Title
Enriched Lawvere Theories for Operational Semantics

Permalink
https://escholarship.org/uc/item/3fz1k4k1

Authors
Baez, John C
Williams, Christian

Publication Date
2019-05-14

DOI
10.4204/eptcs.323.8

Peer reviewed

Enriched Lawvere Theories for Operational Semantics

John C. Baez and Christian Williams
Department of Mathematics
U. C. Riverside
Riverside, CA
92521 USA

`baez@math.ucr.edu, cwill041@ucr.edu`

Enriched Lawvere theories are a generalization of Lawvere theories that allow us to describe the operational semantics of formal systems. For example, a graph-enriched Lawvere theory describes structures that have a *graph* of operations of each arity, where the vertices are operations and the edges are *rewrites* between operations. Enriched theories can be used to equip systems with operational semantics, and maps between enriching categories can serve to translate between different forms of operational and denotational semantics. The Grothendieck construction lets us study all models of all enriched theories in all contexts in a single category. We illustrate these ideas with the *SKI*-combinator calculus, a variable-free version of the lambda calculus.

1 Introduction

Formal systems are not always explicitly connected to how they operate in practice. Lawvere theories [20] are an excellent formalism for describing algebraic structures obeying equational laws, but they do not specify how to compute in such a structure, for example taking a complex expression and simplifying it using rewrite rules. Recall that a Lawvere theory is a category with finite products T generated by a single object t, for "type", and morphisms $t^n \to t$ representing n-ary operations, with commutative diagrams specifying equations. There is a theory for groups, a theory for rings, and so on. We can specify algebraic structures of a given kind in some category C with finite products by a product-preserving functor $\mu \colon \mathsf{T} \to \mathsf{C}$. This is a simple and elegant form of *denotational* semantics. However, Lawvere theories know nothing of *operational* semantics. Our goal here is to address this using "enriched" Lawvere theories.

In a Lawvere theory, the objects are types and the morphisms are terms; however, there are no relations between terms, only equations. The process of computing one term into another should be given by hom-objects with more structure. In operational semantics, program behavior is often specified by labelled transition systems, or labelled directed multigraphs [26]. The edges of such a graph represent rewrites:

$$(\lambda x.x + x \ 2) \xrightarrow{\ \beta\ } 2 + 2 \xrightarrow{\ +\ } 4$$

We can use an enhanced Lawvere theory in which, rather than merely *sets* of morphisms, there are *graphs* or perhaps *categories*. Enriched Lawvere theories are exactly for this purpose.

In a theory T enriched in a category V of some kind of "directed object", including graphs, categories, and posets, the theory has the following interpretation:

types:	objects of T
terms:	morphisms of T
equations between terms:	commuting diagrams
rewrites between terms:	"edges" in hom in V

To be clear, this is not a new idea. Using enriched Lawvere theories for operational semantics has been explored in the past. For example, category-enriched theories have been studied by Seely [29] for the λ-calculus, and poset-enriched ones by Ghani and Lüth [23] for understanding "modularity" in term rewriting systems. They have been utilized extensively by Power, enriching in ω-complete partial orders to study recursion [13] – in fact, there the simplified "natural number" enriched theories which we explore were implicitly considered.

The goal of this paper is to give a simple unified explanation of enriched Lawvere theories and some of their applications to operational semantics. We aim our explanations at readers familiar with category theory but not yet enriched categories. To reduce the technical overhead we only consider enrichment over cartesian closed categories.

In general for a cartesian closed category V, a V-**theory** is a V-enriched Lawvere theory with natural number arities. We consider V as a choice of "method of computation" for algebraic theories. The main idea of this paper is that product-preserving functors between enriching categories allow for the translation between different kinds of semantics. This translation could be called "change of computation"—or, following standard mathematical terminology, **change of base**.

Because operational semantics uses graphs to represent terms and rewrites, one might expect some category like Gph, the category of directed multigraphs, to be our main example of enriching category: that is, the "thing" of n-ary operations, or n-variable terms in a theory, is a directed graph whose edges are rewrites. This is known as *small-step* operational semantics, meaning each edge represents a single instance of a rewrite rule.

When studying formal languages, one wants to pass from this local view to a global view: given a term, one cares about its possible evolutions after not only one rewrite but any finite sequence of rewrites. We study how programs operate in finite time. In computer science, this corresponds to defining a rewrite relation and forming its transitive closure, called *big-step* operational semantics. This is the classic example which change of base aims to generalize.

However, there is a subtlety. We may try to model the translation from small-step to big-step operational semantics using the "free category" functor, which for any directed multigraph forms the category whose objects are vertices and morphisms are finite paths of edges. However, this functor does *not* preserve products. One might hope to cure this using a better-behaved variant of directed multigraphs, such as reflexive graphs. One advantage of reflexive graphs is that that each vertex has a distinguished edge from it to itself; these describe rewrites that "do nothing". Thus, in a product of reflexive graphs there are edges describing the process of rewriting one factor while doing nothing in the other. This lets us handle parallelism. Unfortunately, as we shall explain, the free category functor from reflexive graphs to categories still fails to preserve products.

To obtain a product-preserving change of base taking us from small-step to big-step operational semantics, it seems the cleanest solution is to generalize graphs to *simplicial sets*. A simplicial set is a contravariant functor from the category Δ of finite linear orders and monotone maps to the category of sets and functions. It can be visualized as a space built from "simplices", which generalize triangles to any dimension: point, line, triangle, tetrahedron, etc. For an introduction to simplicial sets, see Friedman [12]. We use sSet to denote the category of simplicial sets, namely $\mathsf{Set}^{\Delta^{\mathrm{op}}}$.

Simplicial sets allow one to generalize rewriting to *higher-dimensional* rewriting, but this is not our focus here. Indeed, we only need two facts about simplicial sets in this paper:

- There is a full and faithful embedding of RGph, the category of reflexive graphs, in sSet, so we can think of a reflexive graph as a special kind of simplicial set (namely one whose n-simplices for $n > 1$ are all degenerate).

- The free category functor FC: sSet → Cat, often called "realization", preserves products.

We thus obtain a spectrum of cartesian closed categories V to enrich over, each connected to the next by a product-preserving functor, which allow us to examine the computation of term calculi in various ways:

Simplicial Sets	sSet-theories represent "small-step" operational semantics:
	— an edge is a *single* term rewrite.
Categories	Cat-theories represent "big-step" operational semantics:
	(Often this means a rewrite to a normal form. We use the term more generally.)
	— a morphism is a *finite sequence* of rewrites.
Posets	Pos-theories represent "full-step" operational semantics:
	— a boolean is the *existence* of a big-step rewrite.
Sets	Set-theories represent denotational semantics:
	— an element is a *connected component* of the rewrite relation.

In Section 2 we review Lawvere theories as a more explicit, but equivalent, presentation of finitary monads. In Section 3, we recall the basics of enrichment over cartesian closed categories. In Section 4 we give the central definition of V-theory, adapted from the work of Lucyshyn-Wright [22]. Using his work we show that a V-theory T gives a monadic adjunction between V and the V-category of models of T in V. This generalizes a fundamental result for Lawvere theories.

In Section 5 we discuss how suitable functors between enriching categories induce *change of base*: they transform theories, and their models, from one method of rewriting to another. Our main examples arise from this chain of adjunctions:

$$\text{sSet} \quad \underset{\underset{\text{UsS}}{\longleftarrow}}{\overset{\overset{\text{FC}}{\longrightarrow}}{\bot}} \quad \text{Cat} \quad \underset{\underset{\text{UC}}{\longleftarrow}}{\overset{\overset{\text{FP}}{\longrightarrow}}{\bot}} \quad \text{Pos} \quad \underset{\underset{\text{UP}}{\longleftarrow}}{\overset{\overset{\text{FS}}{\longrightarrow}}{\bot}} \quad \text{Set}$$

The right adjoints here automatically preserve finite products, but the left adjoints do as well, and these are what we really need:

- The functor FC: sSet → Cat maps a simplicial set (for example a reflexive graph) to the category it freely generates. Change of base along FC maps small-step operational semantics to big-step operational semantics.

- The functor FP: Cat → Pos maps a category C to the poset whose elements are objects of C, with $c \leq c'$ iff C has a morphism from c to c'. Change of base along FP maps big-step operational semantics to full-step operational semantics.

- The functor FS: Pos → Set maps a poset P to the set of "components" of P, where $p, p' \in P$ are in the same component if $p \leq p'$. Change of base along FS maps full-step operational semantics to denotational semantics.

In Section 6 we show that models of all V-theories for all enriching V can be assimilated into one category using the Grothendieck construction. In Section 7 we bring all the strands together and demonstrate these concepts in applications. First we consider the *SKI*-combinator calculus, and then we show how theories enriched over the category of labelled graphs can be used to study bisimulation.

Acknowledgements

2   Lawvere Theories

Algebraic structures are traditionally treated as sets equipped with operations obeying equations, but we can generalize such structures to live in any category with finite products. For example, given any category C with finite products, we can define a monoid internal to C to consist of:

$$
\begin{aligned}
\text{an object} \quad & M \\
\text{an identity element} \quad & e \colon 1 \to M \\
\text{and multiplication} \quad & m \colon M^2 \to M \\
\text{obeying the associative law} \quad & m \circ (m \times M) = m \circ (M \times m) \\
\text{and the right and left unit laws} \quad & m \circ (e \times \mathrm{id}_M) = \mathrm{id}_M = m \circ (\mathrm{id}_M \times e).
\end{aligned}
$$

Lawvere theories formalize this idea. For example, there is a Lawvere theory $\mathsf{Th}(\mathsf{Mon})$, the category with finite products freely generated by an object t equipped with an identity element $e \colon 1 \to t$ and multiplication $m \colon t^2 \to t$ obeying the associative law and unit laws listed above. This captures the "Platonic idea" of a monoid internal to a category with finite products. A monoid internal to C then corresponds to a functor $\mu \colon \mathsf{T} \to \mathsf{C}$ that preserves finite products.

In more detail, let N be any skeleton of the category of finite sets FinSet. Because N is the free category with finite coproducts on 1, N^{op} is the free category with finite products on 1. A **Lawvere theory** is a category with finite products T equipped with a functor $\tau \colon \mathsf{N}^{\mathrm{op}} \to \mathsf{T}$ that is the identity on objects and preserves finite products. Thus, a Lawvere theory is essentially a category generated by one object $\tau(1) = t$ and n-ary operations $t^n \to t$, as well as the projection and diagonal morphisms of finite products.

For efficiency let us call a functor that preserves finite products **cartesian**. Lawvere theories are the objects of a category Law whose morphisms are cartesian functors $f \colon \mathsf{T} \to \mathsf{T}'$ that obey $f\tau = \tau'$. More generally, for any category with finite products C, a **model** of the Lawvere theory T in C is a cartesian functor $\mu \colon \mathsf{T} \to \mathsf{C}$. The models of T in C are the objects of a category $\mathsf{Mod}(\mathsf{T}, \mathsf{C})$, in which the morphisms are natural transformations.

A theory can thus have models in many different contexts. For example, there is a Lawvere theory $\mathsf{Th}(\mathsf{Mon})$, the theory of monoids, described as above. Ordinary monoids are models of this theory in Set, while topological monoids are models of this theory in Top.

For completeness, it is worthwhile to mention the *presentation* of a Lawvere theory: how exactly does the above "sketch" of $\mathsf{Th}(\mathsf{Mon})$ produce a category with finite products? It is precisely analogous to the

presentation of an algebra by generators and relations: we form the free category with finite products on the data given, and impose the required equations. The result is a category whose objects are powers of t, and whose morphisms are composites of products of the morphisms in $\mathsf{Th}(\mathsf{Mon})$, projections, deletions, symmetries and diagonals. A detailed account was given by Barr and Wells [4, Chap. 4].

In 1965, Linton [21] proved that Lawvere theories correspond to "finitary monads" on the category of sets. For every Lawvere theory T, there is an adjunction:

$$\mathsf{Set} \quad \overset{F}{\underset{U}{\overset{\longrightarrow}{\underset{\longleftarrow}{\perp}}}} \quad \mathsf{Mod}(\mathsf{T},\mathsf{Set}).$$

The functor

$$U \colon \mathsf{Mod}(\mathsf{T},\mathsf{Set}) \to \mathsf{Set}$$

sends each model μ to its underlying set, $X = \mu(\tau(1))$. Its left adjoint, the free model functor

$$F \colon \mathsf{Set} \to \mathsf{Mod}(\mathsf{T},\mathsf{Set}),$$

sends each finite set $n \in \mathsf{N}$ to the representable functor $\mathsf{T}(\tau(n),-) \colon \mathsf{T} \to \mathsf{Set}$, and in general any set X to the colimit of all such representables as n ranges over the poset of finite subsets of X. In rough terms, $F(X)$ is the model of all n-ary operations from T on the set X.

If we momentarily abbreviate $\mathsf{Mod}(\mathsf{T},\mathsf{Set})$ as Mod, we obtain an adjunction

$$\mathsf{Mod}(F(n),\mu) = \mathsf{Mod}(\mathsf{T}(\tau(n),-),\mu) \cong \mu(\tau(n)) \cong \mu(\tau(1))^n = \mathsf{Set}(n,U(\mu))$$

where the left isomorphism arises from the Yoneda lemma, and the right isomorphism from the product preservation of μ.

This adjunction induces a monad T on Set:

$$T(X) = \int^{n\in\mathsf{N}} X^n \times \mathsf{T}(n,1). \tag{1}$$

The integral here is a coend, essentially a coproduct quotiented by the equations of the theory and the equations induced by the cartesian structure of the category. This forms the set of all terms that can be constructed from applying the operations to the elements, subject to the equations of the theory. The monad constructed this way is always **finitary**: that is, it preserves filtered colimits [1], or its action on sets is determined by its action on finite sets.

Conversely, for a monad T on Set, its Kleisli category $\mathsf{Kl}(T)$ is the category of all free algebras of the monad, which has all coproducts. There is a functor $k \colon \mathsf{Set} \to \mathsf{Kl}(T)$ that is the identity on objects and preserves coproducts. Thus,

$$k^{\mathrm{op}} \colon \mathsf{Set}^{\mathrm{op}} \to \mathsf{Kl}(T)^{\mathrm{op}}$$

is a cartesian functor, and restricting its domain to N^{op} is a Lawvere theory k_T. To see what this is doing, note that:

$$\mathsf{Kl}(T)^{\mathrm{op}}(n,m) = \mathsf{Kl}(T)(m,n) = \mathsf{Set}(m,T(n))$$

where the latter is considered as m n-ary operations in the Lawvere theory k_T. When T is finitary, the monad arising from this Lawvere theory is naturally isomorphic to T itself.

This correspondence sets up an equivalence between the category Law of Lawvere theories and the category of finitary monads on Set. There is also an equivalence between the category $\mathsf{Mod}(\mathsf{T}, \mathsf{Set})$ of models of a Lawvere theory T and the category of algebras of the corresponding finitary monad T. Furthermore, all this generalizes with Set replaced by any "locally finitely presentable" category [1]. For more details see [4, 20, 24].

One final point, provided to us by Mike Stay: while monads are often associated with functional programming languages such as Haskell, Lawvere theories correspond to *interfaces* or abstract classes in object-oriented programming. In these one declares various constants, types, and abstract functions satisfying tests, and then one implements the interface by assigning these elements, sets, functions, and equations—precisely a model in Set. While people think of monads as the main example of "categories in programming", in fact Lawvere theories are ubiquitous.

3 Enrichment

To incorporate the aspect of computation, we now turn to Lawvere theories that have hom-*objects* rather than mere hom-*sets*. To do this we use enriched category theory [18] and replace sets with objects of a cartesian closed category V, called the "enriching" category or "base". A V-enriched category or V**-category** C is:

$$
\begin{array}{rl}
\text{a collection of objects} & \mathsf{Ob}(\mathsf{C}) \\
\text{a hom-object function} & \mathsf{C}(-,-)\colon \mathsf{Ob}(\mathsf{C}) \times \mathsf{Ob}(\mathsf{C}) \to \mathsf{Ob}(\mathsf{V}) \\
\text{composition morphisms} & \circ_{a,b,c}\colon \mathsf{C}(b,c) \times \mathsf{C}(a,b) \to \mathsf{C}(a,c) \quad \forall a,b,c \in \mathsf{Ob}(\mathsf{C}) \\
\text{identity-assigning morphisms} & id_a\colon 1_{\mathsf{V}} \to \mathsf{C}(a,a) \quad \forall a \in \mathsf{Ob}(\mathsf{C})
\end{array}
$$

such that composition is associative and unital. A V**-functor** $F\colon \mathsf{C} \to \mathsf{D}$ is:

$$
\begin{array}{rl}
\text{a function} & F\colon \mathsf{Ob}(\mathsf{C}) \to \mathsf{Ob}(\mathsf{D}) \\
\text{a collection of morphisms} & F_{ab}\colon \mathsf{C}(a,b) \to \mathsf{D}(F(a),F(b)) \quad \forall a,b \in \mathsf{C}
\end{array}
$$

such that F preserves composition and identity. A V**-natural transformation** $\alpha\colon F \Rightarrow G$ is:

$$
\text{a family} \quad \alpha_a\colon 1_{\mathsf{V}} \to \mathsf{D}(F(a),G(a)) \quad \forall a \in \mathsf{Ob}(\mathsf{C})
$$

such that α is "natural" in a: an evident square commutes. There is a 2-category VCat of V-categories, V-functors, and V-natural transformations.

We can construct new V-categories from old ones by taking products or opposites, in obvious ways. There is also a V-category denoted $\underline{\mathsf{V}}$ with the same objects as V and with hom-objects given by the internal hom:

$$
\underline{\mathsf{V}}(v,w) = w^v \quad \forall v,w \in \mathsf{V}.
$$

The concepts of adjunction and monad generalize straightforwardly to V-categories, and when we speak of an adjunction or monad in the enriched context this generalization is what we mean [18]. For example, there is an adjunction

$$
\underline{\mathsf{V}}(u \times v, w) \cong \underline{\mathsf{V}}(u, w^v)
$$

called "currying".

We can generalize products and coproducts to the enriched context. Given a V-category C, the V-**coproduct** of an n-tuple of objects $b_1, \ldots, b_n \in \mathrm{Ob}(C)$ is an object b equipped with a V-natural isomorphism

$$C(b, -) \cong \prod_{i=1}^{n} C(b_i, -).$$

If such an object exists, we denote it by $\sum_{i=1}^{n} b_i$. This makes sense even when $n = 0$: a 0-ary V-coproduct in C is called a V-**initial object** and denoted as 0_C. When V is cartesian closed, any finite coproduct that exists in V is also a V-coproduct in $\underline{\mathsf{V}}$. In particular,

$$u^{v+w} \cong u^v \times u^w \ \text{ and } \ w^0 \cong 1_\mathsf{V}$$

whenever 0 is an initial object of V. Conversely, any finite V-coproduct that exists in V is also a coproduct in the usual sense.

Similarly, a V-**product** of objects $b_1, \ldots, b_n \in \mathrm{Ob}(C)$ is an object b equipped with a V-natural isomorphism

$$\mathsf{C}(-, b) \cong \prod_{i=1}^{n} \mathsf{C}(-, b_i). \tag{2}$$

If such an object b exists, we denote it by $\prod_{i=1}^{n} b_i$. A 0-ary product in C is called a V-**terminal object** and denoted as 1_C. Whenever V is cartesian closed, the finite products in V are also V-products in $\underline{\mathsf{V}}$. In particular,

$$(u \times v)^w \cong u^w \times v^w \ \text{ and } \ 1_\mathsf{V}^w \cong 1_\mathsf{V}$$

where our chosen terminal object 1_V is also V-terminal. Conversely, any finite V-product in V is also a product in the usual sense.

A general V-category C does not exactly have projections from a V-product to its factors, since given two objects $c, c' \in \mathrm{Ob}(\mathsf{C})$ there is not, fundamentally, a *set* of morphisms from c to c'. Instead there is the hom-object $\mathsf{C}(c, c')$, which is an object of V. However, any object v of V has a set of **elements**, namely morphisms $f \colon 1_\mathsf{V} \to v$. Elements of $\mathsf{C}(c, c')$ act like morphisms from c to c'.

In particular, any V-product $b = \prod_{i=1}^{n} b_i$ gives rise to elements

$$p_i \colon 1_\mathsf{V} \to \mathsf{C}(b, b_i)$$

which serve as substitutes for the projections in a usual product. These elements are defined as composites

$$1_\mathsf{V} \xrightarrow{id_b} \mathsf{C}(b, b) \xrightarrow{\ \sim\ } \prod_{i=1}^{n} \mathsf{C}(b, b_i) \to \mathsf{C}(b, b_i)$$

where the isomorphism comes from Eq. (2) and the last arrow is a projection in V.

Even better, we can bundle up all these elements p_i into a single element

$$p \colon 1_\mathsf{V} \to \prod_{i=1}^{n} \mathsf{C}(b, b_i)$$

which serves as a substitute for the universal cone in a usual product. Starting from p we can recover the V-natural isomorphism in Eq. (2) as follows:

$$\mathsf{C}(-, b) \xrightarrow{\ \sim\ } 1_\mathsf{V} \times \mathsf{C}(-, b) \xrightarrow{p \times 1} \prod_{i=1}^{n} \mathsf{C}(b, b_i) \times \mathsf{C}(-, b) \longrightarrow \prod_{i=1}^{n} \mathsf{C}(-, b_i) \tag{3}$$

where the last arrow is given by composition. Thus, we say a **universal cone** exhibiting b as the V-product of objects $b_1, \ldots, b_n$ is an element $p \colon 1_V \to \prod_{i=1}^{n} C(b, b_i)$ such that the V-natural transformation $C(-, b) \to \prod_{i=1}^{n} C(-, b_i)$ given by Eq. (3) is an isomorphism.

The advantage of this reformulation is that we can say a V-functor $F \colon C \to D$ **preserves finite V-products** if for every universal cone $p \colon 1_V \to \prod_{i=1}^{n} C(b, b_i)$ exhibiting b as the V-product of the objects b_i, the composite

$$1_V \xrightarrow{p} \prod_{i=1}^{n} C(b, b_i) \xrightarrow{\Pi_i F} D(F(b), F(b_i))$$

is universal cone exhibiting $F(b)$ as the V-product of the objects $F(b_i)$.

A bit more subtly, generalizing the exponentials in V, a V-category C can have "powers". Given $v \in Ob(V)$, we say an object $c^v \in Ob(C)$ is a v-**power** of $c \in Ob(C)$ if it is equipped with a V-natural isomorphism

$$C(-, c^v) \cong C(-, c)^v. \tag{4}$$

In the special case $V = Set$ this forces c^v to be the v-fold product of copies of c. As with V-products, it is useful to repackage the isomorphism of Eq. (4) so we can say what it means for a V-functor to preserve v-powers. First, note that this isomorphism gives rise to an element

$$q \colon 1_V \to C(c^v, c)^v,$$

namely the composite

$$1_V \xrightarrow{id_{c^v}} C(c^v, c^v) \xrightarrow{\sim} C(c^v, c)^v.$$

Conversely, any element $q \colon 1_V \to C(c^v, c)^v$ determines a V-natural transformation $e \colon C(-, c^v) \to C(-, c)^v$, and we say e is a **universal cone** if this V-natural transformation is an isomorphism. Next, suppose C and D are V-categories with v-powers. We say a V-functor $F \colon C \to D$ **preserves v-powers** if it maps universal cones to universal cones.

There are just a few more technicalities. A category is **locally finitely presentable** if it is the category of models for a finite limits theory, and an object is **finite** if its representable functor is **finitary**: that is, it preserves filtered colimits [1]. A V-category C is **locally finitely presentable** if its underlying category C_0 is locally finitely presentable, C has finite powers, and $(-)^x \colon C_0 \to C_0$ is finitary for all finitely presentable x. The details are not crucial here: all categories to be considered are locally finitely presentable. We will use V_f to denote the full subcategory of V of finite objects: in sSet, these are simplicial sets with finitely many n-simplices for each n.

4 Enriched Lawvere Theories

Power introduced the notion of enriched Lawvere theory about twenty years ago, "in seeking a general account of what have been called notions of computation" [27]. The original definition is as follows: for a symmetric monoidal closed category $(V, \otimes, 1)$, a "V-enriched Lawvere theory" is a V-category T that has powers by objects in V_f, equipped with an identity-on-objects V-functor

$$\tau \colon \underline{V}_f^{op} \to T$$

that preserves these powers. A "model" of a V-theory is a V-functor $\mu \colon T \to V$ that preserves powers by finite objects of V. There is a category $Mod(T, V)$ whose objects are models and whose morphisms

are V-natural transformations. The monadic adjunction and equivalence of Section 2 generalize to the enriched setting.

In this paper, however, we only consider *natural number* arities, while still retaining enrichment. To do this we use the work of Lucyshyn-Wright [22], who along with Power [25] has generalized Power's original ideas to allow a more flexible choice of arities. We also limit ourselves to the case where the tensor product of V is cartesian. This has a significant simplifying effect, yet it suffices for many cases of interest in computer science.

Thus, in all that follows, we let $(V, \times, 1_V)$ be a cartesian closed category equipped with chosen finite coproducts of the terminal object 1_V, say

$$n_V = \sum_{i \in n} 1_V.$$

Define N_V to be the full subcategory of V containing just these objects n_V. There is also a V-category $\underline{N}_V$ whose objects are those of N_V and whose hom-objects are given as in V. We define the V-category of **arities** for V to be

$$A_V := \underline{N}_V^{op}.$$

We shall soon see that A_V has finite V-products.

Definition 1. We define a V-**theory** (T, τ) to be a V-category T equipped with a V-functor

$$\tau \colon A_V \to T$$

that is the identity on objects and preserves finite V-products.

Definition 2. A **model** of T in a V-category C is a V-functor

$$\mu \colon T \to C$$

that preserves finite V-products.

Just as all the objects of a Lawvere theory are finite products of a single object, we shall see that all the objects of T are finite V-products of the object

$$t = \tau(1_V).$$

Definition 3. We define VLaw, the **category of V-theories**, to be the category for which an object is a V-theory and a morphism from (T, τ) to (T', τ') is a V-functor $f \colon T \to T'$ that preserves finite V-products and has $f\tau = \tau'$.

Definition 4. For every V-theory (T, τ) and every V-category C with finite V-products, we define $\mathrm{Mod}(T, C)$, the **category of models** of (T, τ) in C, to be the category for which an object is a V-functor $\mu \colon T \to C$ that preserves finite V-products and a morphism is a V-natural transformation.

The basic monadicity results for Lawvere theories generalize to V-theories when V is complete and cocomplete, as in the main examples we consider: $V = sSet, Cat, Pos,$ and Set. Under this extra assumption VLaw and $\mathrm{Mod}(T, C)$ can be promoted to V-categories, which we call $\underline{VLaw}$ and $\underline{Mod}(T, C)$. Furthermore, there is a V-functor

$$U \colon \underline{Mod}(T, V) \to \underline{V}$$

sending any model $\mu \colon T \to V$ to its underlying object $\mu(t) \in V$. Recall that monads and adjunctions make sense in VCat, just as they do in Cat. The V-functor U has a left adjoint

$$F \colon \underline{V} \to \underline{Mod}(T, V),$$

and $\underline{Mod}(T, V)$ is equivalent to the V-category of algebras of the resulting monad $T = UF$. More precisely:

Theorem 5. Suppose V is cartesian closed, complete and cocomplete, and has chosen finite coproducts of the terminal object. Let (T, τ) be a V-theory. Then there is a monadic adjunction

$$\underline{V} \quad \overset{F}{\underset{U}{\rightleftarrows}} \bot \quad \underline{Mod}(T, V).$$

Proof. This follows from Lucyshyn-Wright's general theory [22], so our task is simply to explain how. He allows V to be a symmetric monoidal category, and uses a more general concept of algebraic theory with a system of arities given by any fully faithful symmetric monoidal V-functor $j\colon J \to \underline{V}$. For us $J = \underline{N}_V$ and $j\colon \underline{N}_V \to \underline{V}$ is the obvious inclusion; this is his Example 3.7.

Lucyshyn-Wright defines a J-**theory** to be a V-functor $\tau\colon J^{op} \to T$ that is the identity on objects and preserves powers by objects in J (or more precisely, their images under j). For us $J^{op} = A_V$. So, to apply his theory, we need to show that a V-functor $\tau\colon A_V \to T$ preserves powers by objects in N_V if and only if it preserves finite V-products. This is Lemma 16 below.

He defines a model (or "algebra") of a J-theory to be a V-functor $\tau\colon T \to \underline{V}$ that preserves powers by objects in J. He defines a morphism of models to be a V-natural transformation between such V-functors. So, to apply his theory, we also need to show that when $J = \underline{N}_V$, a V-functor $\mu\colon T \to \underline{V}$ preserves powers by objects of J if and only if it preserves finite V-products. This is Lemma 17 below.

A technical concept fundamental to Lucyshyn-Wright's theory is that of an **eleutheric** system of arities $j\colon J \to \underline{V}$. This is one where the left Kan extension of any V-functor $f\colon J \to \underline{V}$ along j exists and is preserved by each V-functor $\underline{V}(x, -)\colon \underline{V} \to \underline{V}$. In Example 7.5.5 he shows that $j\colon \underline{N}_V \to \underline{V}$ is eleutheric when V is countably cocomplete. In Thm. 8.9 shows that when $j\colon J \to \underline{V}$ is eleutheric, and has equalizers, we may form the V-category $\underline{Mod}(T, V)$, and that the forgetful V-functor

$$U\colon \underline{Mod}(T, V) \to \underline{V}$$

is monadic. This is the result we need. So, our theorem actually holds whenever V is cartesian closed, with equalizers and countable colimits, and has chosen finite coproducts of the initial object. $\qquad \square$

Before turning to examples, a word about Lucyshyn-Wright's construction of the left adjoint F and the monad T is in order. These rely on the "free model" on an object $n_V \in V$. This is the enriched generalization of the free model described in Section 2: it is the composite of $\tau^{op}\colon A_V^{op} \to T^{op}$ with the enriched Yoneda embedding $y\colon T^{op} \to [T, V]$:

$$A_V^{op} \quad \overset{\tau^{op}}{\longrightarrow} \quad T^{op} \quad \overset{y}{\to} \quad [T, V]$$

$$n_V \quad \mapsto \quad t^{n_V} \quad \mapsto \quad T(t^{n_V}, -)$$

Since an object of V does not necessarily have a "poset of finite subobjects" over which to take a filtered colimit (as in Set), the extension of this "free model" functor $y\tau^{op}$ to all of V is specified by a somewhat higher-powered generalization: it is the left Kan extension of $y\tau^{op}$ along j.

$$N_V \xrightarrow{\;\; y\tau^{op} \;\;} [T, V]$$
$$j \searrow \quad \Downarrow \eta \quad \nearrow \quad F := \mathrm{Lan}_j y\tau^{op}$$
$$V$$

This is the universal "best solution" to the problem of making the triangle commute up to a V-natural transformation. That is, for any functor $G\colon \mathsf{V} \to [\mathsf{T}, \mathsf{V}]$ and V-natural transformation $\theta\colon y\tau^{\mathrm{op}} \Rightarrow Gj$, the latter factors uniquely through η. From the adjunction between V and the category of models $\mathsf{Mod}(\mathsf{T}, \mathsf{V})$ we obtain a V-enriched monad

$$T = UF\colon \mathsf{V} \to \mathsf{V},$$

and this has a more concrete formula as an enriched coend:

$$T(V) = \int^{n_{\mathsf{V}} \in \mathsf{N}_{\mathsf{V}}} V^{n_{\mathsf{V}}} \times \mathsf{T}(t^{n_{\mathsf{V}}}, t).$$

We next give two examples of a rather abstract nature, where we show how Cat-enriched Lawvere theories can describe categories with extra structure. In Section 7 we study examples more directly connected to operational semantics.

Example 6. When $\mathsf{V} = \mathsf{Cat}$, a V-category is a 2-category, so a V-theory deserves to be called a **2-theory**. For example, let $\mathsf{T} = \mathsf{Th}(\mathsf{PsMon})$ be the 2-theory of pseudomonoids. A pseudomonoid [9] is a weakened version of a monoid: rather than associativity and unitality *equations*, it has 2-isomorphisms called the associator and unitors, which we can treat as *rewrite rules*. To equate various possible rewrite sequences, these 2-isomorphisms must obey equations called "coherence laws". Power [19] has introduced "enriched sketches" as a way of presenting enriched Lawvere theories. Informally, here is a presentation of the 2-theory for pseudomonoids:

$$\mathsf{Th}(\mathsf{PsMon})$$

sort	M	pseudomonoid
operations	$m\colon M^2 \to M$	multiplication
	$e\colon 1 \to M$	identity
rewrites	$\alpha\colon m \circ (m \times \mathrm{id}_M) \overset{\sim}{\Longrightarrow} m \circ (\mathrm{id}_M \times m)$	associator
	$\lambda\colon m \circ (e \times \mathrm{id}_M) \overset{\sim}{\Longrightarrow} \mathrm{id}_M$	left unitor
	$\rho\colon m \circ (\mathrm{id}_M \times e) \overset{\sim}{\Longrightarrow} \mathrm{id}_M$	right unitor
equations		
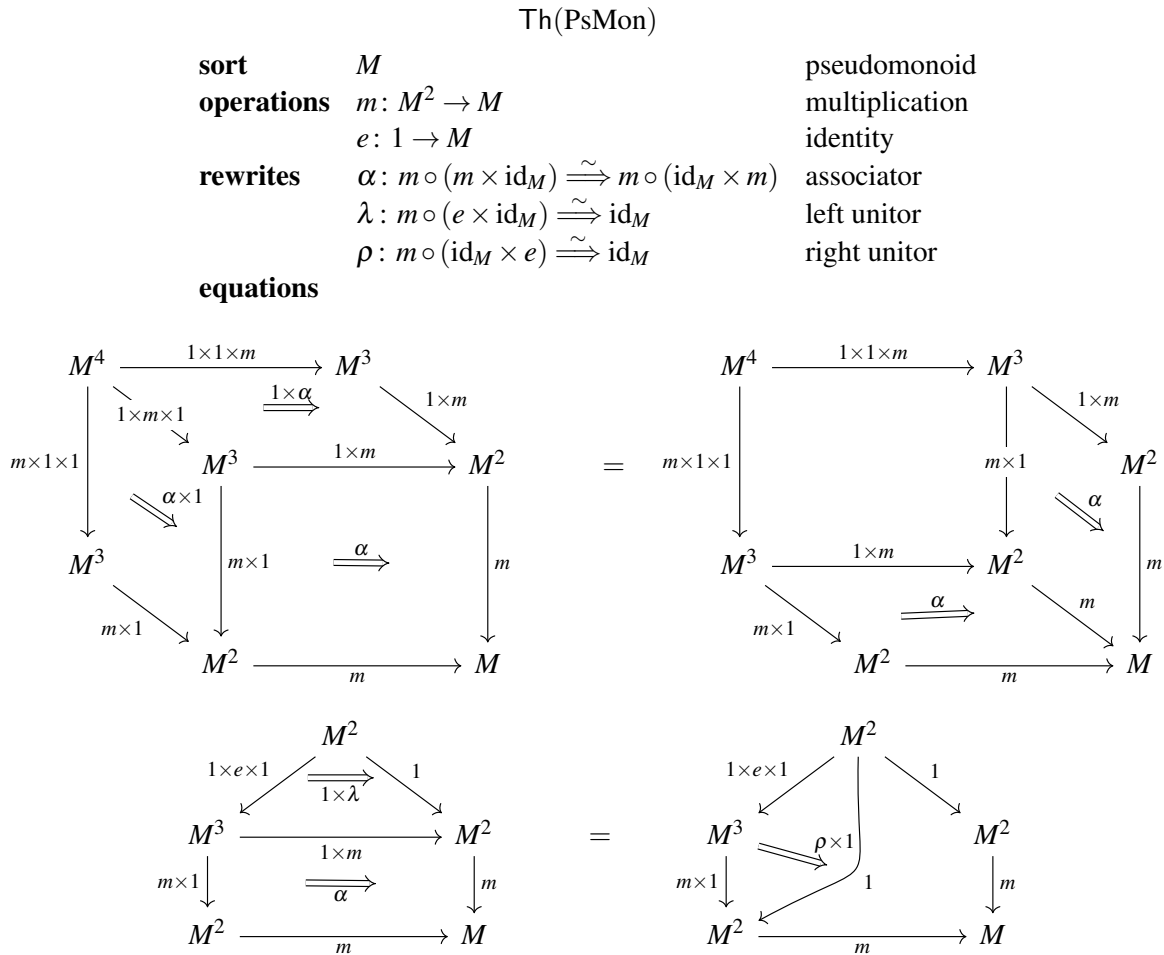
We write the equations as commutative diagrams merely for convenience; they could also be written as equations in a more traditional style. The top diagram expresses the pentagon identity for the associator, while the bottom one expresses the usual coherence law involving the left and right unitors.

Models of $\mathsf{T} = \mathsf{Th}(\mathsf{PsMon})$ in Cat are monoidal categories: let us explore this example in more detail. A model of T is a finite-product-preserving 2-functor $\mu \colon \mathsf{T} \to \mathsf{Cat}$, which sends

$$
\begin{aligned}
t &\mapsto \mathsf{C} \\
m &\mapsto \otimes \colon &\mathsf{C}^2 &\to \mathsf{C} \\
e &\mapsto I \colon &1 &\to \mathsf{C} \\
\alpha &\mapsto a \colon &\otimes \circ (\otimes \times 1_\mathsf{C}) &\Rightarrow \otimes \circ (1_\mathsf{C} \times \otimes) \\
\lambda &\mapsto \ell \colon &I \circ 1_\mathsf{C} &\Rightarrow 1_\mathsf{C} \\
\rho &\mapsto r \colon &1_\mathsf{C} \circ I &\Rightarrow 1_\mathsf{C}
\end{aligned}
$$

such that the coherence laws of the rewrites are preserved. Thus, a model is a category equipped with a tensor product $\otimes$ and unit object I such that these operations are associative and unital up to natural isomorphism; so these models are precisely monoidal categories.

Given two models $\mu, \nu \colon \mathsf{T} \to \mathsf{Cat}$, a morphism of models is a 2-natural transformation $\varphi \colon \mu \Rightarrow \nu$; this amounts to a strict monoidal functor $\varphi \colon (\mathsf{C}, \otimes_C, I_C) \to (\mathsf{D}, \otimes_D, I_D)$. The strictness arises because morphisms between models are 2-natural transformations rather than pseudonatural transformations. There is a substantial amount of theory on pseudomonads and pseudoalgebras [6, 10], but to the authors' knowledge the theory-monad correspondence has not yet been extended to include the case of weak naturality.

Finally, because Cat is complete and cocomplete, the category of models $\mathsf{Mod}(\mathsf{T}, \mathsf{Cat})$ can be promoted to a 2-category $\underline{\mathsf{Mod}}(\mathsf{T}, \mathsf{Cat})$. This is the 2-category of monoidal categories, strict monoidal functors, and monoidal natural transformations.

We can accomplish the same thing on the monad side: a Cat-enriched monad is called a **2-monad**, and T gives rise to the "free monoidal category" 2-monad T on Cat [6]. To apply this 2-monad to $\mathsf{C} \in \mathsf{Cat}$ we first form the free model on C by taking a left Kan extension as above, and then evaluate this model at the generating object. In the same way that the (underlying set of the) free monoid on a set X consists of all finite strings of elements of X, $T(\mathsf{C})$ is the monoidal category consisting of all finite tensor products of objects of C and all morphisms built from those of C by composition and tensoring together with associators and unitors obeying the necessary coherence laws. Morphisms of these algebras are strict monoidal functors, while 2-morphisms are natural transformation. We thus have a 2-equivalence between $\underline{\mathsf{Mod}}(\mathsf{T}, \mathsf{Cat})$ and the 2-category of algebras of T.

In this way, 2-theories generalize equipping *set*-like objects with operations obeying equations to equipping *category*-like objects with operations obeying equations up to transformations that obey equations of their own. In particular, this gives us a way to present graphical calculi such as string diagrams – the language of monoidal categories.

Example 7. Enrichment generalizes operations in more ways than by weakening equations to coherent isomorphisms. We can also use 2-theories to describe other structures that make sense inside 2-categories, such as adjunctions.

For example, we may define a cartesian category X to be one equipped with right adjoints to the diagonal $\Delta_\mathsf{X} \colon \mathsf{X} \to \mathsf{X} \times \mathsf{X}$ and the unique functor $!_\mathsf{X} \colon \mathsf{X} \to 1_\mathsf{Cat}$. These right adjoints are a functor $m \colon \mathsf{X}^2 \to \mathsf{X}$ describing binary products in X and a functor $e \colon 1 \to \mathsf{X}$ picking out the terminal object in X. We can capture the fact that they are right adjoints by providing them with units and counits and imposing the triangle equations. There is thus a 2-theory $\mathsf{Th}(\mathsf{Cart})$ whose models in Cat are categories with chosen

finite products. More generally a model of this 2-theory in any 2-category C with finite products is called a **cartesian object** in C.

<div align="center">

Th(Cart)

</div>

type	X	cartesian object
operations	$m\colon X^2 \to X$	product
	$e\colon 1 \to X$	terminal element
rewrites	$\triangle\colon \mathrm{id}_X \Longrightarrow m \circ \Delta_X$	unit of adjunction between m and Δ_X
	$\pi\colon \Delta_X \circ m \Longrightarrow \mathrm{id}_{X^2}$	counit of adjunction between m and Δ_X
	$\top\colon \mathrm{id}_X \Longrightarrow e \circ {!}_X$	unit of adjunction between e and ${!}_X$
	$\varepsilon\colon {!}_X \circ e \Longrightarrow \mathrm{id}_1$	counit of adjunction between e and ${!}_X$

equations

<div align="center">

$$
\begin{array}{ccc}
& \Delta_X & \\
\Delta_X\circ\triangle \big\Downarrow \quad & \xRightarrow{\quad\ 1\ \quad} & \\
\Delta_X \circ m \circ \Delta_X & \xRightarrow[\pi\circ\Delta_X]{} & \Delta_X
\end{array}
\qquad\qquad
\begin{array}{ccc}
& m & \\
\triangle\circ m \big\Downarrow \quad & \xRightarrow{\quad\ 1\ \quad} & \\
m \circ \Delta_X \circ m & \xRightarrow[m\circ\pi]{} & m
\end{array}
$$

$$
\begin{array}{ccc}
& {!}_X & \\
{!}_X\circ\top \big\Downarrow \quad & \xRightarrow{\quad\ 1\ \quad} & \\
{!}_X \circ e \circ {!}_X & \xRightarrow[\varepsilon\circ{!}_X]{} & {!}_X
\end{array}
\qquad\qquad
\begin{array}{ccc}
& e & \\
\top\circ e \big\Downarrow \quad & \xRightarrow{\quad\ 1\ \quad} & \\
e \circ {!}_X \circ e & \xRightarrow[e\circ\varepsilon]{} & e
\end{array}
$$

</div>

Again we write the equations as commutative diagrams, but this time commutative triangles of 2-morphisms in Th(Cart). These are the triangle equations that force m to be the right adjoint of Δ_X and e to be the right adjoint of ${!}_X$. A model of Th(Cart) is a category with chosen binary products and a chosen terminal object; morphisms in Mod(Th(Cart), Cat) are functors that strictly preserve this extra structure.

The subtle interplay between the cartesian structure of Th(Cart) and the cartesian structure of the object $X \in$ Th(Cart) is an example of the "microcosm principle": objects with a given structure are most generally defined in a context that has the same sort of structure. As seen in the previous example, we can also define pseudomonoids in any 2-category with finite products, but this is excess to requirements: one can in fact define them more generally in any monoidal 2-category [9].

In fact, if we let arities be finite categories, we would have Cat-theories of categories with finite limits and colimits. However, for the purposes of this paper we are using only natural number arities. This suffices for constructing Th(Cart) and also Th(CoCart), the theory of categories with chosen binary coproducts and a chosen initial object. Various other kinds of categories—distributive categories, rig categories, etc.—can also be expressed using Cat-theories with natural number arities. This gives a systematic formalization of these categories, internalizes them to new contexts, and allows for the generation of 2-monads that describe them.

5 Change of Base

We now have the tools to formulate the main idea: a choice of enrichment for Lawvere theories corresponds to a choice of *computation*, and changing enrichments corresponds to a *change of computation*. We propose a general framework in which one can translate between different forms of computation: small-step, big-step, full-step operational semantics, and denotational semantics.

5.1 General results

Suppose that V and W are enriching categories of the sort we are considering: cartesian closed categories equipped with chosen finite coproducts of the terminal object. Suppose $F: V \to W$ preserves finite products. This induces a **change of base** functor $F_*: VCat \to WCat$ [7] which takes any V-category C and produces a W-category $F_*(C)$ with the same objects but with

$$F_*(C)(a,b) := F(C(a,b))$$

for all objects a, b. Composition in $F_*(C)$ is defined by

$$F(C(b,c)) \times F(C(a,b)) \xrightarrow{\sim} F(C(b,c) \times C(a,b)) \xrightarrow{F(\circ_{a,b,c})} F(C(a,b)).$$

The identity-assigning morphisms are given by

$$1_W \xrightarrow{\sim} F(1_V) \xrightarrow{F(id_a)} F(C(a,a)).$$

Moreover, if $f: C \to D \in VCat$ is a V-functor, there is a W-functor $F_*(f): F_*(C) \to F_*(D)$ that on objects equals f and on hom-objects equals $F(f)$. If $\alpha: f \Rightarrow g$ is a V-natural transformation and $c \in Ob(C)$, then we define

$$F_*(\alpha)_c: 1_W \xrightarrow{\sim} F(1_V) \xrightarrow{F(\alpha_c)} F(D(f(c),g(c))).$$

Thus, change of base actually gives a 2-functor from the 2-category of V-categories, V-functors and V-natural transformations to the corresponding 2-category for W.

In fact, the change of base operation gives a 2-functor

$$\begin{array}{ccc} \mathsf{CartCat} & \xrightarrow{(-)_*} & \mathsf{2Cat} \\ (F: V \to W) & \mapsto & (F_*: VCat \to WCat) \end{array}$$

where $\mathsf{CartCat}$ is the 2-category of cartesian closed categories equipped with chosen finite coproducts of the terminal object, finite product preserving functors preserving these chosen finite coproducts, and natural transformations. In particular, if V has not just finite coproducts of the terminal object but all coproducts of this object, there is a map of adjunctions

Each set X is mapped to the X-indexed coproduct of the terminal object in V and conversely each object v of V is represented in Set by the hom-set from the unit to v. The latter induces the "underlying category" change of base, which forgets the enrichment. The former induces the "free V-enrichment" change of base, whereby ordinary Set-categories are converted to V-categories, denoted $\mathsf{C} \mapsto \underline{\mathsf{C}}$. These form an adjunction, because 2-functors preserve adjunctions.

We now study how change of base affects theories and their models. We start by asking when a functor $F\colon \mathsf{V} \to \mathsf{W}$ induces a change of base $F_*\colon \mathsf{VCat} \to \mathsf{WCat}$ that "preserves enriched theories". That is, given a V-theory

$$\tau\colon \mathsf{A}_{\mathsf{V}} \to \mathsf{T}$$

we want to determine conditions for the base-changed functor

$$F_*(\tau)\colon F_*(\mathsf{A}_{\mathsf{V}}) \to F_*(\mathsf{T})$$

to induce a W-theory in a canonical way. Recall that we require V and W to be cartesian closed, equipped with chosen finite coproducts of their terminal objects. We thus expect the following conditions to be sufficient: F should be cartesian, and it should preserve the chosen finite coproducts of the terminal object:

$$F(n_{\mathsf{V}}) = n_{\mathsf{W}}$$

for all n.

Given these conditions there is a W-functor, in fact an isomorphism

$$\tilde{F}\colon \mathsf{A}_{\mathsf{W}} \to F_*(\mathsf{A}_{\mathsf{V}}).$$

On objects this maps n_{W} to n_{V}, and on hom-objects it is simply the identity from

$$\mathsf{A}_{\mathsf{W}}(m_{\mathsf{W}}, n_{\mathsf{W}}) = n_{\mathsf{W}}^{m_{\mathsf{W}}} = (n^m)_{\mathsf{W}}$$

to

$$F(\mathsf{A}_{\mathsf{V}}(m_{\mathsf{V}}, n_{\mathsf{V}})) = F(n_{\mathsf{V}}^{m_{\mathsf{V}}}) = F((n^m)_{\mathsf{V}}) = (n^m)_{\mathsf{W}}$$

where we use Lemma 13 in these computations.

Using this we obtain a composite W-functor

$$\mathsf{A}_{\mathsf{W}} \xrightarrow{\tilde{F}} F_*(\mathsf{A}_{\mathsf{V}}) \xrightarrow{F_*(\tau_{\mathsf{V}})} F_*(\mathsf{T}).$$

This is the identity on objects and preserves finite V-products because each of the factors has these properties. It is thus a W-theory.

Theorem 8. Let V, W be cartesian closed categories with chosen finite coproducts of their terminal objects, and let $F\colon \mathsf{V} \to \mathsf{W}$ be a cartesian functor that preserves these chosen coproducts. Then F_* *preserves enriched theories*: that is, for every V-theory $\tau_{\mathsf{V}}\colon \mathsf{A}_{\mathsf{V}} \to \mathsf{T}$, the W-functor

$$\tau_{\mathsf{W}} := F_*(\tau_{\mathsf{V}}) \circ \tilde{F}\colon \mathsf{A}_{\mathsf{W}} \to F_*(\mathsf{T})$$

is a W-theory. Moreover, F_* *preserves models*: for every model $\mu\colon \mathsf{T} \to \mathsf{C}$ of $(\mathsf{T}, \tau_{\mathsf{V}})$, the W-functor $F_*(\mu)\colon F_*(\mathsf{T}) \to F_*(\mathsf{C})$ is a model of $(F_*(\mathsf{T}), \tau_{\mathsf{W}})$.

Proof. We have shown the first part. For the second, by Lemma 17 it suffices to assume that μ preserves finite N_V-powers and check that $F_*(\mu)$ preserves N_{W}-powers. We leave this as an exercise to the reader.

$\square$

Hence, any cartesian functor that preserves chosen finite coproducts of the terminal object gives a change of base. It thus provides for a method of translating formal languages between various "modes of operation". Moreover, this reasoning generalizes to *multisorted* V-theories, enriched theories which have multiple sorts: given any $n \in \mathbb{N}$, the monoidal subcategory $(N_V)^n$ is also an eleutheric system of arities, so Lucyshyn-Wright's monadicity theorem still applies.

5.2 Examples

Now let us look at some examples. The most important changes of base are the left adjoints in this diagram from Sec. 1:

$$\mathsf{sSet} \quad \overset{\text{FC}}{\underset{\text{UsS}}{\rightleftarrows}} \perp \quad \mathsf{Cat} \quad \overset{\text{FP}}{\underset{\text{UC}}{\rightleftarrows}} \perp \quad \mathsf{Pos} \quad \overset{\text{FS}}{\underset{\text{UP}}{\rightleftarrows}} \perp \quad \mathsf{Set}$$
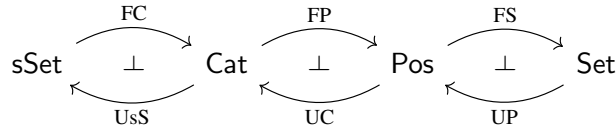
The first step describes the translation from small-step to big-step operational semantics. As already mentioned, we need to use simplicial sets rather than graphs; let us now say more about why.

A first attempt might use directed multigraphs. Such graphs have directed edges and allow multiple edges between any pair of vertices. The category Gph of directed multigraphs is Set^G where G is the category with two objects v and e and two morphisms $s, t \colon e \to v$. The "free category" functor $F \colon \mathsf{Gph} \to \mathsf{Cat}$ gives for every graph G a category $F(G)$ as follows:
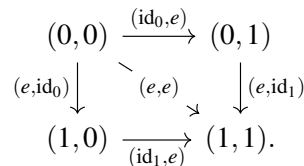
$$\begin{array}{rl} \text{objects} & \text{vertices of } G \\ \text{morphisms} & (e_1, e_2, ..., e_n) \colon s(e_1) \to t(e_n) \quad : \quad \forall i < n \ \ t(e_i) = s(e_{i+1}) \\ \text{composition} & (e_1, e_2, ..., e_m) \circ (e_1', e_2', ..., e_n') = (e_1', ..., e_n', e_1, ..., e_m) \ : \ t(e_n') = s(e_1). \end{array}$$

The morphisms in $F(G)$ are called **edge paths**. Just as an edge describes a single rewrite in small-step operational sematics, an edge path describes a sequence of rewrites in big-step operational semantics. The edge paths with $n = 0$ serve as identity morphisms.

Unfortunately, $F \colon \mathsf{Gph} \to \mathsf{Cat}$ does not preserve products, so it is not a valid base change. To see this, let G_1 be $\{0 \overset{e}{\to} 1\}$, the graph with two vertices and one edge. The product $G_1 \times G_1$ looks like this:

$$\begin{array}{cc} (0,0) & (0,1) \\ & \overset{(e,e)}{\searrow} \\ (1,0) & (1,1). \end{array}$$
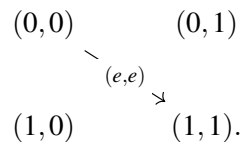
Thus the free category $F(G_1 \times G_1)$ has just one non-identity morphism. On the other hand $F(G_1) \times F(G_1)$ has five non-identity morphisms, shown here:

$$\begin{array}{ccc} (0,0) & \overset{(\mathrm{id}_0, e)}{\longrightarrow} & (0,1) \\ {\scriptstyle (e,\mathrm{id}_0)} \downarrow & \overset{(e,e)}{\searrow} & \downarrow {\scriptstyle (e,\mathrm{id}_1)} \\ (1,0) & \underset{(\mathrm{id}_1, e)}{\longrightarrow} & (1,1). \end{array}$$

where we write id for identity morphisms and e for the edge path consisting of the single edge e. Note that the triangles in this diagram commute. In terms of rewriting, the category $F(G_1 \times G_1)$ only allows

the rewrite $e\colon 0 \to 1$ to occur simultaneously in both factors, while $F(G_1) \times F(G_1)$ allows it to occur independently in either factor, in a commuting way.

To solve this problem one, might try to use reflexive graphs. Such graphs have directed edges and allows multiple edges between any pair of vertices; further, each vertex v is equipped with a distinguished **identity edge** $i(v)$ from v to itself. The category RGph of reflexive graphs is Set^R, where R is the category with two objects v and e, two morphisms $s, t\colon e \to v$, and a morphism $i\colon v \to e$ obeying $si = ti = 1_v$. There is a free category functor $F'\colon \mathsf{RGph} \to \mathsf{Cat}$, which is like the free category functor for Gph except that we identify an edge path $(e_1, \ldots, e_n)$ with the same path having e_i omitted when e_i is an identity edge. Thus, the identity edges of a reflexive graph R become identity morphisms in $F'(R)$.

The advantage of reflexive graphs is that they allow rewrites in a product to occur independently in either factor. For example, let R_1 be the reflexive graph with two vertices and one non-identity edge, $\{0 \xrightarrow{e} 1\}$ (where we do not draw identity edges). The product $R_1 \times R_1$ has five non-identity edges:

$$
\begin{array}{ccc}
(0,0) & \xrightarrow{(i(0),e)} & (0,1) \\
{\scriptstyle(e,i(0))}\Big\downarrow & {\scriptstyle(e,e)}\searrow & \Big\downarrow{\scriptstyle(e,i(1))} \\
(1,0) & \xrightarrow[(i(1),e)]{} & (1,1).
\end{array}
$$

Thus, the free category $F'(R_1 \times R_1)$ has two *noncommuting* triangles. On the other hand, $F'(R_1) \times F'(R_1)$ is the product of the category with a single non-identity morphism $e\colon 0 \to 1$ with itself, so it is this category:

$$
\begin{array}{ccc}
(0,0) & \xrightarrow{(\mathrm{id}_0,e)} & (0,1) \\
{\scriptstyle(e,\mathrm{id}_0)}\Big\downarrow & {\scriptstyle(e,e)}\searrow & \Big\downarrow{\scriptstyle(e,\mathrm{id}_1)} \\
(1,0) & \xrightarrow[(\mathrm{id}_1,e)]{} & (1,1)
\end{array}
$$

with two commuting triangles. Thus $F'\colon \mathsf{RGph} \to \mathsf{Cat}$ again fails to preserve products, though in some sense it comes closer. Simply put, while $F'(R_1 \times R_1)$ allows rewrites to be done independently in either factor, these rewrites fail to commute.

To solve this problem we shall consider RGph as a full subcategory of the category of simplicial sets, sSet. To do this, we treat a reflexive graph as a simplicial set with only degenerate simplices for $n > 1$. There is a left adjoint $FC\colon \mathsf{sSet} \to \mathsf{Cat}$, usually called **realization**, and this functor preserves products [16, Prop. B.0.15]. For example, if we treat R_1 above as a simplicial set and take the product $R_1 \times R_1$ in sSet, this contains triangles that force the triangles in $FC(R_1 \times R_1)$ to commute. Thus, realization provides a useful base change to translate from small-step operational semantics to big-step operational semantics.

The other functors in our chain of left adjoints are simpler. The "free poset" functor $FP\colon \mathsf{Cat} \to \mathsf{Pos}$ maps any category C to the poset whose elements are objects of C, with $c \leq c'$ iff C contains a morphism from c to c'. This is a valid change of base—i.e., it preserves finite products—because the product of posets is defined in the same way as the product of categories. If we apply this change of base to a model of a Cat-enriched theory, we obtain a model of a Pos-enriched theory that says for any pair of terms the presence or absence of a rewrite sequence from one to the other, without distinguishing between different sequences. We call this **full-step operational semantics**.

Finally, we can pass to the purely abstract realm where all computation is already complete. For this we take the left adjoint $FS\colon \mathsf{Pos} \to \mathsf{Set}$ to the functor $UP\colon \mathsf{Set} \to \mathsf{Pos}$ sending any set to the discrete

poset on that set. The functor FS collapses each connected component of the poset to a point; this too preserves finite products. If we apply this change of base to a model of a Pos-enriched theory, we obtain a model of a Set-enriched theory that extracts its denotational semantics by identifying all terms related by rewrites. If the rewrites are terminating and confluent, we can choose a representative term for each equivalence class: the unique term that admits no nontrivial rewrites.

6   The Category of All Models

In addition to base change, there are two other natural and useful ways to go between models of enriched theories. Suppose V is any cartesian closed category with chosen finite coproducts of the terminal object. Let $\mathsf{VMod}(\mathsf{T},\mathsf{C})$ be the category of models of a V-theory T in a V-category C with finite V-products, as in Defn. 4. A morphism of V-theories $f\colon \mathsf{T} \to \mathsf{T}'$ induces a **change of theory** functor between the respective categories of models

$$f^*\colon \mathsf{VMod}(\mathsf{T}',\mathsf{C}) \to \mathsf{VMod}(\mathsf{T},\mathsf{C})$$

defined as pre-composition with f. Similarly, a V-product-preserving V-functor $g\colon \mathsf{C} \to \mathsf{C}'$ induces a **change of context** functor

$$g_*\colon \mathsf{VMod}(\mathsf{T},\mathsf{C}) \to \mathsf{VMod}(\mathsf{T},\mathsf{C}')$$

defined as post-composition with g.

These translations, as well as change of base, can all be packed up nicely using the *Grothendieck construction*: given any functor $F\colon \mathsf{D} \to \mathsf{Cat}$, there is a category $\int F$ that encapsulates all of the categories in the image of F, defined as follows:

$$
\begin{array}{rl}
\text{objects} & (d,x)\colon d \in \mathsf{D},\ x \in F(d) \\
\text{morphisms} & (f\colon d \to d', a\colon F(f)(x) \to x') \\
\text{composition} & (f,a) \circ (f',a') = (f \circ f', a \circ F(f)(a')).
\end{array}
$$

Moreover there is a functor $p_F\colon \int F \to \mathsf{D}$ given as follows:

$$
\begin{array}{rl}
\text{on objects} & p_F\colon (d,x) \mapsto d \\
\text{on morphisms} & p_F\colon (f,a) \mapsto f.
\end{array}
$$

For more details see [7, 15]. We noted in Section 4 that VLaw and $\mathsf{Mod}(\mathsf{T},\mathsf{C})$ can be promoted to V-categories when V is complete and cocomplete: this and further conditions imply that we can use the enriched Grothendieck construction [5], but we focus on the ordinary Grothendieck construction for simplicity.

First, this construction lets us bring together all models of all different V-theories in all different contexts into one category. All the V-theories are objects of VLaw, as in Defn. 3. We can also create a category of all "V-contexts".

Definition 9. Let VCon, the **category of V-contexts** be the category for which an object is a V-category with finite V-products and a morphism is a functor that preserves finite V-products.

There is a functor

$$\mathsf{VMod}\colon \mathsf{VLaw}^{\mathrm{op}} \times \mathsf{VCon} \to \mathsf{Cat}$$

that sends any object (T,C) to $\mathsf{VMod}(\mathsf{T},\mathsf{C})$ and any morphism (f,g) to $f^*g_* = g_*f^*$. The functoriality of VMod summarizes the contravariant change-of-theory and the covariant change-of-context above.

Applying the Grothedieck construction we obtain a category $\int$ VMod. Technically an object of $\int$ VMod is a triple (T, C, μ), but more intuitively it is a model $\mu: T \to C$ of any V-theory T in any V-context C. Similarly, a morphism

$$(f, g, \alpha): (T, C, \mu) \to (T', C', \mu')$$

in VMod consists of:

- a morphism of V-theories $f: T' \to T$,

- a V-functor $g: C \to C'$ that preserves finite V-products, and

- a V-natural transformation $\alpha: g \circ \mu \circ f \Rightarrow \mu'$.

This is a natural way to map between different models of different theories in different contexts.

We can go further by creating a category that even contains all choices of enriching categories V:

Definition 10. Let Enr be the category for which an object is a cartesian closed category V with chosen finite coproducts of the terminal object, and a morphism is a cartesian functor $F: V \to W$ preserving the chosen finite coproducts of the initial object.

There is a functor

$$\text{Mod}: \text{Enr} \to \text{Cat}$$

that maps any object V to $\int$ VMod and any morphism $F: V \to W$ to a functor

$$\text{Mod}(F): \int \text{VMod} \to \int \text{WMod}$$

that has the following effect:

- $\text{Mod}(F)$ maps any object (T, C, μ) to the object $(F_*(T), F_*(C), F_*(\mu))$.

- $\text{Mod}(F)$ maps any morphism (f, g, α) to the morphism $(F_*(f), F_*(g), F_*(\alpha))$.

Thus, we can use the Grothendieck construction once more to pack up all choices of enrichment into one big category:

Theorem 11. There is a category $\int$ Mod in which:

- An object is a choice of cartesian closed category V with chosen finite coproducts of the terminal object, a V-theory T, a V-category C with finite V-products, and a model $\mu: T \to C$.

- A morphism is a cartesian functor $F: V \to W$ preserving the chosen finite coproducts of the terminal object and a morphism $(f, g, \alpha): (F_*(T), F_*(C), F_*(\mu)) \to (T, C, \mu)$ in WMod.

This category allows us to formally treat morphisms between objects of "different kinds", something we often use informally, for example when speaking of a map from a set to a ring, or a group to a topological group. There are many unexplored questions about the large, heterogeneous categories which arise from the Grothendieck construction, regarding what unusual structure may be gained, such as limits and colimits with objects of different types, or identifying "processes" in which the kinds of objects change in an essential way. However, for our purposes we need only recognize that enriched Lawvere theories can be assimilated into one category, providing a single place in which to study change of base, change of theory, and change of context.

7 Applications

In computer science literature, enriched algebraic theories have primarily been studied in the context of "computational effects" [13]. Stay and Meredith have proposed that enriched Lawvere theories can be utilized for the design of programming languages [31]. To circumvent the question of *variable binding*, there is another approach which instead uses an enriched theory as a "compiler" which translates a language with binding to one without. This idea comes from the subject of combinatory logic.

7.1 The *SKI*-combinator calculus

The λ-calculus is an elegant formal language which is the foundation of functional computation, the model of intuitionistic logic, and the internal logic of cartesian closed categories: this is the Curry–Howard–Lambek correspondence [3].

Terms are constructed recursively by *variables*, *application*, and *abstraction*, and the basic rewrite is *beta reduction*, which substitutes the applied term for the bound variable:

$$M,N := x \mid (M\,N) \mid \lambda x.M$$

$$(\lambda x.M\,N) \Rightarrow M[N/x].$$

Despite the apparent simplicity, there are complications regarding substitution. Consider the term $M = \lambda x.(\lambda y.(xy))$: if this is applied to the variable y, then $(M\,y) \Rightarrow \lambda y.(y\,y)$ — but this is not intended, because the y in M is just a placeholder, it is "bound" by whatever will be plugged in, while the y being substituted is "free", meaning it can refer to some other value or function in the program. Hence whenever a free variable is to be substituted for a bound variable, we need to rename the bound variable to prevent "variable capture" (e.g. $(My) \Rightarrow \lambda z.(y\,z)$).

This problem was noticed early in the history of mathematical foundations, even before the λ-calculus, and so Moses Schönfinkel invented **combinatory logic** [28], a basic form of logic without the red tape of variable binding, hence without functions in the usual sense. The *SKI*-calculus is the "variable-free" representation of the λ-calculus; λ-terms are translated via "abstraction elimination" into strings of combinators and applications. This is a technique for programming languages to minimize the subtleties of variables.

The insight of Stay and Meredith [30] is that even though enriched Lawvere theories have no variables, they can be used to study some programming languages through abstraction elimination. When representing such a language as a sSet-theory, vertices—i.e., 0-simplices—in the simplicial set $\hom(1,t)$ serve as closed terms. More generally, vertices in $\hom(t^n,t)$ serve as terms with n free variables. Rewrite rules going between such terms are edges—i.e., 1-simplices—in $\hom(t^n,t)$.

To illustrate this, here is the theory of the *SKI*-calculus:

$$\mathsf{Th}(\mathsf{SKI})$$

type	t
term constructors	$S\colon\quad 1 \to t$
	$K\colon\quad 1 \to t$
	$I\colon\quad 1 \to t$
	$(-\ -)\colon\quad t^2 \to t$
structural congruence	n/a
rewrites	$\sigma\colon\quad (((S-)\ =)\ \equiv) \Rightarrow ((-\ \equiv)\,(=\equiv))$
	$\kappa\colon\quad ((K-)\ =) \Rightarrow -$
	$\iota\colon\quad (I-) \Rightarrow -$

These rewrites are implicitly universally quantified; i.e., they apply to arbitrary subterms $-, =, \equiv$ without any variable binding involved, by using the cartesian structure of the category. They are edges with vertices as follows:

$$(((S-)\ =)\ \equiv)\colon \quad t^3 \xrightarrow{l^{-1}\times t^3} 1\times t^3 \xrightarrow{S\times t^3} t^4 \xrightarrow{(-\,-)\times t^2} t^3 \xrightarrow{(-\,-)\times t} t^2 \xrightarrow{(-\,-)} t$$
$$\sigma\Big\downarrow \qquad\qquad \| \qquad\qquad\qquad\qquad \Downarrow \qquad\qquad\qquad\qquad \|$$
$$((-\ \equiv)\,(=\equiv))\colon \quad t^3 \xrightarrow{t^2\times\Delta} t^4 \xrightarrow{t\times s\times t} t^4 \xrightarrow{(-\,-)\times(-\,-)} t^2 \xrightarrow{(-\,-)} t$$

$$((K-)\ =)\colon \quad t^2 \xrightarrow{l^{-1}\times t^2} 1\times t^2 \xrightarrow{K\times t^2} t^3 \xrightarrow{(-\,-)\times t} t^2 \xrightarrow{(-\,-)} t$$
$$\kappa\Big\downarrow \qquad\qquad \| \qquad\qquad\qquad \Downarrow \qquad\qquad\qquad \|$$
$$-\colon \quad t^2 \xrightarrow{\quad t\times !\quad} t\times 1 \xrightarrow{\quad r\quad} t$$

$$(I-)\colon \quad t \xrightarrow{l^{-1}} 1\times t \xrightarrow{I\times t} t^2 \xrightarrow{(-\,-)} t$$
$$\iota\Big\downarrow \qquad\qquad \| \qquad\qquad \Downarrow \qquad\qquad \|$$
$$-\colon \quad t \xrightarrow{\qquad\qquad t\qquad\qquad} t$$

Here l, r denote the unitors and s the symmetry of the product.

These abstract rules are evaluated on concrete terms by "plugging in" via precomposition. For example:

$$((KS)I)\colon \quad 1 \xrightarrow{S\times I} t^2 \xrightarrow{((K-)\ =)} t$$
$$\kappa\circ(S\times I)\Big\downarrow \qquad\qquad \| \qquad\qquad \Downarrow \qquad\qquad \|$$
$$S\colon \quad 1 \xrightarrow{S\times I} t^2 \xrightarrow{\quad-\quad} t$$

A model of this theory is a sSet-functor $\mu\colon \mathsf{Th}(\mathsf{SKI}) \to \mathsf{sSet}$ that preserves finite sSet-products. This gives a simplicial set $\mu(t)$. The images of the nullary operations S, K, I under μ are distinguished vertices of $\mu(t)$, because μ preserves the terminal object, which "points out" vertices. The image of the binary operation $(-\ -)$ gives for every pair of vertices $(u, v) \in \mu(t)^2$ a vertex $(u\ v)$ in $\mu(t)$ which stands for their application. In this way all possible terms built from S, K, I and application give vertices in $\mu(t)$. Similarly, rewrites going between these terms give edges in $\mu(t)$. Thus, μ gives a map of simplicial sets

$$\mu_{1,t}\colon \mathsf{Th}(\mathsf{SKI})(1,t) \to \mathsf{sSet}(1,\mu(t))$$

that maps the "syntactic" graph of all closed terms and rewrites to the "semantic" graph: each rewrite between terms in the theory is sent to a rewrite between the images of these terms in the model.

The fact that $\mu((--)): \mu(t)^2 \to \mu(t)$ is not just a function but a map of simplicial sets means that pairs of edges $(a \to b, c \to d)$ in $\mathsf{Th}(\mathsf{SKI})(1,t)$ are sent to edges $(a\ b) \to (c\ d)$ in $\mathsf{sSet}(1,\mu(t))$. This gives the full complexity of the theory: given a large term (program), there are many different ways it can be computed—and some take fewer steps than others:

$$((K\ S)\ (((S\ K)\ I)\ (I\ K))) \xrightarrow{((K\ S)\ \sigma)} ((K\ S)\ ((K\ (I\ K))\ (I\ (I\ K))))$$

$$\downarrow {\scriptstyle (((K\ S)\ \iota)\ (I\ (I\ K)))}$$

$$((K\ S)\ ((K\ K)\ (I\ (I\ K))))$$

$$\downarrow {\scriptstyle ((K\ S)\ ((K\ K)\ (I\ \iota)))}$$

$$\kappa \qquad\qquad ((K\ S)\ ((K\ K)\ (I\ K)))$$

$$\downarrow {\scriptstyle ((K\ S)\ ((K\ K)\ \iota))}$$

$$((K\ S)\ ((K\ K)\ K))$$

$$\downarrow {\scriptstyle ((K\ S)\ \kappa)}$$

$$S \xleftarrow{\quad\kappa\quad} ((K\ S)\ K)$$

More generally, the image $\mu(t)^n$ is a simplicial set whose vertices are *SKI*-terms with n free variables and whose edges are n-tuples of rewrites between such terms. This is because the enriched functor μ gives maps of simplicial sets

$$\mu_{t^n,t}: \mathsf{Th}(\mathsf{SKI})(t^n,t) \to \mathsf{sSet}(\mu(t)^n,\mu(t)).$$

As the n-ary operations and rewrites thereof are built up from application and the three rewrites, everything works the same way as in the case $n = 0$.

This process is intuitive, but how do we actually define the model, as a functor, to pick out a specific graph? There are many models of $\mathsf{Th}(\mathsf{SKI})$, but in particular we care about the canonical *free* model, which means that $\mu(t)$ is simply the graph of all closed terms and rewrites in the *SKI*-calculus. This utilizes the enriched adjunction of Thm. 5:

$$\underline{\mathsf{sSet}} \quad\overset{f_{\mathsf{sSet}}}{\underset{u_{\mathsf{sSet}}}{\rightleftarrows}}\quad \bot \quad \underline{\mathsf{Mod}}(\mathsf{Th}(\mathsf{SKI}),\mathsf{sSet})$$

Then the canonical model of closed terms and rewrites is simply the free model on the empty graph, $f_{\mathsf{sSet}}(\emptyset)$, i.e. the V-functor $\mathsf{T}(1,-): \mathsf{T} \to \mathsf{V}$. Hence for us, the syntax and semantics of the *SKI* combinator calculus are unified in the model

$$\mu^{\mathsf{sSet}}_{SKI} := \mathsf{Th}(\mathsf{SKI})(1,-): \mathsf{Th}(\mathsf{SKI}) \to \mathsf{sSet}.$$

Here we reap the benefits of the abstract construction: the graph $\mu^{\mathsf{sSet}}_{SKI}(t)$ represents the small-step operational semantics of the *SKI*-calculus:

$$(\mu(a) \to \mu(b) \in \mu^{\mathsf{sSet}}_{SKI}(t)) \iff (a \Rightarrow b \in \mathsf{Th}(\mathsf{SKI})(1,t)).$$

We can now consider the base changes in Sec. 5.2, to translate between several important kinds of computation for the *SKI*-calculus. Given the above description of $\mathsf{Th}(\mathsf{SKI})$ as enriched in sSet, we can apply the "free category" realization functor to the hom-objects, turning these reflexive graphs into categories.

Here we enjoy the fact that this functor indeed preserves products, which is essential for considering tuples of programs running in parallel: for example if we designate $G_n := \mathsf{Th}(\mathsf{SKI})(t^n, t)$, then the fact that $\mathrm{FC}(G_m \times G_n) \cong \mathrm{FC}(G_m) \times \mathrm{FC}(G_n)$ ensures that the execution of an *m*-term program and an *n*-term program simultaneously (but independently) is the same as executing one, then the other.

Thus FC translates the theory of *SKI* from "small-step" to "big-step" operational semantics: $\mathrm{FC}_*(\mathsf{Th}(\mathsf{SKI}))$ is the theory of the *SKI* calculus, but now with hom-categories whose morphisms represent finite sequences of rewrite edges in the original theory.

We can continue these base-changes to "full-step" and denotational semantics, by applying the "free poset" and "free set" (connected components) functors to the hom-objects of this theory. This process demonstrates the idea of having a "spectrum" of detail with which to analyze the semantics of a programming language, or general algebraic theory.

For example, consider the following computation:
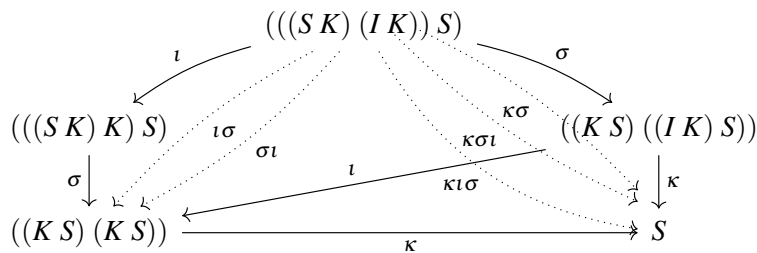
The solid arrows are the one-step rewrites of the initial sSet-theory; applying FC_* gives the dotted composites, and FP_* asserts that all composites between any two objects are equal. Finally, FS_* collapses the whole diagram to *S*. This is a simple demonstration of the basic stages of computation: small-step, big-step, full-step, and denotational semantics.

7.2   Change of theory

We can equip term calculi with *reduction contexts*, which determine when rewrites are valid, thus giving the language a certain **evaluation strategy**. For example, the "weak head normal form" is given by only allowing rewrites on the left-hand side of the term.

We can do this for $\mathsf{Th}(\mathsf{SKI})$ by adding a reduction context marker as a unary operation, and a structural congruence rule which pushes the marker to the left-hand side of an application; lastly we modify the rewrite rules to be valid only when the marker is present:

$$\mathsf{Th}(\mathsf{SKI} + \mathsf{R})$$

$$
\begin{array}{lll}
\textbf{sort} & & t \\
\textbf{term constructors} & S,K,I\colon & 1 \to t \\
& R\colon & t \to t \\
& (-\;-)\colon & t^2 \to t \\
\\
\textbf{structural congruence} & R(-\;=) & = (R-\;=) \\
& RR & = R \\
\\
\textbf{rewrites} & \sigma_r\colon & (((RS-)\;=)\;\equiv) \Rightarrow ((R-\;\equiv)\,(=\;\equiv)) \\
& \kappa_r\colon & ((RK-)\;=) \Rightarrow R- \\
& \iota_r\colon & (RI-) \Rightarrow R-
\end{array}
$$

The *SKI*-calculus is thereby equipped with "lazy evaluation", an essential paradigm in modern programming. This represents a broad potential application of equipping theories with computational methods, such as evaluation strategies.

Moreover, these equipments can be added or removed as needed: using change-of-theory, we can utilize a "free reduction" sSet-functor $f_R\colon \mathsf{Th}(\mathsf{SKI}) \to \mathsf{Th}(\mathsf{SKI}+\mathsf{R})$:

$$
\begin{array}{lll}
\text{objects} & t^n & \mapsto & t^n \\
\text{hom-vertices} & S,K,I & \mapsto & S,K,I \\
& (-\;-) & \mapsto & R(-\;-) \\
\text{hom-edges} & \sigma,\kappa,\iota & \mapsto & \sigma_r,\kappa_r,\iota_r
\end{array}
$$

This essentially interprets ordinary *SKI* as having every subterm be a reduction context. This is a sSet-functor because its hom component consists of graph-homomorphisms

$$
f_{n,m}\colon \mathsf{Th}(\mathsf{SKI})(t^n,t^m) \to \mathsf{Th}(\mathsf{SKI}+\mathsf{R})(t^n,t^m)
$$

which simply send each application to its postcomposition with R, and each rewrite to its "marked" correspondent.

So, by precomposition this induces the change of theory on categories of models:

$$
f_R^*\colon \mathsf{Mod}(\mathsf{Th}(\mathsf{SKI}+\mathsf{R}),\mathsf{C}) \to \mathsf{Mod}(\mathsf{Th}(\mathsf{SKI}),\mathsf{C})
$$

for all semantic categories C, which forgets the reduction contexts.

Similarly, there is a sSet-functor $u_R\colon \mathsf{Th}(SKI+R) \to \mathsf{Th}(SKI)$ which forgets reduction contexts, by sending $\sigma_r,\kappa_r,\iota_r \mapsto \sigma,\kappa,\iota$ and $R \mapsto id_t$; this latter is the only way that the marked reductions can be mapped coherently to the unmarked. However, this means that u_R^* does not give the desired change-of-theory of "freely adjoining contexts", because collapsing R to the identity eliminates the significance of the marker.

This illustrates a key aspect of categorical universal algebra: because change-of-theory is given by precomposition and is thus contravariant, *properties* (equations) and *structure* (operations) can only be removed. This is a necessary limitation, at least in the present setup, but there are ways to make do. These abstract theories are not floating in isolation but are implemented in code: one can simply use a "maximal theory" with all pertinent structure, then selectively forget as needed.

8   Conclusion

We have shown how enriched Lawvere theories provide a framework for unifying the structure and behavior of formal languages. Enriching theories in category-like structures reifies operational semantics by incorporating rewrites between terms, and appropriate functors between enriching categories induce change-of-base functors between categories of enriched theories and models—this simplified condition is obtained by using only natural number arities. This idea is motivated by an example sequence of such functors, which provide a spectrum of detail in which to study the rewriting properties of a theory.

Change of base, along with change of theory and change of context, can be used to create a single category Mod, which consists of all models of all enriched Lawvere theories in all contexts. We have demonstrated these concepts with the theory of combinatory logic, $\mathsf{Th}(\mathsf{SKI})$, describing a change of base from small-step operational semantics to big-step to full-step to denotational semantics.

Finally, we suggest that there are many interesting change-of-base functors, by considering an endofunctor on the category of labelled transition systems, which quotients by the bisimulation relation and is indeed a change of base.

A Natural Number Arities

In this appendix we prove the lemmas required for Theorem 5 and our study of base change in Section 5. Throughout we assume V is cartesian closed with chosen n-fold coproducts n_{V} of its terminal object.

We begin with a study of N_{V}, the full subcategory of V on the objects n_{V}. First we must resolve a potential ambiguity. On the one hand, for any object b of V we can form the exponential $b^{n_{\mathsf{V}}}$. On the other hand, we can take the product of n copies of b, which we call b^n. Luckily these are the same, or at least naturally isomorphic:

Lemma 12. The functors $(-)^{n_{\mathsf{V}}}\colon \mathsf{V} \to \mathsf{V}$ and $(-)^n \colon \mathsf{V} \to \mathsf{V}$ are naturally isomorphic.

Proof. If $a, b \in \mathsf{V}$, then

$$
\begin{aligned}
\mathsf{V}(a, b^{n_{\mathsf{V}}}) &\cong \mathsf{V}(a \times n_{\mathsf{V}}, b) && \text{hom-tensor adjunction} \\
&= \mathsf{V}(a \times (n \cdot 1_{\mathsf{V}}), b) && \text{definition of } n_{\mathsf{V}} \\
&\cong \mathsf{V}(n \cdot (a \times 1_{\mathsf{V}}), b) && \text{products distribute over coproducts} \\
&\cong \mathsf{V}(n \cdot a, b) && \text{unitality} \\
&\cong \mathsf{V}(a, b)^n && \text{definition of coproduct} \\
&\cong \mathsf{V}(a, b^n) && \text{definition of product.}
\end{aligned}
$$

Each of these isomorphisms is natural in a and b, so by the Yoneda lemma $(-)^{n_{\mathsf{V}}} \cong (-)^n$. $\qquad\square$

We can now understand coproducts, products and exponentials in N_{V}:

Lemma 13. If V is any cartesian closed category with chosen coproducts of the initial object then N_{V} is cartesian closed, with finite coproducts. The unique initial object of N_{V} is 0_{V}. The binary coproducts in N_{V} are unique, given by

$$m_{\mathsf{V}} + n_{\mathsf{V}} = (m+n)_{\mathsf{V}}.$$

The unique terminal object of N_{V} is 1_{V}, and the binary products are unique, given by

$$m_{\mathsf{V}} \times n_{\mathsf{V}} = (mn)_{\mathsf{V}}.$$

Exponentials in N_{V} are also unique, given by

$$m_{\mathsf{V}}^{n_{\mathsf{V}}} = (m^n)_{\mathsf{V}}.$$

Proof. In V we know that 0_{V} is an initial object and 1_{V} is a terminal object, by definition. Since the subcategory N_{V} is skeletal 0_{V} is the unique initial object and 1_{V} is the unique terminal object in N_{V}. Similarly, in V we have defined $(m+n)_{\mathsf{V}}$ to be a coproduct of m_{V} and n_{V}, so in N_{V} it is the unique such, and we can unambiguously write

$$m_{\mathsf{V}} + n_{\mathsf{V}} = (m+n)_{\mathsf{V}}.$$

Products distribute over coproducts in any cartesian closed category, so in V we have

$$m_{\mathsf{V}} \times n_{\mathsf{V}} \cong (1_{\mathsf{V}} + \cdots + 1_{\mathsf{V}}) \times (1_{\mathsf{V}} + \cdots + 1_{\mathsf{V}}) \cong (mn)_{\mathsf{V}}$$

where in the second step we use the distributive law twice. It follows that N_{V} has finite products, and since this subcategory is skeletal they are unique, given by

$$m_{\mathsf{V}} \times n_{\mathsf{V}} = (mn)_{\mathsf{V}}.$$

Finally, by Lemma 12 we have

$$m_{\mathsf{V}}{}^{n_{\mathsf{V}}} \cong m_{\mathsf{V}}^{n} \cong \prod_{i=1}^{n} m_{\mathsf{V}} \cong (m^{n})_{\mathsf{V}}.$$

It follows that $\mathsf{N_V}$ has exponentials, and since this subcategory is skeletal they are unique, given by

$$m_{\mathsf{V}}^{n_{\mathsf{V}}} = (m^{n})_{\mathsf{V}}. \qquad \square$$

We warn the reader that $\hom(m_{\mathsf{V}}, n_{\mathsf{V}})$ may not have n^m elements. It does in $\mathsf{sSet}, \mathsf{Cat}, \mathsf{Pos}$ and of course Set, but not in $\mathsf{V} = \mathsf{Set}^k$, where $|\hom(m_{\mathsf{V}}, n_{\mathsf{V}})| = n^{km}$. In fact, whenever $\mathsf{N_V}$ has finite hom-sets it is equivalent to FinSet^k for some k. The reason is that 2_{V} is an internal Boolean algebra in V, so its set of elements $\hom(1_{\mathsf{V}}, 2_{\mathsf{V}})$ must be some Boolean algebra B in Set. A further argument due to Garner and Trimble shows that $\mathsf{N_V}$ is completely characterized, up to equivalence, by this Boolean algebra, and any Boolean algebra can occur [2]. If this Boolean algebra is finite it must be isomorphic to $\{0,1\}^k$ for some $k \geq 0$. In this case, $\mathsf{N_V}$ is equivalent to FinSet^k.

Now suppose C is a V-category. The question arises whether the power of an object $c \in \mathsf{C}$ by n_{V} must also be the V-product of n copies of c. The answer is yes:

Lemma 14. Let C be a V-category and $c \in \mathrm{Ob}(\mathsf{C})$. Then the power $c^{n_{\mathsf{V}}}$ exists if and only if the n-fold V-product c^n exists, in which case they are isomorphic.

Proof. In Section 3 we saw that an object $b \in \mathrm{Ob}(\mathsf{C})$ is an n-fold V-product of copies of c precisely when it is equipped with a universal cone

$$p \colon 1_{\mathsf{V}} \to \mathsf{C}(b,c)^n.$$

Similarly, b is an n_{V}-power of c when it is equipped with a universal cone

$$q \colon 1_{\mathsf{V}} \to \mathsf{C}(b,c)^{n_{\mathsf{V}}}.$$

The universality properties have the same form, and by Lemma 12 the functors $(-)^n \colon \mathsf{V} \to \mathsf{V}$ and $(-)^{n_{\mathsf{V}}} \colon \mathsf{V} \to \mathsf{V}$ are naturally isomorphic. Thus, given either sort of universal cone we get the other, so an object is an n-fold product of copies of c if and only if it is the n_{V}-power of c. $\qquad \square$

Lemma 15. Suppose C is a V-category such that every object is the n-fold V-product c^n of some object c. Then a V-functor $F \colon \mathsf{C} \to \mathsf{D}$ preserves finite V-products if and only if it preserves powers by all objects of $\mathsf{N_V}$.

Proof. Define a "finite V-power" to be a finite V-product of n copies of the same object. The V-functor F preserves finite V-powers if and only if it maps any universal cone

$$p \colon 1_{\mathsf{V}} \to \mathsf{C}(b,c)^n$$

in C to a universal cone in D. Similarly, F preserves powers by all objects of $\mathsf{N_V}$ if and only if it maps any universal cone

$$q \colon 1_{\mathsf{V}} \to \mathsf{C}(b,c)^{n_{\mathsf{V}}}$$

in C to a universal cone in D. Two kinds of universality are involved here, but since they have the same form, and since Lemma 12 says the functors $(-)^n \colon \mathsf{V} \to \mathsf{V}$ and $(-)^{n_{\mathsf{V}}} \colon \mathsf{V} \to \mathsf{V}$ are naturally isomorphic, it follows that F preserves finite V-powers if and only if it preserves powers by all objects of $\mathsf{N_V}$.

It thus suffices to show that F preserves finite V-products if and only if it preserves finite V-powers. This follows from the assumption that every object is the n-fold V-product c^n of some object c. $\qquad \square$

Lemma 16. Let V be cartesian closed with chosen finite coproducts of the terminal object and let T be a V-category. These conditions for a V-functor $\tau\colon A_V \to T$ are equivalent:

1. (T, τ) is a V-theory,

2. τ preserves finite V-products,

3. τ preserves powers by objects of N_V.

Proof. Conditions 1 and 2 are equivalent by definition. Since $A_V = \underline{N}_V^{op}$, finite V-products in A_V are the same as finite V-coproducts in $\underline{N}_V$, which are the same as finite coproducts in N_V. Since every object in $\underline{N}_V$ is a finite coproduct of copies of 1_V, Lemma 15 implies that conditions 2 and 3 are equivalent. □

Lemma 17. Given a V-theory (T, τ) and a V-functor $\mu\colon T \to C$, the following conditions are equivalent:

- μ is a model of (T, τ),

- μ preserves finite V-products,

- μ preserves powers by objects of N_V.

Proof. Conditions 1 and 2 are equivalent by definition. Since τ is the identity on objects and preserves V-products each object of T is of the form t^n where $t = \tau(1_V)$. Thus, Lemma 15 implies that conditions 2 and 3 are equivalent. □

References

[1] J. Adámek & J. Rosický (1994): *Locally Presentable and Accessible Categories*. *London Mathematical Society Lecture Note Series* 189, Cambridge University Press, Cambridge, doi:10.1017/CBO9780511600579.

[2] J. C. Baez (2019): *Can 1+1 have more than two points?* The *n*-Category Café. Available at `https://golem.ph.utexas.edu/category/2019/04/can_11_have_more_than_two_poin.html`.

[3] J. C. Baez & M. Stay (2011): *Physics, topology, logic and computation: a Rosetta Stone*. In B. Coecke, editor: *New Structures for Physics*, Springer, Berlin, pp. 95–172, doi:10.1007/978-3-642-12821-9. Available at `https://arxiv.org/abs/0903.0340`.

[4] M. Barr & C. Wells (1984): *Toposes, Triples and Theories*. *Grundlehren der mathematischen Wissenschaften* 278, Springer, Berlin, doi:10.4204/EPTCS. Available at `https://www.math.mcgill.ca/barr/papers/ttt.pdf`.

[5] J. Beardsley & L. Z. Wong (2019): *The enriched Grothendieck construction*. *Advances in Mathematics* 344, pp. 234 – 261, doi:10.1016/j.aim.2018.12.009. Available at `https://arxiv.org/abs/1804.03829`.

[6] R. Blackwell, G. M. Kelly & A. J. Power (1989): *Two-dimensional monad theory*. *Journal of Pure and Applied Algebra* 59(1), pp. 1–41, doi:10.1016/0022-4049(89)90160-6.

[7] F. Borceux (1994): *Handbook of Categorical Algebra*. Cambridge University Press, Cambridge, doi:10.1112/BLMS/28.4.440.

[8] R. Crole (1994): *Categories for Types*. Cambridge University Press, Cambridge, doi:10.1017/CBO9781139172707.

[9] B. Day & R. Street (1997): *Monoidal bicategories and Hopf algebroids*. *Advances in Mathematics* 129(1), pp. 99–157, doi:10.1006/aima.1997.1649.

[10] E. J. Dubuc (1970): *Kan Extensions in Enriched Category Theory*. *Lecture Notes in Mathematics* 145, Springer, Berlin, doi:10.1007/bfb0060485.

[11] M. Fiore, G. Plotkin & D. Turi (1999): *Abstract syntax and variable binding*. In: *Proceedings, 14th Symposium on Logic in Computer Science*, pp. 193–202.

[12] G. Friedman (2012): *An elementary illustrated introduction to simplicial sets*. Rocky Mountain Journal of Mathematics 42(2), pp. 353–423, doi:10.1216/rmj-2012-42-2-353.

[13] M. Hyland & J. Power (2006): *Discrete Lawvere theories and computational effects*. Theoretical Computer Science 366(1-2), pp. 144–162, doi:10.1016/j.tcs.2006.07.007. Available at `https://core.ac.uk/download/pdf/81105779.pdf`.

[14] M. Hyland & J. Power (2007): *The category theoretic understanding of universal algebra: Lawvere theories and monads*. Electronic Notes in Theoretical Computer Science 172, pp. 437–458, doi:10.1016/j.entcs.2007.02.019.

[15] B. Jacobs (1998): *Categorical Logic and Type Theory*. Elsevier, Amsterdam, doi:10.1016/s0049-237x(98)x8028-6.

[16] A. Joyal (2008): *The theory of quasicategories and its applications*. Available at `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.154.4968`.

[17] P. Karazeris & G. Protsonis (2012): *Left Kan extensions preserving finite products*. Journal of Pure and Applied Algebra 216(8-9), pp. 2014–2028, doi:10.1016/j.jpaa.2012.02.038.

[18] G. Kelly (2005): *The Basic Concepts of Enriched Category Theory*. Reprints in Theory and Applications of Categories 10, doi:10.1112/blms/15.1.96. Available at `http://www.tac.mta.ca/tac/reprints/articles/10/tr10abs.html`.

[19] Y. Kinoshita, J. Power & M. Takeyama (1999): *Sketches*. Journal of Pure and Applied Algebra 143(1-3), pp. 275–291, doi:10.1016/s0022-4049(98)00114-5.

[20] F. W. Lawvere (1963): *Functorial semantics of algebraic theories*. Proceedings of the National Academy of Sciences 50(5), pp. 869–872, doi:10.1073/pnas.50.5.869. Available at `http://tac.mta.ca/tac/reprints/articles/5/tr5abs.html`.

[21] F. E. J. Linton (1966): *Some aspects of equational categories*. In: Proceedings of the Conference on Categorical Algebra, Springer, pp. 84–94, doi:10.1007/978-3-642-99902-4_3.

[22] R. B. B. Lucyshyn-Wright (2015): *Enriched algebraic theories and monads for a system of arities*. Theory and Applications of Categories 31(5). Available at `http://www.tac.mta.ca/tac/volumes/31/5/31-05abs.html`.

[23] C. Lüth & N. Ghani (1997): *Monads and modular term rewriting*. In: Category Theory and Computer Science, Springer, pp. 69–86, doi:10.1007/bfb0026982. Available at `http://www.informatik.uni-bremen.de/~cxl/papers/ctcs97l.pdf`.

[24] B. Milewski (2017): *Category Theory for Programmers*. Available at `https://bartoszmilewskiski.com/2017/08/26/lawvere-theories/`.

[25] K. Nishizawa & J. Power (2009): *Lawvere theories enriched over a general base*. Journal of Pure and Applied Algebra 213(3), pp. 377–386, doi:10.1016/j.jpaa.2008.07.009.

[26] G. D. Plotkin (2004): *A structural approach to operational semantics*. Journal of Logical and Algebraic Methods in Programming 60-61, pp. 17–139, doi:10.1016/j.jlap.2004.05.001. Available at `http://homepages.inf.ed.ac.uk/gdp/publications/sos_jlap.pdf`.

[27] J. Power (1999): *Enriched Lawvere theories*. Theory and Applications of Categories 6(7), pp. 83–93. Available at `http://www.tac.mta.ca/tac/volumes/6/n7/6-07abs.html`.

[28] M. Schönfinkel (1924): *Bausteine zu einer Logik der mathematischen Wissenschaften*. Mathematische Annalen 92, pp. 305–316. Available at `http://www.digizeitschriften.de/dms/img/?PID=GDZPPN002270110`.

[29] R. A. G. Seely (1987): *Modelling computations: a 2-categorical framework*. In: Proceedings of the Second Annual IEEE Symposium on Logic in Computer Science, IEEE Computer Society Press, Ithaca, New York, pp. 22–25.

[30] M. Stay & L. G. Meredith (2017): *Representing operational semantics with enriched Lawvere theories*. Available at `https://arxiv.org/abs/1704.03080`.

[31] M. Stay & L. G. Gregory Meredith (2016): *Logic as a distributive law*. Available at `https://arxiv.org/abs/1610.02247`.