Mathematical Structures in Computer Science (2019), **29**, pp. 1513–1555
doi:10.1017/S0960129519000070

CAMBRIDGE UNIVERSITY PRESS

PAPER

Categorical models of the differential λ-calculus

Robin Cockett* and Jonathan Gallagher†

Department of Computer Science, University of Calgary, Calgary, Canada
*Corresponding author. Email: robin@ucalgary.ca

(Received 7 July 2016; revised 27 November 2018; accepted 4 April 2019; first published online 6 June 2019)

Abstract

The paper shows how the Scott–Koymans theorem for the untyped λ-calculus can be extended to the differential λ-calculus. The main result is that every model of the untyped differential λ-calculus may be viewed as a differential reflexive object in a Cartesian-closed differential category. This extension of the Scott–Koymans theorem depends critically on unraveling the somewhat subtle issue of which idempotents can be split so that differential structure lifts to the idempotent splitting.

The paper uses (total) Turing categories with "canonical codes" as the basic categorical semantics for the λ-calculus. It develops the main result in a modular fashion by showing how to add left-additive structure to a Turing category, and then – on top of that – differential structure. For both levels of structure, it is necessary to identify how "canonical codes" must behave with respect to the added structure and, furthermore, how "universal objects" must behave. The latter is closely tied to the question – which is the crux of the paper – of which idempotents can be split while preserving the differential structure of the setting.

This paper is the full version of a conference paper and includes the proofs which were omitted from that version due to page-length restrictions.

Keywords: λ calculus, differential λ calculus, differential categories

1. Introduction

In Ehrhard and Regnier (2003), the differential λ-calculus was introduced in order to give a syntactic counterpart for the semantic models of linear logic which had already been observed to have a differential (Ehrhard 2002, 2005). In these models, proofs were interpreted as differentiable maps with the linear maps, in the sense of linear logic, becoming – rather elegantly – the maps which were linear in the differential sense. The differential λ-calculus, besides giving a syntax for differentiation, introduced a new aspect as, being an untyped system with a confluent rewriting system, it contained the untyped λ-calculus, and, therefore, provided a model of computability. Furthermore, it was immediately apparent that this calculus was very closely related to the resource λ-calculus (Boudol 1993; Boudol et al. 1999; Boudol and Laneve 1995). This confluence of developments, thus, initiated the study of models of computability in which computable functions could be differentiated.

The categorical semantics for these settings was developed in series of papers. Initially the tensorial (or linear logic) side of the story was developed in Blute et al. (2006): this very much followed the path of Ehrhard's more model-driven development. Tensor differential categories (also written ⊗-differential categories) came equipped with a coalgebra modality, and it was

†This work was partially supported by National Science and Engineering Research Council (NSERC) of Canada.

© Cambridge University Press 2019

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

understood that the differential λ-calculus would then be interpreted in the coKleisli category for this modality.

The next step in this development involved Cartesian differential categories (also written ×-differential categories) Blute et al. (2009). In Bucciarelli et al. (2010) not only was the appropriate notion of being Cartesian closed for Cartesian differential categories developed, but also a sound and complete interpretation of the simply typed differential λ-calculus into Cartesian-closed differential categories was provided. This, furthermore, clarified the relationship between the resource λ-calculus and the differential λ-calculus and suggested that they both have their semantics in Cartesian-closed differential categories.

It remained, however, to provide a precise account of the semantics of the untyped differential λ-calculus. At this stage, it was abundantly clear that the interpretation of the untyped differential calculus should be into some sort of reflexive object in a Cartesian-closed differential category, and thus, there should be an analogue of the Scott–Koymans[1] theorem (Koymans 1982; Scott 1980) which says that every λ-theory is equivalent to the theory of a reflexive object.

In Manzonetto (2012), the investigation of models of the untyped differential λ-calculus was initiated. Manzonetto showed that linear reflexive objects in a Cartesian differential category soundly interpret the untyped differential λ-calculus. Furthermore, he gave a completeness theorem, for the differential λ-calculus, however, with two additional equations:

$$\lambda x.(Ds \cdot t)x = Ds \cdot t \qquad \text{and} \qquad a + a = a.$$

The first equation, called "differential extensionality", is an artifact of the syntax which was used in that work for the differential λ-calculus. Here we shall use a slightly different syntax for which this issue does not arise. An analysis of the equivalence between the syntax used here and the original syntax for the differential λ-calculus is in Gallagher (2018).

The second equation is, however, a very significant restriction on the type of models Manzonetto was considering. In fact, this meant that he did not provided a general completeness theorem which truly paralleled the Scott–Koymans theorem. This paper, thus, revisits this problem and shows that by subtly changing the requirements on the idempotents which one splits one can indeed obtain a complete analogue of the Scott–Koymans theorem: this means that all models of the untyped differential λ-calculus do, indeed, arise as reflexive objects in Cartesian-closed differential categories. In Manzonetto's work, it was assumed – very reasonably – that the idempotents one should split had to be linear. Unfortunately, this meant that, as Curry's pairing combinator, $(a, b) \mapsto \lambda p.p\ a\ b$, is not linear and one is forced to look for an alternative idempotent which can encode pairing. Manzonetto actually found such a combinator: it was the more complicated $(a, b) \mapsto \lambda y.(a + Dy \cdot b)$. However, to make this combinator behave correctly, he paid a heavy price and had to restrict to the case where addition was idempotent.

To access the more general situation, which is the subject of this paper, it is necessary to understand more fully the behavior of idempotents in a Cartesian differential category. To achieve this one not only has to understand their behavior with respect to differentiation but also with respect to the left-additive structure. We shall, therefore, redevelop this program by describing the categorical semantics not only of models of the differential λ-calculus but also of models of the "additive" λ-calculus. This is because moving directly from a Cartesian-closed differential setting to a model of the differential λ-calculus, without this intermediate step, tends to obscure the subtle behavior required of the idempotents which must be split. In fact, in this paper we shall start the development even further back: we actually redevelop the Scott–Koymans theorem itself from the more modern and unifying perspective of Turing categories (Cockett and Hofstra 2008). One justification for this is that, in the theory of Turing categories, idempotents and idempotent completions play a very central role. This predisposes one to take the behavior of idempotents very seriously as one begins to add structure.

Because our aim is to model the λ-calculus, we shall exclusively focus on Turing categories all of whose maps are total: these were introduced by Longo and Moggi (1990) – albeit under a different

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

name – and their realizability theory was developed by Birkedal (1999). Such a Turing category is a weakly Cartesian-closed category with an object, T, called a Turing object which is universal. An object in a category is *universal* in case all objects are retracts of it. In order to model the untyped λ-calculus, with β-equality, we shall employ the notion of a Turing category with canonical codes. This allows us to interpret the term calculus of the untyped λ-calculus very directly into these categorial models. This step does not involve consideration of idempotents. This approach, in fact, closely parallels the approach followed in Koymans (1982), and allows a convenient separation of concerns which is useful later. To recapture the Scott–Koymans theorem, one observes that splitting idempotents yields a Cartesian-closed category in which the Turing object is still universal: this immediately makes the Turing object a reflexive object.

The approach that we follow successively adds structure in a modular fashion. First we add left-additive structure and develop the categorical semantics of the untyped additive λ-calculus and then, following the same pattern, arrive at the categorical semantics of the untyped differential λ-calculus. At each stage, we must appropriately strengthen both the notion of being weakly Cartesian closed (with canonical codes) and of being a universal object. A universal object must have every other object as a retract and, thus, as an idempotent splitting: as the idempotents which can be split change, the definition of a universal object must also change.

The payoff of this reconstruction is that it allows a general analogue for the Scott–Koymans theorem for untyped differential λ-models.

2. The λ-Calculus

The completely type-free logic is a will-O-the-wisp. This has been shown time and time again; yet the vision remains. And formal theories are quite as bad as drugs in keeping such illusions alive. The cold light of day reveals, however, perfectly solid ground.

Dana Scott

The classical notion of a *categorical* model of the untyped λ-calculus is as a reflexive object in a Cartesian-closed category. This is an object U which has as a retract its object of endomorphisms, $[U, U] \lhd^s_r U$.[2] One way to approach the Scott–Koymans completeness result is to interpret the λ-calculus across this retraction: this is rather messy as one must repeatedly use the section and retraction to interpret terms. An alternative approach, which is, in fact, closer to the approach taken by Koymans, is to interpret the λ-calculus into a Turing category with canonical codes and then show that splitting the idempotents results in a Cartesian-closed category in which the Turing object is still universal and thus is a reflexive object. This approach gives an elegant separation of the two concerns (interpretation and idempotent splitting) and, furthermore, allows the introduction of a more direct categorical counterpart to the untyped λ-calculus: namely, Turing categories with canonical codes. This is the approach we shall follow.

2.1 Syntax of the λ-calculus

The term formation rules for a Cartesian theory or a many-sorted equational theory, Table 1, relies of a supply atomic types $A \in \mathcal{A}$ and of typed function symbols $f \in \Omega(A_1, \ldots, A_n; B)$. Terms are then built in context: we insist that all the variables declared in a context are distinct. Furthermore, we only allow the declaration of variables of atomic type. We do, however, allow the formation of variable patterns for Cartesian products of atomic types. This allows a smooth transition to the categorical semantics as we can turn any sequent into a map, i.e. a sequent with one premise

$$p{:}X \vdash t{:}T$$

by simply collecting the context into one type using pair pattern introduction (the "Pair Pattern" rule in Table 1). This does introduce a further complication: one has to be able to substitute patterns. When substituting a pattern, one does a parallel substitution of the variables in

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Table 1. Term formation rules for a Cartesian theory

$$\frac{\{\Gamma \vdash t_i\}_{i=1,\ldots,n} \quad f \in \Omega(A_1,\ldots,A_n;B)}{\Gamma \vdash f(t_1,\ldots,t_n) : B} \text{ Function application}$$

$$\frac{}{\Gamma, x : A, \Gamma' \vdash x : A} \text{ Projection}$$

$$\frac{\Gamma, \Gamma' \vdash s : X \quad \Gamma, p : X, \Gamma' \vdash t : Z}{\Gamma, \Gamma' \vdash t[s/p] : Z} \text{ Cut/substitution}$$

Product rules:

$$\frac{\Gamma, \Gamma' \vdash t : X}{\Gamma, (): 1, \Gamma' \vdash t : X} \text{ Unit pattern} \qquad \frac{\Gamma, x : X, y : Y, \Gamma' \vdash t : Z}{\Gamma, (x,y) : X \times Y, \Gamma' \vdash t : Z} \text{ Pair pattern}$$

$$\frac{\Gamma \vdash t : X \quad \Gamma \vdash s : Y}{\Gamma \vdash (t,s) : X \times Y} \text{ Pairing} \qquad \frac{}{\Gamma \vdash () : 1} \text{ Unit pairing}$$

Table 2. Equalities for a Cartesian theory

$$\frac{t_1 =_\Gamma t_2 : X \in E}{\Gamma \vdash t_1 = t_2 : X}$$

$$\frac{\Gamma \vdash t : X}{\Gamma \vdash t = t : X} \text{ refl} \qquad \frac{\Gamma \vdash t_1 = t_2 : X}{\Gamma \vdash t_2 = t_1 : X} \text{ sym}$$

$$\frac{\Gamma \vdash t_1 = t_2 : X \quad \Gamma \vdash t_2 = t_3 : X}{\Gamma \vdash t_1 = t_3 : X} \text{ trans}$$

$$\frac{\Gamma \vdash s_1 = s_2 : X \quad \Gamma, x : X \vdash t_1 = t_2 : Y}{\Gamma \vdash t_1[s_1/x] = t_2[s_2/x] : Y} \text{ Substitution}$$

the pattern: thus, e.g., the substitution $[(t,s)/(x,y)]$ becomes, equivalently, the parallel substitution $[t/x, s/y]$. The type system ensures that the term and pattern in a substitution have exactly the same form.

By removing the product type and all the product pattern rules and tupling rules, one obtains a "pure" equational theory: this is perhaps the more usual way to present equational logic. This pure logic allows one to form exactly the same (inessential cut eliminated) terms for all the atomic types. Thus, strictly speaking the product rules are unnecessary: their significant benefit, however, is that having the product rules aligns the logic more closely to the categorical semantics and this is the main reason for entertaining them throughout this development.

One can also associate arbitrary equations, E, with a Cartesian theory. Equations are of the form $t_1 =_\Gamma t_2 : X$, where t_1 and t_2 are of type X and can be formed in the context Γ. The rules for generating equalities in a Cartesian theory are given in Table 2. The last rule for substitution of one equality in another ensures that the equalities generated form a congruence.

The syntax of the (untyped) λ-calculus may be described as "unityped" terms in context. This means we allow all the above rules for a Cartesian theory, Table 1, but we have only one atomic

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Table 3. Term formation rules and equality for the λ-calculus

$$\frac{\Gamma \vdash m : U \quad \Gamma \vdash n : U}{\Gamma \vdash mn : U}\ \text{Application} \qquad \frac{\Gamma, x : U, \Gamma' \vdash m : U}{\Gamma, \Gamma' \vdash \lambda x.m : U}\ \text{Abstraction}$$

λ-Equalities

$$\frac{\Gamma \vdash n : U \quad \Gamma, x : X \vdash m : U}{\Gamma \vdash (\lambda x.m)n = m[n/x] : U}\ \beta\text{-equality}$$

$$\frac{\Gamma, y : Y, x : X \vdash m : U \quad \Gamma \vdash n : Y}{\Gamma \vdash (\lambda x.m)[n/y] = \lambda x.(m[n/y]) : U}\ \lambda\text{-substitution}$$

$$\frac{\Gamma, x : X \vdash t_1 = t_2 : U}{\Gamma \vdash \lambda x.t_1 = \lambda x.t_2 : U}\ \lambda\text{-congruence } (\xi\text{-rule})$$

type U (so $\mathcal{A} = \{U\}$). The term formation rules consist of the term formation rules for a Cartesian theory, as in Table 1, and the special terms for the λ-calculus. In addition, we must extend the notion of equality. Not only can E now contain λ-terms, but also there are the equalities between λ-terms. This is collected into Table 3.

Besides β-equality, $(\lambda x.m)n = m[n/x]$, the ξ-rule which allows the use of equalities below abstractions, and the λ-substitution[3] rule, $(\lambda x.m)[n/y] = \lambda x.(m[n/y])$, we allow – as usual – α-conversion of bound variables

$$\lambda x.m \equiv_\alpha \lambda y.(m[y/x]),$$

which we regard as being a given with any binding syntax. We can then generate the theory $\Lambda_\beta(\Omega, E)$ by forming the smallest congruence on terms that contains the above equations. There is an associated category of this theory, $\mathcal{C}(\Lambda_\beta(\Omega, E))$, called the *classifying category* of $\Lambda_\beta(\Omega, E)$:

Objects: Words in $U, \times, 1$.
Arrows: A map $T \longrightarrow S$ corresponds to a sequent with one type on the left hand side

$$p : T \vdash m : S$$

under the term equivalence of $\Lambda_\beta(\Omega, E)$.
Composition: Composition is substitution:

$$\frac{p : X \vdash m : Y \qquad p' : Y \vdash n : Z}{p : X \vdash n[m/p'] : Z}$$

We note, rather modestly, that this is a Cartesian category.

Proposition 2.1. *$\mathcal{C}(\Lambda_\beta(\Omega, E))$ is a Cartesian category.*

The rest of this section develops the properties of this category.

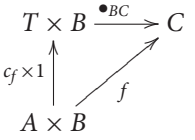
2.2 Total Turing categories with canonical codes

A **(total) Turing category** $\mathbb{X}$ is a Cartesian category with the following:

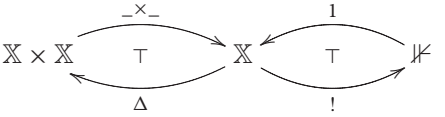
https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

[T.1] An object T and for each pair of objects B, C a map $T \times B \xrightarrow{\bullet_{BC}} C$;

[T.2] For each map $A \times B \xrightarrow{f} C$ a map $A \xrightarrow{c_f} T$ such that



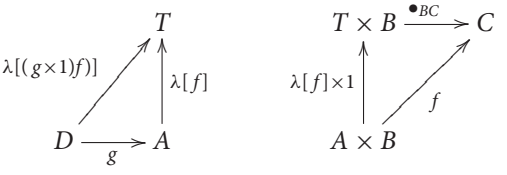
In this development, as we are trying to align our categories to type theoretic syntax, we shall assume throughout that being Cartesian implies that there is a functorially chosen product structure:



A Turing category has **canonical codes** when there is a function

$$\mathbb{X}(A \times B, C) \xrightarrow{\lambda()} \mathbb{X}(A, T)$$

such that



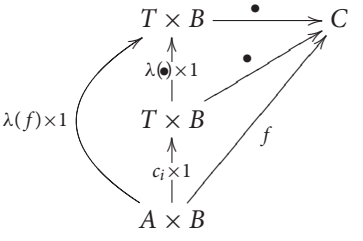
commute. This means that the function, $\lambda[_]$, for forming codes has a substitutional property, namely that $\lambda[(g \times 1)f] = g\lambda[f]$ and that $\lambda[f]$ is indeed a code for f in the sense that $(\lambda(f) \times 1)\bullet_{BC} = f$.

Note that a given Turing category can have many different families of functions which provide canonical codes. At this stage to say that a Turing category has canonical codes is merely to assert that there is such a family of functions $\lambda : \mathbb{X}(A \times B, C) \to \mathbb{X}(A, T)$ for each A, B, and C.

Observe first that this property of having canonical codes can be simplified:

Lemma 2.1. *A Turing category $\mathbb{X}$ has canonical codes if and only if there is a code $\lambda[\ \bullet_{BC}\ ]$ for each $\bullet_{BC} : T \times B \to C$ such that whenever $(c_1 \times 1)\bullet_{BC} = f = (c_2 \times 1)\bullet_{BC}$, i.e. c_1 and c_2 are codes for f, then $c_1\lambda[\ \bullet_{BC}\ ] = c_2\lambda[\ \bullet_{BC}\ ]$.*

Proof. Suppose $\mathbb{X}$ has canonical codes then



https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

so that $(c_1 \times 1)\lambda[\bullet] = \lambda[f] = (c_2 \times 1)\lambda[\bullet]$. Conversely defining $\lambda[f] := c_f\lambda[\bullet]$, where c_f is any code for f, then

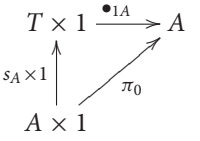
$$\lambda[(g \times 1)f] = \lambda[((gc_f) \times 1) \bullet] := gc_f\lambda[\bullet] =: g\lambda[(c_f \times 1) \bullet] = g\lambda[f].$$

□

The following is an economical "recognition" theorem for Turing categories with canonical codes.

Lemma 2.2. *To have a Turing category with canonical codes is to have a Cartesian category with a universal object T and a map $T \times T \xrightarrow{\bullet} T$ which has canonical codes for all maps $T \times T \to T$.*

Proof. To see that the Turing object T in a Turing category is universal, consider a code s_A for $A \times 1 \xrightarrow{\pi_0} A$:

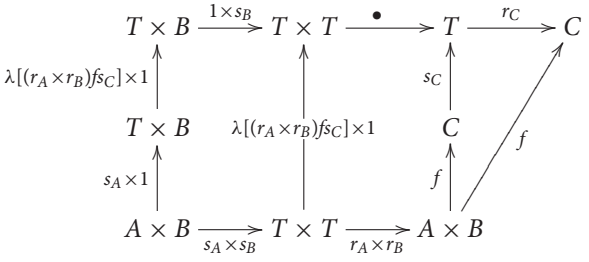


then clearly s is a section with retraction $\langle 1, ! \rangle \bullet_{1A}$.

Conversely to show such a category is a Turing category, we will choose for each object X a section and retraction $X \lhd^{s_X}_{r_X} T$ then we can define a derived application

$$\bullet_{BC} := T \times B \xrightarrow{1 \times s_B} T \times T \xrightarrow{\bullet} T \xrightarrow{r_C} C$$

for each B and C. The canonical code for a map $A \times B \xrightarrow{f} C$ is then defined to be $\lambda[f] := s_A\lambda[(r_A \times r_B)fs_C]$:



To show it is substitutive, consider $g\lambda[f]$, where $g : D \to A$, then $g = s_D r_D g$ so

$$\begin{aligned} g\lambda[f] &:= gs_A\lambda[(r_A \times r_B)fs_C)] = s_D r_D g s_A \lambda[(r_A \times r_B)fs_C)] \\ &= s_D\lambda[((r_D g s_A) \times 1)(r_A \times r_B)fs_C] = s_D\lambda[(r_D \times r_B)(g \times 1)fs_c] \\ &=: \lambda[(g \times 1)f]. \end{aligned}$$

□

Note that the constructed canonical codes rely heavily on the *choice* how each object is viewed as a retract of the Turing object.

We use this to show the following.

Theorem 2.1. *$\mathcal{C}(\Lambda_\beta(\Omega, E))$ is a Turing category with canonical codes.*

Proof. The universal map is $U \times U \xrightarrow{\bullet} U; (m, n) \vdash mn$. By Lemma 2.2, it suffices to show that $\bullet$ has canonical codes for terms $U \times U \to U; (x, y) \vdash t$. Define $\lambda[(x, y) \vdash t] := x \vdash \lambda y.t$, then

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

$$(\lambda([(x,y)\vdash t]\times 1)\bullet = [(x,y)\vdash(\lambda y.t,y)]\bullet$$
$$= [(x,y)\vdash(\lambda y.t)y] = [(x,y)\vdash t]$$

so that $\lambda[(x,y)\vdash t]$ is a code for $[(x,y)\vdash t]$. To show this is a canonical code consider

$$[z\vdash s]\lambda[(x,y)\vdash t] = [z\vdash s][x\vdash\lambda y.t]$$
$$= [z\vdash(\lambda y.t)[s/x]] = [z\vdash(\lambda y.t[s/x])]$$
$$= \lambda[(z,y)\vdash t[s/x]].$$

It remains to show that every object is a retract of U. To see that the terminal object, 1, is a retract of U, it suffices to observe that U has an element: let c be any closed λ-term (e.g. c could equal $\lambda x.x$), then $[()\vdash c]:1\to U$ is an element, and the unique map, $[u\vdash()]:U\to 1$, into the terminal object acts as a retraction. To see that $U\times U\lhd_r^s U$, take s to be $(a,b)\vdash\lambda p.p\ a\ b$, and for the retraction take $z\vdash(z(\lambda xy.x),z(\lambda xy.y))$ (note these are Curry's combinators which were mentioned in the introduction). Once one has the retractions for 1 and $U\times U$ it is easy to generate inductively sections and retractions or any word in $U,\times,1$. □

From a logical perspective, this theorem proves the *completeness* of the untyped λ-calculus with respect to models which are Turing categories with canonical codes. With some more work (some of which is developed in the following), one can exhibit this as part of an adjunction between λ-theories (with function symbols and equations) and Turing categories with canonical codes.

2.3 Interpreting the λ-calculus

We now show that Turing categories with canonical codes are sound models for the untyped λ-calculus. We do this by showing that, for any Turing category $\mathbb{X}$ with canonical codes, given an interpretation, $\mathcal{I}$, of the function symbols of an untyped λ-calculus, $\Lambda_\beta(\Omega,E)$, such that the equations are satisfied, there is a canonical functor

$$[\![_]\!]_{\mathcal{I}}:\mathcal{C}(\Lambda_\beta(\Omega,E))\to\mathbb{X}$$

which carries the universal object onto the Turing object and application onto the Turing morphism $\bullet:T\times T\to T$.

Importantly, a Turing category may have more than one possible Turing object and more than one Turing morphism: one only needs the existence of such structure to be a Turing category. However, in this development we need to strengthen the notion of a Turing category not only to have chosen products but also to have a chosen Turing morphism $\bullet:T\times T\to T$. Furthermore, for a Turing category with canonical codes the function which supplies the canonical codes $\lambda:\mathbb{X}(T\times T,T)\to\mathbb{X}(T,T)$ will be part of this structure. Note that, because every object X is a retract of the Turing object T, fixing this canonical code structure also determines the canonical code structure $\lambda:\mathbb{X}(X\times T,T)\to\mathbb{X}(X,T)$ for every object X: we will refer to these codes as the *basic canonical codes*. Thus, the functors between Turing categories with canonical codes, which we consider here, will all preserve products, the Turing map, and these basic canonical codes on the nose.

With this understanding, let $\mathbb{X}$ be a Turing category with canonical codes, whose Turing object is T and Turing morphism is $T\times T\xrightarrow{\bullet}T$. Given an interpretation $\mathcal{I}$ of Ω which satisfies the equations, we obtain the interpretation of the $\Lambda_\beta(\Omega,E)$, $[\![_]\!]_{\mathcal{I}}:\mathcal{C}(\Lambda_\beta(\Omega,E))\to\mathbb{X}$, as follows:

Objects: On objects define $[\![X]\!]$ to be $X[T/U]$.

Maps: On arrows:

$$[\![p:X\vdash():1]\!] = !_{[\![X]\!]}$$
$$[\![p:X\vdash(m,n):Y\times Z]\!] = \langle[\![p:X\vdash m:Y]\!],[\![p:X\vdash n:Z]\!]\rangle$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

$$\begin{aligned}
[\![x : U \vdash x : U]\!] &= 1_T \\
[\![(p_1, p_2) : X_1 \times X_2 \vdash x : U]\!] &= \pi_i [\![p_i : X_i \vdash x : U]\!] \quad \text{where } x \in p_i \\
[\![p : X \vdash mn : U]\!] &= \langle [\![p : X \vdash m : U]\!], [\![p : X \vdash n : U]\!] \rangle \bullet \\
[\![p : X \vdash \lambda z.m : U]\!] &= \lambda \left[[\![(p, z) : X \times U \vdash m : U]\!] \right] \\
[\![p : X \vdash f(t_1, \ldots, t_n) : U]\!] &= \langle [\![p : X \vdash t_1]\!], \ldots, [\![p : X \vdash t_n]\!] \rangle \mathcal{I}(f),
\end{aligned}$$

where in the last translation, we must choose a standard association for tuples (as we only have binary pairing).

We then have the following.

Theorem 2.2. *If $\mathbb{X}$ is a Turing category with canonical codes and $\mathcal{I}$ is an interpretation of the function symbols Ω which satisfies the equations E, then*

$$[\![_]\!]_{\mathcal{I}} : \mathcal{C}(\Lambda_\beta(\Omega, E) \to \mathbb{X}$$

is a Cartesian functor which preserves the Turing object, the Turing map, and the basic canonical codes.

Proof. First, for $[\![_]\!]$ to be a functor, it must preserve composition in the sense that $[\![[: X \vdash m[n/p'] : U]\!]$ must be $[\![p : X \vdash n : Y]\!][\![p' \vdash m : U]\!]$. This is often called the substitution lemma; the proof is by induction on the structure of m. We consider the case where m is an abstraction:

$$\begin{aligned}
[\![p{:}X \vdash (\lambda y.m)[n/p']{:}U]\!] &= [\![p : X \vdash \lambda y.(m[n/p']) : U]\!] \\
&= \lambda \left[[\![(p, y){:}X \times U \vdash m[n/p'] : U]\!] \right] \\
&= \lambda \left[([\![p : X \vdash n{:}Y]\!] \times 1) [\![(p', y){:}Y \times U \vdash m{:}U]\!] \right] \\
&= [\![p : X \vdash n : Y]\!] \lambda \left[[\![(p', y) : Y \times U \vdash m : U]\!] \right) \\
&= [\![p : X \vdash n : Y]\!] [\![p' : Y \vdash \lambda y.m : U]\!].
\end{aligned}$$

In order for $[\![_]\!]$ to be a functor, it must be well-defined: this amounts to showing that equal terms are carried to equal maps; i.e. whenever $m = n$, then $[\![p : X \vdash m : U]\!] = [\![p : X \vdash n : U]\!]$. Note that while it is more natural to state the above result after stating well-definedness, the following proof relies on the result above. The interesting cases for well-definedness are β equality and congruence under λ:

$$\begin{aligned}
[\![p : X \vdash (\lambda x.m)n : U]\!] &= \langle [\![p : X \vdash \lambda x.m : U]\!], [\![p : X \vdash n : U]\!] \rangle \bullet \\
&= \langle \lambda \left[[\![(p, x) : X \times U \vdash m : U]\!] \right], [\![p : X \vdash n : U]\!] \rangle \bullet \\
&= \langle 1, [\![p : X \vdash n : U]\!] \rangle [\![(p, x) : X \times U \vdash m : U]\!] \\
&= [\![p : X \vdash m[n/x]]\!] \quad \text{(substitution is composition)}, \\
[\![p : X \vdash \lambda x.m : U]\!] &= \lambda \left[[\![(p, x) : X \times U \vdash m : U]\!] \right] \\
&= \lambda \left[[\![(p, x) : X \times U \vdash n : U]\!] \right] \quad \text{(assumption)} \\
&= [\![p : X \vdash \lambda x.n : U]\!].
\end{aligned}$$

The interpretation $\mathcal{I}$ of the function symbols satisfies E in case whenever $t_1 =_\Gamma t_2$ is an equation then pairing up the variable in Γ (in any manner) to form $p : X$ has $[\![p : X \vdash t_1]\!]_{\mathcal{I}} = [\![p : X \vdash t_2]\!]_{\mathcal{I}}$. However, demanding this equality will mean all the equations generated by E will also be satisfied assuming the rules for term equality. □

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

An interpretation of a theory is often conflated with the notion of a model. However, for a λ-theory we shall reserve the term model to mean an interpretation into Turing category with canonical codes whose objects are Cartesian product words in the Turing object 1, T, $T \times T$, $T \times (T \times T)$, ... Given any Turing category with chosen products, it is clear that one can always take the full subcategory determined by these objects and this will supply a model in this sense. Note, in particular, that the classifying category of a λ-theory is always a model in this sense. Furthermore, as we allow arbitrary equations in λ-models, every model is actually equivalent to the classifying category of a λ-theory.

Notice that an interpretation *the* untyped λ-calculus is just an interpretation of the λ-theory with no added symbols or equations, $\Lambda_\beta(\emptyset, \emptyset)$. This immediately means that its classifying category is the initial Turing category with canonical codes.

2.4 Reflexive objects

An important observation for Turing categories is that the idempotent splitting of a Turing category $\mathbb{X}$, $\mathsf{Split}(\mathbb{X})$, is again a Turing category; see Cockett and Hofstra (2008). Furthermore, the Turing object and the Turing morphism are preserved in this splitting. Our next objective is to show that splitting the idempotents of a Turing category which has canonical codes results not only a Turing category with canonical codes but a Cartesian-closed Turing category. As the Turing object is still universal in the idempotent splitting, this, in particular, means that it becomes a reflexive object in idempotent splitting.

Theorem 2.3. *When $\mathbb{X}$ is a Turing category with canonical codes, then $\mathsf{Split}(\mathbb{X})$ is a Cartesian-closed Turing category with canonical codes.*

Proof. The idempotent splitting of $\mathbb{X}$ inherits basic canonical codes. Thus it suffices to show (given that $\mathsf{Split}(\mathbb{X})$ is a Turing category as shown in Cockett and Hofstra (2008)) that we can form the internal hom for idempotents that are on the Turing object as all objects in $\mathsf{Split}(\mathbb{X})$ are isomorphic to such an idempotent. Let $u, v : T \to T$ be idempotents then define

$$\frac{T \times T \xrightarrow{1 \times u} T \times T \xrightarrow{\bullet} T \xrightarrow{v} T}{[u, v] := T \xrightarrow{\lambda[(1 \times u) \bullet v]} T} .$$

Observe that $[u, v]$ so defined is an idempotent:

$$\begin{aligned} [u, v][u, v] &= [u, v][u, v]\lambda(\bullet) \\ &= \lambda(([u, v] \times 1)([u, v] \times 1) \bullet) && \text{(substitutive)} \\ &= \lambda(([u, v] \times 1)(1 \times u) \bullet v) \\ &= \lambda((1 \times u) \bullet v) = [u, v]. \end{aligned}$$

Next define an evaluation $\mathsf{ev} : [u, v] \times u \to v$ by

$$\mathsf{ev} := T \times T \xrightarrow{1 \times u} T \times T \xrightarrow{\bullet} T \xrightarrow{v} T.$$

This is well defined as

$$\begin{aligned} ([u, v] \times u)\mathsf{ev}v &= ([u, v] \times u)(1 \times u) \bullet vv \\ &= (1 \times u)(\lambda((1 \times u) \bullet v) \times 1) \bullet v \\ &= (1 \times u)(1 \times u) \bullet vv \\ &= \mathsf{ev}. \end{aligned}$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

If $f : (w \times u) \to v$, then the canonical code is already a map $\lambda[f] : w \to [u, v]$ in $\mathsf{Split}(\mathbb{X})$ as

$$\begin{aligned} w\lambda[f][u, v] &= \lambda[(w \times 1)f][u, v] \\ &= \lambda[(w \times 1)f]\lambda[(1 \times u) \bullet v)] \\ &= \lambda[\lambda[((w \times 1)f)] \times 1)(1 \times u) \bullet v)] \\ &= \lambda[(1 \times u)\lambda[((w \times 1)f)] \times 1) \bullet v)] \\ &= \lambda[(1 \times u)(w \times 1)fv] \\ &= \lambda[(w \times u)fv] = \lambda[f]. \end{aligned}$$

Furthermore, clearly, evaluating this code returns f:

$$(\lambda[f] \times u)\mathsf{ev} = (\lambda[f] \times u)(1 \times u) \bullet v = (\lambda[f] \times u) \bullet v = (1 \times u)fv = f.$$

Finally, for uniqueness of the code suppose $h = wh[u, v]$ and $(h \times u)\mathsf{ev} = f$, then we have

$$\begin{aligned} h = wh[u, v] &= wh\lambda[(1 \times u) \bullet v] = \lambda[((wh) \times u) \bullet v] \\ &= \lambda[(w \times u)fv] = \lambda[f]. \end{aligned}$$

□

Note that $[1_T, 1_T] = \lambda[\bullet]$ is an idempotent which in $\mathsf{Split}(\mathbb{X})$ witnesses that the Turing object in the splitting is a reflexive object.

Remark 2.1. A category which has a structure but only when one moves to the idempotent splitting is said to *latently* have that structure. Thus, a category which when one splits the idempotents becomes a Cartesian-closed category is called a *latent Cartesian-closed category*. Thus, a Turing category with canonical codes is a latent Cartesian-closed category. Similarly, one may have products only when the idempotents are split, in which case one is a *latent Cartesian category*. The full subcategory of a Turing category, consisting of just the Turing object, is always a latent Cartesian category, as every object is a retract of the Turing object. Indeed, when the Turing category has canonical codes this one object category is also a latent Cartesian-closed category.

This immediately gives the Scott–Koymans theorem as given a model (that is a Turing category with canonical codes in which every object is a product of the Turing object) one can split the idempotents to obtain the Turing object as a reflexive object in a Cartesian-closed Turing category. One can then extract a model from this Turing category. However, splitting the idempotents does not change the model so one has the following.

Theorem 2.4. (Koymans 1982; Scott 1980). *Every model of the untyped λ-calculus is equivalent to a model arising from a reflexive object in a Cartesian-closed Turing category.*

3. The Additive λ-Calculus

The previous section sets the broad outline for this and the next section. In these sections, we will follow our program of adding structure step by step to the untyped λ-calculus. The first step in this program is to consider left-additive structure which is required in order to discuss differential structure. Basically in an additive Cartesian theory allows one to "add" terms (see Table 4). Adding such structure to the λ-calculus has been studied as the "parallel λ-calculus" in Dezani-Ciancaglini (1996) and Faure and Miguel (2009), and as the "algebraic λ-calculus" Vaux (2009). The first paper discusses the case when the addition is idempotent (and non-determinism), the second more generally assumes a semantics with an "aggregate monad." while the third covers the

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Table 4. Term formation rules for additive term logic

$$\frac{\Gamma \vdash m : A \quad \Gamma \vdash n : A}{\Gamma \vdash (m + n) : A}\ \text{Addition} \qquad \frac{}{\Gamma \vdash 0 : A}\ \text{Zero}$$

case when the homsets are vector spaces. Here we shall follow most closely the approach in Vaux (2009) but assume purely additive structure.

A **left-additive category** $\mathbb{X}$ is a category in which each homset $\mathbb{X}(A, B)$ is a commutative monoid, and where

$$f(g + h) = fg + fh \qquad f0 = 0.$$

A map f in a left-additive category is **additive** when $(g + h)f = gf + hf$ and $0f = 0$.

A **Cartesian left-additive category** is a left-additive category with finite products such that

$$(f + g) \times (h + k) = (f \times h) + (g \times k) \qquad 0 \times 0 = 0,$$

and where the diagonal map $\Delta : A \longrightarrow A \times A$ and all projections π_i are additive. An important characterization of Cartesian left-additive categories is as follows.

Proposition 3.1. ((Blute et al. 2009) Proposition 1.2.2). *To have a Cartesian left-additive category is to have a Cartesian category in which each object, X, has a canonical commutative monoid structure,* $X \times X \xrightarrow{+_X} X \xleftarrow{0_X} 1$, *that satisfies the following coherence:*

$$+_{A\times B} := (A \times B) \times (A \times B) \xrightarrow{\text{ex}} (A \times A) \times (B \times B) \xrightarrow{+_A \times +_B} A \times B.$$

3.1 Syntax of the additive λ-calculus

The syntax of the untyped additive λ-calculus is again defined by unityped terms in context. The term formation rules use the usual Cartesian rules of Table 1, the additive rules of Table 4, and the formation rules for the λ-calculus of Table 3.

The equations for β-equality and α-conversion hold together with equations to make $(U, +, 0)$ into a commutative monoid, and equations which govern the interaction of addition with products, λ-abstraction, and application:

$$\begin{aligned} (t_0, t_1) + (s_0, s_1) &= (t_0 + s_0, t_1 + s_1) && \text{and} \quad () = 0, \\ \lambda x.(m + n) &= \lambda x.m + \lambda x.n && \text{and} \quad \lambda x.0 = 0, \\ (m + n)\, a &= m\, a + n\, a && \text{and} \quad 0\, a = 0. \end{aligned}$$

These equations, together with the equations E, generate the theory $\Lambda_{\beta+}(\Omega, E)$.

Again we note that a pure equational theory can be formed by removing all the product structure: as before the (essentially) cut-eliminated terms of atomic types are then precisely the same for the two theories. Having patterns and tuples facilitates the translation into a left-additive category, which is why we continue to entertain this form of the type theory.

As before there is a category associated to the unityped additive logic (based on the single type U), $\mathcal{C}(\Lambda_{\beta+})$, called the classifying category of the additive λ-calculus. The objects are words in U, $\times$, 1 and the arrows are sequents with one premise on the left of the turnstile. Composition is, again, substitution. We state modestly the following.

Proposition 3.2. $\mathcal{C}(\Lambda_{\beta+}(\Omega, E))$ *is a Cartesian category in which U is a commutative monoid.*

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Our next task is to develop the categorical semantics of the additive λ-calculus, and thus, the properties of this category. To do this, we must address two related issues, namely what it means to be an additive *universal object and what it means to have* additive *canonical codes.*

3.2 Additive universal objects

The idea behind a universal object in a category is that the entire category is determined by the monoid of endomorphisms of that object. For plain categories, this just means that every object is a retract of U. However, in a Cartesian left-additive category a universal object must have a stronger property as it must also induce the additive structure on each object. An object U in a left-additive category $\mathbb{X}$ is an **additive universal** object in case every object, A, is a retract of U in such a manner that the retraction r_A, in $A \lhd^{s_A}_{r_A} U$, is an additive map. This requirement ensures that the additive structure on A is determined by that of U:

Lemma 3.1. *In any left-additive Cartesian category with a retract $A \lhd^{s_A}_{r_A} U$ which has r_A additive:*

(i) $A \times A \xrightarrow{+_A} A = A \times A \xrightarrow{s_A \times s_A} U \times U \xrightarrow{+} U \xrightarrow{r_A} A$ *and* $0_A = 0r_A$;
(ii) $e_A = r_A s_A$ *satisfies* $+_U e_A = (e_A \times e_A) +_U e_A$.

Note that (*i*) implies that $(fe + ge)r = (f + g)r$. When an idempotent $e : U \to U$ satisfies part *(ii)*, i.e. $+_U e = (e \times e) +_U e$, we will say e is a **retractively** U**-additive idempotent**.

Proof.

(i) As r_A is absolutely epic, the additive structure on A is completely determined by the requirement that r_A is additive. Thus, it suffices to show that this definition of addition makes an r_A a homomorphism:

$$\begin{array}{ccccc} U \times U & \xrightarrow{+} & U & \xleftarrow{0} & 1 \\ {\scriptstyle r_A \times r_A}\downarrow & & \downarrow{\scriptstyle r_A} & \swarrow{\scriptstyle 0_A} & \\ A \times A & \xrightarrow[+_A]{} & A & & \end{array}$$

The right triangle clearly commutes. The left square commutes as

$$\begin{aligned} (r_A \times r_A)+_A &= (r_A \times r_A)(s_A \times s_A) +_U r_A \\ &= (r_A \times r_A)(s_A \times s_A) +_U r_A \\ &= (r_A \times r_A)(s_A \times s_A)(r_A \times r_A) +_A \quad (r_A \text{ is additive}) \\ &= (r_A \times r_A)+_A = +_U r_A. \end{aligned}$$

(ii) Here is the calculation:

$$\begin{aligned} (e_A \times e_A) +_U e_A &= (e_A \times e_A) +_U r_A s_A \\ &= (e_A \times e_A)(r_A \times r_A) +_A s_A \\ &= (r_A \times r_A) +_A s_A \\ &= +_U r_A s_A = +_U e_A. \end{aligned}$$

□

The following – completely general – lemma lets us define additive structure on objects using U-additive retractions.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Lemma 3.2. *If $(U, +_U, 0_U)$ is a commutative monoid in a Cartesian category, $\mathbb{X}$, and $A \lhd^{s_A}_{r_A} U$ with $e_A = r_A s_A$ U-additive – i.e. $(e_A \times e_A) +_U e_A = +_U e_A$ – then there is a unique commutative monoid structure on A which makes r_A a homomorphism.*

Proof. As r_A is absolutely epic if we can induce an operation on A which is a homomorphism, it will immediately follow that all the equations required of a commutative monoid will hold. We have no choice for the zero it must be $0_A = 0_U r_A$, for the addition we try the formula suggested by the above $+_A := (s_A \times s_A) +_U r_A$. We must check that this makes the homomorphism square commute:

$$\begin{aligned}(r_A \times r_A)+_A &= (r_A \times r_A)(s_A \times s_A) +_U r_A \\ &= (e_A \times e_A) +_U e_A r_A = +_U e_A r_A \quad \text{(As } e_A \text{ is additive)} \\ &= +_U r_A.\end{aligned}$$

□

This means, when U is a universal object, which is a commutative monoid, we may, using Lemma 3.2, induce additive structure on any retract $A \lhd U$, whose induced idempotent is additive: this will automatically make the retract additive. In order to create left-additive structure on the whole category from the additive structure on U, we must select a particular way in which each object is an additive retract so that we may induce a unique additive structure on each object. Furthermore, to ensure the result is a *Cartesian* left-additive category, it is also necessary for these induced additions to be compatible with the product: this means we must demand that the additive structure on the product $A \times B$ be defined componentwise.

A **universal structure**, $\mathcal{U}$, for $U \in \mathbb{X}$ consists of, for each $A \in \mathbb{X}$, a way in which A is a retract of U, $\mathcal{U}(A) = A \lhd^{s_A}_{r_A} U$. Clearly if U has a universal structure, it must be a universal object in the original sense. Recall a Cartesian category, $\mathbb{X}$, here is assumed to have chosen products: so there are chosen functors $\times : \mathbb{X} \times \mathbb{X} \to \mathbb{X}$ and $1 : \nVdash \to \mathbb{X}$ right adjoint to the diagonal and final functors, respectively. In a Cartesian category with chosen products, if $(U, +_U, 0_U)$ is a commutative monoid, then a universal structure $\mathcal{U}$ for U is said to be **additively coherent** in case:

[UAC.1] $\mathcal{U}(U) = U \lhd^{1_U}_{1_U} U$.
[UAC.2] The retraction, $r_{U\times U}$, of $\mathcal{U}(U \times U)$ satisfies:

$$\begin{array}{ccccc} U \times U & & \xrightarrow{\qquad +_U \qquad} & & U \\ {\scriptstyle r_{U\times U}\times r_{U\times U}}\downarrow & & & & \downarrow{\scriptstyle r_{U\times U}} \\ (U \times U) \times (U \times U) & \xrightarrow[\text{ex}]{} & (U \times U) \times (U \times U) & \xrightarrow[+_U \times +_U]{} & U \times U \end{array}$$

[UAC.3] Each idempotent, $e_A = r_A s_A$, of $\mathcal{U}(A)$ is retractively U-additive.
[UAC.4] $\mathcal{U}(A \times B)$ has $s_{A\times B} = (s_C \times s_B) s_{U\times U}$ and $r_{A\times B} = r_{U\times U}(r_A \times r_B)$.

In a left-additive Cartesian category with an additive universal object, there is no guarantee that one has an additively coherent universal structure. However, given an additively coherent universal structure, one can generate a left-additive category as follows.

Proposition 3.3. *If $\mathbb{X}$ is a category with chosen products and a commutative monoid $(U, +_u, 0_U)$ which has an additively coherent universal structure, $\mathcal{U}$, then there is a unique left-additive structure on $\mathbb{X}$ making each retraction in $\mathcal{U}$ additive.*

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Proof. The fact that $\mathcal{U}$ is additively universal allows us to use Lemma 3.2 to induce an additive structure on each object: this structure is the unique structure which makes each retraction additive.

It remains only to check that these additive structures behave coherently with respect to the exchange law so we can apply Proposition 3.1:

$$\begin{aligned}
&+_{A\times B} = (s_{A\times B} \times s_{A\times B}) +_U r_{A\times B}\\
&= ((s_A \times s_B) \times (s_A \times s_B))(s_{U\times U} \times s_{U\times U}) +_U r_{U\times U}(r_A \times r_B)\\
&= ((s_A \times s_B) \times (s_A \times s_B))(s_{U\times U} \times s_{U\times U})(r_{U\times U} \times r_{U\times U})\\
&\quad \mathsf{ex}(+_U \times +_U)(r_A \times r_B)\\
&= \mathsf{ex}((s_A \times s_A) \times (s_B \times s_B))(+_U \times +_U)(r_A \times r_B)\\
&= \mathsf{ex}(+_A \times +_B).
\end{aligned}$$

□

3.3 Left-additive Turing categories

A **left-additive Turing category** is a Turing category which is Cartesian left additive and has each universal application $\bullet_{BC} : T \times B \to C$ additive in its first argument:

$$\langle h + k, g\rangle \bullet = \langle h, g\rangle \bullet + \langle k, g\rangle \bullet \quad \text{and} \quad \langle 0, g\rangle \bullet = 0.$$

A left-additive Turing category $\mathbb{X}$ has **additive canonical codes** when as a Turing category $\mathbb{X}$ has canonical codes and $\lambda[f + g] = \lambda[f] + \lambda[g]$ and $\lambda[0] = 0$.

We have the following recognition theorem for left-additive Turing categories.

Proposition 3.4. *$\mathbb{X}$ is a left-additive Turing category if and only if $\mathbb{X}$ is Cartesian left-additive, with an additively universal object T which has a Turing map $\bullet : T \times T \to T$ which is additive in its first argument.*

Furthermore, the Turing category has additively canonical codes if and only if the Turing map has additively canonical codes.

Proof. Let $\mathbb{X}$ be a left-additive Turing category, then the retraction $\langle 1_T, !_T\rangle \bullet_{1A} : T \to A$, from Proposition 2.2, is additive as

$$(f + g)\,\langle 1, !\rangle \bullet = \langle f + g, !\rangle \bullet = \langle f, !\rangle \bullet + \langle g, !\rangle \bullet = f\,\langle 1, !\rangle \bullet + g\,\langle 1, !\rangle \bullet$$

and similarly for 0. Thus the Turing map is additively universal.

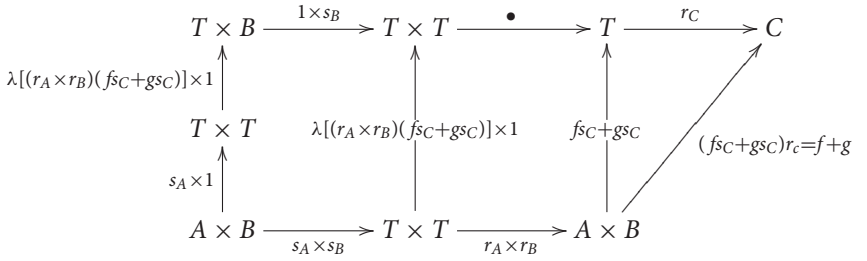
Conversely, given that T is additively universal, we must show that the derived applications $(1 \times s_B) \bullet r_C : T \times B \to C$ are additive in the first argument:

$$\langle f + g, h\rangle\,(1 \times s) \bullet r = \langle f + g, hs\rangle \bullet r = \langle f, hs\rangle \bullet r + \langle g, hs\rangle \bullet r$$

and similarly $\langle 0, h\rangle\,(1 \times s) \bullet r = 0$.

Finally we must show that, if the Turing map has additively universal codes, then the derived applications have additively canonical codes. For this, consider the following diagram:

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press



Thus, the canonical code for $f + g$ is derived from the canonical code for $(r_A \times r_B)(fs_C + gs_C)$ which means

$$\begin{aligned} s_A\lambda[(r_A \times r_B)(fs_C + gs_C)] &= \lambda[(1_A \times r_B)(fs_C + gs_C)] \\ &= \lambda[(1 \times r_B)fs_C + (1 \times r_B)gs_C] \\ &= \lambda[(1 \times r_B)fs_C] + \lambda[(1 \times r_A)gs_C]. \end{aligned}$$

Note that $0_C = 0_T r_C$ so that $\lambda[0_C] = 0_T$; thus, these codes are additively canonical. □

We are now ready to show the following.

Theorem 3.1. *$\mathcal{C}(\Lambda_{\beta+}(\Omega, E))$ is a left-additive Turing category with additively canonical codes.*

Proof. We show that in $\mathcal{C}(\Lambda_{\beta+}(\Omega, E))$ the unitype U has a universal structure which is additively coherent and whence is a Cartesian left-additive category. To complete the theorem, we must then show the U has an additive Turing map with additive canonical codes.

We first need to set up the universal structure for U: as every object is a formal product of the unitype U, it clearly has chosen products and U is a commutative monoid. We define the universal structure for U as follows:

$$\begin{aligned} \mathcal{U}(1) &= 1 \lhd^{0_U}_{!} U, \\ &\quad \text{where } 0_U = [() : 1 \vdash 0 : U] \text{ and } !_U = [x : U \vdash () : 1], \\ \mathcal{U}(U) &= 1 \lhd^{1_U}_{1_U} U, \\ \mathcal{U}(U \times U) &= U \times U \lhd^{s_{U\times U}}_{r_{U\times U}} U, \\ &\quad \text{where } s_{U\times U} = [(x, y) : U \times U \vdash \lambda z.zxy] \\ &\quad \text{and } r_{U\times U} = [z : U \vdash (z(\lambda xy.x), z(\lambda xy.y))], \\ \mathcal{U}(A \times B) &= A \times B \lhd^{s_{A\times B}}_{r_{A\times B}} U, \\ &\quad \text{where, inductively, } s_{A\times B} = (s_A \times s_B)s_{U\times U} \text{ and } r_{A\times B} = r_{U\times U}(r_A \times r_B). \end{aligned}$$

$\mathcal{U}$ depends inductively on the retract structure for 1, U, and $U \times U$. We must verify that this is additively coherent.

[UAC.1] Is true by choice.
[UAC.2] For this, we have

$$\begin{aligned} &(r_{U\times U} \times r_{U\times U})\mathsf{ex}(+_U \times +_U) \\ &\quad = ([a \vdash (a(\lambda xy.x), a(\lambda xy.y))], [b \vdash (b(\lambda xy.x), b(\lambda xy.y))])\mathsf{ex}(+_U \times +_U) \\ &\quad = [(a, b) \vdash ((a(\lambda xy.x), a(\lambda xy.y)), (b(\lambda xy.x), b(\lambda xy.y)))]\mathsf{ex}(+_U \times +_U) \end{aligned}$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

$$
\begin{aligned}
&= [(a,b) \vdash ((a(\lambda xy.x), b(\lambda xy.x)), (a(\lambda xy.y), b(\lambda xy.y)))](+_U \times +_U)\\
&= [(a,b) \vdash ((a(\lambda xy.x) + b(\lambda xy.x)), (a(\lambda xy.y) + b(\lambda xy.y)))]\\
&= [(a,b) \vdash ((a+b)(\lambda xy.x), (a+b)(\lambda xy.y))]\\
&= [(a,b) \vdash ((a+b)(\lambda xy.x), (a+b)(\lambda xy.y))]\\
&= [(a,b) \vdash a+b][z \vdash (\lambda xy.x, \lambda xy.y)]\\
&= +_U r_{U\times U}.
\end{aligned}
$$

[UAC.3] We must prove for each X that e_X is U-additive. We work inductively on the structure of the type:

$X = 1$: Then $e_1 = r_1 s_1 = [x : U \vdash 0 : U]$ which is clearly U-additive.

$X = U$: Then $e_U = r_U s_U = 1_U$ which is clearly additive.

$X = A \times B$: Here we shall inductively assume e_A and e_B are retractively U-additive and we have

$$
\begin{aligned}
+_U e_{A\times B} &= +_U r_{A\times B} s_{A\times B}\\
&= +_U r_{U\times U}(r_A \times r_B) s_{A\times B}\\
&= (r_{U\times U} \times r_{U\times U})\mathsf{ex}(+_U \times +_U)(r_A \times r_B) s_{A\times B}\\
&= (r_{U\times U} \times r_{U\times U})\mathsf{ex}((r_A \times r_A) \times (r_S \times r_S))(+_A \times +_B) s_{A\times B}\\
&= ((r_{U\times U}(r_A \times r_B)) \times (r_{U\times U}(r_A \times r_B)))\mathsf{ex}(+_A \times +_B) s_{A\times B}\\
&= (r_{A\times B} \times r_{A\times B})\mathsf{ex}(+_A \times +_B) s_{A\times B}\\
&= (e_{A\times B} \times e_{A\times B})(r_{A\times B} \times r_{A\times B})\mathsf{ex}(+_A \times +_B) s_{A\times B}\\
&= (e_{A\times B} \times e_{A\times B}) +_U e_{A\times B},
\end{aligned}
$$

where the last step is by reversing first five the steps.

[UAC.4] Is true by construction.

The Turing map is additive in its first argument as $(m+n)t = mt + nt$ and the canonical codes are additive as $\lambda x.(m+n) = (\lambda x.m) + (\lambda x.n)$. □

3.4 Interpreting the additive λ-calculus

Our next objective is to show, that for any left-additive Turing category, $\mathbb{X}$, with additively canonical codes, there is a left-additive functor $[\![_]\!]_{\mathcal{I}} : \mathcal{C}(\Lambda_{\beta+}) \to \mathbb{X}$ for any interpretation, $\mathcal{I}$, of Ω satisfying the equations E (which are now allowed to use the additive structure). From a logical perspective, this says that left-additive Turing categories with additively canonical codes are sound models of the additive λ-calculus.

Let $\mathbb{X}$ be a left-additive Turing category with additive canonical codes with Turing object T and Turing map is $\bullet : T \times T \to T$. The functor $[\![_]\!]_{\mathcal{I}} : \mathcal{C}(\Lambda_{\beta+}) \to \mathbb{X}$ is defined in the same way as before on variables, application, abstraction, and tuples: all we have to describe is the interpretation of the additive structure:

$$
\begin{aligned}
[\![p : X \vdash 0 : U]\!] &= 0 : [\![X]\!] \to T,\\
[\![p : X \vdash m+n : U]\!] &= [\![p : X \vdash m : U]\!] + [\![p : X \vdash n : U]\!] : [\![X]\!] \to T.
\end{aligned}
$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Proposition 3.5. *When $\mathbb{X}$ is a left-additive Turing category with additive canonical codes*

$$[\![_]\!]_{\mathcal{I}} : \mathcal{C}(\Lambda_{\beta+}(\Omega, E)) \to \mathbb{X}$$

is a Cartesian left-additive functor which preserves the Turing object, the Turing map, and its basic canonical codes.

Proof. Substitution must turn into composition: because the theory now has addition, we check that substitution into a sum of terms of U is still composition:

$$\begin{aligned}
&[\![p : X \vdash (m_1 + m_2)[n/p'] : U]\!] \\
&\quad = [\![p : X \vdash m_1[n/p'] + m_2[n/p] : U]\!] \\
&\quad = [\![p : X \vdash m_1[n/p'] : U]\!] + [\![p : X \vdash m_2[n/p'] : U]\!] \\
&\quad = [\![p : X \vdash n : Y]\!][\![p' : Y \vdash m_1 : U]\!] + [\![p : X \vdash n : Y]\!][\![p' : Y \vdash m_2 : U]\!] \\
&\quad = [\![p : X \vdash n : Y]\!]([\![p' : Y \vdash m_1 : U]\!] + [\![p' : Y \vdash m_2 : U]\!]) \qquad \text{left additivity} \\
&\quad = [\![p : X \vdash n : Y]\!][\![p' : Y \vdash m_1 + m_2 : U]\!].
\end{aligned}$$

We must show the new identities of the additive λ-calculus are preserved by the interpretation:

$$\lambda x.(m + n) = \lambda x.m + \lambda x.n \qquad \text{and} \qquad (m + n)t = mt + nt$$

$$\begin{aligned}
&[\![p : X \vdash \lambda x.(m + n) : U]\!] \\
&\quad = \lambda\left[[\![(p, x) : X \times U \vdash m + n : U]\!]\right] \\
&\quad = \lambda\left[[\![(p, x) : X \times U \vdash m : U]\!] + [\![(p, x) : X \times U \vdash n : U]\!]\right] \\
&\quad = \lambda\left[[\![(p, x) : X \times U \vdash m : U]\!]\right] + \lambda\left[[\![(p, x) : X \times U \vdash n : U]\!]\right] \qquad \text{(additivity of codes)} \\
&\quad = [\![(p, x) : X \times U \vdash \lambda x.m : U]\!] + [\![(p, x) : X \times U \vdash \lambda x.n : U]\!] \\
&\quad = [\![(p, x) : X \times U \vdash \lambda x.m + \lambda x.n : U]\!]
\end{aligned}$$

and

$$\begin{aligned}
&[\![p : X \vdash (m + n)t : U]\!] \\
&\quad = \langle [\![p : X \vdash m + n : U]\!], [\![p : X \vdash t : U]\!] \rangle \bullet \\
&\quad = \langle [\![p : X \vdash m : U]\!] + [\![p : X \vdash n : U]\!], [\![p : X \vdash t : U]\!] \rangle \bullet \\
&\quad = \langle [\![p : X \vdash m : U]\!], [\![p : X \vdash t : U]\!] \rangle \bullet + \langle [\![p : X \vdash n : U]\!], [\![p : X \vdash t : U]\!] \rangle \bullet \\
&\quad = [\![p : X \vdash mt : U]\!] + [\![p : X \vdash nt : U]\!] \\
&\quad = [\![p : X \vdash mt + nt : U]\!]
\end{aligned}$$

□

Again with more work, this interpretation can be turned into an adjunction between additive λ-theories and left-additive Turing categories with additive canonical codes.

3.5 Splitting of retractively additive idempotents

An idempotent $A \xrightarrow{e} A$ in a Cartesian left-additive category is **retractively additive** when $+_A e = (e \times e) +_A e$.

We make the following observation about retractively additive idempotents.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Lemma 3.3. *The following hold in any Cartesian left-additive category:*

(i) *If e is an additive idempotent, then e is retractively additive (so, in particular, identities are retractively additive).*
(ii) *If e, e' are retractively additive idempotents, then so is $e \times e'$.*
(iii) *If e, e' are retractively additive idempotents and $ee' = e'e$, then ee' is retractively additive.*

Proof. We prove (iii) here. Suppose e, e' are retractively additive, and that $ee' = e'e$. Then consider

$$\begin{aligned} +ee' &= (e \times e) + ee' \\ &= (e \times e) + e'e \\ &= (e \times e)(e' \times e') + e'e \\ &= (ee' \times ee') + ee'. \end{aligned}$$

□

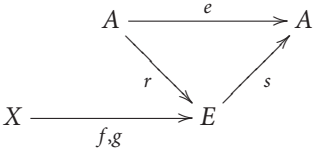
A split idempotent $e = rs$ with $sr = 1$ **splits left additively** when r is additive. Note the following.

Lemma 3.4. *In any Cartesian left-additive category, if $e = rs$ splits left additively then*

– *e is retractively additive;*
– *e is additive if and only if s is additive.*

Thus, every left additively split idempotent is retractively additive, but retractively additive maps need not split. A Cartesian left-additive category is a **split Cartesian left-additive category** when every retractively additive map splits left additively.

In a Cartesian left-additive category, suppose $e = A \xrightarrow{r} E \xrightarrow{s} A$, and consider the following diagram:



One can add f, g directly to obtain $f + g$. One can also add $fs + gs$ and then postcompose with r.

Lemma 3.5. *If rs is a left-additive split idempotent, then $f + g = (fs + gs)r$ and $0 = 0r$.*

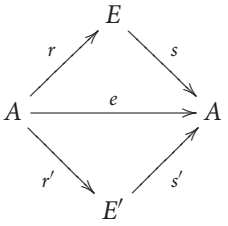
We may restate the above in terms of the commutative monoid structure: if rs left additively splits, then

$$+_E = (s \times s) +_A r \qquad 0_E = 0_A r.$$

Left-additive splittings are unique up to additive isomorphism:

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Proposition 3.6. *If*



commutes with r, r' additive and $sr = 1, s'r' = 1$, then there is a unique isomorphism $B \xrightarrow{\alpha} B'$ such that $r\alpha = r'$ and $\alpha s' = s$. Moreover, α is additive.

Next, we give a construction that completes a Cartesian left-additive category $\mathbb{X}$ to be a split Cartesian left-additive category. Let $\mathcal{E}$ be a product-closed collection of retractively additive idempotents that contains all the identities, then observe that each commutative monoid in $\mathbb{X}$ lifts directly to $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$. Further, each $e \in \mathcal{E}$ splits as

$$1_A \xrightarrow{e} e \xrightarrow{e} 1_A.$$

If e is retractively additive then, for e to be left additively split, Lemma 3.5 tells us that

$$+_e := (e \times e) +_A e \qquad 0_e = 0e.$$

Denote $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$ with this additive structure by $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$.

Proposition 3.7. *Let $\mathbb{X}$ be a Cartesian left-additive category. If $\mathcal{E}$ is a product-closed collection of retractively additive idempotents that contains all the identities, then $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ is a Cartesian left-additive category.*

Proof. As $\mathcal{E}$ is product closed, $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ is a Cartesian category. We will use the commutative monoid characterization of Cartesian left-additive categories to prove that $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ is a Cartesian left-additive category. First note that the maps $0e$ and $+e$ are well-defined maps in $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$.

For the unit law (recalling that the identity on an idempotent e is the map e itself), we must show that $\langle 0e, e\rangle + e = e$:

$$\begin{aligned}\langle 0e, e\rangle + e &= \langle 0, 1\rangle\ (e \times e) + e\\ &= \langle 0, 1\rangle + e\\ &= e.\end{aligned}$$

For the associative law, we must show that $((\pi_0 e + \pi_1 e)e + \pi_2 e)e = (\pi_0 e + (\pi_1 e + \pi_2 e)e)e$:

$$\begin{aligned}((\pi_0 e + \pi_1 e)e + \pi_2 e)e &= ((\pi_0 + \pi_1)e + \pi_2 e)e\\ &= ((\pi_0 + \pi_1) + \pi_2)e\\ &= (\pi_0 + (\pi_1 + \pi_2))e\\ &= (\pi_0 e + (\pi_1 e + \pi_2 e)e)e.\end{aligned}$$

Commutativity follows similarly.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

For the exchange law, use the naturality of ex:

$$\begin{aligned}&((e \times e') \times (e \times e')) +_{A\times B} (e \times e')\\ &\quad = ((e \times e') \times (e \times e'))\mathsf{ex}(+_A \times +_B)(e \times e')\\ &\quad = \mathsf{ex}((e \times e) \times (e' \times e'))(+_A \times +_B)(e \times e')\\ &\quad = \mathsf{ex}((e \times e) +_A e) \times ((e' \times e') +_B e')\end{aligned}$$

as required.

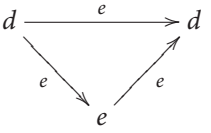
Thus $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ is a Cartesian left-additive category. □

Consider the addition of maps in $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$; suppose $h, k : c \to d$. Then using the above, their sum is

$$\langle h, k\rangle +_d = \langle h, k\rangle(\pi_0 + \pi_1)d = (h + k)d = (hd + kd)d.$$

Corollary 3.1. Let $\mathbb{X}$ be a Cartesian left-additive category, and let $\mathcal{E}$ be the collection of all retractively additive idempotents. Then $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ is a split Cartesian left-additive category.

Proof. It remains to show that all retractively additive idempotents split left additively. Let $e \in \mathcal{E}$ and suppose we view e as a map $d \xrightarrow{e} d$. Then e splits



We must show that $d \xrightarrow{e} e$ is additive. For the zero case, $0de = 0e$. For addition, use the fact that d is retractively additive, $de = e$, and finally that e is retractively additive:

$$(fd + gd)de = (f + g)de = (f + g)e = (fe + ge)e$$

as required. Thus $d \xrightarrow{e} e$ is additive. □

As an easy second corollary, we have the following.

Corollary 3.2. The embedding $\mathbb{X} \hookrightarrow \mathsf{Split}_{\mathcal{E}_\mathcal{D}}(\mathbb{X})$ is a Cartesian left-additive functor.

From Cockett and Hofstra (2008), Section 3.5, we know that $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$ is a Turing category whenever $\mathcal{E}$ is product closed and contains identities. Combined with Proposition 3.1, we have that when $\mathcal{E}$ is the retractively additive idempotents, that $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ is both a Cartesian left-additive category and a Turing category.

Proposition 3.8. *For any left-additive Turing category $\mathbb{X}$, when $\mathcal{E}$ is the retractively additive idempotents, then $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ is a left-additive Turing category in which every retractively additive idempotent splits left additively.*

Proof. We need only to prove that for each pair of idempotents e, d, the application $1_T \times e \xrightarrow{\bullet} d$ is additive in its first argument. For the plus case, let $h, k : c \to 1_T$, so that the sum in $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ is really just $h + k$. Then consider the following:

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

$$\begin{aligned}
&\langle h+k, g\rangle(1_T \times e) \bullet d \\
&\quad = \langle h+k, g\rangle \bullet d \\
&\quad = (\langle h, g\rangle \bullet + \langle k, g\rangle \bullet)d \\
&\quad = (\langle h, g\rangle(1 \times e) \bullet d + \langle k, g\rangle(1 \times e) \bullet d)d
\end{aligned}$$

as required. Thus, $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ is a Cartesian left-additive category. □

3.6 Additive reflexive objects

A Cartesian-closed category that is also a Cartesian left-additive category is **Cartesian-closed left additive** when $\lambda[f+g] = \lambda[f] + \lambda[g]$.

We want to show that, following Theorem 2.3, splitting the retractively additive idempotents of a left-additive Turing category with left-additive canonical codes yields a Cartesian-closed left-additive Turing category. For this, it suffices to show that $[u, v] = \lambda[(1 \times u) \bullet_{AB} v]$ is an additively retractive idempotent whenever u, v are: we already know that it is an idempotent and its splitting gives the internal hom. Here is the calculation:

$$\begin{aligned}
&+_U [u, v] \\
&\quad = +_U \lambda[(1 \times u) \bullet_{AB} v] \\
&\quad = \lambda[(+_U \times u) \bullet_{AB} v] \\
&\quad = \lambda[(1 \times u)(+_U \times 1) \bullet_{AB} v] \\
&\quad = \lambda[(1 \times u)(((\pi_0 \times 1) \bullet_{AB}) \times ((pi_1 \times 1) \bullet_{AB})) +_U v)] \\
&\quad = \lambda[(1 \times u)(((\pi_0 \times 1) \bullet_{AB} v) \times ((\pi_1 \times 1) \bullet_{AB} v)) +_U)v] \quad (v \text{ add. retr.}) \\
&\quad = \lambda[(1 \times u)(((\pi_0 \times 1)(1 \times u) \bullet_{AB} v) \times ((\pi_1 \times 1)(1 \times u) \bullet_{AB} v)) +_U v] \\
&\quad = \lambda[(1 \times u)(((\pi_0 \times 1)([u, v] \times 1) \bullet_{AB}) \times ((\pi_1 \times 1)([u, v] \times 1) \bullet_{AB})) +_U v] \\
&\quad = \lambda[(1 \times u)((((\pi_0[u, v]) \times 1) \bullet_{AB}) \times ((((\pi_1[u, v]) \times 1) \bullet_{AB})) +_U v] \\
&\quad = \lambda[(1 \times u)(((([u, v] \times [u, v]) +_U) \times 1) \bullet_{AB})v] \\
&\quad = \lambda[(((([u, v] \times [u, v]) +_U) \times 1)(1 \times u) \bullet_{AB})v] \\
&\quad = ([u, v] \times [u, v]) +_U \lambda[(1 \times u) \bullet_{AB})v] \\
&\quad = ([u, v] \times [u, v]) +_U [u, v].
\end{aligned}$$

It then immediately follows the following.

Theorem 3.2. *If $\mathbb{X}$ is a left-additive Turing category with additive canonical codes then splitting the additively retractive idempotents,* $\mathsf{Split}_+(\mathbb{X})$*, yields a Cartesian-closed left-additive Turing category.*

This, in turn, means that every model of the additive λ-calculus can be seen to arise as the model of an additive reflexive object in a Cartesian-closed left-additive category.

4. The Differential λ-Calculus

A **Cartesian differential category** $\mathbb{X}$ is a Cartesian left-additive category with a combinator

$$\frac{A \xrightarrow{f} B}{A \times A \xrightarrow[D[f]]{} B}$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

that satisfies seven axioms:

[CD.1] $D[0] = 0$ and $D[f+g] = D[f] + D[g]$.
[CD.2] $\langle 0, g\rangle D[f] = 0$ and $\langle h+k, g\rangle D[f] = \langle h, g\rangle D[f] + \langle k, g\rangle D[f]$.
[CD.3] $D[1] = \pi_0$, $D[\pi_0] = \pi_0\pi_0$, and $D[\pi_1] = \pi_0\pi_1$.
[CD.4] $D[\langle f, g\rangle] = \langle D[f], D[g]\rangle$.
[CD.5] $D[fg] = \langle D[f], \pi_1 f\rangle D[g]$.
[CD.6] $\langle\langle g, 0\rangle, \langle h, k\rangle\rangle D[D[f]] = \langle g, k\rangle D[f]$.
[CD.7] $\langle\langle p, h\rangle, \langle g, k\rangle\rangle D[D[f]] = \langle\langle p, g\rangle, \langle h, k\rangle\rangle D[D[f]]$.

[CD.1] says that differentiation preserves addition, and **[CD.2]** that the derivative is additive in its first variable. **[CD.3]** and **[CD.4]** demand that differentiation is compatible with the product structure of the category. **[CD.5]** is the chain rule. **[CD.6]** says that the derivative is linear (see below) in its first variable, and **[CD.7]** says that one can interchange of partial derivatives.

In a Cartesian differential category, a map f is **linear** when $D[f] = \pi_0 f$. Intuitively – using the term logic which will be shortly introduced – this means

$$\frac{\mathrm{d}f(x)}{\mathrm{d}x}(p) \cdot v = f(v).$$

The notation $\frac{\mathrm{d}f(x,y)}{\mathrm{d}p}(x) \cdot v$ is read as the partial derivative of $f(x)$ with respect to x at the point p acting (linearly) on v. This is an atomic operation which binds the variable x and applies the differential all at once.

The following is Corollary 2.2.3 in Blute et al. (2009):

Proposition 4.1. *If a map is linear, then it is additive. Moreover, the class of linear maps forms a commutative monoid enriched category with by-products.*

The following lemma is quite useful in many calculations.

Lemma 4.1. *Let l be linear and f, g any maps. Then*

$$D[lf] = (l \times l)D[f] \qquad \text{and} \qquad D[gl] = D[g]l.$$

Proof. The first equality is because

$$D[lf] = \langle D[l], \pi_1 l\rangle D[f] = (l \times l)D[f]$$

and the second equality is because

$$D[gl] = \langle D[g], \pi_1 g\rangle D[l] = \langle D[g], \pi_1 g\rangle \pi_0 l = D[g]l.$$

□

In a Cartesian differential category, if $f : A \times B \to C$, we may define the **partial derivative** of f with respect to A as

$$D_{\times,0} := A \times (A \times B) \xrightarrow{\langle 1, 0\rangle \times 1} (A \times B) \times (A \times B) \xrightarrow{D[f]} C.$$

One may obtain the partial derivative of f with respect to B by

$$D_{\times,1} := B \times (A \times B) \xrightarrow{\langle 0, 1\rangle \times 1} (A \times B) \times (A \times B) \xrightarrow{D[f]} C.$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Table 5. Term formation rules for the differential λ-calculus

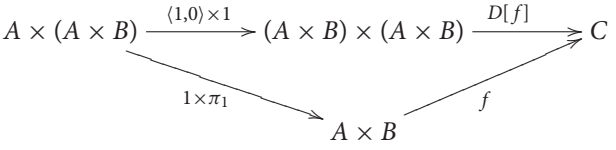
$$\frac{\Gamma, x:X, \Gamma' \vdash m:Y \quad \Gamma, \Gamma' \vdash p:X \quad \Gamma, \Gamma' \vdash v:X}{\Gamma, \Gamma' \vdash \frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot v:Y}$$

We will also make use of maps that are **linear in the first argument.** This means

$$(v, (p, y)) \mapsto \frac{\mathsf{d}f(x,y)}{\mathsf{d}x}\,(p)\cdot v = f(v,y),$$

that is, the partial derivative in the first argument is linear.

In a Cartesian differential category, we express this as



The following lemma is useful in many calculations.

Lemma 4.2. *Suppose* $h : A \times B \longrightarrow C$ *is linear in its first argument. Then* $(1 \times g)h$ *is linear in its first argument for any* g.

Proof. The proof is given by the calculation:

$$\begin{aligned}
&(\langle 1,0\rangle \times 1)D[(1\times g)h]\\
&\quad = (\langle 1,0\rangle \times 1)\left\langle D[1\times g], \pi_1(1\times g)\right\rangle D[h]\\
&\quad = \left\langle (\langle 1,0\rangle \times 1)D[1\times g], (\langle 1,0\rangle \times 1)\pi_1(1\times g)\right\rangle D[h]\\
&\quad = \left\langle (\langle 1,0\rangle \times)\left\langle \pi_0\pi_0, (\pi_1\times\pi_1)D[h]\right\rangle, \pi_1(1\times g)\right\rangle D[h]\\
&\quad = \left\langle \left\langle \pi_0, \langle 0, \pi_1\pi_1\rangle\, D[g]\right\rangle, \pi_1(1\times g)\right\rangle D[h]\\
&\quad = \left\langle \langle \pi_0, 0\rangle\, , \pi_1(1\times g)\right\rangle D[h]\\
&\quad = \left\langle \pi_0, \pi_1(1\times g)\right\rangle (\langle 1,0\rangle \times 1)D[h]\\
&\quad = (1\times(1\times g))(1\times\pi_1)h = (1\times\pi_1)(1\times g)h.
\end{aligned}$$

□

4.1 Syntax of the differential λ-calculus

The syntax of the untyped differential λ-calculus is again defined as unityped terms in context. Term formation uses the rules of Cartesian theories (Table 1), the rules for the λ-calculus (Table 3), the rules for the additive λ-calculus (Table 4), and a single new rule for the differential λ-calculus (Table 5).

Note that the syntax we use is from Blute et al. (2009) and is slightly different from, but equivalent to, the syntax used in Ehrhard and Regnier (2003). The two syntaxes are discussed in some depth in the Gallagher (2018) and shown to be equivalent.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

The following equations on terms hold for the differential λ-calculus:

[Dt.1] $\frac{\mathsf{d}m_1+m_2}{\mathsf{d}x}\,(p)\cdot v=\frac{\mathsf{d}m_1}{\mathsf{d}x}\,(p)\cdot v+\frac{\mathsf{d}m_2}{\mathsf{d}x}\,(p)\cdot v$ and $\frac{\mathsf{d}0}{\mathsf{d}x}\,(p)\cdot v=0$.

[Dt.2] $\frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot(v_1+v_2)=\frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot v_1+\frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot v_2$ and $\frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot 0=0$.

[Dt.3] Linearity of projections: $\frac{\mathsf{d}x}{\mathsf{d}x}\,(a)\cdot v=v$ and $\frac{\mathsf{d}y}{\mathsf{d}x}\,(a)\cdot v=0$ when $x\neq y$.

[Dt.4] The chain rule:

$$\frac{\mathsf{d}m[t/y]}{\mathsf{d}x}\,(p)\cdot v=\frac{\mathsf{d}m[p/x]}{\mathsf{d}y}\,(t[p/x])\cdot\left(\frac{\mathsf{d}t}{\mathsf{d}x}\,(p)\cdot v\right)+\frac{\mathsf{d}m[t[p/x]/y]}{\mathsf{d}x}\,(p)\cdot v.$$

[Dt.5] Linearity of the derivative:

$$\frac{\mathsf{d}\frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot y}{\mathsf{d}y}\,(q)\cdot v=\frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot v,$$

where $y\notin p, m$.

[Dt.6] Interchange of partial derivatives:

$$\frac{\mathsf{d}\frac{\mathsf{d}m}{\mathsf{d}x_1}\,(p_1)\cdot v_1}{\mathsf{d}x_2}\,(p_2)\cdot v_2=\frac{\mathsf{d}\frac{\mathsf{d}m}{\mathsf{d}x_2}\,(p_2)\cdot v_2}{\mathsf{d}x_1}\,(p_1)\cdot v_1.$$

[Dt.7] Currying differentials:

$$\frac{\mathsf{d}\lambda y.m}{\mathsf{d}x}\,(p)\cdot v=\lambda y.\frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot v,$$

where $y\notin \mathsf{fv}(p, v)$.

[Dt.8] Linearity of application:

$$\frac{\mathsf{d}\,y\,z}{\mathsf{d}y}\,(p)\cdot v=v\,z,$$

where z is a variable and $z\neq y$.

The logic for differentials we described here is purely equational: when one adds the product rules, one obtains the logic in Blute et al. (2009). In that logic, a term could be a tuple and so one needed some extra equations such as

$$\frac{\mathsf{d}()}{\mathsf{d}x}\,(p)\cdot v=()\quad\text{and}\quad\frac{\mathsf{d}(t_1,t_2)}{\mathsf{d}x}\,(p)\cdot v=\left(\frac{\mathsf{d}t_1}{\mathsf{d}x}\,(p)\cdot v,\frac{\mathsf{d}t_2}{\mathsf{d}x}\,(p)\cdot v\right).$$

The term logic in Blute et al. (2009) also expressed the chain rule as

$$\frac{\mathsf{d}m[t/y]}{\mathsf{d}x}\,(p)\cdot v=\frac{\mathsf{d}m}{\mathsf{d}y}\,(t[p/x])\cdot\left(\frac{\mathsf{d}t}{\mathsf{d}x}\,(p)\cdot v\right)$$

with the side condition that $x\notin m$. Here we have expressed the chain rule without a side condition. Notice that with the side condition the second term, the chain rule above vanishes and, thus, the chain rule in Blute et al. (2009) is recovered.

Two further features are available in the logic in Blute et al. (2009): parallel substitution and differentiation with respect to a pair. The parallel substitution translates out as a sequential substitution $t[(s_1,s_2)/(x,y)]=t[s_2/y][s_1/x]$. Note there is some loss here as in a parallel substitution one knows $x\notin s_2$ and $y\notin s_1$ which is lost in the seqential substitution.

Differentiation with respect to a pair has to be sequentialized:

$$\frac{\mathsf{d}t}{\mathsf{d}(x,y)}\,((p,q))\cdot(v,w)=\frac{\mathsf{d}t[q/y]}{\mathsf{d}x}\,(p)\cdot v+\frac{\mathsf{d}t[p/x]}{\mathsf{d}y}\,(q)\cdot w.$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Lemma 4.3. *In* $\Lambda_{\beta\partial}$,

(i) *If* $y \notin \mathsf{fv}(m)$ *then* $\frac{\mathsf{d}m}{\mathsf{d}y}\,(p)\cdot v = 0$;
(ii) $\frac{\mathsf{d}m\,n}{\mathsf{d}x}\,(p)\cdot v = \left(\frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot v\right)(n[p/x]) + \frac{\mathsf{d}(m[p/x])\,z_1}{\mathsf{d}z_1}\,(n[p/x])\cdot\frac{\mathsf{d}n}{\mathsf{d}x}\,(p)\cdot v$.

Proof. (i) Note that if $z \notin m$, then $m[0/z] = m$. Then,

$$\frac{\mathsf{d}m}{\mathsf{d}y}\,(a)\cdot v = \frac{\mathsf{d}m[0/z]}{\mathsf{d}y}\,(a)\cdot v = \frac{\mathsf{d}m}{\mathsf{d}z}\,(0[a/y])\cdot\frac{\mathsf{d}0}{\mathsf{d}y}\,(a)\cdot v = \frac{\mathsf{d}m}{\mathsf{d}z}\,(0)\cdot 0 = 0.$$

(ii) The calculation uses **[Dt.3]** and the above

$$\begin{aligned}
&\frac{\mathsf{d}m\,n}{\mathsf{d}x}\,(p)\cdot v\\
&= \frac{\mathsf{d}(z_1\,z_2)[m, n/z_1, z_2]}{\mathsf{d}x}\,(p)\cdot v\\
&= \frac{\mathsf{d}z_1\,n[a/x]}{\mathsf{d}z_1}\,(m[p/x])\cdot\left(\frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot v\right) + \frac{\mathsf{d}m[p/x]\,z_2}{\mathsf{d}z_2}\,(n[p/x])\cdot\left(\frac{\mathsf{d}n}{\mathsf{d}x}\,(p)\cdot v\right)\\
&= \left(\frac{\mathsf{d}m}{\mathsf{d}x}\,(p)\cdot v\right)n[p/x] + \frac{\mathsf{d}m[p/x]\,z_2}{\mathsf{d}z_2}\,(n[p/x])\cdot\left(\frac{\mathsf{d}n}{\mathsf{d}x}\,(p)\cdot v\right).
\end{aligned}$$

□

There is, as expected, an associated category $\mathcal{C}(\Lambda_{\beta\partial})$. We have immediately the following.

Proposition 4.2. *$\mathcal{C}(\Lambda_{\beta\partial})$ is a Cartesian category in which $(U, +_U, 0_U)$ is a commutative monoid.*

4.2 Differential universal objects

In Section 3.2, we introduced the notion of an additive universal object that induces an additive structure on each object. In a Cartesian differential category, we must again strengthen the notion of universal object, so that the derivative on maps $U \to U$ induces a derivative on all maps. An object U in a Cartesian differential category $\mathbb{X}$ is a **differential universal object** in case every object A is a linear retract of U, $A \lhd^{s_A}_{r_A} U$ – so the retraction is linear while the section need not be.

Lemma 4.4. *In any Cartesian differential category with a retract $A \lhd^{s_A}_{r_A} U$ in which r_A is linear,*

(i) *For any $A \xrightarrow{f} B$, $D[f] = (s_A \times s_A)D[r_A\, fs_B]r_B$;*
(ii) *$e_A = r_A s_A$ satisfies $(e_A \times e_A)D[e_A] = D[e_A]$ and $D[e_A]e_A = \pi_0 e_A$.*

When an idempotent satisfies part (*ii*) of the above and is retractively additive, we will say that it is a **retractively linear idempotent**.

Proof. The proof uses Lemma 4.1. For the first point, as r_A, r_B are linear,

$$(s_A \times s_A)D[r_A\, fs_B]r_B = (s_A \times s_A)(r_B \times r_B)D[fs_B r_B] = D[f].$$

That $(e_A \times e_A)D[e_A] = D[e_A]$:

$$\begin{aligned}
(e_A \times e_A)D[e_A] &= (e_A \times e_A)D[r_A s_A] = (e_A r_A \times e_A r_A)D[s_A]\\
&= (r_A \times r_A)D[s_A] = D[r_A s_A] = D[e_A].
\end{aligned}$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

That $D[e_A]e_A = \pi_0 e_A$:

$$D[e_A]e_A = D[e_A]r_A s_A = D[e_A r_A]s_A = D[r_A]s_A = \pi_0 r_A s_A = \pi_0 e_A.$$

□

We now extend the notion of additively coherent universal structure. In a Cartesian category with chosen products, where $(U, +_U, 0_U)$ is a commutative monoid, a universal structure $\mathcal{U}$ for U is **differentially coherent** if it is additively coherent and additionally the following conditions apply:

[UDC.1] There is a differential operator for U, i.e.

$$\frac{U \xrightarrow{f} U}{U \times U \xrightarrow[D[f]]{} U}$$

that satisfies **[CD.1,2,3,5]**.

[UDC.2] Each idempotent $e_A = r_A s_A$ of $\mathcal{U}(A)$ is retractively U-linear in the sense that

$$(e_A \times e_A)D[e_A] = D[e_A] \qquad \text{and} \qquad D[e_A]e_A = \pi_0 e_A.$$

[UDC.3] For $\mathcal{U}(U \times U) = U \times U \lhd^{s_{U\times U}}_{r_{U\times U}} U$, we have that

$$(s_{U\times U} \times s_{U\times U})D[r_{U\times U}\pi_0] = \pi_0\pi_0,$$
$$(s_{U\times U} \times s_{U\times U})D[r_{U\times U}\pi_1] = \pi_0\pi_1.$$

And that for any $f, g : U \to U$:

$$D\left[\langle f, g\rangle s_{U\times U}\right] r_{U\times U} = \langle D[f], D[g]\rangle.$$

[UDC.4] Let $U \xrightarrow{f} U$. The map

$$D^2[f] := (U \times U) \times (U \times U) \xrightarrow{s_{U\times U} \times s_{U\times U}} U \times U \xrightarrow{D[r_{U\times U} D[f]]} U$$

satisfies

$$\langle\langle g, 0\rangle, \langle h, k\rangle\rangle D^2[f] = \langle g, k\rangle D[f],$$
$$\langle\langle 0, h\rangle, \langle g, k\rangle\rangle D^2[f] = \langle\langle 0, g\rangle, \langle h, k\rangle\rangle D^2[f],$$
$$((e_A \times e_A) \times (e_A \times e_A))D^2[f] = D^2[f].$$

Given a differentially coherent universal structure on a category, one can generate a Cartesian differential category.

Proposition 4.3. *If $\mathbb{X}$ is a category with chosen products and a commutative monoid $(U, +_U, 0_U)$ which has a differentially coherent universal structure, $\mathcal{U}$, then there is a Cartesian differential structure on $\mathbb{X}$ that makes each retraction in $\mathcal{U}$ linear.*

We first show that $\mathbb{X}$ has a Cartesian differential structure.
Note the following facts about $\mathcal{U}$ idempotents.

- For all $U \xrightarrow{f} U$, $D[fe_B]r_B = D[f]r_B$.
- For all $A \xrightarrow{f} B$, $(e_A \times e_A)D[r_A f s_B] = D[r_A f s_B]$.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

The differential operator $\mathcal{D}[_]$ is defined on maps $A \longrightarrow B$:

$$\frac{A \xrightarrow[]{f} B}{A \times A \xrightarrow[\mathcal{D}[f]]{} B} \quad = \quad \frac{\dfrac{A \xrightarrow{f} B}{U \xrightarrow[r_A]{} A \xrightarrow[f]{} B \xrightarrow[s_B]{} U}}{\dfrac{U \times U \xrightarrow[D[r_A f s_B]]{} U}{A \times A \xrightarrow[s_A \times s_A]{} U \times U \xrightarrow[D[r_A f s_B]]{} U \xrightarrow[r_B]{} B}} .$$

Proof. We must show that the derived map $\mathcal{D}[_]$ satisfies **[CD.1-7]**.

[CD.1] For the zero case observe that 0_{AB} is $!_A 0_U r_B$. Then note that

$$\begin{aligned} & D[r_A !_A 0_U r_B s_B] \\ &= D[!_U 0_U e_B] = \langle D[0_{UU}], \pi_1 0_{UU} \rangle \, D[e_B] \\ &= \langle 0_{UU}, \pi_1 0_{UU} \rangle \, D[e_B] = 0. \end{aligned}$$

So that by left additivity

$$(s_A \times s_A) D[r_A !_A 0_U r_B s_B] r_B = (s_A \times s_A) 0 r_B = 0 r_B$$

as required. For the $+$ case, we must show that $\mathcal{D}[(f s_B + g s_B) r_B] = (\mathcal{D}[f] s_B + \mathcal{D}[g] s_B) r_B$.

$$\begin{aligned} & (s_A \times s_A) D[r_A (f s_B + g s_B) r_B s_B] r_B \\ &= (s_A \times s_A) D[(r_A f s_B + r_A g s_B) e_B] r_B \\ &= (s_A \times s_A) D[r_A f s_B + r_A g s_B] r_B \\ &= ((s_A \times s_A) D[r_A f s_B] r_B s_B + (s_A \times s_A) D[g] r_B s_B) r_B \end{aligned}$$

as required

[CD.2] For the zero case,

$$\begin{aligned} & \langle !0_U r_A, g \rangle \, (s_A \times s_A) D[r_A f s_B] r_B \\ &= \langle !0_U, g s_A \rangle \, (e_A \times s_A e_A) D[r_A f s_B] r_B \\ &= \langle !0_u, g s_A \rangle \, D[r_A f s_B] r_B \\ &= 0 r_B \end{aligned}$$

as required. For the $+$ case

$$\begin{aligned} & \langle (f s_B + g s_B) r_B, k \rangle \, (s_B \times s_B) D[r_B h s_C] r_C \\ &= \langle f s_B + g s_B, k \rangle \, (e_B \times s_B e_B) D[r_B h s_C] r_C \\ &= \langle f s_B + g s_B, k s_B \rangle \, D[r_B h s_C] r_C \\ &= (\langle f s_B, k s_B \rangle \, D[r_B h s_C] + \langle g s_B, k s_B \rangle \, D[r_B h s_C]) r_C \\ &= (\langle f, k \rangle \, (s_B \times s_B) D[r_B h s_C] r_C s_C + \langle g, k \rangle (s_B \times s_B) D[r_B h s_C] r_C s_C) r_C \end{aligned}$$

as required.

[CD.3] The projections are **[UDC.3]**. $\mathcal{D}[1_A]$ is given by

$$\mathcal{D}[1_A] = (s_A \times s_A) D[r_A s_A] r_A = (s_A \times s_A) \pi_0 e_A r_A = \pi_0 s_A r_A = \pi_0.$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

[CD.4] Let $C \xrightarrow{f} A$ and $C \xrightarrow{g} B$.

$$\begin{aligned}
&\mathcal{D}[\langle f, g\rangle] \\
&\quad = (s_C \times s_C) D[r_C \langle f, g\rangle (s_A \times s_B) s_{U\times U}] r_{U\times U} (r_A \times r_B) \\
&\quad = (s_C \times s_C) D[\langle r_C f s_A, r_C g s_B\rangle s_{U\times U}] r_{U\times U} (r_A \times r_B) \\
&\quad = (s_C \times s_C) \langle D[r_C f s_A], D[r_C g s_B]\rangle (r_A \times r_B) \\
&\quad = \langle \mathcal{D}[f], \mathcal{D}[g]\rangle .
\end{aligned}$$

[CD.5] We must show $\mathcal{D}[fg] = \langle \mathcal{D}[f], \pi_1 f\rangle \mathcal{D}[g]$. In the second to last step, we are using that $(e \times e) D[rfs] = D[rfs]$

$$\begin{aligned}
&(s_A \times s_A) D[r_A f g s_C] r_C \\
&\quad = (s_A \times s_A) D[r_A f s_B r_B g s_C] r_C \\
&\quad = (s_A \times s_A) \langle D[r_A f s_B], \pi_1 r_A f s_B\rangle D[r_B g s_C] r_C \\
&\quad = \langle (s_A \times s_A) D[r_A f s_B], \pi_1 f s_B\rangle D[r_B g s_C] r_C \\
&\quad = \langle (s_A \times s_A) D[r_A f s_B], \pi_1 f\rangle (e_B \times s_B e_B) D[r_B g s_C] r_C \\
&\quad = \langle (s_A \times s_A) D[r_A f s_B] r_B, \pi_1 f\rangle (s_B \times s_B) D[r_B g s_C] r_C .
\end{aligned}$$

[CD.6] First we expand $\mathcal{D}[\mathcal{D}[f]]$. It is

$$\begin{aligned}
&\mathcal{D}[\mathcal{D}[f]] \\
&\quad = (s_{A\times A} \times s_{A\times A}) D[r_{A\times A} \mathcal{D}[f] s_B] r_B \\
&\quad = (s_{A\times A} \times s_{A\times A}) D[r_{A\times A} (s_A \times s_A) D[r_A f s_B] r_B s_B] r_B \\
&\quad = (s_{A\times A} \times s_{A\times A}) D[r_{U\times U} (r_A s_A \times r_A s_A) D[r_A f s_B] r_B s_B] r_B \\
&\quad = (s_{A\times A} \times s_{A\times A}) D[r_{U\times U} D[r_A f s_B]] r_B \\
&\quad = ((s_A \times s_A) \times (s_A \times s_A))(s_{U\times U} \times s_{U\times U}) D[r_{U\times U} D[r_A f s_B]] r_B .
\end{aligned}$$

And then

$$\begin{aligned}
&\langle\langle g, 0 r_A\rangle, \langle h, k\rangle\rangle \mathcal{D}[\mathcal{D}[f]] \\
&\quad = \langle\langle g, 0\rangle, \langle h, k\rangle\rangle ((1 \times r_A) \times (1 \times 1))((s_A \times s_A) \times (s_A \times s_A)) D^2[r_A f s_B] r_B \\
&\quad = \langle\langle g, 0\rangle, \langle h, k\rangle\rangle ((s_A e \times e) \times (s_A e \times s_A e)) D^2[r_A f s_B] r_B \\
&\quad = \langle\langle g s_A, 0\rangle, \langle h s_A, k s_A\rangle\rangle D^2[r_A f s_B] r_B \\
&\quad = \langle g s_A, k s_A\rangle D[r_A f s_B] r_B \\
&\quad = \mathcal{D}[f]
\end{aligned}$$

as required.

[CD.7]

$$\begin{aligned}
&\langle\langle 0 r_A, g\rangle, \langle h, k\rangle\rangle \mathcal{D}[\mathcal{D}[f]] \\
&\quad = \langle\langle 0, g\rangle, \langle h, k\rangle\rangle ((r_A \times 1) \times (1 \times 1))((s_A \times s_A) \times (s_A \times s_A)) D^2[r_A f s_B] r_B \\
&\quad = \langle\langle 0, g\rangle, \langle h, k\rangle\rangle ((e \times s_A e) \times (s_A e \times s_A e)) D^2[r_A f s_B] r_B \\
&\quad = \langle\langle 0, g s_A\rangle, \langle h s_A, k s_A\rangle\rangle D^2[r_A f s_B] r_B \\
&\quad = \langle\langle 0, h s_A\rangle, \langle g s_A, k s) A\rangle\rangle D^2[r_A f s_B] r_B \\
&\quad = \langle\langle 0 r_A, h\rangle, \langle g, k\rangle\rangle \mathcal{D}[\mathcal{D}[f]]
\end{aligned}$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

That $\mathcal{U}$ retractions are linear is given by $\mathcal{D}[r_A] = D[r_A s_A] r_A = \pi_0$. This completes the proof that $\mathbb{X}$ is a Cartesian differential category in which U is differentially universal. □

4.3 Differential Turing categories

A **differential Turing category** is a Turing category which is also a Cartesian differential category and where additionally each universal application $T \times B \xrightarrow{\bullet_{BC}} C$ is linear in its first argument.

The above definition has the property that if c_f is a code for f, then

$$\frac{\mathsf{d}\, f(x,y)}{\mathsf{d}x}\,(a)\cdot v = \frac{\mathsf{d}c_f(x)\, y}{\mathsf{d}x}\,(a)\cdot v = \left(\frac{\mathsf{d}c_f(x)}{\mathsf{d}x}\,(a)\cdot v\right) y$$

so that the derivative of f in its first argument is given by the derivative of c_f.

A differential Turing category $\mathbb{X}$ has **differential canonical codes** when $\mathbb{X}$ has additive canonical codes $\lambda(_)$ that satisfy in addition:

$$D[\lambda(f)] = \lambda(\langle \pi_0 \times 0, \pi_1 \times 1\rangle\, D[f])$$

The following is a recognition theorem for differential Turing categories.

Proposition 4.4. *A Cartesian differential category $\mathbb{X}$ is a differential Turing category if and only if*

(i) $\mathbb{X}$ has a differentially universal object T;

(ii) There is a map $T \times T \xrightarrow{\bullet} T$, which is linear in its first argument, that is universal for maps $T \times T \to T$.

Furthermore, $\mathbb{X}$ is a differential Turing category with differential canonical codes if and only if (i) and (ii) hold and $T \times T \xrightarrow{\bullet} T$ has differential canonical codes.

Proof. We will first show that if $\mathbb{X}$ is a differential Turing category, then T is a differential universal object T – i.e. for each object A, the retraction $T \xrightarrow{\langle 1,!\rangle \bullet_{1A}} A$ is linear. First, note that $(1\times!) = \langle \pi_0, \pi_1 ! \rangle = \langle \pi_0, \pi_0 ! \rangle = \pi_0\, \langle 1, ! \rangle$. Then,

$$\begin{aligned} D[\,\langle 1,!\rangle \bullet_{1A}\,] &= (\,\langle 1,!\rangle \times \langle 1,!\rangle\,) D[\,\bullet_{1A}\,] \\ &= (1 \times \langle 1,!\rangle\,)(\,\langle 1,0\rangle \times 1) D[\,\bullet_{1A}\,] = (1 \times \langle 1,!\rangle\,)(1 \times \pi_1)\bullet_{1A} \\ &= (1\times!)\bullet_{1A} = \pi_0\, \langle 1,!\rangle \bullet_{1A}\,. \end{aligned}$$

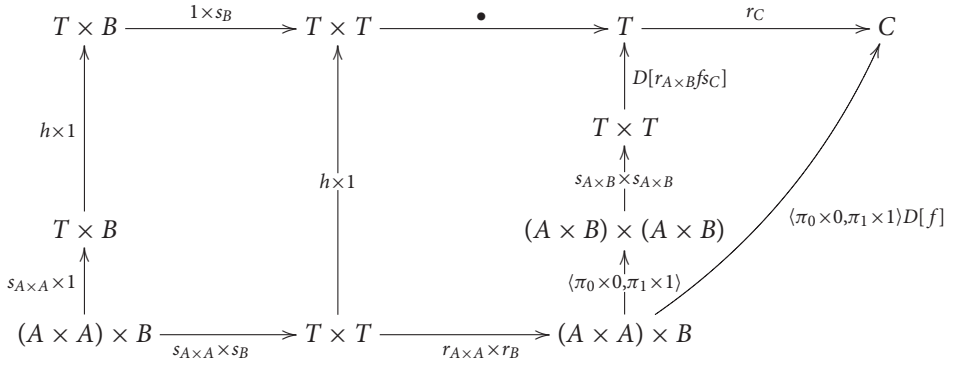
For the converse, suppose that $\mathbb{X}$ has a differentially universal object and that $T \times T \xrightarrow{\bullet} T$ is universal and linear in its first argument. We must show that the derived application, $T \times B \xrightarrow{1\times s} T \times T \xrightarrow{\bullet} T \xrightarrow{r} C$, is linear in its first argument:

$$\begin{aligned} (\,\langle 1,0\rangle \times 1) D[(1 \times s) \bullet r] &= (\,\langle 1,0\rangle \times 1)\, \langle D[1 \times s], \pi_1(1 \times s)\rangle\, D[\,\bullet r] \\ &= (\,\langle 1,0\rangle \times 1)\, \langle \langle \pi_0\pi_0, (\pi_1 \times \pi_1) D[s]\rangle\,, \pi_1(1 \times s)\rangle\, D[\,\bullet r] \\ &= \langle \langle \pi_0, \langle 0, \pi_1\pi_1\rangle\, D[s]\rangle\,, \pi_1(1 \times s)\rangle\, D[\,\bullet r] = \langle \langle \pi_0, 0\rangle\,, \pi_1(1 \times s)\rangle\, D[\,\bullet r] \\ &= \langle \pi_0, \pi_1(1 \times s)\rangle\, (\,\langle 1,0\rangle \times 1) D[\,\bullet r] = \langle \pi_0, \pi_1(1 \times s)\rangle\, (\,\langle 1,0\rangle \times 1) D[\,\bullet\,] r \\ &= \langle \pi_0, \pi_1(1 \times s)\rangle\, (1 \times \pi_1) \bullet r = (1 \times \pi_1)(1 \times s) \bullet r. \end{aligned}$$

Therefore, from a differentially universal object and a universal application $T \times T \xrightarrow{\bullet} T$ that is linear in its first argument, we obtain a differential Turing category.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Next, suppose that $T \times T \xrightarrow{\bullet} T$ has differential canonical codes. Then the derived canonical code on a linear retract is a differential canonical code too. To see this, first consider the canonical code for $\langle \pi_0 \times 0, \pi_1 \times 1 \rangle\, D[f]$, where $f : A \times B \to C$, as in the following diagram:



where $sh := \lambda((1 \times r)\, \langle \pi_0 \times 0, \pi_1 \times 1 \rangle\, (s_{A\times B} \times s_{A\times B}) D[r_A f s_B])$ is the derived canonical code for $\langle \pi_0 \times 0, \pi_1 \times 1 \rangle\, D[f]$. Thus, we must show that $sh = D[\lambda(f)]$ for $\lambda(f)$, the derived canonical code for f:

$$\begin{aligned}
& s_{A\times A} \lambda((r_{A\times A} \times r_B)\, \langle \pi_0 \times 0, \pi_1 \times 1 \rangle\, (s_{A\times B} \times s_{A\times B}) D[r_{A\times B} f s_C]) \\
&\quad = \lambda((1 \times r_B)\, \langle \pi_0 \times 0, \pi_1 \times 1 \rangle\, D[f s_C]) \\
&\quad = \lambda(\, \langle \pi_0 \times r_B 0, \pi_1 \times r_B \rangle\, D[f s_C]) \\
&\quad = \lambda(\, \langle \pi_0 \times 0 r_B, \pi_1 \times r_B \rangle\, D[f s_C]) \qquad (r \text{ is additive}) \\
&\quad = \lambda(\, \langle \pi_0 \times 0, \pi_1 \times 1 \rangle\, ((1 \times r_B) \times (1 \times r_B)) D[f s_C]) \\
&\quad = \lambda(\, \langle \pi_0 \times 0, \pi_1 \times 1 \rangle\, D[(1 \times r_B) f s_C]) \qquad (r \text{ is linear}) \\
&\quad = D[\lambda((1 \times r_B) f s_C)] \qquad (\text{differential canonicity})
\end{aligned}$$

Thus the derived application has differential canonical codes. This completes the proof. □

For completeness, we show that $\mathcal{C}(\Lambda_{\beta\partial})$ is a differential Turing category with differentially canonical codes. We will first show that $\mathcal{C}(\Lambda_{\beta\partial})$ is a Cartesian differential category in which U is a differentially universal object using Proposition 4.3.

The differential structure on $\mathcal{C}(\Lambda_{\beta\partial})$ is defined as follows:

$$\begin{aligned}
D[p : S \vdash m : 1] &:= () \\
D[p : S \vdash m : U] &:= (v, p) : S \times S \vdash \frac{\mathrm{d}m}{\mathrm{d}p}\,(p) \cdot v : U \\
D[p : S \vdash (m_1, m_2) : R \times T] &:= \left\langle D[p : S \vdash m_1 : R], D[p : S \vdash m_2 : T] \right\rangle .
\end{aligned}$$

Proposition 4.5. *$\mathcal{C}(\Lambda_{\beta\partial})$ is a Cartesian differential category in which U is a differentially universal object.*

Proof. In Blute et al. (2009), Section 4.4, it is shown that the classifying category is a Cartesian differential category. Thus, the differential that we defined on $\mathcal{C}(\Lambda_{\beta\partial})$ satisfies **[CD.1–7]**. We still must show that U is a differentially universal object.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

UAC.1–4 were shown in Theorem 3.1. **[UDC.1]** is immediate as in fact the differential structure satisfies **[CD.1–7]**. Next, **[UDC.2–4]** follow from observing that if $r_{U\times U}$ is linear then all the derived retractions r_A are linear. Recall

$$r_{U\times U} := a \vdash (a\,\lambda xy.x, a\,\lambda xy.y).$$

Then $r_{U\times U}$ is linear as

$$\begin{aligned} D[r_{U\times U}] &= D[a \vdash (a\lambda xy.x, a\lambda xy.y)] = \langle D[a \vdash a\lambda xy.x], D[a \vdash a\lambda xy.y]\rangle \\ &= \left\langle (v,a) \vdash \frac{\mathsf{d}a\lambda xy.x}{\mathsf{d}a}\,(a)\cdot v, (v,a) \vdash \frac{\mathsf{d}a\lambda xy.y}{\mathsf{d}a}\,(a)\cdot v \right\rangle \\ &= \langle (v,a) \vdash v\lambda xy.x, (v,a) \vdash v\lambda xy.y\rangle \qquad \text{Lemma 4.3.4} \\ &= \pi_0 r_{U\times U}. \end{aligned}$$

Then **[UDC.2–4]** are all straightforward and use the properties of these linear retractions into and out of the differential. □

Now we show that $\mathcal{C}(\Lambda_{\beta\partial})$ is a differential Turing category with canonical codes.

Theorem 4.1. *$\mathcal{C}(\Lambda_{\beta\partial})$ is a differential Turing category with differentially canonical codes.*

Proof. By Theorem 3.1, $\mathcal{C}(\Lambda_{\beta\partial})$ is a left-additive Turing category with additively canonical codes and Turing object U. By Proposition 4.5, $\mathcal{C}(\Lambda_{\beta\partial})$ is a Cartesian differential category in which U is a differentially universal object.

It remains to show that $\bullet$ is linear in its first argument, and that the canonical codes are differentially canonical.

First, consider $D[(a,b) \vdash ab]$:

$$\begin{aligned} D[(a,b) \vdash ab] &= ((v_1,v_2),(a,b)) \vdash \frac{\mathsf{d}ab}{\mathsf{d}(a,b)}\,((a,b))\cdot(v_1,v_2) \\ &= ((v_1,v_2),(a,b)) \vdash \left(\frac{\mathsf{d}a}{\mathsf{d}(a,b)}\,((a,b))\cdot(v_1,v_2)\right) b + \frac{\mathsf{d}az_1}{\mathsf{d}z_1}\,(b)\cdot\left(\frac{\mathsf{d}b}{\mathsf{d}(a,b)}\,((a,b))\cdot(v_1,v_2)\right) \\ &\quad \text{the above by Lemma 4.3.5} \\ &= ((v_1,v_2),(a,b)) \vdash \left(\frac{\mathsf{d}a}{\mathsf{d}a}\,(a)\cdot v_1\right) b + \frac{\mathsf{d}az_1}{\mathsf{d}z_1}\,(b)\cdot\left(\frac{\mathsf{d}b}{\mathsf{d}b}\,(b)\cdot v_2\right) \qquad \textbf{Dt.3}, \text{ Lemma 4.3.1} \\ &= ((v_1,v_2),(a,b)) \vdash v_1 b + \frac{\mathsf{d}az_1}{\mathsf{d}z_1}\,(b)\cdot v_2 \qquad \textbf{Dt.3} \end{aligned}$$

so that

$$\begin{aligned} &(\langle 1,0\rangle \times 1)D[(a,b) \vdash ab] \\ &\quad = ((u,(x,y)) \vdash ((u,0),(x,y)))(((v_1,v_2),(a,b)) \vdash v_1 b + \frac{\mathsf{d}az_1}{\mathsf{d}z_1}\,(b)\cdot v_2) \\ &\quad = (u,(x,y)) \vdash uy + \frac{\mathsf{d}xz_1}{\mathsf{d}z_1}\,(y)\cdot 0 \\ &\quad = (u,(x,y)) \vdash uy = (\pi_1 \times 1)\bullet \end{aligned}$$

and we have that $\bullet$ is linear in its first argument.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Now, we show that our canonical codes $\lambda((x,y)\vdash t) := x \vdash \lambda y.t$ are differentially canonical.

$$\begin{aligned}
&\lambda(\ \langle \pi_0 \times 0, \pi_1 \times 1\rangle\ D[(x,y)\vdash t]\\
&\quad = \lambda[((v,x),y)\vdash ((v,0),(x,y));((v,w),(x,y))\vdash \frac{\mathsf{d}t}{\mathsf{d}(x,y)}\left((x,y)\right)\cdot(v,w)]\\
&\quad = \lambda[((v,x),y)\vdash \frac{\mathsf{d}t}{\mathsf{d}(x,y)}\left((x,y)\right)\cdot(v,0)]\\
&\quad = \lambda[((v,x),y)\vdash \frac{\mathsf{d}t}{\mathsf{d}x}\,(x)\cdot v] = (v,x)\vdash \lambda y.\frac{\mathsf{d}t}{\mathsf{d}x}\,(x)\cdot v\\
&\quad = (v,x)\vdash \frac{\mathsf{d}\lambda y.t}{\mathsf{d}x}\,(x)\cdot v = D[x\vdash \lambda y.t] = D[\lambda((x,y)\vdash t)]
\end{aligned}$$

therefore, $\mathcal{C}(\Lambda_{\beta\partial})$ is a differential Turing category with canonical codes. □

4.4 Interpreting the differential λ-calculus

Let $\mathbb{X}$ be a differential Turing category with differential canonical codes. We will show that there is a functor $\mathcal{C}(\Lambda_{\beta\partial}) \longrightarrow \mathbb{X}$ that preserves all the differential structure and the canonical differential Turing structure. The functor $[\![_]\!] : \mathcal{C}(\Lambda_{\beta\partial}) \longrightarrow \mathbb{X}$ is defined in the same way as before for variables, applications, abstractions, sums, 0, and tuples of terms. For the differentials,

$$\left[\!\!\left[q\vdash \frac{\mathsf{d}m}{\mathsf{d}p}\,(a)\cdot v\right]\!\!\right] := \left\langle\left\langle [\![q\vdash u]\!], 0\right\rangle, \left\langle [\![q\vdash a]\!], 1\right\rangle\right\rangle D\left[[\![(p,q)\vdash t]\!]\right].$$

Lemma 4.5. Substitution lemma. *For* $[\![_]\!] : \mathcal{C}(\Lambda_{\beta\partial}) \longrightarrow \mathbb{X}$, *where* $\mathbb{X}$ *is a differential Turing category with differential canonical codes:*

$$[\![p\vdash m[n/q]]\!] = [\![p\vdash n]\!][\![q\vdash m]\!]$$

Proof. The proof is by induction on the structure of m. We cover the case when $m = \frac{\mathsf{d}m}{\mathsf{d}p}\,(a)\cdot v$.

$$\begin{aligned}
\left[\!\!\left[p_0\vdash \frac{\mathsf{d}m}{\mathsf{d}p}\,(a)\cdot v[n/q]\right]\!\!\right] &= \left[\!\!\left[\frac{\mathsf{d}m[n/q]}{\mathsf{d}p}\left(a[n/q]\right)\cdot v[n/q]\right]\!\!\right]_{\Gamma}\\
&= \left\langle\left\langle [\![p_0\vdash v[n/q]]\!], 0\right\rangle, \left\langle [\![p_0\vdash a[n/q]]\!], 1\right\rangle\right\rangle D\left[[\![(p,p_0)\vdash m[n/q]]\!]\right].
\end{aligned}$$

Consider briefly just the term $D\left[[\![(p,p_0)\vdash m[n/q]]\!]\right]$:

$$\begin{aligned}
D\left[[\![(p,p_0)\vdash m[n/q]]\!]\right] &= D\left[[\![(p,p_0)\vdash m[(p,n)/(p,q)]]\!]\right]\\
&= D\left[[\![(p,p_0)\vdash (p,n)]\!][\![(p,q)\vdash m]\!]\right]\\
&= \left\langle D\left[[\![(p,p_0)\vdash (p,n)]\!]\right], \pi_1[\![(p,p_0)\vdash (p,n)]\!]\right\rangle D[(p,p_0)\vdash [\![m]\!]]\\
&= \left\langle\left\langle D\left[[\![(p,p_0)\vdash p]\!]\right], D\left[[\![(p,p_0)\vdash n]\!]\right]\right\rangle, \pi_1\left\langle [\![(p,p_0)\vdash p]\!], [\![n]\!]_{(p,\Gamma)}\right\rangle\right\rangle D\left[[\![(p,q)\vdash m]\!]\right]\\
&= \left\langle\left\langle \pi_0\pi_0, (\pi_1\times\pi_1)D\left[[\![p_0\vdash n]\!]\right]\right\rangle, \pi_1\left\langle \pi_0, \pi_1[\![p_0\vdash n]\!]\right\rangle\right\rangle D\left[[\![(p,q)\vdash m]\!]\right].
\end{aligned}$$

Then continue the first calculation:

$$\begin{aligned}
&\left\langle\left\langle [\![p_0\vdash v[n/q]]\!], 0\right\rangle, \left\langle [\![p_0\vdash a[n/q]]\!], 1\right\rangle\right\rangle D\left[[\![(p,p_0)\vdash m[n/q]]\!]\right.\\
&\quad = \left\langle\left\langle [\![p_0\vdash v[n/q]]\!], \langle 0,1\rangle\, D\left[[\![p_0\vdash n]\!]\right]\right\rangle, \left\langle [\![p_0\vdash a[n/q]]\!], [\![p_0\vdash n]\!]\right\rangle\right\rangle D\left[[\![(p,p_0)\vdash m]\!]\right]
\end{aligned}$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

$$
\begin{aligned}
&= \langle\langle [\![n]\!]_\Gamma [\![v]\!]_\Delta, 0\rangle, \langle [\![n]\!]_\Gamma [\![a]\!]_\Delta, [\![n]\!]_\Gamma\rangle\rangle\, D\left[[\![m]\!]_{(p,\Delta)}\right] \\
&= \langle\langle [\![p_0 \vdash n]\!][\![q \vdash v]\!], [\![p_0 \vdash n]\!]0\rangle, \langle [\![p_0 \vdash n]\!][\![q \vdash a]\!], [\![p_0 \vdash n]\!]\rangle\rangle\, D\left[[\![(p,q) \vdash m]\!]\right] \\
&= [\![n]\!]_\Gamma \langle\langle [\![v]\!]_\Delta, 0\rangle, \langle [\![a]\!]_\Delta, 1\rangle\rangle\, D\left[[\![m]\!]_{(p,\Delta)}\right] \\
&= [\![p_0 \vdash n]\!]\left[\!\!\left[q \vdash \frac{\mathsf{d}m}{\mathsf{d}p}\,(a)\cdot v\right]\!\!\right].
\end{aligned}
$$

□

It remains to show that $[\![_]\!]$ is well defined in order to obtain the following.

Proposition 4.6. *When $\mathbb{X}$ is a differential Turing category with differentially canonical codes, then*

$$[\![_]\!] : \mathcal{C}(\Lambda_{\beta\partial}) \to \mathbb{X}$$

is a Cartesian differential functor.

Proof. We must show that each equation of **[Dt.1]**-**[Dt.8]** is preserved by $[\![\ ]\!]$. For **[Dt.1]**–**[Dt.6]**, we refer the reader to Blute et al. (2009), Proposition 4.3.1, as the proofs are long and exactly the same. We consider the new equations **[Dt.7]**,**[Dt.8]** here.

For the left-hand side of **[Dt.7]**,

$$
\begin{aligned}
\left[\!\!\left[q \vdash \lambda y.\frac{\mathsf{d}m}{\mathsf{d}p}\,(a)\cdot v\right]\!\!\right] &= \lambda\left(\left[\!\!\left[(q,y) \vdash \frac{\mathsf{d}m}{\mathsf{d}p}\,(a)\cdot v\right]\!\!\right]\right) \\
&= \lambda\left(\langle\langle [\![(q,y) \vdash v]\!], 0\rangle, \langle [\![(q,y) \vdash a]\!], 1\rangle\rangle\, D\left[[\![(p,(q,y)) \vdash m]\!]\right]\right).
\end{aligned}
$$

And for the right-hand side of **[Dt.7]**, we have

$$
\begin{aligned}
&\left[\!\!\left[q \vdash \frac{\mathsf{d}\lambda y.m}{\mathsf{d}p}\,(a)\cdot v\right]\!\!\right] \\
&= \langle\langle [\![q \vdash v]\!], 0\rangle, \langle [\![q \vdash a]\!], 1\rangle\rangle\, D\left[[\![(p,q) \vdash \lambda y.m]\!]\right] \\
&= \langle\langle [\![q \vdash v]\!], 0\rangle, \langle [\![q \vdash a]\!], 1\rangle\rangle\, D\left[\lambda([\![((p,q),y) \vdash m]\!])\right] \\
&= \langle\langle [\![q \vdash v]\!], 0\rangle, \langle [\![q \vdash a]\!], 1\rangle\rangle\, \lambda\left(\langle \pi_0 \times 0, \pi_1 \times 1\rangle\, D\left[[\![((p,q),y) \vdash m]\!]\right]\right) \\
&\quad \text{(by differential canonicity)} \\
&= \lambda\left(\left(\langle\langle [\![q \vdash v]\!], 0\rangle, \langle [\![q \vdash a]\!], 1\rangle\rangle \times 1\right)\langle \pi_0 \times 0, \pi_1 \times 1\rangle\, D\left[[\![((p,q),y) \vdash m]\!]\right]\right). \\
&\quad \text{(by canonicity)}
\end{aligned}
$$

Now, it suffices to prove that the terms inside of the $\lambda(\ )$ are the same, i.e.

$$
\begin{aligned}
&\left(\langle\langle [\![q \vdash v]\!], 0\rangle, \langle [\![q \vdash a]\!], 1\rangle\rangle \times 1\right)\langle \pi_0 \times 0, \pi_1 \times 1\rangle\, D\left[[\![((p,q),y) \vdash m]\!]\right] \\
&\quad = \langle\langle [\![(q,y) \vdash v]\!], 0\rangle, \langle [\![(q,y) \vdash a]\!], 1\rangle\rangle\, D\left[[\![(p,(q,y)) \vdash m]\!]\right],
\end{aligned}
$$

which we show in the following calculation. But first let $a_\times : (A \times B) \times C \to A \times (B \times C)$; note that $a_\times$ is linear and $(\langle f, g\rangle \times h)a_\times = \langle \pi_0 f, g \times h\rangle$:

$$
\begin{aligned}
&\left(\langle\langle [\![q \vdash v]\!], 0\rangle, \langle [\![q \vdash a]\!], 1\rangle\rangle \times 1\right)\langle \pi_0 \times 0, \pi_1 \times 1\rangle\, D\left[[\![((p,q),y) \vdash m]\!]\right] \\
&\quad = \left\langle\left(\langle [\![q \vdash v]\!], 0\rangle \times 0\right), \left(\langle [\![q \vdash a]\!], 1\rangle \times 1\right)\right\rangle D\left[[\![((p,q),y) \vdash m]\!]\right] \\
&\quad = \left\langle\left(\langle [\![q \vdash v]\!], 0\rangle \times 0\right), \left(\langle [\![q \vdash a]\!], 1\rangle \times 1\right)\right\rangle D\left[a_\times [\![(p,(q,y)) \vdash m]\!]\right]
\end{aligned}
$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

$$= \langle (\langle [\![q \vdash v]\!], 0 \rangle \times 0), (\langle [\![q \vdash a]\!], 1 \rangle \times 1) \rangle (a_\times \times a_\times) D \left[[\![(p, (q, y)) \vdash m]\!] \right]$$
$$= \langle \langle \pi_0 [\![q \vdash v]\!], 0 \times 0 \rangle \langle \pi_0 [\![q \vdash a]\!], 1 \times 1 \rangle \rangle D \left[[\![(p, (q, y)) \vdash m]\!] \right]$$
$$= \langle \langle \pi_0 [\![q \vdash v]\!], 0 \rangle \langle \pi_0 [\![q \vdash a]\!], 1 \rangle \rangle D \left[[\![(p, (q, y)) \vdash m]\!] \right]$$
$$= \langle \langle [\![(q, y) \vdash v]\!], 0 \rangle \langle [\![(q, y) \vdash a]\!], 1 \rangle \rangle D \left[[\![(p, (q, y)) \vdash m]\!] \right].$$

This completes the calculation for **[Dt.7]**. For **[Dt.8]**,

$$\left[\!\!\left[q \vdash \frac{\mathsf{d}\lambda y_0.yy_0}{\mathsf{d}y}\,(a) \cdot v \right]\!\!\right]$$
$$= \langle \langle [\![q \vdash v]\!], 0 \rangle, \langle [\![q \vdash a]\!], 1 \rangle \rangle D \left[[\![(y, q) \vdash \lambda y_0.yy_0]\!] \right]$$
$$= \langle \langle [\![q \vdash v]\!], 0 \rangle, \langle [\![q \vdash a]\!], 1 \rangle \rangle D[\lambda([\![((y, q), y_0) \vdash yy_0]\!])]$$
$$= \langle \langle [\![q \vdash v]\!], 0 \rangle, \langle [\![q \vdash a]\!], 1 \rangle \rangle D[\lambda((\pi_0 \times 1) \bullet)]$$
$$= \langle \langle [\![q \vdash v]\!], 0 \rangle, \langle [\![q \vdash a]\!], 1 \rangle \rangle D[\pi_0 \lambda(\bullet)]$$
$$= \langle \langle [\![q \vdash v]\!], 0 \rangle, \langle [\![q \vdash a]\!], 1 \rangle \rangle (\pi_0 \times \pi_0) D[\lambda(\bullet)]$$
$$= \langle [\![q \vdash b]\!], [\![q \vdash a]\!] \rangle D[\lambda(\bullet)]$$
$$= \langle [\![q \vdash b]\!], [\![q \vdash a]\!] \rangle \lambda(\langle \pi_0 \times 0, \pi_1 \times 1 \rangle D[\bullet]) \qquad \text{(Canonical codes)}$$
$$= \lambda((\langle [\![q \vdash b]\!], [\![q \vdash a]\!] \rangle \times 1) \langle \pi_0 \times 0, \pi_1 \times 1 \rangle D[\bullet])$$
$$= \lambda(\langle [\![q \vdash v]\!] \times 0, [\![q \vdash a]\!] \times 1 \rangle D[\bullet])$$
$$= \lambda(\langle \langle \pi_0 [\![q \vdash v]\!], \pi_0 [\![q \vdash v]\!] 0 \rangle, [\![q \vdash a]\!] \times 1 \rangle D[\bullet])$$
$$= \lambda(\langle \pi_0 [\![q \vdash v]\!] \langle 1, 0 \rangle, [\![q \vdash a]\!] \times 1 \rangle D[\bullet])$$
$$= \lambda(\langle \pi_0 [\![q \vdash v]\!], [\![q \vdash a]\!] \times 1 \rangle (\langle 1, 0 \rangle \times 1) D[\bullet])$$
$$= \lambda(\langle \pi_0 [\![q \vdash v]\!], [\![q \vdash a]\!] \times 1 \rangle (1 \times \pi_1) \bullet) \qquad \text{(Application is linear in its 1st argument}$$
$$= \lambda(\langle \pi_0 [\![q \vdash v]\!], \pi_1 \rangle \bullet) = \lambda(\langle [\![(q, y_0) \vdash v]\!], [\![(q, y_0) \vdash y_0]\!] \rangle \bullet)$$
$$= \lambda([\![(q, y_0) \vdash vy_0]\!]) = [\![q \vdash \lambda y_0.vy_0]\!].$$

These calculations complete the proof. □

4.5 Splitting the retractively linear idempotents

An idempotent $A \xrightarrow{e} A$ is **retractively linear** when it is retractively additive and further

$$D[e]e = \pi_0 e \qquad (e \times e) D[e] = D[e].$$

Lemma 4.6. *The following hold in any Cartesian differential category:*

(i) *If e is a linear idempotent, then e is retractively linear (so all identities are retractively linear).*
(ii) *If e, e' are retractively linear, then so is $e \times e'$.*

A split idempotent $e = rs$ splits **left linearly** when r is linear. Note the following.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Lemma 4.7. *In any Cartesian differential category, if $e = rs$ splits left linearly then*

(i) *e is retractively linear;*
(ii) *e is linear if and only if s is linear.*

Proof. Let $e = rs$ split left linearly.

(i) First

$$D[rs]rs = D[rsr]s = D[r]s = \pi_0 rs$$

and second

$$(rs \times rs)D[rs] = (rs \times rs)(r \times r)D[s] = (r \times r)D[s] = D[rs].$$

(ii) If s is linear, then rs is linear as linear maps compose. Conversely, assume rs is linear

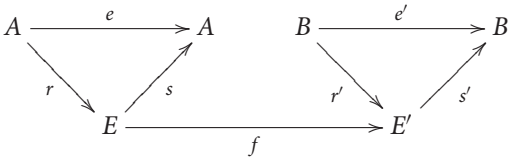
$$D[s] = D[srs] = D[s]rs = D[sr]s = \pi_0 s$$

as required.

□

The above lemma shows that every left linearly split idempotent is retractively linear. In general, however, a retractively linear idempotent need not split. A Cartesian differential category is a **split Cartesian differential category** when every retractively linear idempotent is left linearly split.

Consider split idempotents e, e' in the following diagram:

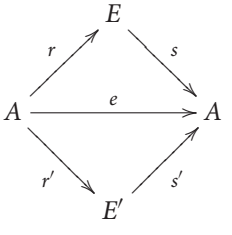


One can simply take the derivative of f to obtain $D[f]$. However, one can compute $D[rfs']$ and precompose with $(s \times s)$ and postcompose with r'. If e is left linearly split, then it splits the derivative in the following sense.

Lemma 4.8. *If rs and rs' are left linearly split idempotents, then $D[f] = (s \times s)D[rfs']r'$.*

Left linear splittings are unique up to a linear isomorphism.

Lemma 4.9. *If*



commutes with r, r' linear, and $sr = 1$ and $s'r' = 1$, then there is a unique isomorphism $E \xrightarrow{\alpha} E'$ such that $r\alpha = r'$ and $\alpha s' = s$. Moreover, α is linear.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Proof. To see that $\alpha := sr'$ is linear:

$$\begin{aligned} D[\alpha] &= (s \times s)D[r\alpha s']r' \\ &= (s \times s)D[r's']r' \\ &= (s \times s)D[r's'r'] \\ &= (s \times s)D[r'] \\ &= \pi_0 sr'. \end{aligned}$$

□

We now give a construction that completes a Cartesian differential category to be a split Cartesian differential category. Let $\mathbb{X}$ be a Cartesian differential category and $\mathcal{E}$ a product-closed collection of retractively linear idempotents. Lemma 4.8 tells us that for $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$ to be a split Cartesian differential category, if $e \xrightarrow{f} e'$ then the differential structure must be

$$D[f] := (e \times e)D[efe']e' = (e \times e)D[f]e'.$$

In fact, $(e \times e)D[f]e' = D[f]e'$ as

$$\begin{aligned} (e \times e)D[f]e' &= (e \times e)\,\langle D[e], \pi_1 e\rangle\, D[f]e' \\ &= \langle (e \times e)D[e], (e \times e)\pi_1 e\rangle\, D[f]e' \\ &= \langle D[e], \pi_1 e\rangle\, D[f]e' = D[f]e'. \end{aligned}$$

We know that $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ has left-additive structure; denote $\mathsf{Split}_{\mathcal{E}_+}(\mathbb{X})$ with the above differential structure as $\mathsf{Split}_{\mathcal{E}_\mathcal{D}}(\mathbb{X})$.

Proposition 4.7. *Let $\mathbb{X}$ be a Cartesian differential category and $\mathcal{E}$ a product-closed collection of retractively linear idempotents. Then $\mathsf{Split}_{\mathcal{E}_\mathcal{D}}(\mathbb{X})$ is a Cartesian differential category.*

Proof. We show the axioms of a Cartesian differential category hold.

[CD.1] To recall the typing, if $f, g : e \to e'$ then $(f + g) : e \to e'$. Then the required identity is $(e \times e)D[(f + g)e']e' = ((e \times e)D[f]e' + (e \times e)D[g]e')e'$.

$$\begin{aligned} &(e \times e)D[(f + g)e']e' \\ &= (e \times e)\left\langle D[f + g], \pi_1(f + g)\right\rangle D[e']e' \\ &= (e \times e)\left\langle D[f + g], \pi_1(f + g)\right\rangle \pi_0 e' \\ &= (e \times e)D[f + g]e' \\ &= (e \times e)(D[f]e' + D[g]e')e' \\ &= ((e \times e)D[f]e' + (e \times e)D[g]e')e'. \end{aligned}$$

The case for 0 holds similarly.

[CD.2] Consider

$$\begin{aligned} &\left\langle (h + k)e, g\right\rangle (e \times e)D[f]e' \\ &= \left\langle h + k, g\right\rangle (e \times e)D[f]e' \\ &= \left\langle h + k, g\right\rangle (e \times e)\,\langle D[e], \pi_1 e\rangle\, D[f]e' \\ &= \left\langle h + k, g\right\rangle \langle D[e], \pi_1 e\rangle\, D[f]e' \end{aligned}$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

$$= \langle h + k, g \rangle D[f]e'$$
$$= (\langle h, g \rangle D[f]e' + \langle k, g \rangle D[f]e')e'$$
$$= (\langle h, g \rangle (e \times e) D[f]e' + \langle k, g \rangle (e \times e) D[f]e')e' \qquad k = ke, g = ge, h = he.$$

The case for 0 holds similarly.

[CD.3] Consider the calculation for π_0.

$$\begin{aligned}
&((e \times e') \times (e \times e'))D[(e \times e')\pi_0]e \\
&\quad = ((e \times e') \times (e \times e')) \langle D[e \times e'], \pi(e \times e') \rangle \pi_0 \pi_0 e \\
&\quad = ((e \times e') \times (e \times e'))D[e \times e']\pi_0 e \\
&\quad = ((e \times e') \times (e \times e'))D[\pi_0 e]e \qquad\qquad = ((e \times e') \times (e \times e'))(\pi_0 \times \pi_0)D[e]e \\
&\quad = ((e \times e') \times (e \times e'))(\pi_0 \times \pi_0)\pi_0 e \\
&\quad = ((e \times e') \times (e \times e'))(\pi_0 \times \pi_0)\pi_0 .[2pt]
\end{aligned}$$

[CD.4] is obvious.

[CD.5] Consider,

$$\begin{aligned}
&\langle (e_1 \times e_1)D[f]e_2, (e_1 \times e_1)\pi_1 f \rangle (e_2 \times e_2)D[g]e_3 \\
&\quad = (e_1 \times e_1) \langle D[f], \pi_1 f \rangle (e_2 \times e_2)D[g]e_3 \\
&\quad = (e_1 \times e_1) \langle D[f], \pi_1 f \rangle (e_2 \times e_2) \langle D[e], \pi_1 e \rangle D[g]e_3 \\
&\quad = (e_1 \times e_1) \langle D[f], \pi_1 f \rangle D[g]e_3 \\
&\quad = (e_1 \times e_1)D[fg]e_3 .
\end{aligned}$$

[CD.6] This follows immediately from the observation that

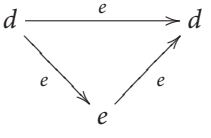
$$\langle \langle g, 0e \rangle, \langle h, k \rangle \rangle (e \times e \times e \times e)D^2[f] = \langle \langle g, 0 \rangle, \langle h, k \rangle \rangle D^2[f].$$

[CD.7] This follows immediately from a similar observation to the previous.

□

Corollary 4.1. Let $\mathbb{X}$ be a Cartesian differential category, and let $\mathcal{E}$ be the collection of retractively linear idempotents. Then $\mathsf{Split}_{\mathcal{E}_\mathcal{D}}(\mathbb{X})$ is a split Cartesian differential category.

Proof. It remains to show that all retractively linear idempotents split left additively. Let $e \in \mathcal{E}$ and suppose we view e as a map $d \xrightarrow{e} d$. Then e splits



We must show that $d \xrightarrow{e} e$ is linear. Then,

$$(d \times d)D[e]e = (d \times d)\pi_0 e = \pi_0 de = \pi_0 e = (e \times e)\pi_0$$

as required. Thus $d \xrightarrow{e} e$ is linear. □

As an easy second corollary, we have the following.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Corollary 4.2. The embedding $\mathbb{X} \hookrightarrow \mathsf{Split}_{\mathcal{E}_\mathcal{D}}(\mathbb{X})$ is a Cartesian differential functor.

We now show an analog of Proposition 3.8. When $\mathcal{E}$ is the class of retractively linear idempotents of a Turing category $\mathbb{X}$, $\mathsf{Split}_{\mathcal{E}_\mathcal{D}}(\mathbb{X})$ again a Turing category, and this is because $\mathcal{E}$ contains the identities. From Corollary 4.1, it is also a split Cartesian differential category. We will show that it is still a differential Turing category as well.

Proposition 4.8. *For any differential Turing category $\mathbb{X}$, when $\mathcal{E}$ is the retractively linear idempotents, then $\mathsf{Split}_{\mathcal{E}_\mathcal{D}}(\mathbb{X})$ is a differential Turing category in which every retractively linear idempotent splits left linearly.*

Proof. We need to prove that for each pair of idempotents e, d, the application map $1_T \times e \xrightarrow{\bullet} d$ is linear in its first argument. Here is the calculation:

$$\begin{aligned}(\langle 1, 0\rangle \times e)D[\bullet]d &= (1 \times e)(\langle 1, 0\rangle \times 1)D[\bullet]d \\ &= (1 \times e)(1 \times \pi_1)D[\bullet]d \\ &= (1 \times (1 \times e)\pi_1)D[\bullet]d\end{aligned}$$

as required. □

4.6 Differential reflexive objects

A Cartesian-closed category that is also a Cartesian differential category is a **Cartesian-closed differential category** when

$$D[\lambda(f)] = \lambda(\langle \pi_0 \times 0, \pi_1 \times 1\rangle\, D[f]).$$

This is the differential currying identity: in the term logic, this equation is written as

$$\frac{\mathsf{d}\lambda x\,.\,f(x,y)}{\mathsf{d}y}\,(p)\cdot v = \lambda x.\frac{\mathsf{d}f(x,y)}{\mathsf{d}y}\,(p)\cdot v.$$

This can be equivalently expressed by demanding that application maps are linear in their first argument. Here is the argument using the term logic.

Lemma 4.10. *A Cartesian differential category which is a Cartesian-closed category satisfies differential currying if and only if all applications are linear in their first argument.*

Proof. First assume $\frac{\mathsf{d}\lambda x.f(x,y)}{\mathsf{d}y}\,(p)\cdot v = \lambda x.\frac{\mathsf{d}f(x,y)}{\mathsf{d}y}\,(p)\cdot v$ then setting $f(x,y) = xy$ (the application), we have

$$\begin{aligned}\frac{\mathsf{d}xy}{\mathsf{d}x}\,(p)\cdot w &= (\lambda y.\frac{\mathsf{d}xy}{\mathsf{d}x}\,(p)\cdot w)y \quad \text{(by } \beta\text{-reduction)} \\ &= \left(\frac{\mathsf{d}\lambda y.xy}{\mathsf{d}x}\,(p)\cdot w\right)y \quad \text{(by assumption)} \\ &= \left(\frac{\mathsf{d}x}{\mathsf{d}x}\,(p)\cdot w\right)y \quad \text{(by extensionality)} \\ &= wy \quad \text{(by linearity of projection)}\end{aligned}$$

which shows that the application is linear in the first argument. Conversely, assume that application is linear in its first argument so that

$$\frac{\mathsf{d}xy}{\mathsf{d}x}\,(p)\cdot w = wy$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

then

$$\begin{aligned}
\lambda y.\frac{\mathsf{d}f(x,y)}{\mathsf{d}x}\,(p)\cdot w &= \lambda y\,.\,\frac{\mathsf{d}(\lambda y.f(x,y))y}{\mathsf{d}x}\,(p)\cdot w \quad \text{(by } \beta\text{-reduction)}\\
&= \lambda y.\frac{\mathsf{d}zy}{\mathsf{d}z}\,(f(p,y))\cdot\frac{\mathsf{d}\lambda y\,.f(x,y)}{\mathsf{d}x}\,(p)\cdot w \quad \text{(chain rule)}\\
&= \lambda y.\left(\frac{\mathsf{d}\lambda y\,.f(x,y)}{\mathsf{d}x}\,(p)\cdot w\right)y \quad \text{(linearity of application)}\\
&= \frac{\mathsf{d}\lambda y\,.f(x,y)}{\mathsf{d}x}\,(p)\cdot w \quad \text{(by extensionality)}
\end{aligned}$$

□

Notice that extensionality (or η-expansion) is used in both directions of this proof so that this proof cannot be adapted to a setting in which just β-reduction holds (i.e. which has canonical codes).

Next, we show that if $\mathcal{E}$ is the collection of retractively linear idempotents in a differential Turing category with canonical codes, then $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$ is a Cartesian-closed differential Turing category. The difficulty in the proof is to show $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$ is closed to forming internal homs. Once we have established this, we will have that $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$ is a Cartesian-closed category that is also a differential Turing category. Moreover, it is then immediate that the coherence for Cartesian-closed differential categories

$$D[\lambda(f)] = \lambda(\,\langle \pi_0 \times 0, \pi_1 \times 1\rangle\, D[f])$$

holds because $\lambda(f)$ in $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$ is the canonical code $\lambda(f)$ from $\mathbb{X}$. Thus, the proof of the following theorem requires only that we show retractively linear idempotents are closed to internal homs.

We make use of Lemma 4.2. Also, recall that if h is linear in its first argument, then

$$\langle \pi_0 \times 0, \pi_1 \times 1\rangle\, D[h] = a_\times(\,\langle 1, 0\rangle \times 1)D[h] = a_\times(1 \times \pi_1)h = (\pi_0 \times 1)h.$$

Theorem 4.2. *When $\mathcal{E}$ is the collection of retractively linear idempotents in a differential Turing category with canonical codes, then $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$ is a Cartesian-closed differential Turing category.*

Proof. Let $u, v \in \mathcal{E}$. It suffices to show that $[u, v] \in \mathcal{E}$; i.e. we must show that $D[[u, v]][u, v] = \pi_0[u, v]$ and that $([u, v] \times [u, v])D[[u, v]] = D[[u, v]]$.

For $D[[u, v]][u, v] = \pi_0[u, v]$,

$$\begin{aligned}
D[[u,v]][u,v] &= D[\lambda((1\times u)\bullet v)]\lambda((1\times u)\bullet v)\\
&= \lambda((D[\lambda((1\times u)\bullet v)]\times 1)(1\times u)\bullet v)\\
&= \lambda((1\times u)(D[\lambda((1\times u)\bullet v)]\times 1)\bullet v)\\
&= \lambda((1\times u)(\lambda(\,\langle\pi_0\times 0,\pi_1\times 1\rangle\, D[(1\times u)\bullet v])\times 1)\bullet v)\\
&= \lambda((1\times u)\,\langle\pi_0\times 0,\pi_1\times 1\rangle\, D[(1\times u)\bullet v]v)\\
&= \lambda((1\times u)\,\langle\pi_0\times 0,\pi_1\times 1\rangle\, D[(1\times u)\bullet]v)\\
&= \lambda((1\times u)(\pi_0\times 1)(1\times u)\bullet v)\\
&= \lambda((\pi_0\times 1)(1\times u)\bullet v)\\
&= \pi_0\lambda((1\times u)\bullet v) = \pi_0[u,v].
\end{aligned}$$

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

For $([u,v]\times[u,v])D[[u,v]] = D[[u,v]]$, first consider,

$$\begin{aligned}
&\langle \pi_0\times 0, \pi_1\times 1\rangle\, D[(1\times u)\bullet v]\\
&\quad = \langle \pi_0\times 0, \pi_1\times 1\rangle\, \langle D[(1\times u)\bullet, \pi_1(1\times u)\bullet\rangle\, D[v]\\
&\quad = \langle \langle \pi_0\times 0, \pi_1\times 1\rangle\, D[(1\times u)\bullet], (\pi_1\times u)\bullet\rangle\, D[v]\\
&\quad = \langle (\pi_0\times u)\bullet, (\pi_1\times u)\bullet\rangle\, D[v].
\end{aligned}$$

So that we may simplify $D[[u,v]]$ as

$$\begin{aligned}
D[[u,v]] &= D[\lambda((1\times u)\bullet v)]\\
&= \lambda(\langle \pi_0\times 0, \pi_1\times\rangle\, D[(1\times u)\bullet v])\\
&= \lambda((1\times u)\,\langle (\pi_0\times 1)\bullet, (\pi_1\times 1)\bullet\rangle\, D[v]).
\end{aligned}$$

Finally we have the calculation:

$$\begin{aligned}
&([u,v]\times[u,v])D[[u,v]]\\
&\quad = ([u,v]\times[u.v])\lambda((1\times u)\,\langle(\pi_0\times 1)\bullet, (\pi_1\times 1)\bullet\rangle\, D[v])\\
&\quad = \lambda((([u,v]\times[u,v])\times 1)(1\times u)\,\langle(\pi_0\times 1)\bullet, (\pi_1\times 1)\bullet\rangle\, D[v])\\
&\quad = \lambda((1\times u)(([u,v]\times[u,v])\times 1)\,\langle(\pi_0\times 1)\bullet, (\pi_1\times 1)\bullet\rangle\, D[v])\\
&\quad = \lambda((1\times u)\,\langle(\pi_0\times 1)([u,v]\times 1)\bullet, (\pi_1\times 1)([u,v]\times 1)\bullet\rangle\, D[v])\\
&\quad = \lambda((1\times u)\,\langle(\pi_0\times 1)(1\times u)\bullet v, (\pi_1\times 1)(1\times u)\bullet v\rangle\, D[v])\\
&\quad = \lambda((1\times u)\,\langle(\pi_0\times 1)\bullet v, (\pi_1\times 1)\bullet v\rangle\, D[v])\\
&\quad = \lambda((1\times u)\,\langle(\pi_0\times 1)\bullet, (\pi_1\times 1)\bullet\rangle\, (v\times v)D[v])\\
&\quad = \lambda((1\times u)\,\langle(\pi_0\times 1)\bullet, (\pi_1\times 1)\bullet\rangle\, D[v]) \qquad (v \text{ is retractively linear})\\
&\quad = D[[u,v]] \qquad (\text{from above})
\end{aligned}$$

Thus $[u,v]$ is retractively linear when u and v are. Therefore, $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$ is a Cartesian-closed differential Turing category. □

Finally, we obtain the Scott–Koymans theorem for the differential λ-calculus.

Theorem 4.3. *Every model of the differential λ-calculus arises as the model of a reflexive object in a Cartesian-closed differential category.*

Proof. A model of the differential λ-calculus gives is a differential Turing category with canonical codes whose objects are Cartesian product words in the Turing object. As before, these all arise as classifying categories of differential λ-theories. Such a Turing category embeds into its idempotent completion with respect to retractively linear idempotents, $\mathcal{I}:\mathbb{X}\to\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$. $\mathsf{Split}_{\mathcal{E}}(\mathbb{X})$ is a Cartesian-closed Turing category in which $\mathcal{I}(T) = 1_T$ is a Turing object. This means as the Turing object is differentially universal that there is a linear retraction $1_T \to [1_T, 1_T]$ and this exhibits $\mathcal{I}(T)$ as a reflexive object as required. □

5. Conclusion

Manzonetto had in mind the relational models, in which addition is idempotent, arising from multisets Manzonetto (2012). His models are, of course, models of our more general theory. However, this development allows for the possibilities of differential models of the λ-calculus with a non-idempotent addition. Such a model, of course, is already implicit in Ehrhard and Regnier (2003);

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

however, this is a "syntactic" model. Establishing sources of semantic models remains a largely unexplored. This, in particular, requires a ready source of models of the typed differential λ-calculus. An unexpected source of such models, described in Gallagher (2018), arise from abstract differential geometry (DG) through coherently closed tangent categories. Abstract DG includes classical DG, synthetic DG, convenient manifolds, etc. This, in particular, suggests that there are semantic (and possibly syntactic) models of computation with a DG flavor.

Notes

1 At the time of the development of the Scott–Koymans theorem, not only was Dana Scott publicizing his results but also Jim Lambek and Phil Scott were in the process of writing their book (Lambek and Scott 1986), and, indeed, were circulating chapters for comment. Their book, in particular, included a section on C-monoids which provide the semantics for the untyped λ-calculus with both η- and β-equality. In the exercises of book, they also described the notion of a "weak" C-monoid and noted that these provide the semantics of the λ-calculus with just β-equality. Their notion of "weak" is exactly our notion of "having canonical codes." Note, however, that our usage here of "weak" for codes *does not admit substitution* and follows the (now) widely accepted categorical usage.

2 Here $[U, U]$ is the internal hom sometimes written U^U or $U \Rightarrow U$. The notation $[U, U] \lhd^s_r U$ means $[U, U]$ is a retract of U with $s : [U, U] \to U$ the section and $r : U \to [U, U]$ the retraction so $sr = 1_{[U,U]}$.

3 It is normal to set up the λ-calculus with just the β-equality and the ξ-rule. Somewhat more unusual is to demand what we have called here the λ-substitution rule. However, the omission of latter rule caused quite a problem in the foundations of the λ-calculus and its models. In fact, two different ways of defining models of the λ-calculus evolved – λ-algebras (promoted by Henk Barendregt) and λ-models (promoted by Dana Scott). The difference between the two is quite subtle and involves the interpretation of λ-abstraction (see Hindley and Longo 1980). The λ-substitution rule, however, makes the issue rather clear: λ-algebras do not admit λ-substitution while λ-models do.

References

Birkedal, L. (1999). *Developing Theories of Types via Computability and Realizability*. Phd. thesis, Carnegie Mellon University.

Blute, R., Cockett, J. and Seely, R. (2006). Differential categories. *Mathematical Structures in Computer Science* **16** 1049–1083.

Blute, R., Cockett, J. and Seely, R. (2009). Cartesian differential categories. *Theory and Application of Categories* **22** 622–672.

Blute, R., Cockett, J. and Seely, R. (2015). Cartesian differential storage categories. *Theory and Application of Categories* **30** 620–686.

Boudol, G. (1993). The λ-calculus with multiplicities. In: *Proceedings of the 4th International Conference on Concurrency Theory*, Lecture Notes in Computer Science, vol 715. Springer, Berlin, Heidelberg. 1–6.

Boudol, G., Curien, P. and Lavatelli, C. (1999). A semantics for λ-calculi with resource. *Mathematical Structures in Computer Science* **9** 437–482.

Boudol, G. and Laneve, C. (1995). λ-*Calculus, Multiplicities, and the* π-*Calculus*. Technical report RR-2581, Institut National de Recherche en Informatique et en Automatique.

Bucciarelli, A., Ehrhard, T. and Manzonetto, G. (2010) Categorical models for simply typed resource calculi. *Electronic Notes in Theoretical Computer Science* **265** 213–230.

Cockett, J. and Gallagher, J. (2016). Categorical models of the λ-calculus revisted. *Electronic Notes in Theoretical Computer Science* **325** 63–83.

Cockett, J. and Hofstra, P. (2008). Introduction to Turing categories. *Annals of Pure and Applied Logic* **156** 183–209.

Dezani-Ciancaglini, M. (1996). *Logical Semantics for Concurrent* λ-*Calculus*. Phd thesis, Katholieke Universiteit Nijmegen.

Ehrhard, T. (2002). On Köthe sequence spaces and linear logic. *Mathematical Structures in Computer Science* **12** 579–623.

Ehrhard, T. (2005). Finiteness spaces. *Mathematical Structures in Computer Science* **15** 615–646.

Ehrhard, T. and Regnier, L. (2003). The differential λ-calculus. *Theoretical Computer Science* **309** 1–41.

Faure, G. and Miguel, A. (2009). *A Categorical Semantics for the Parallel* λ-*Calculus*. Technical report RR-7063, Institut National de Recherche en Informatique et en Automatique.

Fiore, M. (2007). Differential structure in models of multiplicative biadditive intuitionistic linear logic. In: *Typed Lambda Calculi and Applications*, Lecture Notes in Computer Science, vol 4583. Springer, Berlin, Heidelberg. 163–177.

Gallagher, J. (2018). *The Differential* λ-*Calculus: Syntax and Semantics for Differential Geometry*. Phd thesis, University of Calgary.

Hindley, R. and Longo, G. (1980). Lambda-calculus models and extensionality. *Zeitschrift fur Mathematische Legik* **26** 289–310.

Koymans, C. (1982). Models of the λ-calculus. *Information and Control* **52** 306–332.

Lambek, J. and Scott, P. (1986). *An Introduction to Higher Order Categorical Logic*, Cambridge University Press.

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press

Longo, G. and Moggi, E. (1990). A category theoretic characterization of functional completeness. *Theoretical Computer Science* **70** 193–211.
Manzonetto, G. (2012). What is a categorical model of the differential and the resource λ calculi? *Mathematical Structures in Computer Science* **22** 451–520.
Scott, D. (1980). Relating theories of the λ-calculus. In: Curry, H.B. (ed.) *Essays on Combinatory Logic, Lambda Calculus, and Formalism*, Academic Press. 403–450.
Vaux, L. (2009). The algebraic λ-calculus. *Mathematical Structures in Computer Science* **19** 1029–1059.

Cite this article: Cockett R and Gallagher J (2019). Categorical models of the differential λ-calculus. *Mathematical Structures in Computer Science* **29**, 1513–1555. https://doi.org10.1017/S0960129519000070

https://doi.org/10.1017/S0960129519000070 Published online by Cambridge University Press