Models of linear dependent type theory

Martin Lundfall

August 29, 2017

Abstract

The role that types play in type theory can be seen as a simoultaneous generalisation of the concept of a set, proposition and data type. In linear type theory, this view can be extended by interpreting linear types as a particular kind of resource that is *consumed* in the derivation. In this paper, we construct a type theory in which we distinguish between non-linear, dependent types (what we will call *cartesian*), and linear types, where both constructs are allowed to depend on terms of cartesian types. In the interplay between the cartesian and linear types we develop two new type formers, $\sqcap_{x:A} B$ and $\sqsubset_{x:A} B$, with rules similar to Π and Σ, but where the dependent type B (and therefore the resulting construct) is linear. The !-exponential from linear logic is deconstructed into two operators, M and L, allowing a transformation of linear types into cartesian and vice versa. We construct a semantic framework where the type theory is interpreted by constructing a base category of cartesian contexts equipped with two fibrations, whose fibers over a context Γ give rise to the categories of linear and cartesian types in Γ. The operators M and L give rise to a fiberwise adjunction between these.

Contents

1 TODO

Understand universes semantically
Let $\mathcal{L}$ be a (monoidal?) (2,1)-category. For $A, B \in \mathcal{L}_\Gamma$, can one define the type $f =_{[A,B]} g$ to consist of the 2-cells between f and g?
Examples of such categories:
Cat (with natural iso)
Grp
SMC
Kelly's enriched categories for more on enriched cats

2 Introduction

3 Preliminaries

The paper is largely split into two sections; syntax and semantics. Familiarity with basic category theoretic concepts like limits, adjoints, monoidal categories is assumed.

3.1 Dependent type theory

A *type* is a syntactic construction whose meaning and function stems from the ways in which it interacts with the derivation rules of a *type theory*. Approaching type theory from the perspective of logic, we can form a first approximation of the definition of a type as a generalization of a proposition and a set.

3.2 Linear type theory

Linear logic is a substructural logic in which the rules of weakening and contraction:

$$\frac{\Gamma, \Delta \vdash B}{\Gamma, A, \Delta \vdash B}\ \text{Weak} \qquad \frac{\Gamma, A, A, \Delta \vdash B}{\Gamma, A, \Delta \vdash B}\ \text{Contr}$$

are not admissable.
In other words, assumptions cannot be freely assumed or dismissed; they must be used exactly once in the conclusion. In linear logic, we are inclined to think of a sequent $A_1, A_2, \ldots, A_n \vdash B$ as modelling a function or process in which the assumptions $A_1, A_2, \ldots A_n$ are *resources used* to yield B. As a first, toy example, we consider the chemical process of burning hydrogen:

Example 4. Consider the following primitive derivation rule:

$$O_2 \otimes H_2 \otimes H_2 \vdash H_2O \otimes H_2O$$

stating that given an oxygen molecule and two hydrogen molecules, burning yields two water molecules. If weakening was admissable, we would be able to assume an additional hydrogen molecule without changes the antecedent, which does not make sense under the resource interpretation of linear logic.
Just as inuitionistic logic naturally extends to (dependent) type theory under the slogan of propositions as types, linear logic can be extended to a linear type theory, where sequents are decorated with proof terms:

$$x_1 : A_1, x_2 : A_2, \ldots x_n : A_n \vdash t : B$$

and here linearity implies that the free variables of t are $x_1, x_2, \ldots x_n$, each appearing exactly once.
Interest in linear type theory stems from disparate sources. From the perspective of (classical) computer science, it can be used for modelling state and storage, with linear variables denoting resources like pointers or files [9], or even as a theoretical description of the resource handling of a blockchain [11]. In quantum physics, linear types respect the no cloning theorem of quantum states, and are also proposed as a syntax for quantum computation [4]. From a semantic perspective, linear type theory can be interpreted in a symmetric monoidal closed categories, generalising the structure of a cartesian closed categories in which (non-dependent) type theories are interepreted.

4.1 Fibrations

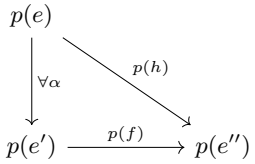
A general heuristic for modelling a mathematical object E dependent over another object B is to specify a *projection* morphism $p : E \to B$ subject to certain constraints. For example, a family of sets A_i indexed by the set I might equally well be understood as the set $E = \bigsqcup_{i \in I} A_i$ together with a morphism $p : E \to I$ such that $p^{-1}(i) = A_i$. This is the idea guiding the concept of a fibration. Dependeding on which kind of mathematical object we are dealing with, we impose various conditions on the projection $p : E \to B$ to be able to *lift* certain structure of the base B into *fibers* $E_b = p^{-1}(b)$ for $b \in B$.

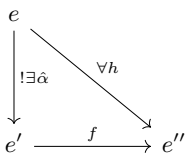
4.1.1 Grothendieck fibrations

In the context of categories, the appropriate notion is that of a **Grothendieck fibration**, often just called **fibration**. In order for a functor $p : E \to B$ to be a fibration, one needs to be able to lift arrows in the base category B to arrows in E. We do this by asking for the existence of certain *cartesian arrows* in E.

Definition 4.1 (Cartesian arrow)**.** Given a functor $p : E \to B$, an arrow $f : e' \to e''$ of E is said to be **cartesian** with respect to p if for every $h : e \to e''$ and $\alpha : p(e) \to p(e')$ such that $p(f)\alpha = p(h)$, there is a unique arrow $\hat{\alpha} : e \to e'$ such that $p(\hat{\alpha}) = \alpha$ and $f\hat{\alpha} = h$.

The situation is illustrated in the following diagrams:





Definition 4.2 (Grothendieck fibration)**.** A functor $p : E \to B$ is a **Grothendieck fibration** if, for every $e \in E$, $b \in B$ and arrow $f : b \to p(e)$ in B, there exists a cartesian arrow f^* in E such that $p(f^*) = f$.

We will refer to a cartesian arrow f^* for which $p(f^*) = f$ as a **lift** of f. Although lifts are not uniquely determined, from the condition of cartesianness their domain will determined up to unique isomorphism. A basic review of Grothendieck fibrations particularly relevant to the semantics of dependent type theory can be found in [7]. We will simply repeat the basic notions that will used in our investigations.

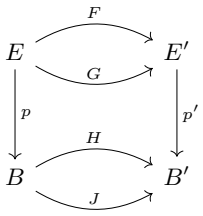
For any object $b \in B$ in the base, we call the subcategory of E which is mapped to b and its identity morphism the **fiber** over b. This will be denoted E_b. A fibration $p : E \to B$ induces for every $u : b \to b'$ in B a functor $u^* : E_{b'} \to E_b$ by sending every object to the domain of the cartesian arrow associated to u^*. Such functors will be unique up to unique natural isomorphism. In general, for two compatible morphisms u and v in the base, a lift of a composition is not identitcal to the composition of a lift, only naturally isomorphic. In other words, we have $u^* \circ v^* \cong (u \circ v)^*$, but not functoriality on the nose. Those fibration for which there are lifts of morphisms such that the equalities hold on the nose are called **Split**.

Definition 4.3 (Cartesian functor)**.** Let $p : E \to B$ and $q : E' \to B$ be fibrations over the same base. A functor $F : E \to E'$ is **cartesian** if $qF = p$ and cartesian morphisms in E with respect to p are mapped to cartesian morphisms in E' with respect to q.

This determines a category $\mathbf{Fib}(B)$, consisting of fibrations over B and cartesian functors between them. More generally, one can construct a category **Fib** whose morphisms from fibrations $p : E \to B$ and $q : E' \to B'$ are pairs of functors (F, G) where $F : E \to E'$ and $G : B \to B'$ such that $G \circ p = q \circ F$ and F preserves cartesian morphisms. In fact, the functor $\mathbf{Fib} \to \mathbf{Cat}$ sending a fibration to its base is itself a fibration whose fibers are $\mathbf{Fib}(B)$ for any small category B.

With the notion of a fibered natural transformation, **Fib** forms a 2-category.

Definition 4.4 (Fibered natural transformation)**.** For two pairs of parallel functors F, H and G, J between fibrations (E, B) and (E', B'), as illustrated in the commutative square:



a **Fibered natural transformation** (λ, λ') between (F, H) and (G, J) is a pair of natural transformations $\lambda : G \to F$ and $\lambda' : A \to B$ such that $p'(\lambda) = p_{\lambda'}$.

Notice that this definition does not ask for F and G to be cartesian functors. But when that is the case, fibered natural transformations between the parallel morphisms (F, A) and (G, A) in **Fib** are the 2-morphisms of this 2-category.

An important special case of this is when $B = B'$ and $H = J = 1_B$. Then a fibered natural transformation $\lambda : F \to G$ is simply a natural transformation such that all of its components are sent to identities via p'. Such a natural transformation is sometimes called *vertical*

Definition 4.5. Let $p : E \to B$ and $q : D \to B$ be fibrations over the same base and $F : E \to D$ and $G : D \to E$ cartesian functors with respect to these. F is called a **fibered left adjoint** of G if $F \dashv G$ in the usual way and the unit η of the adjunction is vertical. Such an adjunction will be called **fiberwise adjunction**.

4.1.2 Other types of fibrations

In exploring models of linear and dependent types, fibrations of other structures will arise. Two important examples will be fibrations of groupoids and fibrations of monoidal categories.

Definition 4.6. A map $p : G \to H$ in **Grpd** is a **fibration of groupoids** if for every $g \in G$ and $f : p(g) \to h$ in H, there exists an object g' and map $\hat{f} : g \to g'$ in G such that $p(g') = h$ and $p(\hat{f}) = f : p(g) \to p(g')$.

When considering fibrations of monoidal categories, we distinguish between the case where both the fibration and the base are monoidal categories, and when we simply want each fiber to be a monoidal category and the induced functors between these to be monoidal functors. The former notion is that of a monoidal fibration:

Definition 4.7. A **monoidal fibration** is a functor $\Phi\colon E \to B$ such that

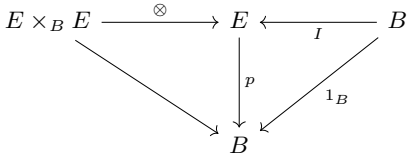
- Φ is a Grothendieck fibration
- E and B are monoidal categories and Φ is a strict monoidal functor, and
- the tensor product of E preserves cartesian arrows.

From this it can be shown that maps in the base induce monoidal functors in between the fibers. See [12] for details.

A weaker notion that does not require the base category to be monoidal is that of a lax monoidal fibration [14]. We simply want each fiber to be a monoidal category and that funcotrs between fibers are monoidal.

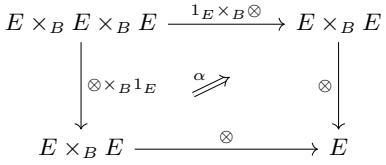
Definition 4.8. A **lax monoidal fibration** is a fibration $p : E \to B$ along with

1. Two functors $\otimes : E \times_B E \to E$ and $I : B \to E$ fitting into the following diagram:

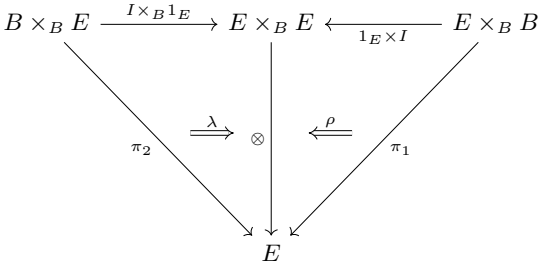


.

2. Three fibered natural isomorphisms α, λ and ρ associated with the diagrams:



and



3. such that α, λ and ρ satisfies the pentagon and triangle identities in each fiber.
4. for every $b \in B$, $\rho_{I_b} = \lambda_{I_b}^{-1} : I_b \otimes I_b \to I_b$

These conditions are sufficient for each fiber to be a monoidal category and for the induced functors between fibers to be monoidal. More detais are given in [14].

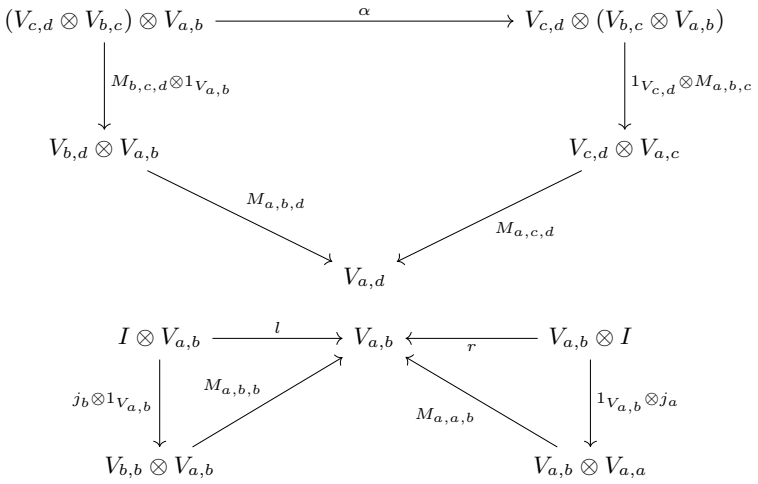
4.2 Enriched and higher categories

The concept of a category can be generalized in a few different directions. In exploring semantics of linear type theory, our first encounter with such structures will be enriched categories, where hom-sets are generalized to "hom-objects" of a monoidal category $\mathcal{V}$.

Definition 4.9. For a monoidal category $\mathcal{V}$, a **$\mathcal{V}$-enriched category** $\mathcal{C}$, consists of the following:

1. A set $\mathcal{C}_0$ of objects,
2. for each pair $a, b \in \mathcal{C}_0$, an object $V_{a,b} \in \mathcal{V}$,
3. for every $a, b, c \in \mathcal{C}_0$, a *composition law*, i.e. a functor $M_{a,b,c} : V_{b,c} \otimes V_{a,b} \to V_{a,c}$ and
4. an identity element $j_a : I \to V_{a,a}$

such that for all a, b, c, d, the following associativity and identity diagrams commute:



where a, l and r are the associator, and left and right unitor ismorphisms associated with the monoidal structure of $\mathcal{V}$.

Note that for the special case of $\mathcal{V} = Set$ we get back the definition of a category. Plenty of examples of constructions in enriched categories can be found in the comprehensive introduction *Basic concepts of enriched category theory* by Max Kelly, [8].

Another generalization of a category is suggested by the direction in which categories generalize sets/classes by allowing arrows between objects. Thinking of arrows as one-dimensional objects between zero-dimensional objects, one can imagine the existence of 2-dimensional arrows between 1-dimensional ones. This leads to the notion of a 2-category. These come in two different forms, strict 2-categories (or just 2-categories) or weak 2-categories (also known as bicategories). Equipped with the notion of an enriched category, strict 2-categories can be defined concisely as:

Definition 4.10. A **(strict) 2-category** is a Cat-enriched category.

Breaking down this definition provides a more illuminating view. A strict 2-category, $\mathcal{C}$, consists of

- a collection of objects $\mathcal{C}_0$,
- for any pair of objects $a, b \in \mathcal{C}_0$, a category $C_{a,b}$, whose objects will be called "1-morphisms" and whose morphisms are renamed "2-morphisms".
- for any object $a \in \mathcal{C}_0$, an "identity" functor $1_a : \mathbf{1} \to \mathcal{C}_{a,a}$,
- for any triple of objects $a, b, c \in \mathcal{C}_0$, a functor $Comp : C_{b,c} \times C_{a,b} \to C_{a,c}$ satisfying the associativity and identity diagrams from the definition of enriched categories.

The action of $Comp$ on 1-morphisms is called "horizontal composition" and is written $gf := Comp(g, f)$ for $f \in \mathcal{C}_{a,b}$ and $g \in \mathcal{C}_{b,c}$, whereas the composition between 2-morphisms $\alpha : f \implies g$, $\alpha' : g \implies h$ for $f, g, h \in \mathcal{C}_{a,b}$ is written $\alpha' \circ \alpha$. Thanks to the functoriality of $Comp$, these satisfy the following *interchange law*:

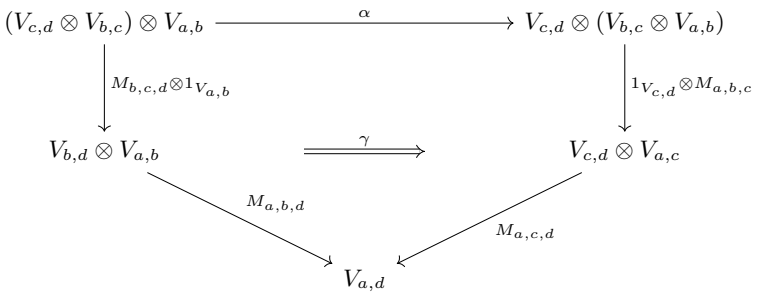
$$(\beta' \circ \beta)(\alpha' \circ \alpha) = (\beta'\alpha') \circ (\beta\alpha)$$

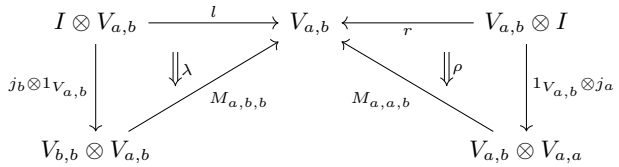
The path from strict 2-categories to weak 2-categories is marched to the category theoretic slogan that it is undesirable to speak of equality between objects in a category. 2-morphisms give us a way of relating 1-morphisms up to isomorphism instead of on-the-nose equality. Thus we are inclined to loosen the restriction of the associativity and unital diagrams in the definition of a 2-category to only commute up to coherent natural isomorphism:

Definition 4.11. A **weak 2-category** $\mathcal{C}$ is a collection of objects, 1-morphisms and 2-morphisms with composition and identity functors as before, such that for all $a, b, c, d \in \mathcal{C}_0$, there exists natural isomorphisms

$$\gamma : M_{a,b,d} \circ (M_{b,c,d} \otimes 1_{V_{a,b}}) \implies M_{a,c,d} \circ (1_{V_{c,d}} \otimes M_{a,b,c}) \circ a$$
$$\lambda : l \implies M_{a,b,b} \circ (j_b \circ 1_{V_{a,b}})$$
$$\rho : r \implies M_{a,a,b} \circ (1_{V_{a,b}} \circ j_a)$$

in other words, associated with the diagrams:





subject to the following coherence conditions:

5 Syntax

The particular linear dependent type theory under consideration is a inspired by the work of Krishnaswami in [9] and Vákár in [13]. Types are either *cartesian*, in which case we simply write $\Gamma \vdash A$ type, or *linear*, written $\Gamma \vdash A$ linear. When making typing judgements of linear terms, contexts will be split into two parts, separated by a semicolon. The first part contains *cartesian* assumptions, for which weakening and contraction is allowed, while the second part is the *linear* part, containing ephemeral assumptions that we are also inclined to think of as resources. The derivation rules for linear types will force the linear variables to occur exactly once in the conclusion. Dependent types are restricted to only depend on terms of cartesian types.

5.1 Structural rules

We will be dealing with the following judgements:

Judgement:	**Meaning:**
$\vdash \Gamma$ ctxt	Γ is a well-formed cartesian context.
$\vdash \Gamma; \Xi$ ctxt	$\Gamma; \Xi$ is a well-formed mixed context
$\Gamma \vdash A$ type	A is a cartesian type in Γ
$\Gamma \vdash A$ linear	A is a linear type in Γ
$\Gamma \vdash M : A$	M is a term of the cartesian type A in Γ
$\Gamma; \Xi \vdash M : A$	M is a (linear) term of the linear type A in $\Gamma; \Xi$
$\Gamma \vdash A \equiv A'$ type	A and A' are equal cartesian types in Γ
$\Gamma \vdash A \equiv A'$ linear	A and A' are equal linear types in $\Gamma; \Xi$
$\Gamma \vdash M \equiv N : A$	M and N are equal cartesian terms of A in Γ
$\Gamma; \Xi \vdash x \equiv y : A$	x and y are equal linear terms of A in $\Gamma; \Xi$

Figure 1: Judgements of linear dependent type theory

The basic structural rules for the linear dependent type theory are given in 2. Omitted are the rules concerning judgemental equality, which specify that it is an equality relation which is congruent with respect to the other structural rules.

$$\frac{}{\vdash \cdot \text{ ctxt}} \text{ CI-Base} \qquad \frac{}{\vdash \cdot; \cdot \text{ ctxt}} \text{ CM-Base}$$

$$\frac{\Gamma \vdash A \text{ type}}{\vdash \Gamma, x : A \text{ ctxt}} \text{ C-int-ext}$$

$$\frac{\Gamma \vdash A \text{ type} \quad \vdash \Gamma, \Delta \text{ ctxt}}{\vdash \Gamma, x : A, \Delta \text{ ctxt}} \text{ C-weak-1}$$

$$\frac{\Gamma \vdash A \text{ type} \quad \vdash \Gamma, \Delta; \Xi \text{ ctxt}}{\vdash \Gamma, x : A, \Delta; \Xi \text{ ctxt}} \text{ C-weak-2}$$

$$\frac{\vdash \Gamma; \Xi \text{ ctxt} \quad \Gamma \vdash A \text{ linear}}{\vdash \Gamma; \Xi, x : A \text{ ctxt}} \text{ C-lin-ext}$$

$$\frac{\Gamma; \Xi, x : A, y : B, \Xi' \vdash t : A'}{\Gamma; \Xi, y : B, x : A, \Xi' \vdash t : A'} \text{ Lin-exch}$$

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma, \Gamma'; \Xi \vdash t : A'}{\Gamma, x : A, \Gamma'; \Xi \vdash t : A'} \text{ Weak-L}$$

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma, \Gamma' \vdash \mathcal{J}}{\Gamma, x : A, \Gamma' \vdash \mathcal{J}} \text{ Weak-I}$$

$$\frac{\Gamma \vdash M : A \quad \Gamma, x : A, \Gamma' \vdash \mathcal{J}}{\Gamma, \Gamma'[M/x] \vdash \mathcal{J}[M/x]} \text{ Int-subst-1}$$

$$\frac{\Gamma \vdash M : A \quad \Gamma, x : A, \Gamma'; \Xi \vdash t : A'}{\Gamma, \Gamma'[M/x]; \Xi[M/x] \vdash t : A'[M/x]} \text{ Int-subst-2}$$

$$\frac{\Gamma; \Xi, x : A \vdash t : B \quad \Gamma; \Xi' \vdash M : A}{\Gamma; \Xi, \Xi' \vdash t[M/x] : B} \text{ Lin-subst}$$

$$\frac{\Gamma, x : A, \Gamma' \text{ ctxt}}{\Gamma, x : A, \Gamma' \vdash x : A} \text{ Int-var}$$

$$\frac{\vdash \Gamma; x : A \text{ ctxt}}{\Gamma; x : A \vdash x : A} \text{ Lin-var}$$

Figure 2: Structural rules

$\mathcal{J}$ denotes a judgement of the form A type, A linear or $M : A$ (for a cartesian type A).

5.2 Cartesian typing rules

The cartesian types that we will use are the standard Π, Σ and Id-types. For Σ and Id we will introduce an extra elimination and computational rule for the case where the type being eliminated into is linear.

$$\frac{\Gamma \vdash A \ \text{type} \quad \Gamma, x : A \vdash B \ \text{type}}{\Gamma \vdash \Pi_{x:A} B \ \text{type}} \ \Pi\text{-F}$$

$$\frac{\Gamma, x : A \vdash b : B}{\Gamma \vdash \lambda x.b : \Pi_{x:A} B} \ \Pi\text{-I}$$

$$\frac{\Gamma \vdash t : \Pi_{x:A} B \ \text{type} \quad \Gamma \vdash M : A}{\Gamma \vdash t(M) : B[M/x]} \ \Pi\text{-E}$$

$$\frac{\Gamma, x : A \vdash b : B \quad \Gamma \vdash M : A}{\Gamma \vdash \lambda x.b(M) \equiv b[M/x] : B[M/x]} \ \Pi\text{-C}$$

$$\frac{\Gamma \vdash A \ \text{type} \quad \Gamma, x : A \vdash B \ \text{type}}{\Gamma \vdash \Sigma_{x:A} B \ \text{type}} \ \Sigma\text{-F}$$

$$\frac{\vdash \Gamma \ \text{ctxt}}{\Gamma \vdash 1 \ \text{type}} \ 1\text{-F}$$

$$\frac{\Gamma \ \text{ctxt}}{\Gamma \vdash \star : 1} \ 1\text{-I}$$

$$\frac{\Gamma, x : 1 \vdash C \ \text{type} \quad \Gamma \vdash c : C[\star/x] \quad \Gamma \vdash M : 1}{\Gamma \vdash \hat{c}[M] : C[M/x]} \ 1\text{-E}$$

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash N : B[M/x]}{\Gamma \vdash (M, N) : \Sigma_{x:A} B} \ \Sigma\text{-I}$$

$$\frac{\begin{array}{c}\Gamma, t : \Sigma_{x:A} B \vdash C \ \text{type} \\ \Gamma, x : A, y : B \vdash c : C[(x, y)/t] \\ \Gamma \vdash s : \Sigma_{x:A} B\end{array}}{\Gamma \vdash \hat{c}[s] : C[s/t]} \ \Sigma\text{-E}_1$$

$$\frac{\begin{array}{c}\Gamma, t : \Sigma_{x:A} B \vdash C \ \text{linear} \\ \Gamma, x : A, y : B; \Xi \vdash c : C[(x, y)/t] \\ \Gamma \vdash s : \Sigma_{x:A} B\end{array}}{\Gamma; \Xi[pr_1(s)/x][pr_2(s)/y] \vdash \hat{c}[s] : C[s/t]} \ \Sigma\text{-E}_2$$

$$\frac{\Gamma \vdash \hat{c}[(a, b)] : C[(a, b)/t]}{\Gamma \vdash \hat{c}[(a, b)] \equiv c[(a, b)/t] : C[(a, b)/t]} \ \Sigma\text{-C}_1$$

$$\frac{\Gamma; \Xi \vdash \hat{c}[(a, b)] : C[(a, b)/t]}{\Gamma; \Xi \vdash \hat{c}[(a, b)] \equiv c[(a, b)/t] : C[(a, b)/t]} \ \Sigma\text{-C}_2$$

$$\frac{\Gamma \vdash \hat{c}[\star] : C[\star/x]}{\Gamma \vdash \hat{c}[\star] \equiv c[\star/x] : C[\star/x]} \ 1\text{-C}$$

Figure 3: (Cartesian) dependent sum and product types

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash N : A}{\Gamma \vdash M =_A N \ \text{type}} \ =\text{-F}$$

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash refl(M) : M =_A M} \ =\text{-I}$$

$$\frac{\begin{array}{c}\Gamma, x, y : A, p : x =_A y \vdash C \ \text{type} \\ \Gamma, z : A \vdash c : C[z/x, z/y, refl(z)/p] \\ \Gamma \vdash M : A \quad \Gamma \vdash N : A \\ \Gamma \vdash P : M =_A N\end{array}}{\Gamma \vdash R^{Id}_{[x,y,p]}(c, M, N, P) : C[M/x, N/y, P/p]} \ =\text{-E}_1$$

$$\frac{\begin{array}{c}\Gamma, x, y : A, p : x =_A y \vdash C \ \text{linear} \\ \Gamma, z : A; \Xi \vdash c : C[z/x, z/y, refl(z)/p] \\ \Gamma \vdash M : A \\ \Gamma \vdash N : A \\ \Gamma \vdash P : M =_A N\end{array}}{\Gamma; \Xi[M/z] \vdash R^{Id}_{[x,y,p]}(c, M, N, P) : C[M/x, N/y, P/p]} \ =\text{-E}_2$$

$$\frac{\Gamma \vdash R^{Id}_{[x,y,p]}(c, M, M, refl(M)) : C[M/x, M/y, refl(M)/p]}{\Gamma \vdash R^{Id}_{[x,y,p]}(c, M, M, refl(M)) \equiv c[M/z] : C[M/x, M/y, refl(M)/p]} \ =\text{-C}_1$$

$$\frac{\Gamma; \Xi \vdash R^{Id}_{[x,y,p]}(c, M, M, refl(M)) : C[M/x, M/y, refl(M)/p]}{\Gamma; \Xi \vdash R^{Id}_{[x,y,p]}(c, M, M, refl(M)) \equiv c[M/z] : C[M/x, M/y, refl(M)/p]} \ =\text{-C}_2$$

Figure 4: (Cartesian) identity type

5.3 Linear typing rules

Perhaps the most important linear types are the $\otimes$- and I-types, as they will provide an interpretation of linear contexts. Semantically, we will not distinguish between the context $\Xi :\equiv x_1 : A_1, x_2 : A_2, \ldots x_n : A_n$ and $x_1 \otimes x_2 \otimes \ldots x_n : A_1 \otimes A_2 \otimes \ldots A_n$.

$$\frac{\Gamma \vdash A \text{ linear} \quad \Gamma \vdash B \text{ linear}}{\Gamma \vdash A \otimes B \text{ linear}} \otimes\text{-F} \qquad \frac{}{\Gamma \vdash I \text{ linear}} \text{I-F}$$

$$\frac{\Gamma; \Xi \vdash a : A \quad \Gamma; \Xi' \vdash b : B}{\Gamma; \Xi, \Xi' \vdash (a, b) : A \otimes B} \otimes\text{-I} \qquad \frac{}{\Gamma; \cdot \vdash * : I} \text{I-I}$$

$$\frac{\Gamma; \Xi' \vdash t : A \otimes B \quad \Gamma; \Xi, x : A, y : B \vdash c : C}{\Gamma; \Xi, \Xi' \vdash \text{let } x, y \text{ be } t \text{ in } c : C} \otimes\text{-E} \qquad \frac{\Gamma; \Xi \vdash c : C \quad \Gamma; \Xi' \vdash t : I}{\Gamma; \Xi, \Xi' \vdash \text{ let } t \text{ be } * \text{ in c : C}} \text{I-E}$$

$$\frac{\Gamma; \Xi \vdash \text{let } x, y \text{ be } (a, b) \text{ in } c : C}{\Gamma; \Xi \vdash \text{let } x, y \text{ be } (a, b) \text{ in } c \equiv c[a/x][b/y] : C} \otimes\text{-C} \qquad \frac{\Gamma; \Xi \vdash \text{ let } * \text{ be } * \text{ in } c : C}{\Gamma; \Xi \vdash \text{ let } * \text{ be } * \text{ in } c \equiv c : C} \text{I-C}$$

Figure 5: Linear $\otimes$ and I type formers

The typing rules for the remaining linear rules are standard.

$$\frac{\Gamma \vdash A \ \text{linear} \quad \Gamma \vdash B \ \text{linear}}{\Gamma \vdash A \multimap B \ \text{linear}} \ \multimap\text{-F}$$

$$\frac{\Gamma;\Xi, x : A \vdash b : B}{\Gamma;\Xi \vdash \lambda x.b : A \multimap B} \ \multimap\text{-I}$$

$$\frac{\Gamma;\Xi \vdash f : A \multimap B \quad \Gamma;\Xi' \vdash a : A}{\Gamma;\Xi,\Xi' \vdash f(a) : B} \ \multimap\text{-E}$$

$$\frac{\Gamma;\Xi \vdash \lambda x.b(a) : B}{\Gamma;\Xi \vdash \lambda x.b(a) \equiv b[a/x] : B} \ \multimap\text{-C}$$

$$\frac{\Gamma \vdash A \ \text{linear} \quad \Gamma \vdash B \ \text{linear}}{\Gamma \vdash A \& B \ \text{linear}} \ \&\text{-F}$$

$$\frac{\Gamma;\Xi \vdash a : A \quad \Gamma;\Xi \vdash b : B}{\Gamma;\Xi \vdash \langle a, b\rangle : A \& B} \ \&\text{-I}$$

$$\frac{\Gamma;\Xi \vdash t : A \& B}{\Gamma;\Xi \vdash \mathrm{fst}(t) : A} \ \&\text{-E}_1$$

$$\frac{\Gamma;\Xi \vdash t : A \& B}{\Gamma;\Xi \vdash \mathrm{snd}(t) : B} \ \&\text{-E}_2$$

$$\frac{\Gamma;\Xi \vdash \mathrm{fst}(\langle a, b\rangle) : A}{\Gamma;\Xi \mathrm{fst}(\langle a, b\rangle) \equiv a : A} \ \&\text{-C}_1$$

$$\frac{\Gamma;\Xi \vdash \mathrm{snd}(\langle a, b\rangle) : B}{\Gamma;\Xi \vdash \mathrm{snd}(\langle a, b\rangle) \equiv b : B} \ \&\text{-C}_2$$

$$\frac{\Gamma \vdash A \ \text{linear} \quad \Gamma \vdash B \ \text{linear}}{\Gamma \vdash A \oplus B \ \text{linear}} \ \oplus\text{-F}$$

$$\frac{\Gamma;\Xi \vdash a : A}{\Gamma\Xi \vdash \mathrm{inl}(a) : A \oplus B} \ \oplus\text{-I}_1$$

$$\frac{\Gamma;\Xi \vdash b : B}{\Gamma\Xi \vdash \mathrm{inr}(b) : A \oplus B} \ \oplus\text{-I}_2$$

$$\frac{\Gamma;\Xi, x : A \vdash c : C \quad \Gamma;\Xi, y : B \vdash d : C; \Gamma;\Xi' \vdash t : A \oplus B}{\Gamma;\Xi,\Xi' \vdash \text{case } t \text{ of } \mathrm{inl}(x) \to c || \mathrm{inr}(y) \to d : C} \ \oplus\text{-E}$$

$$\frac{\Gamma;\Xi \vdash \text{case } \mathrm{inl}(a) \text{ of } \mathrm{inl}(x) \to c || \mathrm{inr}(y) \to d : C}{\Gamma;\Xi \vdash \text{case } \mathrm{inl}(a) \text{ of } \mathrm{inl}(x) \to c || \mathrm{inr}(y) \to d \equiv c[a/x] : C} \ \oplus\text{-C}_1$$

$$\frac{\Gamma;\Xi \vdash \text{case } \mathrm{inr}(b) \text{ of } \mathrm{inl}(x) \to c || \mathrm{inr}(y) \to d : C}{\Gamma;\Xi \vdash \text{case } \mathrm{inr}(b) \text{ of } \mathrm{inl}(x) \to c || \mathrm{inr}(y) \to d \equiv d[b/y] : C} \ \oplus\text{-C}_2$$

$$\frac{}{\Gamma \vdash \top \ \text{linear}} \ \top\text{-F}$$

$$\frac{\vdash \Gamma;\Xi \mathrm{ctxt}}{\Gamma;\Xi \vdash ! : \top} \ \top\text{-I}$$

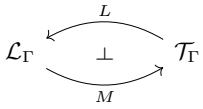
$$\frac{}{\Gamma \vdash 0 \ \text{linear}} \ 0\text{-F}$$

$$\frac{\Gamma;\Xi \vdash t : 0}{\Gamma;\Xi,\Xi' \vdash EFQ(t) : B} \ 0\text{-E}$$

Figure 6: Linear $\multimap$, &, $\oplus$, $\top$ and 0 type formers

5.4 Linear-Cartesian interplay

We introduce two the modal operators M and L, which transfers a linear type/term to its cartesian counterpart and vice versa. Semantically, this will establish a fiberwise monoidal adjunction between the categories of linear and cartesian types:



where the exponential modality from traditional linear logic is understood as the comonad $! = LM$. The decomposition of the exponential into an adjunction goes back to at least [1], and is given an interesting new light in [10], where it is seen as a particular case of a more general procedure of encoding structure in contexts.

Below are the syntactic rules for the operators M and L.

$$\frac{\Gamma \vdash A \text{ type}}{\Gamma \vdash A_L \text{ linear}} \text{ L-F}$$

$$\frac{\Gamma \vdash a : A}{\Gamma; \cdot \vdash a_L : A_L} \text{ L-I}$$

$$\frac{\begin{array}{c}(\Gamma \vdash B \text{ linear}) \\ (\vdash \Gamma; \Xi' \text{ ctxt}) \\ \Gamma; \Xi \vdash y : A_L \quad \Gamma, x : A; \Xi' \vdash t : B\end{array}}{\Gamma; \Xi, \Xi' \vdash \text{let } x \text{ be } y \text{ in } t : B} \text{ L-E}$$

$$\frac{\Gamma; \Xi \vdash \text{let } x \text{ be } s_L \text{ in } t : B}{\Gamma; \Xi \vdash \text{let } x \text{ be } s_L \text{ in } t \equiv t[s/x] : B} \text{ L-C}$$

$$\frac{\begin{array}{c}\Gamma; y : A_L, \Xi \vdash t : B \\ \Gamma; y : A_L, \Xi \vdash t' : B \\ \Gamma, x : A; \Xi \vdash t[x_L/y] \equiv t'[x_L/y] : B \\ \Gamma; \Xi' \vdash a : A_L\end{array}}{\Gamma; \Xi, \Xi' \vdash t[a/y] \equiv t'[a/y]} \text{ L-U}$$

$$\frac{\Gamma \vdash B \text{ linear}}{\Gamma \vdash B_M \text{ type}} \text{ M-F}$$

$$\frac{\Gamma; \cdot \vdash b : B}{\Gamma \vdash b_M : B_M} \text{ M-I}$$

$$\frac{\Gamma \vdash t : B_M}{\Gamma; \cdot \vdash \sigma(t) : B} \text{ M-E}$$

$$\frac{\Gamma \vdash b_M : B_M}{\Gamma; \cdot \vdash \sigma(b_M) \equiv b : B} \text{ M-C}_1$$

$$\frac{\Gamma \vdash \sigma(t)_M : B_M}{\Gamma \vdash \sigma(t)_M \equiv t : B_M} \text{ M-C}_2$$

As a motivation for the semantic interpretation of L and M as adjoint functors given in 7.8, we will demonstrate that they already satisfy the relevant conditions from a syntactic point of view. That is, in the spirit of functional programming, we think of the cartesian and linear types (in a context Γ) of our theory forming the categories $\mathcal{T}$ and $\mathcal{L}$ whose morphisms are terms, and composition is given by substitution. Here, the canonical variable-as-term judgement $\Gamma, x : A \vdash x : A$, or, $\Gamma; y : A_L \vdash y : A_L$, correspond to the identity morphisms.

Theorem 5.1. *M is a (syntactic) functor*

Proof. $\Gamma; x : A \vdash f : B$, the image of f under M is given by:

$$\begin{array}{c}\Gamma; x : A \vdash f : B \\ \Gamma, y : A_M \vdash y : A_M \\ \Gamma, y : A_M; \cdot \vdash \sigma(y) : A \\ \Gamma, y : A_M; \cdot \vdash f[\sigma(y)/x] : B \\ \Gamma, y : A_M \vdash f[\sigma(y)/x]_M : B_M\end{array}$$

Similarly, for $\Gamma, x : A \vdash f : B$ we get $L(f)$ via:

$$\begin{array}{c}\Gamma, x : A \vdash f : B \\ \Gamma, x : A; \cdot \vdash f_L : B_L \\ \Gamma; y : A_L \vdash y : A_L \\ \Gamma; y : A_L \vdash \text{let } x \text{ be } y \text{ in } f_L : B_L\end{array}$$

First, we check that $M(id_A) = id_{MA}$. The image of the identity morphism $\Gamma; x : A \vdash x : A$ is:

$$\Gamma, y : A_M \vdash x[\sigma(y)/x]_M \equiv \sigma(y)_M \equiv y : A_M$$

Then, given morphisms $\Gamma; x : A \vdash f : B$ and $\Gamma; y : B \vdash g : C$, we see that functoriality of M follows directly from functoriality of substitution. Corresponding to $M(g \circ f)$ we have the term:

$$\Gamma, z : A_M \vdash g[f/y][\sigma(z)/x]_M : C_M$$

, while $M(g) \circ M(f)$ is given by:

$$\begin{array}{ll}\Gamma, z : A_M \vdash g[\sigma(f[\sigma(z)/x]_M)/y]_M : C_M & \equiv \\ \Gamma, z : A_M \vdash g[f[\sigma(z)/x]/y]_M : C_M & \equiv \\ \Gamma, z : A_M \vdash g[f/y][\sigma(z)/x]_M : C_M & \end{array}$$

proving functoriality of M.

□

Functoriality of L is not as immediate. For this we really need to utilize the uniqueness rule L-U:

Theorem 5.2. *L is a (syntactic) functor*

Proof. The identity on A, $\Gamma, x : A \vdash x : A$ is mapped to:

$$\Gamma; y : A_L \vdash \text{let } x \text{ be } y \text{ in } x_L : A_L$$

Comparing this with the identity:

$$\Gamma; y : A_L \vdash y : A_L$$

and the equality

$$\Gamma, x : A; \cdot \vdash \text{let } x_L \text{ be } x \text{ in } x_L \equiv x_L : A_L$$

we conclude, using L-U, that let x be y in $x_L = y$.

For types A, B and C, and morphisms $\Gamma, x : A \vdash f : B$, $\Gamma, y : B \vdash g : C$, we want "$L(g \circ f)$":

$$\Gamma; z : A_L \vdash \text{let } x \text{ be } z \text{ in } g[f/y]_L : C_L$$

to equal "$L(g) \circ L(f)$":

$$\Gamma; z : A_L \vdash \text{let } y \text{ be (let } x \text{ be } z \text{ in } f_L) \text{ in } g_L : C_L$$

This is done in the same fashion. From the equality:

$$\Gamma, x : A; \cdot \vdash \text{let } y \text{ be (let } x \text{ be } x_L \text{ in } f_L) \text{ in } g_L \equiv \text{let } y \text{ be } f_L \text{ in } g_L \equiv g_L[f/y] \equiv \text{let } x \text{ be } x_L \text{ in } g[f/y]_L$$

and applying L-U we get the desired equality. □

Furthermore, the categories of cartesian and monoidal types carry a natural monoidal structure, given by $(\times, 1)$ and $(\otimes, I)$, respectively, where $\times$ is simply Σ where the second argument does not depend on the first. M and L are monoidal functors with respect to these.

Definition 5.1. Recall that a functor $F : \mathcal{C} \to \mathcal{D}$ is **monoidal** if there exists a natural transformation $\mu : F \times F \to F$ with components $\mu_{A,B} : \mathcal{F}(A) \otimes_{\mathcal{D}} \mathcal{F}(B) \to \mathcal{F}(A \otimes_{\mathcal{C}} B)$ and a morphism $I_{\mathcal{D}} \to \mathcal{F}(I_{\mathcal{C}})$. Satisfying the following naturality conditions:

1. **(Associativity)** For all objects $x, y, z \in \mathcal{C}$ the following diagram commutes:

$$\begin{array}{ccc}
(F(x) \otimes_{\mathcal{D}} F(y)) \otimes_{\mathcal{D}} F(Z) & \xrightarrow{a^{\mathcal{D}}_{F(x),F(y),F(z)}} & F(x) \otimes_{\mathcal{D}} (F(y) \otimes_{\mathcal{D}} F(z)) \\
\downarrow{\scriptstyle \mu_{x,y} \otimes id} & & \downarrow{\scriptstyle id \otimes \mu_{y,z}} \\
F(x \otimes_{\mathcal{C}} y) \otimes_{\mathcal{D}} F(z) & & F(x) \otimes_{\mathcal{D}} (F(y \otimes_{\mathcal{C}} z)) \\
\downarrow{\scriptstyle \mu_{x \otimes_{\mathcal{C}} y, z}} & & \downarrow{\scriptstyle \mu_{x, y \otimes_{\mathcal{C}} z}} \\
F((x \otimes_{\mathcal{C}} y) \otimes_{\mathcal{C}} z) & \xrightarrow{F(a^{\mathcal{C}}_{x,y,z})} & F(x \otimes_{\mathcal{C}} (y \otimes_{\mathcal{C}} z))
\end{array}$$

   where $a^{\mathcal{C}}$ and $a^{\mathcal{D}}$ denote the associators of the monoidal categories;

2. **(Unitality)** For all $x \in \mathcal{C}$ the following diagrams commutes:

$$\begin{array}{ccc}
1_{\mathcal{D}} \otimes_{\mathcal{D}} F(x) & \xrightarrow{\epsilon \otimes id} & F(1_{\mathcal{C}}) \otimes_{\mathcal{D}} F(x) \\
\downarrow{\scriptstyle \ell^{\mathcal{D}}_{F(x)}} & & \downarrow{\scriptstyle \mu_{1_{\mathcal{C}}, x}} \\
F(x) & \xleftarrow[F(\ell^{\mathcal{C}}_x)]{} & F(1 \otimes_{\mathcal{C}} x)
\end{array}$$

   and

$$\begin{array}{ccc}
F(x) \otimes_{\mathcal{D}} 1_{\mathcal{D}} & \xrightarrow{id \otimes \epsilon} & F(x) \otimes_{\mathcal{D}} F(1_{\mathcal{C}}) \\
\downarrow{\scriptstyle r^{\mathcal{D}}_{F(x)}} & & \downarrow{\scriptstyle \mu_{x, 1_{\mathcal{C}}}} \\
F(x) & \xleftarrow[F(r^{\mathcal{C}}_x)]{} & F(x \otimes_{\mathcal{C}} 1)
\end{array}$$

   where $\ell^{\mathcal{C}}$, $\ell^{\mathcal{D}}$, $r^{\mathcal{C}}$, $r^{\mathcal{D}}$ denote the left and right unitors of the two monoidal categories, respectively.

As our current syntactic endeavors mainly serve as motivation for the semantic interpretation of these operators, given in 7.8, we will not prove the complete statement here. We omit the naturality conditions and only prove the following:

Theorem 5.3. *There are terms t_1, t_2, t_3, t_4, relating the operators M and L so that the judgements:*

$$\Gamma, x : 1 \vdash t_2 : I_M$$
$$\Gamma, x : A_M \times B_M \vdash t_1 : (A \otimes B)_M$$
$$\Gamma; x : I \vdash t_4 : 1_L$$
$$\Gamma; x : A_L \otimes B_L \vdash t_3 : (A \times B)_L$$

hold.

Proof. t_1 is given by the following expression:

$$\Gamma, x : 1 \vdash *_M : I_M$$

and t_2 can easily be derived:

$$\Gamma, x : A_M \times B_M \vdash \sigma(pr_1(x)) \otimes \sigma(pr_2(x)) : (A \otimes B)_M$$

t_3 is given by:

$$\Gamma; x : I \vdash \text{let } x \text{ be } * \text{ in } \star_L : 1_L$$

t_4 is a bit more involved, as it requires two applications of L-E. For clarity, we display the complete proof tree of the term:

$$\text{L-E}\ \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma, x : A, y : B \vdash x : A}{\Gamma, x : A, y : B \vdash y : B}}{\Gamma, x : A, y : B \vdash (x, y) : A \times B}}{\Gamma, x : A, y : B; \cdot \vdash (x, y)_L : (A \times B)_L} \quad \dfrac{}{\Gamma, x : A; y' : B_L \vdash y' : B_L}}{\Gamma, x : A; y' : B_L \vdash \text{let } y \text{ be } y' \text{ in } (x, y)_L : (A \times B)_L} \quad \dfrac{}{\Gamma; x' : A_L \vdash x' : A_L}}{\otimes\text{-E}\ \dfrac{\Gamma; x' : A_L, y' : B_L \vdash \text{let } x \text{ be } x' \text{ in } (\text{let } y \text{ be } y' \text{ in } (x, y)_L) : (A \times B)_L}{\Gamma; z : A_L \otimes B_L \vdash \text{let } (x', y') \text{ be } z \text{ in } (\text{let } x \text{ be } x' \text{ in } (\text{let } y \text{ be } y' \text{ in } (x, y)_L)) : (A \times B)_L}} \quad \dfrac{}{\Gamma; z : A_L \otimes B_L \vdash z : A_L \otimes B_L}$$

□

Finally, we show that M and L can be thought of as adjoint functors, in any context. In other words, we will show that there exists a "natural transformation" $\epsilon : LM \to 1$ satisying the following universal property:
For any $f : L(A) \to B$, there is a unique morphism $g : A \to B_M$, such that $\epsilon_B \circ L(g) = f$.
Translated into the syntax of our type theory, the statement becomes the following:

Theorem 5.4 ($L \dashv M$)**.** *There is a term $\Gamma; \beta_1 : B_{LM} \vdash \epsilon_B : B$ with the following property:*
For any term: $\Gamma; y : A_L \vdash f : B$, there is a unique term $\Gamma, x : A \vdash g : B_M$ such that $\Gamma; y : A_L \vdash \epsilon_B[\text{let } x \text{ be } y \text{ in } g_L/\beta_1] \equiv f : B$.

Proof. The counit $\epsilon : LM \to 1_{\mathcal{L}_\Gamma}$ is at any component B given by:

$$\Gamma, \beta_1 : B_M \vdash \beta_1 : B_M$$
$$\Gamma, \beta_2 : B_M \vdash \beta_2 : B_M$$
$$\Gamma, \beta_2 : B_M; \cdot \vdash \sigma(\beta_2) : B$$
$$\Gamma; \beta_1 : B_{LM} \vdash \text{let } \beta_2 \text{ be } \beta_1 \text{ in } \sigma(\beta_2) : B$$

where the last line is given by applying L-E to the first and third line.
For any $\Gamma; x : A_L \vdash f : B$, we get the corresponding g through:

$$\Gamma, x : A \vdash x : A$$
$$\Gamma, x : A; \cdot \vdash x_L : A$$
$$\Gamma, x : A; \cdot \vdash f[x_L/y] : B$$
$$\Gamma, x : A \vdash f[x_L/y]_M : B_M$$

making Lg the term:

$$\Gamma; y : A_L \vdash \text{let } x \text{ be } y \text{ in } f[x_L/y]_{LM} : B_{LM}$$

our composite $\epsilon_B \circ Lg$ is given by substituting the above for β_1 in the term correpsonding to ϵ_B, yielding:

$$\Gamma; y : A_L \vdash \text{let } \beta_2 \text{ be } (\text{let } x \text{ be } y \text{ in } f[x_L/y]_{LM}) \text{ in } \sigma(\beta_2) : B$$

Finally, if we substitute x_L for y in the above, we can rewrite the expression using L-C to:

$$\Gamma, x : A; \cdot \vdash \text{let } \beta_2 \text{ be } f_{LM} \text{ in } \sigma(\beta_2) : B \equiv$$
$$\Gamma, x : A; \cdot \vdash \sigma(\beta_2)[f_M/\beta_2] \equiv f : B$$

so by L-U, we can transform this equality to the desired

$$\Gamma; y : A_L \vdash \text{let } \beta_2 \text{ be } (\text{let } x \text{ be } y \text{ in } f[x_L/y]_{LM}) \text{ in } \sigma(\beta_2) \equiv f : B$$

It remains to show that for any other term $\Gamma; x : A \vdash h : B_M$ such that $\epsilon_B \circ Lh = f$, we have $g = h$. Syntactically, $\epsilon_B \circ Lh = f$ corresponds to the judgement:

$$\Gamma; y : A_L \vdash \text{let } \beta_2 \text{ be } (\text{let } x \text{ be } y \text{ in } h_L) \text{ in } \sigma(\beta_2) \equiv f : B$$

If we weaken the cartesian context by $x : A$, we can substitute x_L for y and get:

$$\Gamma, x : A; \cdot \vdash \text{let } \beta_2 \text{ be } (\text{let } x \text{ be } x_L \text{ in } h_L) \text{ in } \sigma(\beta_2) \equiv \sigma(h) \equiv f[x_L/y] : B$$

finally, we apply M and get:

$$\Gamma, x : A \vdash \sigma(h)_M \equiv h \equiv f[x_L/y]_M : B$$

□

5.4.1 Linear dependent types

Since we allow linear types to dependend on terms of cartesian types, we can form new versions of the Π- and Σ-types. We will denote these linear variants of Π- and Σ-types by $\sqcap$ and $\sqsubset$, respectively.

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma, x : A \vdash B \text{ linear}}{\Gamma \vdash \sqcap_{x:A} B \text{ linear}} \sqcap\text{-F} \qquad \frac{\Gamma \vdash A \text{ type} \quad \Gamma, x : A \vdash B \text{ linear}}{\Gamma \vdash \sqsubset_{x:A} B \text{ linear}} \sqsubset\text{-F}$$

$$\frac{\Gamma, x : A; \Xi \vdash b : B}{\Gamma; \Xi \vdash \lambda x.b : \sqcap_{x:A} B} \sqcap\text{-I} \qquad \frac{\Gamma \vdash s : A \quad \Gamma; \Xi \vdash b : B[s/x]}{\Gamma; \Xi \vdash (s, b) : \sqsubset_{x:A} B} \sqsubset\text{-I}$$

$$\frac{\Gamma; \Xi \vdash t : \sqcap_{x:A} B \quad \Gamma \vdash a : A}{\Gamma; \Xi \vdash t(a) : B[a/x]} \sqcap\text{-E} \qquad \frac{\Gamma; \Xi \vdash t : \sqsubset_{x:A} B \quad \Gamma, x : A; \Xi', y : B \vdash c : C}{\Gamma; \Xi, \Xi' \vdash \text{let } x, y \text{ be } t \text{ in c} : \text{C}} \sqsubset\text{-E}$$

$$\frac{\Gamma; \Xi \vdash \lambda x.b(a) : \sqcap_{x:A} B}{\Gamma; \Xi \vdash \lambda x.b(a) \equiv b[a/x] : B[a/x]} \sqcap\text{-C} \qquad \frac{\Gamma; \Xi \vdash \text{let } x, y \text{ be } (s, t) \text{ in c} : \text{C}}{\Gamma; \Xi \vdash \text{let } x, y \text{ be } (s, t) \text{ in c} \equiv c[s/x][t/y]} \sqsubset\text{-C}$$

An example of a $\sqcap$ type, consider for any linear type $\Gamma \vdash A$ linear, the n-fold tensor product $A^n :\equiv A \otimes A \ldots A$, which we can define by induction over the natural numbers (using universes) via:

$$A^0 :\equiv I$$
$$A^{\text{suc }(n)} :\equiv A \otimes A^n$$

with this construct, recall the example of burning hydrogen given in 4, now expressed as a linear function $burn : O_2 \otimes H_2 \otimes H_2 \multimap H_2O \otimes H_2O$. With a dependent, linear function type, we can generalize this process to the function:

$$burn : \sqcap_{n:\mathbb{N}} O_2^n \otimes H_2^{2n} \multimap H_2O^{2n}$$

An example of the $\sqsubset$ type comes from Krishnaswami's treatment of linear dependent logic as a way to model imperative programs [9]. Here, a primitive type of memory locations, $\Gamma \vdash Loc$ type, is introduced, the terms x of which can reference a term of any cartesian type A by means of a term of the linear pointer type $[x \mapsto A]$:

$$\frac{\Gamma \vdash A \text{ type}}{\Gamma, x : Loc \vdash [x \mapsto A] \text{ linear}}$$

Given a term $\Gamma \vdash t : A$, one can allocate a memory block of memory to store t at a new location, returning a pointer:

$$\frac{\Gamma \vdash t : A}{\Gamma; \cdot \vdash new_A(t) : \sqsubset_{x:Loc} [x \mapsto A]}$$

Another example takes inspiration from the realm of blockchain technology, where one deals with resources like bitcoin or ether balances. Say we model ownership of ether by a public address (generated from a cryptographic public key) having the linear type $addr_1 : ETH$. Then we can model a decentralized mathematical prize comittee, which give out a reward of 1 ether to whoever proves the Goldbach conjecture.
Since the reward is constant, this is ais not truly a dependent linear function and we abusively notate it by $\multimap$:

$$\cdot; x : ETH \vdash GoldbachReward : \Pi_{x:\mathbb{N}} \Sigma_{y,z:\mathbb{N}} Prime(y) \times Prime(z) \times y + z =_{\mathbb{B}} 2(suc(suc(x))) \multimap ETH$$

To execute the transaction, one would need to exhibit that there is an address x with one ether to begin with and provide a proof of the Goldbach conjecture.

5.5 Universes

Our cartesian and linear types live in separate universes, U and L. Since linear types are type checked in a purely cartesian context, the linear universe is a cartesian type. Our universes are given á la Tarski and are closed under the type formers associated with them. Below, we only outline the closure over the type formers Π, $\otimes$ and $\sqcap$, but this should be sufficient to give the idea. For more detail we refer to Hofmann's *Syntax and Semantics of Dependent types*, [5], or Krishnaswami's *Integrating Linear and Dependent types*, [9].

$$\frac{\vdash \Gamma \text{ ctxt}}{\Gamma \vdash U \text{ type}}\ \mathcal{U}\text{-F} \qquad \frac{\vdash \Gamma \text{ ctxt}}{\Gamma \vdash L \text{ type}}\ \text{L-F}$$

$$\frac{\Gamma \vdash t : U}{\Gamma \vdash El(t) \text{ type}}\ \text{U-El-F} \qquad \frac{\Gamma \vdash t : L}{\Gamma \vdash El(t) \text{ linear}}\ \text{L-El-F}$$

$$\frac{\Gamma \vdash t : L \quad \Gamma \vdash s : L}{\Gamma \vdash t \hat{\otimes} s : L}\ \text{L-}\otimes$$

$$\frac{\Gamma \vdash t : U \quad \Gamma, x : El(t) \vdash v : U}{\Gamma \vdash \hat{\Pi}_{x:El(t)} v}\ \text{U-}\Pi \qquad \frac{\Gamma \vdash t : L \quad \Gamma \vdash s : L}{El(t \hat{\otimes} s) \equiv El(s) \otimes El(t)}\ \text{L-}\otimes\text{-Ty}$$

$$\frac{\Gamma \vdash t : U \quad \Gamma, x : El(t) \vdash v : U}{\Gamma \vdash El(\hat{\Pi}_{x:El(t)} v) \equiv \Pi_{x:El(t)} El(v)}\ \text{U-}\Pi\text{-Ty} \qquad \frac{\Gamma \vdash t : U \quad \Gamma, x : El(t) \vdash s : L}{\Gamma \vdash \hat{\sqcap}_{x:El(t)} s : L}\ \text{L-}\sqcap$$

$$\frac{\Gamma \vdash t : U \quad \Gamma, x : El(t) \vdash s : L}{\Gamma \vdash El(\hat{\sqcap}_{x:El(t)} s) \equiv \sqcap_{x:El(t)} El(s)}\ \text{L-}\sqcap\text{-Ty}$$

6 Semantics

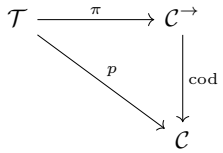
6.1 Structural semantic core

To explore the models of linear dependent type theory we begin by constructing a categorical structure which abstracts the key features of the theory. We will utilize the notion of a *comprehension category*, which provides the most general structure in which we can deal with the structural rules like context extensions and substitutions. Once this has been taken care of we may consider what extra conditions have to be imposed in order for the model to support various type constructors, and then provide concrete models that satisfy these conditions.

There are only two type constructors that will be assumed in the general semantic structure: the linear tensor product and unit. This simplifies the core semantics by allowing us to use symmetric monoidal categories instead of multicategories when interpreting linear types.

The idea behind the core of the semantics is to construct a comprehension category [7], consisting of a base category of contexts, a 'cartesian' fibration consisting of cartesian types equipped with context extensions, and a lax symmetric monoidal fibration consisting of linear types.

Definition 6.1. A **comprehension category** consists of a commutative diagram of functors



where $B^{\to}$ is the arrow category of B and $cod : B^{\to} \to B$ denotes the codomainfunctor such that:

1. $\mathcal{C}$ has a terminal object
2. $p : \mathcal{T} \to \mathcal{C}$ is a Grothendieck fibration,
3. $\pi : \mathcal{T} \to \mathcal{C}^{\to}$ takes cartesian morphisms in $\mathcal{T}$ to cartesian morphisms in $\mathcal{C}^{\to}$

Notice that by the second condition, cartesian morphisms in $\mathcal{T}$ are mapped to pullback squares in $\mathcal{C}$:

A cartesian morphism $(p, q) : f \to g$ in $\mathcal{C}^{\to}$ is a commutative square in $\mathcal{C}$

$$\begin{array}{ccc} B & \xrightarrow{g} & B' \\ \downarrow{\scriptstyle p} & & \downarrow{\scriptstyle q} \\ A & \xrightarrow{f} & A' \end{array}$$

such that for any E', E, $q' : E' \to B'$, $e : E \to E'$ and $p' : E \to A$ as in the following diagram:

$$\begin{array}{ccc} E & \xrightarrow{e} & E' \\ \downarrow{\scriptstyle id_E} & & \downarrow{\scriptstyle q'} \\ E & \xrightarrow{q'e} & B' \\ \downarrow{\scriptstyle p'} & & \downarrow{\scriptstyle q} \\ A & \xrightarrow{f} & A' \end{array}$$

there is a unique arrow $u : E \to B$ such that $p' = p \circ u$ and $q'e = g \circ u$. This is precisely the universal property of the pullback. If $\mathcal{C}$ has all pullbacks, then $\mathrm{cod} : \mathcal{C}^{\to} \to \mathcal{C}$ is a fibration and we immediately have a comprehension category. Maps of $\mathcal{C}$ that arise via $\pi(A)$ for some $A \in \mathcal{T}$ will be called *projections*, and by the remark above, pullbacks of projections always exist. Terms of a type will be interpreted as sections of the corresponding projections. The pullbacks of $\mathcal{C}$ will play the role of substitutions in the type theory, and since substitution is strictly associative in the theory, we need pullbacks of $\mathcal{C}$ to be strictly associative too. This requirement forces us to restrict our attention to *split comprehension categories*, i.e. comprehension categories such that p is a split fibration.

A semantics for linear dependent type theory consists of a split comprehension $p : \mathcal{T} \to \mathcal{C}$ and a split lax monoidal fibration $q : \mathcal{L} \to \mathcal{C}$, illustrated in the following picture:

$$\begin{array}{ccccc} \mathcal{L} & & \mathcal{T} & \xrightarrow{\pi} & \mathcal{C}^{\to} \\ & {\scriptstyle q}\searrow & \downarrow{\scriptstyle p} & \swarrow{\scriptstyle \mathrm{cod}} & \\ & & \mathcal{C} & & \end{array}$$

This structure is sufficient for interpreting the structural core of our theory. We make this precise by proving soundness by induction on the derivation rules of Figure 2. The following lemma will come in handy:

Lemma 7. The pullback of an arrow, g, has a section if g does. In other words, given a pullback of g:

$$\begin{array}{ccc} A \times_C B & \xrightarrow{k} & B \\ \downarrow{\scriptstyle g^*} & & {\scriptstyle g}\downarrow\uparrow{\scriptstyle h} \\ A & \xrightarrow{f} & C \end{array}$$

such that $gh = 1_C$, then h^* exists and is a section of g^*.

Proof. Since A with projections 1_A and hf forms a cone to the cospan, there exists a unique map $u : A \to A \times_C B$ such that $hfu = k$ and $g^*u = 1_A$, making u a section of g^*. □

Theorem 7.1 (Soundness)**.** *A split comprehension category $p : \mathcal{T} \to \mathcal{C}$ together with a split lax monoidal fibration $q : \mathcal{L} \to \mathcal{C}$ is a model for the linear dependent type theory consisting of the structural rules presented in Figure 2.*

Proof. We construct an interpretation function $[[-]]$, which sends:

- Cartesian contexts Γ to objects of $\mathcal{C}$, considered up to definitional equality and renaming of bound variables.
- Mixed contexts $\Gamma; \Xi$ to objects of $\mathcal{C}_{[[\Gamma]]}$.
- Cartesian types A in Γ to objects of $\mathcal{T}_{[[\Gamma]]}$.
- Linear types B in Γ to objects of $\mathcal{L}_{[[\Gamma]]}$.
- Cartesian terms $M : A$ in Γ to sections of $\pi([[A]]) : [[\Gamma, A]] \to [[\Gamma]]$.
- Linear terms $b : B$ in $\Gamma; \Xi$ to morphisms $[[b]] : [[\Xi]] \to [[B]]$.

Proceeding by induction on the derivation rules, we will often abuse notation slightly and denote semantic objects the same as their syntatic counterparts.

- Case of CI-Base: $[[\cdot]]$ is the terminal object $\mathbf{1}$ of $\mathcal{C}$.
- Case of CM-Base: $[[\cdot;\cdot]]$ is the unit of $\mathcal{L}_{\mathbf{1}}$.
- Case of C-cart-ext: By the induction hypothesis, we are given A in $\mathcal{T}_\Gamma$ and need to display an object $\Gamma.A$ in $\mathcal{C}$. This object is the domain of the morphism that A is mapped to via π:

$$\Gamma.A \xrightarrow{\pi_A} \Gamma$$

- Case of C-weak-1: Given $A, \Delta \in \mathcal{T}_\Gamma$, we send Δ through the functor $\pi_A^* : \mathcal{T}_\Gamma \to \mathcal{T}_{\Gamma.A}$ induced by lifting the morphism $\pi_A : \Gamma.A \to \Gamma$. The resulting object $\pi_A^*(\Delta)$ will be sent to the context $p(\pi_A^*(\Delta)) = \Gamma.A.\Delta\{\pi_*\}$ via p, which is the interpretation of the conclusion of the rule C-weak-1.
- Case of C-weak-2: Notice that since π is a cartesian functor, the context above fits into the following pullback:

$$\begin{array}{ccc} \Gamma.A.\Delta\{\pi_A\} & \xrightarrow{q} & \Gamma.\Delta \\ \downarrow{\scriptstyle \pi_{\Delta\{\pi_A\}}} & & \downarrow{\scriptstyle \pi_\Delta} \\ \Gamma.A & \xrightarrow{\pi_A} & \Gamma \end{array}$$

  so by lifting q to the monoidal functor $q^* : \mathcal{L}_{\Gamma.\Delta} \to \mathcal{L}_{\Gamma.A.\pi^*(\Delta)}$. The context we are looking for is the image of Ξ under this functor.
- Case of C-lin-ext: For objects Ξ and A in $\mathcal{L}_\Gamma$, we let their tensor product $\Xi \otimes A$ denote the extended context $\Gamma; \Xi, x : A$.
- Case of Lin-exch. Since our lax monoidal fibration is symmetric, we have $\Xi \otimes A \otimes B \otimes \Xi' \cong \Xi \otimes B \otimes A \otimes \Xi'$ in $\mathcal{L}_\Gamma$. Applying exchange to a judgement corresponds to composing with this isomorphism.
- Case of Weak-L. For any $A \in \mathcal{T}_\Gamma$ and morphism $t : \Xi \to A$ in $\mathcal{L}_{\Gamma,\Gamma'}$, we can transfer t along the functor $q_{A,\Gamma}^* : \mathcal{L}_{\Gamma,\Gamma'} \to \mathcal{L}_{\Gamma.A.\Gamma'\{\pi_A\}}$ induced by the map $q_{A,\Gamma'}$ arising from the following pullback diagram:

$$\begin{array}{ccc} \Gamma.A.\Gamma'\{\pi_A\} & \longrightarrow & \Gamma.\Gamma' \\ \downarrow{\scriptstyle q_{A,\Gamma'}} & & \downarrow{\scriptstyle \pi'_\Gamma} \\ \Gamma.A & \xrightarrow{\pi_A} & \Gamma \end{array}$$

  to yield a morphism $q_{A,\Gamma'}^*(t) : q_{A,\Gamma'}^*(\Xi) \to q_{A,\Gamma'}^*(A')$.
- Case of Weak-I. The morphism $q_{A,\Gamma'}$ above also induces a functor $q_{A,\Gamma'}^* : \mathcal{T}_{\Gamma,\Gamma'} \to \mathcal{T}_{\Gamma.A.\Gamma'.\{\pi_A\}}$.
- For Int-subst-1, the judgement $\mathcal{J}$ can take three forms:

$$\mathcal{J} = B[M/x] \text{ type}$$

$$\mathcal{J} = b[M/x] : B[M/x]$$

$$\mathcal{J} = B[M/x] \text{ linear}$$

  The corresponding categorical structure work by way of relating substitution in the theory to pullbacks in $\mathcal{C}$. We begin with showing that there is a type $B[M/x]$ in the context $\Gamma.\Gamma'[M/x]$.
  Suppose we are given the following objects of $\mathcal{T}$:

$$A \in \mathcal{T}_\Gamma$$
$$\Gamma' \in \mathcal{T}_{\Gamma.A}$$
$$B \in \mathcal{T}_{\Gamma.A.\Gamma'}$$

  and a section:

$$M : \Gamma \to \Gamma.A$$

  of the projection π_A. We first lift M to a cartesian arrow $M^* : \Gamma.\Gamma'\{M\} \to \Gamma'$, which is sent to the following pullback square by π:

$$\begin{array}{ccc} \Gamma.\Gamma'\{M\} & \xrightarrow{q_{(M,\Gamma')}} & \Gamma.A.\Gamma' \\ \downarrow{\scriptstyle \pi_{\Gamma'\{M^*\}}} & & \downarrow{\scriptstyle \pi_{\Gamma'}} \\ \Gamma & \xrightarrow{M} & \Gamma.A \end{array}$$

where $q_{(M,\Gamma')}$ is some map whose existance is given by the structure of the comprehension. We then lift $q_{(M,\Gamma')}$ to obtain a cartesian arrow $q^*_{(M,\Gamma')} : \Gamma.\Gamma'\{M^*\}.B\{q^*_{(M,\Gamma')}\} \to \Gamma.A.\Gamma'.B$ which fits in the following pullback diagram:

$$\begin{array}{ccc} \Gamma.\Gamma'\{M^*\}.B\{q^*_{(M,\Gamma')}\} & \xrightarrow{q_{(q_{(M,\Gamma')},B)}} & \Gamma.A.\Gamma'.B \\ \downarrow{\scriptstyle \pi_{B\{q^*_{(M,\Gamma')}\}}} & & \downarrow{\scriptstyle \pi_B} \\ \Gamma.\Gamma'\{M^*\} & \xrightarrow{q_{(M,\Gamma')}} & \Gamma.A.\Gamma' \end{array}$$

The element $\Gamma.\Gamma'\{M^*\}.B\{q^*_{M,\Gamma'}\}$ of $\mathcal{T}_{\Gamma.\Gamma'\{M^*\}}$ along with its associated projection will be our interpretation of $\Gamma.\Gamma'[M/x] \vdash B[M/x]$.
Now suppose there is a section $b : \Gamma.A.\Gamma' \to \Gamma.A.\Gamma'.B$ of the projection π_B. To display an element of $B[M/x]$ is to give a section of $\pi_{B\{q^*_{(M,\Gamma')}\}}$. By lemma 7, we get such a section by pulling back b along $q_{(q_{(M,\Gamma')},B)}$.
Finally, if B is an object of $\mathcal{L}_{\Gamma.A.\Gamma'}$, then the image of B under the functor $q^*_{(M,\Gamma')} : \mathcal{L}_{\Gamma.A.\Gamma'} \to \mathcal{L}_{\Gamma.\Gamma'\{M^*\}}$ will be our interpretation of $B[M/x]$ as a linear type in the context $\Gamma, \Gamma[M/x]$.

- Case of Int-subst-2. The interpretation of Int-subst-2 is the image of t under $q^*_{(M,\Gamma')}$.
- Case of Lin-subst. Given morphisms $t : \Xi \otimes A \to B$ and $M : \Xi' \to A$ we get a morphism $t \circ (id_\Xi \otimes M) : \Xi \otimes \Xi' \to B$. Precomposing with the isomorphism $\bigotimes \Xi, \Xi' \cong \Xi \otimes \Xi'$ yields the desired morphism $\bigotimes \Xi, \Xi' \to B$.
- Case of Int-var. For any $A \in \mathcal{T}_\Gamma$, the pullback:

$$\begin{array}{ccc} \Gamma.A.A\{\pi_A\} & \xrightarrow{q} & \Gamma.A \\ \downarrow{\scriptstyle \pi_{A\{\pi_A\}}} & & \downarrow{\scriptstyle \pi_A} \\ \Gamma.A & \xrightarrow{\pi_A} & \Gamma \end{array}$$

is in fact (isomorphic to) the product $\Gamma.A \times \Gamma.A$, and so the diagonal map $\Delta : \Gamma.A \to \Gamma.A \times \Gamma.A$ forms a section to both projections. This map is thus the interpretation of the judgement $\Gamma, x : A \vdash x : A$.

- Case of Lin-var. The morphism corresponding to the linear variable term $\Gamma; x : A \vdash x : A$ is given by the identity morphism on A in $\mathcal{L}_\Gamma$.

□

7.1 Semantic type formers

In the following, we assume that the comprehension category comprising the core of our syntax is full.

Definition 7.1. A **model of LDTT** is a full, split comprehension category $\pi : \mathcal{T} \to \mathcal{C}^\to$ and a split lax monoidal fibration $q : \mathcal{L} \to \mathcal{C}$.

Definition 7.2. A model of LDTT **supports I-types** if, for every Γ, there is exists an object $[[I]] \in \mathcal{L}_\Gamma$, equipped with a morphism $[[*]] : I \to [[I]]$, such that for every pair of maps $t : \Xi' \to [[I]]$ and $c : \Xi \to C$, there exists a map $[[\text{let } t \text{ be } * \text{ in } c]] : \bigotimes \Xi, \Xi' \to C$ such that $[[\text{let } * \text{ be } * \text{ in } c]] = c$.

This is trivially satisfied by every model by setting $[[I]] = I$, $[[*]] = 1_I$, and $[[\text{let } t \text{ be } * \text{ in } c]]$ to the composite $\rho(c \otimes [[t]])\alpha : \bigotimes \Xi, \Xi' \cong \Xi \otimes \Xi' \to C \otimes I \cong C$, where ρ is the right unitor for the monoidal structure of $\mathcal{L}_\Gamma$, and α is the isomorphism $\bigotimes \Xi, \Xi' \cong \Xi \otimes \Xi$. As for the identity $\rho(c \otimes 1_I)\alpha = c$, notice that $\alpha : \bigotimes \Xi \cong \Xi \otimes [[\cdot]] = \Xi \otimes I$ is given by ρ^{-1}, so we get $\rho(c \otimes 1_I) = c\rho$, from the naturality of ρ.

Definition 7.3. A model of LDTT **supports $\otimes$-types**, if, for every $A, B \in \mathcal{L}_\Gamma$, there exists an object $[[A \otimes B]] \in \mathcal{L}_\Gamma$, such that for any arrows $a : \Xi \to A$, $b : \Xi' \to B$, there exists an arrow $[[(a \otimes b)]] : \bigotimes Xi, \Xi \to [[A \otimes B]]$ and for arrows $t : \Xi' \to [[A \otimes B]]$ and $c : \bigotimes \Xi \otimes A \otimes B \to C$, there exists an arrow $[[\text{let } x \otimes y \text{ be } t \text{ in } c : C]]$ such that $[[\text{let } x \otimes y \text{ be } a \otimes b \text{ in } c]] = c$.

Again, it is not hard to see that there is a canonical interpretation of this in any model of LDTT sending $[[A \otimes B]]$ to $[[A]] \otimes [[B]]$. Some care is needed in order to always make sure contexts are left associated.
Similarly, one can define what it means for a model of LDTT to support $\multimap, \oplus, \&, \top$ and 0. This is well established and not particularly relevant for the interplay between linear and dependent types we explore here. For a more detailed treatment we refer the reader to [**?**]. For our current purposes, we may think of these conditions as corresponding to weak versions of internal homs, binary products and coproducts, and terminal and initial object respectively. Therefore, whenever the fibers of our lax monoidal fibration is monoidal closed, complete or cocomplete, we know that it supports the corresponding type formers.
What it means for a model of linear dependent type theory to *support* Π-types is directly inhererited from the standard, non-linear case.

Definition 7.4. A model of LDTT **supports Π-types** if, for all $A \in \mathcal{T}_\Gamma$, the induced functor $\pi_A^* : \mathcal{T}_\Gamma \to \mathcal{T}_{\Gamma.A}$ has a right adjoint $\Pi_A : \mathcal{T}_{\Gamma.A} \to \mathcal{T}_\Gamma$ satisfying the following Beck-Chevalley condition:
For all pullbacks in $\mathcal{C}$:

$$\begin{array}{ccc} \Gamma.E & \xrightarrow{q_{E,E'}} & \Delta.E' \\ \downarrow{\scriptstyle \pi_E} & & \downarrow{\scriptstyle \pi_{E'}} \\ \Gamma & \xrightarrow{f} & \Delta \end{array} \tag{1}$$

inducing the following functors between fibers:

$$q_{E,E'}^* : \mathcal{T}_{\Delta.E'} \to \mathcal{T}_{\Gamma.E}$$
$$f^* : \mathcal{T}_\Delta \to \mathcal{T}_\Gamma$$
$$\Pi_E : \mathcal{T}_{\Gamma.E} \to \mathcal{T}_\Gamma$$
$$\Pi_{E'} : \mathcal{T}_{\Delta.E'} \to \mathcal{T}_\Delta$$

The canonical natural transformation $f^*\Pi_{E'} \to \Pi_E q_{E,E'}^*$ induced by the adjunction is a natural isomorphism.

As the rules Σ contains one more eliminator than usual (Σ-E$_2$ in Figure 3), one might wonder whether this requires additional conditions for the semantic type formers to ensure that these are well behaved with respect to the linear fibers. We will shortly see that this is not the case.

Definition 7.5. A model of LDTT **supports Σ-types** if it satisfies the following:

1. For all $A \in \mathcal{T}_\Gamma$, the induced functor $\pi_A^* : \mathcal{T}_\Gamma \to \mathcal{T}_{\Gamma.A}$ has a left adjoint, Σ_A,
2. such that for all pullbacks (as in 1), the natural transformation: $\Sigma_E q^* \to f^*\Sigma_{E'}$ is a natural isomorphism, and
3. the induced map $pair_{A,B} : \Gamma.A.B \to \Gamma.\Sigma_A B$ is an isomorphism

The map $pair_{A,B} : \Gamma.A.B \to \Gamma.\Sigma_A B$ such that $\pi_{\Sigma_A B} pair_{A,B} = \pi_A \pi_B$, arises as the image of the unit $\eta : B \to \pi_A^*(\Sigma_A B)$ in $\mathcal{T}_A$ under the comprehension functor $\pi : \mathcal{T} \to \mathcal{C}^\to$. The inverse of the pairing map will be denoted $(pr_1, pr_2) : \Gamma.\Sigma_A B \to \Gamma.A.B$.
This structure is sufficient to support new elimination rule (Σ-E$_2$):

Theorem 7.2. *If a model of LDTT supports Σ-types, then for every object $C \in \mathcal{L}_{\Gamma.\Sigma_A B}$ and morphism $c : \Xi \to C\{pair_{A,B}\}$ in $\mathcal{L}_{\Gamma.A.B}$ and section $s : \Gamma \to \Gamma.\Sigma_A B$, there exists a morphism $\hat{c}_s : \Xi\{(pr_1, pr_2\} \to C\{s\}$ such that given sections $a : \Gamma \to \Gamma.A$ and $b : \Gamma.A \to \Gamma.A.B$, then $\hat{c}_{(a,b)} = c\{ba\} : \Xi\{ba\} \to C\{ba\}$.*

Proof. The situation is illustrated in the following diagram:

$$\begin{array}{ccc} & \overset{(pr_1,pr_2)}{\curvearrowleft} & \\ \Gamma.A.B & \underset{pair_{A,B}}{\xrightarrow{\quad}} & \Gamma.\Sigma_A B \\ \downarrow{\scriptstyle \pi_{A,B}} & \swarrow{\scriptstyle \pi_{\Sigma_A B}} & \\ \Gamma & & \end{array}$$

Let $\hat{c}_s = c\{((pr_1, pr_2)s)\}$. First, this morphism has the correct target since we have

$$C\{((pr_1, pr_2)s)\}\{pair_{A,B}\} = C\{(pair_{A,B}(pr_1, pr_2)s)\} = C\{s\}$$

, relying on the fact our lax monoidal fibration is split. Secondly, we need to show that given sections $a : \Gamma \to \Gamma.A$ and $b : \Gamma.A \to \Gamma.A.B$, we have $c\{((pr_1, pr_2)(a, b))\} = c\{a\}\{b\}$:

$$\{(pr_1, pr_2)(a, b)\} = \{(pr_1, pr_2)pair_{A,B}ba\} = \{ba\} = \{a\}\{b\}$$

□

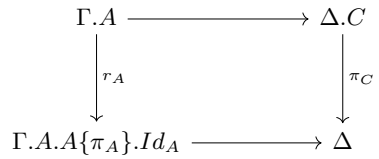
When it comes to Id-types, the situation is not as fortunate. If one wants to keep the theory intensional, we need to add condition (3) to make sure that the semantic identity types satisfy the added elimination rules =-E$_2$ and =-E$_3$ in Figure 4.

Definition 7.6 (Id-types)**.** A model of LDTT **supports Id-types** if, for all $A \in \mathcal{T}_\Gamma$, there exists an object $Id_A \in \mathcal{T}_{\Gamma.A.A\{\pi_A\}}$ and a morphism $r_A : \Gamma.A \to \Gamma.A.A\{\pi_A\}.Id_A$ such that:

1. The following diagram commutes:

$$\begin{array}{ccc} \Gamma.A & & \\ \downarrow{\scriptstyle r_A} & \searrow{\scriptstyle v_A} & \\ \Gamma.A.A\{\pi_A\}.Id_A & \xrightarrow{\pi_{Id_A}} & \Gamma.A.A\{\pi_A\} \end{array}$$

2. For any commutative diagram:



   there exists a lift $J : \Gamma.A.A\{\pi_A\}.Id_A \to \Delta.C$ making the two triangles commute.

3. For any pair of objects, $C \in \mathcal{L}_{\Gamma.A.A^+.Id}$ and $\Xi \in \mathcal{L}_{\Gamma.A}$, sections $M, N : \Gamma \to \Gamma.A$, $P : \Gamma.A.A^+ \to \Gamma.A.A^+.Id$, and morphism $c : \Xi \to C\{r_A\}$, there exists morphisms $\hat{c}^1_{[M,N,P]} : \Xi\{M\} \to C\{M\}\{N\}\{P\}$ and $\hat{c}^2_{[M,N,P]} : \Xi\{N\} \to C\{M\}\{N\}\{P\}$ such that $\hat{c}^1_{[M,M,refl]} = \hat{c}^2_{[M,M,refl]} = c\{M\}$.

A simpler, but stronger condition, implying condition (3) in the intensional setting is that for every $A \in \mathcal{T}_\Gamma$, there is an adjunction $\pi^*_{A,Id} \dashv r^*$ between the fibers of $\mathcal{L}$. This yields for every $c : \Xi \to C\{r\}$, a morphism $\hat{c} : \Xi\{\pi_{A,Id}\} \to C$, which, given sections $M, N : \Gamma \to \Gamma.A$ and $P : \Gamma.A.A^+ \to \Gamma.A.A^+.Id$, we can send to $\hat{c}\{M\}\{N\}\{P\} : \Xi\{M\}\{N\}\{P\}\{\pi_{A,Id}\} \to C\{M\}\{N\}\{P\}$, where $\Xi\{M\}\{N\}\{P\}\{\pi_{A,Id}\} = \Xi\{M\}$ since $\pi_{A,Id}PN = 1_{\Gamma.A}$.

If one were to work within an extensional type theory, where the rule:

$$\frac{\Gamma \vdash p : a =_A b}{\Gamma \vdash a \equiv b : A}$$

is added, then the third condition is not needed:

Theorem 7.3. *A model of LDTT with extensional equality supports Id-types if, for every $A \in \mathcal{T}_\Gamma$, there are objects $Id_A \in \mathcal{T}_{\Gamma.A}$ satisfying condtions 1 and 2 above.*

Proof. In extensional type theory, every identity term $\Gamma, x, y : A \vdash P : x =_A y$ can be shown to be equal to reflexivity by forming the identity type:

$$\Gamma, x, y : A, p : x =_A y \vdash p =_{x=_A x} refl(x) \text{ type}$$

this is well typed, since by $\Gamma, x, y : A, p : x =_A y \vdash p : x =_A y$, we conclude the definitional equality $x \equiv y$, so the types $x =_A y$ and $x =_A x$ are definitionally equal. We can then inhabit the type $p =_{x=_A x} refl(x)$ using the elimination principle from the equality $\Gamma, x : A, refl(x) : x =_A x \vdash refl(refl(x) : refl(x) =_{x=_A x} refl(x)$. Since we have inhabited the propositional equality, the equality holds judgementally by extensionality.
Since we identify judgementally equal terms in our semantics, this implies that any section of $\pi_{A.Id} : \Gamma.A.A^+.Id \to \Gamma.A$ must be equal to $r_A : \Gamma.A \to \Gamma.A.A^+.Id$. Condition 3 then reads rather trivially that for any morphism $c : \Xi \to C\{r_A\}$ and section $M : \Gamma \to \Gamma.A$, there should exist a morphism $\hat{c}_M : Xi\{M\} \to C\{M\}\{r\}$ which we get by $\hat{c}_M = c\{M\}$. □

The semantic counterpart for the mixed types $\sqcap$ and $\sqsubset$ is similar to that of Π and Σ, but instead of $\mathcal{T}$, we ask for adjoints of the induced functors between fibers of $\mathcal{L}$.

Definition 7.7. A model of LDTT **supports $\sqcap$-types** if, for all $A \in \mathcal{T}_\Gamma$, the induced (monoidal) functor $\pi^*_A : \mathcal{L}_\Gamma \to \mathcal{L}_{\Gamma.A}$ has a monoidal right adjoint. It supports $\sqsubset$-types if every π^*_A has a monoidal left adjoint.

As outlined in

Definition 7.8.

8 Models

8.1 Set indexed families

Our first model will be based on the standard set-theoretic interpretation of dependent type theory [5]. The extension of this model to the linear realm is fairly straightforward, and provides a good springboard for examples to come. (It is also sufficient for showing that linear dependent type theory is a proper generalisation of both dependent type theory and linear type theory, as outlined in [13].)

Our linear and cartesian fibrations will both be constructed as fibrations of set indexed families:

Definition 8.1 ($Fam(\mathcal{C})$)**.** For an arbitrary category $\mathcal{C}$, let $Fam(\mathcal{C})$ denote the category whose objects consists of pairs (S, f) where S is a set and f is a function $f : S \to Ob(\mathcal{C})$. Morphisms of $Fam(\mathcal{C})$ are pairs $(u, \alpha) : (S, f) \to (S', g)$ where $u : S \to S'$ and $\alpha : S \to \mathrm{Mor}(\mathcal{C})$ such that $\alpha(s) : f(s) \to g(u(s))$ for all $s \in S$. Composition is given, for two compatible morphisms (u, α) and (v, β), by:

$$(v, \beta) \circ (u, \alpha) = (v \circ u, g(u(s)) \circ f(s))$$

for all $s \in S$.

By projecting a family to its indexing set, we get a fibration $p : Fam(\mathcal{C}) \to \mathbf{Set}$, and if $\mathcal{C}$ has a terminal object, $\top$ such that the hom-sets $\mathcal{C}(\top, A)$ are small for all $A \in \mathcal{C}$, we can form a comprehension category by defining $\pi : Fam(\mathcal{C}) \to \mathbf{Set}^{\to}$ as follows.

On objects, let

$$\pi(S, f) = fst : \{(s, t) \mid s \in S, t : \top \to f(s)\} \to S$$

For a morphism $(u, \alpha) : (S, f) \to (S', g)$, let

$$q_{(u,\alpha)} : \{(s, t) \mid s \in S, t : \top \to f(s)\} \to \{(s', t') \mid s' \in S', t' : \top \to g(s')\}$$

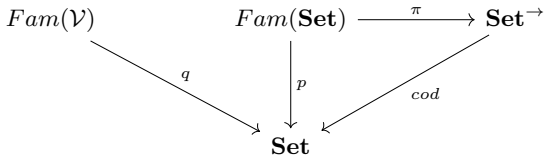
be defined by $q_{(u,\alpha)}(s, t) = (u(s), \alpha(s) \circ t)$. The functor π sends morphisms (u, α) to squares:

$$\begin{array}{ccc} \{(s, t) \mid s \in S, t : \top \to f(s)\} & \xrightarrow{q_{(u,\alpha)}} & \{(s', t') \mid s' \in S', t' : \top \to g(s')\} \\ \downarrow{\scriptstyle fst} & & \downarrow{\scriptstyle fst} \\ I & \xrightarrow{\quad u \quad} & J \end{array}$$

in **Set**.

The cartesian part of our semantic structure will simply be the fibration $p : Fam(\mathbf{Set}) \to \mathbf{Set}$ with $\pi : Fam(\mathbf{Set}) \to \mathbf{Set}^{\to}$ as outlined above. For the linear part, we choose any symmetric monoidal category $\mathcal{V}$ and form the fibration $q : Fam(\mathcal{V}) \to \mathbf{Set}$. With the functor $I : \mathbf{Set} \to Fam(\mathcal{V})$ defined by mapping a set S to the family constant at the unit of $\mathcal{V}$ and $\otimes : Fam(\mathcal{V}) \times_{\mathbf{Set}} Fam(\mathcal{V}) \to Fam(\mathcal{V})$ induced by the monoidal structure of $\mathcal{V}$, this forms a lax monoidal fibration.

We have the following picture:



8.2 Syntactic enriched categories

As an in depth example construction in the families model of linear dependent type theory, recall the notion of an enriched category in as defined in 4.9. Our goal is to construct a syntactic analogue of this construction in the families model, such that the interpretation of this is an enriched category. There are two ways of doing this. The first is a meta construction, in which we ask for the existence of certain types and judgemental equalities to hold, whereas the second, internal definintion, is a type **Enr-Cat**, whose terms are interpreted as enriched categories. The type **Enr-Cat** allows us to formulate theorems about enriched categories directly in the syntax, but requires more involved type formers, such as universes, and the following type, with which we can reason (non-linearly) about the hom-sets of $\mathcal{V}$:

$$\frac{\Gamma \vdash A \ \text{linear} \quad \Gamma \vdash B \ \text{linear}}{\Gamma \vdash [A, B] \ \text{type}} \ \text{hom-F} \qquad \frac{\Gamma \vdash f : [A, B] \quad \Gamma; \Xi \vdash a : A}{\Gamma; \Xi \vdash f(a) : B} \ \text{hom-E}$$

$$\frac{\Gamma; x : A \vdash t : B}{\Gamma \vdash [\lambda x.t] : [A, B]} \ \text{hom-I} \qquad \frac{\Gamma; x : A \vdash t : B \quad \Gamma; \Xi \vdash a : A}{\Gamma; \Xi \vdash [\lambda.x.t](a) \equiv t[a/x] : B} \ \text{hom-C}$$

In the presence of linear function types ($\multimap$) and the M-functor, this type can be just as well be constructed as $(A \multimap B)_M$. However, adding it as a primitive allows us to talk about the hom-sets of $\mathcal{V}$ even when it is not closed. Instead, the semantics of $[-, -]$ will be an object in $\mathcal{T}$ which "represents" the morphisms of $\mathcal{L}$:

Definition 8.2. A model of linear depenendent type theory supports $[-, -]$-types, if, for all objects $A, B \in \mathcal{L}_\Gamma$, there exists an object $T[A, B] \in \mathcal{T}_\Gamma$ such that $\mathcal{T}_\Gamma(X, T[A, B]) \cong \mathcal{L}_{\Gamma.X}(A, B)$.

Definition 8.3 (meta-theoretic enriched categories)**.** A meta-theoretic $\mathcal{V}$-enriched category in a context Γ consists of the following data:

- A (cartesian) type $\Gamma \vdash A$ type,
- for any $x, y : A$, a linear type $\Gamma, x, y : A \vdash B_{x,y}$ linear,
- terms $\Gamma, x, y, z : A; g : B_{y,z}, f : B_{x,y} \vdash M(g, f) : B_{x,z}$,
- $\Gamma, x : A; \cdot \vdash j_x : B_{x,x}$ and
- judgemental equalities $\Gamma, x, y : A; f : B_{x,y} \vdash f \equiv M(j_y \otimes f)$ and $\Gamma, x, y : A; f : B_{x,y} \vdash f \equiv M(f \otimes j_x)$.

The interpretation of this in the families model are precisely $\mathcal{V}$-enriched categories. The underlying set of objects will be $[[A]]$, the interpretation of B is a function $B : [[A]] \times [[A]] \to \mathcal{V}$, which for each pair $x, y \in A$ assigns an object of $\mathcal{V}$.

Definition 8.4 (Internal enriched categories)**.** The type **Enr-cat** is defined as

$$Enr{-}Cat :\equiv \Sigma_{A:\mathcal{U}}\Sigma_{B:\Pi_{x,y:A}}\mathcal{L}\Sigma_{M:\Pi_{x,y,z:A}[B_{y,z}\otimes B_{x,y},B_{x,z}]}\Sigma_{j:\Pi_{x:A}[I,B_{x,x}]}[\lambda f.f] = [\lambda f.M_{x,x,y}(f\otimes j_x)]\times[\lambda f.f] = [\lambda f.M_{x,x,y}(j_y\otimes f)]$$

Since $Fam(\mathbf{Set})$ is an extensional model, whenever the $a =_A b$ is inhabited, we have $[[a]] = [[b]]$. Furthermore, since the equalities p, q and r are all identifying morphisms in the image of the faithful functor M, we have:

$$[[f \circ 1_x]] \equiv [[1_y \circ f]] \equiv [[f]]$$

and

$$[[h \circ (g \circ f)]] \equiv [[(h \circ g) \circ f]]$$

for all $f : B_{x,y}$, $g : B_{y,z}$ and $h : B_{z,w}$, demonstrating that composition is unital and associative in the enriched category. For example, we may choose $\mathcal{V}$ to be the category of groups equipped with the usual tensor product of groups. Here the L functor forms the free abelian group of a set, and the construction above will yield an Ab-enriched category.

8.3 Diagrams

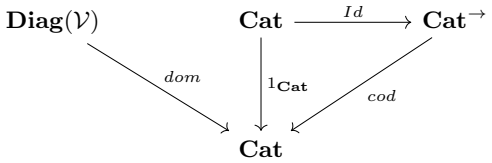
Expanding upon the Monoidal Families example given by Vakar [13], we consider diagrams $J : \mathcal{D} \to \mathcal{V}$ of any shape in a category $\mathcal{V}$. In other words, we construct a category $\mathbf{Diag}(\mathcal{V})$ whose objects are functors with $\mathcal{V}$ as codomain and whose morphisms for diagrams $J : \mathcal{D} \to \mathcal{V}$ and $J' : \mathcal{C} \to \mathcal{V}$ are given by functors $F : \mathcal{C} \to \mathcal{D}$ such that $J \circ F = J'$.

From this we get the obvious domain fibration $dom : \mathbf{Diag}(\mathcal{V}) \to \mathbf{Cat}$.

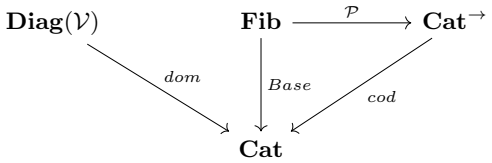
Theorem 8.1. *If $\mathcal{V}$ is a monoidal category, dom is a lax monoidal fibration.*

Proof. □

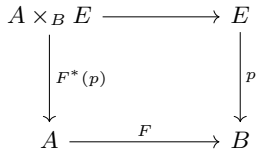
This suggests that we can get a model for linear dependent type theory by letting $\mathcal{L} = \mathbf{Diag}(\mathcal{V})$ in the semantic structures given above. But what about the choice of $\mathcal{T}$? One can trivially choose $\mathcal{T} = \mathbf{Cat}$ to get the following picture:



where $Id : \mathbf{Cat} \to \mathbf{Cat}^{\to}$ is the map sending a category to its identity functor. Another choice would be $\mathcal{T} = \mathbf{Fib}$, the category of grothendieck fibrations, yielding:



where $Base$ is the morphism sending a fibration $p : E \to B \mapsto B$ and $\mathcal{P}$ sends a fibration $p : E \to B$ to its underlying functor $p : E \to B$. For any functor $F : A \to B$ in $\mathbf{Cat}$, there is a pullback:



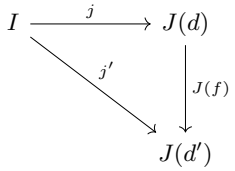
where $F^*(p)$ is a fibration if p is, by Lemma(Prop 2.6 in comprehension categories). This shows that $\mathcal{P}$ sends cartesian arrows to cartesian arrows. READ EXAMPLE 4.6 in Comprehension cats

One could also restrict $\mathbf{Diag}(\mathcal{V})$ to diagrams in $\mathcal{V}$ over groupoids. In that case, $\mathcal{T}$ consists of fibrations of groupoids and the cartsian fragment of the type theory is essentially the groupoid model as presented by Hofmann and Streicher in [6]. This suggests that this model can serve as a setting for linear dependent homotopy type theory.

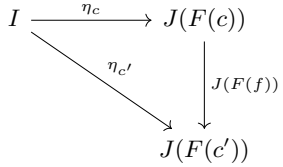
We expand this to a functor $\mathrm{Diag}(\mathcal{V}) : \mathbf{Cat}^{op} \to \mathbf{SMCat}(-, \mathbf{V})$ where $\mathcal{C} \xrightarrow{F} \mathcal{D}$ in $\mathbf{Cat}$ induces a monoidal functor $[\mathcal{D}, \mathcal{V}] \xrightarrow{(-)\circ F} [\mathcal{C}, \mathcal{V}]$. To highlight the connection to the type theory, we let $A\{F\}$ in $\mathbf{Diag}(\mathcal{V})(\mathcal{C})$ denote the image of any object $A \in \mathbf{Diag}(\mathcal{V})(\mathcal{D})$ under this functor. Notice that for any category $\mathcal{C}$, the functor category $[\mathcal{C}, \mathcal{V}]$ naturally carries a monoidal structure by $(F \otimes G)(c) = F(c) \otimes G(c)$, with the constant functor at the unit object $I \in \mathrm{Ob}(\mathcal{V})$ as unit. This functor will also be denoted I.

Theorem 8.2 (Comprehension in $\mathbf{Diag}(\mathcal{V})$)**.** *$Diag(\mathcal{V})$ admits a comprehension. In other words, for any diagram $J : \mathcal{D} \to \mathcal{V}$ there is a representating object $\mathcal{D}.J \xrightarrow{\mathbf{p}_{\mathcal{D}.J}} \mathcal{D}$ in $\mathbf{Cat}_{/\mathcal{D}}$ such that for any morphism $F : \mathcal{C} \to \mathcal{D}$, we have $\mathbf{Diag}(\mathcal{V})(\mathcal{C})(I, J\{F\}) \cong \mathbf{Cat}_{/\mathcal{D}}(F, \mathbf{p}_{(\mathcal{D},J)})$.*

Proof. Let $\mathcal{D}.J$ be given by the slice category $(I \downarrow J)$, whose objects are pairs (d, j), with $d \in \mathcal{D}$, $j \in \mathrm{Hom}(I, J(d))$ and whose morphisms are $f : d \to d'$ such the following diagram commutes:



Compare this with natural transformations $\eta \in \mathbf{Diag}(\mathcal{V})(\mathcal{C})(I, J\{F\}))$. For all objects $c, c' \in \mathcal{C}$ with morphism $c \xrightarrow{f} c'$ we require components $n_\bullet : I \to J \circ F(\bullet)$ such that the following diagram commutes:



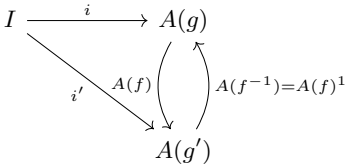
Let $p_{\mathcal{D}.J} : (I \downarrow J) \to \mathcal{D}$ be the obvious forgetful functor, $(d, j) \mapsto d$. Now any functor $G : \mathcal{C} \to (I \downarrow J)$ such that $p_{\mathcal{D}.J} G = F$, must be of the form:

$$G(c) = (F(c), g_c)$$
$$G(f) = F(f)$$

where $g_c : I \to F(c)$ is some morphism satisfying the commutativity conditions above. This uniqely defines a natural transformation $g \in \mathbf{Diag}(\mathcal{V})(\mathcal{C})(I, J\{F\})$. Furthermore, for any natural transformation $\eta \in \mathbf{Diag}(\mathcal{V})(\mathcal{C})(I, J\{F\}$, there is a corresponding functor $\hat{\eta} : \mathcal{C} \to (I \downarrow J)$ mapping $c \mapsto (F(c), \eta_c)$. □

Lemma 9. For any groupoid G and functor $A \in [G, \mathcal{V}]$ the comma category $(I \downarrow A)$ is a groupoid, and its associated projection $p_{G.A} : (I \downarrow A) \to G$ is a fibration.

Proof. A morphism between objects (g, i), (g', i') in $(I \downarrow A)$ are given by a morphism $f : g \to g'$ in G such that $A(f) \circ i = i'$. Since G is a groupoid there exists an inverse f^{-1}. That this map yields an inverse of $(I \downarrow A)$ is immediate from the fact that the following diagram in $\mathcal{V}$ commutes:

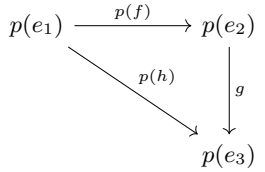


To see that p is a fibration, let (g, i) be an object of $(I \downarrow A)$ and $f : g \to g'$ an arrow in G. Then the object $(g', A(f) \circ i)$ together with f seen as a morphism of $(I \downarrow A)$ form an immediate commutative diagram in $\mathcal{V}$. □

Corollary 10. The restriction of $\mathbf{Diag}(\mathcal{V})$ to the category of groupoids is also a model for IDLTT.

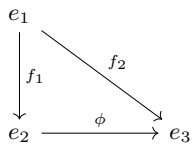
The functor $p_{\mathcal{D}.J}$ will sometimes be referred to as the **projection** associated to J. Recall that a model $\mathcal{C}, \mathcal{L}$ of ILDTT **supports** Σ**-types** if $\mathcal{L}(p_{\Delta.A})$ has a left adjoint for all $\Delta \in \mathcal{C}$, and $A \in \mathcal{L}(\Delta)$. It **supports** Π**-types** if all $\mathcal{L}(p_{\Delta.A})$ have right adjoints. In the $Diag(V)$ model, Σ- and Π-types are left and right adjoints to the functor $p_* : [\Delta, \mathcal{V}] \to [\Delta.A, \mathcal{V}]$ induced by the projection $p_{\Delta.A} : \Delta.A \to \Delta$. These are precisely the left and right Kan extensions along $p_{\Delta.A}$.

Definition 10.1 (coCartesian morphism)**.** Given a functor $p : E \to B$, an arrow $f : e_1 \to e_2$ is coCartesian with respect to p if, for any pair of morphisms $h : e_1 \to e_3$ and $g : p(e_2) \to p(e_3)$ as in the following commutative diagram:



there exists a unique morphism $\hat{g} : e_2 \to e_3$ such that $p(\hat{g}) = g$ and $\hat{g}f = h$.

Lemma 11. unique-target For a functor $p : E \to B$ and $f_1 : e_1 \to e_2$, $f_2 : e_1 \to e_3$ two coCartesian morphisms such that $p(f_1) = p(f_2) = \alpha : b_1 \to b_2$, there is a unique isomorphisms $\phi : e_2 \to e_3$ such that the following diagram commutes:



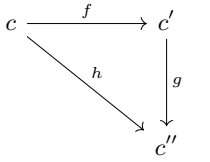
Proof. Since f_1 coCartesian, the composition $Id_{b_2} \circ p(f_1) = p(f_2)$ has a unique lift $\phi : e_2 \to e_3$ in E such that $\phi f_1 = f_2$. Similarly, there is a unqie morphism $\psi : e_3 \to e_2$ such that $\psi f_2 = f_1$. Composing, we get an arrow $\psi\phi f_1 = f_1$. But since we also have $Id_{b_2} \circ p(f_1) = p(f_1)$, the only arrow $\pi : e_2 \to e_2$ such that $\pi f_1 = f_1$ must be the identity on e_2, so ϕ is a isomorphism. □

Definition 11.1 (Opfibration)**.** A functor $p : E \to B$ is an **opfibration** if for every $e_1 \in E$ and arrow $\alpha : p(e_1) \to b$ in the base B, there exists a coCartesian arrow $f : e_1 \to e_2$ in E such that $p(f) = \alpha$.

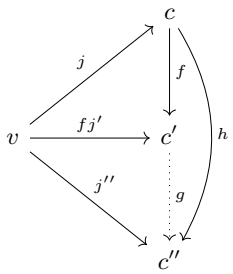
Given such a fibration, for any object $b \in B$, the objects and morphisms of E that are mapped to b and its identity morphism via p form a category, E_b, called the **fiber** over b. From any map $f : b \to b'$ in the base we can construct a functor $\hat{f} : E_b \to E_{b'}$ by sending $e \in E_b$ to the target of a corresponding coCartesian lift $g : e \to e'$. By the previous lemma, such a functor $\hat{f}$ will be unique up to unique isomorphism.

Theorem 11.1. *Given a category $\mathcal{C}$ and functor $A : \mathcal{C} \to \mathcal{V}$, the projection $p : (v \downarrow A) \to \mathcal{C}$ is an opfibration for any $v \in \mathcal{V}$.*

Proof. Let $(c, j : v \to A(c))$ be an object of $(v \downarrow A)$ and $f : c \to c'$ a morphism in $\mathcal{C}$. Then f induces a morphism of between (c, j) and (c', fj) in $(v \downarrow A)$. For any pair of morphisms $h : (c, j) \to (c'', j'')$ and $g : c' \to c''$ such that the following diagram commutes:



we have the following commutative diagram in $(v \downarrow A)$:



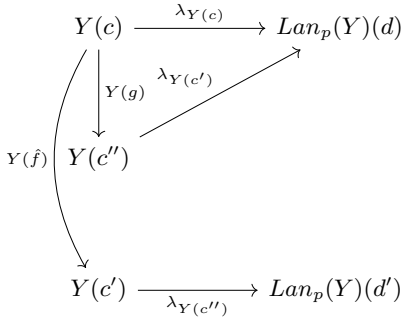
Furthermore, since morphisms in $(v \downarrow A)$ arise from morphisms in $\mathcal{C}$, g is the unique candidate. □

Theorem 11.2. *Let $p : \mathcal{C} \to \mathcal{D}$ be an opfibration and $\mathcal{V}$ cocomplete. Then there exists a functor $Lan_p : [\mathcal{C}, \mathcal{V}] \to [\mathcal{D}, \mathcal{V}]$ which is a left kan extension of p.*

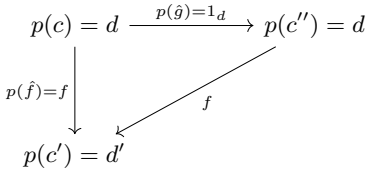
Proof. For any functor $Y \in [\mathcal{C}, \mathcal{V}]$, let $Lan_p(Y)$ be given by:

$$Lan_p(Y)(d) := \varinjlim(\mathcal{C}_d \hookrightarrow \mathcal{C} \xrightarrow{Y} \mathcal{V})$$

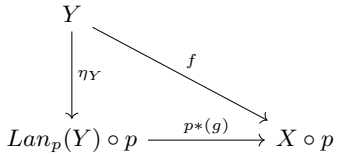
The action of $Lan_p(Y)$ on morphisms $f : d \to d'$, arises from the universal mapping property of the colimits. Specifically, one can distinguish between two cases. If d is not in the image of p, then $Lan_p(Y)(d)$ is the initial object in $\mathcal{V}$ and $Lan_p(Y)(f)$ the unique morphism from it. Otherwise, let $p(c) = d$ for some $c \in \mathcal{C}$ and $\hat{f} : c \to c'$ be a cocartesian arrow such that $p(\hat{f}) = f$. We want to show that $Lan_p(Y)(d')$ forms a cocone of the diagram to which $Lan_p(Y)(d)$ is a limit. To that end, let $g : c \to c''$ be an arbitrary morphism in $\mathcal{C}_d$. We have the following image:



But there should also exist a cocartesian arrow $\hat{g}$ corresponding to $1_d : p(c) \to p(c'')$, so from the following diagram in $\mathcal{D}$:



there must exist a unique arrow $f' : c'' \to c'$ such that $p(f') = f$ and $f'\hat{g} = \hat{f}$. This implies that $Lan_p(Y)(d')$ forms a cocone over the diagram in question, so we can define the $Lan_p(Y)(f)$ to be the unique map $Lan_p(Y)(d) \to Lan_p(Y)(d')$. The action of Lan_p on natural transformations $\epsilon : Y \to Z$ is similarly induced by the UMP of the colimits at each component. We show $Lan_p \dashv p_*$ by exhibiting the unit $\eta : 1_{[\mathcal{D},\mathcal{V}]} \to Lan_p p_*$ with the following universal property. For objects $Y \in [\mathcal{C}, \mathcal{V}]$, $X \in [\mathcal{D}, \mathcal{V}]$ and morphism $f : Y \to X \circ p$ there is a unique morphism $g : Lan_p(Y) \to X$ such that the following diagram commutes:



Again, g arises from the universal property of the colimit: For a given $d \in \mathcal{D}$, all $c \in \mathcal{C}$ such that $p(c) = d$ are mapped to the same object by $X \circ p$, so $X \circ p(c)$ forms a cocone over the diagram $\mathcal{C}_{p(c)} \hookrightarrow \mathcal{C} \xrightarrow{Y} \mathcal{V}$. But $Lan_p(Y)(p(c))$ is the colimit of that diagram, so the g arises from the unique map from the colimit to the cocone. □

Dually, we can use the same argument to conclude that when $\mathcal{V}$ has all limits, the right Kan extension along p exists.

Corollary 12. If $\mathcal{V}$ has all colimits, then $\mathbf{Diag}(\mathcal{V})$ supports Σ-types. If $\mathcal{V}$ has all limits, then $\mathbf{Diag}(\mathcal{V})$ supports Π-types.

12.1 Spectra

13 Ideas

H-spaces generalized to the tensor product?

13.1 Dependent ordered type theory

If one were to drop exchange as a structural rule, this should correspond to dropping the requirement of the semantic category being symmetric. A particular construction which might be interesting to explore in this setting would be simplicial sets equipped with the join. Would this work?

References

[1] A mixed linear and non-linear logic: Proofs, terms and models (preliminary report).

[2] Andrew Barber. *Dual intuitionistic linear logic*. 1996.

[3] P Benton. A mixed linear and non-linear logic: Proofs, terms and models. In *Computer Science Logic*, pages 121–135. Springer, 1995.

[4] Ross Duncan et al. *Types for quantum computing*.

[5] Martin Hofmann. Syntax and semantics of dependent types. In *Extensional Constructs in Intensional Type Theory*, pages 13–54. Springer, 1997.

[6] Martin Hofmann and Thomas Streicher. The groupoid interpretation of type theory. *Twenty-five years of constructive type theory (Venice, 1995)*, 36:83–111, 1998.

[7] Bart Jacobs. Comprehension categories and the semantics of type dependency. *Theoretical Computer Science*, 107(2):169–207, 1993.

[8] Max Kelly. *Basic concepts of enriched category theory*, volume 64. CUP Archive, 1982.

[9] Neelakantan R Krishnaswami, Pierre Pradic, and Nick Benton. Integrating linear and dependent types. In *ACM SIGPLAN Notices*, volume 50, pages 17–30. ACM, 2015.

[10] Daniel R Licata, Michael Shulman, and Mitchell Riley. A fibrational framework for substructural and modal logics (extended version). 2017.

[11] Lucius Gregory Meredith. Linear types can change the blockchain. *arXiv preprint arXiv:1506.01001*, 2015.

[12] Michael Shulman. Framed bicategories and monoidal fibrations. *Theory and Applications of Categories*, 20(18):650–738, 2008.

[13] Matthijs Vákár. Syntax and semantics of linear dependent types. *CoRR*, abs/1405.0033, 2014.

[14] Marek Zawadowski. Lax monoidal fibrations. *Models, Logics, and Higher-Dimensional Categories: A Tribute to the Work of Mihály Makkai (CRM Proceedings 53, 2011)*, pages 341–424, 2011.