scriptstyle\chi'} \\ (S_0, \leq_0, F_0, F_0^c) & \xrightarrow[\varphi]{} & (S_2, \leq_2, F_2, F_2^c) \end{array}$$

Figure 2: Order-sorted pushouts

Theorem 18 (Pushouts of **COSA** signatures)**.** *Consider a span of* **COSA** *signature morphisms* $(S_1, \leq_1, F_1, F_1^c) \xleftarrow{\chi} (S_0, \leq_0, F_0, F_0^c) \xrightarrow{\varphi} (S_2, \leq_2, F_2, F_2^c)$ *such that* χ *has the following properties: (a) it is injective on sorts, (b) it encapsulates all constructors, and (c) it reflects the preorder over sorts.*

There exists a pushout of **COSA** *signatures shown to the right in Figure 2 such that* χ' *has the following properties: (a) it is injective on sorts, (b) it encapsulates all constructors, and (c) it reflects the preorder over sorts.*

Proof. By Theorem 17, there exists a pushout of **OSA** signature morphisms shown to the left in Figure 2. By the pushout construction, φ' is injective on the set $S_1 \setminus \chi(S_0)$ and $S = \varphi'(S_1) \cup \chi'(S_2) = \varphi'(S_1 \setminus \chi(S_0)) \cup \chi'(S_2)$. Since χ is injective on sorts, χ' is injective on sorts too. We define the binary relation $\preceq$ on S as follows: for all $s_2 \in S_2$, $s_2' \in S_2$, $s_1 \in S_1 \setminus \chi(S_0)$ and $s_1' \in S_1 \setminus \chi(S_0)$,

- $\chi'(s_2) \preceq \chi'(s_2')$ if $s_2 \leq_2 s_2'$,
- $\varphi'(s_1) \preceq \varphi'(s_1')$ if $s_1 \leq_1 s_1'$, and
- $\chi'(s_2) \preceq \varphi'(s_1)$ if $s_2 \leq_2 \varphi(s_0)$ and $\chi(s_0) \leq_1 s_1$ for some $s_0 \in S_0$.

Since both $\leq_1$ and $\leq_2$ are reflexive, $\preceq$ is reflexive too. We prove that $\preceq$ is transitive. Assume that $s \preceq s'$ and $s' \preceq s''$. We show that $s \preceq s''$. We have four cases to consider.

1. $s, s', s'' \in \chi'(S_2)$: There exist $s_2, s_2', s_2'' \in S_2$ such that $\chi'(s_2) = s, \chi'(s_2') = s', \chi'(s_2'') = s''$, $s_2 \leq_2 s_2'$ and $s_2' \leq_2 s_2''$. Since $\leq_2$ is transitive, we have $s_2 \leq_2 s_2''$, which implies $s \preceq s''$.

2. $s, s' \in \chi'(S_2)$ and $s'' \in S \setminus \chi'(S_2)$: There exist $s_2, s_2' \in S_2$ and $s_1'' \in S_1 \setminus \chi(S_0)$ such that $\chi'(s_2) = s$, $\chi'(s_2') = s'$ and $\varphi'(s_1'') = s''$. Since $s \leq s'$, we have $s_2 \leq s_2'$. Since $s' \preceq s''$, there exists $s_0' \in S_0$ such that $s_2' \leq_2 \varphi(s_0')$ and $\chi(s_0') \leq_1 s_1''$. By the transitivity of $\leq_2$, we have $s_2 \leq_2 \varphi(s_0')$ and since $\chi(s_0) \leq_1 s_1''$, we get $\chi'(s_2) \preceq \varphi'(s_1'')$. Hence, $s \preceq s''$.

3. $s \in \chi'(S_2)$ and $s', s'' \in S \setminus \chi'(S_2)$: There exist $s_2 \in S_2$ and $s_1', s_1'' \in S_1 \setminus \chi(S_0)$ such that $\chi'(s_2) = s$, $\varphi'(s_1') = s'$ and $\varphi'(s_1'') = s''$. Since $s' \preceq s''$, we have $s_1' \leq s_1''$. Since $s \preceq s'$, there exists $s_0 \in S_0$ such that $s_2 \leq_2 \varphi(s_0)$ and $\chi(s_0) \leq_1 s_1'$. By the transitivity of $\leq_1$, we have $\chi(s_0) \leq_1 s_1''$ and since $s_2 \leq_2 \varphi(s_0)$, we get $\chi'(s_2) \preceq \varphi'(s_1'')$. Hence, $s \preceq s''$.

4. $s, s', s'' \in S \setminus \varphi(S_2)$: There exist $s_1, s_1', s_1'' \in S_1 \setminus \chi(S_0)$ such that $\varphi'(s_1) = s$, $\varphi'(s_1') = s'$, $\varphi'(s_1'') = s''$, $s_1 \leq_1 s_1'$ and $s_1' \leq_1 s_1''$. Since $\leq_1$ is transitive, we have $s_1 \leq_1 s_1''$, which implies $s \preceq s''$.

By the definition of $\preceq$, we have $\preceq \subseteq \preceq$. For the converse inclusion, we show that $\varphi'(\leq_1) \cup \chi'(\leq_2) \subseteq \preceq$. By the definition of $\preceq$, we have $\chi'(\leq_2) \subseteq \preceq$. Assume that $s_1, s_1' \in S_1$ such that $s_1 \leq_1 s_1'$. We show that $\varphi'(s_1) \preceq \varphi'(s_1')$. There are two cases to consider.

1. $s_1' \in \chi(S_0)$: Let $s_0' \in S_0$ such that $\chi(s_0') = s_1'$. Since χ reflects the preorder, there exists $s_0 \in S_0$ such that $\chi(s_0) = s_1$ and $s_0 \leq_0 s_0'$. It follows that $\varphi(s_0) \leq_2 \varphi(s_0')$, which implies $\varphi'(s_1) = \chi'(\varphi(s_0)) \preceq \chi'(\varphi(s_0')) = \varphi'(s_1')$.

2. $s_1' \in S_1 \setminus \chi(S_0)$: We have two subcases.

   (a) $s_1 \in \chi(S_0)$: Let $s_0 \in S_0$ such that $\chi(s_0) = s_1$. Since $\chi(s_0) \leq_1 s_1$ and $\varphi(s_0) \leq_2 \varphi(s_0)$, we have $\chi'(\varphi(s_0)) \preceq \varphi'(s_1')$. Hence, $\varphi'(s_1) \preceq \varphi'(s_1')$.

   (b) $s_1 \in S_1 \setminus \chi(S_0)$: By the definition of $\preceq$, we have $\varphi'(s_1) \preceq \varphi'(s_1')$.

Since $\preceq$ is the least reflexive and transitive binary relation that contains $\varphi'(\leq_1)$ and $\chi'(\leq_2)$, we have $\preceq \subseteq \preceq$, which implies $\preceq = \preceq$.

Let $F^c = \varphi'(F_1^c) \cup \chi'(F_2^c)$. In order to prove that the diagram shown to the right in Figure 2 is a pushout, it suffices to show that φ' and χ' are **COSA** signature morphisms.

We show that χ' from the diagram shown to the right in Figure 2 encapsulates all constructors. Let $s_2 \in S_2$, $w \in S^*$ and $\sigma \in F^c_{w \to \chi'(s_2)}$. By the definition of F^c, there are two cases to consider.

1. $\sigma\colon w \to \chi'(s_2) = \chi'(\sigma_2\colon w_2 \to s_2')$ for some $\sigma_2\colon w_2 \to s_2' \in F_2^c$: Since χ' is injective on sorts, $s_2' = s_2$.

2. $\sigma\colon w \to \chi'(s_2) = \varphi'(\sigma_1\colon w_1 \to s_1)$ for some $\sigma_1\colon w_1 \to s_1 \in F_1^c$: Since $\varphi'(s_1) = \chi'(s_2)$, by the pushout construction, there exists $s_0 \in S_0$ such that $\chi(s_0) = s_1$ and $\varphi(s_0) = s_2$. Since χ encapsulates all constructors, $\chi(\sigma_0\colon w_0 \to s_0) = \sigma_1\colon w_1 \to s_1$ for some $\sigma_0\colon w_0 \to s_0 \in F_0^c$. Let $\sigma_2 = \varphi(\sigma_0)$ and $w_2 = \varphi(w_0)$, and we have $\chi'(\sigma_2\colon w_2 \to s_2) = \sigma\colon w \to \chi'(s_2)$.

Similarly, one can show that φ' quasi encapsulates constructors.

By the definition of $\preceq$, χ' reflects the preorder. It remains to show that φ' doesn't introduce new subsorts on 'old' constrained sorts. Let $s \in S$ and $sc_1 \in S_1^{cs}$ such that $s \leq \varphi'(sc_1)$. By the definition of $\preceq = \preceq$, there are three cases to consider.

1. $s, \varphi'(sc_1) \in \chi'(S_2)$: By the definition of $\preceq$, there exist $s_2, sc_2 \in S_2$ such that $s_2 \leq_2 sc_2$, $\chi'(s_2) = s$ and $\chi'(sc_2) = \varphi'(sc_1)$. Since χ' encapsulates all constructors, $sc_2 \in S_2^{cs}$. By the pushout construction, $sc_1 \in \chi(S_0)$ and $sc_2 \in \varphi(S_0)$. It follows that there exist $sc_0, sc_0' \in S_0$ such that $\chi(sc_0) = sc_1$ and $\varphi(sc_0') = sc_2$. Since $\chi'(\varphi(sc_0)) = \chi'(\varphi(sc_0'))$ and χ' is injective, $\varphi(sc_0) = \varphi(sc_0') = sc_2$. Since $sc_1 \in S_1^{cs}$ and χ encapsulates all constructors, $sc_0 \in S_0^{sc}$. Since $s_2 \leq_2 \varphi(sc_0)$, by Definition 11, there are two subcases.

   (a) $\varphi(s_0) = s_2$ and $s_0 \leq_0 sc_0$ for some $s_0 \in S_0$: We have $\chi(s_0) \leq_1 \chi(sc_0) = sc_1$. Hence, $s = \varphi'(\chi(s_0))$ and $\chi(s_0) \leq_1 sc_1$.

   (b) $s_2 \leq_2 \varphi(sl_0)$ and $sl_0 \leq_0 sc_0$ for some $sl_0 \in S_0^{ls}$: We have $s = \chi'(s_2) \leq \chi'(\varphi(sl_0)) = \varphi'(\chi(sl_0))$ and $\chi(sl_0) \leq_1 \chi(sc_0) = sc_1$. Since χ encapsulates all constructors, $\chi(sl_0) \in S_1^{ls}$. Hence, $\chi(sl_0) \in S_1^{ls}$ such that $s \leq \varphi'(\chi(sl_0))$ and $\chi(sl_0) \leq_1 sc_1$.

2. $s, \varphi'(sc_1) \in S \setminus \chi'(S_2)$: By the definition of $\preceq$, there exists $s_1 \in S_1$ such that $\varphi'(s_1) = s$ and $s_1 \leq_1 sc_1$.

3. $s \in \chi'(S_2)$ and $\varphi'(sc_1) \in S \setminus \chi'(S_2)$: Let $s_2 \in S_2$ such that $\chi'(s_2) = s$. By the definition of $\preceq$, there exists $sr_0 \in S_0$ such that $s_2 \leq_2 \varphi(sr_0)$ and $\chi(sr_0) \leq_1 sc_1$. There are two subcases to consider.

   (a) $sr_0 \in S_0^{ls}$: Since χ encapsulates all constructors, we have $\chi(sr_0) \in S_1^{ls}$. Since $s_2 \leq_2 \varphi(sr_0)$, we have $\chi'(s_2) = s \leq \chi'(\varphi(sr_0)) = \varphi'(\chi(sr_0))$. Hence, $\chi(sr_0) \in S_1^{ls}$ such that $s \leq \varphi'(\chi(sr_0))$ and $\chi(sr_0) \leq_1 sc_1$.

   (b) $sr_0 \in S_0^{cs}$: Since $s_2 \leq_2 \varphi(sr_0)$, by Definition 11, there are two subcases.

      i. $\varphi(s_0) = s_2$ and $s_0 \leq_0 sr_0$ for some $s_0 \in S_0$: Since $\chi(s_0) \leq_1 \chi(sr_0)$ and $\chi(sr_0) \leq_1 sc_1$, we have $\chi(s_0) \leq_1 sc_1$. Hence, $s = \varphi'(\chi(s_0))$ such that $\chi(s_0) \leq_1 sc_1$.

      ii. $s_2 \leq_2 \varphi(sl_0)$ and $sl_0 \leq_0 sr_0$ for some $sl_0 \in S_0^{ls}$: Since $\chi(sl_0) \leq_1 \chi(sr_0)$ and $\chi(sr_0) \leq_1 sc_1$, we have $\chi(sl_0) \leq_1 sc_1$. Since χ encapsulates all constructors, $\chi(sl_0) \in S_0^{ls}$. We have $s_2 \leq_2 \varphi(sl_0)$, which implies $s = \chi'(s_2) \leq \chi'(\varphi(sl_0)) = \varphi'(\chi(sl_0))$. Hence, $\chi(sl_0) \in S_0^{ls}$ such that $s \leq \varphi'(\chi(sl_0))$ and $\chi(sl_0) \leq_1 sc_1$. □

The work reported in [9] shows that the injectivity on sorts and the encapsulation of all constructors are necessary for Theorem 18. The following example shows that the reflection condition is necessary for Theorem 18.

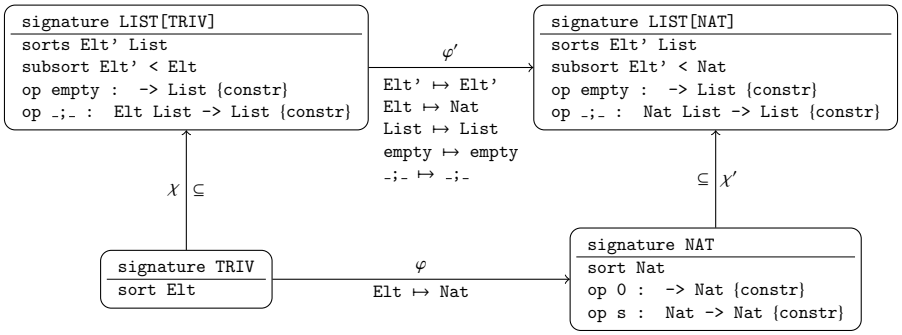


Figure 3: Lists of natural numbers

Example 19 (Lists of natural numbers)**.** *We define lists of natural numbers as the vertex of the pushout of order-sorted signatures depicted in Figure 3. Notice that χ introduces the 'new' subsort* `Elt'` *to the 'old' sort* `Elt`*, which means that χ' doesn't reflect the preorder.*

If we consider the constructor attributes, since χ' introduces the 'new' subsort `Elt'` to the 'old' constrained sort `Nat`, χ' is not a **COSA** signature morphism. It follows that the above diagram is not a pushout in **COSA**.

4. Amalgamation

Amalgamation is an important property in algebraic specification that can be used to give semantics to parameterized specifications. In the present work, it is used to prove the closure of termination under pushouts of signatures. The concept of model amalgamation is a rather basic property of logics, which should not be confused with the existence of a common (elementary) extension of models, a property playing an important role in conventional model theory. In this section, we give sufficient conditions for the amalgamation of pushouts of order-sorted signatures.

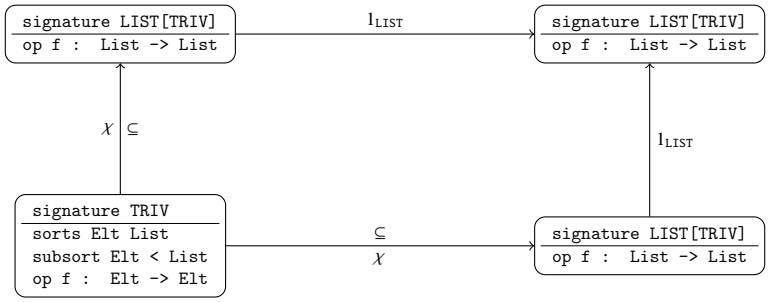
Definition 20 (Amalgamation)**.** *A commuting square of (constructor-based) order-sorted signature morphisms such as the one shown in the diagram below is an* amalgamation square *iff for any Σ_1-algebra A_1 and any Σ_2-algebra A_2 such that $A_1 \restriction_{\varphi_1} = A_2 \restriction_{\varphi_2}$ there exists a unique Σ-algebra A, called* the amalgamation of A_1 and A_2*, such that $A \restriction_{\varphi'_i} = A_i$ for all $i \in \{1, 2\}$.*

$$\begin{array}{ccc} \Sigma_1 & \xrightarrow{\varphi'_1} & \Sigma \\ {\scriptstyle\varphi_1}\uparrow & & \uparrow{\scriptstyle\varphi'_2} \\ \Sigma_0 & \xrightarrow[\varphi_2]{} & \Sigma_2 \end{array}$$

The amalgamation A may be denoted by $A_1 \otimes_{\varphi_1;\varphi'_1} A_2$, or simply, $A_1 \otimes A_2$.

The amalgamation property holds for any pushout of many-sorted algebraic signatures. Unfortunately, not all pushouts of order-sorted signatures have the amalgamation property.

Example 21 (Lists of arbitrary elements)**.** *Let* `LIST[TRIV]` *be the vertex of the pushout of order-sorted signatures shown in the following diagram.*



One can easily check that the commutative square of order-sorted signature morphisms from Example 21 is a pushout by following the description given in the proof of Theorem 17. Now, let A be the `LIST`-algebra such that (a) $A_{\texttt{Elt}} = \{0\}$, $A_{\texttt{List}} = \mathbb{N}$, (b) $\texttt{f}^A\colon \{0\} \to \{0\}$ is the identity, (c) $\texttt{f}^A\colon \mathbb{N} \to \mathbb{N}$ is defined by $\texttt{f}^A(n) = 2 * n$. Let B be the `LIST`-algebra which interprets all symbols in `LIST` as A except for $\texttt{f}\colon \texttt{List} \to \texttt{List}$: $\texttt{f}^B\colon \mathbb{N} \to \mathbb{N}$ is defined by $\texttt{f}^B(n) = 3 * n$. Notice that $A \restriction_\chi = B \restriction_\chi$ but there is no amalgamation $A \otimes_{\chi;1_{\text{LIST}}} B$.

In this section, we give sufficient conditions for the amalgamation of pushouts of order-sorted signatures.

Definition 22 (Preservation)**.** *A signature morphism $\varphi\colon (S, \leq, F) \to (S', \leq', F')$* preserves the subsort polymorphic families of operators *iff for all $\sigma\colon w \to s \in F$ and $\sigma'\colon w' \to s' \in F'$ such that $[\varphi(\sigma)]\colon [\varphi(w)] \to [\varphi(s)] = [\sigma']\colon [w'] \to [s']$ there exists $\sigma : w_1 \to s_1 \in F$ such that $\varphi(\sigma\colon w_1 \to s_1) = \sigma'\colon w' \to s'$ and $[w_1] = [w]$.*

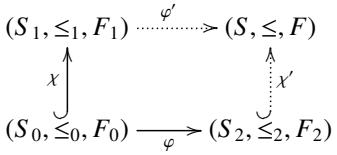


Figure 4: Pushout of order-sorted signatures

Theorem 23 (Amalgamation)**.** *Let $(S_1, \leq_1, F_1) \xleftarrow{\chi} (S_0, \leq_0, F_0) \xrightarrow{\varphi} (S_2, \leq_2, F_2)$ be a span of* **OSA** *signature morphisms such that χ is an inclusion which (a) reflects the preorder over sorts, and (b) preserves the subsort polymorphic families of operators.*

Then there exists an amalgamation pushout depicted in Figure 4 such that χ' is an inclusion which (a) reflects the preorder over sorts, and (b) preserves the subsort polymorphic families of operators.

Proof. Since pushouts are unique up to isomorphism, without any loss of generality we assume that $(S_1, \leq_1, F_1)$ and $(S_2, \leq_2, F_2)$ consist of disjoint symbols.

1. Let $S = S_2 \cup (S_1 \setminus S_0)$, $\chi'\colon S_2 \hookrightarrow S$ be an inclusion, and $\varphi'\colon S_1 \to S$ be the function defined by, $\varphi'(s) = \varphi(s)$ for all $s \in S_0$ and $\varphi'(s) = s$ for all $s \in S_1 \setminus S_0$.

2. Let $\leq = \varphi'(\leq_1) \cup \chi'(\leq_2)$. Since χ reflects the preorder relation, χ' reflects the preorder, too. In other words, χ' doesn't add new subsorts to S_2. Therefore, $\leq$ is reflexive and transitive.

3. Let $F = F_2 \cup \{\sigma_1\colon \varphi'(w_1) \to \varphi'(s_1) \mid \sigma_1\colon w_1 \to s_1 \in F_1 \setminus F_0\}$.

   Let $\chi'\colon (S_2, \leq_2, F_2) \hookrightarrow (S, \leq, F)$ be an inclusion.

   Let $\varphi'\colon (S_1, \leq_1, F_1) \to (S, \leq, F)$ be the signature morphism such that

(a) $\varphi' : S_1 \to S$ is as defined above, and

(b) $\varphi'(\sigma_0 \colon w_0 \to s_0) = \varphi(\sigma_0 \colon w_0 \to s_0)$ for all $\sigma_0 \colon w_0 \to s_0 \in F_0$ and $\varphi'(\sigma_1 \colon w_1 \to s_1) = \sigma_1 \colon \varphi'(w_1) \to \varphi'(s_1)$ for all $\sigma_1 \colon w_1 \to s_1 \in F_1 \setminus F_0$.

Since χ preserves the subsort polymorphic operators, a subsort polymorphic family of operators can be in either F_0 or $(F_1 \setminus F_0)$; it follows that φ' is an extension of φ which maps each subsort polymorphic family of operators to a subset of a subsort polymorphic family of operators; therefore, φ' is an order-sorted signature morphism.

For the second part of the proof, assume $(S_1, \leq_1, F_1)$-algebra A and $(S_2, \leq_2, F_2)$-algebra B. The amalgamation D of A and B is defined in the standard way:

1. $D_{s_2} = B_{s_2}$ for all $s_2 \in S_2$ and $D_{s_1} = B_{s_1}$ for all $s_1 \in S_1 \setminus S_0$.
2. $D_{\sigma_2 : w_2 \to s_2} = B_{\sigma_2 : w_2 \to s_2}$ for all $\sigma_2 \colon w_2 \to s_2 \in F_2$, and $D_{\sigma_1 : \varphi'(w_1) \to \varphi'(s_1)} = A_{\sigma_1 : w_1 \to s_1}$ for all $\sigma_1 \colon w_1 \to s_1 \in F_1 \setminus F_0$. Again, since χ preserves the subsort polymorphic operators, a subsort polymorphic family of operators can be in either F_0 or $(F_1 \setminus F_0)$; it follows that $D_{\sigma : w \to s}$ is well-defined for all function symbols $\sigma \colon w \to s \in F$. □

Theorem 23 can be straightforwardly extended to **COSA** by adding the condition that χ encapsulates all constructors.

The following example shows that the injectivity condition is necessary for Theorem 23.

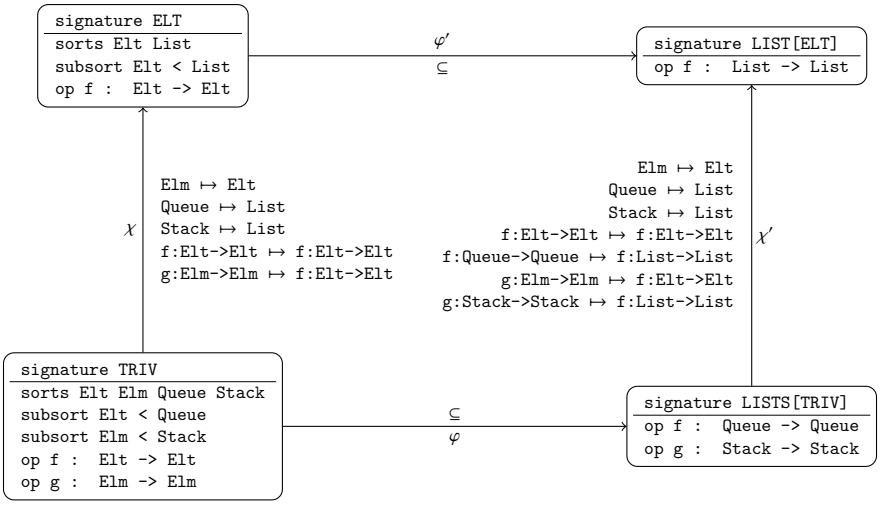


Figure 5: Stacks and queues

Example 24 (Stacks and queues)**.** *Let* ELT $\overset{\varphi'}{\hookrightarrow}$ LIST $\overset{\chi'}{\hookleftarrow}$ LISTS *be the pushout of the span of order-sorted signature morphisms* ELT $\overset{\chi}{\hookleftarrow}$ TRIV $\overset{\varphi}{\hookrightarrow}$ LISTS *depicted in Figure 5.*

It is straightforward to check that the commutative square of signature morphisms in Figure 5 is a pushout of order-sorted signatures. Now, let A be the LISTS-algebra such that (a) $A_{\texttt{Elt}} = A_{\texttt{Elm}} = \{0\}$, $A_{\texttt{Queue}} = A_{\texttt{Stack}} = \mathbb{N}$, (b) $\texttt{f}^A \colon \{0\} \to \{0\}$ and $\texttt{g}^A \colon \{0\} \to \{0\}$ are identities, (c) $\texttt{f}^A \colon \mathbb{N} \to \mathbb{N}$ is defined by $\texttt{f}^A(n) = 2 * n$ and $\texttt{g}^A \colon \mathbb{N} \to \mathbb{N}$ is defined by $\texttt{g}^A(n) = 3 * n$. Let B be the ELT-algebra which interprets all symbols in ELT as A. Notice that $A \restriction_\varphi = B \restriction_\chi$, but there is no amalgamation $A \otimes_{\chi;\varphi'} B$, as $\texttt{f} \colon \texttt{List} \to \texttt{List}$ cannot be interpreted as both $2 * n$ and $3 * n$.

5. Reducibility

In this section we show that sufficient-completeness and reducibility are equivalent under the assumption that termination holds. Then we prove that reducibility is closed under pushouts of signatures.

A basic specification SP is a pair $\langle \Sigma_{SP}, \Gamma_{SP} \rangle$ such that $\Sigma_{SP} = (S_{SP}, \leq_{SP}, F_{SP}, F^c_{SP})$ is a constructor-based order-sorted signature and Γ_{SP} is a set of sentences over Σ_{SP}. The term rewriting system of SP is given by the universally quantified conditional equations in Γ_{SP} regarded as term rewriting rules. In algebraic specification languages executable by rewriting such as CafeOBJ [26] or Maude [27], the term rewriting system of a specification is given by all sentences without the non-executable attribute `nonexec`. The axioms with the `nonexec` attribute have no role in the operational semantics of specifications, but they describe the class of models for the underlying specification.

The associativity of a binary operator $\sigma\colon s\ s \to s$ is described by the quantified equation $\forall x, y, z \cdot \sigma(x, \sigma(y, z)) = \sigma(\sigma(x, y), z)$, and the equivalence class of the terms $\sigma(x, \sigma(y, z))$ and $\sigma(\sigma(x, y), z)$ under the associativity law is represented by $\sigma(x, y, z)$. The quantified equation $\forall x, y \cdot \sigma(x, y) = \sigma(y, x)$ expresses the commutativity of σ. In algebraic specification languages executable by term rewriting, such quantified equations are often used in the matching phase of term rewriting and they are defined using attributes such as `assoc` and `comm`. For example, $\sigma\colon s\ s \to s$ [`comm`] denotes the fact that $\sigma\colon s\ s \to s$ is a commutative operator. Another example of attribute is `ring`, defined for (a) three sorts, let's say `Elt`, `Seq` and `Config`, such that `Elt` $\leq$ `Seq`, (b) a constant symbol $e :\to$ `Seq`, (c) a binary associative operator $\_;\_\colon$ `Seq Seq` $\to$ `Seq` [`assoc id:` e] whose identity is e, and (d) a unary operator $[\_]\colon$ `Seq` $\to$ `Config` [`ring`]. The attribute `ring` means that the rotative law $\forall x_0, x_1, \ldots, x_n \cdot [x_0; x_1; \ldots; x_n] = [x_1; \ldots; x_n; x_0]$ and the reversible law $\forall x_0, x_1, \ldots, x_n \cdot [x_0; x_1; \ldots; x_n] = [x_n; \ldots; x_1; x_0]$ hold, where x_i is a variable of sort `Elt` for all $i \in \{0, \ldots, n\}$. See [17] for details about `ring` and its applications to mobile robot algorithms.

In this paper, we consider only basic specifications SP of the form $\langle \Sigma_{SP}, \Gamma_{SP} \rangle$ such that Γ_{SP} is the union of two disjoint sets: (a) B_{SP} which consists of universally quantified equations such as associativity, commutativity, identity and ring axioms, and (b) E_{SP} which consists of universally quantified conditional equations that give the operational semantics of SP by term rewriting. For any such SP and any set of variables X for Σ_{SP}, we let $\equiv_{B_{SP},X}$ denote the least order-sorted congruence on $T_{\Sigma_{SP}}(X)$ generated by the equations in B_{SP}. We may drop the subscripts B_{SP} and/or X from $\equiv_{B_{SP},X}$ whenever they are clear from the context.

A specification morphism $\varphi\colon$ SP $\to$ SP$'$ consists of a signature morphism $\varphi\colon \Sigma_{SP} \to \Sigma_{SP'}$ such that $\Gamma_{SP'} \models \varphi(\Gamma_{SP})$. A view $\varphi\colon$ SP $\to$ SP$'$ is a specification morphism such that $\varphi(E_{SP}) \subseteq E_{SP'}$ and $\varphi(B_{SP}) \subseteq B_{SP'}$. A parameterized specification $\chi\colon$ P $\to$ SP[P] is a view such that $\chi\colon \Sigma_P \hookrightarrow \Sigma_{SP[P]}$ is an inclusion. In this case, P is the parameter and SP[P] is the body of the parameterized specification.

Given a term t, the positions $p, q, \ldots$ in t are strings of positive natural numbers used for identifying subterms $t|_p, t|_q, \ldots$ of t. The set of positions in t is denoted $\mathtt{Pos}(t)$. As usual, the term $t[t']_p$ denotes the term obtained from t by substituting t' for $t|_p$ at position p. We give the definition of conditional term rewriting modulo a set of equations.

Definition 25 (Conditional rewriting modulo a set of equations [14, 16])**.** *Let* SP *be a specification, and Y a set of variables for Σ_{SP}. The conditional rewriting modulo B_{SP}, $\to_{SP,Y} := \{\to_{SP,Y,s}\}_{s \in S}$, in* SP *is defined as follows:*

1. *for all sorts $s \in S$, $\overset{0}{\to}_{SP,Y,s} = \emptyset$.*
2. *for all $s \in S$ and all $t, u \in T_{\Sigma_{SP}}(Y)_s$, we have $t \overset{m+1}{\to}_{SP,Y,s} u$ iff $t \equiv_{B_{SP},Y} t'$, $t'|_p = \theta(l)$, $u' = t'[\theta(r)]_p$, $u' \equiv_{B_{SP},Y} u$, and $\theta(l_i)(\overset{m}{\to}_{SP,Y,s_i})^*\ \tau_i\ {}^*(\overset{m}{\leftarrow}_{SP,Y,s_i})\theta(r_i)$,*

   for some terms $t', u', \tau_i \in T_{\Sigma_{SP}}(Y)$, sentence $\forall X \cdot l =_{s'} r$ if $\bigwedge_{i=1}^n l_i =_{s_i} r_i \in E_{SP}$, position $p \in \mathtt{Pos}(t')$, and substitution $\theta\colon X \to T_{\Sigma_{SP}}(Y)$.

The relation $\to_{SP,Y}$ is the union $\bigcup_{m \in \mathbb{N}} \overset{m}{\to}_{SP,Y}$.

In applications, the term rewriting rules are sort decreasing, which means if $t'[\theta(l)]_p$ from Definition 25 is well-formed then $t'[\theta(r)]_p$ is well-formed, too.

Fact 26. *Let Y be a set of variables for the signature Σ_{SP} of a specification* SP*.*

1. *For any terms $t, u \in T_{\Sigma_{SP}}(Y)_{s_1} \cap T_{\Sigma_{SP}}(Y)_{s_2}$, where $s_1, s_2 \in S_{SP}$, we have $t \to_{SP,Y,s_1} u$ iff $t \to_{SP,Y,s_2} u$.*
2. *For any terms $t, u \in T_{\Sigma_{SP}}(Y)_s$, where $s \in S$, we have $t \to_{SP,Y,s} u$ iff $t \to_{SP,Y',s} u$, where $Y' = var(t)$.*

We may drop the subscripts Y and s from $\to_{SP,Y,s}$ when they are clear from the context. Given a specification SP, a term t is called *reducible* if there exists a term u such that $t \to_{SP} u$. A term is called a *normal form* if it is not reducible.

SP is terminating if there is no infinite rewrite sequence $t_0 \to_{\mathsf{SP}} t_1 \to_{\mathsf{SP}} \cdots$. If SP has the *termination* property then each term t has a normal form, i.e. $t \to^*_{\mathsf{SP}} u$ and u is a normal form. [4]

Lemma 27 ([14])**.** *The rewriting step is closed under substitutions, i.e. if* $t \to_{\mathsf{SP},Y} u$ *then* $\psi(t) \to_{\mathsf{SP},Z} \psi(u)$ *for all substitutions* $\psi\colon Y \to T_{\Sigma_{\mathsf{SP}}}(Z)$.

As expected, conditional rewriting modulo a set of equations is preserved by the signature morphisms. This result is presented for the convenience of the reader as it is difficult to indicate a source where a modular proof can be found. We proceed by showing a preliminary result.

Lemma 28. *Let* $\varphi\colon \Sigma \to \Sigma'$ *be a* **COSA** *signature morphism and* $\theta\colon X \to T_\Sigma(Y)$ *a substitution. The following commutativity property holds:*

$$\varphi_Y(\theta(t)) = \theta^\varphi(\varphi_X(t)) \text{ for all } t \in T_\Sigma(X),$$

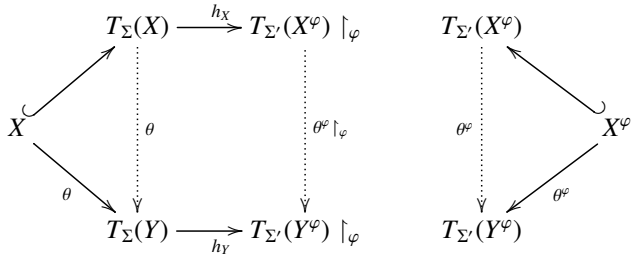
where $\theta^\varphi\colon X^\varphi \to T_{\Sigma'}(Y^\varphi)$ *is defined by* $\theta^\varphi(x, \varphi(s), \Sigma') = \varphi_Y(\theta(x, s, \Sigma))$ *for all variables* $(x, s, \Sigma) \in X$.

Proof. We define the following $(S, \leq, F)$-homomorphisms:

(a) $h_X\colon T_\Sigma(X) \to T_{\Sigma'}(X^\varphi)\restriction_\varphi$ by $h_X(x, s, \Sigma) = (x, \varphi(s), \Sigma')$ for all $(x, s, \Sigma) \in X$, and

(b) $h_Y\colon T_\Sigma(Y) \to T_{\Sigma'}(Y^\varphi)\restriction_\varphi$ by $h_Y(y, s, \Sigma) = (y, \varphi(s), \Sigma')$ for all $(y, s, \Sigma) \in Y$.

Recall that $\varphi_X\colon \Sigma[X] \to \Sigma'[X^\varphi]$ is the extension of φ mapping each $(x, s, \Sigma) \in X$ to $(x, \varphi(s), \Sigma') \in X^\varphi$, where $\Sigma[X] = (S, \leq, F[X], F^c)$. The signature morphism φ_Y is defined, similarly. Note that (a) $h_X(t) = \varphi_X(t)$ for all $t \in T_\Sigma(X)$, and (b) $h_Y(t) = \varphi_Y(t)$ for all $t \in T_\Sigma(Y)$.



We have that $\theta^\varphi\restriction_\varphi(h_X(x, s, \Sigma)) = \theta^\varphi\restriction_\varphi(x, \varphi(s), \Sigma') = \theta^\varphi(x, \varphi(s), \Sigma') = \varphi_Y(\theta(x, s, \Sigma)) = h_Y(\theta(x, s, \Sigma))$ for all $(x, s, \Sigma) \in X$. It follows that $\theta; h_Y = h_X; (\theta^\varphi\restriction_\varphi)$. Hence, $\varphi_Y(\theta(t)) = h_Y(\theta(t)) = \theta^\varphi\restriction_\varphi(h_X(t)) = \theta^\varphi(\varphi_X(t))$ for all $t \in T_\Sigma(X)$. □

The preservation of conditional rewriting modulo equations by the signature morphisms is a corollary of Lemma 28.

Corollary 29. *If* $t \to_{\mathsf{SP},Y} u$ *then* $\varphi_Y(t) \to_{\mathsf{SP}',Y^\varphi} \varphi_Y(u)$ *for all views* $\varphi\colon \mathsf{SP} \to \mathsf{SP}'$.

Proof. We proceed by induction in the definition of $\to_{\mathsf{P},Y}$.

1. $\varphi_Y(\overset{0}{\to}_{\mathsf{SP},Y}) = \emptyset = \overset{0}{\to}_{\mathsf{SP}',Y^\varphi}$.

2. Let $t \overset{n+1}{\to}_{\mathsf{SP},Y} u$ be a rewriting step as in Definition 25, and assume that $\overset{n}{\to}_{\mathsf{SP},Y}$ is closed under φ_Y. We have $t \equiv_{B_{\mathsf{SP}}} t'$, $t'|_p = \theta(l)$, $u' = t'[\theta(r)]_p$, $u' \equiv_{B_{\mathsf{SP}}} u$ and $\theta(l_i)(\overset{n}{\to}_{\mathsf{SP},Y})^*\ \tau_i\ {}^*(\overset{n}{\leftarrow}_{\mathsf{SP},Y})\theta(r_i)$ for some terms $t', u', \tau_i \in T_{\Sigma_{\mathsf{SP}}}(Y)$, sentence $\forall X \cdot l = r$ if $\bigwedge_{i=1}^n l_i = r_i \in E_{\mathsf{SP}}$, position $p \in \mathsf{Pos}(t')$, and substitution $\theta\colon X \to T_{\Sigma_{\mathsf{SP}}}(Y)$.

   - Since $\varphi(B_{\mathsf{SP}}) \subseteq B_{\mathsf{SP}'}$, $\varphi_Y(t) \equiv_{B_{\mathsf{SP}'}} \varphi_Y(t')$ and $\varphi_Y(u') \equiv_{B_{\mathsf{SP}'}} \varphi_Y(u)$.
   - Since $\varphi(E_{\mathsf{SP}}) \subseteq E_{\mathsf{SP}'}$, $\forall X^\varphi . \varphi_X(l) = \varphi_X(r)$ if $\bigwedge_{i=1}^n \varphi_X(l_i) = \varphi_X(r_i) \in E_{\mathsf{SP}'}$.

[4] Normal forms are not unique if SP is not confluent.

Recall that $\theta^{\varphi}\colon X^{\varphi} \to T_{\Sigma_{\mathrm{SP}'}}(Y^{\varphi})$ is defined by $\theta^{\varphi}(x, \varphi(s), \Sigma_{\mathrm{SP}'}) = \varphi_Y(\theta(x, s, \Sigma_{\mathrm{SP}}))$ for all variables $(x, \varphi(s), \Sigma_{\mathrm{SP}'}) \in X^{\varphi}$. Notice that

- $\varphi_Y(t')|_p = \varphi_Y(t'|_p) = \varphi_Y(\theta(l)) = \theta^{\varphi}(\varphi_X(l))$, and
- $\varphi_Y(u') = \varphi_Y(t'[\theta(r)]_p) = \varphi_Y(t')[\varphi_Y(\theta(r))]_p = \varphi_Y(t')[\theta^{\varphi}(\varphi_X(r))]_p$.

By the induction hypothesis, for all $i \in \{1, \ldots, n\}$, $\varphi_Y(\theta(l_i))(\overset{n}{\to}_{\mathrm{SP}',Y^{\varphi}})^*\varphi_Y(\tau_i)$ and $\varphi_Y(\theta(r_i))(\overset{n}{\to}_{\mathrm{SP}',Y^{\varphi}})^*\varphi_Y(\tau_i)$; by Lemma 28, $\theta^{\varphi}(\varphi_X(l_i))(\overset{n}{\to}_{\mathrm{SP}',Y^{\varphi}})^*\varphi_Y(\tau_i)$ and $\theta^{\varphi}(\varphi_X(r_i))(\overset{n}{\to}_{\mathrm{SP}',Y^{\varphi}})^*\varphi_Y(\tau_i)$. Hence, $\varphi_Y(t) \overset{n+1}{\to}_{\mathrm{SP}',Y^{\varphi}} \varphi_Y(u)$.

Assuming that $t \to_{\mathrm{SP},Y} u$ we have $t \overset{n}{\to}_{\mathrm{SP},Y} u$ for some natural number $n \in \mathbb{N}$. It follows that $\varphi_Y(t) \overset{n}{\to}_{\mathrm{SP}',Y^{\varphi}} \varphi_Y(u)$. Hence, $\varphi_Y(t) \to_{\mathrm{SP}',Y^{\varphi}} \varphi_Y(u)$. □

We recall the model-theoretic notion of sufficient-completeness which can be found for example in [6].

Definition 30 (Sufficient-completeness)**.** *A specification* SP *is sufficient-complete if for all order-sorted algebras* $A \in |\mathtt{Mod}^{\mathbf{COSA}}(S_{\mathrm{SP}}, \leq_{\mathrm{SP}}, F_{\mathrm{SP}}, F^{cs}_{\mathrm{SP}})|$ *such that* $A \models \Gamma_{\mathrm{SP}}$*, we have* $A \in |\mathtt{Mod}^{\mathbf{COSA}}(\Sigma_{\mathrm{SP}})|$.

Definition 30 says that a specification SP is sufficient-complete if all models reachable by the constrained operators and satisfying Γ_{SP} are reachable by the constructors. Definition 30 is connected to the denotational semantics of the specifications. In term rewriting literature, there exists an analogous definition of sufficient-completeness which is related to the operational semantics of the specifications.

Definition 31 (Rewriting-based sufficient-completeness)**.** *A specification* SP *is rewriting-based sufficient-complete if for each constrained term t there exists a constructor term u such that* $t \to^*_{\mathrm{SP}} u$.

The above definition is more general than the classical one (see, for example, [20, 21, 22, 16]) since both terms t and u may contain variables of loose sorts or/and operators on loose sorts; this means that the search of a ground constructor term u free of operators on loose sorts such that $t \to^*_{\mathrm{SP}} u$ is replaced by the search of a constructor term u such that $t \to^*_{\mathrm{SP}} u$. As expected, if the term rewriting system of a specification is sufficient-complete then the specification is sufficient-complete.

Proposition 32. *Sufficient-completeness is a consequence of rewriting-based sufficient-completeness.*

Proof. Let SP be a specification and let $A \in |\mathtt{Mod}^{\mathbf{COSA}}(S_{\mathrm{SP}}, \leq_{\mathrm{SP}}, F_{\mathrm{SP}}, F^{cs}_{\mathrm{SP}})|$ such that $A \models \Gamma_{\mathrm{SP}}$. We show that $A \in |\mathtt{Mod}^{\mathbf{COSA}}(\Sigma_{\mathrm{SP}})|$, i.e. A is reachable by the constructors in F^{c}_{SP}. Since A is reachable by the operators in F^{cs}_{SP}, there exist a set of variables X of loose sorts and an evaluation $f\colon X \to A$ such that $f^{\#}\colon T_{\Sigma_{\mathrm{SP}}}(X) \to A$ is surjective on the constrained sorts, where $f^{\#}\colon T_{\Sigma_{\mathrm{SP}}}(X) \to A$ is the unique extension of f to a $(S_{\mathrm{SP}}, \leq_{\mathrm{SP}}, F_{\mathrm{SP}})$-homomorphism. We overload the notation and let $f^{\#}$ denote also the unique extension of f to a Σ^{c}_{SP}-homomorphism $f^{\#}\colon T_{\Sigma^{c}_{\mathrm{SP}}}(X) \to A\restriction_{\Sigma^{c}_{\mathrm{SP}}}$. We prove that $f^{\#}\colon T_{\Sigma^{c}_{\mathrm{SP}}}(X) \to A\restriction_{\Sigma^{c}_{\mathrm{SP}}}$ is surjective on the constrained sorts.

Let $s \in S^{cs}$ and $a \in A_s$. There exists a constrained term $t \in T_{\Sigma_{\mathrm{SP}}}(X)_s$ such that $f^{\#}(t) = a$. Since SP is rewriting-based sufficient-complete, we have $t \to^*_{\mathrm{TR}_{\mathrm{SP}}} u$ for some constructor term $u \in T_{\Sigma^{c}_{\mathrm{SP}}}(X')$, where $X' = var(t)$. It follows that $\Gamma_{\mathrm{SP}} \models \forall X' \cdot t = u$. Since $A \models \Gamma_{\mathrm{SP}}$, we get $A \models \forall X' \cdot t = u$. This means $f^{\#}(u) = f^{\#}(t) = a$. □

The following example shows that the converse implication does not hold, in general.

Example 33. *We define the specification* ELT *as shown in the following diagram.*

```
spec ELT
sort Elt
op c :  -> Elt {constr}
op a :  -> Elt
eq c = a
```

We have $a \not\to^*_{\mathrm{ELT}} c$. This means that ELT is not rewriting-based sufficient-complete. However, ELT is sufficient-complete, since for any algebra A over $(S_{\mathrm{ELT}}, \leq_{\mathrm{ELT}}, F_{\mathrm{ELT}}, F^{cs}_{\mathrm{ELT}})$ which satisfies {c = a} we have $A_{\mathtt{a}} = A_{\mathtt{c}}$.

Definition 34. *A specification is reducible if all basic terms are reducible.*

If there are no conditional equations in SP, the reducibility property is decidable since it is equivalent to the existence of an instance of the left-hand side of an equation, called a redex. For conditional term rewriting, we need to check the existence of a redex as well as the condition of the equation which gives the redex. The reducibility property for conditional term rewriting is known to be undecidable. Fortunately, one can provide sufficient conditions for the reducibility of conditional term rewriting, which can be algorithmically checked. See, for example, [14, 15].

The specification in the following example is reducible, but it is not rewriting-based sufficient-complete.

Example 35. *We define the specification* TRIV *as shown in the following diagram.*

```
spec TRIV
sort Elt
op a :  -> Elt {constr}
op f :  Elt -> Elt
eq f(a) = f(a)
```

Obviously, the term f(a) is reducible but there is no constructor term u such that $\mathtt{f(a)} \to^*_{\mathtt{TRIV}} \mathtt{u}$. Example 35 shows that reducibility is weaker than rewriting-based sufficient-completeness but if termination holds these two properties are equivalent.

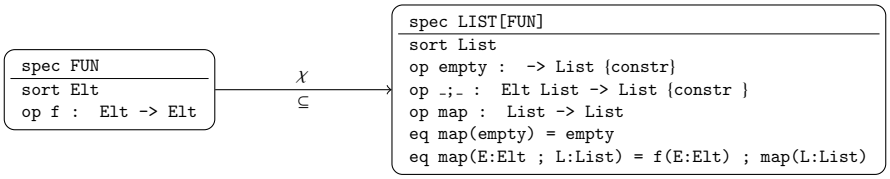
Proposition 36. *Given a specification such that the termination property holds, rewriting-based sufficient-completeness and reducibility are equivalent.*

Proof. The direct implication holds without the termination assumption. The proof is straightforward.

For the converse implication, we assume a specification SP that is terminating and reducible. Let t be a constrained term. Since SP is terminating, let u be a normal form of t. Suppose towards a contradiction that u contains an operator in $F^{cs} \setminus F^c$. Let p be a position such that $u|_p = \sigma(u_0, \ldots, u_n)$, $\sigma \in F^{cs} \setminus F^c$ and u_i are constructor terms. Since $u|_p$ is basic, $u|_p$ is reducible. It follows that u is not a normal form, and therefore, the assumption that u contains an operator in $F^{cs} \setminus F^c$ is false. Hence, SP is rewriting-based sufficient-complete. □

Many higher-order functions used in higher-order programming can be described by some parameterized first-order functions. See, for example, [28] for details. The following example presents a typical higher-order function, called map, which applies a unary function to all elements of a given list.

Example 37 (Parameterized lists)**.** *Let* χ: FUN → LIST[FUN] *be a parameterized specification as shown in the following diagram.*



We show that the classical definition of rewriting-based sufficient-completeness is not applicable to the specification LIST[FUN] from Example 37. Notice that LIST is terminating. See Section 6 for details. In addition, all basic terms are reducible. For example, if t= map(En:Elt ;...; E1:Elt ; empty) then $t \to^*_{\mathtt{LIST}} u$, where u = f(En:Elt);...; f(E1:Elt) ; empty is a constructor term, as it is free of the operator map. Since all basic terms of LIST are reducible, by Proposition 36, LIST is rewriting-based sufficient-complete. By Proposition 32, LIST is sufficient-complete. By the definition of sufficient-completeness given in [15], the term rewriting system of LIST is not sufficient-complete, since, for example, the normal form of t = map(e; empty) is u = f(e); empty, which contains the operator f that is not a constructor. Note that the definition of sufficient-completeness given in [15] is more general than the classical definition of sufficient-completeness given, for example, in [21] or [16], since it allows variables of loose sorts to occur in constructor terms.

According to our experience, the reducibility property is important for equational reasoning with term rewriting, such as the deduction supported by the OBJ languages. In a sense, given a constrained term t, a constructor term u such that $t \to^* u$ is regarded as an answer of t. Rewriting-based sufficient-completeness ensures not only the existence of a constructor term equivalent to a given constrained term (as its model-theoretic counterpart), but also, the computability of it provided that termination holds.

The following result is a corollary of Proposition 15, and it is useful for proving the main theorem of this section, the closure of reducibility under pushouts.

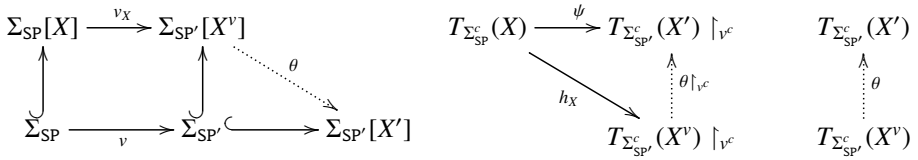


Figure 6: Substitutions

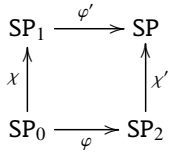
Corollary 38. *Let $v\colon \mathsf{SP} \to \mathsf{SP}'$ be a view and X' a set of variables of loose sorts for the signature $\Sigma_{\mathsf{SP}'}$. There exist a set of variables X of loose sorts for Σ_{SP} and a substitution $\theta\colon X^v \to T_{\Sigma^c_{\mathsf{SP}'}}(X')$, where X^v is the translation of X along v, such that for each sort $s \in S_{\mathsf{SP}}$ and each constructor term $t' \in T_{\Sigma^c_{\mathsf{SP}'}}(X')_{v(s)}$ we have $\theta(v_X(t)) = t'$ for some constructor term $t \in T_{\Sigma^c_{\mathsf{SP}}}(X)_s$.*

Proof. By Proposition 15, there exist a set of loose variables X and a substitution $\psi\colon X \to T_{\Sigma^c_{\mathsf{SP}'}}(X') \restriction_{v^c}$ such that its extension $\psi\colon T_{\Sigma^c_{\mathsf{SP}}}(X) \to T_{\Sigma^c_{\mathsf{SP}'}}(X') \restriction_{v^c}$ to a Σ^c_{SP}-homomorphism is surjective. We define the substitution $\theta\colon X^v \to T_{\Sigma^c_{\mathsf{SP}'}}(X')$ by $\theta(x, v(s), \Sigma_{\mathsf{SP}'}) = \psi(x, s, \Sigma_{\mathsf{SP}})$ for all variables $(x, v(s), \Sigma_{\mathsf{SP}'}) \in X^v$. Let $\theta\colon T_{\Sigma^c_{\mathsf{SP}'}}(X^v) \to T_{\Sigma^c_{\mathsf{SP}'}}(X')$ be the extension of $\theta\colon X^v \to T_{\Sigma^c_{\mathsf{SP}'}}(X')$ to a $\Sigma^c_{\mathsf{SP}'}$-homomorphism. We define the Σ^c_{SP}-homomorphism $h_X\colon T_{\Sigma^c_{\mathsf{SP}}}(X) \to T_{\Sigma^c_{\mathsf{SP}'}}(X^v) \restriction_{v^c}$ by $h_X(x, s, \Sigma_{\mathsf{SP}}) = (x, v(s), \Sigma_{\mathsf{SP}'})$ for all $(x, s, \Sigma_{\mathsf{SP}}) \in X$. Notice that $(\theta \restriction_{v^c})(h_X(x, s, \Sigma_{\mathsf{SP}})) = (\theta \restriction_{v^c})(x, v(s), \Sigma_{\mathsf{SP}'}) = \theta(x, v(s), \Sigma_{\mathsf{SP}'}) = \psi(x, s, \Sigma_{\mathsf{SP}})$ for all variables $(x, s, \Sigma_{\mathsf{SP}}) \in X$. It follows that the diagram shown in the center of Figure 6 is commutative.

Notice that $h_X(t) = v_X(t)$ for all $t \in T_{\Sigma^c_{\mathsf{SP}}}(X)$. Let $t' \in T_{\Sigma^c_{\mathsf{SP}'}}(X')_{v(s)}$ be an arbitrary constructor term. Since ψ is surjective, $\psi(t) = t'$ for some $t \in T_{\Sigma^c_{\mathsf{SP}}}(X)_s$. We have $\psi(t) = (\theta \restriction_{v^c})(h_X(t)) = (\theta \restriction_{v^c})(v_X(t)) = \theta(v_X(t))$. Since t' was arbitrarily chosen, for all $t' \in T_{\Sigma^c_{\mathsf{SP}'}}(X')_{v(s)}$ there exists $t \in T_{\Sigma^c_{\mathsf{SP}}}(X)_s$ such that $\theta(v_X(t)) = t'$. □

All ingredients for proving the closure of reducibility under pushouts are in place.

Theorem 39. *Assume that the commutative diagram of views shown below has the following properties: (a) SP_1 and SP_2 are reducible, (b) $B_{\mathsf{SP}} = \varphi'(B_{\mathsf{SP}_1}) \cup \chi'(B_{\mathsf{SP}_2})$ and $E_{\mathsf{SP}} = \varphi'(E_{\mathsf{SP}_1}) \cup \chi'(E_{\mathsf{SP}_2})$, and (c) $F^{cs}_{\mathsf{SP}} = \varphi'(F^{cs}_{\mathsf{SP}_1}) \cup \chi'(F^{cs}_{\mathsf{SP}_2})$.*



Then SP is reducible.

Proof. Let $t = \sigma(t_1, \ldots, t_n)$ be a basic Σ_{SP}-term, where $\sigma \in F^{cs}_{\mathsf{SP}} \setminus F^{c}_{\mathsf{SP}}$. Without loss of generality, we assume that there exists $\sigma_1 \in F^{cs}_{\mathsf{SP}_1}$ such that $\varphi'(\sigma_1) = \sigma$. The case when there exists $\sigma_2 \in F^{cs}_{\mathsf{SP}_2}$ such that $\chi'(\sigma_2) = \sigma$ is similar.

Notice that $\sigma_1 \notin F^{c}_{\mathsf{SP}_1}$ (if $\sigma_1 \in F^{c}_{\mathsf{SP}_1}$ then $\sigma \in F^{c}_{\mathsf{SP}}$ which is a contradiction). We define $X = var(t)$. By Corollary 38, there exist a set of loose variables X_1 for Σ_{SP_1}, a substitution $\theta\colon X_1^{\varphi'} \to T_{\Sigma^c_{\mathsf{SP}}}(X)$ and a constructor term $\tau_i \in T_{\Sigma^c_{\mathsf{SP}_1}}(X_1)$ for each $i \in \{1, \ldots, n\}$ such that $\theta(\varphi'_{X_1}(\tau_i)) = t_i$ for all $i \in \{1, \ldots, n\}$. Note that $\theta(\varphi'_{X_1}(\sigma_1(\tau_1, \ldots, \tau_n))) = \sigma(\theta(\varphi'_{X_1}(\tau_1)), \ldots, \theta(\varphi'_{X_1}(\tau_n))) = \sigma(t_1, \ldots, t_n)$. Since $\sigma_1(\tau_1, \ldots, \tau_n)$ is a basic term and SP_1 is reducible, $\sigma_1(\tau_1, \ldots, \tau_n)$ is reducible. Since $\varphi'(B_{\mathsf{SP}_1}) \subseteq B_{\mathsf{SP}}$ and $\varphi'(E_{\mathsf{SP}_1}) \subseteq E_{\mathsf{SP}}$, by Corollary 29, $\varphi'_{X_1}(\sigma_1(\tau_1, \ldots, \tau_n))$ is reducible. By Lemma 27, $\theta(\varphi'_{X_1}(\sigma_1(\tau_1, \ldots, \tau_n))) = \sigma(t_1, \ldots, t_n)$ is reducible, too. □

The following result is a corollary of Theorem 39, and it shows that reducibility is closed under parameter instantiation.

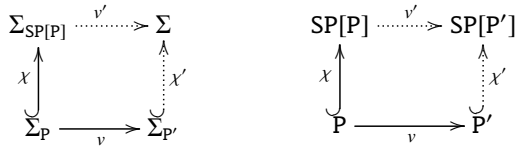


Figure 7: Pushout of views

Corollary 40 (Closure of reducibility under pushouts)**.** *Let* $v\colon \mathsf{P} \to \mathsf{P}'$ *be a view and* $\chi\colon \mathsf{P} \hookrightarrow \mathsf{SP[P]}$ *a parameterized specification such that (a)* $\chi\colon \Sigma_{\mathsf{P}} \hookrightarrow \Sigma_{\mathsf{SP[P]}}$ *reflects the preorder over sorts and encapsulates all operators, and (b)* SP[P] *and* P′ *are reducible. Then there exists a pushout of views as shown to the right in Figure 7 such that its vertex* SP[P′] *is reducible.*

Proof. By Theorem 18, there exists a pushout of signatures as depicted to the left in Figure 7. We define $B = v'(B_{\mathsf{SP[P]}}) \cup \chi'(B_{\mathsf{P}'})$, $E = v'(E_{\mathsf{SP[P]}}) \cup \chi'(E_{\mathsf{P}'})$ and $\Gamma = B \cup E$. Let $\mathsf{SP[P']} = \langle \Sigma, \Gamma \rangle$. By [18, Theorem 11], the diagram shown to the right in Figure 7 is a pushout of views.

Since χ encapsulates all operators, χ' encapsulates all operators, too. Now we show that $F^{cs}_{\mathsf{SP[P']}} = v'(F^{cs}_{\mathsf{SP[P]}}) \cup \chi'(F^{cs}_{\mathsf{P}'})$. Let $\sigma\colon w \to s \in F^{cs}_{\mathsf{SP[P']}}$. There are two cases to consider.

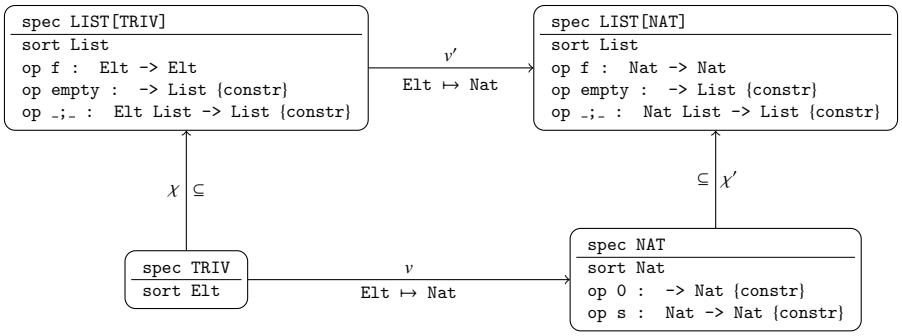
1. $s \in \chi'(S_{\mathsf{P}'})$: There exists $s_2 \in S_{\mathsf{P}'}$ such that $\chi'(s_2) = s$. Since χ' encapsulates all operators, there exists $\sigma_2\colon w_2 \to s_2 \in F_{\mathsf{P}'}$ such that $\chi'(\sigma_2\colon w_2 \to s_2) = \sigma\colon w \to s$. Since χ encapsulates all constructors, χ' encapsulates all constructors, too. It follows that s_2 is a constrained sort. Hence, $\sigma_2\colon w_2 \to s_2 \in F^{cs}_{\mathsf{P}'}$.

2. $s \in S_{\mathsf{SP[P']}} \setminus \chi'(S_{\mathsf{P}'})$: There exists $\sigma_1\colon w_1 \to s_1 \in F_{\mathsf{SP[P]}}$ such that $v'(\sigma_1\colon w_1 \to s_1) = \sigma\colon w \to s$. Since $s \in S_{\mathsf{SP[P']}} \setminus \chi'(S_{\mathsf{P}'})$ is constrained, s_1 is constrained too. Indeed, if $(F^{c}_{\mathsf{SP[P]}})_{w_1 \to s_1} = \emptyset$ then since $v'(s_1) \notin \chi'(S_{\mathsf{P}'})$, $v'((F^{c}_{\mathsf{SP[P]}})_{w_1 \to s_1}) = (F^{c}_{\mathsf{SP[P']}})_{w \to s} = \emptyset$, and we get $(F^{cs}_{\mathsf{SP[P']}})_{w \to s} = \emptyset$, which is a contradiction with the existence of $\sigma\colon w \to s \in F^{cs}_{\mathsf{SP[P']}}$. It follows that $\sigma_1\colon w_1 \to s_1 \in F^{cs}_{\mathsf{SP[P]}}$.

Since both SP and P′ are reducible, by Theorem 39, SP[P′] is reducible too. □

The following example shows that the encapsulation condition from Corollary 40 is necessary.

Example 41 (Lists of natural numbers)**.** *We define lists of natural numbers by instantiating the parameterized specification* $\chi\colon$ `TRIV` $\hookrightarrow$ `LIST[TRIV]` *by the view* v : `TRIV` $\to$ `NAT` *as shown in the following diagram.*



In Example 41, both `LIST[TRIV]` and `NAT` are reducible but `LIST[NAT]` is not reducible. For example `f(0)` is a basic term and there are no rewrite rules to reduce it further.

6. Termination

In this section, we give sufficient conditions to prove that termination is closed under pushouts. It is worth noting that constructors do not play any role in proving termination.

Definition 42 (Relational order-sorted algebra)**.** *Let $\Sigma = (S, \leq, F)$ be an order-sorted signature. $(A, \prec)$ is a relational order-sorted Σ-algebra if A is an order-sorted algebra equipped with an S-sorted relation $\prec = \{\prec_s\}_{s \in S}$ such that*

1. *A_s is not empty for all sorts $s \in S$, and*
2. *$a \prec_{s_1} b$ iff $a \prec_{s_2} b$ for all sorts $s_1, s_2 \in S$ such that $[s_1] = [s_2]$ and all elements $a, b \in A_{s_1} \cap A_{s_2}$.*

A relational order-sorted algebra induces a relation on terms.

Definition 43. *Let $(A, \prec)$ be a relational order-sorted algebra over the signature $(S, \leq, F)$. For any set of variables X for $(S, \leq, F)$, $(A, \prec)$ induces a relation $\prec_A$ on $T_{(S,\leq,F)}(X)$ as follows: for all terms $t, u \in T_{(S,\leq,F)}(X)$, $t \prec_{A,X} u$ iff $\vartheta(t) \prec \vartheta(u)$ for all valuations $\vartheta \colon X \to A$.*

When the set of variables X is understood from the context, we denote $\prec_{A,X}$ simply by $\prec_A$. The following lemma is a preliminary result for generalizing the technique of defining reduction orders by interpreting terms into well-founded domains [14] to order-sorted algebras.

Lemma 44. *Let Σ be an order-sorted signature, X a set of variables for Σ, and $(A, \prec)$ a relational order-sorted Σ-algebra. Then:*

1. *$\prec_A$ is closed under substitutions.*
2. *$\prec_A$ is monotone if $\prec$ is monotone.*
3. *$\prec_A$ is well-founded if $\prec$ is well-founded.*

Proof. We proceed as follows.

1. Assume that $t_1 \prec_A t_2$ and let $\theta \colon X \to T_\Sigma(Y)$ be a substitution. We show that $\theta(t_1) \prec_A \theta(t_2)$: let $\vartheta \colon Y \to A$ a valuation; notice that $\theta; \vartheta \colon X \to A$ is a valuation of X in A; since $t_1 \prec_A t_2$, we have $\vartheta(\theta(t_1)) \prec \vartheta(\theta(t_2))$.
2. Let $(\sigma \colon w \to s) \in F$ and $t, t' \in T_\Sigma(X)_w$ such that $t \prec_A t'$. [5] We show that $\sigma(t) \prec_A \sigma(t')$: let $\vartheta \colon X \to A$ be a valuation; we have $\vartheta(\sigma(t)) = \sigma^A(\vartheta(t))$ and $\vartheta(\sigma(t')) = \sigma^A(\vartheta(t'))$; since $t \prec_A t'$, we have $\vartheta(t) \prec \vartheta(t')$; by the monotonicity of $\prec$, we have $\vartheta(\sigma(t)) = \sigma^A(\vartheta(t)) \prec \sigma^A(\vartheta(t')) = \vartheta(\sigma(t'))$.
3. Suppose towards a contradiction that $\prec$ is well-founded but $\prec_A$ is not. There is an infinite sequence $t_1 \succ_A t_2 \succ_A \ldots \succ_A t_n \succ_A \ldots$. Let $X = \bigcup_{n \in \mathbb{N}} var(t_n)$. Let $\vartheta \colon X \to A$ a valuation, which exists as A has non-empty carrier sets. Then we have an infinite sequence $\vartheta(t_1) \succ \vartheta(t_2) \succ \ldots \succ \vartheta(t_n) \succ \ldots$, which contradicts our assumption that $\prec$ is well-founded. □

The proof of Lemma 44 is a straightforward generalization of the classical results, which is given for the convenience of the reader. Lemma 44 provides the basis for the following definition.

Definition 45 (Well-founded monotone order-sorted algebra)**.** *Let $(A, \prec)$ be a relational order-sorted algebra over the signature $(S, \leq, F)$.*

1. *$(A, \prec)$ is* well-founded *if there is no infinite sequence $a_1 \succ a_2 \succ a_3 \succ \ldots$ for all sorts $s \in S$ and elements $a_i \in A_s$, where $i \in \mathbb{N}$.*
2. *$(A, \prec)$ is* monotone *if for all $\sigma \colon w \to s \in F$ and elements $a, b \in A_w$ such that $a \prec b$ we have $\sigma^A(a) \prec \sigma^A(b)$.* [6]

[5] $(t_1, \ldots, t_n) \prec_A (t'_1, \ldots, t'_n)$ if $t_i \preceq_A t'_i$ for all $i \in \{1, \ldots, n\}$ and $t_i \prec_A t'_i$ for some $i \in \{1, \ldots, n\}$.
[6] $(a_1, \ldots, a_n) \prec (b_1, \ldots, b_n)$ if $a_i \preceq b_i$ for all $i \in \{1, \ldots, n\}$ and $a_i \prec b_i$ for some $i \in \{1, \ldots, n\}$.

In algebraic specification literature, monotone algebras $(A, \prec)$ are known as preorder algebras [26]. Notice that if $A_{s_1} = \emptyset$ for some sort $s_1 \in S$ then for any set of variables $X = \{X_s\}_{s \in S}$ such that $X_{s_1} \neq \emptyset$, there is no valuation $\vartheta \colon X \to A$; it follows that $\prec_{A,X} = T_{(S,\leq,F)}(X) \times T_{(S,\leq,F)}(X)$, which means that $\prec_{A,X}$ is not well-founded. Therefore, the nonempty assumption for the well-founded domains from Definition 42 is needed.

Definition 46 (Monotone order-sorted algebra for a specification)**.** *$(A, \prec)$ is a monotone order-sorted algebra for a specification* SP *whenever*

1. *$(A, \prec)$ is a well-founded monotone order-sorted $(S_{\mathsf{SP}}, \leq_{\mathsf{SP}}, F_{\mathsf{SP}})$-algebra (and not necessarily a constructor-based order-sorted Σ_{SP}-algebra),*
2. *for each set of variables X for Σ_{SP} and any terms $t, t' \in T_{\Sigma_{\mathsf{SP}}}(X)$ such that $t \equiv_{B_{\mathsf{SP}}} t'$, we have $\vartheta(t) = \vartheta(t')$ for all valuations $\vartheta \colon X \to A$, and*
3. *$E_{\mathsf{SP}} \sqsubseteq \succ_A$, i.e. for all sentences $\forall X \cdot l =_s r$ if $\bigwedge_{i=1}^n l_i =_{s_i} r_i \in E_{\mathsf{SP}}$ and all substitutions $\theta \colon X \to T_{\Sigma_{\mathsf{SP}}}(Y)$ the following condition holds: $\theta(l) \succ_{A,Y} \theta(r)$ if for all $i \in \{1, \ldots, n\}$ we have $\theta(l_i) \succ^*_{A,Y} \tau_i \prec^*_{A,Y} \theta(r_i)$ for some term $\tau_i \in T_{\Sigma_{\mathsf{SP}}}(Y)$.*

The following result shows that there are no infinite sequences of the form $t_1 \to_{\mathsf{SP}} t_2 \to_{\mathsf{SP}} t_3 \to_{\mathsf{SP}} \ldots$. This termination result along with the reducibility ensures sufficient-completeness.

Theorem 47. *An order-sorted specification* SP *terminates iff there exists a monotone order-sorted algebra $(A, \prec)$ for* SP.

Proof.

"$\Leftarrow$" We show that $\overset{n}{\to}_{\mathsf{SP}} \subseteq \succ_A$ for all $n \in \mathbb{N}$.

- $n = 0$: We have $\overset{0}{\to}_{\mathsf{SP}} = \emptyset$.
- $n \Rightarrow n + 1$: Assume that $t \overset{n+1}{\to}_{\mathsf{SP},Y} u$. We have $t \equiv_{B_{\mathsf{SP}}} t'$, $t'|_p = \theta(l)$, $u' = t'[\theta(r)]_p$, $u' \equiv_{B_{\mathsf{SP}}} u$, and $\theta(l_i) (\overset{n}{\to}_{\mathsf{SP},Y})^* \tau_i \, {}^*(\overset{n}{\leftarrow}_{\mathsf{SP},Y}) \theta(r_i)$ for some terms $t', u', \tau_i \in T_{\Sigma_{\mathsf{SP}}}(Y)$, sentence $\forall X \cdot l = r$ if $\bigwedge_{i=1}^n l_i = r_i \in E_{\mathsf{SP}}$, position $p \in \mathsf{Pos}(t')$, and substitution $\theta \colon X \to T_{\Sigma_{\mathsf{SP}}}(Y)$. By the induction hypothesis, $\theta(l_i) \succ^*_A \tau_i \prec^*_A \theta(r_i)$. By Definition 46(3), $\theta(l) \succ_A \theta(r)$. By Lemma 44, $t'[\theta(l)]_p \succ_A t'[\theta(r)]_p$. By Definition 46(2), $t \succ_A u$.

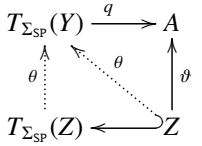
Since $\overset{n}{\to}_{\mathsf{SP}} \subseteq \succ_A$ for all $n \in \mathbb{N}$, we obtain $\to_{\mathsf{SP}} \subseteq \succ_A$. Therefore, there cannot be an infinite sequence $t_1 \to_{\mathsf{SP}} t_2 \to_{\mathsf{SP}} t_3 \to_{\mathsf{SP}} \ldots$, otherwise we get an infinite sequence $t_1 \succ_A t_2 \succ_A t_3 \succ_A \ldots$ which contradicts the fact that $\succ_A$ is well-founded.

"$\Rightarrow$" Let Y be a set of variables for Σ_{SP} such that Y_s is infinite and countable for all sorts $s \in S$. We define $A = T_{\Sigma_{\mathsf{SP}}}(Y)/_{\equiv_{B_{\mathsf{SP}}}}$ and $t_1/_{\equiv_{B_{\mathsf{SP}}}} \succ t_2/_{\equiv_{B_{\mathsf{SP}}}}$ iff $t_1 \to_{\mathsf{SP}} t_2$ for all terms $t_1, t_2 \in T_{\Sigma_{\mathsf{SP}}}(Y)$. Then:

1. The relation $\succ$ is well-defined, since $t_1 \to_{\mathsf{SP}} t_2$ if $t_1 \equiv_{B_{\mathsf{SP}}} t_1'$, $t_2 \equiv_{B_{\mathsf{SP}}} t_2'$ and $t_1' \to_{\mathsf{SP}} t_2'$.
2. For all $s \in S_{\mathsf{SP}}$, since Y_s is not empty, $T_{\Sigma_{\mathsf{SP}}}(Y)_s$ is not empty.
3. $\succ$ is monotone and well-founded as $\to_{\mathsf{SP}}$ is monotone and well-founded.

Let $q \colon T_{\Sigma_{\mathsf{SP}}}(Y) \to A$ be the quotient homomorphism defined by $q(t) = t/_{\equiv_{B_{\mathsf{SP}}}}$ for all terms $t \in T_{\Sigma_{\mathsf{SP}}}(Y)$. We show that the second and the third condition from Definition 46 hold for $(A, \prec)$:

(2) If $t \equiv_{B_{\mathsf{SP}},Z} t'$, where Z is a set of variables for Σ_{SP} and $t, t' \in T_{\Sigma_{\mathsf{SP}}}(Z)$, then for all $\vartheta \colon Z \to A$, since q is surjective, there exists $\theta \colon Z \to T_{\Sigma_{\mathsf{SP}}}(X)$ such that $\theta; q = \vartheta$.



Since $t \equiv_{B_{\mathsf{SP}},Z} t'$, we have $\theta(t) \equiv_{B_{\mathsf{SP}},Y} \theta(t')$, which implies $q(\theta(t)) = q(\theta(t'))$. It follows that $\vartheta(t) = q(\theta(t)) = q(\theta(t')) = \vartheta(t')$.

(3) By Definition 25, $E_{\mathsf{SP}} \sqsubseteq \rightarrow_{\mathsf{SP}}$. Therefore, it suffices to show that $\rightarrow_{\mathsf{SP}} = \succ_A$. We focus on proving $t \rightarrow_{\mathsf{SP},Z} u$ iff $t \succ_{A,Z} u$ for all sets of variables Z and all terms $t, u \in T_{\Sigma_{\mathsf{SP}}}(Z)$.

"$\Rightarrow$" Let $\vartheta \colon Z \to A$. There exists $\theta \colon Z \to T_{\Sigma_{\mathsf{SP}}}(Y)$ such that $\theta; q = \vartheta$. Notice that $q(\theta(t)) = \vartheta(t)$ and $q(\theta(u)) = \vartheta(u)$. Since $t \rightarrow_{\mathsf{SP}} u$, we have $\theta(t) \rightarrow_{\mathsf{SP}} \theta(u)$. Hence, $\vartheta(t) = q(\theta(t)) \succ q(\theta(u)) = \vartheta(u)$.

"$\Leftarrow$" Since Y_s is infinite for all $s \in S$, there exists $\theta \colon Z \to Y$ injective. Let $\vartheta = \theta; q$. Since $t \succ_A u$, $q(\theta(t)) = \vartheta(t) \succ \vartheta(u) = q(\theta(u))$. It follows that $\theta(t) \rightarrow_{\mathsf{SP}} \theta(u)$. By the injectivity of $\theta \colon Z \to Y$, we get $t \rightarrow_{\mathsf{SP}} u$. □

Unlike pushouts and amalgamation, the proof of Theorem 47 is a straightforward generalization of the classical results, which can be found, for example, in [14]; we include it in the paper for readers' convenience. Notice that operational termination [29] is stronger than termination, since it additionally requires that each rewriting step uses a finite number of conditions. In order to prove operational termination, one needs to show, besides $l \succ_A r$, also $l \succ_A l_i$ and $l \succ_A r_i$, for all sentences $\forall X \cdot l = r$ if $\bigwedge_{i=1}^n l_i = r_i \in E_{\mathsf{SP}}$.

We defined the necessary infrastructure for proving the closure of termination under pushouts.

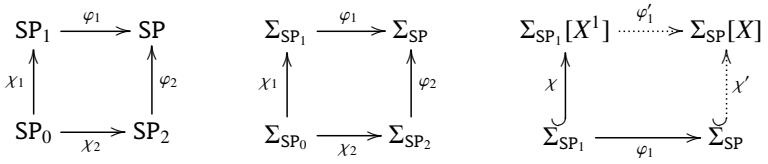


Figure 8: Commutative diagrams

Theorem 48. *Assume a commutative square of views such as shown to the left in Figure 8 satisfying the following properties:*

1. *the commutative square of signature morphisms shown in the center of Figure 8 has the amalgamation property, and*
2. $B_{\mathsf{SP}} = \varphi_1(B_{\mathsf{SP}_1}) \cup \varphi_2(B_{\mathsf{SP}_2})$ *and* $E_{\mathsf{SP}} = \varphi_1(E_{\mathsf{SP}_1}) \cup \varphi_2(E_{\mathsf{SP}_2})$.

Let $(A^1, \prec^1)$ *and* $(A^2, \prec^2)$ *be two monotone order-sorted algebras for* SP_1 *and* SP_2*, respectively, such that*

3. $(A^1, \prec^1) \restriction_{\chi_1} = (A^2, \prec^2) \restriction_{\chi_2}$*, and*
4. $l \succ_{A^i} r$ *for all sentences* $\forall X \cdot l = r$ *if* $\bigwedge_{j=1}^n l_j = r_j \in E_{\mathsf{SP}_i}$*, where* $i \in \{1, 2\}$.

Then there exists a monotone order-sorted algebra $(A, \prec)$ *for* SP *such that* $l \succ_A r$ *for all* $\forall X \cdot l = r$ *if* $\bigwedge_{i=1}^n l_i = r_i \in E_{\mathsf{SP}}$*. Moreover,* SP *is terminating.*

Proof. We define the monotone order-sorted algebra $(A, \prec)$ for SP as the amalgamation of $(A^1, \prec^1)$ and $(A^2, \prec^2)$. We show that $(A, \prec)$ is a monotone order-sorted algebra for SP:

- It is straightforward to show that $(A, \prec)$ is well-founded and monotone as $(A^1, \prec^1)$ and $(A^2, \prec^2)$ are well-founded and monotone.
- Let $e \in E_{\mathsf{SP}}$, where $e := \forall X \cdot l = r$ if $\bigwedge_{i=1}^n l_i = r_i$. We show that $l \succ_A r$. There are two cases:
  1. $e \in \varphi_1(E_{\mathsf{SP}_1})$: There exists $e^1 := \forall X^1 \cdot l^1 = r^1$ if $\bigwedge_{i=1}^n l_i^1 = r_i^1 \in E_{\mathsf{SP}_1}$ such that $\varphi_1(e^1) = e$. Consider the designated pushout shown to the right in Figure 8 for translating Σ_{SP_1}-terms with variables from X^1 along φ_1. We have: $\varphi_1'(l^1) = l$, $\varphi_1'(r^1) = r$, $\varphi_1'(l_i^1) = l_i$ and $\varphi_1'(r_i^1) = r_i$ for all $i \in \{1, \ldots, n\}$. Let $\vartheta \colon X \to A$ be a valuation of X in A. Let A^ϑ be the expansion of A along χ' which interprets X according to ϑ. We have that $\vartheta(l) = A_l^\vartheta = A^\vartheta_{\varphi_1'(l^1)} = (A^\vartheta \restriction_{\varphi_1'})_{l^1}$. Since $l^1 \succ_{A^1} r^1$ and $A^\vartheta \restriction_{\varphi_1'}$ is a χ-expansion of A^1, $(A^\vartheta \restriction_{\varphi_1'})_{l^1} \succ^1 (A^\vartheta \restriction_{\varphi_1'})_{r^1}$. It follows that $A_l^\vartheta \succ A_r^\vartheta$, which means $\vartheta(l) \succ \vartheta(r)$. Since ϑ is an arbitrary valuation of X in A, we conclude that $l \succ_A r$.

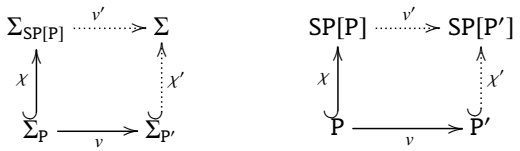


Figure 9: Pushout of views

2. $e \in \varphi_2(E_{\mathrm{SP}_2})$: This case is analogous to the one above.

□

The following is a corollary of Theorem 48.

Corollary 49 (Closure of termination under pushouts)**.** *Let* $\nu\colon \mathtt{P} \to \mathtt{P}'$ *be a view and* $\chi\colon \mathtt{P} \hookrightarrow \mathrm{SP[P]}$ *a parameterized specification such that:*

1. $\chi\colon \Sigma_{\mathtt{P}} \hookrightarrow \Sigma_{\mathrm{SP[P]}}$ *encapsulates all constructors, reflects the preorder over sorts and preserves the subsort polymorphic families of operators, and*
2. *there exist two monotone order-sorted algebras* $(A^1, <^1)$ *and* $(A^2, <^2)$ *for* SP[P] *and* P′*, respectively, such that (a)* $(A^1, <^1) \restriction_\chi = (A^2, <^2) \restriction_\nu$ *and (b)* $l >_{A^1} r$ *for all sentences* $\forall X \cdot l = r$ *if* $\bigwedge_{i=1}^n l_i = r_i \in E_{\mathrm{SP[P]}}$*, and similarly,* $l >_{A^2} r$ *for all sentences* $\forall X \cdot l = r$ *if* $\bigwedge_{i=1}^n l_i = r_i \in E_{\mathtt{P}'}$.

Then there exists a monotone order-sorted algebra $(A, <)$ *for* SP[P′]*. In particular,* SP[P′] *is terminating.*

Proof. By Theorem 18, there exists a pushout of signatures as shown to the left in Figure 9. By Theorem 23, it has the amalgamation property. Let SP[P′] be the specification $\langle \Sigma, \Gamma \rangle$, where $\Gamma = B \cup E$, $B = \nu'(B_{\mathrm{SP[P]}}) \cup \chi'(B_{\mathtt{P}'})$ and $E = \nu'(E_{\mathrm{SP[P]}}) \cup \chi'(E_{\mathtt{P}'})$. It is straightforward to check that the commutative square of views shown to the right in Figure 9 is a pushout. By Theorem 48, there exists a monotone order-sorted algebra for SP[P′], which means that SP[P′] is terminating. □

In the following we investigate termination and reducibility of some examples of order-sorted parameterized specifications inspired from concrete case studies of formal specification and verification.

Example 50 (Lists of natural numbers)**.** *Let* $\chi\colon$ `TRIV` $\hookrightarrow$ `LIST[TRIV]` *be a parameterized specification and* $\nu\colon$ `TRIV` $\to$ `NAT` *a view as depicted in Figure 10. The function* `head` *gives the first element of a list, while the function* `tail` *removes the first element from a list. Notice that both* `head` *and* `tail` *are defined for non-empty lists. The function* `ge` *compares the length of two lists, while the function* `max` *compares two lists and returns the list with the maximum length.*

We define a monotone order-sorted algebra $(A^1, <^1)$ for `LIST[TRIV]`:

- $(A^1_s, <^1_s) = (\mathbb{N}^+, <)$ for all $s \in \{\mathtt{Elt}, \mathtt{Bool}, \mathtt{NeList}\}$ and $(A^1_{\mathtt{List}}, <^1_{\mathtt{List}}) = (\mathbb{N}, <)$;
- $c^{A^1} = 1$, where c is any constant in $\{\mathtt{true}, \mathtt{false}, \mathtt{empty}\}$;
- $\mathtt{head}^{A^1} : \mathbb{N}^+ \to \mathbb{N}^+$ is defined by $\mathtt{head}^{A^1}(n) = 1 + n$ for all $n \in \mathbb{N}^+$, and $\mathtt{tail}^{A^1} = \mathtt{head}^{A^1}$;
- $\mathtt{ge}^{A^1} : \mathbb{N} \to \mathbb{N}$ is defined by $\mathtt{ge}^{A^1}(m, n) = 1 + m + n$ for all $m, n \in \mathbb{N}$, and $\mathtt{max}^{A^1} : \mathbb{N} \to \mathbb{N}$ is defined by $\mathtt{max}^{A^1}(m, n) = 2 + m + n$ for all $m, n \in \mathbb{N}$.

We define a monotone order-sorted algebra $(A^2, <^2)$ for `NAT`:

- $(A^2_{\mathtt{Nat}}, <^2_{\mathtt{Nat}}) = (\mathbb{N}^+, <)$;
- $0^{A^2} = 1$ and $\mathtt{succ}^{A^2} : \mathbb{N}^+ \to \mathbb{N}^+$ is defined by $\mathtt{succ}^{A^2}(n) = 1 + n$;
- $+^{A^2} : \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{N}^+$ is defined by $m \ +^{A^2} \ n = m + 2n$ for all $m, n \in \mathbb{N}^+$;

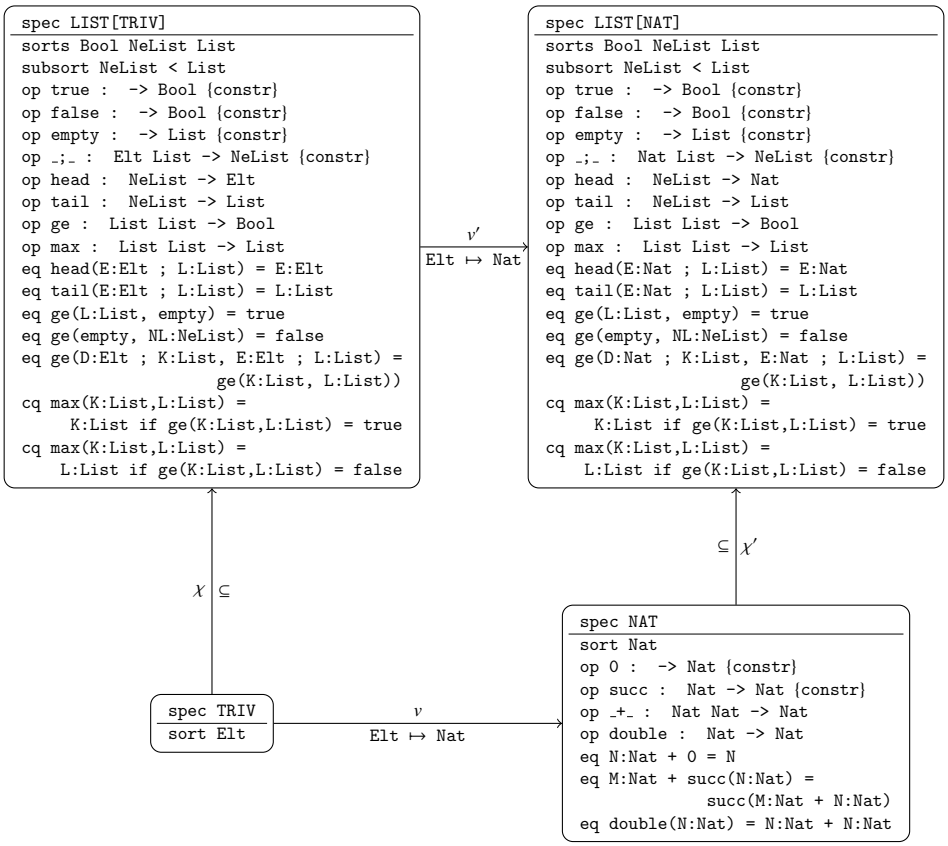


Figure 10: Lists of natural numbers

- $\texttt{double}^{A^2} : \mathbb{N}^+ \to \mathbb{N}^+$ is defined by $\texttt{double}^{A^2}(n) = 4n$.

One can straightforwardly prove that $(A^1, \prec^1)$ and $(A^2, \prec^2)$ are monotone and well-founded, and that the term rewriting rules of `LIST[TRIV]` and `NAT` are decreasing. By Theorem 47, `LIST[TRIV]` and `NAT` are terminating. Since $(A^1, \prec^1) \restriction_\chi = (A^2, \prec^2) \restriction_\varphi$, by Theorem 48, the instance `LIST[NAT]` is terminating.

The parameterized specification χ encapsulates all operators. Basic terms $\texttt{head}(nl)$ and $\texttt{tail}(nl)$ are reducible for all constructor terms nl of the sort `NeList`. Notice that we do not need to consider $\texttt{head}(\texttt{empty})$ as a basic term since `empty` is not of sort `NeList`. For all constructor terms l_1 and l_2 of sort `List`, the basic term $\texttt{ge}(l_1, l_2)$ is reducible; the same remark holds for $\texttt{max}(l_1, l_2)$, since one of the conditions of the last two rewriting rules is joinable as a consequence of the reducibility and termination of $\texttt{ge}(l_1, l_2)$. Thus, `LIST[TRIV]` is reducible. Using similar arguments, one can show that `NAT` is reducible, too. By Corollary 40, the instance `LIST[NAT]` is reducible. Since `LIST[NAT]` is terminating, by Proposition 36, `LIST[NAT]` is sufficient-complete.

Example 51 (List mappings)**.** *Let* $\chi\colon \texttt{FUN} \hookrightarrow \texttt{LIST[FUN]}$ *be a parameterized specification and* $v\colon \texttt{FUN} \to \texttt{NAT}$ *a view as depicted in Figure 11.*

Firstly, we define a class C^1 of monotone order-sorted algebras $(A^1, \prec^1)$ for `LIST[FUN]` such that the interpretation of all sorts in `LIST[FUN]` and all function symbols in $F_{\texttt{LIST[FUN]}} \setminus F_{\texttt{FUN}}$ is fixed, while the interpretation of the function symbol in `FUN` may vary within limits that ensure termination:

- $(A^1_s, \prec^1_s) = (\mathbb{N}^+, <)$ for all $s \in \{\texttt{Elt}, \texttt{NeList}\}$ and $(A^1_{\texttt{List}}, \prec^1_{\texttt{List}}) = (\mathbb{N}, <)$;
- $\texttt{empty}^{A^1} = 0$ and $;^{A^1} : \mathbb{N}^+ \times \mathbb{N} \to \mathbb{N}^+$ is defined by $m \;;^{A^1} n = 1 + m + n$ for all $m \in \mathbb{N}^+$ and $n \in \mathbb{N}$;

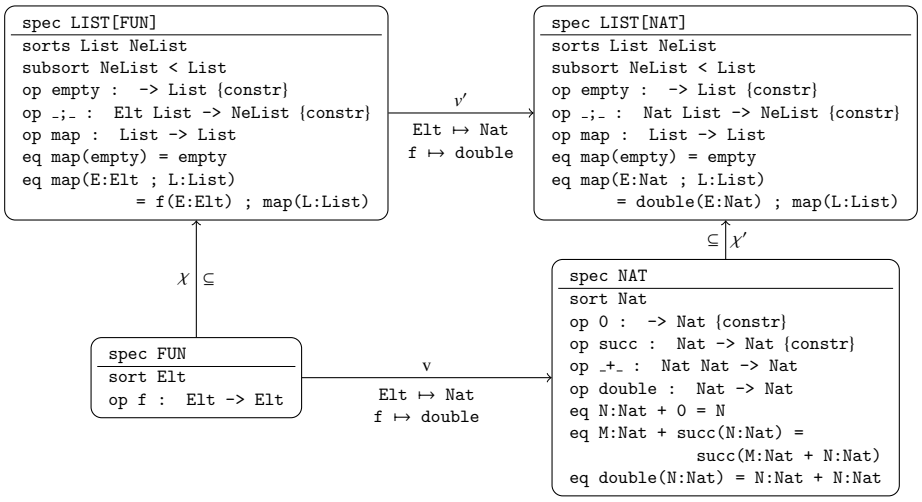


Figure 11: List mappings

- $\mathtt{f}^{A^1} : \mathbb{N}^+ \to \mathbb{N}^+$ is any monotone function satisfying the following properties: (a) $\mathtt{f}^{A^1}(m+n) \geq \mathtt{f}^{A^1}(m) + \mathtt{f}^{A^1}(n)$ for all $m \in \mathbb{N}^+$ and $n \in \mathbb{N}^+$, and (b) $\mathtt{f}^{A^1}(1) > 1$.
- $\mathtt{map}^{A^1} : \mathbb{N} \to \mathbb{N}$ is defined by $\mathtt{map}(0) = 1$ and $\mathtt{map}(n) = 1 + \mathtt{f}^{A^1}(n)$ for all $n \in \mathbb{N}^+$.

A concrete example of monotone order-sorted algebra in C^1 is $(A^1, <^1)$ such that $\mathtt{f}^{A^1} : \mathbb{N}^+ \to \mathbb{N}^+$ is defined by $\mathtt{f}^{A^1}(n) = (p+2) * n$ for all $n \in \mathbb{N}^+$, where p is any natural number. Conditions (a) and (b) above are sufficient for proving that the term rewriting rules of LIST[FUN] are decreasing.

- $\mathtt{map}^{A^1}(\mathtt{empty}^{A^1}) = 1 > 0 = \mathtt{empty}^{A^1}$;
- Let ϑ: {E : Elt, L : List} $\to A^1$ be an arbitrary valuation. We have

  $\vartheta(\mathtt{map}(\mathtt{E} : \mathtt{Elt}\ ;\ \mathtt{L} : \mathtt{List})) = \mathtt{map}^{A^1}(\vartheta(\mathtt{E} : \mathtt{Elt}) ;^{A^1} \vartheta(\mathtt{L} : \mathtt{List})) =$

  $1 + \mathtt{f}^{A^1}(1 + \vartheta(\mathtt{E} : \mathtt{Elt}) + \vartheta(\mathtt{L} : \mathtt{List})) \geq$

  $1 + \mathtt{f}^{A^1}(1) + \mathtt{f}^{A^1}(\vartheta(\mathtt{E} : \mathtt{Elt})) + \mathtt{f}^{A^1}(\vartheta(\mathtt{L} : \mathtt{List})) >$

  $2 + \mathtt{f}^{A^1}(\vartheta(\mathtt{E} : \mathtt{Elt})) + \mathtt{f}^{A^1}(\vartheta(\mathtt{L} : \mathtt{List})) = \vartheta(\mathtt{f}(\mathtt{E} : \mathtt{Elt})\ ;\ \mathtt{map}(\mathtt{L} : \mathtt{List}))$.

  Since ϑ is arbitrary, $\mathtt{map}(\mathtt{E} : \mathtt{Elt}\ ;\ \mathtt{L} : \mathtt{List}) >_{A^1} \mathtt{f}(\mathtt{E} : \mathtt{Elt})\ ;\ \mathtt{map}(\mathtt{L} : \mathtt{List})$.

Let $(A^2, <^2)$ be the monotone order-sorted algebra for NAT defined in Example 50. Let $(A^1, <^1) \in C^1$ such that $\mathtt{f}^{A^1}(n) = \mathtt{double}^{A^2}(n) = 4 * n$ for all $n \in \mathbb{N}^+$. Then we have $(A^1, <^1) \restriction_\chi = (A^2, <^2) \restriction_v$. By Theorem 48, the instance LIST[NAT] is terminating.

The parameterized specification χ encapsulates all operators. LIST[FUN] and NAT are reducible. By Corollary 40, the instance LIST[NAT] is reducible. By Proposition 36, LIST[NAT] is sufficient-complete.

We present another example of typical higher-order function fold defined on sequences, which 'folds' a sequence via a binary function f.

Example 52 (Folding sequences)**.** *We define sequences of natural numbers by instantiating the parameterized specification* χ: BIN $\hookrightarrow$ SEQ[BIN] *by the view* v: BIN $\to$ NAT *as shown in Figure 12.*

We define a class C^1 of monotone order-sorted algebras $(A^1, <^1)$ for RING[BIN] such that the interpretation of all sorts in RING[BIN] and all function symbols in $F_{\mathtt{RING[BIN]}} \setminus F_{\mathtt{BIN}}$ is fixed, while the interpretation of all function symbols in BIN may vary within limits that ensure termination:

- $(A^1_s, <^1_s) = (\mathbb{N}^+, <)$ for all $s \in \{\mathtt{Elt}, \mathtt{Seq}, \mathtt{Config}\}$;

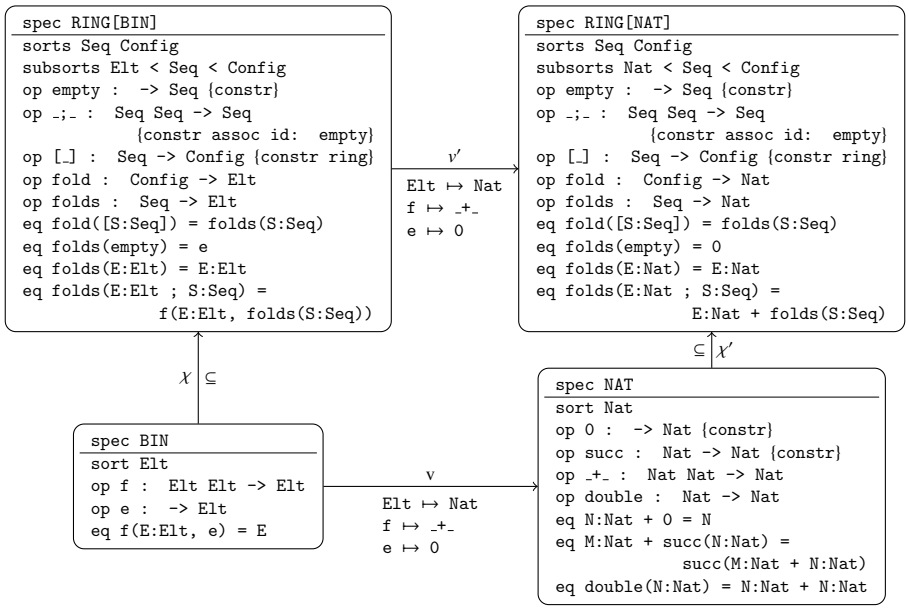


Figure 12: Folding sequences

- $\texttt{empty}^{A^1} = 1$;
- $;^{A^1} : \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{N}^+$ is defined by $n ;^{A^1} m = n + m + m * n$ for all $m, n \in \mathbb{N}^+$,
- $[\_]^{A^1} : \mathbb{N}^+ \to \mathbb{N}^+$ is defined by $[\_]^{A^1}(n) = n$ for all $n \in \mathbb{N}^+$;
- $\texttt{fold}^{A^1} : \mathbb{N}^+ \to \mathbb{N}^+$ is defined by $\texttt{fold}^{A^1}(n) = 4 * n$ for all $n \in \mathbb{N}^+$;
- $\texttt{folds}^{A^1} : \mathbb{N}^+ \to \mathbb{N}^+$ is defined by $\texttt{folds}^{A^1}(n) = 3 * n$ for all $n \in \mathbb{N}^+$;
- $\texttt{e}^{A^1} < 3$;
- $\texttt{f}^{A^1} : \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{N}^+$ is any monotone function with the following property: $3 * (m ;^{A^1} n) > \texttt{f}^{A^1}(m, 3 * n)$ for all $m, n \in \mathbb{N}^+$.

A concrete example of monotone order-sorted algebra in C^1 is $(A^1, <^1)$ such that $\texttt{e}^{A^1} = 1$ and $\texttt{f}^{A^1}(m, n) = m + 2 * n$ for any $m, n \in \mathbb{N}^+$. Notice that $;^{A^1}$ is associative and commutative, which implies that the rotative law and the reversible law hold. We show that the term rewriting rules of `RING[BIN]` are decreasing. Let ϑ: $\{\texttt{E} : \texttt{Elt}, \texttt{S} : \texttt{Seq}\} \to A^1$ be an arbitrary valuation. Then:

- $\vartheta(\texttt{fold}([\texttt{S} : \texttt{Seq}])) = 4 * \vartheta(\texttt{S}: \texttt{Seq}) > 3 * \vartheta(\texttt{S}: \texttt{Seq}) = \vartheta(\texttt{folds}(\texttt{S} : \texttt{Seq}))$
- $\vartheta(\texttt{folds}(\texttt{empty})) = 3 > \vartheta(\texttt{e})$
- $\vartheta(\texttt{folds}(\texttt{E} : \texttt{Elt})) = 3 * \vartheta(\texttt{E} : \texttt{Elt}) > \vartheta(\texttt{E} : \texttt{Elt})$
- $\vartheta(\texttt{folds}(\texttt{E} : \texttt{Elt} ; \texttt{S}: \texttt{Seq})) = 3 * (\vartheta(\texttt{E} : \texttt{Elt}) ;^{A^1} \vartheta(\texttt{S}: \texttt{Seq})) > \texttt{f}^{A^1}(\vartheta(\texttt{E} : \texttt{Elt}), 3 * \vartheta(\texttt{S}: \texttt{Seq})) = \vartheta(\texttt{f}(\texttt{E} : \texttt{Elt} ; \texttt{folds}(\texttt{S}: \texttt{Seq})))$
- $\vartheta(\texttt{f}(\texttt{E} : \texttt{Elt}, \texttt{e})) > 1 = \vartheta(\texttt{E} : \texttt{Elt})$

Let $(A^2, <^2)$ be the monotone order-sorted algebra for `NAT` defined in Example 50. Let $(A^1, <^1) \in C^1$ such that $\texttt{e}^{A^1} = 0^{A^2}$ and $\texttt{f}^{A^1}(m, n) = m +^{A^2} n$ for all $m, n \in \mathbb{N}^+$. Since $(A^1, <^1) \restriction_\chi = (A^2, <^2) \restriction_\nu$, by Theorem 48, the instance `RING[NAT]` is terminating.

Since χ introduces two operators, `fold`: `Config` $\rightarrow$ `Elt` and `folds`: `Seq` $\rightarrow$ `Elt`, to the 'old' sort `Elt`, χ does not encapsulate all operators. Therefore, we cannot apply Corollary 40. However, one can check directly that `RING`[`NAT`] is reducible. By Proposition 36, `RING`[`NAT`] is sufficient-complete.

7. Conclusions

The proof of Theorem 18 is not possible without condition 2(c)ii of Definition 11 which was introduced in this paper. It follows that the present study led to improvements within the logic framework of **COSA**. The arguments used in the proof of Theorem 18 are applicable to constructor-based logical frameworks built on top of other versions of **OSA** such as the ones presented in [23] or [24].

Our notion of sufficient-completeness is more general than the one defined in [20, 21, 22, 16] as our definition is not restricted to ground terms and it allows the constructor terms to contain operators of loose sorts. This is due to the fact that we are interested in properties of all reachable models not only initial models. For example, our framework covers the specification of lists of arbitrary elements given in Example 37, which is not sufficient-complete by the definition given in [20, 21, 22, 16]. For this reason, the present approach is closer to the applications in the formal specification and verification of systems. In addition, our framework allows conditional equations.

The technique of interpreting terms into well-founded domains is proved to be stable under pushouts of signatures using amalgamation. In the future we are planning to show that other techniques for proving termination are stable under pushouts. An interesting problem is to provide incremental proofs of termination for the specification imports. Incremental proofs of termination have been proposed in [30, 31, 16, 15] but much work is needed to cover a wider range of applications. Another future direction of research is the development of theorem proving techniques based on term rewriting in the context given by a hybrid logic such as the ones proposed in [32, 33, 34].

References

References

[1] R. M. Burstall, J. A. Goguen, The semantics of Clear, a specification language, in: D. Bjørner (Ed.), Abstract Software Specifications, 1979 Copenhagen Winter School, January 22 - February 2, 1979, Proceedings, Vol. 86 of Lecture Notes in Computer Science, Springer, 1979, pp. 292–332.

[2] A. E. Haxthausen, F. Nickl, Pushouts of order-sorted algebraic specifications, in: M. Wirsing, M. Nivat (Eds.), Algebraic Methodology and Software Technology, 5th International Conference, AMAST '96, Munich, Germany, July 1-5, 1996, Proceedings, Vol. 1101 of Lecture Notes in Computer Science, Springer, 1996, pp. 132–147.

[3] J. Meseguer, Membership algebra as a logical framework for equational specification, in: F. Parisi-Presicce (Ed.), Recent Trends in Algebraic Development Techniques, 12th International Workshop, WADT'97, Tarquinia, Italy, June 1997, Selected Papers, Vol. 1376 of Lecture Notes in Computer Science, Springer, 1997, pp. 18–61.

[4] M. Bidoit, R. Hennicker, A. Kurz, Observational logic, constructor-based logic, and their duality, Theor. Comput. Sci. 298 (3) (2003) 471–510.

[5] M. Bidoit, R. Hennicker, Constructor-based observational logic, J. Log. Algebr. Program. 67 (1-2) (2006) 3–51.

[6] D. Găină, K. Futatsugi, K. Ogata, Constructor-based logics, J. UCS 18 (16) (2012) 2204–2233.

[7] D. Găină, M. Zhang, Y. Chiba, Y. Arimoto, Constructor-based inductive theorem prover, in: R. Heckel, S. Milius (Eds.), Algebra and Coalgebra in Computer Science - 5th International Conference, CALCO 2013, Warsaw, Poland, September 3-6, 2013. Proceedings, Vol. 8089 of Lecture Notes in Computer Science, Springer, 2013, pp. 328–333.

[8] D. Găină, I. Ţuţu, A. Riesco, Specification and verification of invariant properties of transition systems, in: 25th Asia-Pacific Software Engineering Conference, APSEC 2018, Nara, Japan, December 4-7, 2018, IEEE, 2018, pp. 99–108.

[9] D. Găină, K. Futatsugi, Initial semantics in logics with constructors, Journal of Logic and Computation 25 (1) (2015) 95–116.

[10] D. Găină, Interpolation in logics with constructors, Theor. Comput. Sci. 474 (2013) 46–59.

[11] D. Găină, Downward Löwenheim-Skolem Theorem and interpolation in logics with constructors, J. Log. Comput. 27 (6) (2017) 1717–1752.

[12] R. Diaconescu, Institution-independent Model Theory, 1st Edition, Studies in Universal Logic, Birkhäuser Basel, 2008.

[13] J. Meseguer, Order-sorted parameterization and induction, in: J. Palsberg (Ed.), Semantics and Algebraic Specification, Essays Dedicated to Peter D. Mosses on the Occasion of His 60th Birthday, Vol. 5700 of Lecture Notes in Computer Science, Springer, 2009, pp. 43–80.

[14] Terese, Term Rewriting Systems, Vol. 55 of Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, 2003.

[15] M. Nakamura, K. Ogata, K. Futatsugi, Incremental proofs of termination, confluence and sufficient completeness of OBJ specifications, in: S. Iida, J. Meseguer, K. Ogata (Eds.), Specification, Algebra, and Software - Essays Dedicated to Kokichi Futatsugi, Vol. 8373 of Lecture Notes in Computer Science, Springer, 2014, pp. 92–109.

[16] F. Schernhammer, J. Meseguer, Incremental checking of well-founded recursive specifications modulo axioms, in: P. Schneider-Kamp, M. Hanus (Eds.), PPDP, ACM, 2011, pp. 5–16.

[17] H. T. T. Doan, A. Riesco, K. Ogata, An environment for specifying and model checking mobile ring robot algorithms, in: M. Ghaffari, M. Nesterenko, S. Tixeuil, S. Tucci, Y. Yamauchi (Eds.), Stabilization, Safety, and Security of Distributed Systems - 21st International Symposium, SSS 2019, Pisa, Italy, October 22-25, 2019, Proceedings, Vol. 11914 of Lecture Notes in Computer Science, Springer, 2019, pp. 111–126.
[18] J. Goguen, R. Burstall, Institutions: Abstract model theory for specification and programming, Journal of the Association for Computing Machinery 39 (1) (1992) 95–146.
[19] R. Diaconescu, J. Goguen, P. Stefaneas, Logical support for modularisation, in: Logical Environments, Cambridge University Press, 1993, pp. 83–130.
[20] J.-P. Jouannaud, E. Kounalis, Automatic proofs by induction in equational theories without constructors, in: A. Meyer (Ed.), Proceedings of the First Annual IEEE Symp. on Logic in Computer Science, LICS 1986, IEEE Computer Society Press, 1986, pp. 358–366.
[21] D. Kapur, P. Narendran, H. Zhang, On sufficient-completeness and related properties of term rewriting systems, Acta Inf. 24 (4) (1987) 395–415.
[22] D. Kapur, P. Narendran, D. J. Rosenkrantz, H. Zhang, Sufficient-completeness, ground-reducibility and their complexity, Acta Inf. 28 (4) (1991) 311–350.
[23] J. A. Goguen, J. Meseguer, Order-sorted algebra I: Equational deduction for multiple inheritance, overloading, exceptions and partial operations, Theoretical Computer Science 105 (2) (1992) 217–273.
[24] A. Poigné, Parametrization for order-sorted algebraic specification, J. Comput. Syst. Sci. 40 (2) (1990) 229–268.
[25] K. Futatsugi, D. Găină, K. Ogata, Principles of proof scores in CafeOBJ, Theor. Comput. Sci. 464 (2012) 90–112.
[26] R. Diaconescu, K. Futatsugi, CafeOBJ Report, World Scientific, Singapore, 1998.
[27] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, C. L. Talcott, All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic, Vol. 4350 of Lecture Notes in Computer Science, Springer, 2007.
[28] J. A. Goguen, Higher-Order Functions Considered Unnecessary for Higher-Order Programming, Addison-Wesley Longman Publishing Co., Inc., USA, 1990, pp. 309–351.
[29] S. Lucas, C. Marché, J. Meseguer, Operational termination of conditional term rewriting systems, Inf. Process. Lett. 95 (4) (2005) 446–453.
[30] X. Urbain, Modular & incremental automated termination proofs, J. Autom. Reasoning 32 (4) (2004) 315–355.
[31] C. Marché, X. Urbain, Modular and incremental proofs of AC-termination, J. Symb. Comput. 38 (1) (2004) 873–897.
[32] D. Găină, Birkhoff style calculi for hybrid logics, Formal Asp. Comput. 29 (5) (2017) 805–832.
[33] D. Găină, Foundations of logic programming in hybrid logics with user-defined sharing, Theor. Comput. Sci. 686 (2017) 1–24.
[34] D. Găină, Forcing and calculi for hybrid logics, Journal of the ACM 67 (4) (2020) 1–55.