Kleisli Categories with Display Maps

Moana Jubert
Inria Paris, Université Paris-Cité, IRIF, CNRS, Paris, France

Abstract

In this paper, we show that the Kleisli category $\mathcal{C}_T$ for a monad $T : \mathcal{C} \longrightarrow \mathcal{C}$ is sometimes a structured display map category, meaning that it provides a minimal categorical setting for dependent type theory. If the underlying category $\mathcal{C}$ is itself a (structured) display map category, we prove that under mild conditions the canonical functor $F_T : \mathcal{C} \longrightarrow \mathcal{C}_T$ induces a bijection between the contexts, types, and terms of both $\mathcal{C}$ and $\mathcal{C}_T$, thus establishing a kind of "collapsing" result. Finally, we provide several examples and nonexamples, most notably showing that Δ^{op} equipped with the class of surjective maps is a (structured) display map category.

2012 ACM Subject Classification Theory of computation → Type theory; Theory of computation → Categorical semantics

Keywords and phrases Kleisli categories, Structured display map categories, Dependent type theory

Digital Object Identifier 10.4230/LIPIcs.TYPES.2025.9

Acknowledgements I owe a great debt of gratitude to the scientific diversity of my laboratory, the Institut de Recherche en Informatique Fondamentale (IRIF).

1 Introduction

This paper is about constructing new categorical models of dependent type theories from well-behaved monads. It emerged from one of the author's ongoing lines of inquiry into defining a "language of profunctors". The path presented here proved unpromising for this specific goal, most notably because of the negative result of Example 4.12. In addition, other successful lines of inquiry already exist in the literature, such as *virtual equipment type theory* by Max New and Daniel Licata [24], further developed by Hayato Nasu [23]. As a result, we really discuss profunctors only to explain the reasoning that led to this paper, and its core material may be read independently. We believe nonetheless that the community will find the results we collected along the way useful and interesting.

A profunctor $\mathcal{A} \nrightarrow \mathcal{B}$ from a small category $\mathcal{A}$ to a small category $\mathcal{B}$ consists of the data of a bifunctor $\mathcal{B}^{\mathrm{op}} \times \mathcal{A} \longrightarrow \mathbf{Set}$. Profunctors compose, and they assemble into a bicategory $\mathbf{Prof}$. There is a canonical, covariant embedding $\mathbf{Cat} \hookrightarrow \mathbf{Prof}$ which, to some extent, may be read as "decorating" $\mathbf{Cat}$ with the extra information of profunctors, allowing one to do "more" reasoning on small categories. This way of presenting profunctors motivated, among other things, the abstract notion of *proarrow equipment* [30] for which $\mathbf{Cat}$ equipped with profunctors is the primary example.

We now say that a profunctor is *representable* if it is the image of a functor under the canonical embedding. Presheaves over a small category $\mathcal{C}$ are special cases of profunctors $* \nrightarrow \mathcal{C}$ from the one-object category $*$ to the category $\mathcal{C}$, and accordingly a presheaf is representable exactly when it is so as a profunctor. Many concepts – if not *all* – may be reformulated in terms of representable presheaves and, by extension, representable profunctors. This prompted us to carefully study the interaction between general and representable profunctors so as to come up with a sort of *calculus of profunctors*[1] – a "language" for which $\mathbf{Prof}$, equipped with the class of all representable profunctors, would be the model.

[1] Part of this calculus has already been worked out by Jean Bénabou in [5], where profunctors are instead called *distributors*.

If such a language is possible, then **Prof** should fit into any one of the categorical interpretations[2] of type theory, modulo some changes. One possibility introduced in [1, §3.1.1] is that of *structured display map categories*, which are simply defined as categories equipped with a class of arrows stable under pullbacks, called *display maps*. This differs from the well-known notion of *display map category* in that we do not ask the class of display maps to be *replete*, *i.e.*, invariant under isomorphisms. It is natural to ask whether the class of representable profunctors could play the role of display maps for **Prof**.

Of course, **Prof** is a bicategory, so we would need to tweak the categorical interpretations in some way. It is worth mentioning the work of Benedikt Ahrens et al. in this direction [2], where they develop the foundations of *comprehension bicategories*. Let us set the dimensional issues aside for a moment to look at **Prof** from afar. If we squint our eyes, **Prof** looks like the Kleisli (bi)category of some (pseudo)monad on **Cat** sending any small category $\mathcal{C}$ to its large category of presheaves $\widehat{\mathcal{C}}$. Representable profunctors are the image of arrows under the canonical embedding $\mathbf{Cat} \longhookrightarrow \mathbf{Prof}$ induced by this monad. Then we ask whether the Kleisli (bi)category **Prof** equipped with the arrows in the image of the canonical embedding is a structured display map category. Obviously, such a monad could not possibly exist because of size issues. But *monads need not be endofunctors* [3], and the functor $\mathcal{C} \longmapsto \widehat{\mathcal{C}}$ from the category of small categories **Cat** into the category of locally small categories **CAT** is a canonical example of a *relative pseudomonad* [15] for which **Prof** is the associated Kleisli bicategory. Now, if we reformulate our reasoning in 1-categorical terms, given a (true) monad (T, η, μ) on a category $\mathcal{C}$, its associated Kleisli category $\mathcal{C}_T$ always comes with a canonical functor $F_T : \mathcal{C} \longrightarrow \mathcal{C}_T$, and we can ask whether the category $\mathcal{C}_T$ equipped with the arrows in the image of F_T (they are said, by analogy, to be *representable*) is a structured display map category. In this paper, we give sufficient conditions to answer this last question positively.

These sufficient conditions are adapted from an article by Dominique Bourn [7], who studied the existence of pullbacks in Kleisli categories in order to understand the relationship between their internal categories and the so-called *T-categories* in the sense of Albert Burroni [8]. Just as in Dominique Bourn's work, we see that *cartesian monads* (*cf.* Definition 2.5) are exceptionally well-behaved, in that when the base category has all pullbacks, the Kleisli category is always a structured display map category. We go one step further, however, by considering the case where the base category itself forms a structured display map category. This level of precision allows us to consider more examples of Kleisli categories that are structured display map categories for a proper subclass of the representable arrows. As a notable example, we explain in Example 4.8 how the category Δ^{op} (opposite of the simplex category Δ) is a Kleisli category, as well as a (structured) display map category when equipped with the class of surjective maps in Δ.

Our Contributions

We use and extend (Proposition 2.9) the results of Dominique Bourn to see that some Kleisli categories are in fact structured display map categories. If both the base category and its associated Kleisli category are structured display map categories, then we prove that, under mild conditions, there is an adjunction between the two in the 2-category of structured display map categories (Theorem 3.10, Corollary 3.11). We also show that under some conditions the contexts, types, and terms of the two are in bijection (Corollary 3.15). Finally, we provide an opinionated list of examples (*e.g.*, Example 4.8).

[2] We can mention, for example, *contextual categories* [9], *categories with families* [14], and *comprehension categories* [19]. They are all comparable to one another, as shown in [1].

Outline of This Paper

In Section 2, we recall Kleisli categories and study their pullbacks. We prove results about structured display map categories in Section 3, and, finally, in Section 4, we walk through several examples.

2 Kleisli Categories

Let (T, η, μ) be a monad on an arbitrary category $\mathcal{C}$.

▶ **Definition 2.1.** *We write $\mathcal{C}_T$ for the associated* Kleisli category *of T where*
- *objects $X, Y, \ldots$ of $\mathcal{C}_T$ are the same as those of $\mathcal{C}$;*
- *an arrow of $\mathcal{C}_T$, which we write $\sigma : X \longrightarrow\!\!\!\!\!\rightarrow Y$ with a marking, is the data of an arrow $\sigma : X \longrightarrow TY$ in $\mathcal{C}$.*

Identities and composition are defined using the unit η and multiplication μ of T. There is a canonical pair of adjoint functors

$$\mathcal{C} \underset{U_T}{\overset{F_T}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathcal{C}_T$$

where F_T is the identity on objects and sends any arrow $f : A \longrightarrow B$ of $\mathcal{C}$ to $\eta_B \circ f : A \longrightarrow\!\!\!\!\!\rightarrow B$, and U_T coincides with T on objects and sends any arrow $\sigma : X \longrightarrow\!\!\!\!\!\rightarrow Y$ of $\mathcal{C}_T$ to the arrow $\mu_Y \circ T\sigma : TX \longrightarrow TY$. Moreover, we have $U_T \circ F_T = T$ by definition.

▶ **Definition 2.2.** *An arrow of $\mathcal{C}_T$ lying in the image of F_T is said to be* representable. *We draw representable arrows $F_T(f) : A \longrightarrow B$ without the marking, and write* $\mathrm{Repr}(T) \coloneqq F_T(\mathrm{Arr}(\mathcal{C}))$ *for the class of all representable arrows of $\mathcal{C}_T$.*

Of course, the representing arrow of a representable arrow need not be unique. It is unique precisely when F_T is a faithful functor, which is equivalent to the unit η being monomorphic, *i.e.*, $\eta \circ f = \eta \circ g$ implies $f = g$.

▶ **Example 2.3.** Consider the monad $\mathcal{P} : \mathbf{Set} \longrightarrow \mathbf{Set}$ on **Set** sending a set A to its set of subsets $\mathcal{P}(A)$. Its associated Kleisli category is called **Rel**, and the data of an arrow $R : A \longrightarrow\!\!\!\!\!\rightarrow B$ is exactly the data of a *relation* $R \subseteq A \times B$. Representable arrows are exactly the (total) functional relations. The unit is monomorphic, and therefore the functor $F_{\mathcal{P}} : \mathbf{Set} \longrightarrow \mathbf{Rel}$ is faithful (but trivially *not* full).

$\mathcal{C}_T$ is, up to embedding, the full subcategory of *free* T-algebras of the category $\mathcal{C}^T$ of algebras over T – the so-called *Eilenberg-Moore category*. While $\mathcal{C}^T$ may often enjoy good properties such as the existence of limits and colimits, this is rarely the case for $\mathcal{C}_T$ (see, *e.g.*, [6, §4.3] for a discussion on the existence of limits and colimits in $\mathcal{C}^T$). In [28], the author gives sufficient conditions for $\mathcal{C}_T$ to be complete or cocomplete, depending on whether the obvious fully faithful inclusion $I : \mathcal{C}_T \hookrightarrow \mathcal{C}^T$ is a left or right adjoint. There is also a canonical functor $U^T : \mathcal{C}^T \longrightarrow \mathcal{C}$, and U_T factors through it via I, as depicted below

$$\begin{array}{ccc} \mathcal{C}_T & \xhookrightarrow{\;I\;} & \mathcal{C}^T \\ & {\scriptstyle U_T} \searrow \quad \swarrow {\scriptstyle U^T} & \\ & \mathcal{C} & \end{array}$$

▶ **Proposition 2.4.** *The functor U_T both preserves and reflects limits.*

Proof. U_T preserves limits because it is a right adjoint. It is well known that the functor $U^T : \mathcal{C}^T \longrightarrow \mathcal{C}$ creates limits and so, in particular, reflects them (see, *e.g.*, [27, Theorem 5.6.5] or [21, Exercise VI.2.2]). Now, if a cone in $\mathcal{C}_T$ is sent to a limit cone in $\mathcal{C}$ via $U_T = U^T \circ I$, then it is a limit cone in $\mathcal{C}^T$. But $\mathcal{C}_T$ is a full subcategory of $\mathcal{C}^T$, so it must be a limit cone in $\mathcal{C}_T$ as well. This proves that U_T reflects limits. ◀

Fix a subclass $\mathcal{A} \subseteq \mathrm{Arr}(\mathcal{C})$ of arrows of $\mathcal{C}$. We want to study the stability of $F_T(\mathcal{A})$ – which is a subclass of the representable arrows of $\mathcal{C}_T$ – under pullbacks. Proposition 2.4 above provides the most general (and therefore least convenient) criterion. Suppose we have the following cospan in $\mathcal{C}_T$

$$\begin{array}{ccc} & & A \\ & & \downarrow{\scriptstyle F_T(f)} \\ Y & \xrightarrow[\sigma]{} & B \end{array}$$

where the vertical arrow is representable and belongs to $F_T(\mathcal{A})$. Assuming $\mathcal{C}$ has enough pullbacks, we may look at the pullback of its image under U_T

$$\begin{array}{ccc} TA \times_{TB} TY & \xrightarrow{\psi} & TA \\ {\scriptstyle \phi}\downarrow & \lrcorner & \downarrow{\scriptstyle Tf} \\ TY & \xrightarrow[U_T(\sigma)]{} & TB \end{array} \tag{1}$$

If we can prove that $TA \times_{TB} TY \cong TX$ for some $X \in \mathcal{C}$, $\phi \cong Tp$ for some $p : X \longrightarrow Y$ belonging to $\mathcal{A}$, and $\psi \cong U_T(\tau)$ for some $\tau : X \longrightarrow\!\!\!\!\!\rightarrow A \in \mathcal{C}_T$, then, since U_T reflects pullbacks by Proposition 2.4, the square in $\mathcal{C}_T$ below must be a pullback as well

$$\begin{array}{ccc} X & \xrightarrow{\tau} & A \\ {\scriptstyle F_T(p)}\downarrow & \lrcorner & \downarrow{\scriptstyle F_T(f)} \\ Y & \xrightarrow[\sigma]{} & B \end{array}$$

and both vertical arrows belong to $F_T(\mathcal{A})$. Instead of (1) we may consider the following simpler pullback

$$\begin{array}{ccc} Y \times_{TB} TA & \longrightarrow & TA \\ {\scriptstyle \sigma^* Tf}\downarrow & \lrcorner & \downarrow{\scriptstyle Tf} \\ Y & \xrightarrow[\sigma]{} & TB \end{array}$$

and hope that $F_T(\sigma^* Tf)$ is somehow related to the pullback of $F_T(f)$ along σ in $\mathcal{C}_T$ when it exists. The rest of this section is dedicated to making this idea precise. We first present a simple criterion, due to Dominique Bourn, for a class $\mathcal{A} \subseteq \mathrm{Arr}(\mathcal{C})$ of arrows to be such that $F_T(\mathcal{A})$ is stable under pullbacks (Theorem 2.7).

▶ **Definition 2.5.** *A natural transformation is said to be* cartesian *when the naturality squares are all pullbacks. Similarly, we say that a functor is* cartesian *when it sends pullback squares to pullback squares. A monad is* cartesian *when all of its components are cartesian*[3].

▶ **Example 2.6.** The *partiality monad*

$$\begin{aligned} P : \mathbf{Set} &\longrightarrow \mathbf{Set} \\ X &\longmapsto X + * \end{aligned}$$

is cartesian. Its Kleisli category consists of sets and partial functions between them, and the representable arrows are exactly the total functions. Note that **Set** has all pullbacks, and so P preserves them.

▶ **Theorem 2.7** (Dominique Bourn [7, Proposition 3.12])**.** *Let (T, η, μ) be a monad on a category $\mathcal{C}$. Suppose that we have a class $\mathcal{A} \subseteq \mathrm{Arr}(\mathcal{C})$ of arrows of $\mathcal{C}$ such that the following conditions hold:*

1. *For every $f : A \longrightarrow B$ belonging to $\mathcal{A}$ and $\sigma : Y \longrightarrow TB$, the pullback*

$$\begin{array}{ccc} Y \times_{TB} TA & \dashrightarrow & TA \\ {\scriptstyle \sigma^* Tf}\downarrow & \lrcorner & \downarrow{\scriptstyle Tf} \\ Y & \xrightarrow[\sigma]{} & TB \end{array} \tag{2}$$

   does exist in $\mathcal{C}$, is preserved by T, and is such that $\sigma^* Tf$ belongs to $\mathcal{A}$;
2. *μ is cartesian.*

Then the class $F_T(\mathcal{A})$ is stable under pullbacks.

Proof. We reproduce the proof from Dominique Bourn. Using both hypotheses, we can paste together the following pullbacks:

$$\begin{array}{ccccc} T(Y \times_{TB} TA) & \longrightarrow & T^2 A & \xrightarrow{\mu_A} & TA \\ {\scriptstyle T(\sigma^* Tf)}\downarrow & \lrcorner & {\scriptstyle T^2 f}\downarrow & \lrcorner & \downarrow{\scriptstyle Tf} \\ TY & \xrightarrow[T(\sigma)]{} & T^2 B & \xrightarrow[\mu_B]{} & TB \end{array}$$

The outer rectangle is a pullback by the pasting law. But it is also the image of the commutative square in $\mathcal{C}_T$

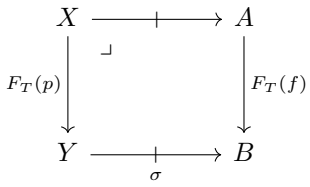
$$\begin{array}{ccc} Y \times_{TB} TA & \longrightarrow\!\!\!\!\!\!\mid\ & A \\ {\scriptstyle F_T(\sigma^* Tf)}\downarrow & & \downarrow{\scriptstyle F_T(f)} \\ Y & \xrightarrow[\sigma]{}\!\!\!\!\!\!\mid\ & B \end{array}$$

via the functor $U_T : \mathcal{C}_T \longrightarrow \mathcal{C}$. Now, U_T reflects pullbacks by Proposition 2.4, hence the desired result. ◀

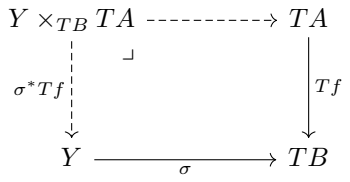
[3] Some authors additionally require that the underlying category has all pullbacks.

Dominique Bourn's criterion is not only sufficient but also necessary when the monad is cartesian. If a representable pullback exists, then it *must* be of the form given by the pullback in diagram (2). This is a new result highlighting the robustness of this construction. We first prove the proposition below, showing that the representable arrow $F_T(\sigma^* Tf)$ is always a retract of the actual pullback in $\mathcal{C}_T$, provided it exists and is itself representable.

▶ **Proposition 2.8.** *Suppose we have a pullback square in $\mathcal{C}_T$ as below*

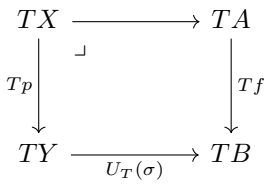


where the left and right vertical arrows are representable. If the following pullback in $\mathcal{C}$ exists

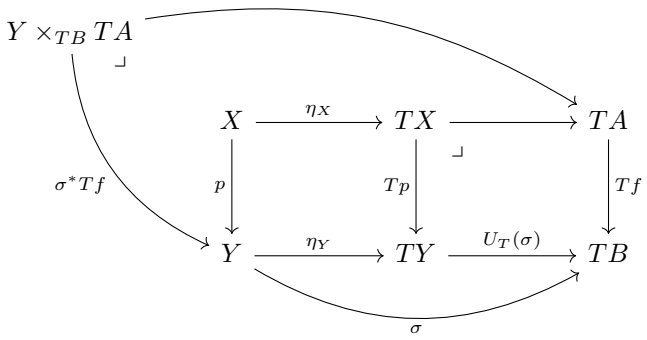


then the arrow $F_T(\sigma^* Tf)$ is a retract of $F_T(p)$ in $\mathcal{C}_T/Y$.

Proof. Since U_T preserves limits, it sends the first pullback square to a pullback square in $\mathcal{C}$



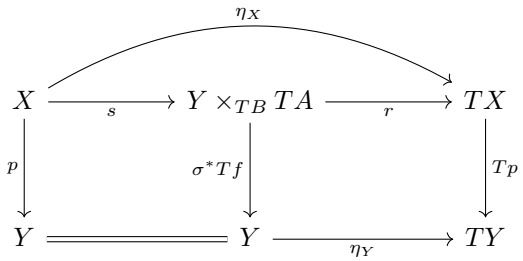
We now put together the two diagrams above and insert the naturality square of the unit η



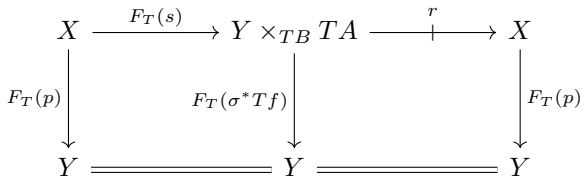
(3)

The bottom triangle commutes by drawing the appropriate string diagram. By the universal property of the pullback, the commutation of the outer rectangle implies the existence of a unique arrow $r : Y \times_{TB} TA \longrightarrow TX$. Similarly, the commutation of the inner rectangle implies the existence of a unique arrow $s : X \longrightarrow Y \times_{TB} TA$. Both arrows fit into the

following commutative diagram in $\mathcal{C}$



The top triangle commutes by the universal property of the pullback and uniqueness of arrows. We get that the diagram in $\mathcal{C}_T$ below commutes, using the fact that $Tp \circ r$ in $\mathcal{C}$ is equal to $F_T(p) \circ r$ in $\mathcal{C}_T$, as can be checked by drawing the appropriate string diagram



We are left to show that $r \circ F_T(s) = 1_X$ in $\mathcal{C}_T$. This unfolds in $\mathcal{C}$ as

$$\mu_X \circ Tr \circ \eta_{Y \times_{TB} TA} \circ s = \eta_X$$

This is true by rewriting and diagram chasing. ◀

With the additional hypothesis that the unit η of the monad is *cartesian*, we can finally show that this retract collapses and that $F_T(\sigma^* Tf)$ is indeed isomorphic to the pullback. This is the statement of Proposition 2.9.

▶ **Proposition 2.9.** *The arrows $\sigma^* Tf : Y \times_{TB} TA \longrightarrow Y$ and $p : X \longrightarrow Y$ in the diagram* (3) *are isomorphic when η is cartesian.*

Proof. The left inner square in the diagram (3) is a pullback because η is cartesian, and the right inner square is also a pullback by hypothesis. The whole inner rectangle must be a pullback by the pasting law, and by uniqueness of pullbacks, it must be isomorphic to the outer rectangle, *i.e.*, $\sigma^* Tf : Y \times_{TB} TA \longrightarrow Y$ is isomorphic to $p : X \longrightarrow Y$. ◀

To conclude this section, we discuss two general examples.

▶ **Example 2.10.** Let $\mathcal{C}$ be a category with all pullbacks and fix $\mathcal{A} := \mathrm{Arr}(\mathcal{C})$. Cartesian monads over $\mathcal{C}$ are the best of all possible worlds, since they trivially satisfy the hypotheses of Theorem 2.7. As a result, $F_T(\mathcal{A}) =: \mathrm{Repr}(T)$ is automatically stable under pullbacks. When applied to Example 2.6, we recover the fact that the pullback of a total function along a partial function is again a total function[4]. Many more examples will be covered in Section 4.

▶ **Example 2.11.** If T is a cartesian monad and $\mathcal{A} \subseteq \mathrm{Arr}(\mathcal{C})$ is a class of arrows of $\mathcal{C}$ stable under pullbacks such that $T(\mathcal{A}) \subseteq \mathcal{A}$, then the hypotheses of Theorem 2.7 are also satisfied. For example, again with the partiality monad, the class Mono of injective maps in **Set** is stable under pullbacks and we indeed have $P(\mathrm{Mono}) \subseteq \mathrm{Mono}$. We conclude that the class $F_P(\mathrm{Mono})$ of arrows of $\mathbf{Set}_P$ is stable under pullbacks.

[4] Note that the Kleisli category $\mathbf{Set}_P$ of the partiality monad has, in fact, all pullbacks.

3 Structured Display Map Categories

3.1 Definition and Properties

▶ **Definition 3.1** (Benedikt Ahrens et al. [1, Definition 18]). *Let $\mathcal{C}$ be an arbitrary category and $\mathcal{A} \subseteq \mathrm{Arr}(\mathcal{C})$ a class of arrows of $\mathcal{C}$. The pair $(\mathcal{C}, \mathcal{A})$ is a* structured display map category *if $\mathcal{A}$ is stable under pullbacks, meaning that for any arrows $f : A \longrightarrow B \in \mathcal{A}$ and $g : Y \longrightarrow B \in \mathcal{C}$ there exists a pullback square*

$$\begin{array}{ccc} X & \longrightarrow & A \\ {\scriptstyle p}\downarrow & \lrcorner & \downarrow{\scriptstyle f} \\ Y & \xrightarrow[g]{} & B \end{array}$$

such that $p : X \longrightarrow Y \in \mathcal{A}$. The arrows in $\mathcal{A}$ are called display maps.

▶ Remark 3.2. To say that $\mathcal{A}$ is stable under pullbacks is different from saying that *every pullback* of an arrow of $\mathcal{A}$ belongs to $\mathcal{A}$. We merely ask that among the possible choices of pullbacks up to isomorphism, at least one of them is in $\mathcal{A}$.

If we ask for $\mathcal{A}$ to contain every pullback, we get the stronger notion of *display map category* – without the "structured" – which is much more familiar to type theorists. A display map category is a structured display map category such that the class of arrows is invariant under isomorphisms (the class is said to be *replete*). Display map categories are more restrictive and therefore better-behaved.

We chose the more general setting of structured display map categories because it does not require us later to think too much about the isomorphisms inside a Kleisli category, as they are too often mysterious. We would like to point out that a structured display map category with no nontrivial isomorphisms is automatically a display map category by definition. This is, for example, the case of Δ^{op} in Example 4.8.

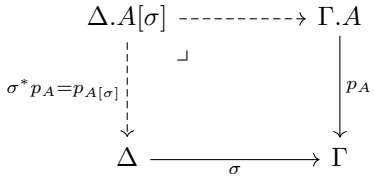
▶ Remark 3.3. It is, of course, always possible to saturate the class $\mathcal{A}$ of a structured display map category $(\mathcal{C}, \mathcal{A})$ so that it is invariant under isomorphisms. This defines a 2-functor **sDMC** $\longrightarrow$ **DMC** into the full sub-2-category of "mere" display map categories, and this 2-functor is in fact left 2-adjoint to the inclusion of **DMC** into **sDMC**. We also mention that structured display map categories are exactly the *full* comprehension categories where the comprehension is a "true" category inclusion, in that there is an *isomorphism* of 2-categories between **sDMC** and some 2-category of full comprehension categories. These two observations may be found in more detail in [1, §3.1.1].

▶ **Example 3.4.** Given a dependent type theory $\mathbb{T}$, we may construct a structured display map category $(\mathcal{C}_{\mathbb{T}}, \mathcal{A}_{\mathbb{T}})$ in the following way: the objects of $\mathcal{C}_{\mathbb{T}}$ are the valid contexts $\Gamma, \Delta, \Xi, \ldots$ of $\mathbb{T}$, and an arrow $\sigma : \Delta \longrightarrow \Gamma$ between contexts is a *substitution* between contexts (see, *e.g.*, the definition of *context morphism* in [18, §2.2.2]). We will keep the terminology "substitution" specific to type theory, *i.e.*, it is an n-tuple of terms satisfying some properties. Otherwise, we will use the more generic term "context morphism" for a general structured display map category. If we have a dependent type $\Gamma \vdash A$ of $\mathbb{T}$, there is a canonical substitution

$$p_A : \Gamma, x : A \longrightarrow \Gamma$$

from the context Γ extended with a fresh variable x of type A to Γ. We write $\Gamma.A$ rather than $\Gamma, x : A$, and $p_A : \Gamma.A \longrightarrow \Gamma$ is essentially a projection (which "forgets" the variable of

type A). We now define $\mathcal{A}_{\mathbb{T}}$ to be the collection of all such canonical substitutions. This class is indeed stable under pullbacks, as dependent types are stable under substitution



The above example shows how a structured display map category $(\mathcal{C}, \mathcal{A})$ may be seen as a minimal categorical model of "core" dependent type theory, where by "core" we mean the pure structural aspects of type dependency, prior to any type constructions. The contexts of the type theory are interpreted as objects of $\mathcal{C}$, the substitutions between contexts are interpreted as arrows of $\mathcal{C}$, and the dependent types are interpreted as display maps – a display map $p_A : \Gamma, x : A \longrightarrow \Gamma$ interprets the judgment $\Gamma \vdash A$ in the theory. To have, *e.g.*, Π-types or Σ-types, one would need additional structure on a structured display map category, just as is the case in other minimal settings like comprehension categories. We will discuss contexts, types, and terms of structured display map categories in Section 3.2 below.

▶ **Corollary 3.5.** *Suppose we have a monad (T, η, μ) on a category $\mathcal{C}$ and a class $\mathcal{A} \subseteq \mathrm{Arr}(\mathcal{C})$. Under the hypotheses of Theorem 2.7, the pair $(\mathcal{C}_T, F_T(\mathcal{A}))$ is a structured display map category.*

Proof. By definition. ◀

Structured display map categories come with their own flavor of morphism. We will see that when $(\mathcal{C}, \mathcal{A})$ and $(\mathcal{C}_T, F_T(\mathcal{A}))$ are structured display map categories, then under mild conditions the canonical functors $F_T : \mathcal{C} \longrightarrow \mathcal{C}_T$ and $U_T : \mathcal{C}_T \longrightarrow \mathcal{C}$ are instances of such morphisms.

▶ **Definition 3.6** (Benedikt Ahrens et al. [1, Definition 21])**.** *A morphism $F : (\mathcal{C}, \mathcal{A}) \longrightarrow (\mathcal{D}, \mathcal{B})$ of structured display map categories is the data of a functor $F : \mathcal{C} \longrightarrow \mathcal{D}$ such that*

- *$F(\mathcal{A}) \subseteq \mathcal{B}$;*
- *pullbacks along arrows of $\mathcal{A}$ are sent to pullbacks along arrows of $\mathcal{B}$.*

Structured display map categories assemble into a 2-category **sDMC** *where the 2-cells are all the natural transformations.*

Let us now fix a structured display map category $(\mathcal{C}, \mathcal{A})$.

▶ **Proposition 3.7.** *If (T, η, μ) is a monad on $\mathcal{C}$ such that μ is cartesian and*

$$T : (\mathcal{C}, \mathcal{A}) \longrightarrow (\mathcal{C}, \mathcal{A})$$

is a morphism of structured display map categories, then the pair $(\mathcal{C}_T, F_T(\mathcal{A}))$ is a structured display map category.

Proof. Unfolding the definitions, it is easy to check that the hypotheses of Theorem 2.7 are satisfied (see Example 2.11). ◀

▶ **Example 3.8.** If T is a cartesian monad on $\mathcal{C}$ such that $T(\mathcal{A}) \subseteq \mathcal{A}$, then

$$T : (\mathcal{C}, \mathcal{A}) \longrightarrow (\mathcal{C}, \mathcal{A})$$

is a morphism of structured display map categories, and by applying the above proposition, we get that $(\mathcal{C}_T, F_T(\mathcal{A}))$ is a structured display map category.

▶ **Proposition 3.9.** *Suppose that (T, η, μ) is a monad on $\mathcal{C}$ such that $(\mathcal{C}_T, F_T(\mathcal{A}))$ is a structured display map category. If $T(\mathcal{A}) \subseteq \mathcal{A}$, then*

$$U_T : (\mathcal{C}_T, F_T(\mathcal{A})) \longrightarrow (\mathcal{C}, \mathcal{A})$$

is a morphism of structured display map categories.

Proof. We have that $U_T \circ F_T = T$; therefore, $U_T(F_T(\mathcal{A})) = T(\mathcal{A}) \subseteq \mathcal{A}$ by hypothesis. Now, U_T preserves limits by Proposition 2.4 and therefore sends pullbacks of arrows in $F_T(\mathcal{A})$ to pullbacks of arrows in $\mathcal{A}$. ◀

▶ **Theorem 3.10.** *Suppose that (T, η, μ) is a monad on $\mathcal{C}$ such that $(\mathcal{C}_T, F_T(\mathcal{A}))$ is a structured display map category, and $T : (\mathcal{C}, \mathcal{A}) \longrightarrow (\mathcal{C}, \mathcal{A})$ is a morphism of structured display map categories. Then*

$$F_T : (\mathcal{C}, \mathcal{A}) \longrightarrow (\mathcal{C}_T, F_T(\mathcal{A}))$$

is also a morphism of structured display map categories. Moreover, there is an adjunction

$$(\mathcal{C}, \mathcal{A}) \underset{U_T}{\overset{F_T}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} (\mathcal{C}_T, F_T(\mathcal{A}))$$

in the 2-category **sDMC**.

Proof. The first condition for F_T to be a morphism of structured display map categories is trivially satisfied. A pullback square in $\mathcal{C}$ with the vertical arrows in $\mathcal{A}$ is sent by $U_T \circ F_T = T$ to a pullback square in $\mathcal{C}$ with the vertical arrows in $\mathcal{A}$ by hypothesis. But U_T reflects limits by Proposition 2.4; hence, the image of the pullback square under F_T must be a pullback square as well. Now, U_T is also a morphism of structured display map categories by Proposition 3.9, and since $F_T \dashv U_T$ in **Cat**, they must also be adjoint in **sDMC** because the 2-category contains all natural transformations – *i.e.*, the unit and counit exist in **sDMC** and satisfy the triangle identities. ◀

▶ **Corollary 3.11.** *Suppose that $(\mathcal{C}, \mathcal{A})$ is a structured display map category, and (T, η, μ) is a cartesian monad on $\mathcal{C}$ such that $T(\mathcal{A}) \subseteq \mathcal{A}$. Then*

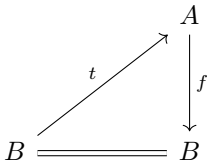
1. *$T : (\mathcal{C}, \mathcal{A}) \longrightarrow (\mathcal{C}, \mathcal{A})$ is a morphism of structured display map categories;*
2. *$(\mathcal{C}_T, F_T(\mathcal{A}))$ is a structured display map category;*
3. *there is an adjunction $F_T \dashv U_T$ in the 2-category* **sDMC** *between $(\mathcal{C}, \mathcal{A})$ and $(\mathcal{C}_T, F_T(\mathcal{A}))$.*

Proof. $T : (\mathcal{C}, \mathcal{A}) \longrightarrow (\mathcal{C}, \mathcal{A})$ is a morphism of structured display map categories by Example 3.8. We can then apply Proposition 3.7 to prove item 2, and we get item 3 by combining items 1 and 2 and applying Theorem 3.10 above. ◀

This last corollary is arguably one of the most useful results for "generating" new structured display map categories, as cartesian monads satisfying the hypotheses seem to be somewhat frequent in the wild. For example, it instantly applies to the partiality monad on **Set** with the class of monomorphisms, as seen in Example 2.11. We give a table with more examples in Section 4.1.

3.2 Contexts, Types, and Terms

▶ **Definition 3.12.** *Consider a structured display map category* $(\mathcal{C}, \mathcal{A})$*. We call* contexts *the objects of* $\mathcal{C}$ *and* types *the arrows in* $\mathcal{A}$*. Given a type* $f : A \longrightarrow B \in \mathcal{A}$*, a* term *of* f *is a section* $t : B \longrightarrow A$ *of* f *in* $\mathcal{C}$*, as illustrated below*



Continuing with Example 3.4, one can check that a substitution $t : \Gamma \longrightarrow \Gamma.A$ which is also a section of a display map $p_A : \Gamma.A \longrightarrow \Gamma$ indeed amounts to a judgment $\Gamma \vdash t : A$ of $\mathbb{T}$.

Let us fix a monad (T, η, μ) on a category $\mathcal{C}$ such that both $(\mathcal{C}, \mathcal{A})$ and $(\mathcal{C}_T, F_T(\mathcal{A}))$ are structured display map categories. When $F_T : \mathcal{C} \longrightarrow \mathcal{C}_T$ is a morphism of structured display map categories, then, since it is also *bijective-on-objects*, the functor establishes a bijective correspondence between the contexts of $(\mathcal{C}, \mathcal{A})$ and $(\mathcal{C}_T, F_T(\mathcal{A}))$. When η is monomorphic, the functor F_T is faithful. This entails, in particular, that both $\mathcal{A}$ and $F_T(\mathcal{A})$ are in bijective correspondence, witnessed by the functor F_T. Since types are modeled by display maps, F_T bijectively sends types of $(\mathcal{C}, \mathcal{A})$ to types of $(\mathcal{C}_T, F_T(\mathcal{A}))$. It thus only remains to compare the terms of $(\mathcal{C}, \mathcal{A})$ and $(\mathcal{C}_T, F_T(\mathcal{A}))$. The goal of the rest of this section is to see that under some conditions, the functor F_T also bijectively sends terms of types of $(\mathcal{C}, \mathcal{A})$ to terms of types of $(\mathcal{C}_T, F_T(\mathcal{A}))$.

▶ **Lemma 3.13.** *Suppose we have a monad* (T, η, μ) *on a category* $\mathcal{C}$ *such that* η *is both monomorphic and cartesian. Then, for any pair* σ, τ *of composable arrows of* $\mathcal{C}_T$*, if* $\tau \circ \sigma$ *and* τ *are representable, then so is* σ*.*

Proof. This is exactly one implication of [7, Proposition 3.6]. ◀

▶ **Theorem 3.14.** *Suppose we have a monad* (T, η, μ) *on a category* $\mathcal{C}$ *such that* η *is both monomorphic and cartesian. Then, given an arrow* $f : A \longrightarrow B$ *of* $\mathcal{C}$*, the sections of* f *in* $\mathcal{C}$ *and the sections of* $F_T(f) : A \longrightarrow B$ *in* $\mathcal{C}_T$ *are in one-to-one correspondence.*

Proof. F_T trivially sends sections of f to sections of $F_T(f)$ by functoriality. In the other direction, if $t : B \longrightarrow\!\!\!\!\!\rightarrow A$ is a section of $F_T(f)$ in $\mathcal{C}_T$, then both $F_T(f) \circ t = 1_B$ and $F_T(f)$ are representable. By Lemma 3.13, we conclude that t must be representable as well, and since η is monomorphic, the representing arrow, which we write $t' : B \longrightarrow A$, is unique. Now we have that $F_T(f \circ t') = F_T(f) \circ F_T(t') = F_T(f) \circ t = 1_B$ in $\mathcal{C}_T$. By uniqueness of the representing arrow of 1_B in $\mathcal{C}_T$, we conclude that $f \circ t' = 1_B$ in $\mathcal{C}$, *i.e.*, t' is a section of f. It is easy to see that these operations are inverses of each other. ◀

We immediately deduce the following "collapsing result".

▶ **Corollary 3.15.** *Suppose we have a monad* (T, η, μ) *on a category* $\mathcal{C}$ *such that* η *is both monomorphic and cartesian. Suppose moreover that we are given a class* $\mathcal{A} \subseteq \mathrm{Arr}(\mathcal{C})$ *of arrows such that* $(\mathcal{C}, \mathcal{A})$ *and* $(\mathcal{C}_T, F_T(\mathcal{A}))$ *are structured display map categories. Then the contexts, types, and terms of both* $(\mathcal{C}, \mathcal{A})$ *and* $(\mathcal{C}_T, F_T(\mathcal{A}))$ *are in bijective correspondence.*

This bijective correspondence is witnessed by the functor $F_T : \mathcal{C} \longrightarrow \mathcal{C}_T$ *when it is a morphism of structured display map categories (*cf. *Theorem 3.10).*

Proof. This is trivially a consequence of Theorem 3.14 above and the various remarks we made about F_T. ◀

▶ **Example 3.16.** Consider the partiality monad $P : \mathbf{Set} \longrightarrow \mathbf{Set}$ again. This monad is cartesian, and therefore $(\mathbf{Set}_P, \mathrm{Repr}(P))$ is a structured display map category. An arrow $\sigma : \Delta \longmapsto \Gamma$ of $\mathbf{Set}_P$, *i.e.*, an arrow $\Delta \longrightarrow P(\Gamma)$ in $\mathbf{Set}$, may be thought of as a partially defined substitution from a context Δ to a context Γ. The unit of the monad is both monomorphic and cartesian, and by Corollary 3.15 we conclude that the contexts, types, and terms of both $(\mathbf{Set}, \mathrm{Arr}(\mathbf{Set}))$ and $(\mathbf{Set}_P, \mathrm{Repr}(P))$ are in bijection. But the structured display map category $(\mathbf{Set}_P, \mathrm{Repr}(P))$ has, by contrast, strictly more *context morphisms*, *i.e.*, arrows between contexts.

In general, a context $F_T(\Gamma) \in \mathcal{C}_T$ may be semantically very different from $\Gamma \in \mathcal{C}$.

▶ **Example 3.17.** Consider again the partiality monad $P : \mathbf{Set} \longrightarrow \mathbf{Set}$ from Example 3.16. The terminal object $* \in \mathbf{Set}$ interprets the *empty context*, and the inclusion $\varnothing \longrightarrow *$ of the empty set into the singleton interprets the empty type over the empty context, *i.e.*, the object $\varnothing \in \mathbf{Set}$ interprets the *empty-type-in-empty-context*.

But $*$ is *not* the terminal object in $\mathbf{Set}_P$. The terminal object, which also happens to be initial, is in that case $\varnothing \in \mathbf{Set}_P$. The morphism F_T actually sends the empty-type-in-empty-context $\varnothing \in \mathbf{Set}$ to the terminal object $\varnothing \in \mathbf{Set}_P$, and the empty context $* \in \mathbf{Set}$ is sent to something else.

To summarize, under the hypotheses of both Theorem 3.10 and Corollary 3.15, the functor $F_T : (\mathcal{C}, \mathcal{A}) \longrightarrow (\mathcal{C}_T, F_T(\mathcal{A}))$ bijectively transports contexts, types, and terms of $(\mathcal{C}, \mathcal{A})$ to $(\mathcal{C}_T, F_T(\mathcal{A}))$, although the images under F_T may be very different from the point of view of type theory (see Example 3.17). This can be read both positively and negatively. On the one hand, since $\mathcal{C}$ is generally conceptually simpler to understand than the Kleisli category $\mathcal{C}_T$, the fact that $(\mathcal{C}_T, F_T(\mathcal{A}))$ is "essentially the same as" $(\mathcal{C}, \mathcal{A})$ is useful in itself. On the other hand, this suggests that a structured display map category $(\mathcal{C}_T, F_T(\mathcal{A}))$ is really interesting only when η is not monomorphic. We would like to stress again that even if $F_T : (\mathcal{C}, \mathcal{A}) \longrightarrow (\mathcal{C}_T, F_T(\mathcal{A}))$ "collapses" the contexts, types, and terms, the operation of taking the structured display map category $(\mathcal{C}_T, F_T(\mathcal{A}))$ almost always strictly adds new context morphisms to $(\mathcal{C}, \mathcal{A})$.

Finally, a bijection between terms in type theory implies a bijection between substitutions, as substitutions are n-tuples of terms. Since substitutions are interpreted as context morphisms in structured display map categories, if two structured display map categories have the same terms, then surely they should have the same context morphisms. The discussion and examples above show that this is not the case. This illustrates the "frugal framework" of structured display map categories: to have a fine-grained categorical model, matching the type-theoretic intuitions, one would need additional structure and properties on the base category.

4 Examples

We will now see several other examples and nonexamples.

▶ **Example 4.1.** The identity monad on a category $\mathcal{C}$ trivially satisfies the hypotheses of both Theorem 2.7 and Theorem 3.10.

▶ **Example 4.2.** When $\mathcal{C}$ has all pullbacks and $\mathcal{A} = \mathrm{Arr}(\mathcal{C})$, then cartesian monads over $\mathcal{C}$ satisfy the hypotheses of both theorems as well. This includes, in particular, *parametric right adjoint* monads and therefore *polynomial* monads. Indeed, we have the following inclusions

$$\text{Polynomial monads} \subseteq \text{Parametric right adjoint monads} \subseteq \text{Cartesian monads}$$

▶ **Example 4.3.** Suppose that $\mathcal{E}$ is an elementary topos. Then $(\mathrm{Epi}, \mathrm{Mono})$ is an *orthogonal factorization system* on $\mathcal{E}$, where the left class is the class of all epimorphisms and the right class is the class of all monomorphisms. By definition, $\mathcal{E}$ has finite limits and so, in particular, all pullbacks, and it is well known that the right class of an orthogonal factorization system is stable under pullbacks. We conclude that $(\mathcal{E}, \mathrm{Mono})$ is a structured display map category – in this case, the display maps are the monomorphisms,[5] and so, in essence, the dependent types are merely the propositions, *i.e.*, they express the *subobjects*.

Now, let $T : \mathcal{E} \longrightarrow \mathcal{E}$ be a cartesian monad on $\mathcal{E}$. It turns out that in many cases – most notably in the examples below – we have that $T(\mathrm{Mono}) \subseteq \mathrm{Mono}$. As a result, we are in the special case of Corollary 3.11 and so we conclude that $(\mathcal{E}_T, F_T(\mathrm{Mono}))$ is a structured display map category, and moreover we have an adjunction

$$(\mathcal{E}, \mathrm{Mono}) \underset{U_T}{\overset{F_T}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} (\mathcal{E}_T, F_T(\mathrm{Mono}))$$

in the 2-category **sDMC**. Furthermore, if the unit of the monad is monomorphic, then by Corollary 3.15 the contexts, types, and terms of the structured display map categories $(\mathcal{E}, \mathrm{Mono})$ and $(\mathcal{E}_T, F_T(\mathrm{Mono}))$ are in bijective correspondence.

▶ **Example 4.4.** Every locally cartesian closed category $\mathcal{C}$ has all pullbacks, and therefore $(\mathcal{C}, \mathrm{Arr}(\mathcal{C}))$ is always a structured display map category. Following Example 4.3, all elementary topoi are locally cartesian closed categories. In this case, we know precisely that the syntax of $(\mathcal{C}, \mathrm{Arr}(\mathcal{C}))$ is *extensional Martin-Löf type theory* [10]. This translates concretely into the fact that the internal logic of $\mathcal{C}$ is dependent type theory with Σ-types, Π-types, and extensional identity types.

Now, suppose we have a monad (T, η, μ) on $\mathcal{C}$ such that $(\mathcal{C}_T, \mathrm{Repr}(T))$ is a structured display map category. We know that a simple way to achieve this is for the monad T to be cartesian or, even better, polynomial[6].

We pointed out earlier that not all arrows of $\mathcal{C}_T$ are representable in general. As a consequence, $(\mathcal{C}_T, \mathrm{Repr}(T))$ *cannot* be of the form $(\mathcal{D}, \mathrm{Arr}(\mathcal{D}))$ since $\mathrm{Repr}(T) \neq \mathrm{Arr}(\mathcal{C}_T)$, and therefore $(\mathcal{C}_T, \mathrm{Repr}(T))$ is not the structured display map category coming from a locally cartesian closed category. Because of the biequivalence in [10], we conclude that the syntax of $(\mathcal{C}_T, \mathrm{Repr}(T))$ cannot be extensional Martin-Löf type theory, even though the contexts, types, and terms of both $(\mathcal{C}, \mathrm{Arr}(\mathcal{C}))$ and $(\mathcal{C}_T, \mathrm{Repr}(T))$ may be in bijection by Corollary 3.15.

4.1 Notions of Computation

Eugenio Moggi had the spectacular insight to provide categorical semantics for programs with computational effects using monads [22]. Given a monad T that represents a *notion of computation*, a program "with computational effects T" may be viewed as an arrow $A \longrightarrow TB$,

[5] We have in this case that the display maps contain the isomorphisms and are stable under composition. This brings us one step closer to the notion of *clan* introduced by André Joyal [20].

[6] See, *e.g.*, [17] for an illustration "in the wild" of polynomial monads over locally cartesian closed categories.

i.e., an arrow $A \mathrel{-\!\!\!\rightarrow\!\!\!\!+} B$ in the associated Kleisli category. Composition of programs works as expected, *i.e.*, by composing the arrows in the Kleisli category. We provide a table with some of the notions of computation originally found in Eugenio Moggi's paper, together with an indication of whether the monad is cartesian, whether its unit is monomorphic, and whether the monad sends monomorphisms to monomorphisms. The en dashes (–) indicate that the property is irrelevant in that case.

$T : \mathbf{Set}$	$\longrightarrow$	$\mathbf{Set}$	cartesian?	η monomorphic?	$T(\text{Mono}) \subseteq \text{Mono}$?
X	$\longmapsto$	$X + *$	yes	yes	yes
X	$\longmapsto$	$\mathcal{P}_{\text{Fin}}(X)$	no	–	–
X	$\longmapsto$	$(S \times X)^S$	no	–	–
X	$\longmapsto$	$X + E$	yes	yes	yes
X	$\longmapsto$	R^{R^X}	no	–	–
X	$\longmapsto$	$\mu A.X + A^U$	yes	yes	yes
X	$\longmapsto$	$A^* \times X$	yes	yes	yes
X	$\longmapsto$	X^A	no	–	–

Consider, for example, the *exception monad* $T_E : X \longmapsto X + E$, for which Example 3.16 is a special case. We see in the table above that the monad is cartesian, its unit is monomorphic, and it is such that $T_E(\text{Mono}) \subseteq \text{Mono}$. Moreover, **Set** is the primary example of an elementary topos, so we conclude from Example 4.3 that there is an adjunction

$$(\mathbf{Set}, \text{Mono}) \underset{U_{T_E}}{\overset{F_{T_E}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} (\mathbf{Set}_{T_E}, F_{T_E}(\text{Mono}))$$

in the 2-category **sDMC**, and F_{T_E} bijectively sends contexts, types, and terms of $(\mathbf{Set}, \text{Mono})$ to $(\mathbf{Set}_{T_E}, F_{T_E}(\text{Mono}))$.

▶ Remark 4.5. Pierre-Marie Pédrot and Nicolas Tabareau developed their own monadic approach to implementing effects in dependent type theory called the *weaning translation* [25]. They consequently defined an *exceptional translation* of the Calculus of Inductive Constructions into itself [26]. The "collapsing" result of Corollary 3.15 suggests that our approach is *not* the correct one for implementing effects in dependent type theory. Nonetheless, we would like to know how our work relates to theirs and, more specifically, how their exceptional translation compares with our framework when applied to the exception monad.

4.2 Perfect Operator Categories

Barwick introduced *operator categories* to capture the properties of various higher operads and their algebras [4]. In this setting, for example, the categories Γ^{op} and Δ^{op}, opposites of Segal's category and the simplex category, respectively, are Kleisli categories[7] for monads over so-called *perfect* operator categories. We refer to Barwick for the exact definitions of operator categories and their "perfect" version.

Every perfect operator category Φ comes with a canonical parametric right adjoint monad $T_\Phi : \Phi \longrightarrow \Phi$ [4, §5.12]. While a perfect operator category is some kind of category of finite sets with structure, its canonical monad, to quote Barwick, "adds as few points as possible in as many directions as possible". When Φ has all pullbacks, we get, for free, that $(\Lambda(\Phi), \text{Repr}(T_\Phi))$ is a structured display map category (where $\Lambda(\Phi) \coloneqq \Phi_{T_\Phi}$ is the Kleisli category for the monad T_Φ and is called the *Leinster category* by Barwick).

[7] This is reportedly an old observation of André Joyal and Ross Street.

As a result of T_Φ being a parametric right adjoint, the category $\Lambda(\Phi)$ also has a strict factorization system (see [4, Lemma 7.3], or [29, Proposition 2.6]) where the right class is exactly the class of all representable arrows (they are called *active* in this context). Note that a strict factorization system is a slightly weaker notion than an orthogonal factorization system, in that the classes are required to contain all isomorphisms in the latter case.

▶ **Example 4.6.** The most trivial example of a perfect operator category is the one-object category $*$. Its canonical monad $T_* : * \longrightarrow *$ must be the only endofunctor on $*$, *i.e.*, the identity functor. The category $*$ obviously has all pullbacks, and therefore we have that $(\Lambda(*), \mathrm{Repr}(T_*)) = (*, \mathrm{Arr}(*))$ is the "trivial" structured display map category.

▶ **Example 4.7.** Let us denote by $\mathbf{F}$ the category of finite sets. This is a perfect operator category, and the canonical monad $T_\mathbf{F} : \mathbf{F} \longrightarrow \mathbf{F}$ merely adds a point, meaning that $T_\mathbf{F}$ is nothing but the partiality monad on finite sets.

Its associated Kleisli category is Γ^{op}, the category opposite to Segal's category, and since $\mathbf{F}$ obviously has all pullbacks, we conclude that $(\Gamma^{\mathrm{op}}, \mathrm{Repr}(T_\mathbf{F}))$ is a structured display map category.

▶ **Example 4.8.** We will now see our most important example. The category $\mathbf{O}$ of totally ordered finite sets and order-preserving maps is also a perfect operator category, and the canonical monad $T_\mathbf{O} : \mathbf{O} \longrightarrow \mathbf{O}$ adds both a maximal and a minimal element to a totally ordered finite set. Its associated Kleisli category turns out to be Δ^{op}, and so we could secretly hope that Δ^{op} may be equipped with display maps in some way. At first glance, $\mathbf{O}$ does not have all pullbacks, so the pair $(\Delta^{\mathrm{op}}, \mathrm{Repr}(T_\mathbf{O}))$ fails to be a structured display map category right away. One could think that this could be rectified by noticing that Δ^{op} has no nontrivial isomorphisms, and therefore the factorization system induced by $T_\mathbf{O}$ on Δ^{op} is orthogonal on the nose. As such, we would have that the right class – namely, the representable arrows – is stable under pullbacks as is the case for every orthogonal factorization system, and therefore the pair $(\Delta^{\mathrm{op}}, \mathrm{Repr}(T_\mathbf{O}))$ would be a structured display map category. This way, we would completely bypass the need for $\mathbf{O}$ to have all pullbacks. Again, we are defeated by the basic fact that Δ^{op} does not have all pullbacks either – Δ does not have all *pushouts* – and so the representable arrows could not possibly be stable under them. The pullbacks that do exist in Δ^{op} are entirely characterized in [13, §3], and the representable arrows are notoriously stable under pullback along arrows from the left class of the factorization system [16, Lemma 2.7].

There is a solution if we consider a restricted class of arrows of $\mathrm{Repr}(T_\mathbf{O})$. The class Mono of monomorphisms of $\mathbf{O}$, which consists of all the injective maps between totally ordered finite sets, is actually stable under pullbacks. As a consequence, we have that $(\mathbf{O}, \mathrm{Mono})$ is a structured display map category. We also observe that $T_\mathbf{O}(\mathrm{Mono}) \subseteq \mathrm{Mono}$.

We can explicitly spell out the arrows in $F_{T_\mathbf{O}}(\mathrm{Mono})$. Let us denote by $\mathbf{n}$ the totally ordered finite set with n elements in $\mathbf{O}$. We have that $F_{T_\mathbf{O}}$ sends every $\mathbf{n}$ to the object $[n]$ of Δ^{op}, and an arrow $f : \mathbf{m} \longrightarrow T_\mathbf{O}(\mathbf{n})$ in $\mathbf{O}$ corresponds to an arrow $f : [m] \longleftarrow [n]$ in Δ^{op} (that is, an arrow $f : [n] \longrightarrow [m]$ in Δ). The exact translation is outside the scope of this paper. One can check, through this translation, that the class $F_{T_\mathbf{O}}(\mathrm{Mono})$ consists exactly of all the arrows $f : [m] \twoheadleftarrow [n]$ of Δ^{op} that are *surjective maps* in Δ.

Let us denote this class of surjective maps by Surj_Δ. Since $T_{\mathbf{O}}(\mathrm{Mono}) \subseteq \mathrm{Mono}$, we are in the situation of Corollary 3.11. We conclude that $(\Delta^{\mathrm{op}}, \mathrm{Surj}_\Delta)$ is a structured display map category,[8] and we have an adjunction

$$(\mathbf{O}, \mathrm{Mono}) \underset{U_{T_{\mathbf{O}}}}{\overset{F_{T_{\mathbf{O}}}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} (\Delta^{\mathrm{op}}, \mathrm{Surj}_\Delta)$$

in the 2-category **sDMC**. Finally, the unit of the monad $T_{\mathbf{O}}$ is monomorphic; therefore, $F_{T_{\mathbf{O}}}$ bijectively sends contexts, types, and terms of $(\mathbf{O}, \mathrm{Mono})$ to $(\Delta^{\mathrm{op}}, \mathrm{Surj}_\Delta)$.

There are two 2-categories of perfect operator categories $\mathbf{Op}^{\mathrm{perf}} \subseteq \mathbf{Adm}^{\mathrm{perf}}$ defined by Barwick. The category $*$ is initial in both 2-categories and terminal in $\mathbf{Adm}^{\mathrm{perf}}$, while $\mathbf{F}$ is terminal in $\mathbf{Op}^{\mathrm{perf}}$. The category $\mathbf{O}$ sits somewhere in between. It is an open problem to characterize all perfect operator categories that have all pullbacks, or whose class of monomorphisms is stable under pullbacks. Additionally, it is not known to us whether the opposite $\square^{\mathrm{op}}$ of any of the various flavors of *categories of cubes*, which are of great interest to both homotopy theorists and homotopy type theorists, is the Kleisli category associated to some perfect operator category.

4.3 Other Examples

▶ **Example 4.9.** Given an *alphabet* A, we can define the set of *words* A^* over this alphabet, *i.e.*, A^* is the *free monoid* generated by A. The assignment

$$\begin{aligned} L : \mathbf{Set} &\longrightarrow \mathbf{Set} \\ A &\longmapsto A^* \end{aligned}$$

induces a polynomial monad on **Set**, and therefore $(\mathbf{Set}_L, \mathrm{Repr}(L))$ is a structured display map category. Observe that the unit is also monomorphic. L is called the *list monad* by computer scientists.

▶ **Example 4.10.** Thomas Colcombet and Daniela Petrişan proved that *subsequential transducers* from an input alphabet A to an output alphabet B are in one-to-one correspondence with functors $\mathcal{I}_A \longrightarrow \mathbf{Set}_{T_B}$ satisfying some conditions [12, Lemma 11]. T_B is defined as the following monad

$$\begin{aligned} T_B : \mathbf{Set} &\longrightarrow \mathbf{Set} \\ X &\longmapsto B^* \times X + * \end{aligned}$$

This is the writer monad composed with the partiality monad. This monad is cartesian, and its unit is monomorphic; therefore, we can directly apply Example 4.3 and get that $(\mathbf{Set}_{T_B}, \mathrm{Repr}(T_B))$ is a structured display map category with the same contexts, types, and terms as $(\mathbf{Set}, \mathrm{Arr}(\mathbf{Set}))$.

In their article, Colcombet and Petrişan also showed that $\mathbf{Set}_{T_B}$ has an (orthogonal) factorization system (E, M) [12, Lemma 15]. One can check that the right class M is in fact exactly $F_{T_B}(\mathrm{Mono})$, and since $T_B(\mathrm{Mono}) \subseteq \mathrm{Mono}$, we have an adjunction

$$(\mathbf{Set}, \mathrm{Mono}) \underset{U_{T_B}}{\overset{F_{T_B}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} (\mathbf{Set}_{T_B}, M)$$

in the 2-category **sDMC**. Moreover, the structured display map categories $(\mathbf{Set}, \mathrm{Mono})$ and $(\mathbf{Set}_{T_B}, M)$ have the same contexts, types, and terms.

[8] It is even a display map category because Δ^{op} has no nontrivial isomorphisms.

4.4 Nonexamples

▶ **Example 4.11.** The free *commutative* monoid monad, in contrast with Example 4.9, is *not* cartesian for deep reasons that are detailed in [11, Proposition 5.1].

▶ **Example 4.12.** There is no hope for the powerset monad $\mathcal{P} : X \longmapsto \mathcal{P}(X)$ and its variants to be such that $(\mathbf{Set}_{\mathcal{P}}, \mathrm{Repr}(\mathcal{P}))$ is a structured display map category. One can check, using the pullback of $\mathcal{P}(\{0,1\}) \longrightarrow \mathcal{P}(\{0\})$ along itself, that this is indeed impossible.

5 Conclusion and Future Directions

The final example, Example 4.12, seems to indicate that the approach presented in this paper is *not* the correct one for **Prof**, as the relative pseudomonad $\mathcal{C} \longmapsto \widehat{\mathcal{C}}$ may be seen as a "categorification" of the powerset monad. Most critically, an obstruction at the level of sets surely carries over to the categorification through the canonical embedding $\mathbf{Set} \longhookrightarrow \mathbf{Cat}$.

There is still more work to be done; for example, we could focus on a simple monad like the exception monad $T_E : X \longmapsto X + E$ and see whether we can extract a syntax from $(\mathbf{Set}_{T_E}, F_{T_E}(\mathcal{A}))$ for different choices of $\mathcal{A} \subseteq \mathrm{Arr}(\mathbf{Set})$. Then we can compare the result with the exceptional translation of Pierre-Marie Pédrot and Nicolas Tabareau (*cf.* Remark 4.5). We also mentioned in Example 4.4 that $(\mathcal{C}_T, \mathrm{Repr}(T))$ cannot be a model of extensional Martin-Löf type theory in general. This entails that the basic type constructors in $\mathcal{C}_T$, whatever they may be, must be noticeably different from those in extensional theory in a way that may be worth investigating.

References

1 Benedikt Ahrens, Peter LeFanu Lumsdaine, and Paige Randall North. Comparing Semantic Frameworks for Dependently-Sorted Algebraic Theories. In Oleg Kiselyov, editor, *Programming Languages and Systems*, pages 3–22. Springer Nature, 2025. doi:10.1007/978-981-97-8943-6_1.

2 Benedikt Ahrens, Paige Randall North, and Niels van der Weide. Bicategorical type theory: Semantics and syntax. *Mathematical Structures in Computer Science*, 33(10):868–912, 2023. doi:10.1017/S0960129523000312.

3 Thosten Altenkirch, James Chapman, and Tarmo Uustalu. Monads need not be endofunctors. *Logical Methods in Computer Science*, Volume 11, Issue 1, 2015. doi:10.2168/LMCS-11(1:3)2015.

4 Clark Barwick. From operator categories to higher operads. *Geometry & Topology*, 22(4):1893–1959, April 2018. doi:10.2140/gt.2018.22.1893.

5 Jean Bénabou. Distributors at Work, 2000.

6 Francis Borceux. *Handbook of Categorical Algebra: Volume 2: Categories and Structures*, volume 2 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, 1994. doi:10.1017/CBO9780511525865.

7 Dominique Bourn. Kleisli categories, T-categories and internal categories. *Theory and Applications of Categories*, 41(34):1108–1159, 2024.

8 Albert Burroni. T-catégories (catégories dans un triple). *Cahiers de topologie et géométrie différentielle*, 12(3):215–321, 1971.

9 John Cartmell. Generalised algebraic theories and contextual categories. *Annals of Pure and Applied Logic*, 32:209–243, 1986. doi:10.1016/0168-0072(86)90053-9.

10 Pierre Clairambault and Peter Dybjer. The biequivalence of locally cartesian closed categories and Martin-Löf type theories. *Mathematical Structures in Computer Science*, 24(6):e240606, 2014. doi:10.1017/S0960129513000881.

11 Maria Manuel Clementino, Dirk Hofmann, and George Janelidze. The monads of classical algebra are seldom weakly cartesian. *Journal of Homotopy and Related Structures*, 9(1):175–197, 2014. doi:10.1007/s40062-013-0063-2.

12 Thomas Colcombet and Daniela Petrişan. Automata Minimization: A Functorial Approach. *Logical Methods in Computer Science*, Volume 16, Issue 1, 2020. doi:10.23638/LMCS-16(1:32)2020.

13 Carmen Constantin, Tobias Fritz, Paolo Perrone, and Brandon T. Shapiro. Weak cartesian properties of simplicial sets. *Journal of Homotopy and Related Structures*, 18(4):477–520, 2023. doi:10.1007/s40062-023-00334-1.

14 Peter Dybjer. Internal type theory. In Stefano Berardi and Mario Coppo, editors, *Types for Proofs and Programs*, pages 120–134. Springer, 1996. doi:10.1007/3-540-61780-9_66.

15 M. Fiore, N. Gambino, M. Hyland, and G. Winskel. Relative pseudomonads, Kleisli bicategories, and substitution monoidal structures. *Selecta Mathematica*, 24(3):2791–2830, 2018. doi:10.1007/s00029-017-0361-3.

16 Imma Gálvez-Carrillo, Joachim Kock, and Andrew Tonks. Decomposition spaces, incidence algebras and Möbius inversion I: Basic theory. *Advances in Mathematics*, 331:952–1015, 2018. doi:10.1016/j.aim.2018.03.016.

17 Nicola Gambino and Joachim Kock. Polynomial functors and polynomial monads. *Mathematical Proceedings of the Cambridge Philosophical Society*, 154(1):153–192, 2013. doi:10.1017/S0305004112000394.

18 Martin Hofmann. Syntax and semantics of dependent types. In Martin Hofmann, editor, *Extensional Constructs in Intensional Type Theory*, pages 13–54. Springer, 1997. doi:10.1007/978-1-4471-0963-1_2.

19 Bart Jacobs. Comprehension categories and the semantics of type dependency. *Theoretical Computer Science*, 107(2):169–207, 1993. doi:10.1016/0304-3975(93)90169-T.

20 Andre Joyal. Notes on Clans and Tribes, 2017. doi:10.48550/arXiv.1710.10238.

21 Saunders Mac Lane. *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics*. Springer, 2nd. ed., softcover version of original hardcover edition 1998 edition, 1978. doi:10.1007/978-1-4757-4721-8.

22 Eugenio Moggi. Notions of computation and monads. *Information and Computation*, 93(1):55–92, 1991. doi:10.1016/0890-5401(91)90052-4.

23 Hayato Nasu. An Internal Logic of Virtual Double Categories, 2025. doi:10.48550/arXiv.2410.06792.

24 Max S. New and Daniel R. Licata. A Formal Logic for Formal Category Theory. In Orna Kupferman and Pawel Sobocinski, editors, *Foundations of Software Science and Computation Structures*, pages 113–134, 2023. doi:10.1007/978-3-031-30829-1_6.

25 Pierre-Marie Pédrot and Nicolas Tabareau. An effectful way to eliminate addiction to dependence. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12, 2017. doi:10.1109/LICS.2017.8005113.

26 Pierre-Marie Pédrot and Nicolas Tabareau. Failure is Not an Option. In Amal Ahmed, editor, *Programming Languages and Systems*, pages 245–271. Springer International Publishing, 2018. doi:10.1007/978-3-319-89884-1_9.

27 Emily Riehl. *Category Theory in Context*. Aurora Dover Modern Math Originals. Dover Publications, Inc, 2016.

28 Jenö Szigeti. On limits and colimits in the Kleisli category. *Cahiers de topologie et géométrie différentielle*, 24(4):381–391, 1983.

29 Mark Weber. Familial 2-functors and parametric right adjoints. *Theory and Applications of Categories*, 18(22):665–732, 2007.

30 R. J. Wood. Abstract pro arrows I. *Cahiers de topologie et géométrie différentielle*, 23(3):279–290, 1982.