Amalgamation in Casl via Enriched Signatures

Lutz Schröder[1], Till Mossakowski[1], and Andrzej Tarlecki[2,3]

[1] BISS, Department of Computer Science, Bremen University
[2] Institute of Informatics, Warsaw University
[3] Institute of Computer Science, Polish Academy of Sciences

Abstract. We construct a representation of the institution of the algebraic specification language Casl in an institution called enriched Casl. The latter comes with a model theory that is compatible with the original one and, moreover, has the amalgamation property, which fails for the standard Casl institution. Moreover, using a further representation of enriched signatures in terms of left exact categories, we show that the extended institution satisfies the converse of amalgamation, thus providing a necessary and sufficient criterion for amalgamability.
Amalgamation for enriched Casl means that a variety of results for institutions with amalgamation, such as computation of normal forms and theorem proving for structured specifications, can now be used for Casl. In particular, the previously suggested institution-independent semantics of architectural specifications is actually applicable to Casl.

Introduction

The use of formal methods within the development process of software systems is important especially for complex or safety-critical systems. Here, 'formal' implies that the specification is based on a logical system, with a rigorously defined syntax and semantics. It has been recognized that structuring operations for the specification of software systems can be studied largely independently of the underlying logical system; the most prominent formalization of this concept is the notion of *institution* [15]. Several institution independent languages for structuring specifications, among them ASL+ [3, 24, 25], module algebra [12], ACT TWO [13, 14] and development graphs [21] have been investigated. The recently developed language Casl [10, 11, 22] has an institution independent semantics of both structured and architectural specifications (where the latter have the purpose of structuring *implementations* rather than specifications). An institution independent proof system for ASL-like specifications (parametrized over a proof system for the respective institution) has been developed in [7].

For each of these institution independent specification languages, central results require the underlying institution to have the so-called amalgamation property (also referred to as exactness). Roughly speaking, this property states that models of given signatures can be combined to yield a uniquely determined model of a compound signature, provided that the original models are mutually compatible. The amalgamation property allows the computation of normal forms for specifications [5, 7]; it is a prerequisite for good behaviour w.r.t. parametrization [14] and conservative extensions [12]. The combination of implementations in the semantics of architectural specifications crucially depends on amalgamation [26]. The proof system for development graphs with hiding [21], which allow a management of change for structured specifications, is sound only for institutions with amalgamation. Last but not least, a Z-like state based language has been developed over an arbitrary institution with amalgamation [4].

Many standard logical systems (like multisorted equational and first-order logic with the respective standard notions of model) admit amalgamation, so quite often this property is taken for granted in work on specification formalisms (cf. e.g. [25]). However, the expected amalgamation property fails in the setting of order-sorted algebra (when subsort relations are interpreted as arbitrary injections[1]), in particular in the CASL institution. Generally, the amalgamation property fails to hold if there are components in the models that are not named in the signatures, e.g. the implicit universe in unsorted first-order logic (which destroys amalgamation for disjoint unions of signatures), the implicit set of kinds in LF [16], the implicit set of worlds in temporal or modal logics, or the implicit subsort injections in the CASL logic.

The solution proposed here is to represent the CASL institution, in a way that is compatible with the respective model theories, in an institution (called enriched CASL) that does have amalgamation. In this way, most of the results cited above can be applied to CASL by forming compound (colimit) signatures in the enriched signature category. The institution independent form of this technique is used in the semantics of CASL architectural specifications as outlined in [26].

We refer to [1, 20] for categorical terminology left unexplained here.

1 Institutions

The considerations ahead deal with aspects of the notion of *institution* [15]. An institution I consists of a category **Sign** of *signatures*, a *model functor*

$$\mathbf{Mod} : \mathbf{Sign}^{op} \to \mathbf{CAT},$$

where **CAT** denotes the quasicategory of categories and functors [1], and further components which formalize sentences and satisfaction. In this context, we need only the model functor.

A cocone for a diagram in **Sign** is called *amalgamable* if it is mapped to a limit under **Mod**. I (or **Mod**) has the *(finite) amalgamation property* if (finite) colimit cocones are amalgamable, i.e. if **Mod** preserves (finite) limits. If, conversely, **Mod** reflects limits, I (or **Mod**) is called *definitionally complete*. (Possibly more appropriate notions such as 2-limits or bilimits [6] are out of the

[1] This overcomes the drawbacks of the subsorts-as-inclusions approach, namely that satisfaction is not closed under isomorphism, unless extra conditions like local filtration are assumed, which, however, behave bad w.r.t. colimits.

scope of this paper). Definitional completeness, which makes the colimit property a necessary condition for amalgamability of cocones, is required, e.g., to ensure 'almost completeness' for the static architectural semantics of [26]. It is easily seen that, under cocompleteness of the signature category, a model functor that has the amalgamation property is definitionally complete iff it reflects isomorphisms. Informally speaking, definitional completeness means that identifying symbols, adding a new symbol to a signature (without constraining it by axioms), or altering 'properties' of a symbol always modifies the model class.

2 Standard CASL signatures

The specification language CASL (*Common Algebraic Specification Language*) has been designed by CoFI, the international *Common Framework Initiative for Algebraic Specification and Development* [10]. Its features include first-order logic, partial functions, subsorts, and structured and architectural specifications.

We sketch the definition of CASL signatures and their models; for further details see [8, 11]. A *(standard)* CASL *signature* Σ consists of a preordered set S of sorts and sets of total and partial function symbols and predicate symbols. Function and predicate symbols are written $f : w \to s$ and $p : w$, respectively, where s is a sort and w is a string of sorts, thus determining their *name* and *profile*. Symbols with identical names are said to be in the *overloading relation* if their argument sorts have a common subsort and their result sorts (in the case of function symbols) have a common supersort. Partial function symbols may become total on subsorts, but not vice versa.

A *signature morphism* consists of an order-preserving map between the associated sort preorders and maps between the symbol sets that are compatible with symbol profiles, preserve totality (i.e. may map partial to total function symbols, but not vice versa), and preserve the overloading relation. This defines the (cocomplete) signature category **CASLsign**.

A *model* of a CASL signature is an interpretation of the sorts by sets and of the sort preorder by *injective maps* between these sets (in other words: a functor from the thin category associated to the sort preorder into the category of sets and injective maps), of the partial (total) function symbols by partial (total) functions between the sets specified by their profiles, and of the predicate symbols by relations. The interpretations of overloaded symbols are required to agree on common subsorts of the argument sorts via the corresponding subsort injections. *Model morphisms* are defined by the usual homomorphism condition (also w.r.t. the sort injections) and preservation of predicate satisfaction.

Thus, we have a category $\mathbf{Mod}_{CASL}(\Sigma)$ of Σ-models; this assigment extends to a model functor $\mathbf{Mod}_{CASL} : \mathbf{CASLsign}^{op} \to \mathbf{CAT}$ in the standard way. It is well known that amalgamation fails for this model functor; the simplest example is the following:

Example 1. Let Σ be the signature with sorts s and t (and no operations), and let Σ_1 be the extension of Σ by the subsort relation $s < t$. Then the pushout

$$\begin{tikzcd} \Sigma \arrow[r] \arrow[d] & \Sigma_1 \arrow[d] \\ \Sigma_1 \arrow[r] & \Sigma_1 \end{tikzcd}$$

in **CASLsign** fails to be amalgamable (since two models of Σ_1, compatible w.r.t. the inclusion of Σ, may interpret the subsort injection differently).

Incidentally, $\mathbf{Mod}_{CASL}$ also fails to be definitionally complete: e.g., the inclusion $\Sigma_1 \hookrightarrow \Sigma_2$, where Σ_1 contains sorts $s < t$ and an operation $f : s$ and Σ_2 contains an additional symbol $f : t$, induces an isomorphism on the level of models.

3 Enriched signatures

We now introduce a category of enriched signatures in which standard signatures can be represented via a suitable functor. Moreover, we equip this signature category with a model functor which has the amalgamation property and is definitionally complete and which extends the original model functor up to a natural isomorphism; this enables us to treat amalgamability in the extended setting.

Example 1 suggests that the failure of amalgamation for standard signatures can be remedied by replacing the sort preorder by a sort category that admits more than one embedding between two sorts (similar category sorted algebras, although without a view on amalgamation, appear in [23]). This will be the main feature to distiguish enriched and standard signatures. We will certainly continue to require embeddings to be monomorphic; categories in which all morphisms are monomorphisms will be called *left cancellable*.

Moreover, there is an elegant way of handling overloading of function and predicate symbols: introduce left and right actions of the sort category on the symbols; in the case of a unary function symbol $f : s \to t$ and sort embeddings $d : u \to s$, $e : t \to v$, these actions give rise to function symbols $f \cdot d : u \to t$ and $e \cdot f : s \to v$:

$$\begin{tikzcd} & t \arrow[r, "e"] & v \\ u \arrow[r, "d"'] \arrow[ur, phantom, "\nearrow f\cdot d"] & s \arrow[u, "f"] \arrow[ur, phantom, "\nearrow e\cdot f"] & \end{tikzcd}$$

(The right action also applies to predicate symbols.) The appropriate behaviour of models w.r.t. overloading can then be ensured by requiring that the diagrams that arise from the actions as above are translated to commutative diagrams of maps (operations) in the models. This requirement replaces the rather cumbersome overloading axioms for models needed in the case of standard signatures;

similarly, overloading preservation for signature morphisms now becomes a much more straightforward equivariance condition.

Thus, we arrive at the following

Definition 2. An *enriched (*CASL*) signature* Σ consists of

(i) a left cancellable *sort category* $\mathbf{S}$ with morphisms called *embeddings*;
(ii) a class F of *function symbols* and a class P of *precidate symbols*; symbols have profiles as in the case of standard signatures;
(iii) a unary *totality* predicate on F;
(iv) a *left action* of $\mathbf{S}$ on F which assigns to each function symbol $f : w \to s$ and each sort morphism $e : s \to t$ a function symbol $e \cdot f : w \to t$;
(v) a *right (multi)action* of $\mathbf{S}$ on F which assigns to each function symbol $f : s_1 \dots s_n \to t$ and each tuple of sort embeddings $d = (d_i : u_i \to s_i)_{i=1,\dots,n}$ a function symbol $f \cdot d : u_1 \dots u_n \to t$, and a similar right action on P.

These data are subject to the following axioms:

(i) The associative law and the identity law hold in the obvious sense (e.g., in the above notation, $id \cdot f = f$ and $(e \cdot f) \cdot d = e \cdot (f \cdot d)$).
(ii) For a sort embedding e and function symbols f, g with appropriate profiles, $e \cdot f = e \cdot g$ implies $f = g$.
(iii) Let f, e, and d be as above. If f is total, then $f \cdot d$ and $e \cdot f$ are total. Moreover, if $e \cdot f$ is total, then f is total.

A *signature morphism* σ between enriched signatures consists of a functor between the sort categories and a pair of maps between the classes of function and predicate symbols, respectively; all three components are denoted by σ. σ is required to be compatible with symbol profiles and to preserve totality. Moreover, σ is assumed to be *equivariant* w.r.t. the actions of the sort categories; i.e. if f, e, and d are as above, then

$$\sigma(e) \cdot \sigma(f) = \sigma(e \cdot f) \quad \text{and} \quad \sigma(f) \cdot \sigma(d) = \sigma(f \cdot d),$$

similarly for predicate symbols. An enriched signature is called *small* if its sort category and its symbol classes are small. Small enriched signatures and signature morphisms form a category **enrCASLsign**.

More precisely, one should say that **enrCASLsign** is a 2-category: 2-cells between signature morphisms are natural transformations between the functor parts that satisfy the obvious naturality condition w.r.t. symbols. In particular, one has a notion of equivalence of enriched signatures defined in the usual way via 'inverses up to isomorphism'. As in the case of categories, it is straightforward to show that such a definition of equivalence is equivalent to the following:

Definition 3. A signature morphism σ is called an *equivalence* if the functor part of σ is an equivalence, σ is, in the obvious sense, full and faithful on symbols, and whenever $\sigma(f)$ is a total function symbol, then so is f.

A crucial point is that the collection of all sets and partial maps can now be regarded as a signature. (Due to the rather different treatment of overloading, there is no obvious way to make this work for the signatures defined in [23]). More precisely: We have an enriched signature which has the category of sets and (total) injective maps as sort category and n-ary partial functions and relations as function and predicate symbols, respectively, with the obvious assignment of profiles and with totality of partial functions as the totality predicate. The actions of the sort category are given by composition (in the case of the right action, by composition with cartesian products of maps or, for predicates, by taking preimages under such products). This enriched signature will be denoted by $\mathbf{Set_p}$.

This enables us to define models in the spirit of Lawvere [19]:

Definition 4. A model of an enriched signature Σ with sort category $\mathbf{S}$ is a signature morphism

$$\Sigma \to \mathbf{Set_p}.$$

A *morphism* between two such models σ, τ is a natural transformation μ between σ and τ, regarded as functors $\mathbf{S} \to \mathbf{Set}$, such that the usual homomorphism condition w.r.t. function symbols holds and satisfaction of predicates is preserved.

Thus, we have a *model category* of Σ, which we denote by $\mathbf{Mod}_{enr}(\Sigma)$. A signature morphism $\sigma : \Sigma_1 \to \Sigma_2$ induces a functor $\mathbf{Mod}_{enr}(\sigma) : \mathbf{Mod}_{enr}(\Sigma_2) \to \mathbf{Mod}_{enr}(\Sigma_1)$ which acts on objects by composition of signature morphisms. This defines the model functor $\mathbf{Mod}_{enr} : \mathbf{enrCASLsign}^{op} \to \mathbf{CAT}$.

Now the representing functor $\mathsf{Enr} : \mathbf{CASLsign} \to \mathbf{enrCASLsign}$ acts on standard signatures by first forming a suitable completion of the symbol sets — closing symbol profiles under the sort preorder — to account for the actions of the sort embeddings, and then reinterpreting the data in the usual way (i.e. the sort preorder is interpreted as a thin category, and the actions of the sort category are defined using the mentioned closure of symbol profiles). Thanks to overloading preservation, morphisms between standard signatures have a well-defined extension to the respective closures. Thus, this assignment on objects extends to a functor as required. Note that Enr is faithful, but is neither full nor injective on objects and, of course, does *not* preserve colimits.

Now it is easily verified that one indeed has a natural isomorphism

$$\mathbf{Mod}_{enr} \circ \mathsf{Enr}^{op} \to \mathbf{Mod}_{CASL}$$

which maps an $\mathsf{Enr}(\Sigma)$-model to the restriction of the symbol interpretation to Σ. In particular, as indicated above,

Proposition 5. *A cocone in* $\mathbf{CASLsign}$ *is amalgamable w.r.t.* $\mathbf{Mod}_{CASL}$ *iff its image under* Enr *is amalgamable w.r.t.* $\mathbf{Mod}_{enr}$.

Finally,

Proposition 6. $\mathbf{enrCASLsign}$ *is cocomplete.*

4 Amalgamation and definitional completeness

In order to prove amalgamation and definitional completeness for enriched CASL, we introduce a further representation of enriched signatures in a suitable class of small categories which is easily seen to have the said properties; we then go on to show that the representation is sufficiently well-behaved to transfer these properties back to the setting of enriched signatures. In an intermediate step, we introduce a coding of enriched signatures as equational partial specifications in the sense of [9], thus essentially providing an extension of the existing embedding of the CASL logic into partial first-order logic with equality.

Let **Lex** denote the category of small left exact (lex) categories (i.e. small categories with finite limits) and left exact (lex, finite limit preserving) functors. The *model category* $\mathbf{Mod}_{lex}(\mathbf{A})$ of a small lex category $\mathbf{A}$ is the category $\mathrm{Lex}(\mathbf{A}, \mathbf{Set})$ of lex functors $\mathbf{A} \to \mathbf{Set}$ and natural transformations. By composition of lex functors, this assignment extends to a functor

$$\mathbf{Mod}_{lex} : \mathbf{Lex}^{op} \to \mathbf{CAT}.$$

$\mathbf{Mod}_{lex}$ trivially has the amalgamation property, i.e. preserves limits. Moreover, being a 2-functor, $\mathbf{Mod}_{lex}$ preserves equivalences. Conversely,

Theorem 7. *$\mathbf{Mod}_{lex}$ reflects equivalences.*

This is an immediate consequence of the following lemma, which is proved using the methods discussed in [6], Ch. 6:

Lemma 1. *Let $\mathbf{A}$ be a lex category. A lex functor $F : \mathbf{A} \to \mathbf{Set}$ is finitely presentable in $\mathbf{Mod}_{lex}(\mathbf{A})$ (i.e. $hom_{\mathbf{Mod}_{lex}(\mathbf{A})}(F, \_)$ preserves filtered (directed) colimits [2]) iff it is representable.*

(Thus, an equivalence between the lex functor categories induces an equivalence between the categories of representables, which are equivalent to the given lex categories via the Yoneda Embedding.)

Proof. Filtered colimits of lext exact functors are formed pointwise ([6], 6.2.2). Thus, representable functors are finitely presentable in $\mathbf{Mod}_{lex}(\mathbf{A})$, since, by the Yoneda Lemma, $hom_{\mathbf{Mod}_{lex}(\mathbf{A})}(hom_{\mathbf{A}}(A, \_), \_)$ is isomorphic to evaluation at A.

Conversely, a left exact functor $F : \mathbf{A} \to \mathbf{Set}$ is also flat, hence a filtered colimit of representable functors ([6], 6.3.2 and 6.3.7). Thus, the same argument as in [6], Lemma 6.5.10, shows that F is a retract of a representable functor. But since $\mathbf{A}$ is left exact, hence in particular Cauchy complete ([6], 6.5), the representables are closed under retracts in $\mathbf{Mod}_{lex}(\mathbf{A})$.

An (equational partial) *specification* $\mathcal{S} = (\Omega, \mathcal{A})$ is defined (slightly extending the definition in [9]) as follows:

Ω is a *signature* consisting of sorts, including a *terminal* sort T, and (partial) operation symbols with profiles as before, including a distinguished operation i : T. This signature gives rise to a notion of term (with indicated profile) in the usual way. For convenience, tuples of terms are regarded as terms; so if $\alpha_i : w \to s_i$ are terms for $i = 1, \dots, n$, then we write $\alpha = (\alpha_i) : w \to v$, where $v = (s_1, \dots, s_n)$. x_v denotes 'the' term $(x_1, \dots, x_n)$, where x_i is a variable of sort s_i, $i = 1, \dots, n$. If β is a term in the x_i, then $\beta\alpha$ denotes the term obtained by simultaneously substituting the α_i for the x_i.

$\mathcal{A}$ is a set of *axioms* that take the form of finite implications: an *existential equation* is a pair $\phi = (\alpha_1, \alpha_2)$ of terms (with identical profiles), written $\alpha_1 \stackrel{e}{=} \alpha_2$, and a *finite implication* is an implication of the form $\phi \Rightarrow \psi$, where ϕ and ψ are existential equations. By virtue of the tuple notation, the definition of existential equation covers finite conjunctions of such equations; we use notations like $\phi \wedge \psi$ with the obvious meaning. For $\phi = (\alpha_1, \alpha_2)$, $\phi\beta$ denotes $\alpha_1\beta \stackrel{e}{=} \alpha_2\beta \wedge \beta \stackrel{e}{=} \beta$. Equations $\alpha \stackrel{e}{=} \alpha$ are sometimes abbreviated as $\operatorname{def} \alpha$.

In Figure 1, we present the rules of a deduction system for existential equality associated to a specification $\mathcal{S} = (\Omega, \mathcal{A})$. This system is obviously sound w.r.t. the notion of model defined below; it will turn out to be complete as well. We write $\Phi \vdash \phi$ if an existential equation ϕ can be deduced from a set Φ of existential equations by means of these rules. If $\phi \vdash \psi$ for existential equations ϕ, ψ, we say that $\phi \Rightarrow \psi$ is a *theorem*.

$$
\frac{s \text{ is a sort}}{x_s \stackrel{e}{=} x_s} \qquad
\frac{}{x_{\mathrm{T}} \stackrel{e}{=} i} \qquad
\frac{\alpha \stackrel{e}{=} \beta}{\beta \stackrel{e}{=} \alpha} \qquad
\frac{\alpha \stackrel{e}{=} \beta,\ \beta \stackrel{e}{=} \gamma}{\alpha \stackrel{e}{=} \gamma} \qquad
\frac{\phi, \psi}{\phi \wedge \psi} \qquad
\frac{\phi \wedge \psi}{\phi} \qquad
\frac{\phi \wedge \psi}{\psi}
$$

$$
\frac{\begin{array}{c}\alpha \stackrel{e}{=} \alpha \\ \beta \text{ is a subterm of } \alpha\end{array}}{\beta \stackrel{e}{=} \beta} \qquad
\frac{\begin{array}{c}\phi \Rightarrow \psi \in \mathcal{A} \\ \phi\alpha\end{array}}{\psi\alpha} \qquad
\frac{\begin{array}{c}\alpha_1 \stackrel{e}{=} \alpha_2, \beta_1 \stackrel{e}{=} \beta_2 \\ \alpha_i\beta_i \text{ is well-formed, } i = 1, 2\end{array}}{\alpha_1\beta_1 \stackrel{e}{=} \alpha_2\beta_2} \ (*)
$$

Fig. 1. Deduction rules for existential equality

A morphism between specifications is defined as a theory morphism in the usual sense, i.e. a signature morphism which preserves the terminal sort (and the distinguished operation) and transforms axioms into theorems. Thus, specifications form a category **epSpec**.

A *model* of the signature Ω is an algebra that interprets all symbols as partial operations in the usual way; the terminal sort is always interpreted as a singleton set. An existential equation is satisfied in a model iff both terms are defined and equal for all interpretations of the variables; the satisfaction of a finite implication is defined in the obvious way. A model of $\mathcal{S}$ is a model of Ω that satisfies all finite implications in $\mathcal{A}$; it is clear how this leads to a model functor $\mathbf{Mod}_{eps} : \mathbf{epSpec}^{op} \to \mathbf{CAT}$.

Using the deduction system of Figure 1, we can now construct a lex category $\mathsf{Th}_{eps}(\mathcal{S})$ from $\mathcal{S}$: the objects of $\mathsf{Th}_{eps}(\mathcal{S})$ are pairs (w, ϕ), where w is a string of sorts and ϕ is an existential equation in the variables of x_w. Morphisms $(w, \phi) \to$

(v,ψ) are terms $\alpha : w \to v$ such that $\phi \vdash \psi\alpha$, taken modulo existential equality deducible from ϕ. The identity on (w,ϕ) is represented by x_w. Composition is defined via simultaneous substitution of representing terms. This is a well-defined operation thanks to the congruence rule $(*)$ in Figure 1 and the following meta-theorem (proved by induction over the proof length), which also shows that the composite is indeed a morphism:

Proposition 8. *If ϕ and ψ are existential equations in variables x_i of sort s_i and α is a tuple of terms α_i of sort s_i, $i = 1,\dots,n$, then $\phi \vdash \psi$ implies $\phi\alpha \vdash \psi\alpha$.*

As expected, 'concatenation' of objects defines finite products, and $x_w : (w, \phi \wedge \alpha_1 \overset{e}{=} \alpha_2) \to (w,\phi)$ is an equalizer of $\alpha_1, \alpha_2 : (w,\phi) \to (v,\psi)$ in $\mathsf{Th}_{eps}(\mathcal{S})$. Thus, $\mathsf{Th}_{eps}(\mathcal{S})$ is indeed a lex category. Similarly as in [9], one easily verifies that one has an equivalence of categories $\mathbf{Mod}_{lex}(\mathsf{Th}_{eps}(\mathcal{S})) \to \mathbf{Mod}_{eps}(\mathcal{S})$ (natural in $\mathcal{S}$) and concludes, using the fact that representable functors are models of $\mathsf{Th}_{eps}(\mathcal{S})$, that the deduction system of Figure 1 is complete.

It is straightforward to translate an enriched CASL signature Σ into an equational partial specification $\mathsf{Spec}(\Sigma) = (\Omega, \mathcal{A})$: the sorts in Ω are the sorts of Σ and a new terminal sort T; the symbols in Ω are the function and predicate symbols of Σ (where the range of all predicate symbols is T), the embedding symbols with the obvious unary profiles, and a new symbol $i : \mathrm{T}$. The axioms are given in Figure 2. It is easily verified that $\mathbf{Mod}_{eps} \circ \mathsf{Spec}^{op}$ is naturally isomorphic to $\mathbf{Mod}_{enr}$.

Remark 9. The translation functor Spec can be modified to retain predicates and include an elementhood predicate and partial projection functions for subsorts, thus providing an extension to enriched CASL of the existing embedding of the CASL logic $SubPFOL^=$ (subsorted partial first-order logic with equality) into $PFOL^=$ (partial first-order logic with equality) [8].

Putting the two translations together, we have a representation $\mathsf{Th}_{enr} := \mathsf{Th}_{eps} \circ \mathsf{Spec}$ of enriched signatures as lex categories. Moreover, we have a natural 'forgetful functor' $\mathsf{U} : \mathbf{Lex} \to \mathbf{enrCASLsign}$: given a lex category $\mathbf{A}$, the sorts of $\mathsf{U}(\mathbf{A})$ are the objects of $\mathbf{A}$ and the function symbols are the partial morphisms (spans containing a monomorphism, taken modulo isomorphy) with profiles determined by the finite product structure. Such a symbol is total iff the associated partial morphism is total, i.e. iff the associated monomorphism is an isomorphism. Total monomorphisms double as embeddings and partial function symbols with terminal codomain (taken modulo isomorphy of terminal objects) as predicate symbols.

Theorem 10. *Th_{enr} is essentially a left adjoint for U.*

Proof. We have to show that, given a signature morphism $\sigma : \Sigma \to \mathsf{U}(\mathbf{A})$, there exists a lex functor $\sigma^\# : \mathsf{Th}_{enr}(\Sigma) \to \mathbf{A}$ such that $\mathsf{U}(\bar{\sigma})\eta = \sigma$, where η denotes the obvious map $\Sigma \to \mathsf{U}(\mathsf{Th}_{enr}(\Sigma))$. It is clear that such a functor is unique up to a natural isomorphism.

$$\left.\begin{array}{l}\operatorname{def}(e\cdot f)(x_w)\Rightarrow (e\cdot f)(x_w)\overset{e}{=}e(f(x_w))\\ \operatorname{def}e(f(x_w))\Rightarrow (e\cdot f)(x_w)\overset{e}{=}e(f(x_w))\end{array}\right\}\begin{array}{l}\text{if } f:w\to s \text{ is a function symbol}\\ \text{and } e:s\to t \text{ is an embedding;}\end{array}$$

$$\left.\begin{array}{l}\operatorname{def}(f\cdot d)(x_v)\Rightarrow (f\cdot d)(x_v)\overset{e}{=}f(d(x_v))\\ \operatorname{def}f(d(x_v))\Rightarrow (f\cdot d)(x_v)\overset{e}{=}f(d(x_v))\end{array}\right\}\begin{array}{l}\text{if } f \text{ is a function or predicate symbol}\\ \text{and } d:v\to w \text{ is a tuple of embeddings;}\end{array}$$

$$e(d(x_s))\overset{e}{=}(e\circ d)(x_s),\quad \text{if } d:s\to t \text{ and } e:t\to u \text{ are embeddings;}$$

$$id_s(x_s)=x_s,\quad \text{if } s \text{ is a sort;}$$

$$\operatorname{def}f(x_w),\quad \text{if } f:w\to s \text{ is a total function symbol;}$$

$$e(x_s)\overset{e}{=}e(y_s)\Rightarrow x_s\overset{e}{=}y_s,\quad \text{if } e:s\to t \text{ is an embedding.}$$

Fig. 2. Axioms associated to an enriched CASL signature

σ gives rise, in the usual way, to a 'model' $[\![\_]\!]$ of Σ in $\mathbf{A}$: $[\![\_]\!]$ is defined inductively first for sort strings, then for terms $\alpha : w \to v$ as partial morphisms $[\![w]\!] \hookleftarrow \operatorname{def}[\![\alpha]\!] \to [\![v]\!]$ (where pairing and composition are defined via finite limits). For existence equations $\phi \equiv \alpha \overset{e}{=} \beta$ with profile w, $[\![(w,\phi)]\!]$ is defined as a limit of the double span

$$\begin{tikzcd}
\operatorname{def}[\![\alpha]\!] \arrow[r] \arrow[d, hook] & {[\![v]\!]} \\
{[\![w]\!]} & \operatorname{def}[\![\beta]\!]. \arrow[l, hook'] \arrow[u]
\end{tikzcd}$$

Note that this limit is a subobject of $[\![w]\!]$. One then proves a soundness theorem which states that, whenever $\phi \vdash \psi$, then there is an inclusion between the subobjects $[\![(w,\phi)]\!]$ and $[\![(w,\psi)]\!]$ of $[\![w]\!]$. Moreover, using the limit property it is easily shown that each term $\alpha : w \to v$ gives rise to a morphism $[\![\alpha]\!] : [\![(w, \psi\circ\alpha)]\!] \to [\![(v,\psi)]\!]$ which is a restriction of $[\![\alpha]\!] : [\![\operatorname{def}\alpha]\!] \to [\![v]\!]$.

Now if $\phi \vdash \psi\circ\alpha$, then we can define $[\![\alpha]\!] : [\![(w,\phi)]\!] \to [\![(v,\psi)]\!]$ as the composite

$$[\![(w,\phi)]\!] \hookrightarrow [\![(w,\psi\circ\alpha)]\!] \to [\![(v,\psi)]\!];$$

this is consistent with all previous definitions.

$[\![\_]\!]$ defines a functor $\mathsf{Th}_{enr}(\Sigma) \to \mathbf{A}$, since the soundness theorem implies that terms which represent the same morphism in $\mathsf{Th}_{enr}(\Sigma)$ are interpreted as the same morphism in $\mathbf{A}$. It remains to be shown that this functor is lex. Preservation of equalizers is easily seen by chasing a suitable large diagram (which can be made a bit smaller by the observation that it suffices to consider equalizers of pairs of morphisms into objects of the form $(v, ()))$. Preservation of finite products follows from the fact that finite products commute with the limits that define the $[\![(w, \phi)]\!]$ (using a lemma which states that $\mathrm{def}[\![\alpha \times \beta]\!] = \mathrm{def}[\![\alpha]\!] \times \mathrm{def}[\![\beta]\!]$).

Corollary 11. Th_{enr} *preserves colimits.*

This implies

Theorem 12. $\mathbf{Mod}_{enr}$ *has the amalgamation property.*

Proof. By virtue of the facts collected above, a limit in $\mathbf{enrCASLsign}^{op}$ is mapped to a cone in **CAT** that factors through the limit in **CAT** via an equivalence. Since the objects of model categories are given by a representable functor (namely, the functor $hom(\_, \mathbf{Set_p})$), and since representable functors preserve limits, this equivalence is bijective on objects, hence an isomorphism.

(This is, of course, rather easier proved directly. However, there is an independent interest in Theorem 10.) Moreover, the representation is, in a certain sense, 'faithful':

Theorem 13. Th_{enr} *reflects equivalences.*

The proof relies on two statements concerning deduction in $\mathsf{Spec}(\Sigma) = \mathcal{S}$:

Definition 14. A term α in $\mathcal{S}$ is said to *reduce* to an embedding (tuple of embeddings) d in Σ if all symbols in α are embeddings, and their composite is d. In this case we write $\alpha \twoheadrightarrow d$. Reduction to a function or predicate symbol f in Σ ($\alpha \twoheadrightarrow f$) is defined analogously, using also the left and right actions of the sort category. if α is of the form $\beta g \gamma$, where g is a function symbol, β reduces to an embedding e, γ reduces to a tuple d of embeddings, and $e \cdot g \cdot d = f$.

Lemma 2. *Let α and β be terms, and let $h : w \to s$ be a symbol in $\mathcal{S}$ such that* $\mathrm{def}\,\alpha \vdash \beta \stackrel{e}{=} h(x_w)$*. Then $\beta \twoheadrightarrow h$.*

Proof. Call an existence equation $\alpha_1 \stackrel{e}{=} \alpha_2$ *reductively equivalent* if, for each $a \in \Omega$, $\alpha_1 \twoheadrightarrow a$ is equivalent to $\alpha_2 \twoheadrightarrow a$. The set of reductively equivalent equations is closed under application of the deduction rules of Figure 1 (with the axioms given in Figure 2). Thus, since $\mathrm{def}\,\alpha$ is reductively equivalent, so is $\beta \stackrel{e}{=} h(x_w)$; this implies the claim.

Lemma 3. *If* $\vdash \mathrm{def}\, f(x_w)$ *in $\mathcal{S}$ for a function symbol $f : w \to s$ in Σ, then f is total.*

Proof. Call an existence equation ϕ *leaf-total* if, whenever a term of the form $g\beta$, where g is a function symbol and β reduces to a tuple of embeddings d, occurs as a subterm in ϕ, then $g \cdot d$ is a total symbol. The set of leaf-total equations is closed under deduction. Thus, $\mathrm{def}\, f(x_w)$ as in the statement is leaf-total, which implies the claim.

In a nutshell, this means that the only terms that are provably equal to symbols are the obvious candidates, and that only total function symbols are provably total (note that, even for standard CASL signatures, there are provably total *terms* that cannot be expressed using only total function symbols!).

Proof. (Theorem 13)

Let $\sigma : \Sigma_1 \to \Sigma_2$ be a morphism of enriched signatures such that $\mathsf{Th}(\sigma)$ is an equivalence.

σ is isomorphism-dense: Let s be a sort in Σ_2. Then there exists an isomorphism $\alpha : (s, ()) \to \mathsf{Th}(\sigma)(B)$ for some object B of $\mathsf{Th}(\Sigma_1)$. Lemma 2 implies that both α and its inverse reduce to embeddings, so that s is isomorphic to an object of the form $\sigma(t)$ in Σ_2.

σ is faithful: Let $f, g : w \to s$ be function symbols in Σ_1 (predicate symbols and embeddings are treated analogously) such that $\sigma(f) = \sigma(g)$. Then by fullness of $\mathsf{Th}(\sigma)$, $x_w : (w, \mathrm{def}\, f(x_w) \wedge \mathrm{def}\, g(x_w)) \to (w, \mathrm{def}\, g(x_w))$ is an isomorphism, the inverse of which must be x_w by Lemma 2; correspondingly for f. Thus, we have morphisms $f(x_w), g(x_w) : (w, \mathrm{def}\, f(x_w)) \to (s, ())$ in $\mathsf{Th}(\Sigma_1)$. By faithfullness of $\mathsf{Th}(\sigma)$, these morphisms are equal, i.e. $\mathrm{def}\, f(x_w) \vdash f(x_w) \stackrel{e}{=} g(x_w)$. This implies $f = g$ by Lemma 2.

σ is full on embeddings: Let $e : \sigma(s) \to \sigma(t)$ be an embedding in Σ_2. By fullness of $\mathsf{Th}(\sigma)$, there exists a morphism $\alpha : (s, ()) \to (t, ())$ in $\mathsf{Th}(\Sigma_1)$ such that $\mathsf{Th}(\sigma)(\alpha) = e$, i.e. $\vdash \sigma(\alpha) = e(x_s)$. By Lemma 2, $\sigma(\alpha) \twoheadrightarrow e$; hence, $\alpha \twoheadrightarrow d$ for some embedding $d : s \to t$, and $\sigma(d) = e$.

σ is full on symbols: Let $f : \sigma(w) \to \sigma(s)$ be a function symbol. By isomorphism-density of $\mathsf{Th}\,\sigma$, we have an isomorphism $(w, \mathrm{def}\, f(x_w)) \to \mathsf{Th}\,\sigma(v, \phi)$ for some (v, ϕ) in $\mathsf{Th}\,\Sigma_1$. By Lemma 2, this isomorphism and its inverse must be tuples of embeddings, arising, by fullness of σ on embeddings, as the image under σ of a tuple $d : w \to v$ of sort isomorphisms and its inverse.

Now the morphism $f \cdot \sigma(d)^{-1}(x_{\sigma(v)}) : \mathsf{Th}\,\sigma(v, \phi) \to (\sigma(s), ())$ has a preimage $\alpha : (v, \phi) \to (s, ())$ under $\mathsf{Th}(\sigma)$. Since $\sigma(d) : (w, \mathrm{def}\, f(x_w)) \to \mathsf{Th}\,\sigma(v, \phi)$ is an isomorphism, $\mathrm{def}\, f(x_w) \vdash f(x_w) \stackrel{e}{=} \sigma(\alpha)\sigma(d)$, so that $\sigma(\alpha d) \twoheadrightarrow f$ by Lemma 2. This implies $\alpha d \twoheadrightarrow g$ for some symbol g, and $\sigma(g) = f$.

σ reflects totality: If $f : w \to s$ is a function symbol such that $\sigma(f)$ is total, then by fullness of $\mathsf{Th}\,\sigma$, $x_w : (w, \mathrm{def}\, f(x_w)) \to (w, ())$ is an isomorphism. This implies $\vdash \mathrm{def}\, f(x_w)$. By Lemma 3, f is total.

By Theorems 7 and 13, $\mathbf{Mod}_{enr}$ reflects equivalences. Since it is trivial to show that a signature morphism σ is bijective on sorts if $\mathbf{Mod}_{enr}(\sigma)$ is an isomorphism, this implies

Corollary 15. $\mathbf{Mod}_{enr}$ *is definitionally complete.*

In particular, we now have a necessary and sufficient criterion for amalgamability in **CASLsign**:

Corollary 16. *A cocone in* **CASLsign** *is amalgamable iff its image under* Enr *is a colimit in* **enrCASLsign**.

5 Conclusions and future work

We have constructed a representation of the CASL institution (the extension to sentences, disregarded here, is straightforward) in enriched CASL, an institution with a category of enriched signatures that has the amalgamation property. This makes a number of results (e.g. concerning normal forms and proof systems) about institution independent specification languages applicable to CASL.

In more detail, we have modified Reynolds' approach [23] to subsorting via sort categories by using actions of the sort category on function and predicate symbols. In this way we elegantly deal with the problems of both overloading and amalgamation. Moreover, the associated logic admits a reduction to partial conditional equational logic. For the latter, we provide a sound and complete proof system, extending and simplifying the work of [9].

Typically, the use of enriched CASL will be as follows. Specifications are written in ordinary CASL. In situations where the user inputs a certain combination (colimit) of signatures (e.g. when writing an instantiation of a parameterized specification), the natural requirement will be to check whether this combination becomes a colimit also in the category of enriched CASL signatures, thus guaranteeing amalgamability of models. At this stage, enriched CASL remains completely hidden from the user. In contrast to this, there are also situations where a combination of signatures is automatically produced by a tool (e.g. during a proof in a development graph or during static analysis of architectural specifications). In these situations, it is advisable to use colimits that result in properly enriched signatures as intermediate results rather than restrict to those colimits in the enriched signature category that happen to be CASL signatures. Theorem proving in the enriched CASL logic is eased by the fact that this logic can be embedded into partial first-order logic with equality in much the same way as the CASL logic.

Moreover, we have shown that enriched CASL is definitionally complete, which means that we can expect amalgamation only for combinations of signatures that are actually colimits in the enriched signature category. Thanks to this characterization of a model-theoretic condition by a syntactic condition concerning signature combination, the model-theoretic and the static architectural semantics of [26] essentially coincide. Algorithms related to the actual computation of colimits of enriched signatures (a prerequisite for the development of tools for architectural specifications and proof support) are discussed in the forthcoming [18].

Future lines of research include the generalization of the techniques developed in this work to arbitrary institutions, resulting in particular in a generic procedure for 'making institutions amalgamable'. It appears that institutions where finite amalgamation fails fall into two classes — one where preservation of pushouts fails, and one where preservation of initial objects (and coproducts) fails. The institutions with unnamed universes mentioned in the introduction belong to the latter class. It is plausible to assume that these institutions can — in analogy to the extension of unsorted to multisorted logic — be made amalgamable by introducing 'multiple universes'.

Moreover, we conjecture that not only the amalgamation property, but also the Craig interpolation property (in the weakened form valid in multisorted algebra), which fails in standard CASL, holds in enriched CASL. This property, which seems to be related to amalgamation in some way, is required by the institution independent proof calculus for ASL-like specifications [7]. Finally, the representation of enriched signatures as left exact categories (which form a cartesian closed category) may provide the proper framework for a semantics of CASL architectural terms with bound variables [11] in the spirit of [26].

Acknowledgements

Partial support by the CoFI Working Group (ESPRIT WG 29432) is gratefully acknowledged. Moreover, we wish to thank Bartek Klin and Piotr Hoffman for collaboration and Maura Cerioli for pointing out the lack of amalgamation for CASL models.

References

[1] J. Adámek, H. Herrlich, and G. E. Strecker, *Abstract and concrete categories*, 2nd ed., Wiley Interscience, New York, 1990.

[2] J. Adamek and J. Rosicky, *Locally presentable and accessible categories*, Cambridge, 1994.

[3] D. Aspinall, *Types, subtypes, and ASL+*, Recent Trends in Data Type Specification, LNCS, vol. 906, Springer, 1994, pp. 116–131.

[4] H. Baumeister, *Relations between abstract datatypes modeled as abstract datatypes*, Ph.D. thesis, Universität des Saarlandes, 1998.

[5] J.A. Bergstra, J. Heering, and P. Klint, *Module algebra*, J. ACM **37** (1990), no. 2, 335–372.

[6] F. Borceux, *Handbook of categorical algebra 1*, Cambridge, 1994.

[7] T. Borzyszkowski, *Logical systems for structured specifications*, Theoret. Comput. Sci., to appear.

[8] M. Cerioli, A. Haxthausen, B. Krieg-Brückner, and T. Mossakowski, *Permissive subsorted partial logic in* CASL, Algebraic Methodology and Software Technology (AMAST 97), LNCS, vol. 1349, Springer, 1997, pp. 91–107.

[9] I. Claßen, M. Große-Rhode, and U. Wolter, *Categorical concepts for parameterized partial specifications*, Math. Struct. Comput. Sci. **5** (1995), 153–188.

[10] CoFI, *The Common Framework Initiative for algebraic specification and development, electronic archives*, accessible by WWW[2] and FTP[3].

[11] CoFI Language Design Task Group, CASL *Summary, version 1.0*, Documents/CASL/Summary, in [10], July 1999.

[12] R. Diaconescu, J. Goguen, and P. Stefaneas, *Logical support for modularisation*, Logical Environments, Cambridge, 1993, pp. 83–130.

[13] H. Ehrig and M. Große-Rhode, *Functorial theory of parameterized specifications in a general specification framework*, Theoret. Comput. Sci. **135** (1994), 221–266.

[14] H. Ehrig and B. Mahr, *Fundamentals of algebraic specification 2*, Springer, 1990.

[2] http://www.brics.dk/Projects/CoFI
[3] ftp://ftp.brics.dk/Projects/CoFI

[15] J. Goguen and R. Burstall, *Institutions: Abstract model theory for specification and programming*, J. ACM **39** (1992), 95–146.

[16] R. Harper, F. Honsell, and G. D. Plotkin, *A framework for defining logics*, J. ACM **40** (1993), 143–184.

[17] B. Klin, *An implementation of static semantics for architectural specifications in* CASL *(in Polish)*, Master's thesis, Warsaw University, 2000.

[18] B. Klin, P. Hoffman, A. Tarlecki, T. Mossakowski, and L. Schröder, *Checking amalgamability conditions for* CASL *architectural specifications*, work in progress; see also [17].

[19] F. W. Lawvere, *Functorial semantics of algebraic theories*, Proc. Natl. Acad. Sci. USA **50** (1963), 869–872.

[20] S. Mac Lane, *Categories for the working mathematician*, Springer, 1997.

[21] T. Mossakowski, S. Autexier, and D. Hutter, *Extending development graphs with hiding*, Fundamental Approaches to Software Engineering, LNCS, Springer, 2001, to appear.

[22] P. D. Mosses, CASL*: A guided tour of its design*, 13th Workshop on Abstract Datatypes WADT 98, LNCS, vol. 1589, Springer, 1999, pp. 216–240.

[23] J. C. Reynolds, *Using category theory to design implicit conversions and generic operators*, Semantics-Directed Compiler Generation, LNCS, vol. 94, Springer, 1980, pp. 211–258.

[24] D. Sannella, S. Sokołowski, and A. Tarlecki, *Towards formal development of programs from algebraic specifications: Parameterisation revisited*, Acta Inform. **29** (1992), 689–736.

[25] D. Sannella and A. Tarlecki, *Specifications in an arbitrary institution*, Inform. and Comput. **76** (1988), 165–210.

[26] L. Schröder, T. Mossakowski, A. Tarlecki, P. Hoffman, and B. Klin, *Semantics of architectural specifications in* CASL, Fundamental Approaches to Software Engineering (FASE2001), LNCS, Springer, 2001, to appear.