Preservation of Equations by Monoidal Monads

Louis Parlant
University College London, UK
l.parlant@cs.ucl.ac.uk

Jurriaan Rot
Radboud University, Nijmegen, The Netherlands
jrot@cs.ru.nl

Alexandra Silva
University College London, UK
alexandra.silva@ucl.ac.uk

Bas Westerbaan
University College London, UK
bas@westerbaan.name

Abstract

If a monad T is monoidal, then operations on a set X can be lifted canonically to operations on TX. In this paper we study structural properties under which T preserves equations between those operations. It has already been shown that any monoidal monad preserves linear equations; *affine* monads preserve *drop* equations (where some variable appears only on one side, such as $x \cdot y = y$) and *relevant* monads preserve *dup* equations (where some variable is duplicated, such as $x \cdot x = x$). We start the paper by showing a converse: if the monad at hand preserves a drop equation, then it must be affine. From this, we show that the problem whether a given (drop) equation is preserved is undecidable. A converse for relevance turns out to be more subtle: preservation of certain dup equations implies a weaker notion which we call n-relevance. Finally, we identify a subclass of equations such that their preservation is equivalent to relevance.

2012 ACM Subject Classification Theory of computation → Categorical semantics

Keywords and phrases monoidal monads, algebraic theories, preservation of equations

Digital Object Identifier 10.4230/LIPIcs.MFCS.2020.77

Related Version A full version, which includes proofs, is available at `https://arxiv.org/abs/2001.06348`.

Funding This work was partially supported by the ERC Starting Grant ProFoundNet (grant code 679127), the Marie Curie Fellowship CARBS (grant code 795119), and the EPSRC Standard Grant CLeVer (EP/S028641/1).

Acknowledgements We are grateful to Dan Marsden, Robin Piedeleu, Edmund Robinson and Maaike Zwart for inspiring discussions and suggestions.

1 Introduction

Monads are fundamental structures in programming language semantics as they encapsulate many common side-effects such as non-determinism, exceptions, or randomisation. Their structure has been studied not only from an operational point of view but also from a categorical and algebraic perspective. One question that has attracted much attention is that of monad composition: given two monads T and S, is the composition TS again a monad? The answer to the question in full generality is subtle and examples have shown that even in simple cases the correct answer might be surprisingly tricky to find and prove. This subtlety is illustrated, for instance, by Klin and Salamanca's result that the powerset monad does not compose with itself [9], invalidating repeated claims to the contrary in the literature.

Monad composition can be viewed in different ways. On the one hand, a sufficient condition is given by the existence of the categorical notion of a distributive law between the monads. On the other hand, if one takes into account the algebraic structure of the inner monad, which can be presented in a traditional operations plus equations fashion, one can turn the original question into a preservation question: does the outer monad preserve all operations and equations of the inner algebra? More generally, starting from a set A with some additional algebraic structure, the question arises naturally: is this structure preserved by the application of a monad? For example, consider a set A that has the structure of a group with binary operation $\cdot$ and unit 1. As we apply the *powerset monad* $\mathcal{P}$ the set $\mathcal{P}A$ of subsets of A, is $\mathcal{P}A$ is again a group? Can $\cdot$ be interpreted as a binary operation on elements of $\mathcal{P}A$? Which subset do we identify with the constant 1? In a nutshell, does our monad preserve algebraic features, i.e., the operations and equations defining the structure of A?

This question has already been studied in the late 50s, albeit not in categorical terms. In [4], Gautam introduces the notion of *complex algebra* – the transformation of a Σ-algebra with carrier A into a Σ-algebra on $\mathcal{P}A$, where Σ is an arbitrary signature. Gautam gives a range of positive and negative results for equation preservation. In particular, he shows that commutativity of a binary operation ($x \cdot y = y \cdot x$) and unitality ($x \cdot 1 = x$) are unconditionally preserved by $\mathcal{P}$. These are examples of *linear* equations, in which each variable appears exactly once on each side. Negative results are given for non-linear equations. First, if variables appear more than once (we call these *dup* equations; e.g., $x \cdot x = x$), this is not preserved by $\mathcal{P}$. Second, if a variable appears on one side of the equation only (we call these *drop* equations, for instance $x \cdot 0 = 0$), then the powerset does not preserve it either.

In this paper, we examine the question of equation preservation at the general level of a *monoidal monad* T. Monoidal monads give a canonical lifting to Σ-algebras, of which Gautam's complex algebra construction for $\mathcal{P}$ is a special case. We provide a comprehensive characterisation of monad classes and equations that are preserved inside a certain class.

Part of this question has been studied in the literature: in [14], Manes and Mulry show that for a monad to preserve linear equations, it suffices to have a *symmetric monoidal* structure. This argument is further developed in [3], where the authors give sufficient conditions on T for preserving the types of equations outlined by Gautam. In particular, it was shown that so-called *relevant* monads preserve *dup* equations, and *affine* monads preserve *drop* equations. It remained open whether relevance or affineness were necessary conditions for preservation. We now settle this question and provide an extensive characterisation of equations preserved by classes of monoidal monads.

We start with preliminaries on monoidal categories and monads (Section 2), and continue with a technical section recalling how monoidal monads offer a canonical lifting of all algebraic signatures (Section 3). We then present the main contributions of the present paper:

1. We prove a monoidal monad preserves strict-drop equations if and only if it is affine (Section 4).
2. We characterise a large class of *dup* equations, for which preservation is equivalent to the monad being relevant. We then prove that for a restricted class of *dup* equations preservation requires a weaker version of relevance, which we call n-relevance (Section 5).
3. Orthogonally, we prove a more algorithmic result: given a monad and an equation, we show that the general problem of preservation is undecidable (Section 4).

A summary of the classes of equations used to derive the preservation results and concrete examples of monads and the class in which they fall appear in Figure 1a.

We remark that the provided necessary conditions on preservation of equations can be used in contrapositive form to show that certain monads do not preserve certain equations.

For instance, from Theorem 12, the main result of Section 4, we know that if a monad is not affine, it does not preserve any drop equations in general. This generalises Gautam's result for $\mathcal{P}$ and provides a range of other examples that do not preserve drop equations: the maybe monad $X + 1$, the monad $M \times X$ for M a non-trivial monoid, and the multiset monad $\mathbf{M}_{\mathbb{S}}$, for $\mathbb{S}$ a non-trivial semiring. Similarly, the results in Section 5 allow us to show that certain (in fact, many) monads do not preserve dup equations, just by showing that they are not relevant.

2 Preliminaries

We recall basic notions related to monoidal categories, monads and algebras for a functor.

Cartesian monoidal categories. A *monoidal category* consists of a category $\mathbf{C}$ equipped with: a bifunctor $\otimes\colon \mathbf{C} \times \mathbf{C} \to \mathbf{C}$, an object I called *unit* or *identity*, a natural isomorphism $\alpha_{X,Y,Z}\colon (X \otimes Y) \otimes Z \to X \otimes (Y \otimes Z)$ called *associator*, a natural isomorphism $\rho_X\colon X \otimes I \to X$ called *right unitor*, and a natural isomorphism $\rho'_X\colon I \otimes X \to X$ called *left unitor*, where α, ρ and ρ' are subject to coherence conditions [13]. A *Cartesian monoidal category* is a monoidal category whose monoidal structure is given by product (denoted by $\times$) and a terminal object (denoted by 1). A category with finite products forms a Cartesian monoidal category, by making a choice of products for each pair of objects. Any Cartesian monoidal category $\mathbf{C}$ is symmetric monoidal, witnessed by a natural isomorphism denoted by $\mathsf{swap}_{X,Y}\colon X \times Y \to Y \times X$. For a product $\prod_{i\in I} X_i$, we denote the projections by $\pi_j\colon \prod_{i\in I} X_i \to X_j$. Given an object X, we define the *diagonal* Δ_X by pairing: $\Delta_X = \langle id_X, id_X\rangle\colon X \to X \times X$. Further, for a functor $T\colon \mathbf{C} \to \mathbf{C}$ we let $\chi_{X,Y} = \langle T\pi_1, T\pi_2\rangle\colon T(X \times Y) \to TX \times TY$. Throughout this paper, we will mainly focus on the Cartesian monoidal category **Set** of sets and functions.

Monads. A *monad* on a category $\mathbf{C}$ is a triple (T, η, μ) consisting of an endofunctor $T\colon \mathbf{C} \to \mathbf{C}$, a natural transformation $\eta\colon \mathrm{Id} \Rightarrow T$ called *unit* and a natural transformation $\mu\colon TT \Rightarrow T$ called *multiplication*, such that $\mu \circ T\mu = \mu \circ \mu T$ and $\mu \circ \eta T = \mathrm{id} = \mu \circ T\eta$.

▶ **Example 1** (Monads).

1. The *powerset monad* is given by the functor $\mathcal{P}\colon \mathbf{Set} \to \mathbf{Set}$, which maps a set X to its set of subsets, together with unit η mapping x to $\{x\}$, and μ given by union. This monad restricts to its finitary version with the functor $\mathcal{P}_f\colon \mathbf{Set} \to \mathbf{Set}$ mapping a set to its finite subsets, and similarly to the *non-empty powerset* $\mathcal{P}^+$.
2. For a semiring $\mathbb{S}$, let $\mathbf{M}_{\mathbb{S}}$ be the *generalised multiset* monad defined as follows. A multiset $\xi \in \mathbf{M}_{\mathbb{S}}X$ is a map $X \to \mathbb{S}$ with finite support, also written as the formal sum $\sum_x \xi(x)x$. For $f\colon X \to Y$ we define $(\mathbf{M}_{\mathbb{S}}f)(\xi)(y) = \sum_{x;\ f(x)=y} \xi(x)$.The unit is defined by $\eta(x) = 1 \cdot x$ and the multiplication is given by $\mu(\delta)(f) = \sum_x \delta(f)f(x)$, viz. $\mu(\sum_i s_i \sum_j t_{ij}x_{ij}) = \sum_{i,j} s_i t_{ij} x_{ij}$.
3. The *distribution monad* is given by $\mathcal{D}\colon \mathbf{Set} \to \mathbf{Set}$, $\mathcal{D}X = \{\nu\colon X \to [0,1] \mid \sum_{x\in X} \nu(x) = 1, \nu \text{ with finite support}\}$. The action on morphisms, unit, and multiplication are as in $\mathbf{M}_{\mathbb{S}}$.
4. For any commutative monoid M, the functor $X \mapsto M \times X$ is a monad with $\eta(x) = (1, x)$ and $\mu(v, (w, x)) = (vw, x)$.
5. For a set A, $X \to X^A$ forms a monad. The unit is defined by $\eta_X(x) = (a \mapsto x)$, and the multiplication μ_X maps $s \in (X^A)^A$ to $(a \mapsto s(a)(a))$.

Monoidal monads. The notion of *monoidal monad* captures a well-behaved interaction between a monad and the monoidal structure of the underlying category.

Let $\mathbf{C}$ be a Cartesian monoidal category. A *(lax) monoidal functor* is an endofunctor $F\colon \mathbf{C} \to \mathbf{C}$ together with natural transformations $\psi_{X,Y}\colon FX \times FY \to F(X \times Y)$ and $\psi^0\colon 1 \to F1$ satisfying certain laws. A monoidal functor is called *symmetric* if $\psi_{Y,X} \circ \mathsf{swap}_{FX,FY} = F\mathsf{swap}_{X,Y} \circ \psi_{X,Y}$ for all X, Y. A monad (T, η, μ) on $\mathbf{C}$ is called *monoidal* if T is monoidal, the unit η and multiplication μ are monoidal natural transformations, and the associated natural transformation $\psi_{X,Y}\colon TX \otimes TY \to T(X \otimes Y)$ satisfies $\psi^0 = \eta_1$. Since the underlying category $\mathbf{C}$ is symmetric monoidal, monoidal monads are equivalent to commutative monads [12, 10], see also [6]. It follows that any monoidal monad on a symmetric monoidal category is symmetric.

For any $n \geq 2$, we define $\psi^n\colon TX_1 \times \cdots \times TX_n \to T(X_1 \times \cdots \times X_n)$ as the n-ary version of ψ, associatively constructed from the binary version.

▶ **Example 2.** All the monads from Example 1 are monoidal, by defining ψ as follows:

1. $\psi_{X,Y}\colon \mathcal{P}X \times \mathcal{P}Y \to \mathcal{P}(X \times Y)$ is given by Cartesian product: $\psi_{X,Y}(U, V) = U \times V$.
2. $\psi_{X,Y}\colon \mathbf{M}_{\mathbb{S}}X \times \mathbf{M}_{\mathbb{S}}Y \to \mathbf{M}_{\mathbb{S}}(X \times Y)$ is given by $\psi_{X,Y}(\xi_1, \xi_2) = \lambda(x, y).\xi_1(x) \cdot \xi_2(y)$.
3. $\psi_{X,Y}\colon \mathcal{D}X \times \mathcal{D}Y \to \mathcal{D}(X \times Y)$ is given by $\psi_{X,Y}(\nu_1, \nu_2) = \lambda(x, y).\nu_1(x) \cdot \nu_2(y)$.
4. $\psi_{X,Y}\colon (M \times X) \times (M \times Y) \to M \times (X \times Y)$ is given by $\psi_{X,Y}((v, x), (w, y)) = (vw, (x, y))$.
5. $\psi_{X,Y}\colon X^A \times Y^A \to (X \times Y)^A$ is defined pointwise as $\psi_{X,Y}(f, g)(a) = (f(a), g(a))$.

Algebraic Constructs. A *signature* Σ is a set of operation symbols, together with a natural number $|\sigma|$ for each $\sigma \in \Sigma$, called the *arity* of σ. For X a set, the set of Σ-*terms* over X is the least set Σ^*X such that $X \subseteq \Sigma^*X$ and, if $t_1, \ldots, t_{|\sigma|} \in \Sigma^*X$ for some $\sigma \in \Sigma$ then $\sigma(t_1, \ldots t_n) \in \Sigma^*X$. We write $\mathrm{Var}(t)$ for the set of variables appearing in the term t.

An *algebraic theory* $\mathbb{T}$ is a triple (Σ, V, E) where Σ is a signature, V is a set of variables, and $E \subseteq \Sigma^*V \times \Sigma^*V$ is a relation. We refer to elements of E as *equations* or *axioms* of $\mathbb{T}$, and denote an equation $(u, v) \in E$ by $u = v$. When two Σ-terms t_1, t_2 can be proved equal using equational logic and the axioms of $\mathbb{T}$, we write $t_1 = t_2$. More precisely, $t_1 = t_2$ if t_1 and t_2 are related by the least congruence containing E which is also closed under substitution.

For instance, the theory of monoids has a signature containing one constant 1 and a binary symbol $\cdot$ and the axioms of associativity and unit: $x \cdot 1 = x = 1 \cdot x$ and $x \cdot (y \cdot z) = (x \cdot y) \cdot z$.

For a signature Σ , a Σ-*algebra* $\mathcal{A}$ consists of a carrier set A and, for each symbol $\sigma \in \Sigma$ of arity $|\sigma|$, a morphism $\sigma_{\mathcal{A}}\colon A^{|\sigma|} \to A$. Given a Σ-algebra $\mathcal{A}$ and a map $f\colon V \to A$ to its carrier, we inductively define $f^\sharp\colon \Sigma^*V \to A$ by $f^\sharp(x) = f(x)$ and $f^\sharp(\sigma(t_1, \ldots, t_n)) = \sigma_{\mathcal{A}}(f^\sharp(t_1), \ldots, f^\sharp(t_n))$. Given an equation $t_1 = t_2$ with $t_1, t_2 \in \Sigma^*V$ we say $\mathcal{A}$ *satisfies* $t_1 = t_2$, denoted by $\mathcal{A} \models t_1 = t_2$, if $f^\sharp(t_1) = f^\sharp(t_2)$ for every map $f\colon V \to A$. This extends to sets of equations $E \subseteq \Sigma^*V \times \Sigma^*V$ by $\mathcal{A} \models E$ iff $\mathcal{A} \models t_1 = t_2$ for all $(t_1, t_2) \in E$.

Categorically speaking, a signature Σ gives rise to a polynomial functor $\mathsf{H}_\Sigma\colon \mathbf{Set} \to \mathbf{Set}$, defined by $\mathsf{H}_\Sigma X = \coprod_{\sigma \in \Sigma} X^{|\sigma|}$. A Σ-algebra as defined above is then precisely an algebra for H_Σ, i.e., a set A together with a map $\mathsf{H}_\Sigma A \to A$.The category of Σ-algebras and Σ-algebra morphisms is denoted $\mathbf{Alg}(\Sigma)$. In particular, the set $\mathsf{F}_\Sigma X$ of free Σ-terms on X is a Σ-algebra, and F_Σ forms a functor. For a set of equations $E \subseteq \Sigma^*V \times \Sigma^*V$, we denote by $\mathbf{Alg}(\Sigma, E)$ the full subcategory of $\mathbf{Alg}(\Sigma)$ consisting of those algebras satisfying E.

3 Preserving Operations and Equations

In this section we explain how to lift operations, which allows us to define preservation of equations. We conclude with a technical reformulation of equations and preservation that will be useful later. In this section, we assume T is a monoidal monad on $\mathbf{Set}$.

Lifting operations. Let Σ be a signature. Given a Σ-algebra $\mathcal{A}$ with carrier A, we define a Σ-algebra $\widehat{T}\mathcal{A}$ on TA: for each operator $\sigma \in \Sigma$, we set $\sigma_{\widehat{T}\mathcal{A}} \equiv \big((TA)^{|\sigma|} \xrightarrow{\psi^{|\sigma|}} TA^{|\sigma|} \xrightarrow{T\sigma_{\mathcal{A}}} TA \big)$. This gives a lifting $\widehat{T}\colon \mathbf{Alg}(\Sigma) \to \mathbf{Alg}(\Sigma)$ of T. In fact, this is a lifting of the *monad* T, as it arises from a canonical distributive law of H_Σ over the monad T from [20] (cf. [18]).

▶ **Example 3.** Consider the powerset monad and an algebra $\mathcal{A}$ with carrier A and a binary operation $\cdot$. The lifted operation is given by $U \cdot_{\widehat{\mathcal{P}}\mathcal{A}} V \equiv \{u \cdot v;\ u \in U, v \in V\}$. Liftings for other monads are easily obtained from Example 2.

Preserving equations. The lifting of algebraic operations allows interpretation of equations after application of T: if $t_1 = t_2$ holds on a Σ-algebra $\mathcal{A}$, we can interpret t_1 and t_2 as terms of $\widehat{T}\mathcal{A}$ and verify equality. This leads to the central notion of *preservation* of equations.

▶ **Definition 4.** *Let Σ be a signature, V a set of variables, and $t_1, t_2 \in \Sigma^* V$. We say that T* preserves the equation $t_1 = t_2$ *if for every Σ-algebra $\mathcal{A}$ we have $\widehat{T}\mathcal{A} \models t_1 = t_2$ provided $\mathcal{A} \models t_1 = t_2$. A set of equations is preserved if each one of them is.*

Equivalently, T preserves E if $\widehat{T}$ restricts to $\mathbf{Alg}(\Sigma, E)$. In particular, if S is a monad such that $\mathbf{Alg}(\Sigma, E) \cong \mathcal{EM}(S)$, then T preserving E implies that TS is again a monad.

▶ **Example 5.** Does the powerset monad preserve the equation of commutativity? Let us consider an algebra $\mathcal{A}$ with carrier A featuring a commutative operation $+$. Recalling the lifting defined in example 3, we can verify that for $U, V \in \mathcal{P}A$ we have: $U +_{\widehat{\mathcal{P}}\mathcal{A}} V = \{u + v \mid u \in U, v \in V\} = \{v + u \mid v \in V, u \in U\} = V +_{\widehat{\mathcal{P}}\mathcal{A}} U$. Therefore $\mathcal{P}$ preserves commutativity.

▶ **Example 6.** Consider now an algebra $\mathcal{A}$ with carrier A and an idempotent operation $\cdot$ and the monad $\mathcal{D}$ of probability distributions. Let $a, b \in A$, such that $a \cdot b \neq a$, $a \cdot b \neq b$ and $a \cdot b \neq b \cdot a$. Let $\nu \in \mathcal{D}\mathcal{A}$ be the distribution on A such that $\nu(a) = \nu(b) = 0.5$. Note that, for all $u, v \in A$, we have $\nu(u) \cdot \nu(v) \neq 0$ iff $u, v \in \{a, b\}$. Now we compute: $(\nu \cdot_{\widehat{\mathcal{D}}\mathcal{A}} \nu)(a \cdot b) = \sum\{\nu(u) \cdot \nu(v) \mid u, v \in A \text{ s.t. } u \cdot v = a \cdot b\} = \nu(a) \cdot \nu(b) = 0.25 \neq 0 = \nu(a \cdot b)$. Thus we have $\nu \neq \nu \cdot_{\widehat{\mathcal{D}}\mathcal{A}} \nu$, which means that idempotence is not preserved by $\mathcal{D}$.

In subsequent sections we will treat several classes of equations, which are preserved by different types of monads. Here, we first recall a fundamental result about preservation: any monoidal monad preserves linear equations.

▶ **Definition 7.** *An equation $t_1 = t_2$ is called* linear *when $\mathrm{Var}(t_1) = \mathrm{Var}(t_2)$, and every variable occurs exactly once in t_1 and once in t_2.*

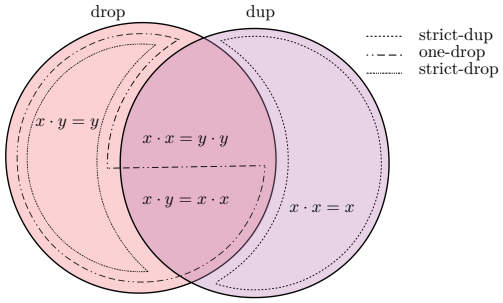
For instance, the laws of associativity, commutativity and unit are linear. Note that if an equation is *not* linear, then either there is a variable which occurs on one side but not the other (which we will refer to as a *drop* equation, Definition 8) or there is a variable that occurs twice (referred to as a *dup* equation, Definition 16). Manes and Mulry showed that any monoidal monad $T\colon \mathbf{Set} \to \mathbf{Set}$ preserves linear equations [14]. This generalises Gautam's result that the powerset monad preserves linear equations [4].

4 Affine Monads and Drop Equations

In this section, we study preservation of *drop* equations, which are non-linear equations where at least one variable occurs on one side but not the other. Preservation of such equations by a monoidal monad T will be related to a property of monoidal monads called *affineness*.

Monad	Affine	Relevant
$\mathcal{P}$	×	×
$\mathcal{P}^+$	✓	×
$\mathcal{D}$	✓	×
$X+1$	×	✓
X^A	✓	✓
$M \times X$	✓ iff M trivial	✓ iff M idempotent
$\mathbf{M}_{\mathbb{S}}$	✓ iff $\mathbb{S}$ trivial	✓ iff $\mathbb{S}$ trivial

(a) Affineness and relevance of well-known monads.



(b) Classes of equations.

Figure 1

▶ **Definition 8** (Drop equations). *An equation $t_1 = t_2$ is called*

1. drop *when at least one variable appears in t_1 but not in t_2 (or conversely);*
2. one-drop *when a variable appears once in t_1 and does not appear in t_2 (or conversely);*
3. strict-drop *when it is not linear and each variable of V appears at most once in t_1 and t_2.*

The set of strict-drop equations is included in the set of one-drop equations, and the latter in the set of drop equations. Both inclusions are strict. Strict-drop equations can equivalently be characterised as equations $t_1 = t_2$ where neither t_1 nor t_2 contains duplicate variables, and at least one variable occurs on one side but not the other. The various classes of drop equations are pictured in Figure 1b, which also contains dup equations (Definition 16).

For instance, the law of absorption $x \cdot 0 = 0$ is a strict-drop equation. The equation $x \cdot (y \cdot y) = y \cdot y$ shows one occurrence of x on the left side and none on the right side, therefore it is a one-drop equation. It is not a strict-drop equation. The equation $x \cdot x = y \cdot y$ is drop but not one-drop. Finally, $x \cdot x = x$ is not drop.

The property of T that we focus on now is *affineness*, introduced by Kock [11]; see also [6].

▶ **Definition 9** ([11]). *A monoidal monad T on a Cartesian monoidal category $\mathbf{C}$ is called* affine *if it has one of the following three equivalent properties: $T1$ is final; diagram ♠ commutes; diagram ♡ commutes, for all A and B.*

$$T1 \xrightarrow{!} 1 \xrightarrow{\eta_1} T1 \qquad \spadesuit \qquad TA \times TB \xrightarrow{\psi} T(A \times B) \xrightarrow{\chi} TA \times TB \qquad \heartsuit$$

See Figure 1a for (non-)examples of affine monads. Regarding preservation of equations by affine monads, we recall the following result.

▶ **Theorem 10** ([3]). *Any affine monoidal monad T on* **Set** *preserves strict-drop equations.*

Theorem 10 does not extend to one-drop equations, as follows from a later result: Example 24 gives a one-drop equation whose preservation implies relevance.

From drop preservation to affineness. We proceed to prove the converse of Theorem 10: if a monoidal monad T preserves a strict-drop equation, then it is affine (Theorem 12).

▶ **Lemma 11.** *Let $t_1 = t_2$ be a one-drop equation, and T: **Set** $\to$ **Set** a monoidal monad. If $t_1 = t_2$ holds on $T1$, then T is affine.*

Since any equation $t_1 = t_2$ trivially holds on 1, by Lemma 11 we obtain:

▶ **Theorem 12.** *Let Σ be a signature, and let $t_1 = t_2$ be a one-drop equation with $t_1, t_2 \in \Sigma^* V$ and $\mathrm{Var}(t_1) \cup \mathrm{Var}(t_2) = V$. Let $T\colon \mathbf{Set} \to \mathbf{Set}$ be a monoidal monad. If T preserves $t_1 = t_2$, then T is affine.*

The above theorem is a stronger result than the converse of Theorem 10: preservation of *one-drop* equations suffices for affineness. Equivalently, if a monad is *not* affine, then we know that it does not preserve any drop equations in general. This generalises Gautam's result that $\mathcal{P}$ does not preserve any one-drop equation [4]. Other examples of monads that, by Theorem 12, do not preserve one-drop equations are $M \times X$ for M a non-trivial monoid, and $\mathbf{M}_{\mathbb{S}}$ for $\mathbb{S}$ a non-trivial semiring (see Figure 1a).

▶ Remark 13. Theorem 12 treats preservation of single equations. Another consequence of Lemma 11 is that, if a monoidal monad $T\colon \mathbf{Set} \to \mathbf{Set}$ preserves a non-empty *set* of equations E that includes a one-drop equation, then it is affine. To see this, note that any equation holds on the algebra 1; therefore also on $T1$, hence T is affine by Lemma 11.

Decidability. The previous section establishes the equivalence between affineness of a monad T and preservation of one-drop equations. We now use this result to analyse a more algorithmic question: is it decidable whether a monad T (presented by finitely many operations and equations) preserves a given equation $t_1 = t_2$? Unfortunately the answer is negative, which we prove by showing that the question whether a given monad is affine is undecidable.

▶ **Theorem 14.** *The following problem is undecidable: given a finite signature Σ and a finite set E of equations, is the monad T presented by (Σ, E) affine?*

Proof. We use an encoding of the following decision problem, which is known to be undecidable [1]: given a finite presentation (G, R) of a monoid $\mathcal{M}$, is $\mathcal{M}$ trivial?

Let (G, R) be a finite presentation of a monoid $\mathcal{M}$. Let Σ be the set of unary operations f_g, for $g \in G$, and E the set of equations $f_{g_1}(f_{g_2}(\dots f_{g_n}(x)\dots)) = f_{h_1}(f_{h_2}(\dots f_{h_k}(x)\dots))$, for each $(g_1 \dots g_n, h_1 \dots h_k) \in R$. Note that (Σ, E) corresponds to the theory of $\mathcal{M}$-actions; let T be the monad presented by this theory. Now, one can show that $T1$ is isomorphic to $\mathcal{M}$, and thus that $T1 = 1$ iff $\mathcal{M}$ is trivial. ◀

Using the equivalence between preserving a class of equations and affineness, together with the latter being undecidable, we obtain a general result on equation preservation.

▶ **Corollary 15.** *The following problem is undecidable: given a finite theory (Σ, E) and an equation $t_1 = t_2$, does the monad T presented by (Σ, E) preserve $t_1 = t_2$?*

5 Relevant Monads and Dup Equations

We now relate so-called *dup* equations, featuring duplications of variables, to *relevant* monads.

▶ **Definition 16** (Dup equations). *An equation $t_1 = t_2$ is called*

1. dup *when at least one variable appears more than once in t_1 or in t_2;*
2. 2-dup *when it is dup and each variable appears at most twice in t_1 or t_2;*
3. strict-dup *when it is not linear and each variable of $\mathrm{Var}(t_1) \cup \mathrm{Var}(t_2)$ appears at least once in t_1 and in t_2.*

Equivalently, an equation is strict-dup when it is not drop, and some variable appears at least twice in t_1 or t_2. Every strict-dup equation is a dup equation. See Figure 1b for an overview of dup and drop equations. For example, the law of idempotence $x = x \cdot x$ is a

strict-dup equation. Distributivity of $\cdot$ over $+$, written $x \cdot (y + z) = x \cdot y + x \cdot z$ is strict-dup as well. The equation $x \cdot (y \cdot y) = y \cdot y$ is dup, because y is duplicated, but it is not strict-dup.

Relevant monads are introduced by Kock in [11], and extensively studied by Jacobs in [6].

▶ **Definition 17.** *A monoidal monad $T\colon \mathbf{C} \to \mathbf{C}$ on a Cartesian monoidal category $\mathbf{C}$ is* relevant *if one of the following two equivalent conditions hold: diagram ♠ commutes for all objects A; diagram ♡ commutes for all objects A, B:*

$$\begin{array}{ccc} TA & \xrightarrow{\Delta} & TA \times TA \\ & \searrow^{T\Delta} & \downarrow\psi \\ & & T(A \times A) \end{array} \quad ♠ \qquad \begin{array}{ccc} T(A \times B) & \xrightarrow{\chi} & TA \times TB \\ & \searrow^{\mathrm{id}} & \downarrow\psi \\ & & T(A \times B) \end{array} \quad ♡$$

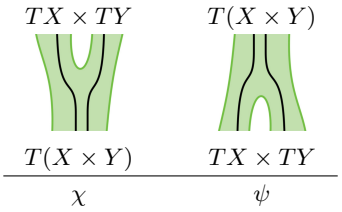
See Figure 1a for examples of relevant monads. We recall the following result.

▶ **Theorem 18** ([3])**.** *Any relevant monad on* **Set** *preserves strict-dup equations.*

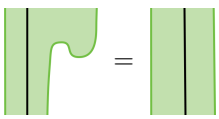
Does this theorem extend to an equivalence? Can a non-relevant monad preserve a dup equation? Again, Gautam provides an answer in the case of the powerset, which is not a relevant monad: $\mathcal{P}$ does not preserve dup equations. We now study the general question for a monoidal monad T. For this purpose, we define a framework of string-like diagrams to easily represent the application of T on categorical objects.

Diagrammatic framework. Our diagrams are inspired from well-known graphical representations for monoidal categories. The concept of *functorial boxes* is introduced by Cockett and Seely in [2] then further examined by Melliès in [17], where functors are represented as boxes surrounding morphisms and objects. In a similar fashion as our framework, monoidal properties allow to gather several objects inside a single sleeve. More details on this representation are given by McCurdy [16], although he chooses to focus on monoidal functors satisfying the Frobenius property, which we do not assume here. Let us first summarise a few central ideas of our diagram calculus.

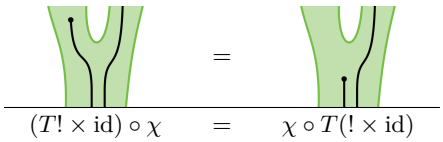
An object X of our category is now represented by a thread (or "wire"), and the application of T on this object results in a "sleeve" covering it. We read a diagram from bottom to top and represent products implicitly as horizontal adjacency. For instance, the morphism $\chi\colon T(X \times Y) \to TX \times TY$ is modelled by a cup-like shape where one sleeve containing two objects splits into two sleeved objects. ψ is modelled in the opposite way and merges two sleeved objects into a single sleeve.



Note that the object 1 is not represented in our diagrams. By the isomorphism $X \times 1 \simeq 1$, we can imagine the presence of 1 as a vertical thread anywhere on the diagram without affecting calculations. Some deformations of the outline of sleeves are allowed: for instance, the equality in the diagram below corresponds to the right unitality of a monoidal functor. Note that the neither the product nor the unitor ρ is explicitly represented in the diagram.

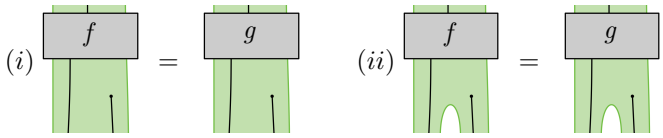


We can "delete" an object by mapping it to the final object 1, represented as an unfinished vertical thread. Naturality of χ and ψ allow to "pull" these threads to the bottom of the diagram, as shown in the equation below.

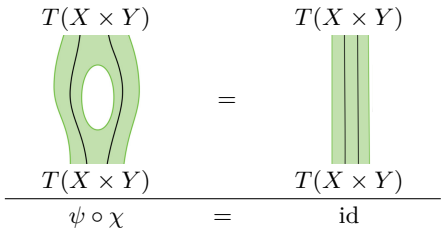


The following result means that unfinished threads may be ignored.

▶ **Lemma 19.** *If we have any of the two equalities* (i) *or* (ii) *below, then* $f = g$.

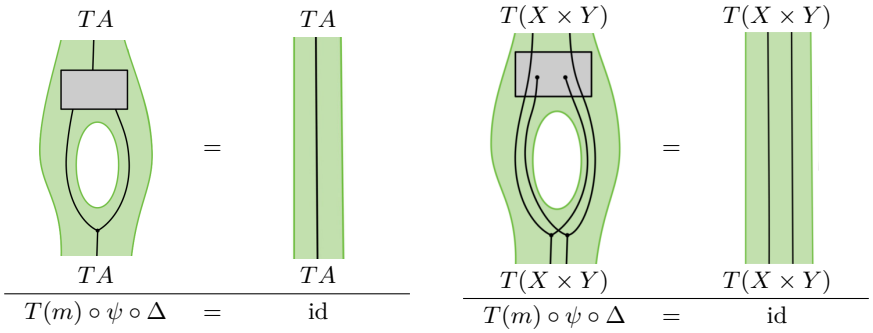


Finally, we focus on the composition $\psi \circ \chi$, which splits a sleeved product, then reunites both components into one sleeve. Recall that relevance means that this composition yields the identity (Definition 17). In [16], this diagrammatic equality is presented as a property of connectivity of functorial regions. We like to describe the graphical aspect of this property as follows: applying χ followed by ψ results in a "bubble" in our sleeve, surrounded by two threads representing arbitrary objects, and relevance allows to pop this bubble, as in (1). In the rest of this section, we develop a method to reduce complex equational problems to this "bubble" property.



(1)

Proving relevance from dup preservation. First, we consider the simplest dup equation: idempotence of a binary operation. Assuming that T preserves it, our strategy is to define an algebra $\mathcal{A}$ and an operation m such that $m(x, x) = x$, and then to derive the relevance of T from the preservation of the idempotence of m. For such an algebra (whose carrier is written A), we draw the following diagrams after this paragraph. The grey box represents our binary idempotent operation m, hence the leftmost diagram represents the term $m(x, x)$ on the lifted algebra $\widehat{T}\mathcal{A}$. By preservation of idempotence, it must be equal to the identity modelled by the right diagram in the leftmost equality. In order to derive the property of relevance from this equality, we conveniently choose A and m. For any two sets X, Y, we define $A \equiv X \times Y$ and $m((a, b), (c, d)) \equiv (a, d)$. Categorically $m = (\pi_1 \times \pi_2)$. It is idempotent: $m((a, b), (a, b)) = (a, b)$. Hence for this algebra, the left equality becomes the right one:



Pulling down the threads corresponding to deleted objects, we obtain diagram (1).

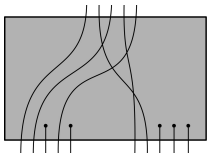
▶ **Theorem 20.** *Let T be a monoidal monad. If T preserves $m(x,x) = x$, then T is relevant.*

Proof. We show here the categorical version of our diagrammatic proof:

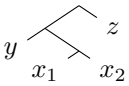
$$\begin{aligned}
\mathrm{id} &= T(m) \circ \psi \circ \Delta && x * x \text{ holds on } \widehat{T}\mathcal{A} \\
&= T(m) \circ \psi \circ \chi \circ T\Delta && \text{see below} \\
&= T(\pi_1 \times \pi_2) \circ \psi \circ \chi \circ T\Delta && \text{definition of } m \\
&= \psi \circ (T\pi_1 \times T\pi_2) \circ \chi \circ T\Delta && \text{naturality} \\
&= \psi \circ \chi \circ T(\pi_1 \times \pi_2) \circ T\Delta && \text{naturality} \\
&= \psi \circ \chi
\end{aligned}$$

In the second step, we used that $\Delta_{TA} = \chi_{A,A} \circ T\Delta_A$, a property that holds trivially for any monoidal monad on **Set** and set A (see for instance [6]). ◀

$$(yx)z = (y(xx))z. \qquad (2)$$



(3)

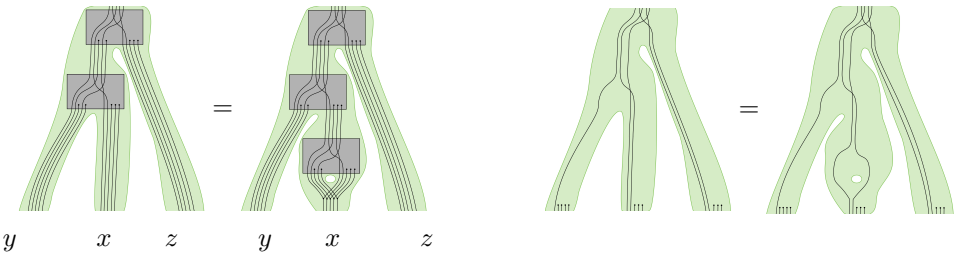


(4)

Our method to show relevance from idempotence preservation can be applied to a more general class of equations. Let us now consider a term t that only contains binary operations and no variable duplication. We focus on equalities of the form $t[x] = t[x \cdot x]$ with $\cdot \in \Sigma$, in other words where both sides only differ by one variable duplication. We sketch the method by treating a concrete example: equation (2) above. Note that this equality is of the desired form with $t[N] \equiv (yN)z$. Assume T preserves (2) and let X be a set. Once again, we define a convenient algebra: its carrier is X^5, and every binary operation of Σ is interpreted with the morphism $m\colon X^5 \times X^5 \to X^5$, graphically represented in (3). The map m can be categorically defined as $\langle \pi_1, \pi_7, \pi_2, \pi_6, \pi_4 \rangle$, but we will rather describe it as making a series of connections (seen as wires) between the components of its output and of its left and right inputs. The outputting wires are respectively labelled L^*, R^*, LR^*, RL^* and LRL^*. We represent the syntax tree of t in (4) (x_1 and x_2 representing the two duplicates of x). Encoding locations in the tree as words with letters L and R, the node where the duplication occurs is labelled LR. Let us now picture this tree with m-boxes on each node. By construction, the label we gave to each outputting wire describes the set of locations in the tree that are traversed by this thread.

▶ **Lemma 21.** *The binary operation m on X^5 satisfies the equation* (2).

Proof. As $(yx)z = (y(xx))z$ is preserved by T, the equality on the left in the diagram below holds. From this, pulling down the unfinished threads, we get the equality on the right.



As in the case of idempotence case, we now have two wires "wrapping" the bubble (LR^* and LRL^*). By Lemma 19, we can ignore the unfinished wires and obtain diagram (1). ◀

For any equation of the form $t[x] = t[x \cdot x]$, we can design m satisfying the equality and allowing to show relevance. We only have to make sure that the box representing m features two wires realising the connections L^* and R^*, as well as two others labelled wR^* and wL^*, where w is the word on $\{L, R\}^*$ describing the location of the variable duplication in t.

▶ **Theorem 22.** *T is relevant if it preserves $t[x] = t[x \cdot x]$ when t only contains binary ops.*

Although this result may seem too specialised, the process described above actually applies to other equations. Recall that our strategy relies on defining a convenient algebra to derive relevance from the preservation of a particular equation. If T preserves $(x+x)\cdot y = x\cdot y$, in particular T preserves it on an algebra where $\cdot$ and $+$ are interpreted as identical, hence why we only needed to define one binary operation in the previous paragraph. We may generalise this even further to treat n-ary operations: if $f(x, x, z) = x \cdot z$ is preserved by T, then in particular it is on an algebra where $f(x, x, z) = (x \cdot x) \cdot z$ (as long as we can define such an algebra where the considered equation also holds). Since we have treated $(x \cdot x) \cdot z = x \cdot z$ above, our conclusion also applies to $f(x, x, z) = (x \cdot x) \cdot z$. We have obtained the property of relevance from any possible case featuring binary operations, thus we also obtain for free the case of n-ary operations (where $n > 1$).

▶ **Theorem 23.** *Let t be a term without constants. T is relevant if it preserves $t[x] = t[x \cdot x]$.*

We can generalise this even further to cover equations outside the strict-dup class. If we use the above approach to treat an equation $t[x] = t[x \cdot x]$, it turns out we can slightly modify the equation without affecting our result. Consider the example $(yx)z = (y(xx))z$ again. At the end of our process, the only component of the y that is connected to the output is the first one (through the L^* wire). By construction, adding another iteration of m with y on its left input would not change this fact. Let us then substitute y with yv in the equation (for v any new variable). The position of v is coded as the word LLR, which does not belong to the language of any of our outputting wires, therefore v has no influence on the matching of the outputs. In other words, the definition of m allowing to prove relevance from the preservation of $(yx)z = (y(xx))z$ also applies to $((yv)x)z = (y(xx))z$, which is a one-drop equation. One could even substitute v with a more complicated term to obtain another equation, whose preservation would still lead to relevance. This last theorem applies therefore to many equalities outside the case $t[x] = t[x \cdot x]$, even though the exact class described by these modifications is cumbersome to define.

▶ **Example 24.** Because Thm. 23 applies to $z(xx) = zx$, it is also the case for $z(xx) = (zy)x$.

n-relevance. We have shown that the preservation of $x \cdot x = x$ implies relevance. What about the preservation of $f(x,x,x) = x$? We will see that it does not imply relevance, but rather a weaker property which we will call 3-relevance. To define n-relevance in general, we introduce n-ary variations of existing maps: Δ^n is the n-times duplication operator $X \to X^n$ and $\chi^n \equiv \langle T\pi_1, \ldots, T\pi_n \rangle$. Now, we say T is n-relevant iff $\psi^n \circ \Delta^n = T\Delta^n$. Or equivalently iff $\psi^n \circ \chi^n = \mathrm{id}$.

▶ **Proposition 25.** *Relevance implies n-relevance for $n \geq 2$.*

For a commutative monoid M and the monad $M \times X$ from Example 4, we have $(\psi^n \circ \chi^n)(v,(x_1,\ldots,x_n)) = \psi^n((v,x_1),\ldots,(v,x_n)) = (v^n,(x_1,\ldots,x_n))$ and so $M \times X$ is n-relevant iff $w^n = w$ for all $w \in M$. Hence monads may be n-relevant but not m-relevant for any $n > m$. For affine monads the difference disappears:

▶ **Proposition 26.** *Given any $n \in \mathbb{N}$, if T is n-relevant and affine, then T is relevant.*

As promised, we relate n-relevance to preservation of equations:

▶ **Theorem 27.** *Assume Σ features an n-ary operation f^n. T preserves $f^n(x,\ldots,x) = x$ if and only if T is n-relevant.*

We have seen that the preservation of some 2-dup non-drop equations, like $xx = x$ and $(xx)y = xy$, implies relevance. However, the following statement shows that there are 2-dup non-drop equations whose preservation does not imply relevance. Indeed, $\mathbf{M}_{\mathbb{Z}_2}$ is not relevant, but:

▶ **Proposition 28.** *The generalised multiset monad $\mathbf{M}_{\mathbb{Z}_2}$ preserves $x(yy) = yx$.*

Preservation of discerning equations. We present a class of equations for which relevance is necessary for preservation. A *2-discerning equation* $t_1 = t_2$ is a 2-dup non-drop equation, where only one variable, say x_1 out of $x_1,\ldots,x_n$ is duplicated and only one side, say t_2, which can distinguish the places where x_1 is duplicated in the following sense: the linear equation $s_2 = s_2'$ in $x_1, x_1', x_2, \ldots, x_n$ fixed by $t_2 = s_2[x_1'/x_1]$ and $s_2' = s_2[x_1/x_1', x_1'/x_1]$ is not derivable from $t_1 = t_2$.

▶ **Example 29.** The equation $x(yy) = yx$ is *not* 2-discerning as it implies $xy = yx$ and in particular $x(yy') = x(y'y)$. On the other hand $y(xy) = yx$ is 2-discerning. This requires one to show that $y(xy') = y'(xy)$ isn't derivable from $y(xy) = yx$, which is easily seen by noting that all terms equal to $y'(xy)$ must start with y' as well. In fact, all of the following equations are 2-discerning, which are essentially all the remaining candidates on two variables: $(yy)x = yx, (yx)y = yx, (xy)y = yx, y(yx) = yx$, and $y(xy) = yx$.

Theorem 31 states that relevance is equivalent to preservation of 2-discerning equations. In the proof [18], we assume that T is finitary – that is, T is presentable as an algebraic theory $\mathbb{T}$, i.e. TX is the free $\mathbb{T}$-algebra over X; Tf for $f\colon X \to Y$ maps a term $M \in TX$ to $M[x/f(x)]$; η_X maps x to the term x and μ_X maps a term over terms to the collapsed term. The argument for arbitrary monads (which are presented by infinitary algebraic theories) is similar, but heavier on paper. We first relate relevance of a finitary monad to its presentation; this algebraic characterisation can be found with slightly different notation in Figure 7 of [7].

▶ **Proposition 30.** *Suppose T is a monoidal monad on* **Set** *presented by an algebraic theory $\mathbb{T}$. Then T is relevant iff for every n-ary operator f of $\mathbb{T}$ we have $\overleftrightarrow{f}((x_{ij})_{ij}) = \vec{f}((x_{ii})_i))$, where $(x_{ij})_{ij}$ is a $n \times n$ matrix over any set X, $\vec{f}((y_i)_i) \equiv f(y_1,\ldots,y_n)$ and $\overleftrightarrow{f}((y_{ij})_{ij}) \equiv f(f(y_{11},\ldots,y_{1n}),\ldots,f(y_{n1},\ldots,y_{nn}))$.*

For instance, in an algebraic theory presenting a relevant monoidal monad, we must have $f^2 = f$ for any unary f and $g(g(a,b),g(c,d)) = g(a,d)$ for any binary g.

▶ **Theorem 31.** *Suppose $t_1 = t_2$ is a 2-discerning equation and T is a monoidal monad on* **Set**. *Then T is relevant if and only if T preserves $t_1 = t_2$.*

6 Related work

The notions of relevant and affine monads are systematically studied in [11, 6], but those works do not treat preservation of equations. Pioneering work on the preservation of algebraic features by a monad goes back to [4]. Without any notion of category theory, Gautam pinpoints exactly which equations are preserved by the powerset monad and highlights the importance of variable duplications or deletions.

Methods for combining the features of two monads have been discussed in several papers. Hyland et al. work out two canonical constructions in [5], the *sum* and the *tensor* of monads. Our work is closer to King and Wadler's study in [8], as they use distributive laws. Later, Manes and Mulry show in [14] and [15] that a correspondence between categorical properties of one monad and algebraic features of another may lead to a distributive law. Their work also sheds light on the importance of a monoidal structure in the problem of lifting signatures. This approach is then generalised to affine and relevant monads in [3]. Our contributions extend this work by showing on the one hand that some of the sufficient conditions of [3] are necessary (Theorems 12, 31, 23), and on the other hand that some algebraic features require finer categorical conditions to be preserved (Theorem 27).

In Section 5, we rely on presenting monads with algebraic theories in order to study their composition. This approach is related to Zwart and Marsden's method in [21]. In that paper, the authors elaborate on the concept of composite theory, introduced and studied by Pirog and Staton in [19], and establish the absence of distributive laws under different algebraic conditions on the considered monads. The main difference is that in [21], the focus is on combining algebraic theories, whereas the current paper relates algebraic structures to categorical properties of monads (relevance and affineness). These properties, along with other categorical conditions, are characterised algebraically in a similar manner to Proposition 30 by Kammar and Plotkin in [7], and our work connects this characterisation with the preservation of certain equations.

7 Conclusions and Future work

We have systematically related preservation of drop and dup equations to affineness and relevance of monoidal monads, respectively. There are several avenues for future work. First, this paper focuses on monads on **Set**. Generalising this work to monads on arbitrary Cartesian monoidal categories would be a natural follow-up and would require to use a more abstract notion of equation. Second, our work focuses specifically on monoidal liftings. Therefore, the distributive laws that we obtain from preservation are of a specific shape. It would be interesting to algebraically characterise which distributive laws arise in this way. Finally, we would like to go beyond the world of monoidal liftings, possibly allowing a non-affine monad to preserve drop equations for such non-canonical liftings. This could give a new perspective on the construction of distributive laws.

References

1 Ronald V. Book and Friedrich Otto. String-rewriting systems. In *String-Rewriting Systems*, pages 35–64. Springer, 1993.
2 J.R.B. Cockett and R.A.G. Seely. Linearly distributive functors. *Journal of Pure and Applied Algebra*, 143(1-3):155–203, 1999.
3 Fredrik Dahlqvist, Alexandra Silva, and Louis Parlant. Layer by layer: composing monads. In *ICTAC*, 2018.
4 N.D. Gautam. The validity of equations of complex algebras. *Archiv für Mathematische Logik und Grundlagenforschung*, 3(3-4):117–124, 1957.
5 Martin Hyland, Paul Blain Levy, Gordon Plotkin, and John Power. Combining continuations with other effects. In *Proc. Continuations Workshop*, 2004.
6 Bart Jacobs. Semantics of weakening and contraction. *Annals of pure and applied logic*, 69(1):73–106, 1994.
7 Ohad Kammar and Gordon D. Plotkin. Algebraic foundations for effect-dependent optimisations. In *POPL*, pages 349–360. ACM, 2012.
8 David J. King and Philip Wadler. Combining monads. In *Functional Programming, Glasgow 1992*, pages 134–143. Springer, 1993.
9 Bartek Klin and Julian Salamanca. Iterated covariant powerset is not a monad. *MFPS XXXIV*, 2018.
10 Anders Kock. Monads on symmetric monoidal closed categories. *Archiv der Mathematik*, 21(1):1–10, 1970.
11 Anders Kock. Bilinearity and cartesian closed monads. *Mathematica Scandinavica*, 29(2):161–174, 1972.
12 Anders Kock. Strong functors and monoidal monads. *Archiv der Mathematik*, 23(1):113–120, 1972.
13 Saunders Mac Lane. *Categories for the working mathematician*, volume 5. Springer, 2013.
14 Ernie Manes and Philip Mulry. Monad compositions i: general constructions and recursive distributive laws. *Theory and Applications of Categories*, 18(7):172–208, 2007.
15 Ernie Manes and Philip Mulry. Monad compositions ii: Kleisli strength. *Mathematical Structures in Computer Science*, 18(3):613–643, 2008.
16 Micah Blake McCurdy. Graphical methods for Tannaka duality of weak bialgebras and weak Hopf algebras in arbitrary braided monoidal categories. *arXiv preprint arXiv:1110.5542*, 2011.
17 Paul-André Melliès. Functorial boxes in string diagrams. In *International Workshop on Computer Science Logic*, pages 1–30. Springer, 2006.
18 Louis Parlant, Jurriaan Rot, Alexandra Silva, and Bas Westerbaan. Preservation of equations by monoidal monads, 2020. `arXiv:2001.06348`.
19 Maciej Pirog and Sam Staton. Backtracking with cut via a distributive law and left-zero monoids. *Journal of Functional Programming*, 27, 2017.
20 Ana Sokolova, Bart Jacobs, and Ichiro Hasuo. Generic trace semantics via coinduction. *Logical Methods in Computer Science*, 3, 2007.
21 Maaike Zwart and Dan Marsden. No-go theorems for distributive laws. In *LICS*, pages 1–13. IEEE, 2019.