

A Total Approach to Partial Algebraic Specification

José Meseguer¹ and Grigore Roşu²

¹ University of Illinois at Urbana-Champaign

² NASA Ames Research Center - RIACS
M/S 269-2, Moffett Field, CA 94035-1000
Tel: +1 (650) 604-4609, Fax: +1 (650) 604-3594
`grosu@email.arc.nasa.gov`

Abstract. Partiality is a fact of life, but at present explicitly partial algebraic specifications lack tools and have limited proof methods. We propose a sound and complete way to support execution and formal reasoning of explicitly partial algebraic specifications within the total framework of membership equational logic (MEL) which has a high-performance interpreter (Maude) and proving tools. This is accomplished by a sound and complete mapping $\text{PMEL} \rightarrow \text{MEL}$ of *partial* membership equational (PMEL) theories into total ones. Furthermore, we characterize and give proof methods for a practical class of theories for which this mapping has “almost-zero representational distance,” in that the partial theory and its total translation are *identical* up to minor syntactic sugar conventions. This then supports very direct execution of, and formal reasoning about, partial theories at the total level. In conjunction with tools like Maude and its proving tools, our methods can be used to execute and reason about partial specifications such as those in the equational sublanguage of CASL.

Keywords. Algebraic Specification, Partiality, Membership Equational Logic.
Track B.

1 Introduction

Any algebraic specification formalism worth its salt has somehow to deal with partiality, because in practice many functions are partial. This can be done within *explicitly partial* formalisms, or, alternatively, within *total* formalisms supporting notions of subtype (subsort) so that partial functions then appear as appropriate restrictions to given subtypes of corresponding total functions.

Various explicitly partial specification formalisms were explored from the early days of algebraic specification by different authors, e.g., [13, 3] (see [1, 9] for surveys and bibliographies on equational specification, including partial approaches). Reichel’s book [20] gives a systematic study of algebraic specification in an explicitly partial way using *existence equations*. The comparison between explicitly partial formalisms has been facilitated by Mossakowski’s work using maps of institutions [17]. The interest in explicitly partial specifications has been recently revived by the CASL specification language [7], which supports subsorts and partial operations, with both *existence* and *strong* equations.

At present, the main practical limitations of explicitly partial equational specifications appear in the areas of *execution*, *formal reasoning*, and *tools*, where the state of the art is considerably less developed than for total approaches. Execution requires adequate notions of confluence and termination. As far as we know, the only theoretical study of such matters at the explicitly partial level is the work of Hintermeier, C. Kirchner, and H. Kirchner in the context of *galactic algebras* [12]; but we

are not aware of any systems supporting execution of explicitly partial specifications, galactic or otherwise. In fact, we are not aware of any tools that *directly support* either execution or formal reasoning for explicitly partial equational specifications. The only tool that we know of supporting theorem proving for CASL specifications is the HOL-CASL tool [18], where partial specifications are translated into total ones in higher-order logic [18, 19]; in particular, inductive reasoning is supported using second order logic in HOL. This can be viewed as an *indirect* method of supporting theorem proving for explicitly partial specifications: the reasoning happens in a different logic, and the original specifications are modified considerably by the translation.

Developing well-engineered interpreters and theorem provers that directly support execution and formal reasoning for explicitly partial specifications seems a worthwhile goal; however, this will certainly require a strong commitment along several years by several research groups to carry out the new theoretical developments required and to build appropriate tools. In the meantime, are there any viable alternatives to support both execution and reasoning for partial specifications? The main goal of this paper is to propose one such alternative, based on *total* equational specifications.

An important advantage of our proposal is that there are several interpreters and theorem proving tools for total equational specification formalisms with varying degrees of support for partiality via subsorts: two based on order-sorted algebra (OBJ3 [11] and CafeOBJ [8]) and one based on membership equational logic (Maude [5]). Specifically, our proposal uses a *logical framework* approach based on the ideas:

1. *unification* of different explicitly partial equational specification formalisms (including all partial formalisms in the so-called Mossakowski’s web [17]) within the *partial membership equational logic framework* (PMEL) [15];
2. a *conservative map of logics* embedding PMEL within the (*total*) *membership equational logic* framework (MEL) [15]; this embedding preserves initial algebras, free algebras, and left adjoints to forgetful functors induced by theory maps in both directions; furthermore, one can *borrow* the MEL inference system to get a complete PMEL inference system [4, 15];
3. characterization of a class of theories of practical interest for which the above embedding $\text{PMEL} \rightarrow \text{MEL}$ becomes a mapping with *almost-zero representational distance*; that is, except for minor syntactic sugar differences, the partial specification remains *unchanged* when translated to a total one.
4. execution in a tool supporting MEL specifications such as Maude of the corresponding total versions of the explicitly partial equational specifications, which can then be reinterpreted as execution of the original PMEL specifications; similarly, inductive theorem proving support for PMEL specifications using tools such as the Maude ITP and other Maude formal tools [6].

The theoretical foundations underlying 1–2 were already developed in [15]. Based on such foundations it is possible to execute and reason about PMEL specifications via their corresponding MEL translation. However, for *arbitrary* PMEL specifications the MEL-based execution and reasoning are somewhat indirect, in the sense that the original PMEL specification *has to be modified*. This is because *definedness* in PMEL is expressed as *having a sort* in MEL, but definedness is always *inherited by subterms*.

For example, a PMEL existence equation $f(g(x, y)) = k(h(x))$ (stating that both terms are defined and are equal) with x, y , say of sort s , must be translated into the set of MEL axioms, called its *envelope*, $f(g(x, y)) = k(h(x)), f(g(x, y)) : \top_1$,

$g(x, y) : \top_2$, $k(h(x)) : \top_1$, and $h(x) : \top_3$, where \top_1, \top_2, \top_3 are the *top sorts* in the appropriate connected components of the sort poset.

In this work, we give conditions and proof techniques allowing us to embed, execute, and reason about explicitly partial specifications in the MEL total framework *without modifying them*, without having to add envelopes (ideas 3–4). Thus, because of the almost-zero representational distance thus obtained and the “borrowing” of the MEL logic to get a sound and complete PMEL proof system, we can support execution and reasoning for PMEL specifications in a *very direct way* within a total framework. In particular, since the PMEL specifications that pass the test do not have to be modified, for *execution purposes* we can rely on the well-developed theory of MEL execution by rewriting presented in [2] and supported by the Maude interpreter.

2 Total Membership Equational Logic

In this section we recall some (total) membership equational logic (MEL) definitions and notations needed in the paper. The interested reader is referred to [15] for a comprehensive exposition of MEL.

A *signature* Ω in MEL is a triple (K, Σ, π) where K is a set of *kinds*, Σ is a K -sorted (or, more precisely, K -kinded) algebraic signature, and $\pi : S \rightarrow K$ is a function that assigns to each element in its domain, called a *sort*, a kind. Given a signature Ω in MEL, an Ω -(*membership*) *algebra* A is a Σ -algebra together with a set $A_s \subseteq A_{\pi(s)}$ for each sort $s \in S$, and an Ω -*homomorphism* $h : A \rightarrow B$ is a Σ -morphism such that for each $s \in S$ we have $h_{\pi(s)}(A_s) \subseteq B_s$. We let \mathbf{Alg}_Ω denote the category of Ω -algebras and Ω -homomorphisms.

Given a signature Ω and a K -indexed set of *variables*, an *atomic* (Ω, X) -*equation* has the form $t = t'$ where $t, t' \in T_{\Sigma, k}(X)$, and an *atomic* (Ω, X) -*membership* has the form $t : s$ where s is a sort and $t \in T_{\Sigma, \pi(s)}(X)$. An Ω -*sentence* in MEL has the form $(\forall X) a$ **if** $a_1 \wedge \dots \wedge a_n$, where a, a_1, \dots, a_n are atomic (Ω, X) -equations or (Ω, X) -memberships, and $\{a_1, \dots, a_n\}$ is a set (no duplications). If $n = 0$ then the Ω -sentence is called *unconditional* and written $(\forall X) a$. Given an Ω -algebra A and a map $\theta : X \rightarrow A$, then $A, \theta \models_\Omega t = t'$ iff $A_{\theta(t)} = A_{\theta(t')}$, and $A, \theta \models_\Omega t : s$ iff $A_{\theta(t)} \in A_s$. A *satisfies* $(\forall X) a$ **if** $a_1 \wedge \dots \wedge a_n$, written $A \models_\Omega (\forall X) a$ **if** $a_1 \wedge \dots \wedge a_n$, iff for each $\theta : X \rightarrow A$, if $A, \theta \models_\Omega a_1$ and ... and $A, \theta \models_\Omega a_n$ then $A, \theta \models_\Omega a$. A *specification* (or *theory*) $T = (\Omega, \Gamma)$ in MEL consists of a signature Ω and a set of Ω -sentences Γ . An Ω -algebra A *satisfies* (or is a model of) $T = (\Omega, \Gamma)$, written $A \models T$, iff it satisfies each sentence in Γ . We let $\mathbf{Alg}(T)$ denote the full subcategory of \mathbf{Alg}_Ω of Ω -algebras satisfying T . We let \mathbf{Th}_{MEL} denote the category of theories in MEL (see [15] for the definition of morphisms). Theories T and T' are *semantically equivalent* in MEL, written $T \equiv_{\text{MEL}} T'$, iff they have the same models, i.e., iff $\mathbf{Alg}(T) = \mathbf{Alg}(T')$. It is known [15] that MEL admits complete deduction, so $T \equiv_{\text{MEL}} T'$ is equivalent to $T \Leftrightarrow T'$, where $(\Omega, \Gamma) \Rightarrow (\Omega', \Gamma')$ means that $\Gamma \vdash_\Omega \varphi'$ for each $\varphi' \in \Gamma'$.

To make specifications easier to read and to emphasize that order-sorted specifications are a special case of membership equational ones, the following syntactic sugar conventions are widely accepted and supported by Maude [5]:

Subsorts. Given sorts s, s' with $\pi(s) = \pi(s') = k$, the declaration $s < s'$ is syntactic sugar for the conditional membership $(\forall x : k) x : s' \text{ if } x : s$.

Operations. If $\sigma \in \Omega_{k_1 \dots k_n, k}$ and $s_1, \dots, s_n, s \in S$ with $\pi(s_1) = k_1, \dots, \pi(s_n) = k_n$, $\pi(s) = k$, then the declaration $\sigma : s_1 \times \dots \times s_n \rightarrow s$ is syntactic sugar for $(\forall x_1 : k_1, \dots, x_n : k_n) \sigma(x_1, \dots, x_n) : s \text{ if } x_1 : s_1 \wedge \dots \wedge x_n : s_n$.

Variables. $(\forall x : s, X)$ **a if** $a_1 \wedge \dots \wedge a_n$ is syntactic sugar for the Ω -sentence $(\forall x : \pi(s), X)$ **a if** $a_1 \wedge \dots \wedge a_n \wedge x : s$. With this, the operation declaration $\sigma : s_1 \times \dots \times s_n \rightarrow s$ is equivalent to $(\forall x_1 : s_1, \dots, x_n : s_n) \sigma(x_1, \dots, x_n) : s$.

3 Partial Membership Equational Logic

Partial membership equational logic (abbreviated PMEL), as shown in Section 15 of [15], has good properties as logical framework for partial algebraic specification. It generalizes in a straightforward way several well-known partial specification formalisms and furthermore can specify up to isomorphism all the categories of models for the different partial specification formalisms in [17].

If (S, \leq) is a poset then $\hat{S} = S / \equiv_{\leq}$, with \equiv_{\leq} the smallest equivalence relation containing \leq , is the set of its connected components and we will call them its *kinds* for convenience and let k or $[s]$ denote one of them. A *signature* in partial membership equational logic is a triple $\Omega = (S, \leq, \Sigma)$ with (S, \leq) a poset of *sorts* s.t. each of its connected components k has a top element \top_k , and Σ is an \hat{S} -sorted signature. A *partial Ω -algebra* is an S -indexed set with $A_s \subseteq A_{s'}$ for each $s \leq s'$, together with a *partial function* $A_f : A_{\top_{k_1}} \times \dots \times A_{\top_{k_n}} \dashrightarrow A_{\top_k}$ for each $f : k_1 \dots k_n \rightarrow k$ in Σ . We let \mathbf{PAlg}_{Ω} denote the category of partial Ω -algebras (see [15] for morphisms).

Given an S -indexed set of variables X , we let \hat{X} denote the \hat{S} -indexed set associated to X in the obvious way ($\hat{X}_k = \bigcup_{s \in k} X_s$). Notice that the same variable can occur in more than one of the sets X_s , but that $X_{[s]}$ will have only one occurrence of it. We often write $x : s$ to denote the fact that $x \in X_s$, and X is often given by a list of pairs (variable:sort) separated by comma. Then an *atomic (Ω, X) -equation* has the form $t = t'$ where $t, t' \in T_{\Sigma, k}(\hat{X})$, and an *atomic (Ω, X) -membership* has the form $t : s$, where $t \in T_{\Sigma, [s]}(\hat{X})$. An Ω -sentence in PMEL has the form $(\forall X)$ **a if** $a_1 \wedge \dots \wedge a_n$, where a, a_1, \dots, a_n are atomic (Ω, X) -equations or atomic (Ω, X) -memberships, and a_1, \dots, a_n form a set (no duplications). If $n = 0$ then the Ω -sentence is called *unconditional* and written $(\forall X)$ **a**. As in membership equational logic, (total) operation declarations $\sigma : s_1 \dots s_m \rightarrow s$ are syntactic sugar for formulae $(\forall x_1 : s_1, \dots, x_m : s_m) \sigma(x_1, \dots, x_m) : s$. Given a partial Ω -algebra A and a map $\theta : X \rightarrow A$, then $A, \theta \models_{\Omega} t = t'$ iff $A_{\theta(t)}$ and $A_{\theta(t')}$ are both defined and $A_{\theta(t)} = A_{\theta(t')}$, and $A, \theta \models_{\Omega} t : s$ iff $A_{\theta(t)}$ defined and $A_{\theta(t)} \in A_s$. A *satisfies* $(\forall X)$ **a if** $a_1 \wedge \dots \wedge a_n$, written $A \models_{\Omega} (\forall X)$ **a if** $a_1 \wedge \dots \wedge a_n$, if and only if for each $\theta : X \rightarrow A$, if $A, \theta \models_{\Omega} a_1$ and ... and $A, \theta \models_{\Omega} a_n$ then $A, \theta \models_{\Omega} a$. In this paper, we consider therefore what is called *existential* satisfaction in the literature on partiality. Other types of satisfaction (weak, strong) can be defined using the existential one in most practical situations; a more comprehensive study of these will be given elsewhere in a work dedicated entirely to PMEL.

A *specification* (or *theory*) $P = (\Omega, \Gamma)$ in PMEL consists of a signature Ω and a set of Ω -sentences Γ . A partial Ω -algebra A satisfies (or is a model of) $P = (\Omega, \Gamma)$, written $A \models P$, iff it satisfies each sentence in Γ . Given a partial Ω -theory P , we let $\mathbf{PAlg}(P)$ denote the full subcategory of \mathbf{PAlg}_{Ω} of partial algebras satisfying P . Ω -theories P and P' are semantically equivalent in PMEL, written $P \equiv_{\text{PMEL}} P'$, iff they have the same models, i.e., $\mathbf{PAlg}(P) = \mathbf{PAlg}(P')$.

Example 1. We give a partial membership equational specification for (small) categories. Let the kind k have two sorts, *Object* and *Arrow*, such that *Object* $<$ *Arrow*

(the identity arrow is identified with its object), and two total operation declarations $s, t: \text{Arrow} \rightarrow \text{Object}$, which are just syntactic sugar for the formulae $(\forall x : k) s(x) : \text{Object}$ if $x : \text{Arrow}$ and $(\forall x : k) t(x) : \text{Object}$ if $x : \text{Arrow}$. Consider also an operation $;- : k \times k \rightarrow k$ and the following formulae:

$$\begin{aligned}
(\forall O : \text{Object}) s(O) = O & \quad (1) \\
(\forall O : \text{Object}) t(O) = O & \quad (2) \\
(\forall O : \text{Object}, A : \text{Arrow}) O; A = A \text{ if } s(A) = O & \quad (3) \\
(\forall O : \text{Object}, A : \text{Arrow}) A; O = A \text{ if } t(A) = O & \quad (4) \\
(\forall A, A' : \text{Arrow}) A; A' : \text{Arrow} \text{ if } t(A) = s(A') & \quad (5) \\
(\forall A, A' : \text{Arrow}) t(A) = s(A') \text{ if } A; A' : \text{Arrow} & \quad (6) \\
(\forall A, A' : \text{Arrow}) s(A; A') = s(A) \text{ if } t(A) = s(A') & \quad (7) \\
(\forall A, A' : \text{Arrow}) t(A; A') = t(A') \text{ if } t(A) = s(A') & \quad (8) \\
(\forall A, A', A'' : \text{Arrow}) (A; A'); A'' = A; (A'; A'') \text{ if } t(A) = s(A') \wedge t(A') = s(A'') & \quad (9)
\end{aligned}$$

Equations (1), (2) give the source and the target of an identity, (3) and (4) are the usual identity axioms, (5) and (6) define composition of arrows as a partial operation, (7) and (8) give the source and the target of the composition of two arrows when defined, and (9) states the associativity of composition. The partial algebras satisfying the theory above are exactly the (small) categories.

We recall the notion of Ω -envelope of a PMEL sentence [15]:

$$\begin{aligned}
\text{env}_\Omega(t = t') &= t = t' \wedge \bigwedge_{u \in NVST(t, t')} u : \top_{k(u)} \\
\text{env}_\Omega(t : s) &= t : s \wedge \bigwedge_{u \in NVST(t) - \{t\}} u : \top_{k(u)} \\
\text{env}_\Omega((\forall X) a \text{ if } a_1 \wedge \dots \wedge a_n) &= \{(\forall X) a' \text{ if } \text{env}_\Omega(a_1, \dots, a_n) \mid a' \in \text{env}_\Omega(a)\},
\end{aligned}$$

where $NVST(t)$ and $NVST(t, t')$ are the sets of all non-variable subterms of t and of t and t' , respectively, $\text{env}_\Omega(a_1, \dots, a_n)$ is a shorthand for $\text{env}_\Omega(a_1) \wedge \dots \wedge \text{env}_\Omega(a_n)$, and $a' \in \text{env}_\Omega(a)$ means that a' is among the conjuncts in $\text{env}_\Omega(a)$. Being concerned with defining the institution of PMEL, [15] actually extended the definition of sentences to allow conjunctions as conclusions. The envelope map extends trivially to sets of sentences and moreover, given a specification $P = (\Omega, \Gamma)$ in PMEL, the *envelope* of P is the specification $\text{env}(P) = (\Omega, \text{env}_\Omega(\Gamma))$. Notice that $\text{env}(\text{env}(P)) = \text{env}(P)$ and that $P \equiv_{\text{PMEL}} \text{env}(P)$.

Example 2. The envelope of the specification of categories in Example 1, written in shorten notation and omitting the declarations for the total operations, is:

$$\begin{aligned}
(\forall O : \text{Object}) s(O) = O & \quad (1) \\
(\forall O : \text{Object}) s(O) : \text{Arrow} & \quad (1.1) \\
(\forall O : \text{Object}) t(O) = O & \quad (2) \\
(\forall O : \text{Object}) t(O) : \text{Arrow} & \quad (2) \\
(\forall O : \text{Object}, A : \text{Arrow}) O; A = A \text{ if } s(A) = O \wedge s(A) : \text{Arrow} & \quad (3) \\
(\forall O : \text{Object}, A : \text{Arrow}) O; A : \text{Arrow} \text{ if } s(A) = O \wedge s(A) : \text{Arrow} & \quad (3.1) \\
(\forall O : \text{Object}, A : \text{Arrow}) A; O = A \text{ if } t(A) = O \wedge t(A) : \text{Arrow} & \quad (4) \\
(\forall O : \text{Object}, A : \text{Arrow}) A; O : \text{Arrow} \text{ if } t(A) = O \wedge t(A) : \text{Arrow} & \quad (4.1) \\
(\forall A, A' : \text{Arrow}) A; A' : \text{Arrow} \text{ if } C(A, A') & \quad (5) \\
(\forall A, A' : \text{Arrow}) t(A) = s(A') \text{ if } A; A' : \text{Arrow} & \quad (6) \\
(\forall A, A' : \text{Arrow}) t(A) : \text{Arrow} \text{ if } A; A' : \text{Arrow} & \quad (6.1) \\
(\forall A, A' : \text{Arrow}) s(A') : \text{Arrow} \text{ if } A; A' : \text{Arrow} & \quad (6.2) \\
(\forall A, A' : \text{Arrow}) s(A; A') = s(A) \text{ if } C(A, A') & \quad (7) \\
(\forall A, A' : \text{Arrow}) s(A; A') : \text{Arrow} \text{ if } C(A, A') & \quad (7.1) \\
(\forall A, A' : \text{Arrow}) s(A) : \text{Arrow} \text{ if } C(A, A') & \quad (7.2) \\
(\forall A, A' : \text{Arrow}) t(A; A') = t(A') \text{ if } C(A, A') & \quad (8) \\
(\forall A, A' : \text{Arrow}) t(A; A') : \text{Arrow} \text{ if } C(A, A') & \quad (8.1) \\
(\forall A, A' : \text{Arrow}) t(A') : \text{Arrow} \text{ if } C(A, A') & \quad (8.2) \\
(\forall A, A', A'' : \text{Arrow}) (A; A'); A'' = A; (A'; A'') \text{ if } C(A, A') \wedge C(A', A'') & \quad (9) \\
(\forall A, A', A'' : \text{Arrow}) (A; A'); A'' : \text{Arrow} \text{ if } C(A, A') \wedge C(A', A'') & \quad (9.1) \\
(\forall A, A', A'' : \text{Arrow}) A; (A'; A'') : \text{Arrow} \text{ if } C(A, A') \wedge C(A', A'') & \quad (9.2) \\
(\forall A, A', A'' : \text{Arrow}) A; A' : \text{Arrow} \text{ if } C(A, A') \wedge C(A', A'') & \quad (9.3) \\
(\forall A, A', A'' : \text{Arrow}) A'; A'' : \text{Arrow} \text{ if } C(A, A') \wedge C(A', A'') & \quad (9.4)
\end{aligned}$$

where $C(X, Y)$ is a shorthand for “ $t(X) = s(Y) \wedge t(X) : \text{Arrow} \wedge s(Y) : \text{Arrow}$ ”.

4 From Partial to Total Membership Equational Logic

Since MEL admits complete deduction and since it is already efficiently implemented, one would want to reduce PMEL proof tasks to MEL proof tasks. More precisely, one would like to have an efficient and automatic translator, say tr , taking Ω -specifications P and Ω -sentences φ in PMEL to sentences and specifications $tr(P)$ and $tr(\varphi)$ such that $P \models \varphi$ if and only if $tr(P) \models tr(\varphi)$. Notice that such a translator would automatically produce complete deduction for PMEL. It is worth mentioning that such translators that “borrow” other logics are usually obtained from (generalized) maps of institutions [4, 15].

There is a straightforward translation of specifications in PMEL to specifications in MEL, in which (almost) nothing changes. Unfortunately, this translation is not always correct as seen shortly. There is another translation, based on the construction of envelopes, which is always correct but usually unnecessarily long. We show conditions under which the two translations are equivalent, thus providing a simple way to use an executable specification system like Maude [5] and its proving tools [6] to execute and verify properties of specifications in PMEL.

4.1 A Straightforward Translation

The simplest possible translation of PMEL sentences and specifications is to let them (almost) unchanged. More precisely, given a specification $P = (\Omega, T)$ in PMEL, let \overline{P} be $(\overline{\Omega}, Ax_{\overline{\Omega}} \cup \overline{T})$ in MEL, where if $\Omega = (S, <, \Sigma)$ and φ is an Ω sentence in PMEL, then

- $\overline{\Omega}$ is the signature (K, Σ, π) in MEL with $K = \hat{S}$ and $\pi(s) = [s]$;
- $Ax_{\overline{\Omega}}$ is the set of $\overline{\Omega}$ -sentences associated to Ω by the expected syntactic sugar conventions, i.e., containing a sentence $(\forall x:k) x:s' \text{ if } x:s$ for each pair $s < s'$ in connected component k ;
- $\overline{\varphi}$ is exactly φ under the appropriate syntactic sugar conventions for MEL and the appropriate reinterpretation of operations in Ω into operations in $\overline{\Omega}$; this operation can be trivially extended to sets of Ω -sentences.

We call $(\overline{\cdot}) : \mathbf{Th}_{\text{PMEL}} \rightarrow \mathbf{Th}_{\text{MEL}}$ the *desugaring* map, let $\mathbf{Th}_{\text{MEL}}^{\top}$ denote its image, and call the theories in $\mathbf{Th}_{\text{MEL}}^{\top}$ *topped*. Because of the consistent syntactic sugar conventions in both PMEL and MEL, there should be no syntactic difference between P and \overline{P} in practice.

The theory of small categories in Example 1 is what is called later a *duplex theory*, which implies (by Definition 2) that $P \models (\forall X) t = t'$ iff $\overline{P} \models (\forall X) t = t'$ and $\overline{P} \models (\forall X) t : \text{Arrow}$.

Intuitively, the straightforward translation from PMEL to MEL presented in this subsection is *the* desired translation, because it has what it can be called “almost-zero representation distance” between P and \overline{P} . A natural question is, however, whether it is correct or not. As seen later in the paper, it is correct for the example above but, as shown in the extended version [16], it is neither complete nor sound in general, that is, there are Ω -theories P, P' and Ω -sentences φ, φ' in PMEL such that $P \models \varphi$ but $\overline{P} \not\models \overline{\varphi}$, and $P' \not\models \varphi'$ but $\overline{P'} \models \overline{\varphi'}$.

It is a major goal of this paper to give conditions under which the straightforward translation from PMEL to MEL is both sound and complete.

4.2 The Correct Translation

Given a signature $\Omega = (S, <, \Sigma)$ and an Ω -sentence $\varphi = (\forall X) a \text{ if } a_1 \wedge \dots \wedge a_n$ in PMEL, let $\overline{\Omega}$ and Ax_{Ω} be defined as in Subsection 4.1, and let $\overline{env}_{\Omega}(\varphi)$ be the set of $\overline{\Omega}$ -sentences $\{(\forall X) a' \text{ if } env_{\Omega}(a_1, \dots, a_n) \mid a' \in env_{\Omega}(a)\}$ (we used the same notational conventions as in Section 3), with the appropriate interpretation of kinds and operations in Ω into kinds and operations in $\overline{\Omega}$. \overline{env}_{Ω} can be extended to sets of Ω -formulae in the obvious way. Given a specification $P = (\Omega, \Gamma)$ in PMEL, let $\overline{env}(P)$ be the specification in MEL $(\overline{\Omega}, Ax_{\Omega} \cup \overline{env}_{\Omega}(\Gamma))$. Note that $\overline{env}(P) = env(P)$. For a topped theory $\overline{P} \in \mathbf{Th}_{\text{MEL}}^{\top}$, let $env^{\top}(\overline{P})$ be $\overline{env}(P)$.

It is important to notice that despite the fact that P is semantically equivalent to $env(P)$ in PMEL, it is *not* the case that \overline{P} and $\overline{env}(P)$ are equivalent in MEL, because the straightforward translation does not preserve the semantical equivalence of specifications in PMEL (see Subsection 4.1).

As shown in [15], for any signature Ω in PMEL, any $\overline{\Omega}$ -algebra A and any partial Ω -algebra B , one can build a partial Ω -algebra A° by forgetting all the elements that don't have a sort and letting the operations undefined accordingly, and one can build an $\overline{\Omega}$ -algebra B^{\bullet} by freely adding to B all the undefined, or "error", values. Moreover, given an Ω -theory P in PMEL then $(-)^{\circ}$ and $(-)^{\bullet}$ can be organized as functors $(-)^{\circ} : \mathbf{Alg}(env^{\top}(P)) \rightarrow \mathbf{PAlg}(P)$ and $(-)^{\bullet} : \mathbf{PAlg}(P) \rightarrow \mathbf{Alg}(env^{\top}(P))$. The following results are proved in Section 14 in [15]:

Proposition 1. *Given an Ω -theory P and an Ω -sentence φ in PMEL, and a (total) $\overline{\Omega}$ -algebra A , then*

1. *The functor $(-)^{\bullet} : \mathbf{PAlg}(P) \rightarrow \mathbf{Alg}(\overline{env}(P))$ is a left adjoint left inverse to $(-)^{\circ} : \mathbf{Alg}(\overline{env}(P)) \rightarrow \mathbf{PAlg}(P)$,*
2. *$(A \models_{\overline{\Omega}} \overline{env}_{\Omega}(\varphi) \text{ iff } A^{\circ} \models_{\Omega} \varphi)$ and $(A \models \overline{env}(P) \text{ iff } A^{\circ} \models P)$,*
3. *$P \models \varphi \text{ iff } \overline{env}(P) \models \overline{env}_{\Omega}(\varphi)$.*

Thus, soound complete deduction for PMEL can be obtained by defining $P \models \varphi$ iff $\overline{env}(P) \vdash \overline{env}_{\Omega}(\varphi)$.

\overline{env} can be actually organized as a generalized map of institutions [15]. Note that, due to 1. in Proposition 1, $(\mathbf{Alg}(\overline{env}(P)))^{\circ} = \mathbf{PAlg}(P)$ and there is in fact a perfect correspondence between free algebras F_P in $\mathbf{PAlg}(P)$ and free algebras $F_{\overline{env}(P)}$ in $\mathbf{Alg}(\overline{env}(P))$, since we have $F_P^{\bullet} = F_{\overline{env}(P)}$ and $F_{\overline{env}(P)}^{\circ} = F_P$ (because a composition of left adjoints is also a left adjoint). In particular, there is a perfect correspondence between initial algebras I_P in $\mathbf{PAlg}(P)$ and initial algebras $I_{\overline{env}(P)}$ in $\mathbf{Alg}(\overline{env}(P))$.

4.3 Putting All the Maps Together

There is also a map S from $\mathbf{Th}_{\text{MEL}}^{\top}$ to $\mathbf{Th}_{\text{PMEL}}$, called *sugaring*, which, due to the syntactic sugar conventions in both PMEL and MEL, is also just syntactic identity in practice. Given $T = (\Omega, \Gamma) \in \mathbf{Th}_{\text{MEL}}^{\top}$, let $S(T)$ be the theory $(S(\Omega), S(\Gamma))$ in $\mathbf{Th}_{\text{PMEL}}$, where:

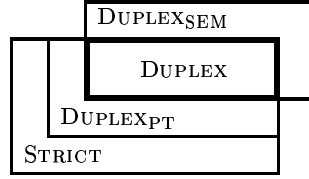
- $S(\Omega)$ is the PMEL signature obtained from Ω by forgetting its kinds (but keeping its sorts) and adding $s < s'$ for each sorts s, s' such that the sentence $(\forall x : k) x : s' \text{ if } x : s$ is in Γ ;
- $S((\forall X) a \text{ if } C)$ be the PMEL $S(\Omega)$ -sentence $(\forall \tilde{X}) a \text{ if } \tilde{C}$, where \tilde{X} is the set of (sorted) variables $\bigcup_{x:k \in X} \{x : s \mid x : s \in C\}$ and \tilde{C} is the conjunction of all the atomic sentences in $C - \{x : s \mid x : s \in C\}$.

Duplex proof-theoretic seems to be the right concept if one only considers the provability aspects of the partial and total theories. However, one may also be interested in models, in that a theory is expected to have the same models regardless of how it is viewed, as total or as partial, i.e., to be duplex semantic.

Theorem 1. (for a detailed proof see [16]) *The following hold:*

1. $\text{DUPLEX}_{\text{PT}} \subsetneq \text{STRICT}$;
2. $\text{DUPLEX}_{\text{SEM}} \not\subseteq \text{STRICT}$ and $\text{STRICT} \not\subseteq \text{DUPLEX}_{\text{SEM}}$;
3. $\text{DUPLEX}_{\text{PT}} \not\subseteq \text{DUPLEX}_{\text{SEM}}$ and $\text{DUPLEX}_{\text{SEM}} \not\subseteq \text{DUPLEX}_{\text{PT}}$;
4. $\text{DUPLEX}_{\text{SEM}} \cap \text{STRICT} \subsetneq \text{DUPLEX}_{\text{PT}}$.

The four subcategories of topped total membership equational logic theories can be represented as in the picture below, where each rectangle represents one subcategory and each inclusion of rectangles is proper:



6 Envelope Invariance as a Central Concept

As argued above, in order for a partial theory to be safely regarded, via appropriate syntactic sugar conventions, as total it must be duplex. However, in algebraic specification practice we would like the strongest possible notion of “duplex” theory, assuring exact correspondence not only at the duplex semantic and proof-theoretic levels, but also ensuring exact correspondence of *initial algebra semantics* and of semantics of *free functors* associated to forgetful functors along a theory morphism (for *parameterized* specifications), so that proofs by *induction* at the total level can be “borrowed” at the partial level. As suggested by Proposition 1 and shown in full detail in [15], all these requirements hold when a partial theory P is translated to $\overline{\text{env}}(P)$ instead of \overline{P} , thus inspiring us to introduce the following key notion:

Definition 3. $T \in \text{Th}_{\text{MEL}}^{\top}$ is **envelope invariant**¹ iff $\text{Alg}(T) = \text{Alg}(\text{env}^{\top}(T))$. We let ENVINV denote the full subcategory of envelope invariant theories.

If P is a theory in PMEL such that \overline{P} is envelope invariant then notice that $\text{Alg}(\overline{P}) = \text{Alg}(\overline{\text{env}}(P))$. Therefore, by Proposition 1 (also the comments after it) there is an exact correspondence between free models of P in PMEL and free models of \overline{P} in MEL , which allows one to do inductive proofs in \overline{P} , using the well-understood machinery of total MEL efficiently implemented in Maude [5] and its inductive theorem prover [6], rather than in P at the partial level without tool support. The following result shows that envelope invariance also satisfies the duplex requirements in the previous section, thus giving it a central role in our total approach to partiality.

Theorem 2. (for a detailed proof see [16]) $\text{ENVINV} \subsetneq \text{DUPLEX}$.

¹ What we here call an envelope invariant theory was called a *strict* theory in [15].

7 Proving Envelope Invariance

Envelope invariance, besides being semantically and proof-theoretically the strongest and most satisfactory notion of “duplex” theory, is also a property that holds for many theories of interest and can be established by practical proof methods.

First note that, since $T \in \mathbf{Th}_{\text{MEL}}^\top$ is envelope invariant iff we have $\mathbf{Alg}(T) = \mathbf{Alg}(\text{env}^\top(T))$, by the completeness theorem of MEL [15], this holds iff $T \Leftrightarrow \text{env}^\top(T)$. Using the inference rules in [15], this gives us a general *semidecision procedure* for envelope invariance by trying to prove both implications in parallel. We conjecture that for an arbitrary theory T the property may be undecidable.

Nevertheless, there are large classes of theories of interest in algebraic specification practice, for which either envelope invariance holds, or practical algorithms exist that will either decide the property or output useful proof obligations. For example, *sortable* theories are envelope invariant, where $T \in \mathbf{Th}_{\text{MEL}}^\top$ is sortable iff its membership axioms consisting of the subsort inclusions and the operator declarations can prove $(\forall X) u : \top_k$, for any subterm u in any axiom in T . Such proofs are decidable, because *order-sorted parsing* of terms is decidable [10]; yet $S(T)$ may still have models in which some operations are partial.

A broad class of practical interest is the class of those $T \in \mathbf{Th}_{\text{MEL}}^\top$ that are *Church-Rosser* and *terminating* in the precise sense of [2]. This is a key class of *executable specifications*, for which we can use tools like Maude [5]. In such a class we can try to decide the equivalence $T \Leftrightarrow \text{env}^\top(T)$ by rewriting. That is, we can use T and the **implication introduction** rule in [15] to try to prove $T \Rightarrow \text{env}^\top(T)$ by rewriting in an automated way; and in the exact same way we can use $\text{env}^\top(T)$ to try to prove $T \Leftarrow \text{env}^\top(T)$. The details of how this proof method establishes the envelope invariance of the category theory specification in Section 3 can be found in <http://ase.arc.nasa.gov/grosu/download/categ.maude>. For theories $T \in \mathbf{Th}_{\text{MEL}}^\top$ that are Church-Rosser, terminating and *unconditional* (in their sugared form), checking envelope invariance by the above method becomes *decidable*, because in such a case we have $T \subseteq \text{env}^\top(T)$, so that we only need to prove $T \Rightarrow \text{env}^\top(T)$, which, with T Church-Rosser and terminating and with $\text{env}^\top(T)$ unconditional, is decidable by rewriting in T .

It does not seem obvious that $T \in \mathbf{Th}_{\text{MEL}}^\top$ Church-Rosser and terminating implies that $\text{env}^\top(T)$ is so too. For this reason we are also experimenting with a more conservative *incremental* automated proof procedure. This method generalizes to conditional theories the method described above for unconditional ones: it only uses rewriting in (increasing subsets of) T to establish the equivalence $T \Leftrightarrow \text{env}^\top(T)$. Details of how this method proves envelope invariance for the theories of categories and of categories with pullbacks can be found on the web at the address <http://ase.arc.nasa.gov/grosu/download/categ-tower.maude>. More experimentation is needed to ascertain which of these two approaches, or some combination, is the best candidate for implementation, but it seems clear that an automated procedure along these lines will be able to decide envelope invariance of Church-Rosser and terminating specifications in many cases, or otherwise will output useful proof obligations to the user. Such proof obligations may also help uncover subtle flaws in a specification.

8 Conclusion and Future Work

We have proposed a total approach to partial equational specification that, under reasonable conditions, has almost-zero representational distance, in the sense that the partial theory and its total translation are identical up to syntactic sugar. For this purpose, we have studied and compared in full detail several natural notions of “strict” and “duplex” theory that can be viewed as partial or total without changing the axioms. The main conclusion of our study is that *envelope invariance* is the *semantically stronger* notion of duplex theory, including preservation *in both directions* of initial, free, and free extension algebras, and *allowing doing both execution and inductive theorem proving* for partial specifications at the total level in a very direct way.

We also have begun the study of and experimentation with practical *proof methods* that can be used to check envelope invariance and can decide the property in some cases. Our initial experiments suggest that many theories of practical interest will pass the check, and that the check is useful to *detect semantic errors* in large partial specifications, such as those in [14].

With the exception of [12], at present there seems to be limited direct theoretical support for *rewriting techniques* for explicitly partial equational specifications, and no automated deduction techniques directly supporting inductive theorem proving for such specifications, except for the indirect support via a translation into higher-order logic of the HOL-CASL tool [18]. In particular, we know of no tools supporting execution of explicitly partial equational specifications. One important practical application of our approach is that our methods, combined with existing tools for execution and theorem proving of MEL specifications such as the Maude interpreter [5], the Maude inductive theorem prover, and the rest of Maude’s formal tool environment [6] offer a promising *short- and mid-term alternative* for very direct execution and formal analysis of explicitly partial equational specifications. In particular, our approach could be used to execute and prove properties of partial specifications in the equational subset of CASL [7].

However, more research is needed to further advance these ideas and to develop tools supporting them, including research in:

1. experimentation with *case studies* and systematic comparison of the relative advantages of different *proof procedures* for checking envelope invariance;
2. development of *checking tools* automating such proof procedures and integration of such tools in the Maude system;
3. study of *initial and free notions of envelope invariance*; that is, conditions under which theories with initiality and freeness constraints (as opposed to theories with “loose semantics”) are envelope invariant;
4. more generally, study of *modularity issues*, including study of conditions under which module operations themselves are “duplex”. We already know that PMEL and MEL are related by an “extension map of institutions” preserving many important semantic properties such as initial and free models [15], but, as shown for example in [19], modularity issues are quite delicate and should be studied carefully.

References

1. M. Bidoit, H.-J. Kreowski, P. Lescanne, F. Orejas, and D. Sannella, editors. *Algebraic System Specification and Development. A Survey and Annotated Bibliography*, volume 501 of *LNCS*. Springer, 1991.
2. A. Bouhoula, J.-P. Jouannaud, and J. Meseguer. Specification and proof in membership equational logic. *Theoretical Computer Science*, 236:35–132, 2000.
3. M. Broy and M. Wirsing. Partial abstract types. *Acta Informatica*, 18:47–64, 1982.
4. M. Cerioli and J. Meseguer. May I borrow your logic? (Transporting logical structures along maps). *Theoretical Computer Science*, 173(2):311–347, 1997.
5. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and J. Quesada. Maude: specification and programming in rewriting logic. SRI International, January 1999, <http://maude.csl.sri.com>.
6. M. Clavel, F. Durán, S. Eker, and J. Meseguer. Building equational proving tools by reflection in rewriting logic. In *CAFE: An Industrial-Strength Algebraic Formal Method*. Elsevier, 2000. <http://maude.csl.sri.com>.
7. Cofi task group on semantics, CASL — The common algebraic specification language, Semantics. www.brics.dk/Projects/CoFI/Documents/CASL, July 1999.
8. R. Diaconescu and K. Futatsugi. *CafeOBJ Report: The Language, Proof Techniques, and Methodologies for Object-Oriented Algebraic Specification*. World Scientific, 1998. AMAST Series in Computing, volume 6.
9. M. Gogolla and M. Cerioli. What is an Abstract Data Type after all? Technical report, DISI – University of Genova, 1994.
10. J. Goguen and J. Meseguer. Order-sorted algebra I: Equational deduction for multiple inheritance, overloading, exceptions and partial operations. *Theoretical Computer Science*, 105(2):217–273, 1992.
11. J. Goguen, T. Winkler, J. Meseguer, K. Futatsugi, and J.-P. Jouannaud. Introducing OBJ. In *Software Engineering with OBJ: algebraic specification in action*, pages 3–167. Kluwer, 2000.
12. C. Hintermeier, C. Kirchner, and H. Kirchner. Dynamically-typed computations for order-sorted equational presentations. *Journal of Symbolic Computation*, 25(4):455–526, April 1998.
13. H. Kaphengst and H. Reichel. Initial algebraic semantics for non-context-free languages. In M. Karpinski, editor, *Fundamentals of Computation Theory*, volume 56 of *LNCS*, pages 120–126. Springer, 1977.
14. M. Lowry, T. Pressburger, and G. Roşu. Certifying domain-specific policies. In *Proceedings, International Conference on Automated Software Engineering (ASE'01)*, pages 81–90. IEEE, 2001. San Diego, California.
15. J. Meseguer. Membership algebra as a logical framework for equational specification. In *Proceedings, WADT'97*, volume 1376 of *LNCS*, pages 18–61, 1998.
16. J. Meseguer and G. Roşu. A total approach to partial algebraic specification. Extended version at ase.arc.nasa.gov/grosu/tapase.html, 2002.
17. T. Mossakowski. Equivalences among various logical frameworks of partial algebras. In *Computer Science Logic*, volume 1092 of *LNCS*, pages 403–433, 1996.
18. T. Mossakowski. Introduction into HOL-CASL. www.tzi.de/cofi/, 2001.
19. T. Mossakowski. Relating CASL with other specification languages: the institution level. *Theoretical Computer Science*, To appear. www.tzi.de/~till.
20. H. Reichel. *Initial Computability, Algebraic Specifications, and Partial Algebras*. Oxford University Press, 1987.