Equational Logic and Categorical Semantics for Multi-Languages

Samuele Buro[a] and Roy Crole[b] and Isabella Mastroeni[a][1,2,3]

[a] *University of Verona, Italy and* [b] *University of Leicester, UK*

Abstract

Programming language interoperability is the capability of two programming languages to interact as parts of a single system. Each language may be optimized for specific tasks, and a programmer can take advantage of this. HTML, CSS, and JavaScript yield a form of interoperability, working in conjunction to render webpages. Some object oriented languages have interoperability via a virtual machine host (.NET CLI compliant languages in the Common Language Runtime, and JVM compliant languages in the Java Virtual Machine). A high-level language can interact with a lower level one (Apple's Swift and Objective-C). While there has been some research exploring the interoperability mechanisms (Section 1) there is little development of theoretical foundations. This paper presents an approach to interoperability based around theories of equational logic, and categorical semantics.

We give ways in which two languages can be blended, and interoperability reasoned about using equations over the blended language. Formally, *multi-language equational logic* is defined within which one may deduce valid equations starting from a collection of axioms that postulate properties of the combined language. Thus we have the notion of a *multi-language theory* and much of the paper is devoted to exploring the properties of these theories. This is accomplished by way of *category theory*, giving us a very general and flexible semantics, and hence a nice collection of models. Classifying categories are constructed, and hence equational theories furnish each categorical model with an internal language; from this we can also establish soundness and completeness. A set-theoretic semantics follows as an instance, itself sound and complete. The categorical semantics is based on some pre-existing research, but we give a presentation that we feel is easier and simpler to work with, improves and mildly extends current research, and in particular is well suited to computer scientists. Throughout the paper we prove some interesting properties of the new semantic machinery. We provide a small running example throughout the paper to illustrate our ideas, and a more complex example in conclusion.

Keywords: categorical logic, equational logic, interoperability, multi-languages, order-sorted signatures and theories, programming languages, subsort polymorphism.

1 Introduction

The theory of equational algebra has been a compelling topic since the early days of *universal algebra* [33,2]. Research on *equational logic*, addressing the problem of *reasoning by deduction* about term equality, has been prolific (see [34,17] for surveys). In particular, many *sound* and *complete* deduction systems have arisen. For instance, if Sg is a *one-sorted*, *many-sorted*, or *order-sorted* signature (sorts and function symbols), such systems appear in [2], [10], and [13], respectively. These developments have had a remarkable impact on *operational semantics* and *automatic theorem proving*. In particular, the pioneering works of [8,18] to operationalize equational deduction led to the theory of *term rewriting systems* [15,35] which has extensive applications.

Multi-languages are programming languages arising from the combination of already existing languages [27,1,32,9,14,26,24,21]. Intuitively, terms of multi-languages are obtained by performing cross-language substitutions. For instance, the multi-language designed in [21] allows programmers to replace ML expressions with Scheme expressions and vice versa. Potential benefits are code reuse and software interoperability. In order to provide a semantics, [21] introduces new constructs to regulate the flow of values between the underlying languages, the so-called *boundary functions*.

[1] Email: `samuele.buro@univr.it`

[2] Email: `rlc3@le.ac.uk`

[3] Email: `isabella.mastroeni@univr.it`

But what are the formal properties and semantics of multi-languages? Indeed, how general can we be? And in any case, what is a good formal definition of a multi-language in the first place? Buro and Mastroeni [3] extended the approach of [21] to the broader class of *order-sorted algebras*, providing a systematic and more general way to define multi-languages, but they did not address equational reasoning. We do so here. In more detail, a multi-language is specified by combining two order-sorted signatures Sg_1 and Sg_2. But how exactly should the signatures be combined? And what is the formal equational theory of "term interoperability"? More precisely, since equations will not only be defined between two terms both of which are over just *one* of the Sg_i, but also defined between terms containing symbols from both Sg_1 *and* Sg_2, what is a good deduction system for such multi-language equations? We tackle these questions by specifying *multi-language equational logic* which is shown to be sound and complete.

Contributions in Detail: Conceptually we lift the basic syntactic theories of *order-sorted equational logic* [13], and models of the theories, to the algebraic multi-language framework defined by [3]. The models in [13] are built from sets, but we adapt the categorical approach in [20]. *The main contribution is a deductive system for multi-languages with a* sound *and* complete *categorical semantics. We also prove some interesting semantic properties.* There is a running example application throughout the paper, and an outline of a more elaborate application in Section 3.3 where we combine an imperative language and a lambda calculus. Our account of order-sorted equational theories builds on and refines [20], with all our deductive systems presented with uniform and clear inductive rules. Further, we include explicit type information in equality judgements, and include axioms that may be conditional equations. We give a simplified categorical semantics along with categorical type-theory correspondence and classifying category, and also give an explicit connection to free set-algebra semantics.

Remarks and Intuitions: We work with *order-sorted signatures* [13]. As such, our languages enjoy *subsort* polymorphism; there is no provision for *parametric* polymorphism. Such signatures may satisfy criteria known as monotonicity and regularity. Since the intuitions of these criteria are usually omitted from technical papers, we make some remarks. Language terms t are built inductively by applying function symbols f to existing terms (which begin as constants). Such symbols f can be polymorphically sorted. As such, if one input sort of f is a subsort of another input sort, we would like the term t to be subsort polymorphic with respect to the output sorts. *Monotonicity* formalises this requirement. One needs to place some control over such polymorphism: one way of doing so is to apply a requirement whereby every term t has a least sort. *Regularity* [13] is such a condition. To ensure every term t to has a least sort, we might naively achieve this by requiring each subsort-polymorphic function symbol f to have a least input and least output sort, hoping that t would do so inductively. We would need to give all constants a least sort to start the induction; and we could simply stipulate that each constant has just one sort. In initially building terms, if there were any function symbol f with a (polymorphic) input sort that is a strict subsort of the sort of a constant, such f could not be applied to that constant. We are led to refine our naive idea: Fix any f. Consider *only those constants that this f can be polymorphically applied to.* Now fix one such constant of sort s; this imposes a lower bound for the input sort of f, namely s. We then require that for all such polymorphic instances of f, there is one instance with a least input sort s, where of course $\mathsf{s} \leq s$, and a least output sort. This requirement, formalised, is *Regularity.*

Paper Structure: In Section 2 we present a transparent rule based deduction system for order-sorted equational logic with conditional axioms, together with a categorical semantics which is proved sound and complete. In Section 3 we present a similar set of results for multi-languages. In Section 4 we give some further examples of multi-languages. In Section 3.3 we present an outline of an extended example in which a traditional lambda calculus and an imperative while language are blended as a single multi-language.

2 Order-Sorted Equational Logic

We review order-sorted equational theories (see for example [13,20]). Here we give an improved presentation that is syntactically simpler than in loc cit, and further we extend theories to include conditional equational axioms. We also present a detailed but stylistically improved summary of the categorical models from [20], along with a simpler construction of the classifying category (up to equivalence). We then prove a result relating the classifying category to free order-sorted algebras.

2.1 Order-Sorted Algebras

A set S is usually regarded as a **set of sorts** or **set of ground types**. Often S is partially ordered by $\leq$, and then $S^n \triangleq S \times \cdots \times S$ (n-times cartesian product for $n \geq 1$) inherits the pointwise order, with typical instances written $w \leq w'$. If $w \in S^n$, we usually make explicit its components by writing $w \triangleq s_1, \ldots, s_n$, sometimes referring to a **sequence** of sorts.

We write $(A_s \mid s \in S)$ for a **family indexed by** S where each A_s is sometimes a set, but more generally an object in a category $\mathcal{C}$. We sometimes refer to the family as an S**-sorted set** (or, an S**-sorted object**). Such indexed families are simply functors A in the presheaf category $\mathcal{S}et^S$ ($\mathcal{C}^S$). As such, an S**-sorted function**

$$\frac{-}{\Gamma, x\colon s, \Gamma' \vdash x\colon s} \qquad \frac{-}{\Gamma \vdash k\colon s}\ (k\colon s) \qquad (\mathsf{Fn})\ \frac{(\forall 1 \leq i \leq a)\ \Gamma \vdash t_i\colon s_i}{\Gamma \vdash f(t_1,\dots,t_a)\colon s}\ (f\colon s_1,\dots,s_a \to s) \qquad \frac{\Gamma \vdash t\colon s}{\Gamma \vdash t\colon r}\ (s \leq r)$$

Fig. 1. Proved terms generated by an order-sorted signature Sg.

(**S-sorted morphism**) $h\colon A \to B$ is simply a morphism (that is, natural transformation) in $\mathcal{S}et^S$ ($\mathcal{C}^S$) where S is a set or poset.

In this paper all categories have finite products, and functors preserve them up to *isomorphism*. If A, B, and A_i ($1 \leq i \leq n$) are objects in a category $\mathcal{C}$, we write $A \times B$ for the binary product of A and B, $A_1 \times \cdots \times A_n$ or $\prod_1^n A_i$ for the finite product of the A_i, and 1 for the terminal object. Mediating morphisms for binary product are written $\langle f, f' \rangle$, and as usual $f \times f' \triangleq \langle f \circ \pi, f' \circ \pi' \rangle$ (for suitable f and f' and the usual projections). We adopt the obvious extension of notation for finite products. If A is an S-sorted object and $w \triangleq s_1,\dots,s_n$, we denote by A_w the product $A_{s_1} \times \cdots \times A_{s_n}$. Likewise, if f is an S-sorted morphism, then the morphism f_w is defined by $f_{s_1} \times \cdots \times f_{s_n}$. The coproduct object of A and B is written $A+B$. We write $l_1 \dots l_n$ or $l_1,\dots,l_n$ for a typical finite list, and we may abbreviate just to $\boldsymbol{l}$. In the special case of a list of sorts $s_1,\dots,s_n$ we usually abbreviate to w. When we define order-sorted signatures Sg_1, Sg_2, Sg, Sg', etc., we shall implicitly assume that their posets of sorts are denoted by $(S_1, \leq_1)$, $(S_2, \leq_2)$, $(S, \leq)$, $(S', \leq')$, etc., respectively.

Key ingredients of *order-sorted equational theories* are the definitions of *signature* and *algebra*. The former defines the symbols from which the terms of a language are built, and the latter provides terms with a meaning. This meaning can be both set-theoretic and category-theoretic [13,20].

Definition 2.1 (Order-Sorted Signature) *An **order-sorted signature** Sg is specified by*

- *a poset $(S, \leq)$ of **sorts**;*
- *a collection of **function symbols** $f\colon s_1,\dots,s_a \to s$ each with **arity** $a \geq 1$ and $(w,s) \in S^a \times S$ the **rank** of f where $w \triangleq s_1,\dots,s_a$;*
- *a collection of **constants** $k\colon s$, each of a* unique ***rank** s (just a single sort); and*
- *a **monotonicity requirement** that whenever $f\colon w_1 \to s$ and $f\colon w_2 \to r$ with $w_1 \leq w_2$, then $s \leq r$.*

*By an **operator** we mean either a function symbol or a constant.*

A key property of such signatures Sg, related to polymorphism, is *regularity*. We will shortly show how to build a set of *terms* out of Sg, and regularity ensures that each term has a unique least sort [13, Proposition 2.10]. (All signatures in this paper are assumed regular).

Definition 2.2 (Regularity of an Order-Sorted Signature) *An order-sorted signature Sg is **regular** if for each $f\colon w \to s$ and for each lower bound $w_l \leq w$ the set $\{\, (w', s') \mid f\colon w' \to s' \wedge w_l \leq w' \,\} \subseteq S^n \times S$ has a minimum, called the **least rank of f with respect to** w_l.*

Raw terms over a signature Sg are defined by $t ::= x \mid k \mid f(t_1,\dots,t_a)$ with $x \in Var$ (a countably infinite set of variables), k a constant, and f a function symbol with arity a. A **context** is a finite list of ordered pairs $x\colon s$ formed by a variable x and a sort s in Sg. We usually define a context by writing $\Gamma \triangleq [x_1\colon s_1,\dots,x_n\colon s_n]$. We denote context concatenation of Γ and Γ' by Γ, Γ'. We work with sorting judgements of the form $\Gamma \vdash t\colon s$. Those that are generated by the sorting rules in Figure 1 are called **proved terms**. Note that a term t may have more than one sort s for which $\Gamma \vdash t\colon s$ is a proved term. However there is always a unique least sort.

Lemma 2.3 (Terms Have A Least Sort) *Suppose that Γ is a context and t a raw term for a given regular signature Th. If there is any sort s for which $\Gamma \vdash t\colon s$ is a proved term, then there is a least such sort, s_t.*

Proof. One uses rule induction. The proof is easy, though in the literature a key step is often omitted. By induction, for the rule Fn, one easily uses regularity to obtain a sort, say $\check{s}$, such that $\Gamma \vdash f(t_1,\dots,t_a)\colon \check{s}$. Now $\check{s}$ is a candidate for the least sort of $f(t_1,\dots,t_a)$. Most authors state that such a sort $\check{s}$ is least. This is true, but proving it so requires a separate (though trivial) rule induction. ◻

We denote by $t[u/x]$ the **substitution** of the raw term u for the variable x in t, and by $t[\boldsymbol{u}/\boldsymbol{x}]$ the **simultaneous substitution** of raw terms $\boldsymbol{u} \triangleq u_1,\dots,u_n$ for variables $\boldsymbol{x} \triangleq x_1,\dots,x_n$.

Definition 2.4 (Inclusion Structure and FPI-category) *An **inclusion structure** $\mathbb{I}$ in a category $\mathcal{C}$ is specified by a subposet (subcategory) $\mathbb{I}$ of $\mathcal{C}$ such that*

- *for any two objects A and B of $\mathcal{C}$, the unique morphism $A \rightarrowtail B$ in $\mathbb{I}$, if any, is a monic in $\mathcal{C}$;*
- *for any object A in $\mathcal{C}$, the identity id_A is in $\mathbb{I}$ (so $\mathbb{I}$ is a* luff *subcategory: it has the same objects as $\mathcal{C}$);*

$$\frac{-}{[\![\Gamma, x\colon s, \Gamma' \vdash x\colon s]\!] \triangleq \pi\colon [\![\Gamma]\!] \times [\![s]\!] \times [\![\Gamma']\!] \to [\![s]\!]} \qquad \frac{-}{[\![\Gamma \vdash k\colon s]\!] \triangleq [\![k]\!] \circ !\colon [\![\Gamma]\!] \to 1 \to [\![s]\!]}\ (k\colon s)$$

$$\frac{(\forall 1 \leq i \leq a)\ \ [\![\Gamma \vdash t_i\colon s_i]\!] \triangleq m_i\colon [\![\Gamma]\!] \to [\![s_i]\!]}{[\![\Gamma \vdash f(t_1,\ldots,t_a)\colon s]\!] \triangleq [\![f]\!] \circ \langle m_1,\ldots,m_a\rangle\colon [\![\Gamma]\!] \to (\prod_1^a [\![s_i]\!]) \to [\![s]\!]}\ (f\colon s_1,\ldots,s_a \to s)$$

$$\frac{[\![\Gamma \vdash t\colon s]\!] = m\colon [\![\Gamma]\!] \to [\![s]\!]}{[\![\Gamma \vdash t\colon r]\!] = [\![s \leq r]\!] \circ m\colon [\![\Gamma]\!] \to [\![s]\!] \to [\![r]\!]}\ (s \leq r)$$

Fig. 2. Categorical semantics for proved terms.

- *if $\iota_1\colon A_1 \rightarrowtail B_1$ and $\iota_2\colon A_2 \rightarrowtail B_2$ are morphisms in $\mathbb{I}$, then so to is $\iota_1 \times \iota_2\colon A_1 \times A_2 \rightarrowtail B_1 \times B_2$.*

*A pair $(\mathcal{C}, \mathbb{I})$ is called an **FPI-category**. The intuition is that products model lists of sorts, and inclusions model subsort polymorphism.*

Thus, an FPI-category can be used as the basis for a definition of an *algebra* for a signature, namely

Definition 2.5 (Order-Sorted Algebra) *Given an order-sorted signature Sg, an Sg-**algebra** $\mathbf{A}$ in an FPI-category $(\mathcal{C}, \mathbb{I})$ is specified by*

- *an object $[\![s]\!]_{\mathbf{A}}$ in $\mathcal{C}$ for each sort s and object $[\![w]\!]_{\mathbf{A}} \triangleq [\![s_1]\!]_{\mathbf{A}} \times \cdots \times [\![s_n]\!]_{\mathbf{A}}$ for each $w \triangleq s_1,\ldots,s_n \in S^n$;*
- *morphisms $[\![f\colon w \to s]\!]_{\mathbf{A}}\colon [\![w]\!]_{\mathbf{A}} \to [\![s]\!]_{\mathbf{A}}$ and $[\![k]\!]_{\mathbf{A}}\colon 1 \to [\![s]\!]_{\mathbf{A}}$ for each $f\colon w \to s$ and $k\colon s$; and*
- *a morphism $[\![s \leq r]\!]_{\mathbf{A}}\colon [\![s]\!]_{\mathbf{A}} \rightarrowtail [\![r]\!]_{\mathbf{A}}$ in $\mathbb{I}$ for each $s \leq r$ in S, where we set $[\![s \leq s]\!]_{\mathbf{A}} \triangleq id_{[\![s]\!]}$*

such that if the function symbol f appears with more than one rank $f\colon w_1 \to s$ and $f\colon w_2 \to r$ in Sg with $s_1,\ldots,s_a \triangleq w_1 \leq w_2 \triangleq r_1,\ldots,r_a$, then the following diagram commutes:

$$\begin{tikzcd}
{[\![s_1]\!]_{\mathbf{A}} \times \cdots \times [\![s_a]\!]_{\mathbf{A}}} \arrow[r, "{[\![f\colon w_1 \to s]\!]_{\mathbf{A}}}"] \arrow[d, "{[\![s_1 \leq r_1]\!]_{\mathbf{A}} \times \cdots \times [\![s_a \leq r_a]\!]_{\mathbf{A}}}"'] & {[\![s]\!]_{\mathbf{A}}} \arrow[d, "{[\![s \leq r]\!]_{\mathbf{A}}}"] \\
{[\![r_1]\!]_{\mathbf{A}} \times \cdots \times [\![r_a]\!]_{\mathbf{A}}} \arrow[r, "{[\![f\colon w_2 \to r]\!]_{\mathbf{A}}}"] & {[\![r]\!]_{\mathbf{A}}}
\end{tikzcd}$$

From now on, we drop the algebra subscript and the ranks of function symbols in the semantic brackets whenever they are clear by context.

Definition 2.6 (Order-Sorted Homomorphism) *Let Sg be an order-sorted signature and let $\mathbf{A}$ and $\mathbf{B}$ be Sg-algebras. An Sg-**homomorphism** $h\colon \mathbf{A} \to \mathbf{B}$ is an S-sorted morphism $(h_s\colon [\![s]\!]_{\mathbf{A}} \to [\![s]\!]_{\mathbf{B}} \mid s \in S)$ such that given $f\colon s_1,\ldots,s_a \to s$, $k\colon s$, and $s \leq r$ in Sg the following diagrams commute:*

$$\begin{tikzcd}
{[\![s_1]\!]_{\mathbf{A}} \times \cdots \times [\![s_a]\!]_{\mathbf{A}}} \arrow[r, "{[\![f]\!]_{\mathbf{A}}}"] \arrow[d, "h_{s_1} \times \cdots \times h_{s_a}"'] & {[\![s]\!]_{\mathbf{A}}} \arrow[d, "h_s"] \\
{[\![s_1]\!]_{\mathbf{B}} \times \cdots \times [\![s_a]\!]_{\mathbf{B}}} \arrow[r, "{[\![f]\!]_{\mathbf{B}}}"] & {[\![s]\!]_{\mathbf{B}}}
\end{tikzcd}
\qquad
\begin{tikzcd}
1 \arrow[r, "{[\![k]\!]_{\mathbf{A}}}"] \arrow[dr, "{[\![k]\!]_{\mathbf{B}}}"'] & {[\![s]\!]_{\mathbf{A}}} \arrow[d, "h_s"] \\
 & {[\![s]\!]_{\mathbf{B}}}
\end{tikzcd}
\qquad
\begin{tikzcd}
{[\![s]\!]_{\mathbf{A}}} \arrow[r, "{[\![s \leq r]\!]_{\mathbf{A}}}"] \arrow[d, "h_s"'] & {[\![r]\!]_{\mathbf{A}}} \arrow[d, "h_r"] \\
{[\![s]\!]_{\mathbf{B}}} \arrow[r, "{[\![s \leq r]\!]_{\mathbf{B}}}"] & {[\![r]\!]_{\mathbf{B}}}
\end{tikzcd}$$

We define $h_w \triangleq h_{s_1} \times \cdots \times h_{s_a}$ provided that $w \triangleq s_1,\ldots,\times s_a$.

Given an order-sorted signature Sg, the class of all the order-sorted Sg-algebras and the class of all the order-sorted Sg-homomorphisms form a category denoted by $\mathcal{OSAlg}(\mathcal{C}, \mathbb{I})_{Sg}$.

If $\Gamma \vdash t\colon s$ is a proved term in a *regular* signature Sg and $\Gamma \triangleq [x_1\colon s_1,\ldots,x_n\colon s_n]$, any Sg-algebra $\mathbf{A}$ induces a (unique) morphism from $[\![\Gamma]\!]_{\mathbf{A}} \triangleq [\![s_1]\!]_{\mathbf{A}} \times \cdots \times [\![s_n]\!]_{\mathbf{A}}$ to $[\![s]\!]_{\mathbf{A}}$ in $\mathcal{C}$ according to the inductive definition that appears in Figure 2. We denote such an arrow by $[\![\Gamma \vdash t\colon s]\!]_{\mathbf{A}}$ and we refer to it as the **semantics of** $\Gamma \vdash t\colon s$.

Since terms can be assigned different types in one given context, we should consider whether the definition in Figure 2 is a sensible one. As such, we have the following lemma, where one sees that semantics of substitutions of terms is given as usual by morphism composition:

Lemma 2.7 (Well-Defined Semantics) *Given a proved term $\Gamma \vdash t\colon s$ and an algebra $\mathbf{A}$:*

- *The semantic morphism $[\![\Gamma \vdash t\colon s]\!]$ is unique; that is, the assignment $\xi \mapsto [\![\xi]\!]$ is a total function.*

Let $\Gamma \triangleq [x_1\colon s'_1, \dots, x_n\colon s'_n]$ be a context in (AxSub) and (AxCSub).

$$\text{(AxSub)}\ \frac{\Gamma \vdash t = t'\colon s \in Ax \qquad (\forall 1 \le i \le n)\ \Gamma' \vdash u_i\colon s'_i}{\Gamma' \vdash t[\boldsymbol{u}/\boldsymbol{x}] = t'[\boldsymbol{u}/\boldsymbol{x}]\colon s} \qquad \text{(AxCSub)}\ \frac{\bigwedge_1^m\ \Gamma \vdash t_\alpha = t'_\alpha\colon s_\alpha \implies \Gamma \vdash t = t'\colon s \in Ax \qquad (\forall 1 \le i \le n)\ \Gamma' \vdash u_i\colon s'_i \qquad (\forall 1 \le \alpha \le m)\ \Gamma' \vdash t_\alpha[\boldsymbol{u}/\boldsymbol{x}] = t'_\alpha[\boldsymbol{u}/\boldsymbol{x}]\colon s_\alpha}{\Gamma' \vdash t[\boldsymbol{u}/\boldsymbol{x}] = t'[\boldsymbol{u}/\boldsymbol{x}]\colon s}$$

$$\text{(Ref)}\ \frac{\Gamma \vdash t\colon s}{\Gamma \vdash t = t\colon s} \qquad \text{(Sym)}\ \frac{\Gamma \vdash t = t'\colon s}{\Gamma \vdash t' = t\colon s} \qquad \text{(Trans)}\ \frac{\Gamma \vdash t = t'\colon s \qquad \Gamma \vdash t' = t''\colon s}{\Gamma \vdash t = t''\colon s}$$

$$\text{(Subsort)}\ \frac{\Gamma \vdash t = t'\colon s}{\Gamma \vdash t = t'\colon r}\ (s \le r) \qquad \text{(Cong)}\ \frac{\Gamma \vdash t\colon s \qquad (\forall 1 \le i \le n)\ \Gamma' \vdash u_i = u'_i\colon s_i}{\Gamma' \vdash t[\boldsymbol{u}/\boldsymbol{x}] = t[\boldsymbol{u'}/\boldsymbol{x}]\colon s}\ (\Gamma \triangleq [x_1\colon s_1, \dots, x_n\colon s_n])$$

Fig. 3. Theorems generated by an order-sorted theory $Th \triangleq (Sg, Ax)$.

- *The algebra induces a functor $(S, \le) \to \mathbb{I}$ between posetal categories, where $s \le s' \mapsto [\![s \le s']\!]\colon [\![s]\!] \rightarrowtail [\![s']\!]$, and so as a consequence the semantics can be factored through the morphism $[\![\Gamma \vdash t\colon \mathsf{s}_t]\!]$, that is to say, $[\![\Gamma \vdash t\colon s]\!] = [\![\mathsf{s}_t \le s]\!] \circ [\![\Gamma \vdash t\colon \mathsf{s}_t]\!]$.*
- *Let $\Gamma \triangleq [x_1\colon s_1, \dots, x_n\colon s_n]$. If we have proved terms $\Gamma \vdash t\colon s$ and $\Gamma' \vdash u_i\colon s_i$ where $1 \le i \le n$, then $\Gamma' \vdash t[\boldsymbol{u}/\boldsymbol{x}]\colon s$ and $[\![\Gamma' \vdash t[\boldsymbol{u}/\boldsymbol{x}]\colon s]\!] = [\![\Gamma \vdash t\colon s]\!] \circ \langle [\![\Gamma' \vdash u_1\colon s_1]\!], \dots, [\![\Gamma' \vdash u_n\colon s_n]\!] \rangle$.*

Equations between proved terms are defined only for *coherent* signatures. To define coherence, first let $\equiv$ be the symmetric and transitive closure of $\le$. The equivalence classes induced by $\equiv$ on S are the **connected components** of $(S, \le)$; and $(S, \le)$ is **locally filtered**, if for every two sorts s' and s'' in the same *connected component* there is a sort s such that $s', s'' \le s$. Then a signature Sg is said to be **coherent** if and only if it is regular and locally filtered. The intuition is that terms of sort s' and s'' respectively could potentially be judged equal if they have a "common super-sort" s.

Definition 2.8 (Order-Sorted Equation and Satisfaction) *Let Sg be a coherent signature. An **equation (in-context)** in Sg is denoted by $\Gamma \vdash t = t'\colon s$, where*

- *there are sorts s', s'' such that $\Gamma \vdash t\colon s'$ and $\Gamma \vdash t'\colon s''$ are proved terms in Sg;*
- *s' and s'' fall in the same connected component of $(S, \le)$; and*
- *s is a common supersort of s' and s'' (which exists by the coherence condition).*

*A **conditional equation (in-context)** in Sg is a list of $m+1$ equations-in-context (where $m \ge 1$) suggestively denoted by $\bigwedge_{\alpha=1}^m\ \Gamma \vdash t_\alpha = t'_\alpha\colon s_\alpha \implies \Gamma \vdash t = t'\colon s$. We say that an algebra $\mathbf{A}$ **satisfies** an equation if $[\![\Gamma \vdash t\colon s]\!] = [\![\Gamma \vdash t'\colon s]\!]$. We say that an algebra $\mathbf{A}$ **satisfies** a conditional equation if for all morphisms $u\colon U \to [\![\Gamma]\!]$, $[\![\Gamma \vdash t_\alpha\colon s_\alpha]\!] \circ u = [\![\Gamma \vdash t'_\alpha\colon s_\alpha]\!] \circ u$ implies $[\![\Gamma \vdash t\colon s]\!] \circ u = [\![\Gamma \vdash t'\colon s]\!] \circ u$.*

We define an **order-sorted theory** $Th \triangleq (Sg, Ax)$ to be a pair consisting of a signature Sg and a **set of axioms** Ax. Each axiom is either an equation or a conditional equation. The **theorems** of the theory Th are those *equations* generated by the rules of *equational logic* in Figure 3.

Lemma 2.9 (Generalised Substitution) *The following rule is admissible by a routine rule induction*

$$\text{(Sub)}\ \frac{\Gamma \vdash t = t'\colon s \qquad (\forall 1 \le i \le n)\ \Gamma' \vdash u_i\colon s_i}{\Gamma' \vdash t[\boldsymbol{u}/\boldsymbol{x}] = t'[\boldsymbol{u}/\boldsymbol{x}]\colon s}\ (\Gamma \triangleq [x_1\colon s_1, \dots, x_n\colon s_n])$$

Let $Th \triangleq (Sg, Ax)$ be an order-sorted theory. If an Sg-algebra $\mathbf{A}$ satisfies all the axioms in Ax, we call $\mathbf{A}$ a **model** of Th. The **category of models** $\mathcal{OSMod}(\mathcal{C}, \mathbb{I})_{Th}$ is the full subcategory of $\mathcal{OSAlg}(\mathcal{C}, \mathbb{I})_{Sg}$ given by all the models of Th in $(\mathcal{C}, \mathbb{I})$.

Lemma 2.10 (Satisfaction is Well-Defined) *As a consequence of Lemma 2.7, satisfaction is well-defined up to subsort-polymorphic equality, as follows: Suppose that we have a theorem $\Gamma \vdash t = t'\colon s$ satisfied in a model $\mathbf{A}$. If $\Gamma \vdash t = t'\colon \hat{s}$ is also a theorem, then it too is satisfied.*

Proof. The existence of least sorts s_t and $\mathsf{s}_{t'}$ means that s and $\hat{s}$ are connected, and so have a super-type s'. Thus each term has this type s', and the result follows by using factorisation from Lemma 2.7 and the left-cancellation properties of monomorphisms. □

The category $Pres_{\times,\rightarrowtail}((\mathcal{C}, \mathbb{I}), (\mathcal{D}, \mathbb{J}))$ is defined by having objects functors $F\colon \mathcal{C} \to \mathcal{D}$ such that finite products are preserved and F restricts to a functor $F_{|\mathbb{I}}\colon \mathbb{I} \to \mathbb{J}$ (that is monics are also preserved). Suppose that we have a model $\mathbf{A}$ in $\mathcal{OSMod}(\mathcal{C},\mathbb{I})_{Th}$. Then there is a model $F_*\mathbf{A}$ in $\mathcal{OSMod}(\mathcal{D},\mathbb{J})_{Th}$ that is, roughly speaking, defined by "taking the image of Sg in $(\mathcal{C},\mathbb{I})$ induced by the model $\mathbf{A}$, and *applying* F". Equally one may "apply F to homomorphisms of models" and this process (which is absolutely standard in categorical type theory/logic; see for example [6,16]) leads to a functor $Ap_{\mathbf{A}}\colon Pres_{\times,\rightarrowtail}((\mathcal{C},\mathbb{I}),(\mathcal{D},\mathbb{J})) \to \mathcal{OSMod}(\mathcal{D},\mathbb{J})_{Th}$. A **classifying category** $Cl(Th)$ for a theory Th is an FPI-category such that there is an equivalence of categories

$$Ap_{\mathbf{G}}\colon Pres_{\times,\rightarrowtail}(Cl(Th),(\mathcal{D},\mathbb{J})) \simeq \mathcal{OSMod}(\mathcal{D},\mathbb{J})_{Th}$$

and thus models of Th in $(\mathcal{D},\mathbb{J})$ correspond to such structure preserving functors with source $Cl(Th)$.

Theorem 2.11 (Existence of Classifying Category) *There is an FPI-category $Cl(Th)$ constructed out of the syntax of Th, in which there is a **generic** model of Th with the property that equality of morphisms corresponds to derivability of term equations. At an abstract level this notion is standard in categorical type theory/logic; see for example [6,16]. We feel that our concrete construction is simpler than that found in [20], and regard this as a small contribution.* Such existence proofs are notoriously tricky to get completely correct, and there are notable errors in the literature. We use matching contexts and permutation invariance [7,29] to replace the usual substitutions that rename variables, and we think this makes our proofs simpler to state and prove (and hence less error prone).

Proof. Let Var be a (countable) fixed set of variables $\{V_1, V_2, \ldots\}$. We call the context $\Gamma_{\boldsymbol{s}} \triangleq [V_1\colon s_1,\ldots,V_n\colon s_n])$ the **primary** context for any sorts $s_1,\ldots,s_n$. In general, below, the metavariables x, y, z etc, possibly subscripted, *range over* Var. Thus $\Gamma \triangleq [x_1\colon s_1,\ldots,x_n\colon s_n]$ is a typical context as before, and we say that any such context **matches** $\boldsymbol{s} \triangleq s_1,\ldots,s_n$.

First we define the objects $\mathbf{T}(\Gamma)$ of $\mathcal{F}OSAlg$. Of course $\mathbf{T}(\Gamma)$ must be an S-sorted set, and the components are sets of equivalence classes

$$\mathbf{T}(\Gamma)_s \triangleq Term(\Gamma)_s/\sim \qquad \text{where} \qquad Term(\Gamma)_s \triangleq \{\, t \mid \Gamma \vdash t\colon s \text{ in } Th \,\}$$

where we define the equivalence relation $t \sim t'$ just in case we can derive $\Gamma \vdash t = t'\colon s$ in Th, and write $[\,t\,]$ for a typical equivalence class.

Let Γ and Γ' match $s_1,\ldots,s_n$ and $s'_1,\ldots,s'_m$ respectively. The morphisms $h\colon \mathbf{T}(\Gamma) \to \mathbf{T}(\Gamma')$ must be S-sorted functions $(h_s\colon \mathbf{T}(\Gamma)_s \to \mathbf{T}(\Gamma')_s \mid s \in S)$. These are specified by lists $h_s \triangleq \{\Gamma' \vdash t_1,\ldots,t_n\}$ where $t_i \in Term(\Gamma')_{s_i}$ and where

$$h_s([\,\hat{t}\,] \in \mathbf{T}(\Gamma)_s) \triangleq [\,\hat{t}[t_1,\ldots,t_m/x_1,\ldots,x_m]\,] \in \mathbf{T}(\Gamma')_s$$

It is easy to check this is well defined. Note that if

$$h \triangleq \{\Gamma' \vdash t_1,\ldots,t_n\}\colon \mathbf{T}(\Gamma) \to \mathbf{T}(\Gamma') \qquad \text{and} \qquad h' \triangleq \{\Gamma'' \vdash t'_1,\ldots,t'_m\}\colon \mathbf{T}(\Gamma') \to \mathbf{T}(\Gamma'')$$

then we have $h \circ h'$ defined by

$$\{\Gamma'' \vdash t_1[t'_1,\ldots,t'_m/x_1,\ldots,x_m],\ldots,t_n[t'_1,\ldots,t'_m/x_1,\ldots,x_m]\}$$

It is tedious but routine to verify that this gives rise to a category, relying crucially on the substitution rules for equation derivation. $\square$

Theorem 2.12 (Soundness and Completeness) *Let Th be an order-sorted theory. $\Gamma \vdash t = t'\colon s$ is a theorem of Th if and only if $\Gamma \vdash t = t'\colon s$ is satisfied by every model of Th.*

Proof. Soundness follows by rule induction for Figure 3. For completeness, suppose that $\Gamma \vdash t = t'\colon s$ is satisfied in any model. Then in particular it is satisfied in the generic model $\mathbf{G}$ in the classifying category $Cl(Th)$. Thus we have $[\![\Gamma \vdash t\colon s]\!]_{\mathbf{G}} = [\![\Gamma \vdash t'\colon s]\!]_{\mathbf{G}}$ and so we have $(\Gamma \mid t) = (\Gamma \mid t')$ which holds precisely when $\Gamma \vdash t = t'\colon s$ is a theorem. $\square$

We conclude this section with a new result, although it is motivated by analogous theorems [28]. The proof also makes use of matching contexts and permutations of variables.

Theorem 2.13 (Relationship to Free Algebras) *There is an equivalence between FPI-categories $Cl(Th)$ and $(\mathcal{F}OSAlg^{op},\mathbb{J})$ where $\mathcal{F}OSAlg$ is the category of free order-sorted algebras over finite sets of* *variables, and order-sorted homomorphisms. Moreover the equivalence is given by an FPI-functor* $\Phi\colon \mathit{Cl}(\mathit{Th}) \simeq (\mathcal{F}\mathit{OSAlg}^{op}, \mathbb{J})$.

Proof. With a view to showing that $(\mathcal{F}\mathit{OSAlg}^{op}, \mathbb{J})$ is an FPI-category, we shall show that $\mathcal{F}\mathit{OSAlg}$ is has finite coproducts, and then define $\mathbb{J}$. Given objects $\mathbf{T}(\Gamma)$ and $\mathbf{T}(\Gamma')$ then the binary coproduct object is given by $\mathbf{T}(\Delta \triangleq \Gamma_{\boldsymbol{s}}, \Gamma_{\boldsymbol{s}'})$ where the sorts match Γ and Γ' respectively. The coproduct insertions are given by $\{\Delta \vdash v_1, \dots, v_n\}$ and $\{\Delta \vdash v_{n+1}, \dots, v_{n+m}\}$. Given morphisms

$$\{\Gamma'' \vdash t_1, \dots, t_n\}\colon \mathbf{T}(\Gamma) \to \mathbf{T}(\Gamma'') \qquad \text{and} \qquad \{\Gamma'' \vdash t'_1, \dots, t'_m\}\colon \mathbf{T}(\Gamma') \to \mathbf{T}(\Gamma'')$$

then the mediating morphism is $\{\Gamma'' \vdash t_1, \dots, t_n, t'_1, \dots, t'_m\}$. Note that $\mathbf{T}(\varnothing)$ is the initial object.

Suppose that $s_i \leq r_i$ for each $1 \leq i \leq n$. Then there is an epic morphism $i \triangleq \{\Gamma_{\boldsymbol{s}} \vdash x_1, \dots, x_n\}\colon \mathbf{T}(\Gamma_{\boldsymbol{r}}) \to \mathbf{T}(\Gamma_{\boldsymbol{s}})$; it's easy to verify that this is an epimorphism, and hence yields a monomorphism in $\mathcal{F}\mathit{OSAlg}^{op}$. The luff subcategory $\mathbb{J}$ has all of its morphisms the monomorphisms $i^{op} \triangleq \{\Gamma_{\boldsymbol{s}} \vdash x_1, \dots, x_n\}\colon \mathbf{T}(\Gamma_{\boldsymbol{s}}) \to \mathbf{T}(\Gamma_{\boldsymbol{r}})$. This is certainly an inclusion category.

Now we prove the equivalence. We define a functor $\Phi\colon \mathit{Cl}(\mathit{Th}) \to (\mathcal{F}\mathit{OSAlg}^{op}, \mathbb{J})$ as follows. Given a morphism $(\Gamma \mid t_1) \dots (\Gamma \mid t_m)\colon \boldsymbol{s} \to \boldsymbol{r}$ then Φ sends this to

$$\{\Gamma_{\boldsymbol{s}} \vdash \pi t_1, \dots, \pi t_m\}\colon \mathbf{T}(\Gamma_{\boldsymbol{r}}) \to \mathbf{T}(\Gamma_{\boldsymbol{s}})$$

where permutation π is specified by $\pi\colon x_i \mapsto V_i$. We check this is well defined. Suppose that

$$(\Gamma \mid t_1) \dots (\Gamma \mid t_m) = (\Gamma' \mid t'_1) \dots (\Gamma \mid t'_m).$$

We need to check that

$$\{\Gamma_{\boldsymbol{s}} \vdash \pi t_1, \dots, \pi t_m\} \triangleq \Phi((\Gamma \mid t_1) \dots (\Gamma \mid t_m)) = \Phi((\Gamma' \mid t'_1) \dots (\Gamma \mid t'_m)) \triangleq \{\Gamma_{\boldsymbol{s}} \vdash \pi' t'_1, \dots, \pi' t'_m\}$$

By definition we have $\Gamma \vdash t_j = \rho t'_j \colon r_j$ where ρ is specified by $\rho\colon x'_i \mapsto x_i$. We can deduce, using substitution rules for equations, that $\pi\Gamma \vdash \pi t_j = \pi(\rho t'_j)\colon r_j$ and this is exactly $\Gamma_{\boldsymbol{s}} \vdash \pi t_j = \pi' t'_j \colon r_j$ as required, since $\pi' = \pi \circ \rho$. We feel that the use of permutations, while equivalent to the use of simultaneous variable renamings by substitution, improves readability and more importantly simplifies calculations by making use of judgements that are permutation invariant.

(Φ is essentially surjective):

For any object $\boldsymbol{s}$ in $\mathit{Cl}(\mathit{Th})$ we have $\Phi(\boldsymbol{s}) = \mathbf{T}(\Gamma_{\boldsymbol{s}})$. But one easily shows that for any $\mathbf{T}(\Gamma)$ where Γ matches $\boldsymbol{s}$, we have $\mathbf{T}(\Gamma) \cong \mathbf{T}(\Gamma_{\boldsymbol{s}})$ where the inverse homomorphisms "swap variables" x_i and V_i.

(Φ is faithful): Let

$$\Phi((\Gamma \mid t_1) \dots (\Gamma \mid t_m)) = \Phi((\Gamma' \mid t'_1) \dots (\Gamma \mid t'_m)).$$

We need to check that $\Gamma \vdash t_j = \rho t'_j \colon r_j$. By the assumption we have

$$\{\Gamma_{\boldsymbol{s}} \vdash \pi t_1, \dots, \pi t_m\} = \{\Gamma_{\boldsymbol{s}} \vdash \pi' t'_1, \dots, \pi' t'_m\}$$

Hence $\Gamma_{\boldsymbol{s}} \vdash \pi t_j = \pi t'_j \colon r_j$. Therefore we can deduce that $\Gamma \vdash t_j = (\pi^{-1} \circ \pi') t'_j \colon r_j$; we are done since $\pi^{-1} \circ \pi' = \rho$.

(Φ is full): Let

$$\{\Gamma_{\boldsymbol{s}} \vdash t_1, \dots, t_m\}\colon \Phi(\boldsymbol{r}) = \mathbf{T}(\Gamma_{\boldsymbol{r}}) \to \Phi(\boldsymbol{s}) = \mathbf{T}(\Gamma_{\boldsymbol{s}})$$

Then $(\Gamma_{\boldsymbol{s}} \mid t_1, \dots, t_m)$ is the appropriate $\mathit{Cl}(\mathit{Th})$ morphism.

(Φ is an object of $\mathit{Pres}_{\times,\rightarrowtail}(\mathit{Cl}(\mathit{Th}), (\mathcal{F}\mathit{OSAlg}^{op}, \mathbb{J}))$): We need to check that $\Phi_{|\mathbb{I}}\colon \mathbb{I} \to \mathbb{J}$ where $\mathbb{I}$ is the inclusion category of $\mathit{Cl}(\mathit{Th})$. Let $\boldsymbol{s} \leq \boldsymbol{r}$. Note that $\Phi((\Gamma \mid x_1) \dots (\Gamma \mid x_n)\colon \boldsymbol{s} \to \boldsymbol{r})$ is the monomorphism $\{\Gamma_{\boldsymbol{s}} \vdash x_1, \dots, x_n\}\colon \Gamma_{\boldsymbol{r}} \to \Gamma_{\boldsymbol{s}}$ □

3 Multi-Language Equational Logic

3.1 Fundamentals of Multi-Languages

Throughout this section we often refer to a **Running Example**, introduced below and subsequently extended, to illustrate how the theory works in a concrete setting (see Section 3.3 for the outline of a more complex example).

$\mathtt{0}\colon \mathbb{nat}$	zero constant	$(\forall c \in \mathtt{Char})\ \ c\colon \mathbb{chr}$	character constant
$\mathtt{s}\colon \mathbb{nat} \to \mathbb{nat}$	successor function	$\mathtt{next}\colon \mathbb{chr} \to \mathbb{chr}$	next character function
$\mathtt{+}\colon \mathbb{nat}, \mathbb{nat} \to \mathbb{nat}$	addition operator	$\mathtt{+}\colon \mathbb{str}, \mathbb{str} \to \mathbb{str}$	string concatenation

(a) Sg_1 operators. (b) Sg_2 operators.

Fig. 4. Operators of order-sorted signatures Sg_1 and Sg_2.

Sorts	$[\![\mathbb{nat}]\!]_{\mathbf{A}_1} \triangleq \mathbb{N}$	$[\![\mathbb{chr}]\!]_{\mathbf{A}_2} \triangleq \mathtt{Char}$ $[\![\mathbb{str}]\!]_{\mathbf{A}_2} \triangleq \mathtt{Char}^*$
Operators	$[\![\mathtt{0}]\!]_{\mathbf{A}_1} \triangleq * \mapsto 0\colon 1 \to \mathbb{N}$ $[\![\mathtt{s}]\!]_{\mathbf{A}_1} \triangleq n \mapsto n+1\colon \mathbb{N} \to \mathbb{N}$ $[\![\mathtt{+}]\!]_{\mathbf{A}_1} \triangleq (n_1, n_2) \mapsto n_1 + n_2\colon \mathbb{N}^2 \to \mathbb{N}$	$(\forall c \in \mathtt{Char})\ [\![c]\!]_{\mathbf{A}_2} \triangleq * \mapsto c\colon 1 \to \mathtt{Char}$ $[\![\mathtt{next}]\!]_{\mathbf{A}_2} \triangleq c \mapsto c'\colon \mathtt{Char} \to \mathtt{Char}$ $[\![\mathtt{+}]\!]_{\mathbf{A}_2} \triangleq (s_1, s_2) \mapsto s_1 s_2\colon (\mathtt{Char}^*)^2 \to \mathtt{Char}^*$
Subsorts		$[\![\mathbb{chr} \leq_2 \mathbb{str}]\!]_{\mathbf{A}_2} \triangleq c \mapsto c\colon \mathtt{Char} \to \mathtt{Char}^*$

Fig. 5. Categorical semantics of Sg_1 and Sg_2.

Running Example. Our example is defined using the following order-sorted signatures:

- The signature Sg_1 defines the symbols of a language for constructing simple mathematical expressions over natural numbers in Peano's notation. Let the poset of sorts $(S_1, \leq_1)$ of Sg_1 be a poset with a single sort $\mathbb{nat}$ denoting the type of natural numbers, and let the operators be those in Figure 4a.
- Let $c \in \mathtt{Char} \triangleq \{\mathtt{a}, \mathtt{b}, \ldots, \mathtt{z}\}$ be the metavariable ranging over a finite set $\mathtt{Char}$ of characters. The signature Sg_2 defines a language to build strings over $\mathtt{Char}$. The set of sorts S_2 of Sg_2 carries the sort $\mathbb{str}$ for strings and the sort $\mathbb{chr}$ for characters. The subsort relation $\leq_2$ is the reflexive relation on S_2 plus $\mathbb{chr} \leq_2 \mathbb{str}$ (*i.e.*, characters are one-symbol strings), and the operator symbols in Sg_2 appear in Figure 4b.

We model Sg_1 and Sg_2 by the order-sorted algebras $\mathbf{A}_1$ and $\mathbf{A}_2$ (see Figure 5) in $(\mathcal{S}et, \mathcal{I}ncl)$, the FPI-category of sets with inclusion functions forming the inclusion structure. The symbol c' in the definition of $[\![\mathtt{next}]\!]_{\mathbf{A}_2}$ denotes the character that follows c in $\mathtt{Char}$ (assuming the standard alphabetical order).

Remark 3.1 The forthcoming definitions and results gradually define and illustrate multi-languages, and give relationships between multi-languages and order-sorted languages. A multi-language signature 3.2 is specified as two order-sorted signatures (as in the **Running Example**) together with *an interoperability relation* between the two signatures. This determines the terms of the multi-language. Note that the relation is *not* a universal property of the underlying signatures; and also note a multi-language signature *explicitly* provides users with the *original two* language specifications. We show in 3.3 that there are nice notions of categories of signatures, both order-sorted and multi-language. We shall also see that we can exhibit a functor that maps a multi-language signature to an order-sorted signature (the *associated signature* 3.4), blending the two original signatures into one. After defining multi-language algebras 3.5 and homomorphisms 3.6, Theorem 3.7 will provide, via associated signatures, a clear semantic relationship between multi and order-sorted languages. In particular, it suggests how to give the definitions of (multi-language) terms, equations, and satisfaction using the associated signature. We can then reason equationally about *interoperability* of the two given languages. This takes us to Section 3.2 where we study equational reasoning in detail.

In order to define multi-language signatures we introduce some crucial notation. We denote by $+$ the **disjoint union** of two sets: the insertion morphisms that form a coproduct in the category of sets are injective functions, thus they have left inverses (and one has a model of disjoint union). In the following, if S_1 and S_2 are two sets of sorts and $s \in S_i$ with $i \triangleq 1, 2$, we write

$$\boxed{s_i \quad \text{for the element} \quad \iota_i(s) \in S_1 + S_2 \quad \text{where} \quad \iota_i(s) \triangleq (s, i) \in S_1 \times \{1\} \cup S_2 \times \{2\}}$$

Thus in relationships $s_i \ltimes s'_j$ we have $s \in S_i$ and $s' \in S_j$. This is a very useful notation but perhaps requires care on first reading. Moreover, if $w \triangleq s_1, \ldots, s_n \in S_i^n$, then we write w_i for $(s_1)_i, \ldots, (s_n)_i$.

Definition 3.2 (Multi-Language Signature) *A* ***multi-language signature*** $SG \triangleq (Sg_1, Sg_2, \ltimes)$ *is*

- *a pair of order-sorted signatures* Sg_1 *and* Sg_2 *with posets of sorts* $(S_1, \leq_1)$ *and* $(S_2, \leq_2)$*, respectively; and*
- *a* (binary) relation ***join*** $\ltimes$ *over* $S_1 + S_2$ *such that* $s_i \ltimes s'_j$ *with* $i, j \in \{1, 2\}$ *and* $i \neq j$.

The idea is that if $s_i \ltimes s'_j$, and $\Gamma \vdash t\colon s$ is a proved term in *one* language, then t can be used in place of a term t' such that $\Gamma' \vdash t'\colon s'$ in the *other* language: as in [21], "ML code can be used in place of Scheme code". This is made precise in due course.

Definition 3.3 $\mathcal{OSSg}$ *is the* ***category of order-sorted signatures*** *with morphisms* $h\colon Sg_1 \to Sg_2$ *given by*

- *a monotone function* $h\colon (S_1, \leq_1) \to (S_2, \leq_2)$ *(where we will write* $h(w) \triangleq h(s_1), \ldots, h(s_n)$ *for* $w \triangleq s_1, \ldots, s_n \in S_1^n$*), and*
- *a mapping* h *from the operators in* Sg_1 *to those in* Sg_2 *that preserves rank: given* $k\colon s$ *in* Sg_1*, then* $h(k)\colon h(s)$ *in* Sg_1*; and given* $f\colon w \to s$ *in* Sg_1*, then* $h(f)\colon h(w) \to h(s)$ *in* Sg_2.

Moreover, we denote by $\mathcal{MLSg}$ *the* ***category of multi-language signatures*** *in which a morphism*

$$H \triangleq (h_1, h_2)\colon (Sg_1, Sg_2, \ltimes) \to (Sg'_1, Sg'_2, \ltimes')$$

is defined by two morphisms $h_1\colon Sg_1 \to Sg'_1$ *and* $h_2\colon Sg_2 \to Sg'_2$ *in* $\mathcal{OSSg}$ *such that they preserve the join relation, namely* $s_i \ltimes s'_j$ *in* $(Sg_1, Sg_2, \ltimes)$ *implies* $(h_i(s)_i \ltimes' (h_j(s'))_j$ *in* $(Sg'_1, Sg'_2, \ltimes')$.

Definition 3.4 (Associated Signature) *Let* $SG \triangleq (Sg_1, Sg_2, \ltimes)$ *be a multi-language signature. The* ***associated signature*** $\overline{SG}$ *of* SG *is the order-sorted signature defined as follows:*

- *the poset of sorts is given by* $(S_1 + S_2, \leq)$*, where* $s_i \leq r_j$ *if* $i = j$ *and* $s \leq_i r$*;*
- *if* $f\colon w \to s$ *is an operator in* Sg_i *for some* $i \triangleq 1, 2$*, then* $f_i\colon w_i \to s_i$ *is a function symbol in* $\overline{SG}$*;*
- *if* $k\colon s$ *is a constant in* Sg_i *for some* $i \triangleq 1, 2$*, then* $k_i\colon s_i$ *is a constant in* $\overline{SG}$*; and*
- *a* ***conversion operator*** $\hookrightarrow_{s_i, s'_j}\colon s_i \to s'_j$ *for each constraint* $s_i \ltimes s'_j$.

The **associated signature functor** $\overline{(-)}\colon \mathcal{MLSg} \to \mathcal{OSSg}$ maps each multi-language signature $SG \triangleq (Sg_1, Sg_2, \ltimes)$ to its associated signature $\overline{SG}$, and each multi-language signature morphism $H\colon SG \to SG'$ to the order-sorted signature morphism $\overline{H}\colon \overline{SG} \to \overline{SG'}$ given by $\overline{H}(s_i) \triangleq (h_i(s))_i$ for each $s \in S_i$ (hence $s_i \in S_1 + S_2$) and $\overline{H}(f_i) \triangleq (h_i(f))_i$ for each $f \in Sg_i$ (hence f_i in SG).

Running Example. $\overline{(-)}$ embeds the multi-language signature SG into $\mathcal{OSSg}$, providing the order-sorted version $\overline{SG}$ of the multi-language. $\overline{SG}$ generates Sg_i-terms (see Section 3.2) as well as hybrid multi-language terms involving conversion operators such as $[c\colon \mathbb{chr}_2, n\colon \mathbb{nat}_1] \vdash +_1(\hookrightarrow_{\mathbb{chr}_2, \mathbb{nat}_1}(c), n)\colon \mathbb{nat}_1$. From now on, we use colours in the examples for disambiguating the left and the right inclusion in place of subscripts $_1$ and $_2$. Moreover, we use an infix notation whenever the operators lend themselves well to do so. That is, the previous term is represented by $[c\colon \mathbb{chr}, n\colon \mathbb{nat}] \vdash \hookrightarrow_{\mathbb{chr}, \mathbb{nat}}(c) + n\colon \mathbb{nat}$.

Such a functor outlines an *embedding* of multi-language signatures into order-sorted signatures, enabling us to see a multi-language as an ordinary language. Indeed, it is easy to see that $\overline{(-)}$ is both injective on objects and a faithful functor.

Definition 3.5 (Multi-Language Algebra) *Let* $SG \triangleq (Sg_1, Sg_2, \ltimes)$ *be a multi-language signature. An* ***SG-algebra*** $\mathbf{A}$ *in an FPI-category* $(\mathcal{C}, \mathbb{I})$ *is given by*

- *a pair of order-sorted algebras* $\mathbf{A}_1$ *and* $\mathbf{A}_2$ *in* $(\mathcal{C}, \mathbb{I})$ *over* Sg_1 *and* Sg_2*, respectively; and*
- *a* ***boundary*** *morphism* $[\![s_i \ltimes s'_j]\!]_{\mathbf{A}}\colon [\![s]\!]_{\mathbf{A}_i} \to [\![s']\!]_{\mathbf{A}_j}$ *in* $\mathcal{C}$ *for each constraint* $s_i \ltimes s'_j$.

An algebra sets out the meaning of a multi-language: The meaning of the underlying languages, and *how* terms of sort $s \in S_i$ can be interpreted as terms of sort $s' \in S_j$. Put differently, "boundary morphisms regulate the flow of values across $\mathbf{A}_1$ and $\mathbf{A}_2$" [22].

Definition 3.6 (Multi-Language Homomorphism) *Let* $SG \triangleq (Sg_1, Sg_2, \ltimes)$ *be a multi-language signature, and let* $\mathbf{A}$ *and* $\mathbf{B}$ *be two SG-algebras. An* SG***-homomorphism*** $h\colon \mathbf{A} \to \mathbf{B}$ *is given by a pair of order-sorted homomorphisms* $h_1\colon \mathbf{A}_1 \to \mathbf{B}_1$ *and* $h_2\colon \mathbf{A}_2 \to \mathbf{B}_2$ *such that they commute with boundary functions, namely, if* $s_i \ltimes s'_j$*, then the following diagram commutes:*

$$\begin{tikzcd}
{[\![s]\!]_{\mathbf{A}_i}} \arrow[r, "{[\![s_i \ltimes s'_j]\!]_{\mathbf{A}}}"] \arrow[d, "(h_i)_s"'] & {[\![s']\!]_{\mathbf{A}_j}} \arrow[d, "(h_j)_{s'}"] \\
{[\![s]\!]_{\mathbf{B}_i}} \arrow[r, "{[\![s_i \ltimes s'_j]\!]_{\mathbf{B}}}"] & {[\![s']\!]_{\mathbf{B}_j}}
\end{tikzcd}$$

Given a multi-language signature SG, the class of all SG-algebras and SG-homomorphisms form a category denoted by $\mathcal{MLAlg}(\mathcal{C}, \mathbb{I})_{SG}$. We have a simple connection between this category and $\mathcal{OSAlg}(\mathcal{C}, \mathbb{I})_{\overline{SG}}$, outlined in Theorem 3.7, after more of the Running Example.

Running Example. Suppose we are interested in a multi-language $SG \triangleq (Sg_1, Sg_2, \ltimes)$ according to the following specifications:

- terms denoting natural numbers can be used in place of characters according to the function $\mathrm{chr}\colon \mathbb{N} \to \texttt{Char}$ that maps the natural number n to the n-th character symbol modulo $|\texttt{Char}|$; and
- terms denoting strings can be used in place of natural numbers according to the function $\mathrm{len}\colon \texttt{Char}^* \to \mathbb{N}$, namely the length of the string.

In order to get such a multi-language, we provide (1) the join relation $\ltimes$ on $S_1 + S_2$ and (2) a boundary morphism $[\![s_i \ltimes s'_j]\!]_{\mathbf{A}}\colon [\![s]\!]_{\mathbf{A}_i} \to [\![s']\!]_{\mathbf{A}_j}$ for each constraint $s_i \ltimes s'_j$ introduced by $\ltimes$:

- $\mathbb{not}_1 \ltimes \mathbb{chr}_2$ and $\mathbb{not}_1 \ltimes \mathbb{str}_2$ with boundaries $[\![\mathbb{not}_1 \ltimes \mathbb{chr}_2]\!]_{\mathbf{A}}(n) \triangleq [\![\mathbb{not}_1 \ltimes \mathbb{str}_2]\!]_{\mathbf{A}}(n) \triangleq \mathrm{chr}(n)$; and
- $\mathbb{chr}_2 \ltimes \mathbb{not}_1$ and $\mathbb{str}_2 \ltimes \mathbb{not}_1$ with boundaries $[\![\mathbb{chr}_2 \ltimes \mathbb{not}_1]\!]_{\mathbf{A}}(c) \triangleq \mathrm{len}(c) = 1$ and $[\![\mathbb{str}_2 \ltimes \mathbb{not}_1]\!]_{\mathbf{A}}(s) \triangleq \mathrm{len}(s)$.

The next theorem yields a formal correspondence between multi-languages and order-sorted languages: We can make a multi-language signature SG into an order-sorted one by applying the functor $\overline{(-)}$, and thus blending the underlying languages. Nevertheless, we do not lose any semantical information if we consider the category of algebras over SG and $\overline{SG}$.

Theorem 3.7 *There is a natural isomorphism between the category of multi-language algebras over SG and the category of order-sorted algebras over the associated signature $\overline{SG}$*

$$\eta\colon MLAlg(\mathcal{C}, \mathbb{I}) \Rightarrow OSAlg(\mathcal{C}, \mathbb{I}) \circ \overline{(-)} \qquad \textit{inducing} \qquad \mathcal{MLAlg}(\mathcal{C}, \mathbb{I})_{SG} \cong \mathcal{OSAlg}(\mathcal{C}, \mathbb{I})_{\overline{SG}}$$

where there are functors $MLAlg(\mathcal{C}, \mathbb{I})\colon \mathcal{MLSg} \to \mathcal{Cat}^{\mathrm{op}}$ and $OSAlg(\mathcal{C}, \mathbb{I})\colon \mathcal{OSSg} \to \mathcal{Cat}^{\mathrm{op}}$ that map signatures to their category of algebras in $(\mathcal{C}, \mathbb{I})$.

Proof. The functors are defined on objects by $MLAlg(\mathcal{C}, \mathbb{I})(SG) \triangleq \mathcal{MLAlg}(\mathcal{C}, \mathbb{I})_{SG}$ and $OSAlg(\mathcal{C}, \mathbb{I})(Sg) \triangleq \mathcal{OSAlg}(\mathcal{C}, \mathbb{I})_{Sg}$. Now, let $SG \triangleq (Sg_1, Sg_2, \ltimes)$ and $SG' \triangleq (Sg'_1, Sg'_2, \ltimes')$, and let $H \triangleq (h_1, h_2)\colon SG \to SG'$ be a signature morphism between them. We shall define a functor $H^*\colon \mathcal{MLAlg}(\mathcal{C}, \mathbb{I})_{SG'} \to \mathcal{MLAlg}(\mathcal{C}, \mathbb{I})_{SG}$ and let $MLAlg(\mathcal{C}, \mathbb{I})(H) \triangleq H^*$. Let $\mathbf{A}$ be an SG'-algebra in $\mathcal{MLAlg}(\mathcal{C}, \mathbb{I})_{SG'}$. Then, the multi-language SG-algebra $H^*\mathbf{A}$ is defined as follows:

- We define its order-sorted components $(H^*\mathbf{A})_1$ and $(H^*\mathbf{A})_2$. Let $i \triangleq 1, 2$:
  - the interpretation of sorts is given by $[\![s]\!]_{(H^*\mathbf{A})_i} \triangleq [\![h_i(s)]\!]_{\mathbf{A}_i}$ for each $s \in S_i$;
  - given the function symbol $f\colon w \to s$ in Sg_i, we define $[\![f\colon w \to s]\!]_{(H^*\mathbf{A})_i} \triangleq [\![h_i(f)\colon h_i(w) \to h_i(s)]\!]_{\mathbf{A}_i}$;
  - the constant symbol $k\colon s$ in Sg_i is interpreted by letting $[\![k]\!]_{(H^*\mathbf{A})_i} \triangleq [\![h_i(k)]\!]_{\mathbf{A}_i}$; and
  - $[\![s \leq_i r]\!]_{(H^*\mathbf{A})_i} \triangleq [\![h_i(s) \leq'_i h_i(r)]\!]_{\mathbf{A}_i}$ for each subsort constraint $s \leq_i r$ in Sg_i.

  The fact that $(H^*\mathbf{A})_i$ is a proper order-sorted Sg_i-algebra is ensured by the properties of the (multi-language) signature morphism H.
- Boundary morphisms are defined by $[\![s_i \ltimes s'_j]\!]_{H^*\mathbf{A}} \triangleq [\![h_i(s) \ltimes' h_j(s')]\!]_{\mathbf{A}}$ for each constraint $s_i \ltimes s'_j$ in SG.

In order to define the action of H^* on homomorphisms, suppose that $g\colon \mathbf{A} \to \mathbf{B}$ is a multi-language SG'-homomorphism in $\mathcal{MLAlg}(\mathcal{C}, \mathbb{I})_{SG'}$. Then, $(H^*g)_i\colon (H^*\mathbf{A})_i \to (H^*\mathbf{B})_i$ is defined by the S_i-sorted morphisms

- $\big((H^*g)_i\big)_s \triangleq (g_i)_{h_i(s)}$, which is well-defined since $g_i\colon \mathbf{A}_i \to \mathbf{B}_i$ is an order-sorted Sg'_i-homomorphism and $[\![h_i(s)]\!]_{\mathbf{A}_i} = [\![s]\!]_{(H^*\mathbf{A})_i}$.

The commutativity of the diagram in Definition 3.6 is given by a tedious but simple diagram chase.

$$\begin{tikzcd}
{[\![h_i(s)]\!]_{\mathbf{A}_i}} \arrow[rrr, "{[\![h_i(s)\ltimes' h_j(s')]\!]_{\mathbf{A}}}"'] \arrow[dr, equal] \arrow[ddd, "(g_i)_{h_i(s)}"] & & & {[\![h_j(s')]\!]_{\mathbf{A}_j}} \arrow[dl, equal] \arrow[ddd, "(g_j)_{h_j(s')}"'] \\
& {[\![s]\!]_{(H^*\mathbf{A})_i}} \arrow[r, "{[\![s_i \ltimes s'_j]\!]_{H^*\mathbf{A}}}"] \arrow[d, "\left((H^*g)_i\right)_s"'] & {[\![s']\!]_{(H^*\mathbf{A})_j}} \arrow[d, "\left((H^*g)_j\right)_{s'}"] & \\
& {[\![s]\!]_{(H^*\mathbf{B})_i}} \arrow[r, "{[\![s_i \ltimes s'_j]\!]_{H^*\mathbf{B}}}"'] \arrow[dl, equal] & {[\![s']\!]_{(H^*\mathbf{B})_j}} \arrow[dr, equal] & \\
{[\![h_i(s)]\!]_{\mathbf{B}_i}} \arrow[rrr, "{[\![h_i(s)\ltimes' h_j(s')]\!]_{\mathbf{B}}}"] & & & {[\![h_j(s')]\!]_{\mathbf{B}_j}}
\end{tikzcd}$$

We next define a functor $h^*\colon \mathcal{OSAlg}(\mathcal{C},\mathbb{I})_{Sg_2} \to \mathcal{OSAlg}((\mathcal{C},\mathbb{I}))_{Sg_1}$ and set $\mathit{OSAlg}(\mathcal{C},\mathbb{I})(h) \triangleq h^*$, where $h\colon Sg_1 \to Sg_2$ is an order-sorted signature morphism. This is similar to the definition of H^* above. First pick any (order-sorted) Sg_2-algebra $\mathbf{A}$. We need to define the order-sorted Sg_1-algebra $h^*\mathbf{A}$. We define

- objects $[\![s \in S_1]\!]_{h^*\mathbf{A}} \triangleq [\![h(s)]\!]_{\mathbf{A}}$ in $\mathcal{C}$ and hence $[\![w]\!]_{h^*\mathbf{A}} \triangleq [\![h(s_1)]\!]_{\mathbf{A}} \times \cdots \times [\![h(s_n)]\!]_{\mathbf{A}}$ for $w \triangleq s_1,\ldots,s_n \in S_1^n$;
- morphisms $[\![f\colon w \to s \in Sg_1]\!]_{h^*\mathbf{A}} \triangleq [\![h(f)\colon h(w) \to h(s) \in Sg_2]\!]_{\mathbf{A}}\colon [\![w]\!]_{h^*\mathbf{A}} \to [\![s]\!]_{h^*\mathbf{A}}$ and morphisms $[\![k \in Sg_1]\!]_{h^*\mathbf{A}} \triangleq [\![h(k)]\!]_{\mathbf{A}}\colon 1 \to [\![s]\!]_{h^*\mathbf{A}}$; and
- a morphism $[\![s \le_1 r \in S_1]\!]_{h^*\mathbf{A}} \triangleq [\![h(s) \le_2 h(r)]\!]_{\mathbf{A}}\colon [\![s]\!]_{h^*\mathbf{A}} \rightarrowtail [\![r]\!]_{h*\mathbf{A}}$ in $\mathbb{I}$.

We omit the verification that semantics of operators commutes with the semantics of subsorting, although this is essentially immediate since $\mathbf{A}$ is an Sg_2-algebra. Now let $g\colon \mathbf{A} \to \mathbf{B}$ be an order-sorted Sg_2-homomorphism. We define the Sg_1-homomorphism $h^*(g)\colon h^*\mathbf{A} \to h^*\mathbf{B}$ by setting the components to be $(h^*(g))_{s\in S_1} \triangleq g_{h(s)}\colon [\![h(s)]\!]_{\mathbf{A}} \to [\![h(s)]\!]_{\mathbf{B}}$.

Now we define the natural transformation η by specifying the components $\eta_{SG}\colon \mathcal{MLAlg}(\mathcal{C},\mathbb{I})_{SG} \to \mathcal{OSAlg}(\mathcal{C},\mathbb{I})_{\overline{SG}}$. Pick any SG-homomorphism $h\colon \mathbf{A} \to \mathbf{B}$. First we define the (order-sorted) $\overline{SG}$-algebra $\eta_{SG}\mathbf{A}$ by setting

- $[\![s_i \in S_1 + S_2]\!]_{\eta_{SG}\mathbf{A}} \triangleq [\![s]\!]_{\mathbf{A}_i}$ in $\mathcal{C}$ and $[\![w]\!]_{\eta_{SG}\mathbf{A}} \triangleq [\![s_1]\!]_{\mathbf{A}_i} \times \cdots \times [\![s_n]\!]_{\mathbf{A}_i}$ for each $w \triangleq s_1,\ldots,s_n \in (S_1+S_2)^n$;
- morphisms $[\![f_i\colon w_i \to s_i]\!]_{\eta_{SG}\mathbf{A}} \triangleq [\![f\colon w \to s]\!]_{\mathbf{A}_i}$ and $[\![k_i]\!]_{\eta_{SG}\mathbf{A}} \triangleq [\![k]\!]_{\mathbf{A}_i}\colon 1 \to [\![s]\!]_{\mathbf{A}_i}$; and
- $[\![\hookrightarrow_{s_i,s'_j}\colon s_i \to s'_j]\!]_{\eta_{SG}\mathbf{A}} \triangleq [\![s_i \ltimes s'_j]\!]_{\mathbf{A}}\colon [\![s]\!]_{\mathbf{A}_i} \to [\![s']\!]_{\mathbf{A}_j}$
- $[\![s_i \le r_j]\!]_{\eta_{SG}\mathbf{A}} \triangleq [\![s \le_i r]\!]_{\mathbf{A}_i}\colon [\![s]\!]_{\mathbf{A}_i} \rightarrowtail [\![r]\!]_{\mathbf{A}_i}$ in $\mathbb{I}$ for each $s_i \le r_j$ in $S_1 + S_2$ (don't forget that $i = j$).

where the required commutation properties follow immediately since the $\mathbf{A}_i$ are Sg_i-algebras. Second we define $\overline{SG}$-homomorphism $\eta_{SG}(h)\colon \eta_{SG}\mathbf{A} \to \eta_{SG}\mathbf{B}$. Since by the definition of h there are Sg_i-homomorphisms $h_i\colon \mathbf{A}_i \to \mathbf{B}_i$, we can define $\eta_{SG}(h)_{s_i \in S_1+S_2} \triangleq (h_i)_s\colon [\![s]\!]_{\mathbf{A}_i} \to [\![s]\!]_{\mathbf{B}_i}$, and $\eta_{SG}(h)_{s_i}$ inherits the required commuting properties from h (commuting with $\hookrightarrow_{s_i,s'_j}$), and from the h_i (commuting with the f_i).

One can see that η_{SG} is an isomorphism by reversing the construction and by noting there is a one-one correspondence between the sorts, operators, and subsort constraints in SG and those in Sg_1 and Sg_2. This is straightforward. More difficult is naturality of η which we show next.

Let $H \triangleq (h_1,h_2)\colon SG \triangleq (Sg_1,Sg_2,\ltimes) \to SG' \triangleq (Sg'_1,Sg'_2,\ltimes')$. Then we need to show that the diagram below commutes.

$$\begin{tikzcd}
\mathcal{MLAlg}(\mathcal{C},\mathbb{I})_{SG} \arrow[r, "\eta_{SG}"] & \mathcal{OSAlg}(\mathcal{C},\mathbb{I})_{\overline{SG}} \\
\mathcal{MLAlg}(\mathcal{C},\mathbb{I})_{SG'} \arrow[r, "\eta_{SG'}"] \arrow[u, "H^*"] & \mathcal{OSAlg}(\mathcal{C},\mathbb{I})_{\overline{SG'}} \arrow[u, "\overline{H}^*"']
\end{tikzcd}$$

Pick any morphism $g\colon \mathbf{A} \to \mathbf{B}$ in $\mathcal{MLAlg}(\mathcal{C},\mathbb{I})_{SG'}$. First we need to show that $\eta_{SG}(H^*\mathbf{A}) = \overline{H}^*(\eta_{SG'}\mathbf{A})$. Let us check only that these $\overline{SG}$-algebras provide equal meaning to sorts $s_i \in S_1 + S_2$ where we have $(h_i(s))_i \in S'_1 + S'_2$.

$$[\![s_i]\!]_{\eta_{SG}(H^*\mathbf{A})} = [\![s \in S_i]\!]_{(H^*\mathbf{A})_i} = [\![h_i(s) \in S'_i]\!]_{\mathbf{A}_i} = [\![(h_i(s))_i \in S'_1 + S'_2]\!]_{\eta_{SG'}\mathbf{A}} = [\![\overline{H}(s_i)]\!]_{\eta_{SG'}\mathbf{A}} = [\![s_i]\!]_{\overline{H}^*(\eta_{SG'}\mathbf{A})}$$

Now g furnishes us with SG'-homomorphisms $g_i \colon \mathbf{A}_i \to \mathbf{B}_i$, and we need to show that

$$\eta_{SG}(H^* g) = \overline{H}^*(\eta_{SG'} g) \colon [\![h_i(s) \in S'_i]\!]_{\mathbf{A}_i} \to [\![h_i(s) \in S'_i]\!]_{\mathbf{B}_i}$$

is an equality of $\overline{SG}$-homomorphisms. But this follows from the following calculation on components of $S_1 + S_2$-sorted morphisms

$$(\eta_{SG}(H^* g))_{s_i} = ((H^* g)_i)_s = (g_i)_{h_i(s)} = (\eta_{SG'} g)_{(h_i(s))_i} = (\eta_{SG'} g)_{(\overline{H}(s_i))} = (\overline{H}^*(\eta_{SG'} g))_{s_i}$$

and the proof is completed. □

Running Example. The multi-language semantics of the term introduced in the previous example is given by the algebra $\eta_{SG}\mathbf{A}$ which leads to $[\![[c\colon \mathtt{chr}, n\colon \mathtt{nat}] \vdash \hookrightarrow_{\mathtt{chr},\mathtt{nat}} (c) + n\colon \mathtt{nat}]\!]_{\eta_{SG}\mathbf{A}} = (c, n) \mapsto n{+}1\colon \mathtt{Char} \times \mathbb{N} \to \mathbb{N}$.

3.2 Equational Reasoning in a Multi-Language Context

In this section we define multi-language proved terms, and give them a semantics. Then we define multi-language equations and semantic satisfaction. From this we can define theories and models, and hence prove soundness and completeness.

Let $SG \triangleq (Sg_1, Sg_2, \ltimes)$ be a multi-language signature. A **(multi-language) proved term** $\Gamma \vdash t\colon s_i$ is a proved term over the associated signature $\overline{SG}$. It follows that if $\Gamma \vdash t\colon s$ is a proved term over Sg_i, then $\underline{\Gamma \vdash t\colon s}$ is a proved term in $\overline{SG}$, where $\underline{\Gamma \vdash t\colon s} \triangleq \underline{\Gamma} \vdash \underline{t}\colon \underline{s}$ and

- $\underline{s} \triangleq s_i$ for each $s \in S_i$; and $\underline{\Gamma} \triangleq [x_1\colon \underline{s_1}, \dots, x_n\colon \underline{s_n}]$ for each context $\Gamma \triangleq [x_1\colon s_1, \dots, x_n\colon s_n]$ over Sg_i;
- $\underline{t}$ is recursively defined over the syntax of raw terms generated by Sg_i: $\underline{x} \triangleq x$; $\underline{k} \triangleq k_i$; and $\underline{f(t_1, \dots, t_a)} \triangleq f_i(\underline{t_1}, \dots \underline{t_n})$.

Due to the injectivity of this construction, we shall refer to it as the *inclusion* of an order-sorted term into the multi-language, and we informally say that a multi-language "contains" the underlying languages. Furthermore, the definition of multi-language terms also includes *hybrid* terms that are not the result of the inclusion of an order-sorted term but which are constructed using the conversion operators in the associated signature.

Given a multi-language SG-algebra $\mathbf{A}$, the categorical **semantics of a (multi-language) term** $\Gamma \vdash t\colon s_i$ is the order-sorted semantics of $\Gamma \vdash t\colon s_i$ induced by $\eta_{SG}\mathbf{A}$, namely $[\![\Gamma \vdash t\colon s_i]\!]_{\mathbf{A}} \triangleq [\![\Gamma \vdash t\colon s_i]\!]_{\eta_{SG}\mathbf{A}}$. As expected, a multi-language preserves the semantics of the underlying terms:

Proposition 3.8 *Let $\mathbf{A}$ be a multi-language SG-algebra over $SG \triangleq (Sg_1, Sg_2, \ltimes)$. If $\Gamma \vdash t\colon s$ is a proved term over Sg_i, then $[\![\underline{\Gamma \vdash t\colon s}]\!]_{\mathbf{A}} = [\![\underline{\Gamma \vdash t\colon s}]\!]_{\eta_{SG}\mathbf{A}} = [\![\Gamma \vdash t\colon s]\!]_{\mathbf{A}_i}$.*

Regularity and coherence for a multi-language signature $SG \triangleq (Sg_1, Sg_2, \ltimes)$ are defined with respect to its associated signature. That is, SG is **regular** (resp., **coherent**) if $\overline{SG}$ is regular (resp., coherent). It is immediate that SG is regular (resp., coherent) if and only if Sg_1 and Sg_2 are regular (resp., coherent).

Definition 3.9 (Multi-Language Equation and Satisfaction) *Let SG be a coherent multi-language signature. A (conditional)* ***equation*** *for SG is an order-sorted (conditional) equation over $\overline{SG}$. A multi-language algebra $\mathbf{A}$* ***satisfies*** *any such (conditional) equation if the (conditional) equation is satisfied by $\eta_{SG}\mathbf{A}$.*

An immediate consequence of Proposition 3.8 is that every Sg_i-equation satisfied by $\mathbf{A}_i$ is also satisfied by the multi-language algebra $\mathbf{A}$ (in its inclusion form provided by the mapping $\underline{(-)}$).

A **multi-language theory** $TH \triangleq (SG, AX)$ is a pair of a multi-language signature SG and a set of (conditional) multi-language equations AX over SG, namely the **axioms** of the theory. The **theorems** of TH are the equations $\Gamma \vdash t = t'\colon s_i$ derivable from $(\overline{SG}, AX)$. A multi-language SG-algebra that satisfies all the axioms in AX is said a **model** of TH, and $\mathcal{MLMod}(\mathcal{C}, \mathbb{I})_{TH}$ denotes the full subcategory of models of $\mathcal{MLAlg}(\mathcal{C}, \mathbb{I})_{TH}$. We now introduce the categories of theories in order to define the *associated theory* of a multi-language theory. From now on, when we write order-sorted theories Th_1, Th_2, Th, Th', etc., we assume they are defined as $Th_1 \triangleq (Sg_1, Ax_1)$, $Th_2 \triangleq (Sg_2, Ax_2)$, $Th \triangleq (Sg, Ax)$, $Th' \triangleq (Sg', Ax')$, etc., respectively.

Running Example. Let $Th_1 \triangleq (Sg_1, Ax_1)$ and $Th_2 \triangleq (Sg_2, Ax_2)$ be the order-sorted theories over Sg_1 and Sg_2 axiomatized by the equations provided in Figure 6. We can generate from Ax_1 and Ax_2 a set AX of multi-language equations by applying $\underline{(-)}$ to each equation. For instance, $(eq_{1,1}) \triangleq [n\colon \mathtt{nat}] \vdash \mathtt{0} + n = n\colon \mathtt{nat}$

$(eq_{1,1})$	$[n\colon \mathtt{nat}] \vdash \mathtt{0} \;\mathtt{+}\; n = n\colon \mathtt{nat}$	$(eq_{2,1})$	$\vdash \mathtt{next(a)} = \mathtt{b}\colon \mathtt{chr}$
$(eq_{1,2})$	$[n\colon \mathtt{nat}, m\colon \mathtt{nat}] \vdash \mathtt{s}(n) \;\mathtt{+}\; m = \mathtt{s}(n \;\mathtt{+}\; m)\colon \mathtt{nat}$	$(eq_{2,\dots})$	$\vdash \ldots = \ldots\colon \mathtt{chr}$
$(eq_{1,3})$	$[n\colon \mathtt{nat}, m\colon \mathtt{nat}] \vdash n \;\mathtt{+}\; m = m \;\mathtt{+}\; n\colon \mathtt{nat}$	$(eq_{2,26})$	$\vdash \mathtt{next(z)} = \mathtt{a}\colon \mathtt{chr}$

(a) Th_1 axioms. (b) Th_2 axioms.

Fig. 6. Axioms of order-sorted theories Th_1 and Th_2.

becomes $\overline{(eq_{1,1})} \triangleq [n\colon \mathtt{nat}] \vdash \mathtt{0} \;\mathtt{+}\; n = n\colon \mathtt{nat}$. Note that a substantial change occurs when mapping an order-sorted equation to a multi-language one. Consider again $(eq_{1,1})$. A substitution in the order-sorted world can only plug t, where $\Gamma \vdash t\colon \mathtt{nat}$ in Sg_1, into the variable $n\colon \mathtt{nat}$. However, a multi-language substitution can substitute any t', where $\Gamma' \vdash t'\colon \mathtt{nat}$ in $\overline{SG}$, for $n\colon \mathtt{nat}$ in the lifted equation $\overline{(eq_{1,1})}$—including, crucially, the possibility that t' is a hybrid multi-language term.

The behaviour of boundary morphisms can be axiomatized by adding the following equations to AX:

$$
\begin{array}{lr}
(EQ_1) & \vdash \hookrightarrow_{\mathtt{nat},\mathtt{chr}} (\mathtt{0}) = \mathtt{a}\colon \mathtt{chr}\\
(EQ_2) & \vdash \hookrightarrow_{\mathtt{nat},\mathtt{str}} (\mathtt{0}) = \mathtt{a}\colon \mathtt{str}\\
(EQ_3) & [c\colon \mathtt{chr}] \vdash \hookrightarrow_{\mathtt{chr},\mathtt{nat}} (c) = \mathtt{s(0)}\colon \mathtt{nat}\\
(EQ_4) & [c\colon \mathtt{chr}] \vdash \hookrightarrow_{\mathtt{str},\mathtt{nat}} (c) = \mathtt{s(0)}\colon \mathtt{nat}\\
(EQ_5) & [n\colon \mathtt{nat}] \vdash \hookrightarrow_{\mathtt{nat},\mathtt{chr}} (\mathtt{s}(n)) = \mathtt{next}(\hookrightarrow_{\mathtt{nat},\mathtt{chr}} (n))\colon \mathtt{chr}\\
(EQ_6) & [n\colon \mathtt{nat}] \vdash \hookrightarrow_{\mathtt{nat},\mathtt{str}} (\mathtt{s}(n)) = \mathtt{next}(\hookrightarrow_{\mathtt{nat},\mathtt{str}} (n))\colon \mathtt{str}\\
(EQ_7) & [s\colon \mathtt{str}, v\colon \mathtt{str}] \vdash \hookrightarrow_{\mathtt{str},\mathtt{nat}} (s \;\mathtt{+}\; v) = \hookrightarrow_{\mathtt{str},\mathtt{nat}} (s) \;\mathtt{+}\; \hookrightarrow_{\mathtt{str},\mathtt{nat}} (v)\colon \mathtt{nat}
\end{array}
$$

Definition 3.10 *Let $\mathcal{OSTh}$ be the* ***category of order-sorted theories*** *whose morphisms $h\colon Th_1 \to Th_2$ are signature morphisms $h\colon Sg_1 \to Sg_2$ in $\mathcal{OSSg}$ that preserve theorems, that is, if $\Gamma \vdash t = t'\colon s_i$ is a theorem of Th_1 with $\Gamma \triangleq [x_1\colon s_1, \ldots, x_n\colon s_n]$, then $\Gamma' \vdash h(t) = h(t')\colon h(s_i)$ is a theorem of Th_2, where $\Gamma' \triangleq [x_1\colon h(s_1), \ldots, x_n\colon h(s_n)]$ and $h(t)$ and $h(t')$ are inductively defined over the syntax according to the action of h on function symbols and constants.*

The ***category of multi-language theories*** *is denoted by $\mathcal{MLTh}$ and a theory morphism $H\colon (SG_1, AX_1) \to (SG_2, AX_2)$ is a signature morphism $H\colon SG_1 \to SG_2$ in $\mathcal{MLSg}$ such that if $\Gamma \vdash t = t'\colon s_i$ is a theorem of (SG_1, AX_1) with $\Gamma \triangleq [x_1\colon s_1, \ldots, x_n\colon s_n]$, then $\Gamma' \vdash \overline{H}(t) = \overline{H}(t')\colon \overline{H}(s_i)$ is a theorem of (SG_2, AX_2), where $\Gamma' \triangleq [x_1\colon \overline{H}(s_1), \ldots, x_n\colon \overline{H}(s_n)]$.*

Functors $MLAlg(\mathcal{C}, \mathbb{I})$ and $OSAlg(\mathcal{C}, \mathbb{I})$ can be easily extended to $MLMod(\mathcal{C}, \mathbb{I})\colon \mathcal{MLTh} \to \mathcal{C}at^{\mathrm{op}}$ and $OSMod(\mathcal{C}, \mathbb{I})\colon \mathcal{OSTh} \to \mathcal{C}at^{\mathrm{op}}$, respectively, such that they associate to each signature its corresponding category of models. Then, $\overline{(-)}\colon \mathcal{MLTh} \to \mathcal{OSTh}$ is defined by $\overline{TH} \triangleq (\overline{SG}, AX)$ on objects and by $\overline{H}$ on morphisms $H\colon TH_1 \to TH_2$.

Proposition 3.11 *$MLMod(\mathcal{C}, \mathbb{I})$ and $OSMod(\mathcal{C}, \mathbb{I}) \circ \overline{(-)}$ are isomorphic functors. Let η be the natural isomorphism between them and TH a multi-language theory. Then, η_{TH} is the isomorphism between categories $\mathcal{MLMod}(\mathcal{C}, \mathbb{I})_{TH}$ and $\mathcal{OSMod}(\mathcal{C}, \mathbb{I})_{TH}$.*

Theorem 3.12 (Soundness and Completeness) *Let TH be a multi-language theory. $\Gamma \vdash t = t'\colon s$ is a theorem of TH if and only if $\Gamma \vdash t = t'\colon s$ is satisfied by every model of TH.*

3.3 An Extended Example

The Running Example has the sole purpose of illustrating our theory in an elementary way: we are very much aware of its limitations. Here we give a taste of a more realistic example. For space reasons, we can convey only the main ideas: full details are in an extended version of the paper citearxiv-version-of-this-paper.

We define a new multi-language by blending a simple functional core with a minimal imperative language. The former is of course suited to writing programs that are easier to reason about, whereas the latter provides a more straightforward procedural and low-level approach to software development. We formalise the *simply-typed lambda calculus* and a simple *imperative language* as two equational theories, and we blend them together in order to provide the gist of an interoperability between the functional and imperative paradigms. More complex examples can be built along the lines of the one presented here.

We assume Imp and $\lambda^{\to}$ to be the signatures of a small imperative language and the simply-typed lambda-calculus, respectively. In the following, we use colours blue for denoting Imp code and red for $\lambda^{\to}$-terms. The interoperability we wish to provide should allow the use of $\lambda^{\rightarrow}$-terms as Imp-expressions and vice versa. For instance, we would like to write multi-language programs such as `x = (λy:int . y + y) 1`, which encodes the assignment to the variable `x` of the value obtained by applying the $\lambda^{\rightarrow}$-function `(λy:int . y + y)` to the Imp-numeral `1`. Although minimal, its interpretation requires several applications of the boundary functions: First, we need to compute the result of the function application, which in turn needs the evaluation of `1` as a $\lambda^{\rightarrow}$-term. Then, the resulting term has to be converted back to an Imp-numeral in order to be assigned to `x`.

The multi-language signature $\lambda^{\rightarrow}$-Imp providing the desired interoperability is given by coupling the signatures Imp and $\lambda^{\rightarrow}$ with the join relation specified by $\mathbb{e} \ltimes \mathbb{exp}$ and $\mathbb{exp} \ltimes \mathbb{e}$, where $\mathbb{e}$ is the sort of Imp-expressions and $\mathbb{exp}$ the sort of $\lambda^{\rightarrow}$-terms. The semantics of the generated multi-language programs is obtained by introducing a boundary function for each $\ltimes$-constraint. For instance, given a standard denotational semantics for both the underlying languages, the boundary function $[\![\mathbb{e} \ltimes \mathbb{exp}]\!]$ can provide each Imp-expression with a $\lambda^{\rightarrow}$-meaning in the following way: Let e be the semantics of such an Imp-expression. We can first transform a $\lambda^{\rightarrow}$-environment to an Imp-environment, run e on its conversion, and then move the resulting Imp-values to suitable $\lambda^{\rightarrow}$-values.

The equational axiomatization of such a boundary function can be specified by the following multi-language equations: (1) $\hookrightarrow_{\mathbb{e},\mathbb{exp}}(i) = i$ and (2) $\hookrightarrow_{\mathbb{e},\mathbb{exp}}(x) = x$. The first equation allows $\lambda^{\rightarrow}$-integers to be converted to Imp-numerals of the same form. In more realistic examples, the conversion of values across languages should take into consideration different machine representations (for instance, if the $\lambda^{\rightarrow}$ language does not admit an explicit representation of integers, we may convert the integer i to its corresponding Church-numeral). Equation (2) provides a match between Imp and $\lambda^{\rightarrow}$ variables with the same name. This enables a natural way for moving stored values across the two languages. For instance, the multi-language program `x = (λy:int . y + y) z` acts in the same way of the previously described one but applying the $\lambda^{\rightarrow}$-function to the value stored in the Imp-variable `z`.

On the other hand, the boundary function $[\![\mathbb{exp} \ltimes \mathbb{e}]\!]$ works in a dual manner for providing $\lambda^{\rightarrow}$-terms with an Imp-meaning. Given all these specifications, the equational logic provides the following chain of equalities:

$$\texttt{x = (}\lambda\texttt{y:int . y + y) 1} \quad = \quad \texttt{x = 1 + 1} \quad = \quad \texttt{x = 2} \quad = \quad \texttt{x = 2}$$

4 Further Multi-Language Constructions

Buro and Mastroeni [3] provides three different multi-language constructions based on boundary morphism properties (although in their work, morphisms are only set-theoretic functions). In Section 3, we studied a categorical equational logic for the simplest construction. Here we briefly discuss the other two, each a refinement of the first.

The first refinement of multi-language signatures is accomplished by allowing all conversion operators $\hookrightarrow_{s_i,s'_j}: s_i \to s'_j$ in the associated signature to be replaced by subsort polymorphic operators $\hookrightarrow: s_i \to s'_j$ that *do not carry* any sort information. One can check that any associated signature $\overline{SG}$ defined in this way remains an order-sorted signature if and only if the following additional constraint holds for SG:

$$s_i \ltimes s'_j,\ r_i \ltimes r'_j, \text{ and } s_i \leq_i r_i \text{ imply } s'_j \leq_j r'_j \tag{1}$$

Multi-language algebras are then restricted by the following monotonicity requirement:

$$s_i \ltimes s'_j,\ r_i \ltimes r'_j, \text{ and } s_i \leq_i r_i \text{ imply } [\![s' \leq_j r']\!]_{\mathbf{A}_j} \circ [\![s_i \ltimes s'_j]\!]_{\mathbf{A}} = [\![r_i \ltimes r'_j]\!]_{\mathbf{A}} \circ [\![s' \leq_j r']\!]_{\mathbf{A}_j} \tag{2}$$

In this new multi-language construction, we can prove the following version of Theorem 3.7:

Theorem 4.1 *Assume (1) and (2) for multi-language signatures and algebras, respectively. There is a natural isomorphism $\eta\colon MLAlg(\mathcal{C},\mathbb{I}) \Rightarrow OSAlg(\mathcal{C},\mathbb{I}) \circ \overline{(-)}$ inducing $\mathcal{MLA}lg(\mathcal{C},\mathbb{I})_{SG} \cong \mathcal{OSA}lg(\mathcal{C},\mathbb{I})_{\overline{SG}}$, where there are functors $MLAlg(\mathcal{C},\mathbb{I})\colon \mathcal{MLS}g \to \mathcal{C}at^{\mathrm{op}}$ and $OSAlg(\mathcal{C},\mathbb{I})\colon \mathcal{OSS}g \to \mathcal{C}at^{\mathrm{op}}$ that map signatures to their category of algebras in $(\mathcal{C},\mathbb{I})$.*

Proof. The proof is almost identical to the proof of Theorem 3.7. That each $\eta_{SG}\mathbf{A}$ is a proper order-sorted algebra boils down to the fact that each $[\![\hookrightarrow: s_i \to s'_j]\!]_{\eta_{SG}\mathbf{A}}$ commutes with the desired morphisms in $\mathbb{I}$; but this commutativity follows immediately from (2). ◻

The second refinement of multi-language signatures aims to achieve a multi-language construction which consists only of the union of the underlying languages, that is no conversion operator is added to the associated signature and single-language operators are not tagged. Such a construction is particularly useful when mode ling the extension of a language rather than the union of two already existing languages.

The notion of multi-language signature is refined by assuming that

- $(S_1 + S_2, \ltimes)$ is a poset; and
- $f\colon w \to s$ in Sg_i and $f\colon w' \to s'$ in Sg_j with $w_i \ltimes w'_j$, then $s_i \ltimes s'_j$.

and the associated signature $\overline{SG}$ is defined as follows:

- the poset of sorts is given by $(S_1 + S_2, \leq)$, where $s_i \leq r_j$ if $i = j$ and $s \leq_i r$ or $i \neq j$ and $s_i \ltimes r_j$;
- if $f\colon w \to s$ is a function symbol in Sg_i, then $f\colon w_i \to s_i$ is a function symbol in $\overline{SG}$, and similarly for constants.

Multi-language algebras now force boundary morphisms to act as subsort morphisms. This means that if the function symbol f appears with more than one rank $f\colon w \to s$ and $f\colon w' \to r$ in Sg_i and Sg_j, respectively, with $(s_1)_i, \dots, (s_a)_i \triangleq w \ltimes w' \triangleq (r_1)_j, \dots, (r_a)_j$, then the following diagram commutes:

$$\begin{tikzcd}
\llbracket s_1 \rrbracket_{\mathbf{A}_i} \times \cdots \times \llbracket s_a \rrbracket_{\mathbf{A}_i} \arrow[r, "\llbracket f\colon w_1 \to s \rrbracket_{\mathbf{A}_i}"] \arrow[d, "\llbracket (s_1)_i \ltimes (r_1)_j \rrbracket_{\mathbf{A}} \times \cdots \times \llbracket (s_a)_i \ltimes (r_a)_j \rrbracket_{\mathbf{A}}"'] & \llbracket s \rrbracket_{\mathbf{A}_i} \arrow[d, "\llbracket s_i \ltimes r_j \rrbracket_{\mathbf{A}}"] \\
\llbracket r_1 \rrbracket_{\mathbf{A}_j} \times \cdots \times \llbracket r_a \rrbracket_{\mathbf{A}_j} \arrow[r, "\llbracket f\colon w_2 \to r \rrbracket_{\mathbf{A}_j}"] & \llbracket r \rrbracket_{\mathbf{A}_j}
\end{tikzcd}$$

Theorem 4.2 *Assume these new hypotheses for multi-language signatures and algebras, respectively. There is a natural isomorphism* $\eta\colon MLAlg(\mathcal{C}, \mathbb{I}) \Rightarrow OSAlg(\mathcal{C}, \mathbb{I}) \circ \overline{(-)}$ *inducing* $\mathcal{MLAlg}(\mathcal{C}, \mathbb{I})_{SG} \cong \mathcal{OSAlg}(\mathcal{C}, \mathbb{I})_{\overline{SG}}$*, where (as before) there are functors* $MLAlg(\mathcal{C}, \mathbb{I})\colon \mathcal{MLSg} \to \mathcal{C}at^{\mathrm{op}}$ *and* $OSAlg(\mathcal{C}, \mathbb{I})\colon \mathcal{OSSg} \to \mathcal{C}at^{\mathrm{op}}$.

Conclusions and Future Research

Equational logic is a simple fragment of first-order logic with several applications to computer science [12,34,17]. In this paper, we have addressed the problem of equational deduction in a multi-language context. We have lifted the order-sorted equational logic of [13] to the algebraic framework of multi-languages introduced by [3], and we have proved the soundness and the completeness of the resulting deduction system. The main benefit of the theoretical development in this paper is a solid mathematical foundation for reasoning about equalities in a multi-language context.

Among all the applications, one motivation for extending the theory of equational logic to a multi-language context resides in the possibility of providing *operational semantics* to multi-languages, in a similar way to [11]. In future work, we plan to investigate this in the context of *rewriting logic* [31], where axioms might be partitioned into a set R of *rewriting rules* and a set E of equations in order to perform *rewriting modulo* E.

We know that there is considerable practical interest in understanding how real languages and systems may be integrated to exploit advantages of each individual system. To make real progress, we believe that practical advances need to be made in synchrony with theoretical developments, with each approach supporting and informing the other. To this end, we are pursuing practical developments of the work presented in this paper. As a side note, we have also begun to look at the implementation of our examples within Maude [5].

We deduce *unconditional* equations but allow *conditional* axioms. This approach has merit from the point of view of practical specifications, and reasoning about them. That said, one could be rather more expressive if one allows conditional equations as primary judgements of a deduction system. In such a case, the semantics of judgements could be given in an internal manner by making use of categories with equalisers [19]. We are currently working on such a system, with a view to giving a sound and complete semantics, and the results will appear in a future paper. There are interesting questions concerning the appropriate category theory, and the answers will have connections to work such as [25]. And further, since equational theories give rise to *free algebra monads* [30], further studies should investigate the possibility of extending/generalizing the results in this paper to the notion of *monad* [23]. Here, however, our intention has been to provide an account that is very general (categorical) but not so abstract that applications become obfuscated.

Finally, we wish to note that the approach presented in this paper generalises to the combination of an arbitrary number n of languages by recursively combining the (associated theory of the) first $Th_1, \dots, Th_{n-1}$ theories with Th_n. Such a modularity property strengthens the framework both from a theoretical and practical perspective, enabling the construction of complex theories on the basis of more elementary ones.

References

[1] Amal Ahmed and Matthias Blume. An equivalence-preserving CPS translation via multi-language semantics. In *Proceedings of the 16th ACM SIGPLAN International Conference on Functional Programming*, volume 46, pages 431–444, New York, NY, USA, 2011. ACM.

[2] Garrett Birkhoff. On the structure of abstract algebras. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 31, pages 433–454, Cambrdige, UK, October 1935. Cambridge University Press.

[3] Samuele Buro and Isabella Mastroeni. On the multi-language construction. In Luís Caires, editor, *Programming Languages and Systems (ETAPS)*, pages 293–321, Berlin/Heidelberg, DE, 2019. Springer.

[4] Samuele Buro and Isabella Mastroeni. On the semantic equivalence of language syntax formalisms. 2019.

[5] Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and Carolyn Talcott. *All about maude-a high-performance logical framework: how to specify, program and verify systems in rewriting logic*. Springer-Verlag, 2007.

[6] R. L. Crole. *Categories for Types*. Cambridge Mathematical Textbooks. Cambridge University Press, 1993. xvii+335 pages, ISBN 0521450926HB, 0521457017PB.

[7] Roy L. Crole. α-Equivalence Equalities. *Theoretical Computer Science*, 433:1–19, May 2012.

[8] Trevor Evans. On multiplicative systems defined by generators and relations: I. normal form theorems. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 47, pages 637–649. Cambridge University Press, 1951.

[9] Michael Furr and Jeffrey S. Foster. Checking type safety of foreign function calls. *SIGPLAN Not.*, 40(6):62–72, June 2005.

[10] Joseph Goguen and José Meseguer. Completeness of many-sorted equational logic. *Houston Journal of Mathematics*, 11(3):307–334, 1985.

[11] Joseph A. Goguen, Jean-Pierre Jouannaud, and José Meseguer. Operational semantics for order-sorted algebra. In *International Colloquium on Automata, Languages, and Programming*, pages 221–231. Springer, 1985.

[12] Joseph A Goguen and KAi Lin. Specifying, programming and verifying with equational logic. In *We Will Show Them!(2)*, pages 1–38, 2005.

[13] Joseph A. Goguen and José Meseguer. Order-sorted algebra I: equational deduction for multiple inheritance, overloading, exceptions and partial operations. *Theoretical Computer Science*, 105(2):217–273, 1992.

[14] Kathryn E. Gray. Safe cross-language inheritance. In Jan Vitek, editor, *ECOOP 2008 – Object-Oriented Programming*, pages 52–75, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[15] Gerard Huet and Derek C. Oppen. Equations and rewrite rules: A survey. Technical report, Stanford, CA, USA, 1980.

[16] Peter T Johnstone. *Sketches of an elephant: a Topos theory compendium*. Oxford logic guides. Oxford Univ. Press, 2002.

[17] Claude Kirchner and Hélène Kirchner. Equational logic and rewriting. In *Computational Logic*, pages 255–282, 2014.

[18] Donald E Knuth and Peter B Bendix. Simple word problems in universal algebras. In *Automation of Reasoning*, pages 342–376. Springer, 1983.

[19] M. Makkai and G.E. Reyes. *First Order Categorical Logic*. Lecture Notes in Mathematics. Springer Verlag, 1977.

[20] Narciso Martí-Oliet and José Meseguer. Inclusions and Subtypes I: First-order Case. *Journal of Logic and Computation*, 6(3):409–438, 1996.

[21] Jacob Matthews and Robert Bruce Findler. Operational semantics for multi-language programs. *ACM Transactions on Programming Languages and Systems*, 31(3):12:1–12:44, 2009.

[22] Jacob Matthews and Robert Bruce Findler. Operational Semantics for Multi-Language Programs. *ACM Transactions on Programming Languages and Systems*, 31(3):1–44, 2009.

[23] Eugenio Moggi. Notions of computation and monads. *Information and computation*, 93(1):55–92, 1991.

[24] Peter-Michael Osera, Vilhelm Sjöberg, and Steve Zdancewic. Dependent interoperability. In Koen Claessen and Nikhil Swamy, editors, *Proceedings of the Sixth Workshop on Programming Languages Meets Program Verification*, pages 3–14, New York, NY, USA, 2012. ACM.

[25] E. Palmgren and S.J. Vickers. Partial Horn logic and cartesian categorie. *Annals of Pure and Applied Logic*, 145(3):314 – 353, 2007.

[26] Daniel Patterson, Jamie Perconti, Christos Dimoulas, and Amal Ahmed. Funtal: Reasonably mixing a functional language with assembly. In *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 495–509, New York, NY, USA, 2017. ACM.

[27] James T. Perconti and Amal Ahmed. Verifying an open compiler using multi-language semantics. In *Proceedings of the 23rd European Symposium on Programming Languages and Systems*, pages 128–148, Berlin, DE, 2014. Springer.

[28] A. M. Pitts. Categorical logic. In S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science, Volume 5. Algebraic and Logical Structures*, chapter 2, pages 39–128. Oxford University Press, 2000.

[29] A. M. Pitts. Nominal techniques. *ACM SIGLOG News*, 3(1):57–72, January 2016.

[30] Gordon Plotkin. Some varieties of equational logic. In *Algebra, Meaning, and Computation*, pages 150–156. Springer, 2006.

[31] Traian Florin Şerbănuţă, Grigore Roşu, and José Meseguer. A rewriting logic approach to operational semantics. *Information and Computation*, 207(2):305–340, 2009.

[32] Gang Tan and Greg Morrisett. Ilea: Inter-language analysis across Java and C. *SIGPLAN Not.*, 42(10):39–56, October 2007.

[33] Alfred Tarski. Equational logic and equational theories of algebras. In *Studies in Logic and the Foundations of Mathematics*, volume 50, pages 275–288. Elsevier, Amsterdam, NL, 1968.

[34] Walter Taylor. Equational logic. *Houston Journal of Mathematics*, 47(2), 1979.

[35] Terese. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.

[36] Alberto Verdejo and Narciso Martí-Oliet. Executable structural operational semantics in maude. *The Journal of Logic and Algebraic Programming*, 67(1-2):226–293, 2006.