Combining Effects: Sum and Tensor

Citation for published version:
Hyland, JME, Plotkin, G & Power, AJ 2006, 'Combining Effects: Sum and Tensor', *Theoretical Computer Science*, vol. 357, no. 1-3, pp. 70-99. https://doi.org/10.1016/j.tcs.2006.03.013

Digital Object Identifier (DOI):
10.1016/j.tcs.2006.03.013

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Theoretical Computer Science

Combining effects: sum and tensor

Martin Hyland,[1] Gordon Plotkin[2] and John Power[2] ⋆

[1] Dept. of Mathematics, University of Cambridge, Cambridge CB3 0WB, England.
email: M.Hyland@dpmms.cam.ac.uk

[2] Laboratory for the Foundations of Computer Science, School of Informatics, University of Edinburgh, King's Buildings, Edinburgh EH9 3JZ, Scotland.
email: gdp@inf.ed.ac.uk, ajp@inf.ed.ac.uk

Abstract. We seek a unified account of modularity for computational effects. We begin by reformulating Moggi's monadic paradigm for modelling computational effects using the notion of enriched Lawvere theory, together with its relationship with strong monads; this emphasises the importance of the operations that produce the effects. Effects qua theories are then combined by appropriate bifunctors on the category of theories. We give a theory for the sum of computational effects, which in particular yields Moggi's exceptions monad transformer and an interactive input/output monad transformer. We further give a theory of the commutative combination of effects, their tensor, which yields Moggi's side-effects monad transformer. Finally we give a theory of operation transformers, for redefining operations when adding new effects; we derive explicit forms for the operation transformers associated to the above monad transformers.

1 Introduction

We seek a unified account of modularity for computational effects. More precisely, we seek a mathematical theory that supports the combining of computational effects such as exceptions, side-effects, interactive I/O (i.e., input/output), probabilistic nondeterminism, and nondeterminism. Ideally, we should like to develop natural mathematical operations for the combination of effects, together with associated relevant theory. There is more than one such operation: for example, as we shall see, the combination of side-effects and nondeterminism is of a different nature to the combination of I/O and non-determinism, and, again, one is sometimes interested in different ways to combine even the same pair of effects, for example, side-effects and exceptions. This paper is devoted to two such ways of combining effects: their sum, which, as we shall see, may be employed for combining both exceptions and interactive I/O with other effects; and their commutative combination, their tensor, which, as we shall see, may be employed for combining side-effects with other effects.

⋆ This work has been done with the support of EPSRC grants GR/M56333 and GR/S86372/01

In order to give such operations, we first need a unified way to model the various computational effects individually. In this we start by following Eugenio Moggi, who, in [39,41], gave a unified category-theoretic account of computational effects, which he called notions of computation. He modelled each effect by means of a strong monad T on a base category C with finite products. In the case $C = \mathbf{Set}$, the monads corresponding to the effects listed above are: $T_E = - + E$, $T_S = (S \times -)^S$, $T_{I/O}$, where $T_{I/O}(X) = \mu Y.\,(Y^I + (O \times X) + X)$ [39,40,41], the distributions with countable support monad $\mathcal{D}_\omega$, and the non-empty finite powerset monad $\mathcal{F}^+$. Here E is a countable set of *exceptions*, S is a set of *states*, typically analysed as V^L where V is a countable set of *values* and L is a finite set of *locations*, I is a countable set of *inputs*, and O is a countable set of *outputs*. Corresponding monads exist for a general category C, provided it has appropriate additional structure. Moggi's unified approach has proved useful, particularly in functional programming [7,8].

Strong monads in hand, we seek certain binary operations on them. We seek an operation $\circ$ on strong monads such that:

$$T_E \circ T = TT_E = T(- + E)$$

the computationally natural combination of exceptions with I/O and both forms of nondeterminism, and the usual computationally natural combination with side-effects. We also seek an operation $\otimes$ on strong monads such that:

$$T_S \otimes T = T(S \times -)^S$$

the computationally natural combination of side-effects with I/O and both forms of nondeterminism; it is also a possible, if less natural, combination with exceptions which we discuss further below. Finally, we seek an operation $*$ on strong monads such that:

$$(T_{I/O} * T)X \cong \mu Y.T(Y^I + (O \times Y) + X)$$

the computationally natural combination of interactive I/O with the above effects other than state.

So we ask: can we give a mathematical theory yielding such binary operations on strong monads? Modulo a few side conditions, the answer is positive, and, in order to do so, we make fundamental use of the correspondence between monads and Lawvere theories, which are invariant forms of equational theories [34]. The base category used in denotational semantics is not **Set** but, rather, an order-theoretic one, such as ω-**Cpo**; the objects of this latter category are the ω-cpos, i.e., partial orders with least upper bounds of increasing ω-chains, and the morphisms are the continuous functions, i.e., maps of partial orders that preserve the least upper bounds. So we work with *enriched* Lawvere theories, supported by the correspondence between them and *strong* monads in [49] (and see the expository [53]).

Here, we are following an algebraic programme that shifts focus away from monads to the study of natural programming operations that yield the required effects, with the monads given by free algebras for natural theories for the operations [47] (and see too [44,45]). For instance, rather than emphasise the side-effects monad $(S \times -)^S$, we emphasise the operations *lookup* and *update* associated with side-effects, and the equations relating them. In the case where $S = V^L$, the operation *lookup* can be considered as an L-indexed family of V-ary operations, and *update* can be considered as an $L \times V$-indexed family; the idea is that $\mathit{lookup}_l(x)$ proceeds with x_v if the contents of l is v and $\mathit{update}_{\langle l,v \rangle}(y)$ proceeds with y, having updated l with v. The equations describe the interactions between the *lookup* and *update* operations on the same or different locations. Again, rather than emphasise the powerdomain for nondeterminism, we emphasise the operation of nondeterministic choice $\vee$ with its equations for associativity, symmetry, and idempotence [18,43]. This change in emphasis is computationally natural for all the examples of computational effects listed above. Not all computationally natural operations arise in this way though, only the *algebraic* ones, defined below; an important example of a non-algebraic operation is the *handle* operation, for dealing with raised exceptions. Algebraic operations are in bijective correspondence with *generic effects*, also defined below; sometimes one, sometimes the other, is the more natural in programming languages. In [40] Moggi defined a wider class of operations than ours, including both our algebraic operations and our generic effects, but imposing a weaker naturality condition than we do. However, it is by virtue of restricting to the smaller class of algebraic operations that we are able to find a mathematical theory.

Having reformulated our account of computational effects in terms of enriched Lawvere theories, we can reformulate our questions in terms of corresponding operations on theories. We may ask if there is a mathematical theory yielding an operation $L' \circ L$ on enriched Lawvere theories L' and L such that, for example, in the case of **Set**, if L' is the Lawvere theory L_E associated with exceptions, then $L' \circ L$ corresponds to $T(- + E)$, where L corresponds to T, and similarly for our other two anticipated operations $L' \otimes L$ and $L' * L$. The answers to these reformulated questions are remarkably natural, and, in various guises, have existed since the discovery of the notion of Lawvere theory in the 1960's [13,54]. In the first and third cases, the required operation on Lawvere theories is just their sum, so $L' \circ L = L' * L = L' + L$. Here one has the operations for each of the two theories, subject to the equations for each of the two theories, but with no equations relating them. And in the second case, the required operation $L' \otimes L$ is the *tensor* or *Kronecker* product of theories, which amounts to taking the operations of both theories and demanding that they commute with each other, while retaining the equations of both. For instance, combining side-effects with nondeterminism, and assuming there are three values, one would have the commutation equation:

$$\mathit{lookup}_l(x_1 \vee y_1, x_2 \vee y_2, x_3 \vee y_3) = \mathit{lookup}_l(x_1, x_2, x_3) \vee \mathit{lookup}_l(y_1, y_2, y_3)$$

In a functional language with references and nondeterminism this would induce the program equivalence:

$$\textbf{let } x \textbf{ be } !y \textbf{ in } (M \textbf{ or } N) \equiv (\textbf{let } x \textbf{ be } !y \textbf{ in } M) \textbf{ or } (\textbf{let } x \textbf{ be } !y \textbf{ in } N)$$

where $!M$ is the dereferencing operator and M **or** N is non-deterministic choice; the semantics of $!M$ is given by the generic effect corresponding to *lookup*. There is a similar commutation equation for *update* and $\vee$, with a corresponding induced program equivalence. References for mathematical theory that supports the tensor product are [19,20,21], for which this is a leading example.

The published work most closely related to ours is that of Moggi and Cenciarelli on monad transformers. They defined a monad transformer to be a function $F : |\text{Mon}(C)| \to |\text{Mon}(C)|$ from the set of strong monads on a category C with finite products to itself [40,9,7]. For example, their monad transformer for side-effects takes a monad T to the monad $T(S\times-)^S$, assuming C is cartesian closed, and S is an object of C. Our view of monad transformers is as specialisations of our binary operations for combining effects. Moggi and Cenciarelli's monad transformers agree with ours, as they must; the difference is that we have an associated mathematical theory, including a computationally natural explanation in terms of the equations governing the interaction between the two effects.

The question we pose could equally be posed by asking how one might derive the side-effects monad transformer from the side-effects monad qua monad, but the work on monad transformers to date has not answered that. Moreover, our work involves no asymmetry: there seems no a priori reason why the combination of side-effects with nondeterminism should be achieved by applying a side-effects monad transformer to the nondeterminism monad rather than vice-versa. And, as is well known, in the case of exceptions the side-effects monad transformer does not give the required result for the usual interpretation of the combination. Rather than the standard $(S \times (- + E))^S$ it gives $((S \times -) + E)^S$.

Possible uses of the latter combination are for non-recoverable errors and, more speculatively, for language features for undoing partially-completed transactions, e.g., rollback in database languages. In regard to the latter it is worth noting that the combination supports a non-standard exception-handling mechanism, that, when an exception is raised, restores the state to what it was when the handler was defined, and then executes the handler. This combination of state and exceptions has been termed 'transitional' in [11], and 'snapback' in [4]. In practice, monads for various kinds of exceptions, and even states, would be combined; some sample calculations are given in Section 7.

There is also relevant previous unpublished work by Paul Levy. He observed that the sum of any monad T with that for exceptions $-+E$ is given by $T(-+E)$. He also defined a universal notion of commutative combination of monads and showed that $T(S\times-)^S$ is the commutative combination of T and $(S\times-)^S$ with that definition. When the monads have rank, this definition agrees with ours, but the universal construction seems unlikely to exist for all pairs of monads.

As is well known [5] the composition of monads is also a monad if there is a distributive law between them. This idea has been used to explain several of the monad transformers that arise in computation, such as the exceptions monad transformer [29,23]. However there are several not so explicable, such as the state and resumptions monad transformers. The work on dyads [50,51,52] yields a generalisation of distributive laws that covers the state monad transformer but not the resumptions one.

As discussed in [40,9], for a theory of modularity, having shown how to combine effects one needs to know how to redefine operations, lifting them from the old effects to the combinations with the new ones. Such redefinitions are commonly given on a case-by-case basis, see [7] for a systematic treatment of redefinitions in this way. Here, by restricting to algebraic operations we present a theory prescribing the needed liftings. We define a notion of *operation transformer* which, by standard universal algebra, is equivalent to that of a monad map. Since we have such maps canonically associated with the $+$ and $\otimes$ constructions, we can derive operation transformers for them and so also for their associated monad transformers. We note that in [40] Moggi shows how to redefine members of a certain class of (his) operations; these include our generic effects but not our algebraic operations.

The paper is organised as follows. We generally first investigate the unenriched case, which largely amounts to the situation where computational effects are modelled in **Set**. We then explain the more general enriched situation that includes base categories such as ω-**Cpo**. This allows us to deal with nontermination, i.e., partiality. In Section 2, we describe the relationship between monads and Lawvere theories, and explain how the latter appear in our leading examples. In Section 3, we develop a theory for the sum of Lawvere theories and explain how this gives rise to the exceptions and interactive I/O monad transformers. In fact they are both examples of a more general sum, of a monad with a free monad; the corresponding monad transformer appears in a slightly different form in [9], under the name of the generalised resumptions monad transformer. In Section 4, we explore the tensor product of Lawvere theories and in Section 5, we show that the commutative combination of side-effects with any other Lawvere theory yields the side-effects monad transformer; we also consider some other examples, including the 'complexity' monad transformer from [9] and the analysis of parallel computation in [18]. In Section 6 we consider operation transformers and, so far as we can, give explicit definitions of them for the various monad transformers previously considered. In Section 7, we propose a canonical formula for combining the main computational effects we treat in the paper, and discuss several other issues concerning the combination of effects by sum and tensor. Finally, in Appendix A, we outline the fundamental 2-categorical theory that underlies our main results.

A word on explicit definitions is appropriate here. Where possible we give explicit definitions of monad and operation transformers. We do not claim that these formulae are particularly new, though they may sometimes be more general than have appeared previously. The point is rather that they arise canonically from our general understanding of monad transformers as specialisations of natural binary constructions on Lawvere theories, namely sum and tensor, and from operation transformers via their relation to monad maps. On a slightly different note, we use standard mathematical notation for these explicit definitions; this can be contrasted with Moggi who prefers to use a suitable type theory to aid comprehensibility; again functional programmers prefer to give corresponding definitions in a functional programming language, typically Haskell.

A clear omission from this paper is the study of distributivity: this seems to be how various forms of nondeterminism combine. For example, the distributivity of probabilistic choice over ordinary choice when combining probabilistic nondeterminism with nondeterminism is discussed in [57,37,58], and the distributivity of each of internal and external nondeterminism over each other is discussed in [17]. Another important question concerns the combination of effects with local state [47]: this paper only concerns global state. In [47] local state is specified using an additional operation *block* together with additional equations, and is modelled using a presheaf category, thereby avoiding any need for models with infinitely many locations. But it is unclear yet how best to integrate this work with enriched Lawvere theories, let alone consider combinations with other effects. Finally, we have not considered the relationship of all these effects with that of continuations; this will be substantially different as, unlike all the other cases, the continuations monad does not have a rank. One can still treat the continuations monad algebraically but that means introducing operations of unbounded rank, which does not make computational sense to us. We therefore believe continuations should be treated separately. In this connection it is worth noting that, were it not for the desire to include continuations in the treatment of monad transformers, Moggi and Cenciarelli might have taken them to be functors $F : \mathrm{Mon}(C) \to \mathrm{Mon}(C)$ equipped with with a natural transformation $I \to F$, rather than mere functions; the natural transformation would be used to extend (algebraic) operations, as explained in Section 6 below.

2 Monads and Lawvere theories

For simplicity of exposition, we start by restricting our attention to the base category **Set**. All our monads on **Set** are of *countable rank*, which means that, in a precise sense, they are of bounded size [26,3]. The category of monads with countable rank is equivalent to the category of countable Lawvere theories, which we now define. All our mathematics generalises to arbitrary rank, but, as all our examples are of countable rank, which includes finite rank, we restrict our exposition to that case.

Let $\aleph_1$ denote a skeleton of the category of countable sets and all functions between them. So $\aleph_1$ has an object for each natural number n and an object for $\aleph_0$. Up to equivalence, $\aleph_1$ is the free category with countable coproducts on 1. So, in referring to $\aleph_1$, we implicitly make a choice of the structure of its countable coproducts.

Definition 1. *A* countable Lawvere theory *consists of a small category L with countable products and a strict countable-product preserving identity-on-objects functor $I : \aleph_1^{\mathrm{op}} \longrightarrow L$. A map of countable Lawvere theories from L to L' is a strict countable-product preserving functor from L to L' that commutes with I and I'.*

We sometimes refer to morphisms of a Lawvere theory as *operations*.

Definition 2. *A* model *of a countable Lawvere theory L in any category C with countable products is a countable-product preserving functor $M : L \longrightarrow C$.*

Note that, following Lawvere [33], and see [5], we ask here for ordinary preservation of countable products. Every such model is equivalent to one in which powers of of 1 are strictly preserved. However, the conventional non-strict version is more convenient in practice: for example, for any model $M : L \longrightarrow C$ of L and countable product-preserving functor $U : C \longrightarrow D$, the composite UM is then also a model of L; it also fits well with a 2-categorical treatment [48].

For any countable Lawvere theory L and any category with countable products C, we thus have the category $\mathrm{Mod}(L, C)$ of models of L in C; the maps are given by all natural transformations: the naturality condition implies that they respect countable product structure. There is a canonical forgetful functor $U_L : \mathrm{Mod}(L, C) \longrightarrow C$. If it has a left adjoint F_L, this forgetful functor exhibits $\mathrm{Mod}(L, C)$ as coherently equivalent to the category T_L-Alg for the monad $T_L = U_L F_L$ thereby induced by L on C. If C is locally countably presentable [3], then: the required left adjoint F_L exists; U_L, and so also T_L, have countable rank; and $\mathrm{Mod}(L, C)$ is locally countably presentable.

For a converse in the case that $C = \mathbf{Set}$, given a monad T with countable rank on $\mathbf{Set}$, the category $\mathrm{Kl}(T)^{\mathrm{op}}_{\aleph_1}$ determined by restricting $\mathrm{Kl}(T)$, the Kleisli category of T to the objects of $\aleph_1$ is a countable Lawvere theory L_T, and the functor from T-Alg to $\mathrm{Mod}(L_T, \mathbf{Set})$ induced by the restriction is an equivalence of categories. An enriched, thereby more general, version of the following result appears in [49].

Theorem 1. *The construction sending a countable Lawvere theory L to T_L together with that sending a monad T with countable rank to L_T induce an equivalence of categories between the category of countable Lawvere theories and the category of monads with countable rank on* **Set**. *Moreover, the comparison functor exhibits an equivalence between the categories* $\mathrm{Mod}(L, \mathbf{Set})$ *and* T_L-Alg.

The usual way in which to obtain Lawvere theories is by means of sketches, with the Lawvere theory given freely on the sketch: Barr and Wells' book [6] treats sketches in loving detail. To give a sketch amounts to giving operations and equations; for countable Lawvere theories one allows the operations to be of countable arity.

We now consider our main examples from the point of view of countable Lawvere theories. Given a category C with a terminal object and a set X, we write $\underline{X}$ for the X-fold copower of 1, i.e., $\coprod_X 1$.

Example 1. **Exceptions** The countable Lawvere theory L_E for exceptions is the free countable Lawvere theory generated by an operation $\mathit{raise} : 0 \longrightarrow E$, where E is a countable set of *exceptions*. In terms of operations and equations this corresponds to an E-indexed family of nullary operations with no equations. In terms of models M of L_E one has:

$$M(\mathit{raise}) : 1 = M(1)^0 \longrightarrow M(1)^E$$

which corresponds to an evident map $E \longrightarrow M(1)$, again showing how codomains of operations correspond to parameterisation.

Note our use here of the countable set E for the codomain of the operation of the Lawvere theory; strictly speaking we should instead have used the corresponding object of $\aleph_1$. It is, however, conceptually convenient to allow ourselves such minor liberties.

The monad induced by L_E is $T_E = - + E$, the exceptions monad mentioned above. More generally, if C is any category with countable sums and a terminal object then the monad induced by L_E on C is $- + \underline{E}$.

Given a category C and a set X, we write $(X \times -)$ for the X-fold copower $\coprod_X -$, and $(-)^X$ for the X-fold power $\prod_X -$.

Example 2. **Side-Effects** The countable Lawvere theory L_S for side-effects, where $S = V^L$, with V countable and L finite, is the free countable Lawvere theory generated by the operations $\mathit{lookup} : V \longrightarrow L$ and $\mathit{update} : 1 \longrightarrow L \times V$ subject to the seven natural equations listed in [47], four of them specifying interaction equations for *lookup* and *update* and three of them specifying commutation equations. Note, as in the case of exceptions, the use of codomains, here L and $L \times V$, to handle indexing at the Lawvere theory level. It is shown in [47] that this Lawvere theory induces the side-effects monad mentioned above. More generally, if C is any category with countable powers and copowers then, slightly generalising the result in [47], the monad induced by L_S on C again has the form $(S \times -)^S$.

For the next example, we first need some discussion of free and initial algebras and free monads. Given any endofunctor Σ on a category C, we write $(\mu y.\Sigma y, \alpha_\Sigma)$ for the initial Σ-algebra if it exists. If C has binary sums, the free Σ-algebra on an object x can be identified with $(\mu y.\,(\Sigma y + x), \alpha_{(\Sigma - + x)})$, and the one exists if and only the other does. These free algebras exist if, for example, C is locally countably presentable and Σ has countable rank.

Next, if the forgetful functor from Σ-alg to C has a left adjoint, we say that the resulting monad is the *free* monad on Σ and write it as Σ^*. If Σ^* exists, the category Σ^*-Alg for Σ^* qua monad is isomorphic to the category Σ-alg for the endofunctor Σ.

We see from the above that, if C has binary sums, then Σ^* can be identified with $\mu y.\,(\Sigma y + -)$ and the one exists if and only if the other does. We further see that if C is locally countably presentable and Σ has countable rank, then Σ^* exists and has countable rank [25].

Example 3. **Interactive I/O** The countable Lawvere theory $L_{I/O}$ for interactive I/O is the free countable Lawvere theory generated by the operations $\mathit{read} : I \longrightarrow 1$ and $\mathit{write} : 1 \longrightarrow O$, where I is a countable set of *inputs* and O of *outputs*. In terms of operations and equations this corresponds to an operation of arity I together with an O-indexed family of unary operations, with no equations. The monad $T_{I/O}$ for interactive I/O corresponding to this Lawvere theory is the free monad on $\Sigma_{I/O}$ where $\Sigma_{I/O}(Y) = Y^I + (O \times Y)$ is the *signature* functor determined by the two operations. By the above remarks, we have:

$$T_{I/O}(X) = \mu Y.\,(Y^I + (O \times Y) + X)$$

These are also the forms of $\Sigma_{I/O}$ and $T_{I/O}$ in the more general situation where the monad is that induced by $L_{I/O}$ on a locally countably presentable category C.

Exceptions and interactive I/O exemplify a general pattern of 'absolutely free' theories. Consider the free countable Lawvere theory L on the operations $op_\lambda : I_\lambda \to O_\lambda$, for $\lambda \in \Lambda$, where the I_λ and the O_λ are countable sets. Define the corresponding signature functor to be the 'polynomial' functor:

$$\Sigma(Y) = \sum_{\lambda \in \Lambda} O_\lambda \times Y^{I_\lambda}$$

Then the monad corresponding to L is Σ^*, the free monad on Σ, and, as we see from the above, it can be given explicitly by:

$$T(X) = \mu Y.\,(\sum_{\lambda \in \Lambda} O_\lambda \times Y^{I_\lambda} + X)$$

The signature functor for exceptions is, evidently, $\Sigma_E(Y) = E$ and we have already given that for interactive I/O. As before, these are also the forms of Σ and T in the more general situation where the monad is that induced by L on a locally countably presentable category C. We remark that polynomial functors of the above kind appear in the context of categorical models of dependent type theory, where the existence of Σ^* corresponds to the existence of W-types [38,14].

Example 4. **Nondeterminism** The countable Lawvere theory L_N for (binary) nondeterminism is the countable Lawvere theory freely generated by a binary operation $\vee : 2 \longrightarrow 1$ subject to equations for associativity, commutativity and idempotence, i.e., the countable Lawvere theory for a semilattice; the corresponding monad on **Set** is the finite non-empty subset monad, $\mathcal{F}^+$.

Example 5. **Probabilistic Nondeterminism** The countable Lawvere theory L_{P_f} for finite probabilistic nondeterminism is that freely generated by $[0,1]$-many binary operations $+_p : 2 \longrightarrow 1$, for $p \in [0,1]$, subject to the equations for associativity, commutativity and idempotence given in [16]. The induced monad on **Set** is the distributions with finite support monad, $\mathcal{D}_f$.

There is also a countable Lawvere theory L_{P_ω} for countable probabilistic nondeterminism. This is the theory of *superconvex spaces* [32]; it has operations $\sum_p$ of countably infinite arity, indexed by sequences $p \in [0,1]^\omega$ whose sum is 1, and subject to the two elegant equations:

1. $\sum_{n\geq 0} \delta^m_n x_n = x_m$
2. $\sum_{n\geq 0} p_n \sum_{m\geq 0} q_{nm} x_m = \sum_{m\geq 0} (\sum_{n\geq 0} p_n q_{nm}) x_m$

using an evident notation, and where δ_n^m is the Kronecker delta function. The induced monad on **Set** is the distributions with countable support monad, $\mathcal{D}_\omega$.

Superconvex spaces have, admittedly, a rather profligate collection of operations. However one can economise: they can all be defined in terms of one such operation, for example that where $p_n = 2^{-(n+1)}$; it seems not to be known whether there is an elegant equational axiomatisation in terms of this operation alone: an elegant non-equational axiomatisation has been given in [10]. There is an evident variation of *convex spaces* which uses instead finite sequences of reals in $[0, 1]$, subject to the analogous axioms; this yields an alternative and elegant presentation of the theory for finite probabilistic nondeterminism.

Of course **Set** is not the category of primary interest in denotational semantics. One is more interested in ω-**Cpo**, and variants, in order to model recursion. As we now outline, the relationship between countable Lawvere theories and monads with countable rank generalises without fuss to one between countable enriched Lawvere theories and strong monads with countable rank on such categories. We enrich with respect to a category V that is locally countably presentable as a cartesian closed category: ω-**Cpo** is one such. It is worth observing that the category of directed complete partial orders (dcpos), a standard alternative to ω-**Cpo** in the literature, is not locally presentable.

The least obvious point to note when enriching Lawvere theories is that the notion of countable product of a single generator does not generalise most naturally to a notion of countable product but rather to a notion of countable *cotensor* [26]. The notion of cotensor is the most natural enrichment of the notion of a power-object. Given an object x of a category C and given a set A, the A-fold power x^A satisfies the defining condition that there is a bijection of sets:

$$C(y, x^A) \cong C(y, x)^A$$

natural in y. Enriching this, given an object x of a V-category C and given an object a of V, the cotensor x^a satisfies the defining condition that there is an isomorphism in V:

$$C(y, x^a) \cong C(y, x)^a$$

V-natural in y. When $C = V$, x^a is the exponential. We say that the cotensor x^a is *countable* if a is countably presentable. When V is **Set**, the countably presentable objects are exactly the countable sets. These objects are harder to characterise in the case of ω-**Cpo**, but it can be shown that they include all the ω-continuous ω-cpos.

These cotensors are used to enable enriched Lawvere theories to have as arities objects of V other than discrete ones, meaning those of the form $\underline{X}$. As an example, taking V to be **Poset**, this allows us not only to consider objects such as $x^{\underline{2}}$ $(= x^2 = x \times x)$ in a locally ordered category, but also to consider objects such as $x^{\leq}$, where $\leq$ is Sierpinski space, the two-point partial order $\bot \leq \top$. This possibility allows us, in describing **Poset**-theories, to incorporate inequations. For, suppose one wishes to say that $f \leq g$ for two morphisms $f, g : x \to y$; this is accomplished by introducing a third morphism $h : x \to y^{\leq}$ and asserting the equations $f = y^{\perp} \circ h$ and $g = y^{\top} \circ h$, where $\perp$ and $\top$ are the two evident maps from 1 to $\leq$.

There is an evident dual notion of *tensor* $a \otimes x$ generalising the notion of copower. It satisfies the defining condition that there is an isomorphism in V:

$$C(a \otimes x, y) \cong C(x, y)^a$$

V-natural in y; this will prove useful below. When $V = \mathbf{Set}$, $A \otimes x$ is the A-fold copower of x, i.e., $A \times x$. When $C = V$, $a \otimes x$ is the product of a and x.

If C is a locally countably presentable as a V-category [27] (e.g., when $C = V$) then it has both tensors and cotensors, the V-functor $a \otimes -$ has countable rank, since it is a colimit, and so does $(-)^a$ if a is countably presentable.

We can now proceed to the definition of countable Lawvere V-theories. Define $V_{\aleph_1}$ to be a skeleton of the full sub-V-category of V determined by the countably presentable objects of V. It is equivalent to the free V-category with countable tensors on 1 [26,49], so as before we assume a choice of this structure.

Definition 3. *A* countable Lawvere V-theory *is a small V-category L with countable cotensors together with a strict countable-cotensor preserving identity-on-objects V-functor $I : V_{\aleph_1}^{\mathrm{op}} \longrightarrow L$. A map of countable Lawvere V-theories from L to L' is a strict countable-cotensor preserving V-functor from L to L' that commutes with I and I'. A* model *of L in a V-category C with countable cotensors is a countable-cotensor preserving V-functor $M : L \longrightarrow C$.*

Routinely generalising the unenriched case, for any countable Lawvere V-theory L and any V-category with countable cotensors C, we have a V-category of models of L in C, $\mathrm{Mod}(L, C)$; the homobjects are given by homobjects of all V-natural transformations [26], and the V-naturality condition implies they respect countable cotensors. There is a canonical forgetful V-functor U_L from $\mathrm{Mod}(L, C)$ to C. If it has a left V-adjoint F_L, this forgetful V-functor exhibits $\mathrm{Mod}(L, C)$ as coherently equivalent to the V-category T_L-Alg for the V-monad $T_L = U_L F_L$ thereby induced by L on C. If C is locally countably presentable as a V-category, then: the required left V-adjoint F_L exists; U_L, and so also T_L, have countable rank; and $\mathrm{Mod}(L, C)$ is locally countably presentable as a V-category.

For a converse in the case that $C = V$, given a V-monad T with countable rank on V, the V-category $\mathrm{Kl}(T)_{\aleph_1}^{\mathrm{op}}$ determined by restricting $\mathrm{Kl}(T)$ to the objects of $V_{\aleph_1}$ is a countable Lawvere V-theory L_T, and the V-functor from T-Alg to $\mathrm{Mod}(L_T, \mathbf{Set})$ induced by the restriction is an equivalence of V-categories.

To give a V-enriched V-monad is equivalent to giving a strong monad on V [31]. So, in order to make the comparison with Moggi's definition a little more direct, we express the main abstract result of [49] in terms of strong monads.

Theorem 2. *If V is locally countably presentable as a cartesian closed category, the constructions of T_L from L and of L_T from T induce an equivalence of categories between the category of countable Lawvere V-theories on V and the category of strong monads on V with countable rank. Moreover, the comparison V-functor exhibits an equivalence between the V-categories* $\mathrm{Mod}(L, V)$ *and* T_L-Alg.

A common and important way to generate countable Lawvere V-theories is by taking the free countable Lawvere V-theory on an unenriched countable Lawvere theory. Given an unenriched countable Lawvere theory L, the free countable Lawvere V-theory on L is generated by the operations and equations of L. Note that it will typically have more objects as there may be countably presentable objects other than the discrete ones; these additional objects may, in turn, generate additional maps.

Write L_V for the free V-theory on L. Then the category $\mathrm{Mod}(L, V)$ is isomorphic to the underlying ordinary category of the V-category $\mathrm{Mod}(L_V, V)$. And the ordinary monad T_0 generated by the forgetful functor from $\mathrm{Mod}(L, V)$ to V is the underlying ordinary monad of the V-monad generated by the forgetful V-functor from $\mathrm{Mod}(L_V, V)$ to V.

So the passage from L to L_V is simple, and we typically overload notation a little by dropping the subscript on L_V, thus using the notation L for both an ordinary Lawvere theory and the free V-theory on it. It can additionally happen, as in the case $V = \omega$-**Cpo**, that a functor has at most one enrichment to a V-functor, making V-enrichment a property rather than extra structure. In that case one can also gloss over the difference between the enriched and the ordinary monads.

Before passing to examples with $V = \omega$-**Cpo** we first present some remarks on initial and free objects and monads in an enriched context. Given any V-endofunctor Σ on a V-category C, let $(\mu y.\Sigma y, \alpha_\Sigma)$ denote the initial Σ-algebra if it exists. If C has V-cotensors, the initial Σ-algebra is the same as the initial algebra of Σ_0, the underlying ordinary functor of Σ, with one existing if and only if the other does; more generally, the underlying category Σ-alg_0 of the V-category Σ-alg is isomorphic to Σ_0-alg and the forgetful V-functor from Σ-alg to C has a left V-adjoint if and only if the forgetful functor from Σ_0-alg to C_0 has an ordinary left adjoint, and these adjoints necessarily agree with each other.

Next, if C has binary sums, the free Σ-algebra on an object x can be given in terms of initial algebras as $(\mu y.\, (\Sigma y + x), \alpha_{(\Sigma - + x)})$, with one existing if and only the other does. These free algebras exist if, for example, C is locally countably presentable as a V-category, e.g., when $C = V$ and Σ has countable rank.

If the forgetful functor from Σ-alg to C has a left V-adjoint, we say that the resulting V-monad is the free V-monad on Σ and write it as Σ^*. If Σ^* exists, then its underlying monad is the free monad on Σ_0, and the V-category Σ^*-Alg for Σ^* qua V-monad is isomorphic to the V-category Σ-alg for Σ.

We see from the above that when C has binary sums, Σ^* is $\mu y.\, (\Sigma y + -)$ with one existing if and only if the other does. We also then see that the monad Σ^* exists when C is locally countably presentable as a V-category, e.g., when $C = V$ and Σ has countable rank.

We conclude this section by looking at some computationally relevant examples of enriched theories in the case $V = \omega$-**Cpo**. The first example of a countable Lawvere ω-**Cpo**-theory does not arise freely from an unenriched countable Lawvere theory.

Example 6. **Nontermination** The countable Lawvere ω-**Cpo**-theory L_Ω for nontermination is the theory freely generated by a nullary operation $\Omega : 0 \longrightarrow 1$ subject to the condition that there is an inequality:

$$\begin{tikzcd}
1 \arrow[r] \arrow[dr, "\mathrm{id}"'] & 0 \arrow[d, "\Omega"] \arrow[dl, phantom, "\geq"] \\
 & 1
\end{tikzcd}$$

where the unlabelled map is the unique map determined because 0 is the initial object of $V_{\aleph_1}$ and therefore the terminal object of $V_{\aleph_1}^{\mathrm{op}}$. The models of L_Ω in ω-**Cpo** are the ω-cpos with a least element. The corresponding strong monad T_Ω is the lifting construction $(-)_\perp$ which adds a new least element. It is worth noting that there is at most one morphism from L_Ω to any other countable ω-**Cpo**-theory L; this reflects the fact that a least element is unique, if it exists.

Adding a nontermination effect allows us to model recursion in the context of ω-**Cpo**; if we then also want to model other effects we have to combine them with nontermination: simply adapting to ω-**Cpo** by switching to $L_{\omega\text{-}\mathbf{Cpo}}$ from L does not suffice. (See [44] for sufficient conditions for the use of ω-**Cpo**-enriched categories and monads to model a call-by-value language with effects and recursion.) We will therefore study the combination of our example effects with nontermination. This is accomplished by combining the relevant $L_{\omega\text{-}\mathbf{Cpo}}$ with L_Ω, either by sum or by tensor, but one must be careful in the case of probabilistic nondeterminism: see the discussion below.

There is, perhaps, something rather ad hoc about the present treatment of recursion: one simply plucks ω-**Cpo** 'out of the air' and then adds a nontermination effect. One wonders if recursion itself can in some sense be thought of as an effect; if not, a more satisfactory treatment of the combination of effects with recursion could perhaps be obtained in the context of axiomatic or synthetic domain theory [12].

We now look again at the examples considered above in the case of **Set**. For exceptions, interactive I/O and side-effects this is matter of generalising from countable sets to countably presented ω-cpos.

Example 7. **Absolutely Free Theories** Consider the free countable Lawvere ω-**Cpo**-theory L on operations $op_\lambda : I_\lambda \to O_\lambda$ for $\lambda \in \Lambda$ where the I_λ and the O_λ are countably presentable ω-cpos. The signature functor $\Sigma : \omega\text{-}\mathbf{Cpo} \to \omega\text{-}\mathbf{Cpo}$ is defined similarly to before, by:

$$\Sigma(Q) = \sum_{\lambda \in \Lambda} O_\lambda \times Q^{I_\lambda}$$

where we mean the usual products and powers in ω-**Cpo**. The monad induced by L is again Σ^*, the free ω-**Cpo**-monad on Σ; it can be given explicitly, again similarly to before, by:

$$T(P) = \mu Q.\,(\sum_{\lambda \in \Lambda} O_\lambda \times Q^{I_\lambda} + P)$$

As before too, all this generalises to a locally countably presentable ω-**Cpo**-category C, where Σ and T have the same form as above, but replacing products and powers by tensors and cotensors, and where T is now the monad on C induced by L. The exceptions and interactive I/O signature functors, Σ_E and $\Sigma_{I/O}$, and monads, T_E and $T_{I/O}$, are again special cases.

Example 8. **Side-Effects** We can generalise the set V of values, but not the finite set L of locations, to be any countably presentable ω-cpo. Then we take the countable Lawvere ω-**Cpo**-theory L_S to be freely given by the diagrams in [47], but allowing this more general kind of state. Further generalising the result given there, we obtain that, if C is any ω-**Cpo**-category with tensors and cotensors of countably presentable ω-cpos, then the ω-**Cpo**-monad induced on C by L_S is $(S \otimes -)^S$.

In the cases where the countably presentable ω-cpos are discrete, the Lawvere ω-**Cpo**-theories are the free countable Lawvere ω-**Cpo**-theories on the corresponding countable Lawvere theories.

Example 9. **Nondeterminism** The countable Lawvere ω-**Cpo**-theory L_N for binary nondeterminism is again that of a semilattice, and so it is the ω-**Cpo**-theory freely generated by the corresponding countable Lawvere theory. The induced strong monad T_N is the (convex) powerdomain monad [42,18], taking that on ω-**Cpo** to be synonymous with the free ω-**Cpo**-semilattice monad.

There are two more powerdomains: the upper or Smyth one and the lower or Hoare one. As essentially mentioned in [18] these can also be described by ω-**Cpo**-theories. The upper one is given by adding the inequation:

$$x \vee y \leq y$$

and the lower one by the opposite inequation:

$$x \vee y \geq y$$

One can give explicit descriptions of these powerdomains, if necessary making further restrictions on the class of partial orders under consideration; see [15] for a recent treatment in the context of the category of dcpos.

Example 10. **Probabilistic Nondeterminism** A computationally natural presentation of a countable Lawvere ω-**Cpo**-theory for probabilistic nondeterminism combined with nontermination [16,22,24,47] can be obtained by taking the axioms for L_{P_f} and L_Ω together with an infinitary axiom relating the least element with probabilistic choice. This axiom says that any element x is equal to the limit of the increasing sequence:

$$\Omega, x +_{1/2} \Omega, x +_{1/2} (x +_{1/2} \Omega), \ldots$$

It can be stated within the framework of Lawvere ω-**Cpo**-theories by making use of the countably presentable ω-cpo:

$$0 \leq 1 \leq \ldots \leq n \leq \ldots \leq \omega$$

which can be thought of as the 'standard ω-chain.'

This interaction between probabilistic choice and nontermination is unpleasant. Fortunately, however, there is a natural and elegant alternative presentation of the theory, consisting of the axioms of L_{P_ω} together with that of L_Ω. There is then no need for any additional axiom on the interaction of probabilistic choice and nontermination.

We should comment that the standard theory of probabilistic powerdomains was not developed in the category of ω-cpos, but rather in that of dcpos. The initial definition was in terms of valuations and was shown to be equivalent to an algebraic one on the subclass of the continuous dcpos. Given that background, it seems reasonable to transplant the algebraic treatment of probabilistic nondeterminism to ω-**Cpo** to the present context.

Finally we remark that although our examples mainly concern **Set** and ω-**Cpo**, everything can be done more generally. For everything other than partiality one can replace **Set** by any category V that is locally countably presentable as a cartesian closed category, interpreting the theories of the various effects in our examples as countable Lawvere V-theories. To include partiality and to have the least upper bounds needed for recursion one can take V to be locally countably presentable as a cartesian closed ω-**Cpo**-category [27].

3 The sum of effects

Our leading examples of the sum of effects are given by the combination of exceptions with all the other computational effects we consider: side-effects, interactive I/O and nondeterminism, and by the combination of interactive I/O with all other effects we consider except for side-effects. A succession of results support the construction of the sum of theories.

Theorem 3. *The category of countable Lawvere theories is cocomplete.*

This may be proved using the equivalence between countable Lawvere theories and monads on **Set** of countable rank, together with the analysis of [28]. The construction of the sum is complicated, especially when attempted in terms of monads: a general construction involves a transfinite induction, with inductive steps being given by a complicated coequaliser [25]. But all our examples of Lawvere theories are given freely on equational theories. And in those terms, the sum is easy to describe: one takes the operation symbols of both equational theories, renamed, if necessary, to avoid confusion, and takes the axioms of both. The complication arises in passing from the induced equational theory to the Lawvere theory freely generated by it, as, in doing so, one may apply the operations of one theory to the operations of the other, hence the generally transfinite induction.

Even in terms of equational theories, care is required. For instance, given Lawvere theories L and L', there are always maps of Lawvere theories given by coprojections $L \longrightarrow L + L'$ and $L' \longrightarrow L + L'$. But these coprojection functors need not be faithful. For instance, L might be the trivially collapsing theory, i.e., its equations may force L to be equivalent to 1. In that case, $L + L'$ is also equivalent to 1, so the coprojection from L' is trivial.

From Theorem 3 we know the sum exists, and, when starting with equational theories, we know how to describe it. But for the purposes of calculation, it is still convenient to have a more explicit construction of the sum qua monad. And, under a condition that includes the examples of exceptions and interactive I/O, we can provide that. The key point is that, in both of these cases, the monads are generated by operations subject to no equations. So they may be described as the free monads on endofunctors Σ with countable rank, namely the signature functors Σ_E and $\Sigma_{I/O}$. We will give characterisations of the sum of a general monad and a free one. It turns out that we can work much more generally than on **Set**. For simplicity of exposition, we shall start with a category C and assert conditions on it as convenient.

It is worth remarking that using these ideas one already has an explicit characterisation of sum in the case where both monads are free. It is straightforward to show, e.g., by considering the categories of algebras, that the sum of ${\Sigma_1}^*$ and ${\Sigma_2}^*$ is $(\Sigma_1 + \Sigma_2)^*$, where we now mean the pointwise sum of functors; this is subject to the proviso that C has binary sums and that $(\Sigma_1 + \Sigma_2)^*$ exists, which it does when C is locally countably presentable and the Σ_i have countable rank. As an example, the sum of the exceptions and I/O monads is $\mu Y.\, (E + Y^I + O \times Y + -)$.

We now turn to our more general case, the explicit characterisation of the sum of a monad and a free monad. Given an endofunctor $H : C \to C$ and a monad $T : C \to C$, a *distributive law* of H over T is a natural transformation $\lambda : HT \to TH$ subject to commutativity of the evident two diagrams expressing coherence of λ with respect to the monad structure of T; this is a slight variation on the usual notion of a distributive law of a monad over a monad as in [5].

Proposition 1. *For any endofunctor Σ and monad (T, μ, η) over a category C, the natural transformation:*

$$\lambda \quad = \qquad \Sigma TT \xrightarrow{\Sigma\mu} \Sigma T \xrightarrow{\eta\Sigma T} T\Sigma T$$

is a distributive law of ΣT over T.

Corollary 1. *The monad T lifts to a monad T' on ΣT-*alg *given by:*

$$T'((x, \alpha)) = (Tx, \Sigma TTx \xrightarrow{\lambda_x} T\Sigma Tx \xrightarrow{T\alpha} Tx)$$

We denote by $(\Sigma + T)$-Alg the category for which an object consists of an object x of C together with two structures on it: a T-structure $\beta : Tx \to x$, for T as a monad, and a Σ-structure $\gamma : \Sigma x \to x$, for Σ as a functor, and with maps being those maps in C that preserve the two structures. The reason for this notation is that $(\Sigma + T)$-Alg is the category of algebras for the monad $\Sigma^* + T$ if that sum exists. There is a canonical functor $\natural : T'\text{-Alg} \longrightarrow (\Sigma + T)\text{-Alg}$ that sends $((x, \alpha : \Sigma T x \to x), \beta : Tx \to x)$ to $(x, \alpha \cdot \Sigma\eta_x : \Sigma x \to x, \beta : Tx \to x)$.

Lemma 1. *The functor* $\natural : T'\text{-Alg} \longrightarrow (\Sigma + T)\text{-Alg}$ *is an isomorphism of categories.*

Proof. The inverse of $\natural$ sends the object $(x, \beta : Tx \to x, \gamma : \Sigma x \to x)$ to the object $((x, \gamma \cdot \Sigma\beta : \Sigma T x \to x), \beta)$. Funtoriality and the proof that this is an inverse are mundane.

Theorem 4. *Given an endofunctor* $\Sigma : C \longrightarrow C$ *and a monad* $T : C \longrightarrow C$*, if the free monads* Σ^* *and* $(\Sigma T)^*$ *exist then the sum of monads* $\Sigma^* + T$ *exists in the category of monads over* C *and is given by a canonical monad structure on the composite* $T(\Sigma T)^*$*.*

Proof. If $(\Sigma T)^*$ exists, the category ΣT-alg for the endofunctor ΣT is isomorphic to the category $(\Sigma T)^*$-Alg for $(\Sigma T)^*$ qua monad. So T' is a lifting of T to the category of algebras for the monad $(\Sigma T)^*$. So we have a distributive law of $(\Sigma T)^*$ over T, yielding a monad structure on $T(\Sigma T)^*$, with $T(\Sigma T)^*$-Alg isomorphic to T'-Alg [5]. If Σ^* also exists, $(\Sigma + T)$-Alg is, by construction, isomorphic to $(\Sigma^* + T)$-Alg, yielding the result.

Note that this result is general and does not refer to local countable presentability. However if C is locally countably presentable and both Σ and T have countable rank then $T(\Sigma T)^*$ has countable rank and so it is also the sum in the category of monads over C of countable rank.

We remark that one can work much more generally still: the characterisation of the sum in Theorem 4 can be made within an arbitrary 2-category with a few limits. In that sense, Theorem 4 is inherently a category-theoretic result, not relying on any substantial fact about **Cat**. The proof at the 2-categorical level is a straightforward application of Street's formal theory of monads [56].

In [9], Cenciarelli and Moggi introduce a generalised resumptions monad transformer, sending T to $\mu z.T(\Sigma z + x)$. Their resumptions monad is the special case where Σ is the identity. This amounts to adding the theory L_d of a unary operator $d : 1 \to 1$ with no equations to L_T; one can think of the operation d as 'suspending' or 'delaying' computation. It is straightforward to show, using Proposition 5.3 of [55], that $\mu z.T(\Sigma z + x)$ exists if and only if $(\Sigma T)^* x$ does, and that $T(\Sigma T)^* x$ and $\mu z.T(\Sigma z + x)$ are then isomorphic. So our $T(\Sigma T)^*$ is simply another form of the generalised resumptions monad transformer; the point here, as elsewhere, is that we have derived it as part of a general theory of combinations of monads. We can summarise this discussion with the following corollary of Theorem 4:

Corollary 2. *Given an endofunctor* $\Sigma : C \longrightarrow C$ *and a monad* $T : C \longrightarrow C$*, if* Σ^* *and* $\mu z.T(\Sigma z + x)$ *exist, then the sum of monads* $\Sigma^* + T$ *exists in the category of monads on* C *and is given by a canonical monad structure on* $\mu z.T(\Sigma z + -)$*.*

In giving this result, we have used a distributive law of ΣT over T. But if we restrict our attention to a monad T and a free monad on an endofunctor Σ where, for definiteness, T and Σ have countable rank on a locally countably presentable category, we can provide a more direct proof of this result, not referring to a distributive law. The category of monads with countable rank on a locally countably presentable category C is monadic over the category of endofunctors with countable rank on C (see [28]). The construction $\mu z.T(\Sigma z + x)$ extends to an endofunctor on the category of monads with rank that lifts an endofunctor on the category of endofunctors with rank. That fact, together with a Bekič-style result and use of Beck's monadicity theorem [5], yields a direct proof that the construction $\mu z.T(\Sigma z + x)$ agrees with $\Sigma^* + T$.

Theorem 4 yields a characterisation of the monads generated by the sum of the exceptions Lawvere theory with any other Lawvere theory, and also by the sum of the interactive I/O Lawvere theory with any other Lawvere theory. In the case of exceptions, one can give a simpler proof, essentially by observing that the endofunctor is the constant at E and by routinely using that fact to simplify the above argument. In direct terms, the argument is as follows:

Corollary 3. *Given a category C with binary sums, an object E of C, and a monad T on C, the sum of the monads $(- + E)$ and T exists and is given by the monad $T(- + E)$.*

Proof. (Direct proof) The category $T(-+E)$-Alg is isomorphic to T'-Alg, where T' is the monad on $(- + E)$-Alg determined by lifting T, using the canonical distributive law of $- + E$ over T. By direct calculation, the latter category is in turn isomorphic to $((-+E)+T)$-Alg: a T'-algebra consists of an object x together with a morphism $E \to x$ and a T-structure on x, that is, a $((-+E)+T)$-algebra.

This result explains how the exceptions monad transformer, sending a monad T_L to the composite $T_L(- + E)$, arises: one takes the disjoint union of the two sets of operations and retains the equations for T_L. And this explanation brings with it the theory of coproducts, such as their associativity and commutativity, and their interaction with other operations.

For interactive I/O, the above argument seems as simple as is likely to be found. It duly induces an interactive I/O monad transformer that sends a monad T on a locally countably presentable category C to the monad $T + T_{I/O}$ given as $T(\mu y.\,((Ty)^I + (O \times Ty) + -))$; the other form of this monad discussed above is $\mu z.T(z^I + (O \times z) + -)$.

While we have shown that there is a canonical monad structure on $T(\Sigma T)^*$, we have not given it explicitly. However explicit formulae can be extracted from the proof, and we now give them. First we need some notation. Suppose we are given an endofunctor $F : C \longrightarrow C$, and that the forgetful functor from F-alg to C has a left adjoint, so that the free monad F^* on F exists. Then we write η_F for the unit of the adjunction, $\alpha_F x$ for the free algebra map, $\alpha_F x : FF^*x \to F^*x$, and, for any F-algebra $a = (y, \beta)$ and $f : x \to y$, we write $\mathrm{I}_{F,x,a}(f) : (F^*x, \alpha_F x) \to a$ for the unique morphism of F-algebras such that $f = \mathrm{I}_{F,x,a}(f) \circ \eta_F x$. Sometimes, as here, it is convenient to use applicative rather than subscript notation with natural transformations.

Proceeding to the calculation, it will be convenient to write H for ΣT. The unit is:

$$I \xrightarrow{\eta_H} H^* \xrightarrow{\eta H^*} TH^*$$

where η is the unit of T. For the multiplication, we first give the distributive law:

$$\lambda^* : H^*T \to TH^*$$

whose existence is shown in the proof of Theorem 4. It is:

$$\lambda^*_x = \mathrm{I}_{H,Tx,a}(f)$$

where $f = T(\eta_H x)$, $a = (TH^*, \beta)$ and $\beta : H(TH^*)x \to TH^*x$ is the composition:

$$H(TH^*)x \xrightarrow{\lambda_{H^*x}} THH^*x \xrightarrow{T(\alpha_H x)} TH^*x$$

where μ is the multiplication of T. With this, the multiplication of TH^* is:

$$TH^*TH^* \xrightarrow{T\lambda^*H^*} TTH^*H^* \xrightarrow{\mu\mu_H} TH^*$$

where μ_H is the multiplication of H^*. The extension of a morphism $x \to TH^*y$ along the unit of TH^* can, as usual, be defined from the monad structure. It can also be defined more directly via an intermediate morphism $H^*x \to TH^*y$ defined much like λ^* was, using the properties of the free H-algebra on x.

To complete our analysis of the sum of Lawvere theories, we give a closedness result. At present, we do not have a substantial relevant application of this result, but it does bear comparison with a result in [20] that has proved to be of some value, so we mention it:

Definition 4. *Given a countable Lawvere theory L and a category C with countable products, denote by $\mathrm{Mod}^*(L, C)$ the identity-on-objects/fully faithful factorisation of the forgetful functor $U_L : \mathrm{Mod}(L, C) \longrightarrow C$.*

So the objects of $\mathrm{Mod}^*(L, C)$ are the models of L in C and the maps are just maps in C. We have the following theorem:

Theorem 5. *There is a coherent natural equivalence between $\mathrm{Mod}^*(L + L', C)$ and $\mathrm{Mod}^*(L, \mathrm{Mod}^*(L', C))$.*

A simple proof follows directly from the fact that a $(T + T')$-algebra consists of a set x together with both a T-structure and a T'-structure on it.

The formal work in this section enriches without fuss. In particular, the sum is again the correct operation in the enriched setting. And our notation, calculations and formulae lift routinely. The only point that does not enrich routinely is the informal discussion about equational theories: as best we know, there is currently no enriched notion of equational theory corresponding to enriched Lawvere theories. However, that part of the above discussion could be phrased in terms of sketches, for which an enriched account does exist or at least can readily be gleaned from the literature [30].

Spelling out the situation, we have, first, that it is straightforward to calculate sums of free V-monads. They are given by the formula:

$$\Sigma_1^* + \Sigma_2^* = (\Sigma_1 + \Sigma_2)^*$$

Next, we have the following series of results:

Theorem 6. *The category of countable Lawvere V-theories is cocomplete.*

Theorem 7. *Given a V-endofunctor $\Sigma : C \longrightarrow C$ and a V-monad $T : C \longrightarrow C$, if the free V-monads Σ^* and $(\Sigma T)^*$ exist then the sum of V-monads $\Sigma^* + T$ exists in the category of V-monads over C and is given by a canonical V-monad structure on the composite $T(\Sigma T)^*$.*

The above explicit calculation of the canonical monad structure applies verbatim to the enriched situation, using the usual conventions of enriched category theory [26].

Corollary 4. *Given a V-endofunctor $\Sigma : C \longrightarrow C$ and a V-monad $T : C \longrightarrow C$, if Σ^* and $\mu z.T(\Sigma z + -)$ exist, then the sum of V-monads $\Sigma^* + T$ exists in the category of V-monads on C and is given by a canonical V-monad structure on $\mu z.T(\Sigma z + -)$.*

Corollary 5. *Given a V-category C with binary sums, an object E of C, and a V-monad T on C, the sum of the V-monads $(- + E)$ and T exists and is given by the V-monad $T(- + E)$.*

We now specialise to the case where $C = V = \omega$-**Cpo**. There are no interactions between nontermination and raising an exception, or inputting or outputting a value, so they can all be combined by sum. All these possibilities, and more, are covered by the general form of absolutely free ω-**Cpo**-theories discussed above. So with:

$$\Sigma(Q) = \sum_{\lambda \in \Lambda} O_\lambda \times Q^{I_\lambda}$$

Theorem 7 tells us that the sum of Σ^* and T_Ω can be given in the form:

$$(\mu Q.\, \sum_{\lambda \in \Lambda} O_\lambda \times Q_\perp^{I_\lambda} + -)_\perp$$

and Corollary 4 tells us that it can alternatively be given in the form:

$$\mu Q.\, (\sum_{\lambda \in \Lambda} O_\lambda \times Q^{I_\lambda} + -)_\perp$$

So, for example, the combination of all three of exceptions, interactive I/O and nontermination can be written as $(\mu Q.\, E + Q_\perp^I + (O \times Q_\perp) + -)_\perp$ or, alternatively, as $\mu Q.\, (E + Q^I + (O \times Q) + -)_\perp$.

4 The commutative combination of effects

In this section, we consider the tensor product $L \otimes L'$ of countable Lawvere theories L and L' [13,54]. We move immediately to the central definition of the section, for base category **Set**.

The category $\aleph_1$ not only has countable coproducts, but also has finite products, which we denote by $a \times a'$. The object $a \times a'$ may also be seen as the coproduct of a copies of a', so, given an arbitrary map $f' : a' \longrightarrow b'$ in a countable Lawvere theory, it is immediately clear what we mean by the morphism $a \times f' : a \times a' \longrightarrow a \times b'$. We define $f \times a'$ by conjugation, and, in the following, we suppress the canonical isomorphisms.

Definition 5. *Given countable Lawvere theories L and L', the countable Lawvere theory $L \otimes L'$, called the tensor product of L and L', is defined by the universal property of having maps of countable Lawvere theories from L and L' to $L \otimes L'$, with commutativity of all operations of L with respect to all operations of L', i.e., given $f : a \longrightarrow b$ in L and $f' : a' \longrightarrow b'$ in L', we demand commutativity of the diagram:*

$$\begin{tikzcd}
a \times a' \arrow[r, "a \times f'"] \arrow[d, "f \times a'"'] & a \times b' \arrow[d, "f \times b'"] \\
b \times a' \arrow[r, "b \times f'"'] & b \times b'
\end{tikzcd}$$

The tensor product always exists because it is defined by operations and equations, or equivalently by a sketch [5,6]. Its existence also follows by appeal to Appendix A. In terms of equational theories, the tensor product is also easy to describe: one takes the operation symbols and equations of each of the two theories, again renaming the operation symbols if necessary to avoid confusion, and adds equations expressing that each operation of one theory commutes with each operation of the other. For example if f is a binary operation symbol of one theory and g is a ternary one of the other, one adds the equation:

$$f(g(x_{11}, x_{12}, x_{13}), g(x_{21}, x_{22}, x_{23})) = g(f(x_{11}, x_{21}), f(x_{12}, x_{22}), f(x_{13}, x_{23}))$$

The equational form of the commutative combination can be useful as a basis for the calculation of specific examples.

As usual, we first develop the abstract theory in the unenriched case following [13,54], giving the enriched version afterwards.

Proposition 2. *There is a canonical extension of the tensor product $\otimes$ to a symmetric monoidal structure on the category of countable Lawvere theories.*

A proof for this proposition is elementary. The unit for the tensor product is the initial Lawvere theory, i.e, the theory generated by no operations and no equations. This is the initial object of the category of Lawvere theories, so is also the unit for the sum. It induces the identity monad.

The result gives some indication of the definiteness of the tensor product, but not much: there are typically many symmetric monoidal structures on categories, such as finite product or finite coproduct, and usually many others satisfying no particular universal property. But what is much less common, and is central to the proof of our main theorem about the combination of side-effects with other effects, and indeed is central to the understanding of what commutativity means, is a characterisation of $L \otimes L'$ in terms of the categories of models of L and L'. Relevant, delicate 2-categorical analysis supporting this is in Appendix A; here, we simply state the characterisation in its own terms, the result following from Theorem 16.

Theorem 8. *For any category C with countable products, there is a coherent equivalence of categories between* $\mathrm{Mod}(L \otimes L', C)$ *and* $\mathrm{Mod}(L, \mathrm{Mod}(L', C))$.

The analysis of this section extends readily to the enriched setting. The V-category $V_{\aleph_1}$ not only has countable tensors but also has finite products, just as the ordinary category $\aleph_1$ not only has countable coproducts but also has finite products. Our analysis of $a \times f'$ in the unenriched setting extends routinely to the enriched setting, except here, of course, we must express the analysis in terms of the object $L(a', b')$ of V rather than in terms of an arrow $f' : a' \longrightarrow b'$. The key fact is that the cotensor $(a^x)^y$ is canonically isomorphic to the cotensors $a^{(x \times y)}$ and $(a^y)^x$. Consistently with this, we must express the commutativity condition of the theorem in terms of homobjects of V rather than in terms of arrows like f'.

Definition 6. *Given countable Lawvere V-theories L and L', the countable Lawvere V-theory $L \otimes L'$, which we call the tensor product of L and L', is defined by the universal property of having maps of countable Lawvere V-theories from L and L' to $L \otimes L'$, subject, suppressing canonical isomorphisms, to commutativity of:*

$$\begin{tikzcd}
L(a,b) \times L'(a',b') \arrow[r] \arrow[d] & L(a \times b', b \times b') \times L'(a \times a', a \times b') \arrow[d, "\mathrm{comp}"] \\
L(a \times a', b \times a') \times L'(b \times a', b \times b') \arrow[r, "\mathrm{comp}"'] & L(a \times a', b \times b')
\end{tikzcd}$$

Once again, the tensor product exists because it is determined by the free theory on an enriched sketch [30]. But it may equally, indeed more elegantly, be proved to exist by appeal to an enriched version of Appendix A, from which the following result also follows:

Theorem 9. *The construction $L \otimes L'$ is symmetric monoidal on the category of countable Lawvere V-theories. Further, for any V-category C with countable cotensors, there is a coherent equivalence of V-categories between* $\mathrm{Mod}(L \otimes L', C)$ *and* $\mathrm{Mod}(L, \mathrm{Mod}(L', C))$.

5 Calculating the tensor product, in particular of side-effects with other effects

Here, we calculate certain tensor products in more detail, particularly that for side-effects with other computational effects. Our central result shows that, under appropriate hypotheses, our theory of the tensor product of computational effects agrees with Moggi's definition of the side-effects monad transformer. It is as follows:

Theorem 10. *Let L_S be the countable Lawvere theory for side-effects, let L be any countable Lawvere theory, and let C be a locally countably presentable category. Then the induced V-monad $T_{L_S \otimes L}$ on C is isomorphic to $T_L(S \otimes -)^S$.*

Proof. By Theorem 9, the category $\mathrm{Mod}(L_S \otimes L, C)$ is coherently equivalent to to the category $\mathrm{Mod}(L_S, \mathrm{Mod}(L, C))$. Since C is locally countably presentable, there is an adjunction $F_L \dashv U_L : \mathrm{Mod}(L, C) \to C$ with $T_L = U_L F_L$, and, further, the category $\mathrm{Mod}(L, C)$ is complete and cocomplete, and so has countable products and coproducts. Therefore, by [47], the monad induced by L_S on $\mathrm{Mod}(L, C)$ is $(S \times -)^S$.

Right adjoints preserve products and left adjoints preserve coproducts. So $T_{L_S \otimes L}$, which is the monad given by the composite forgetful functor from the category $\mathrm{Mod}(L_S, \mathrm{Mod}(L, C))$ to C, must be given by:

$$T_{L_S \otimes L} = U_L(S \times F_L -)^S \cong (U_L F_L(S \times -))^S \cong T_L(S \times -)^S$$

as required.

We do not require rank, in particular countability, for the result. We could define a notion of theory that does not involve a rank, retain a correspondence with strong monads, and make the commutative combination of the theorem, but the general theory becomes more complicated because, as remarked in the introduction, tensor products of such theories seem not always to exist.

Corollary 6. *The side-effects theory for $S = V^L$ is the L-fold tensor product of the side-effects theory for $S = V$ (i.e., the case where $L = 1$).*

Proof. By the theorem, the tensor product of two side-effects theories, one for S and the other for S', is the side-effects theory for $S \times S'$. Now use induction and the finiteness of L.

Explicit formulae follow for the monad structure of $T_L(S \times -)^S$. Let η and μ be the unit and multiplication of T_L. Then the unit of $T_L(S \times -)^S$ at x is $\mathrm{Curry}(\eta_{(S \times x)})$, where Curry is the transpose function, and the monad multiplication at x is:

$$T_L(S \times (T_L(S \times x)^S))^S \xrightarrow{T_L(\mathrm{eval}\sigma)^S} (T_L T_L(S \times x))^S \xrightarrow{\mu^S_{S \times x}} T_L(S \times x)^S$$

where eval is the evaluation function, and σ is the symmetry of product.

The theorem generalises readily to the enriched setting. Generalising Example 8 from ω-**Cpo**-enrichment to V-enrichment, we allow the ω-cpo of values to be instead any countably presentable object of V, but again retain the finite set of locations. The countable Lawvere V-theory L_S is again freely given by the diagrams in [47], but allowing the more general kind of state. And, once more generalising [47], we obtain that, if C is any V-category with tensors and cotensors of countably presentable objects, then the V-monad induced on C by L_S is $(S \otimes -)^S$.

Theorem 11. *Let L_S be the countable Lawvere V-theory for side-effects, let L be any countable Lawvere V-theory, and let C be locally countably presentable as a V-category. Then the induced V-monad $T_{L_S \otimes L}$ on C is isomorphic to $T_L(S\otimes -)^S$.*

Proof. The proof agrees verbatim with that of Theorem 10, subject to systematic reference to enrichment.

The analogous formulae for the monad structure hold in the enriched case; note too that the explicit description of the V-monad for the tensor agrees with that in the unenriched case.

As an immediate application of the theorem, we see that the ω-**Cpo**-monad induced by the commutative combination of the ω-**Cpo**-theories for side-effects and nontermination is $((S\otimes -)_\bot)^S$. The conditions for the tensor product amount here to saying that the operations for state commute with Ω, that is, they are *strict*. For example, we have the equation:

$$update_{\langle l,v\rangle}(\Omega) = \Omega$$

which reflects the computational idea that we do not wish to distinguish between nontermination and an updating operation immediately followed by nontermination. This contrasts with, for example, the combination of interactive I/O and nontermination where one does not wish to impose strictness.

By analysis of the structure of our argument for the commutative combination of side-effects with other effects, we can generalise a little. It follows from Theorem 9 that, if C is locally countably as a V-category, then the V-category $\mathrm{Mod}(L' \otimes L, C)$ is coherently equivalent to the V-category $\mathrm{Mod}(L', \mathrm{Mod}(L, C))$. And we know that the V-monad $T_{L' \otimes L}$ is given by the left adjoint to the forgetful V-functor from $\mathrm{Mod}(L' \otimes L, C)$ to C. But $\mathrm{Mod}(L, C)$ is locally countably presentable as a V-category since C is. So the forgetful V-functor:

$$U_{L'} : \mathrm{Mod}(L', \mathrm{Mod}(L, C)) \longrightarrow \mathrm{Mod}(L, C)$$

has a left V-adjoint $F_{L'}$. Combining these observations, it follows that $T_{L' \otimes L}$ is given by the composite $U_L T_{L'} F_L$, where $T_{L'} = U_{L'} F_{L'}$ is the V-monad on $\mathrm{Mod}(L, C)$ induced by L'. The proof of our characterisation of the commutative combination of side-effects with other effects simply amounted to describing $T_{L'}$, then using the preservation properties of left and right adjoints in order to calculate the V-monad $U_L T_{L'} F_L$.

Using this technique, we now analyse the combination of nondeterminism and nontermination. Here $V = \omega$-**Cpo**, $L' = L_N$ and $L = L_\Omega$.

Proposition 3. *1. The unit map η_{T_N} is strict at ω-cpos with a least element.*

2. The monad T_N cuts down to a functor on $\mathrm{Mod}(L_\Omega, \omega\text{-}\mathbf{Cpo})$*, taken as the category of ω-cpos with a least element and strict ω-continuous maps.*

Proof. 1. For the first claim, let P be an ω-cpo with a least element. We know that for any ω-continuous map $f : P \to T_N(P)$ there is a unique ω-continuous homomorphism $\overline{f} : T_N(P) \to T_N(P)$ that makes the following diagram commute:

$$\begin{tikzcd}
P \arrow[d, "\eta_N"'] \arrow[dr, "f"] & \\
T_N(P) \arrow[r, "\overline{f}"'] & T_N(P)
\end{tikzcd}$$

(we are omitting object subscripts on unit maps for the sake of notational clarity). Moreover, this assignment of homomorphisms to continuous functions is itself continuous. Now consider the constant map $k : P \to T_N(P)$ sending every element of P to $\eta_N(\perp)$; one easily sees that $\overline{k}$ is the constant map sending every element of $T_N(P)$ to $\eta_N(\perp)$. But then, since η_N is monotonic one has that $k \le \eta_N$, and so $\overline{k} \le \overline{\eta_N} = \mathrm{id}_{T_N(P)}$, which proves that $\eta_N(\perp) = \perp$, as required.

2. For the second claim, we first let P be an ω-cpo with a least element, to show that $T_N(P)$ is in $\mathrm{Mod}(L_N, \mathrm{Mod}(L_\Omega, \omega\text{-}\mathbf{Cpo}))$. We know from the first part that $T_N(P)$ has a least element; we also have to show that the semilattice operation $\cup$ is a morphism in $\mathrm{Mod}(L_\Omega, \omega\text{-}\mathbf{Cpo}))$ which, since it is certainly continuous, amounts to showing that $\perp \cup \perp = \perp$, which follows from idempotence. Next, we have to show that T_N preserves morphisms. So let Q be another ω-cpo with a least element, and consider a strict ω-continuous map $f : P \to Q$. Then as $T_N(f) = \overline{\eta_N f}$ and the unit map is strict, so is $T_N(f)$, concluding the proof.

It follows directly from this proposition that T_N cuts down to the monad T_C considered more generally above, inheriting its monadic structure from T_N. We therefore calculate that the commutative combination of T_N and T_Ω is $U_\Omega T_C F_\Omega = T_N U_\Omega F_\Omega = T_N T_\Omega$, with the first equality arising from the fact that T_C lifts T_N from ω-**Cpo** to $\mathrm{Mod}(L_\Omega, \omega\text{-}\mathbf{Cpo})$. It is worth recollecting that such liftings correspond to distributive laws, here one of T_Ω over T_N, with the resulting monad being again $T_N T_\Omega$.

The case of the combination of nondeterminism and nontermination is odd. The extra conditions imposed by the tensor product amount to saying that the two operations commute, i.e., that the following equation holds:

$$\Omega \vee \Omega = \Omega$$

However this is an instance of idempotency and so the sum and tensor product coincide in this case. There is an argument for regarding the combination as a tensor product since we use the theory of tensors to give an explicit form for the combination.

The combination of probabilistic nondeterminism and nontermination is entirely analogous to that of nondeterminism and nontermination, provided one uses the theory of countable probabilistic nondeterminism. So the analogue of Proposition 3 holds, with the analogous proof, and one can view the combination as a tensor product of the two theories, with the induced monad being given by the composition of that for probabilistic nondeterminism with that for nontermination.

Finally, we should mention here an argument for treating the combination of nondeterminism and nontermination as a sum rather than a tensor. It is natural to give the semantics of deadlock as the empty set, as in, e.g., [35,1,2], and so to consider a variant L_Z of L_N which is the theory of a commutative semilattice with a zero. But then one would not consider the tensor product of this with L_Ω as that would identify (the constants for) nontermination and deadlock. So one must take the combination to be the sum.However one then finds that the induced monad is *not* isomorphic to either composition of T_Ω and T_Z, the monad on ω-**Cpo** induced by L_Z. This leaves the analogy with L_N somewhat imperfect, and so we choose not to prefer either of the two ways of regarding the combination of L_N and L_Ω.

Having considered the combination of exceptions and interactive I/O with nontermination in Section 3, we have now considered all combinations of the individual effects under discussion in this paper with nontermination. It is then interesting to consider combinations of pairs of effects with nontermination, since that is what we require when combining nontermination effects under the additional requirement of being able to model recursion.

For exceptions, we can combine the above results on the combination of a single effect with nontermination by application of Corollary 5. For example, combining exceptions with side-effects and nontermination, we wish the operations associated with side-effects and nontermination to commute, but no others. We therefore want the ω-**Cpo**-monad induced by the ω-**Cpo**-theory $L_E + (L_S \otimes L_\Omega)$, and that is $((S \otimes (- + E))_\perp)^S$.

For side-effects, we can instead combine the above results on the combination of a single effect with nontermination by application of Theorem 11. For example, combining side-effects with interactive I/O and nontermination one wants the ω-**Cpo**-monad induced by $L_S \otimes (L_{I/O} + L_\Omega)$. We already know that the ω-**Cpo**-monad induced by $L_{I/O} + L_\Omega$ is $\mu R.\,(R^I + (O \otimes R) + -)_\perp$; so the ω-**Cpo**-monad we seek is $(\mu R.(R^I + (O \otimes R) + (S \times -))_\perp)^S$.

For interactive I/O, we can combine the above results on the combination of a single effect with nontermination by application of Theorem 7 or Corollary 4. For example, for the combination of interactive I/O with nondeterminism and nontermination we want the monad on ω-**Cpo** induced by $L_{I/O} + (L_N \otimes L_\Omega)$. By Corollary 4 this is $\mu Q.\,T_N((Q^I + (O \times Q) + -)_\perp)$. If we take the value of this monad at the initial (empty) ω-cpo, we get a solution to the following recursive domain equation:

$$P \cong T_N((P^I + (O \times P))_\bot)$$

This is a variation on the various domains of processes used in the denotational semantics of Milner's CCS and its variants, see again [35,1,2]. The difference lies in the exact nature of the interactive I/O monad chosen (see Section 7 below for a general such monad) and the use of the theory of semilattices with a zero to model deadlock.

Finally, we present two more examples of the commutative combination of effects. First we consider the treatment of resumptions as used for the semantics of parallel imperative programming languages. As mentioned above, Cenciarelli and Moggi's resumptions monad is $\mu z.T(z + x)$ in a category C with binary sums. Using Corollary 4 this gives the usual notion of resumptions in ω-**Cpo** taking T to be $T_N T_\Omega(S \times -)^S$ and $x = 1$. In terms of ω-**Cpo**-theories this is the monad induced by $L_d + (L_S \otimes (L_N \otimes L_\Omega))$ where we recall that L_d is the theory of a unary operator with no equations, regarded as the operation of suspending computation. Note that in this theory d does not commute with nondeterminism. If we wanted d to commute with nondeterminism, but not with state or nontermination, we would naturally be led to consider the theory $L_{HP} = (L_d + (L_S \otimes L_\Omega)) \otimes L_N$ instead. We now show that the work of Hennessy and Plotkin [18] can be considered in terms of the latter theory.

They work with an algebra of resumptions in the category $ND_\bot$ of ω-**Cpo**-semilattices with a least element, which is $\mathrm{Mod}(L_N L_\Omega, \omega\text{-}\mathbf{Cpo})$; the algebra is given by:

$$R' = \mu B.\, (S \otimes (B_\bot + I))^S$$

where I is the tensor unit of $ND_\bot$, and $(-)_\bot$ is the comonad of the adjunction:

$$F \dashv U : ND_\bot \to ND$$

where ND is the category of ω-**Cpo**-semilattices. This is not in the form we want; we instead switch to the category ND and consider the algebra:

$$R = \mu A.\, ((S \otimes (A + I))_\bot)^S$$

where now $(-)_\bot$ is the monad UF and I is the tensor unit of ND. One can show that $R \cong UR'$ and for the semantic analysis of [18] one can switch to R.

This is now in the required form and the corresponding resumptions monad on ND is $\mu A.T(A + -)$ where $T = ((S \otimes -)_\bot)^S$. By Theorem 11, the latter is the ω-**Cpo**-monad on ND induced by $L_S \otimes L_\Omega$, and, further, by the discussion on absolutely free ω-**Cpo**-theories, the ω-**Cpo**-monad induced by L_d is the free ω-**Cpo**-monad on the identity signature ω-**Cpo**-functor. So by Corollary 4, the ω-**Cpo**-monad on ND induced by the theory $L_d + (L_S \otimes L_\Omega)$ is the resumptions monad $\mu A.T(A + -)$ just described.

It then follows that, by the above analysis of the tensor product of theories, the monad T_{HP} on ω-**Cpo** induced by L_{HP} is given by $U_N(\mu A.T(A + -))F_N$.

We could thus view [18] as indeed implicitly working with the theory L_{HP} as, qua ω-cpo:

$$R = U_N(\mu A.T(A + I)) \cong U_N(\mu A.T(A + F_N(1))) \cong T_{HP}(1)$$

Another, perhaps more natural, possibility, would be to recast [18] in terms of an ND-theory of the form $L_d + (L_S \otimes L_\Omega)$. However ND is locally countably presentable as a symmetric monoidal category rather than as a cartesian closed one, and cartesian closure is the prevailing assumption of this paper. All the theory of this paper generalises to such symmetric monoidal closed categories except, perhaps, that associated with the state monad. That, prima facie, explicitly depends on cartesian closure as the axioms in [47] make explicit use of diagonals on arities. We nonetheless expect a suitable generalisation to be found, but, for the moment, the situation remains unclear.

For our last example of the commutative combination of effects, let M be a monoid in V and consider the combination of any computational effect, given by L or equivalently T_L, with the *complexity* monad $M \otimes -$. We have assumed throughout the paper that V is cartesian closed, so $M \otimes -$ is just $M \times -$, but our argument here is more general, so we indicate that by our notation. There is a canonical distributive law of the monad $M \otimes -$ over T_L, obtained using the strength $t : M \otimes T_L - \longrightarrow T_L(M \otimes -)$ of T_L. So $T_L(M \otimes -)$ acquires a canonical monad structure.

Theorem 12. *Let L be any countable Lawvere V-theory, let M be a monoid in V, and let L_M be the Lawvere V-theory induced by the monad $M \otimes -$. Then $T_L(M \otimes -)$ is the V-monad induced by $L_M \otimes L$.*

Proof. To give a model of L in $(M \otimes -)$-Alg is equivalent to giving a model $m : L \longrightarrow V$ of L in V, together with an M-action $\alpha : M \otimes m(1) \longrightarrow m(1)$ on $m(1)$, such that the corresponding map $\bar{\alpha} : m(1) \longrightarrow m(1)^M$ is a map of models. This in turn is equivalent to giving a T_L-algebra (x, β) and an M-action $\alpha : M \otimes x \longrightarrow x$ on x such that $\bar{\alpha} : x \longrightarrow x^M$ is a map of T_L-algebras. But that in turn is equivalent to giving a $T_L(M \otimes -)$-algebra by generalities about distributive laws of monads [5]. These equivalences are all functorial, yielding an isomorphism from $T_{L_M \otimes L}$-Alg to $T_L(M \otimes -)$-Alg and hence an isomorphism of monads between $T_{L_M \otimes L}$ and $T_L(M \otimes -)$.

This result does not make substantial use of size conditions on either V or L and can be generalised readily to an arbitrary strong monad.

Corollary 7. *The tensor product of $M \otimes -$ and $M' \otimes -$ is $(M \otimes M') \otimes -$.*

6 Operation transformers

We now consider a theory of operation transformers, specifically algebraic operation transformers. Algebraic operations were studied in [45,46] in an enriched setting, with V complete, cocomplete and symmetric monoidal closed; here we restrict our attention to the case where V is locally countably presentable as a cartesian closed category. Let T be a strong monad on V. Then an *operation* of sort (a, b) on T is a transformation of the form:

$$\alpha_x : (Tx)^a \longrightarrow (Tx)^b$$

where a and b are objects of V; it is *algebraic* if it is a V-natural transformation with respect to x as an object of $\mathrm{Kl}(T)$. For example, the nonempty finite powerset monad $\mathcal{F}^+$ supports the binary choice algebraic operation:

$$\vee_X : (\mathcal{F}^+X)^2 \longrightarrow \mathcal{F}^+X$$

where $\vee_X(u, v) = u \cup v$. Other **Set**-based examples of algebraic operations detailed in [46] are a binary probabilistic choice operation $+_r$ for every real number r in the interval $[0, 1]$ on the monad of finite distributions, a *raise* operation of sort $(0, E)$ on the monad for exceptions, *read* and *write* operations of sorts $(I, 1)$ and $(1, O)$ respectively on the monad $T_{I/O}$ for interactive I/O, and *lookup* and *update* operations of sorts (V, L) and $(1, L \times V)$ respectively, on the monad for side-effects. As mentioned in the introduction, the operation *handle*, for handling exceptions, is not an algebraic operation. It can be considered as an operation of sort $(1 + E, 1)$, being defined as the composition

$$(TX)^{1+E} \longrightarrow (X \times (TX)^E) + (E \times (TX)^E) \overset{[\eta_T\pi_0,\mathrm{eval}]}{\longrightarrow} TX$$

However it is only natural with respect to **Set**, not $\mathrm{Kl}(- + E)$.

The main result of [45,46] asserted that the enriched Yoneda embedding induces a bijection between maps $b \longrightarrow a$ in $\mathrm{Kl}(T)$, i.e., maps $b \longrightarrow Ta$ in V, and algebraic operations $\alpha_x : (Tx)^a \longrightarrow (Tx)^b$. The correspondence is as follows: given a map $f : b \longrightarrow a$ in $\mathrm{Kl}(T)$, the corresponding algebraic operation is:

$$(Tx)^a \quad\cong\quad (a \Rightarrow x) \xrightarrow{f \Rightarrow x} (b \Rightarrow x) \quad\cong\quad (Tx)^b$$

where $x \Rightarrow y$ is the Kleisli exponential; and given an algebraic operation α_x, the corresponding map in $\mathrm{Kl}(T)$ is:

$$1 \xrightarrow{\ulcorner \eta_a \urcorner} (Ta)^a \xrightarrow{\alpha_a} (Ta)^b$$

The map in $\mathrm{Kl}(T)$ is called the *generic effect* corresponding to α, and, in the case of infinitary operations such as *lookup* and *update*, the generic effect typically appears more directly in a programming language than does the corresponding algebraic operation [46].

In all our examples, the objects a and b lie in the full sub-V-category of V given by $V_{\aleph_1}$. Recall that the Lawvere V-theory L_T induced by a strong monad T is precisely the restriction of $\mathrm{Kl}(T)^{\mathrm{op}}$ to the objects of $V_{\aleph_1}$. So we can reformulate a mild restriction of the main result of [45,46] to read:

Theorem 13. *Given a strong monad T with countable rank on V, the enriched Yoneda embedding induces a bijection between maps $a \longrightarrow b$ in L_T and algebraic operations:*

$$\alpha_x : (Tx)^a \longrightarrow (Tx)^b$$

This result yields the liftings we seek: given countable Lawvere V-theories L and L', we have coprojections $\mathrm{inl} : L \longrightarrow L + L'$ and $\mathrm{inr} : L' \longrightarrow L + L'$. So, by two applications of the theorem and one application of the coprojection, each algebraic operation on T_L is sent to an algebraic operation on $T_{L+L'}$; ditto for L'. There are also canonical maps $L \longrightarrow L \otimes L'$ and $L' \longrightarrow L \otimes L'$, yielding liftings of algebraic operations on T_L and $T_{L'}$ to algebraic operations on $T_{L\otimes L'}$ in just the same way.

To give the coprojection $\mathrm{inl} : L \longrightarrow L + L'$ is equivalent to giving a corresponding coprojection $\mathrm{inl} : T_L \longrightarrow T_L + T_{L'}$, and applying the functor inl to an arrow $a \longrightarrow b$ in L is equivalent to composing the corresponding monad map inl with $a \longrightarrow b$ seen as a generic effect $b \longrightarrow T_L a$. Ditto for inr and for replacing $+$ by $\otimes$.

Motivated by these remarks, we make a definition of operation transformer that has the spirit of the idea of monad transformer but for algebraic operations rather than monads. Given a strong monad T on V, define the V-category $\mathrm{Op}(T)$ to have the same objects as V, with $\mathrm{Op}(T)(a, b)$ defined, using the bijection of [46], to make an arrow of $\mathrm{Op}(T)$ an algebraic operation:

$$\alpha_x : (Tx)^a \longrightarrow (Tx)^b$$

So, $\mathrm{Op}(T)$ is isomorphic to $\mathrm{Kl}(T)^{\mathrm{op}}$ [46], and so there is a canonical V-functor $J_T : V^{\mathrm{op}} \longrightarrow \mathrm{Op}(T)$. Moreover, when T has countable rank, the restriction of $\mathrm{Op}(T)$ to the objects of $V_{\aleph_1}$ is, by the theorem, isomorphic to L_T: size issues do not play a major role here.

Definition 7. *Given strong monads T and T', an* operation transformer *from* $\mathrm{Op}(T)$ *to* $\mathrm{Op}(T')$ *is a V-functor* $\mathrm{op} : \mathrm{Op}(T) \longrightarrow \mathrm{Op}(T')$ *commuting with the canonical V-functors J_T and $J_{T'}$.*

It follows from the definitions that, for every strong monad T, the V-category $\mathrm{Op}(T)$ has V-cotensors and every operation transformer preserves them.

Proposition 4. *To give an operation transformer from* $\mathrm{Op}(T)$ *to* $\mathrm{Op}(T')$ *is equivalent to giving a map of strong monads from T to T'. If T and T' have countable rank, to give an operation transformer is further equivalent to giving a map of Lawvere V-theories from L_T to $L_{T'}$.*

Suppose that we have an operation transformer from $\mathrm{Op}(T)$ to $\mathrm{Op}(T')$, with corresponding map of monads $\tau : T \rightarrow T'$, that α is an algebraic operation of sort (a, b), and that α' is the result of applying the operation transformer to α. Then α' is a *lifting* of α in the sense that the following diagram commutes for all objects x:

$$\begin{tikzcd}
(Tx)^a \arrow[r, "\alpha_x"] \arrow[d, "(\tau_x)^a"'] & (Tx)^b \arrow[d, "(\tau_x)^b"] \\
(T'x)^a \arrow[r, "\alpha'_x"'] & (T'x)^b
\end{tikzcd}$$

To see this, suppose α corresponds to the generic effect $\bar{\alpha}\colon b \longrightarrow Ta$. Then α is given by the composite:

$$b \times (Tx)^a \xrightarrow{\bar{\alpha} \times (Tx)^a} Ta \times (Tx)^a \longrightarrow Tx$$

and α' is given by the composite:

$$b \times (T'x)^a \xrightarrow{\bar{\alpha} \times (T'x)^a} Ta \times (T'x)^a \xrightarrow{\tau_a \times (T'x)^a} T'a \times (T'x)^a \longrightarrow T'x$$

The dinaturality of evaluation then yields the commutativity of the diagram.

For explicit constructions of operation transformers, first consider sum. Let C be locally countably presentable as a V-category. Then, given a V-endofunctor Σ and a V-monad (T, μ, η) on C, and assuming Σ^* and $(\Sigma T)^*$ exist, we have shown that $\Sigma^* + T$ exists and is given by a canonical V-monad structure on $T(\Sigma T)^*$. It is routine to verify that the coprojections $\Sigma^* \longrightarrow \Sigma^* + T$ and $T \longrightarrow \Sigma^* + T$ are given by the V-monad maps:

$$\Sigma^* \xrightarrow{(\Sigma\eta_T)^*} H^* \xrightarrow{\eta_T H^*} TH^*$$

and:

$$T\eta_{H^*}\colon T \longrightarrow TH^*$$

where, as before, we now find it convenient to write H for ΣT. So, the liftings of generic effects $b \longrightarrow \Sigma^* a$ and $b \longrightarrow Ta$ are given by composition with these V-monad maps. The transformer $\mathrm{Op}(T) \longrightarrow \mathrm{Op}(TH^*)$ can be described directly as follows: the lifting of an algebraic operation:

$$\alpha_x\colon (Tx)^a \longrightarrow (Tx)^b$$

is:

$$\alpha_{H^*x}\colon (TH^*x)^a \longrightarrow (TH^*x)^b$$

And a partial description of the transformer $\mathrm{Op}(\Sigma^*) \longrightarrow \mathrm{Op}(TH^*)$ is given as follows. By the enriched Yoneda lemma, to give a map $b \longrightarrow \Sigma a$ is equivalent to giving a V-natural transformation:

$$\beta_x\colon x^a \longrightarrow (\Sigma x)^b$$

natural in x as an object of V. But a map $b \longrightarrow \Sigma a$ gives rise, by composition with the unit $\Sigma a \longrightarrow \Sigma^* a$ to a generic effect $b \longrightarrow \Sigma^* a$ and hence to an algebraic operation:

$$\beta^*_x : (\Sigma^* x)^a \longrightarrow (\Sigma^* x)^b$$

on Σ^*. So every V-natural transformation β as above gives rise to an algebraic operation β^*, to which one may apply the operation transformer. It follows by routine calculation that the lifting of β^* is:

$$(TH^*x)^a \xrightarrow{\beta_{TH^*x}} (HH^*x)^b \xrightarrow{(\alpha_H x)^b} (H^*x)^b \xrightarrow{(\eta_T H^* x)^b} (TH^*x)^b$$

We now consider various examples for $C = \mathbf{Set}$; corresponding examples with nontermination for $C = \omega\text{-}\mathbf{Cpo}$ are readily available.

Example 11. **Exceptions** For modelling exceptions, Σ is the constant at E, and so $\Sigma^* = (\Sigma T)^* = - + E$. and the canonical $- + E \to T(- + E)$ is $\eta(- + E)$. The operation *raise*, of sort $(0, E)$, is $\ulcorner \mathrm{inr} \urcorner$ at X; it arises from the identity map $E \to \Sigma 0$ and its lifting is $\ulcorner \eta_{X+E} \mathrm{inr} \urcorner$ at X. The operation *handle* is not an algebraic operation, so our theory does not include it.

Example 12. **Interactive I/O** The monad for interactive I/O is given by:

$$T_{I/O} X = \mu Y. \, (Y^I + (O \times Y) + X)$$

which is Σ^*, where $\Sigma = \Sigma_{I/O}$. The generic effects for interactive input and output, $\mathit{read} : 1 \longrightarrow T_{I/O}(I)$ and $\mathit{write} : O \longrightarrow T_{I/O}(1)$, then arise from the maps $\mathrm{inr} \ulcorner \mathrm{id} \urcorner : 1 \to \Sigma I$ and $\mathrm{inl}(\mathrm{id}, \mathrm{t}) : O \to \Sigma 1$ respectively; they are $(\eta_\Sigma \Sigma I) \mathrm{inr} \ulcorner \mathrm{id} \urcorner$ and $(\eta_\Sigma \Sigma 1) \mathrm{inl}(\mathrm{id}, \mathrm{t})$.

The liftings of these generic effects are:

$$\mathit{read}' : 1 \longrightarrow T(\mu Y. \, ((TY)^I + (O \times TY) + I))$$

and:

$$\mathit{write}' : O \longrightarrow T(\mu Y. \, ((TY)^I + (O \times TY) + 1))$$

where $\mathit{read}' = \eta_{(\Sigma T)^*} (\eta_\Sigma \Sigma T I) \mathrm{inr} \ulcorner \eta_I \urcorner$ and $\mathit{write}' = \eta_{(\Sigma T)^*} (\eta_\Sigma \Sigma T 1) \mathrm{inl}(\mathrm{id}, \eta_1 \mathrm{t})$. In turn, *read* and *write* themselves are liftings of generic effects *read* and *write* for the input monad and the output monad respectively.

We do not know any more direct expression of the corresponding algebraic operations than using the formula given above for obtaining them from the generic effects. The generic effects appear more typically in programming languages [46], although one does see them in process languages such as Milner's CCS [36].

Example 13. **Side-Effects** For the tensor product of L_S with any countable Lawvere theory L, the theory maps $L \longrightarrow L \otimes L_S$ and $L \longrightarrow L \otimes L_S$ correspond to the evident monad maps:

$$T_L \longrightarrow T_L(S \times -)^S$$

and:

$$\eta_L(S\times -)^S : (S\times -)^S \longrightarrow T_L(S\times -)^S$$

As it is the generic effects for side-effects rather than the corresponding algebraic operations that typically appear directly in programming languages [46], we just consider them. The generic effect corresponding to the algebraic operation *lookup* is:

$$\mathit{deref} : L \longrightarrow (S\times V)^S$$

defined on **Set** by:

$$\mathit{deref}(l)(s) = (s, s(l))$$

Its lifting by composition with the $\eta_L(S\times -)^S$ is the generic effect:

$$\mathit{deref}' : L \longrightarrow T(S\times V)^S$$

defined by:

$$\mathit{deref}'(l)(s) = \eta_T(s, s(l))$$

The situation for the generic effect $\mathit{assign} : L\times V \longrightarrow (S\times 1)^S$ corresponding to the algebraic operation *update* is similar.

Liftings of operations on T_L are simply characterised. Suppose that α is such an operation, say of sort (A, B). Then its lifting to an operation on $T_L(S\times -)^S$ is:

$$(T_L(S\times X)^S)^A \cong (T_L(S\times X)^A)^S \xrightarrow{(\alpha_{S\times X})^S} (T_L(S\times X)^S)^B \cong (T_L(S\times X)^B)^S$$

7 Discussion

In this paper we have shown how to combine different effects in terms of natural operations on Lawvere theories rather than on the corresponding monads. That has allowed us to give an account of two standard ways to combine effects: taking their sum, and taking their commutative combination, or tensor product. We then derived explicit forms for some corresponding combinations of monads. Sum and tensor account for most of the examples in which effects are combined in practice; we have yet to consider distributive combinations, local state and continuations. We have also given canonical ways to lift algebraic operations when adding effects; we have yet to consider other operations such as *handle*.

Of course, one may combine more than two effects, so the operations we define may be used several times. This leads us to propose a formula for combining exceptions, side-effects, interactive I/O and (binary) nondeterminism:

$$L_E + (L_S \otimes (L_{I/O} + L_N))$$

where we have used the standard combinations, as described above, of exceptions, state, and side-effects with other effects. In terms of monads this is :

$$TX = (\mu Y.\, \mathcal{F}(Y^I + (O\times Y) + (S\times(X+E))))^S$$

We also propose similar formulae replacing L_N by L_{P_ω}, or other forms of nondeterminism, and also for combinations of just some of these effects; the latter amounts to replacing the Lawvere theories for the effects not combined by the trivial (initial) Lawvere theory. These formulae yield exactly those interactions between operations given when considering binary combinations above, and so:

1. The equations for each effect are retained in the theory for the combination, and no more are added.
2. The equations for the binary interactions we have considered above are retained in the combination, and no more are added.
3. There are no ternary, or higher, interactions.

We do not have any independent justification of these formulae; perhaps a theory of observation of computational effects would help. The formulae proposed do however coincide with all the cases we are aware of in the literature.

Observe that the formula is, in a sense, linear, having the form:

$$F_E(F_S(F_{I/O}(L_N)))$$

where each F is derived from $+$ or $\otimes$ applied to a particular Lawvere theory. This explains why monad transformers have appeared in functional programming [7]: one has a monad transformer for each effect and method of its combination, and, modulo our correspondence with Lawvere theories, those monad transformers are precisely the F's. This paper yields the additional point that they arise from general binary operations on Lawvere theories; indeed for state there are two relevant operations and two possible monad transformers. It is less clear whether nondeterminism is as simple because of the symmetry involved in the combination of internal and external nondeterminism [17].

The above discussion does not, of course, take recursion and nontermination into account. Here one would start with nontermination, and then add the other effects:

$$L_E + (L_S \otimes (L_{I/O} + (L_N \otimes L_\Omega)))$$

where we are now working in ω-**Cpo**.

The linearity of these formulae is all the more remarkable if one considers the whole range of possibilities for combining several theories by sum and tensor. We can illustrate these best by an example. Consider four theories arranged in a square:

$$\begin{tikzcd}
L_1 \arrow[r, dash] \arrow[d, dash] & L_2 \arrow[d, dash] \\
L_3 \arrow[r, dash] & L_4
\end{tikzcd}$$

We can combine these into one theory, with the operations of one L_i commuting with those of another if and only if they are adjacent in the square. The resulting theory is the colimit of the diagram:

$$\begin{tikzcd}
L_1 \arrow[r] \arrow[d] & L_1 \otimes L_2 & L_2 \arrow[l] \arrow[d] \\
L_1 \otimes L_3 & & L_2 \otimes L_4 \\
L_3 \arrow[r] \arrow[u] & L_3 \otimes L_4 & L_4 \arrow[l] \arrow[u]
\end{tikzcd}$$

This idea is general, applying to any irreflexive graph of theories; let us call such combinations of theories 'graphical combinations.' One can show that any 'polynomial combination' of theories, built up out of $+$ and $\otimes$, is equivalent to such a graphical combination; the present example is a case in point: the corresponding polynomial combination is $(L_1 + L_4) \otimes (L_2 + L_3)$. One may then ask whether the converse is true: are all graphical combinations polynomial?

To answer such questions, we consider formal polynomials and graphs. Formal polynomials are built out of variables using $+$ and $\otimes$, obtaining, for example, $x \otimes (y + z)$ and $(x \otimes y) + (x \otimes z)$. Given such a polynomial and a list of variables $x_1, \ldots, x_n$ including all those occurring in the polynomial, we obtain a functor $\mathbf{Set}^n \to \mathbf{Set}$ in an evident way. One can then also ask when two such formal polynomials are *equivalent*, meaning that the corresponding functors are naturally isomorphic; this does not depend on the choice of the list of variables, and we will generally suppress mention of the choice.

The two example polynomials just given are not so equivalent; to see this note that the initial theory is an identity (up to isomorphism) for both $+$ and $\otimes$, and so, taking y and z to be the initial theory, if the two were equivalent so would be x and $x + x$. On the other hand, there are some evident equivalences based on the commutativity and associativity natural isomorphisms for $+$ and $\otimes$; let us call these 'simple' equivalences.

Formal graphs are finite undirected irreflexive graphs whose nodes are labelled by variables. They denote functors of theories in a way that will be evident from the above discussion of graphical colimits of theories (we again understand a given list of variables here). Formalising remarks made above, one can associate a formal graph with every formal polynomial in such a way that the two denote naturally isomorphic functors:

– to every variable one associates the graph with one node, labelled by that variable
– to every polynomial of the form $p + q$ one associates the disjoint sum of the graphs associated to p and q
– to every polynomial of the form $p \otimes q$ one associates the graph obtained from the disjoint sum of the graphs associated to p and q by adding edges between every node in the first graph and every node in the second one

Note that the graph so associated to a polynomial, other than a variable, is either disconnected or else has a disconnected complement.

Proposition 5. *Two polynomials are simply equivalent if and only if their associated graphs are isomorphic.*

Proof. (Sketch) Necessity is obvious; sufficiency follows from the fact that a graph and its complement cannot both be disconnected.

Theorem 14. *The functors associated to two formal graphs are naturally isomorphic if and only if the two graphs are isomorphic.*

Proof. We again just give a sketch of the proof. The implication from right to left is clear. In the other direction, let x_i be the variables occurring in the two graphs, for $i = 1, n$. Consider the two theories L and L' obtained from the formal graphs by taking x_i to be L_i where, as an equational theory, L_i is given by two unary function symbols f and g subject to the equation $f^{p_i}(x) = g^{p_i}(x)$, where p_i is the ith prime number. Then, as L and L' are isomorphic, so are the two semigroups $L(1, 1)$ and $L'(1, 1)$. Note that these semigroups are both generated by copies of f and g, there being one copy for each node of the graph in question.

As the equations are length-preserving, the semigroup isomorphism must map generators to generators; furthermore, the prime numbers associated to corresponding generators via the above equations must be the same. Thus the isomorphism determines a bijection between the nodes of the two graphs that respects the variables labelling them. This bijection also respects the graph structure, as that yields commutations in the two theories, and hence the two semigroups and such commutations are also preserved by the semigroup isomorphism.

Note that the proof only makes use of the object part of the two functors. So we also have that two formal graphs are isomorphic if and only if the object parts of their associated functors are isomorphic.

We now in a position to answer the two questions formulated above. First, not all graphical combinations are equivalent to polynomial ones: for a counterexample one may take take any nontrivial graph such that neither it nor its complement is disconnected; the simplest example is the four-node graph:

$$\begin{tikzcd} x_1 \arrow[r, dash] & x_2 \arrow[r, dash] & x_3 \arrow[r, dash] & x_4 \end{tikzcd}$$

Second, we can characterise polynomial equivalence:

Corollary 8. *Two polynomials are equivalent if and only if they are simply equivalent.*

The graphical method is convenient for calculating combinations of theories, and we now present two examples concerning state and exceptions. First, suppose we wish to combine the usual exceptions theory L_E with the state theory L_S and a theory L_{E_u} for non-recoverable errors. One would wish only the last two to commute, giving the graph:

$$\begin{tikzcd} L_E & L_S \arrow[r, dash] & L_{E_u} \end{tikzcd}$$

Following the above translation of polynomials into graphs ‘backward,’ one notes that this is a disjoint combination of the subgraph with the theory L_E and the subgraph with the theories L_S and L_{E_u}, which latter are connected. The corresponding polynomial combination of theories is therefore:

$$L_E + (L_S \otimes L_{E_u})$$

Passing to monads, and making use of monad transformers, the corresponding monad can be written as:

$$F_E(F_S(T_{E_u}))$$

Using previous results, one can then calculate an explicit form for this monad:

$$((S \times (- + E)) + E_u)^S$$

Suppose now we wish instead to have a theory L_E for standard exceptions, a theory L_{E_r} for exceptions for rollback, as discussed in Section 1, and two theories for state: L_S for ordinary state, not subject to rollback, and L_{S_r} for state subject to rollback. One would then naturally have the two state theories commute and also the rollback state and exception theories commute. This gives the graph:

$$\begin{tikzcd}
L_E & L_{E_r} \arrow[d, no head] \\
L_S \arrow[r, no head] & L_{S_r}
\end{tikzcd}$$

which yields the polynomial combination:

$$L_E + (L_{S_r} \otimes (L_{E_r} + L_S))$$

The corresponding monad is $F_E(F_{S_r}(F_{E_r}(T_S)))$, with the explicit form:

$$(S \times ((S_r \times (- + E)) + E_r))^{S \times S_r}$$

The reader may enjoy the exercise of adding in unrecoverable errors.

The advantage of the graphical method is that one has only to consider the pairwise relationships; everything else then follows, including which monad transformers are to be applied, if any. We hasten to add that this last example is hardly realistic; for example, for database languages one needs to model databases, which would involve a less naive form of state, and one further needs to bring in additional structure to enable the modelling of parallelism.

All the binary combinations of effects we have considered in this paper have already appeared in the literature. In principle there could be other computationally interesting combinations, even just using the sum and the commutative combination of theories. However, as far as we can tell, there is not much of interest is to be found in this way, although it is certainly good to try. For example, let us consider the four theories, L_E, L_S, $L_{I/O}$ and L_N, working in **Set**. Taking symmetry into account, there are twenty such combinations of these, of which so far we have only considered seven, namely the six given in the above formula and the transactional combination $L_E \otimes L_S$. However, apart from a known generalisation of the interactive I/O theory, and a possible non-interactive form of input/output, none of the other thirteen possibilities seems to yield anything new of computational interest.

Let us look first at the situation 'along the diagonal,' beginning with the sum of theories. For exceptions we have that $L_E + L_{E'}$ is isomorphic to $L_{E+E'}$, so we obtain nothing new there. For state, we have not studied the sum $L_S + L_{S'}$, but neither are we aware of any natural computational interpretation of this theory: what could it mean to have two disjoint sets of states where assignment (or update) in one did not commute with assignment (or update) in the other? We also do not know an explicit form for the induced monad.

For interactive I/O we obtain a known generalisation of what we have been considering. Take L_I, the 'input' theory, to be the absolutely free theory of an operator $\mathit{read}: I \to 1$ and take L_O, the 'output' theory, to be the absolutely free theory of an operator $\mathit{write}: 1 \to O$; evidently $L_{I/O}$ is the same as $L_I + L_O$. Then a general form of theory, closed under sums, is given by:

$$\sum_{i=1,m} L_{I_i} + \sum_{j=1,n} L_{O_j}$$

This is the absolutely free theory on operators $\mathit{read}: I_i \to 1$ and $\mathit{write}_j: 1 \to O$, and it can be used to model m input channels and n output channels. The induced monad is given explicitly by:

$$T(X) = \mu Y. \, (\sum_{i=1,m} Y^{I_i} + \sum_{j=1,n} (O_j \times Y) + X)$$

Everything we said above concerning the simpler case of $L_{I/O}$ and its combinations generalises naturally to this more general theory. However, nothing essentially new is thereby learnt, although, certainly, the applications are wider; it was therefore convenient for us to phrase our discussions above in terms of the special case.

Finally, turning to nondeterminism, while we have not previously considered the sum $L_N + L_N$, neither do we know a computational interpretation; the closest we are aware of is the combination of internal and external nondeterminism, as considered in, e.g., [17]; but there one naturally imposes additional distributivity equations, as mentioned above.

Let us now consider the commutative combination of theories along the diagonal. For exceptions we have that $L_E \otimes L_{E'}$ is the theory of a single constant with no equations, other than in the trivial case where one of E or E' is empty. For state we have that $L_S \otimes L_{S'}$ is $L_{S \times S'}$ which again yields nothing new. For I/O while we have not previously considered $L_{I/O} \otimes L_{I'/O'}$ neither can we think of a natural computational interpretation of it, nor of an explicit form for the induced monad. There could be some interest in combinations like $L_I \otimes L_O$ in which input and output commute; they may be of use in modelling some kind of non-interactive I/O or stream-based computation. One can at any rate obtain an explicit form for the induced monad: the monad induced by L_O is $O^* \times -$ where O^* is the free monoid on O; so, by Theorem 12, that induced by $L_I \otimes L_O$ is $T_I(O^* \times -)$. Finally, one can show that $L_N \otimes L_N$ is L_N, obtaining nothing new.

Of the remaining five 'off-diagonal' combinations, all are new but none seems to have any computational interest, except perhaps for $L_{I/O} \otimes L_N$ in which nondeterminism commutes with interactive I/O. This suggests some sort of trace model of communicating processes, analogous to the model of [18] for parallel imperative programs. However, if one tried, for example, to model concurrency in the style of Milner [36] one would fail as his parallel operator is sensitive to the order of communication and choice. It seems reasonable to judge this last case as unclear.

As we have seen, in every case of extending effects considered in this paper we obtained a morphism of theories (equivalently of monads); it is these morphisms which yield the natural transformations associated to the monad transformers. Further, according to Proposition 4, operation transformers are equivalent to such morphisms. However, taking the case of **Set** for simplicity, it follows that the equations holding for an effect are included in those holding for the extension (modulo the theory morphism). But this is odd as, if anything, one would expect a decrease in the equations holding as there is an increase in the available contexts for discriminating computations: the extension will have more operations available for constructing such contexts. It would be interesting to have an independent justification of the conservation of equations; perhaps this could be accomplished through a theory of the observation of effects.

Acknowledgements

We should like to acknowledge useful conversations with Paul Levy and helpful comments from the referees.

References

1. S. Abramsky, Experiments, Powerdomains and Fully Abstract Models for Applicative Multiprogramming, *Proc. FCT* (ed. M. Karpinski), LNCS Vol. 158, pp. 1–13, 1983.
2. S. Abramsky, A Domain Equation for Bisimulation, *Inf. and Comput.*, Vol. 92, No. 2, pp. 161–218, 1991.
3. J. Adámek and J. Rosický, *Locally Presentable and Accessible Categories*, London Mathematical Society Lecture Note Series, Vol. 189, Cambridge University Press, 1994.
4. M. S. Ager, O. Danvy and J. Midtgaard, *A Functional Correspondence between Monadic Evaluators and Abstract Machines for Languages with Computational Effects*, BRICS Technical Report, RS-03-35, Department of Computer Science, Aarhus University, 2003, available at http://www.brics.dk/RS/03/Abs/BRICS-RS-03-Abs/BRICS-RS-03-Abs.html.

5. M. Barr and C. Wells, *Toposes, Triples and Theories*, Springer-Verlag, 1985.
6. M. Barr and C. Wells, *Category Theory for Computing Science*, Prentice-Hall, 1990.
7. N. Benton, J. Hughes, and E. Moggi, *Monads and Effects*, in *APPSEM '00 Summer School, 2000* (eds. G. Barthe, P. Dybjer, L. Pinto and J. Saraiva), LNCS Vol. 2395, pp. 42–122, Berlin: Springer Verlag, 2000.
8. R. Bird, *Introduction to Functional Programming Using Haskell*, Prentice Hall, 1998.
9. P. Cenciarelli and E. Moggi, A Syntactic Approach to Modularity in Denotational Semantics, *Proc. 5th. Biennial Meeting on Category Theory and Computer Science*, CWI Technical report, 1993.
10. M. Escardó and A. Simpson, A Universal Characterization of the Closed Euclidean Interval (Extended Abstract), *Proc. LICS '01*, pp. 115–125, IEEE Press, 2001.
11. A. Filinski, Representing Layered Monads, Proc. 26th. POPL, pp. 175–188, ACM Press, 1999.
12. M. P. Fiore, A. Jung, E. Moggi, P. O'Hearn, J. Riecke, G. Rosolini and I. Stark, Domains and Denotational Semantics: History, Accomplishments and Open Problems, in *Bulletin of EATCS*, No. 59, pp. 227–256, 1996.
13. P. J. Freyd, Algebra-Valued Functors in General and Tensor Products in Particular, *Colloq. Math. Wroclaw*, Vol. 14, pp. 89–106, 1966.
14. N. Gambino and M. Hyland, Wellfounded Trees and Dependent Polynomial Functors, in *Proc. TYPES 2003* (eds. S. Berardi, M. Coppo and F. Damiani), LNCS Vol. 3085, pp. 210–225, Berlin: Springer Verlag, 2004.
15. G. Gierz, K. H. Hofmann, K. Keimel, J. D. Lawson, M. Mislove and D. S. Scott, *Continuous Lattices and Domains*, Encyclopedia of Mathematics, Vol. 93, Cambridge: Cambridge University Press, 2003.
16. R. Heckmann, Probabilistic Domains, *Proc. CAAP '94*, LNCS, Vol. 136, pp. 21-56, Springer-Verlag, 1994.
17. M. C. B. Hennessy, *Algebraic Theory of Processes*, MIT Press, 1988.
18. M. C. B. Hennessy and G. D. Plotkin, Full Abstraction for a Simple Parallel Programming Language, *Proc. MFCS '79* (ed. J. Bečvář), LNCS, Vol. 74, pp. 108-120, Springer-Verlag, 1979.
19. J. M. E. Hyland and A. J. Power, Pseudo-Commutative Monads, *Proc. MFPS XVII*, ENTCS Vol. 45, Elsevier, 2001.
20. J. M. E. Hyland and A. J. Power, Two-Dimensional Linear Algebra, *Proc. CMCS 2001*, ENTCS Vol. 47, Elsevier, 2001.
21. J. M. E. Hyland and A. J. Power, Pseudo-Commutative Monads and Pseudo-Closed 2-Categories, *J. Pure Appl. Algebra*, Vol. 175, Nos. 1–3, pp. 141–185, 2002.
22. C. Jones, *Probabilistic Non-Determinism*, Ph.D. Thesis, University of Edinburgh, Report ECS-LFCS-90-105, 1990.
23. M. Jones and L. Duponcheel, *Composing Monads*, Technical Report YALEU/DCS/RR-1004, Yale University, Dept. Comp. Sci., 1993.
24. C. Jones and G. D. Plotkin, A Probabilistic Powerdomain of Evaluations, *Proc. LICS '89*, pp. 186–195, IEEE Press, 1989.
25. G. M. Kelly, A Unified Treatment of Transfinite Constructions for Free Algebras, Free Monoids, Colimits, Associated Sheaves, and so on, *Bull. Austral. Math. Soc.*, Vol. 22, pp. 1–83, 1980.
26. G. M. Kelly, *Basic Concepts of Enriched Category Theory*, Cambridge University Press, 1982.
27. G. M. Kelly, Structures Defined by Finite Limits in the Enriched Context I, *Cahiers de Topologie et Géométrie Différentielle*, Vol. 23, No. 1, pp. 3–42, 1982.

28. G. M. Kelly and A. J. Power, Adjunctions whose Counits are Coequalizers, and Presentations of Finitary Enriched Monads, *J. Pure Appl. Algebra*, Vol. 89, pp. 163–179, 1993.
29. D. J. King and P. Wadler, Combining Monads, *Proc. 1992 Glasgow Workshop on Functional Programming* (eds. J. Launchbury and P. M. Samson), pp. 134–143, Workshops in Computing, Berlin: Springer-Verlag, 1992.
30. Y. Kinoshita, A. J. Power, and M. Takeyama, Sketches, *J. Pure Appl. Algebra*, Vol. 143, pp. 275–291, 1999.
31. A. Kock, Monads on Symmetric Monoidal Closed Categories, *Arch. Math.*, Vol. 21, pp. 1–10, 1970.
32. H. König, Theory and Applications of Superconvex Spaces, *Aspects of Positivity in Functional Analysis*, pp. 79–118, North-Holland Math. Stud., Vol. 122, Amsterdam: North-Holland, 1986.
33. F. W. Lawvere, Functorial Semantics of Algebraic Theories and Some Algebraic Problems in the context of Functorial Semantics of Algebraic Theories, Ph.D. thesis, Columbia University, 1963, and in *Reports of the Midwest Category Seminar II*, 41–61, 1968, and reprinted in *Theory and Applications of Categories*, No. 5, pp. 1–121, 2004.
34. E. G. Manes, *Algebraic Theories*, Graduate Texts in Mathematics, Vol. 26, Springer-Verlag, 1976.
35. G. Milne and R. Milner, Concurrent Processes and Their Syntax, *JACM*, Vol. 26, No. 2, pp. 302–321, 1979.
36. R. Milner, *Communication and Concurrency*, New York: Prentice Hall, 1989.
37. M. W. Mislove, Nondeterminism and Probabilistic Choice: Obeying the Laws, *Proc. CONCUR 2000* (ed. C. Palamidessi), LNCS, Vol. 1877, pp. 350–364, Springer-Verlag, 2000.
38. I. Moerdijk and E. Palmgren, Wellfounded Trees in Categories, in *Ann. Pure Appl. Logic*, Vol. 104, Nos. 1–3, pp. 189–218, 2000.
39. E. Moggi, Computational Lambda-Calculus and Monads, *Proc. LICS '89*, pp. 14–23, IEEE Press, 1989.
40. E. Moggi, *An Abstract View of Programming Languages*, University of Edinburgh, Report ECS-LFCS-90-113, 1989.
41. E. Moggi, Notions of Computation and Monads, *Inf. and Comp.*, Vol. 93, No. 1, pp. 55–92, 1991.
42. G. D. Plotkin, A Powerdomain Construction, *SIAM J. Comput.*, Vol. 5, No. 3, pp. 452–487, 1976.
43. G. D. Plotkin, *Domains*, 1983, available at http://homepages.inf.ed.ac.uk/gdp/publications.
44. G. D. Plotkin and A. J. Power, Adequacy for Algebraic Effects, *Proc. FOSSACS 2001* (eds. F. Honsell and M. Miculan), LNCS, Vol. 2030, pp. 1–24, Springer-Verlag, 2001.
45. G. D. Plotkin and A. J. Power, Semantics for Algebraic Operations (extended abstract), *Proc. MFPS XVII* (eds. S. Brookes and M. Mislove), ENTCS, Vol. 45, Elsevier, 2001.
46. G. D. Plotkin and A. J. Power, Algebraic Operations and Generic Effects, *Applied Categorical Structures*, Vol. 11, No. 1, pp. 69–94, 2003.
47. G. D. Plotkin and A. J. Power, Notions of Computation Determine Monads, *Proc. FOSSACS '02*, (eds. M. Nielsen and U. Engberg), LNCS, Vol. 2303, pp. 342–356, Springer-Verlag, 2002.
48. A. J. Power, Why Tricategories? *Inf. and Comp.*, Vol. 120, No. 2, pp. 251–262, 1995.

49. A. J. Power, Enriched Lawvere Theories, *Theory and Applications of Categories*, Vol. 6, pp. 83–93, 2000.
50. A. J. Power, Modularity in Denotational Semantics, *Proc. MFPS XIII*, ENTCS, Vol. 6 (eds. S. Brookes and M. Mislove), Elsevier, 1997.
51. A. J. Power and E. P. Robinson, Modularity and Dyads, *Proc. MFPS XV* (eds. S. Brookes, A. Jung, M. Mislove and A. Scedrov), ENTCS Vol. 20, Elsevier, 1999.
52. A. J. Power and G. Rosolini, A Modular Approach to Denotational Semantics, *Proc. ICALP 98*, LNCS 1443, pp. 351–362, 1998.
53. E. Robinson, Variations on Algebra: Monadicity and Generalisations of Equational Theories, in *A Festschrift for Professor Rod Burstall*, TACS, Vol. 13, Nos. 3,4 & 5, pp. 308–326, 2002.
54. H. Schubert, *Categories*, Springer-Verlag, 1972.
55. A. Simpson and G. D. Plotkin, Complete Axioms for Categorical Fixed-Point Operators, *Proc. 15th. Symp. on Logic in Computer Science*, pp. 30–41, IEEE Computer Society Press, 2000.
56. R. Street, The Formal Theory of Monads, *J. Pure Appl. Algebra*, Vol. 2, pp. 149–168, 1972.
57. R. Tix, *Continuous D-cones: Convexity and Powerdomain Constructions*, Ph.D. thesis, Technische Universitat Darmstadt, Aachen: Shaker Verlag, Aachen, 1999.
58. R. Tix, K. Keimel and G. Plotkin, *Semantic Domains for Combining Probability and Non-Determinism*, Electronic Notes in Theoretical Computer Science, Vol. 129, pp. 1–104, Amsterdam: Elsevier, 2005.

A Pseudo-commutativity and pseudo-closedness

The simplest way we know to explain the extent to which we have a natural closed structure on the category of small categories with countable products is in terms of 2-monads on **Cat** as developed in [19,21], cf also [20]. The 2-monad of interest to us is the 2-monad T_{cp} for which the 2-category of algebras, pseudo-maps, and 2-cells is the 2-category of small categories with countable products, functors that preserve countable products in the usual sense, and natural transformations.

Definition 8. *A symmetric pseudo-commutativity for a* 2*-monad* T *on* **Cat** *consists of a family of invertible natural transformations:*

$$\begin{tikzcd}
TA \times TB \arrow[r, "t^*"] \arrow[d, "t"'] & T(A \times TB) \arrow[r, "Tt"] \arrow[d, phantom, "\Downarrow \gamma_{A,B}"] & T^2(A \times B) \arrow[d, "\mu_{A\times B}"] \\
T(TA \times B) \arrow[r, "Tt^*"'] & T^2(A \times B) \arrow[r, "\mu_{A\times B}"'] & T(A \times B)
\end{tikzcd}$$

natural in A *and* B *and subject to coherence with respect to the symmetry of* **Cat** *and one coherence axiom with respect to each of the strength, unit, and multiplication of* T.

The monad T_{cp} has a unique symmetric pseudo-commutativity. The first main definition of [21] gives a notion of *pseudo-closed* structure for a 2-category: it is almost as strong as closed structure, but one needs to relax the definition of closed structure just a little in order to account for the distinctions between preservation and strict preservation of structure such as countable product structure: the reason, in our setting, that we do not quite have a closed structure is that, given a category C with countable products, the category $\mathrm{Mod}(\aleph_1^{\mathrm{op}}, C)$ is equivalent but not isomorphic to C. We do not spell out the detailed definition of pseudo-closed 2-category here. The main result of [21] (see [19] for a formulation directed more towards a computer science audience) is as follows:

Theorem 15. *If T is a symmetric pseudo-commutative accessible 2-monad on* **Cat***, the 2-category of T-algebras and pseudo-maps of T-algebras has a pseudo-monoidal pseudo-closed structure induced by the pseudo-commutative structure of T, coherently with respect to the closed structure of* **Cat***.*

Corollary 9. *The 2-category of small categories with countable products, countable product preserving functors, and natural transformations is pseudo-monoidal pseudo-closed, coherently with respect to the closed structure of* **Cat***.*

The heart of this result as it applies to us is that the construction that sends a pair of small categories C and D with countable products to the category $CP(C, D)$ of countable product preserving functors from C to D is a well behaved construction. Moreover, for any small categories C and C' with countable products, there is a small category $C \otimes C'$ with countable products together with a well behaved equivalence of categories between $CP(C, CP(C, D))$ and $CP(C \otimes C', D)$ natural in D. The theorem only determines the construction $C \otimes C'$ up to coherent equivalence of categories, but, when restricted to countable Lawvere theories, it agrees up to equivalence with the construction we gave in the paper. Thus we may conclude the following:

Theorem 16. *The construction $L \otimes L'$ on countable Lawvere theories extends to a coherent pseudo-monoidal pseudo-closed structure on the 2-category of small categories with countable products, and, for any small category C with countable products, there is a coherent equivalence of categories between $\mathrm{Mod}(L \otimes L', C)$ and $\mathrm{Mod}(L, \mathrm{Mod}(L', C))$.*

Upon inspection of the proof one can see that the smallness assumption on the category C is not needed.