UNIVERSITY OF CAMBRIDGE
Department of Computer Science and Technology

Univalent Semantics for Varying Evaluation Strategies

Beatrice Szilvasy

St Catharine's College

May 12, 2026

Submitted in partial fulfillment of the requirements for the
Computer Science Tripos, Part III

Total page count: 52
Main chapters (excluding front-matter, references and appendix): 39 pages (pp 1–39)
Main chapters word count: 8271
Methodology used to generate that word count: `texcount`

Declaration

I, Beatrice Szilvasy of St Catharine's College, being a candidate for Part III of the Computer Science Tripos, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose. In preparation of this report, I adhered to the Department of Computer Science and Technology AI Policy. [The project required the approved use of {insert name of technology} and such use is acknowledged in the text at the relevant sections.] [I am content for my report to be made available to the students and staff of the University.]

Signed:

Date:

Abstract

TODO : this is the original as in the proposal. I will revise this as the work gets wrapped up.

Call-by-value and call-by-name are dual evaluation strategies in programming languages; when these are mixed in an effectful language, function composition ceases to be associative, rendering ordinary categories unsuitable for modelling their denotational semantics. Suitable constructs such as duploids have been studied in previous work, but in a univalent setting it is desirable to work with "univalent" versions, where equivalences correspond to identity types. In this work we will study and formalise the suitable notions of univalence for these constructs using the Rocq UniMath library.

Acknowledgements

This project would not have been possible without the wonderful support of ....

Contents

1. HALF-POLISHED Introduction

This dissertation is concerned with the semantics of programming languages in which composition is non-associative. That is, languages in which valid programs

$$\texttt{let } y := (\texttt{let } x := e_1 \texttt{ in } e_2) \texttt{ in } e_3 \tag{1.1}$$

and

$$\texttt{let } x := e_1 \texttt{ in } (\texttt{let } y := e_2 \texttt{ in } e_3) \tag{1.2}$$

(e_3 does not reference x) may compute differently.[1] One way in which such programs may arise is in the presence of side effects with mixed evaluation order.

To illustrate the effect of evaluation order, consider the program:

$$\texttt{let } x := (\texttt{print("hello"); } 1) \texttt{ in } x + x \tag{1.3}$$

where evaluating $\texttt{print}(v)$ has the side-effect of outputting v to a console. There are two canonical evaluations of Program 1.3 depending on the choice of evaluation order for the "`let`" construct.

- Using *call-by-name* (CBN), evaluate the expression $\texttt{let } x := e_1 \texttt{ in } e_2$ by first substituting e_1 for x in e_2. Then evaluate the resulting expression $e_1[x := e_2]$. Using call-by-name, Program 1.3 outputs `hellohello` and returns 2.

- Using *call-by-value* (CBV), evaluate $\texttt{let } x := e_1 \texttt{ in } e_2$ by first evaluating e_1 to its value v. Then substitute v for x in e_2. Finally, evaluate the resulting expression $e_2[x := v]$. Using call-by-value, Program 1.3 outputs just `hello` but still returns 2.

So long as all `let` expressions in a language use the same choice of evaluation order out of call-by-value or call-by-name, composition will remain associative. Combining

[1] TODO: Perhaps Levy's "let e_1 be x in e_2" syntax would be better?

call-by-value and call-by-name in a single language exposes non-associative composition. Let us write

$$\texttt{let}\ x \overset{\mathrm{V}}{\coloneqq} e_1\ \texttt{in}\ e_2 \qquad \text{for call-by-value, and}$$
$$\texttt{let}\ x \overset{\mathrm{N}}{\coloneqq} e_1\ \texttt{in}\ e_2 \qquad \text{for call-by-name.}$$

Then consider, for an example of non-associativity, the program

$$\begin{aligned} &\texttt{let}\ y \overset{\mathrm{N}}{\coloneqq} (\texttt{let}\ x \overset{\mathrm{V}}{\coloneqq} (\texttt{print("hello"); }1) \\ &\qquad\qquad \texttt{in}\ (\texttt{print("world"); }x))\ \texttt{in}\ y+y \end{aligned} \tag{1.4}$$

of the left-associated form (Program 1.1), and

$$\begin{aligned} &\texttt{let}\ x \overset{\mathrm{V}}{\coloneqq} (\texttt{print("hello"); }1)\ \texttt{in} \\ &(\texttt{let}\ y \overset{\mathrm{N}}{\coloneqq} (\texttt{print("world"); }x)\ \texttt{in}\ y+y) \end{aligned} \tag{1.5}$$

its reassociation to the right (Program 1.2). Recall that call-by-name `let` substitutes the entire assigned expression into the body, while call-by-value `let` evaluates it first and then substitutes. Then we can see that evaluating Program 1.4 will output `helloworldhelloworld`, but Program 1.5 will output `helloworldworld`, and both will return 2. Thus we have two programs that differ only up to associativity, and yet compute differently.

It is typical to interpret the semantics of programming languages in *categories* (Definition 3.1), but this breaks down for modelling non-associative computation. Let us consider $\texttt{let}\ y \coloneqq (\texttt{let}\ x \coloneqq e_1\ \texttt{in}\ e_2)\ \texttt{in}\ e_3$ (Program 1.1) and $\texttt{let}\ x \coloneqq e_1\ \texttt{in}\ (\texttt{let}\ y \coloneqq e_2\ \texttt{in}\ e_3)$ (Program 1.2) to be interpreted in some category $\mathcal{C}$, and assign to each expression $x : A \vdash e : B$ some morphism $[\![e]\!] : [\![A]\!] \to [\![B]\!]$ in $\mathcal{C}$. The question of whether Program 1.1 and Program 1.2 compute differently then amounts, informally,[2] to the following putative equation:

$$\Big([\![e_1]\!] \mathbin{;} [\![e_2]\!] \Big) \mathbin{;} [\![e_3]\!] \overset{?}{=} [\![e_1]\!] \mathbin{;} \Big([\![e_2]\!] \mathbin{;} [\![e_3]\!] \Big). \tag{1.6}$$

However, this is trivialized by the associativity of composition (Definition 3.1.4.b) in any category $\mathcal{C}$. If we wish to take seriously the idea that equations of the form in Equation 1.6 do not necessarily hold, then we must generalize the notion of a category.

[2]This would normally require extra bookkeeping to keep multiple variables in scope.

In this dissertation, it will be Munch-Maccagnoni's *duploids* [21] that generalize and take the role of categories for modelling non-associative computation.

For what purpose should we mix evaluation orders? One motivation is that it allows subsuming the theories of both call-by-name and call-by-value into a single paradigm such as *call-by-push-value* [15] (CBPV). The denotational semantics of call-by-push-value can be modelled "indirectly" by adjunctions [14], generalizing the widespread use of monads for the semantics of call-by-value [19]. [TODO: sing more praises for Levy and CBPV here because everything good about CBPV also applies for the related semantics in duploids.] Duploids fit into the picture by providing "direct" denotational semantics for mixed evaluation order, and have a correspondence with the aforementioned adjunction models [21]. [TODO: more motivation can't hurt?]

The purpose of this dissertation is to develop the theory of duploids formally in (a variant of) Martin Löf type theory and from a *univalent point of view* [24] (see Section 2.2). [TODO: explain UF, briefly? I don't want to dwell on this too long in the introduction because it needs too much background, which is what the next chapter is for.] We[3] formalize some existing results about duploids within the Rocq library UniMath [9]. Some of this work involves transferring existing theorems over to a more recent definition of duploids (see Remark 4.4). [NOTE: would be really cool to prove the full "structure theorem" for single-sorted duploids, since that does not yet exist in the literature.] We define what it means for a duploid $\mathcal{D}$ to be *univalent*: that isomorphisms $a \cong b$ in $\mathcal{D}$ (of a suitable kind) render the related objects $a, b : \mathcal{D}$ indistinguishable to the type theory. We show that the original construction of the duploid arising from an adjunction fails to be univalent, and then provide a novel construction of the same duploid which **is** univalent.

1.1. NOTES Outline

Rendered: May 12, 2026

These are my running notes about the structure of this text. Chapters are marked (in the TOC and their heading) with their completion status:

- NOTES means (rendered) comments from me to the reader of the draft (also typically me). These are also scattered through the draft as TODO, FIXME and NOTE tags, so that the reader knows which parts are incomplete and how.

[3] Me, my notebooks, and my proverbial rubber duck.

- TODO means that the section has not been written, or at least is not yet structured in a way that is actually coherent. Paragraphs may need to be rewritten or moved to different sections.
- VERY-ROUGH and ROUGH mean that some content exists and is intended to be legible, but content may be lacking and some things might not be coherent.
- HALF-POLISHED means the section is intended to be comprehensible as-is to the target audience, although is intended to undergo further refinement.
- "DONE" may happen in the future.

2. ROUGH Background

2.1. ROUGH Mixed Evaluation Order

It is a well-established tradition in theoretical computer science to consider programs as terms in variants of the lambda calculus. Although Church originally studied a *pure*, side-effect free, lambda calculus for the purposes of logic [3], many languages of practical use include *computational effects*, or simply *effects*, such as input/output (IO), mutable state, or non-termination.

A simply typed lambda calculus term $\Gamma \vdash e : A$ may be interpreted in any cartesian closed category as a morphism $[\![e]\!] : [\![\Gamma]\!] \to [\![A]\!]$ [13]. This interpretation is sound (and complete), in that if two terms $\Gamma \vdash e_1, e_2 : A$ are $\beta\eta$-equivalent, then their interpretations $[\![e_1]\!], [\![e_2]\!] : [\![\Gamma]\!] \to [\![A]\!]$ are equal in all cartesian closed categories (and vice-versa).

Pointing out that $\beta\eta$-equivalence is the wrong notion of program equivalence in the presence of computational effects, Moggi [19, 20] suggested the use of monads to model the semantics of effectful call-by-value lambda calculi instead. This way, given a suitable monad

$$(\mathcal{C} \xrightarrow{T} \mathcal{C}, \quad \mathsf{Id}_{\mathcal{C}} \overset{\eta}{\Rightarrow} T, \quad T^2 \overset{\mu}{\Rightarrow} T)$$

on a category $\mathcal{C}$, a compuational lambda term $\Gamma \vdash e : A$ can be interpreted as a morphism $[\![e]\!] : [\![\Gamma]\!] \to T\,[\![A]\!]$ in $\mathcal{C}$. The objects of $a : \mathcal{C}$ are the types of **values**, while objects $Ta : \mathcal{C}$ are the types of **computations** producing values of the type $a : \mathcal{C}$.

Moggi's interpretation of the computational lambda calculus involves a sort of "compilation" of lambda terms into a monadic style, inserting extra "bookkeeping code," such as units and multiplications, that were not present in the source. In the language of Führmann [7], monads constitute an *indirect semantics* for the call-by-value computational lambda calculus, in contrast to the *direct semantics* provided by, say, cartesian closed categories for the pure lambda calculus. The difference being that the calculus's syntax is much more closely related to the constructs readily available in a direct semantics, whereas an indirect semantics requires an extra transformation step.

Führmann [7] provides a direct semantics for the computational lambda calculus which he calls *abstract Kleisli categories* but which we shall call *thunk-force categories* [21].

Dually, it is possible to use comonads to model the semantics of call-by-name lambda calculi [14, 21]. This way, given a suitable comonad $U \colon \mathcal{D} \to \mathcal{D}$ on a category $\mathcal{D}$, we can interpret effectful call-by-name lambda terms $\Gamma \vdash e : A$ as morphisms $[\![e]\!] : U\,[\![\Gamma]\!] \to [\![A]\!]$. The dualized analogue of a thunk-force category is a *runnable monad* [21]. TODO: the types $a : \mathcal{D}$ are the stacks, but what are the objects $Ua : \mathcal{D}$?

The use of categories for the semantics of programming languages forces the choice of a fixed evaluation order. Otherwise, as seen in the introduction, composition ceases to be associative, and the use of categories becomes impossible.

This is undesirable for a number of reasons. Firstly, as argued by Levy [15], the combination of call-by-value and call-by-name into a single subsuming paradigm, call-by-push-value, obviates the need to reason about call-by-value and call-by-name separately. This reduces the burden of reasoning about models for both paradigms to reasoning about only one. Call-by-push-value has an indirect semantics in certain adjunctions [14]. Both call-by-value and call-by-name have reasonable translations into call-by-push-value in a way that preserves their equational theory and semantics.

Another motivation is advanced in Munch-Maccagnoni's thesis [22]. The relaxation of associativity captures meaningful properties about the types and programs, namely polarization. Guillaume proposes a direct semantics for mixed evaluation order called *duploids* [21]. Duploids subsume the direct semantics of thunk-force categories and runnable monads, and they also subsume the use of monads and comonads for call-by-value and call-by-name. Specifically, each adjunction $L \dashv R \colon \mathcal{P} \to \mathcal{N}$ gives rise to a duploid $\mathcal{E}[L \dashv R]$ which contains as subcategories the Kleisli category of $RL \colon \mathcal{P} \to \mathcal{P}$, and the co-Kleisli category of $LR \colon \mathcal{N} \to \mathcal{N}$.

The original study of duploids was done on what I call *two-sorted duploids*. These have two separate sorts of object: one sort of **positive** objects, or **values**; and one sort of **negative** objects, or **stacks**. In this work we shall study *single-sorted duploids* instead. In this case, positive and negative objects are distinguished not syntactically, but *semantically* by an associativity condition. [TODO: I could do well being more specific about these things.]

Although the background and context of this work lies in the semantics of programming languages, programming languages themselves will play little role beyond this section. Instead, we focus our attention entirely on the mathematical structures themselves.

2.2. ROUGH Preliminary Type Theory

This dissertation consists of an English language discussion about a body of mathematics that takes place in the formal language of homotopy type theory. The theorems, definitions, and proofs are presented in ordinary mathematical English, but they are also assigned meaning as types and terms in the language of homotopy type theory. In this section I explain some notation and terminology I will use throughout.

Some knowledge of homotopy type theory and univalent foundations is recommended for readers, as it is necessary to understand the motivations and main results. A sufficient introduction to univalent mathematics as a whole has been written by Grayson [8]. For textbook accounts, I suggest Rijke's *Introduction to Homotopy Type Theory* [23], or the well-known *HoTT Book* [24]. Though insufficient for understanding this dissertation, Escardó has a concise self-contained introduction to the univalence axiom itself [6]. Knowledge of basic dependent type theory is assumed from here on out.

As documented in the UniMath [9] repository, we work in Martin-Löf type theory [18] with intensional identity types and no general W-types. Write $\prod_{x:B} E[x]$ for dependent products, and $\sum_{x:B} E[x]$ and for dependent sums; $A \times B$ for binary products, and $A \amalg B$ for binary sums; $A \to B$ for non-dependent function types.

Identification Types

Perhaps the most important aspect of univalent foundations is its treatment of equality between two two terms $a, b : A$ or types A, B type. There are two distinct notions of equality in Martin-Löf type theory.

The first notion is *judgemental equality*, written $A \equiv B$ type for types, $a \equiv b : A$ for terms, or simply $a \equiv b$ for either. Judgemental equality is symmetric, reflexive, transitive. It is also respected by all judgements: assuming judgemental equality $a \equiv b$, the type or term a can be substituted throughout for b during type checking. Judgemental equality includes computation ("beta") rules: for example $(\lambda x : B.\, a)\ b \equiv a[x := b] : E[b]$ for $b : B$ and $x : B \vdash a : E[x]$; and uniqueness ("eta") rules: for example $f \equiv (\lambda x : B.\, f\ x) : \prod_{x:B} E[x]$.

Judgemental equality also includes definitions: $\mathsf{xyz} \overset{\text{def}}{\equiv} a : A$ implies $\mathsf{xyz} \equiv a : A$. Using the computation and uniqueness rules, we may write definitions in an implicit form, so that a need not be given explicitly. For example, the function

$$\mathsf{curry} \colon (\mathbb{N} \times \mathbb{N} \to \mathbb{N}) \to \mathbb{N} \to \mathbb{N} \to \mathbb{N}$$

may be defined equivalently either in "closed form" as

$$\mathsf{curry} \overset{\text{def}}{\equiv} \lambda f\ x\ y.\ f\ \langle x, y\rangle\,, \tag{2.1}$$

or "implicitly" as

$$\mathsf{curry}\ f\ x\ y \overset{\text{def}}{\equiv} f\ \langle x, y\rangle\,. \tag{2.2}$$

The judgemental equation 2.2 follows from Equation 2.1 by computation rules, and the uniqueness of Equation 2.1 as defining Equation 2.2 follows by the uniqueness rules. It is left for the reader to infer the closed form of any implicit definition.

Judgemental equality is a **judgement**, not a type. Due to this, theorems in expressed in univalent foundations may not talk about judgemental equality. Instead, Martin-Löf type theory includes a second notion of equality: the *identification type*. For any type A and terms $a, b : A$, the identification type is written $a =_A b$ or simply $a = b$. A term $p : a =_A b$ of identification type is sufficient to render a and b indistinguishable within the type theory using *identification induction* as discussed below.

The identification type comes equipped with a constructor $\mathsf{refl}_a : a =_A a$. Judgemental equality implies identification, in that $a \equiv b : A$ gives $\mathsf{refl}_a : a =_A b$. The converse, $p : a =_A b$ implying $a \equiv b : A$, is called *equality reflection*, and we do not allow it. Its absence is the marking characteristic of "intensional" dependent type theory.

Instead of equality reflection, the identification type permits *identification induction*. To define a function $f : \prod_{x,y:A} \prod_{z:x=_A y} E[x, y, z]$, we need only define the case $f\ x\ x\ \mathsf{refl}_x : E[x, x, \mathsf{refl}_x]$. This is justified by the eliminator $\mathsf{J}_{x,y,z.E[x,y,z]}$ with term formation rule

$$\frac{p : a =_A b \qquad r : \prod_{x:A} E[x, x, \mathsf{refl}_x]}{\mathsf{J}_{x,y,z.E[x,y,z]}(p, r) : E[a, b, p]} \tag{2.3}$$

and computation rule

$$\frac{r : \prod_{x:A} E[x, x, \mathsf{refl}_x]}{\mathsf{J}_{x,y,z.E[x,y,z]}(\mathsf{refl}_a, r) \equiv r\ a : E[a, a, \mathsf{refl}_a]}\,. \tag{2.4}$$

When identification induction is used, it is left for the reader to reconstruct the closed-form term using J. More typically, given $p : a =_A b$ we will informally use substitution to replace occurrences of a with b or vice-versa, particularly when the specific p is not relevant.

Truncation Levels

In the logic of some foundations such as ZFC, the concepts of "propositions" and "sets" are distinct. The latter constitutes "substance" whereas the former describes it. By contrast, in univalent foundations it is the type that takes the role of both substance and its description. TODO: finish this paragraph?

The fundamental behaviour of a proposition over "substance" is that it is logically irrelevant. This is the concept known as *proof irrelevance*: though two proofs of a proposition may be textually different to the human reader, the logic itself is unable to distinguish one from the other. In univalent foundations, we **define** propositions to be those types which satisfy proof irrelevance.

Definition 2.1. A type A is a *proposition* (or *mere proposition* for emphasis) when for all elements $a, b : A$ there is a chosen identification $a =_A b$.

Being a proposition is the second-lowest class in a hierarchical classification of types called *truncation levels*. Truncation levels describe how trivial the type of identifications for a given type is. The first truncation level is the *contractible types*, the second truncation level is propositions, the third is *sets*, and above that we have *n-groupoids* for $n \geq 1$.

Definition 2.2. A type A is *contractible* when there is a chosen element $\mathsf{pt} : A$ and for all elements $a : A$ there is a chosen identification $\mathsf{pt} =_A a$.

Definition 2.3. A type A is a *set*[1] when $a =_A b$ is a proposition for all $a, b : A$; a *1-groupoid* when $a =_A b$ is a set, and generally a *$(k+1)$-groupoid* when $a =_A b$ is a k-groupoid.

Example 2.4. The empty type `empty` is a proposition, and the unit type `unit` is contractible. The booleans `bool` and naturals $\mathbb{N}$ are both sets but not propositions. All propositions are also sets, all sets are also 1-groupoids, and all k-groupoids are also $(k+1)$-groupoids.

Write $\mathtt{iscontr}(A)$, $\mathtt{isaprop}(A)$ and $\mathtt{isaset}(A)$ for the types of A respectively being contractible, a proposition, and a set. Assuming function extensionality, it can be proven that these are propositions.

As a special case, when A is a dependent sum $A \equiv \sum_{x:B} E[x]$ we shall write $\exists! x : B.\, E[x]$ for $\mathtt{iscontr}(\sum_{x:B} E[x])$. Using this notation, given a function $f\colon A \to B$,

[1] This is orthogonal to matters of size.

we write $\mathtt{isweq}(f) \overset{\text{def}}{\equiv} \prod_{y:B} \exists! x : A.\, f\ x =_B y$ for the type of f being an *equivalence*. Write $A \simeq B$ for the type of equivalences, that is $A \simeq B \overset{\text{def}}{\equiv} \sum_{f:\, A \to B} \mathtt{isweq}(f)$.

Sometimes the data of a type (which need not be a proposition) A is not so important, only whether it is inhabited. Or to use set-theoretic terminology, whether such an element *exists*. The *propositional truncation* of A, written $\|A\|$, is the proposition which captures this idea. It comes with constructors

$$\frac{e : A}{|e| : \|A\|} \qquad \frac{e_1 : A \qquad e_2 : A}{|e_1| =_{\|A\|} |e_2|}, \tag{2.5}$$

and for any proposition P an eliminator

$$\frac{h : \|A\| \qquad f \colon A \to P}{\mathsf{u}_P(h, f) : P} \quad \text{satisfying} \quad \frac{t : A \qquad f \colon A \to P}{\mathsf{u}_P(|t|, f) \equiv f\ t : P}. \tag{2.6}$$

When A is a dependent sum $A \equiv \sum_{x:B} E[x]$, we shall write $\exists x : B.\, E[x]$ for $\left\| \sum_{x:B} E[x] \right\|$.

Universes

A *universe* $(\mathcal{U}, \mathsf{El})$ is a type $\mathcal{U}$ of *codes* along with, for each element $A : \mathcal{U}$, a decoding $\mathsf{El}(A)$ type. We shall abuse notation and use simply $\mathcal{U}$ to refer to the universe. For a universe $(\mathcal{U}, \mathsf{El})$, we say that a type A is *$\mathcal{U}$-small* if there is an element $\mathsf{code}(A) : \mathcal{U}$ such that $\mathsf{El}(\mathsf{code}(A)) \equiv A$ type.[2] By contrast, a type is *large* with respect to some universes $\mathcal{U}, \mathcal{U}', \ldots$ when it is not small in any of them. We shall typically omit El and code, so that $A : \mathcal{U}$ implies A type, and vice-versa when A is $\mathcal{U}$-small.

Convention 2.5. We shall often simply say that a type is "small" or "large" when the universes it is small or large with respect to are obvious or have not given a name. This rule will also apply to other things that may be "small" or "large."

We will typically assume, unless otherwise mentioned, that a universe $\mathcal{U}$ is closed under all of the type constructors mentioned so far (products, sums, $\mathtt{unit}$, etc.). This holds of most of the universes we shall consider.

Example 2.6. Given a universe $(\mathcal{U}, \mathsf{El})$, there is a universe $(\mathtt{hSet}_{\mathcal{U}}, \mathsf{El}')$ of $\mathcal{U}$-small sets. The codes of $\mathtt{hSet}_{\mathcal{U}}$ are the pairs $\sum_{A:\mathcal{U}} \mathtt{isaset}(A)$, and the decoding is given by $\mathsf{El}'\langle A, h_A \rangle \overset{\text{def}}{\equiv} \mathsf{El}(A)$. Likewise, $\mathtt{hProp}_{\mathcal{U}}$ is the universe consisting of $\mathcal{U}$-small propositions

[2] TODO: should this be only an equivalence?

$\sum_{A:\mathcal{U}} \texttt{isaprop}(A)$, defined in much the same way. The universe $\texttt{hProp}_{\mathcal{U}}$ is closed under dependent products, binary products, dependent sums, and identifications, but **not** binary sums. It also does not include `bool` or $\mathbb{N}$.

A *univalent universe* is a universe $\mathcal{U}$ where the canonical map

$$\begin{aligned} &\texttt{id\_to\_weq} : \prod_{A,B:\mathcal{U}} (A =_{\mathcal{U}} B) \to (A \simeq B) \\ &\texttt{id\_to\_weq}\ A\ A\ \mathsf{refl}_A \stackrel{\text{def}}{\equiv} \mathsf{id}_A \end{aligned} \tag{2.7}$$

given by identification induction is an equivalence for all types $A, B : \mathcal{U}$. For a given universe $\mathcal{U}$, the *univalence axiom* states that $\mathcal{U}$ is univalent. In short, the univalence axiom states that the identifications $A =_{\mathcal{U}} B$ between two types are characterized by the equivalences $A \simeq B$ between those types. These, in turn, depend on the structure of the identifications within A and B themselves.

Example 2.7. If a universe $\mathcal{U}$ is univalent, then so too is $\texttt{hSet}_{\mathcal{U}}$ and $\texttt{hProp}_{\mathcal{U}}$. The converse need not be true in general.

The type of equivalences between types, say $\texttt{bool} \simeq \texttt{bool}$, is not in general a proposition. Thus, the univalence axiom notably provides a way of constructing identifications which are provably **not** identifiable with refl. This property is perfectly consistent with J [10]; indeed there are models including the univalence axiom that are as consistent as ZFC with some strongly inaccessible cardinals [11, 12].

We will at times the existence of a tower of universes $\mathcal{U}_0, \mathcal{U}_1, \ldots$. These are such that each $\mathcal{U}_k$ is $\mathcal{U}_{k+1}$-small. Moreover, we assume the universes are cumulative, if A is $\mathcal{U}_k$-small, then A is $\mathcal{U}_{k'}$-small for any $k' \geq k$. We will need only the first three universes[3]. We shall explicitly mention the univalence axiom when assumed. UniMath further assumes a propositional resizing axiom, but we will not be using it ourselves.[4, 5]

Classical Mathematics and Logic

By contrast to "univalent foundations," I use the term "set-based foundations" to include both ZF(C) and type theories where all types are sets (in the sense above).

[3] TODO: I think? Assuming all our duploids are $\mathcal{U}_0$-small and therefore live in $\mathcal{U}_1$, the largest type we use is the ($\mathcal{U}_2$-small) reflexive graph of $\mathcal{U}_1$-small reflexive graphs.

[4] For technical and historical reasons, UniMath asks Rocq not to check universes, and so has $\mathcal{U} : \mathcal{U}$. As such, checking universe levels is the responsibility of us humans.

[5] UniMath uses propositional resizing to define propositional truncation.

Many theorems and definitions of set-based foundations may be recovered in univalent foundations by assuming certain types to be sets, such as by reasoning about $\mathtt{hSet}_{\mathcal{U}}$ instead of $\mathcal{U}$ for a fixed universe $\mathcal{U}$. The main expressive power lost by moving from set-based foundations to univalent foundations is the ability to express statements that are invariant under equivalence (of types, categories, or any other mathematical object).

It is an occasional misconception is that univalent foundations is incompatible with non-constructive reasoning such as the *law of excluded middle* (`LEM`) or the *axiom of choice* (`AxiomOfChoice`). However, these are perfectly admissible so long as they are stated correctly in terms of sets and propositions. They are also often rendered unnecessary by univalence. We shall use the law of excluded middle on one such occasion.

Further Conventions

Table 2.1 describes English language examples and their type-theoretic interpretation. In particular, I use a propositionally truncated interpretation of "there exists" and "or;" I use "there is" and "either … or" for the constructive variants. In addition to the informal English language interpretations, the final column suggests homotopical interpretations. These may be realized more formally by models of univalent foundations such as Voevodsky's in simplicial sets [11, 12].

Convention 2.8. I use upper case for types (A, B, C, E, P), calligraphic font for categories and the like $(\mathcal{C}, \mathcal{D}, \mathcal{N}, \mathcal{P})$, lower case for objects and morphisms within categories $(a, b : \mathcal{C},\ f, g \colon a \to b)$, upper case for functors $(F, G, H \colon \mathcal{C} \to \mathcal{D})$, and Greek letters for natural transformations $(\alpha, \beta \colon F \Rightarrow G)$.

Notation 2.9. Where a type contains a binary operator such as $\to$, $=$, or $\simeq$, we may use the infix notation $a \xrightarrow{f} b$ to mean $f \colon a \to b$. Sharing expressions in such infix notation, as in $t_1 \overset{p}{=} t_2 \overset{q}{=} t_3$, denotes the independent judgements $p : t_1 = t_2$ and $q : t_2 = t_3$. It may additionally denote the appropriate composition of p and q of the type $t_1 = t_3$ too, depending on the context.

Convention 2.10. A word or phrase written in *italics* is a technical term introduced at that point, and may be found in the index. When a theorem or definition has been formalized in UniMath, we shall write the name of its definition in `monospace` font, hyperlinked to the file and line number in the UniMath repository. An example is "a *category* (`category`) is …" In anonymized versions, these links will necessarily be

Table 2.1.: Example interpretations of mathematical English. $E[x]$ denotes a type indexed by x, and P denotes a proposition.

Mathematical English	Homotopy Type Theory	Homotopy Theory
Utterance	Judgement	Interpretation
description of a ***type*** *as below*	A type	A is a homotopy type
desc. of a ***type*** *mentioning* $x : B$	$x : B \vdash E[x]$ type	E is a fibre bundle over B
description of a ***term***	t or $t : A$	$t \in A$
$t_1 : A$ is $t_2 : A$	$t_1 \equiv t_2 : A$	$t_1 = t_2$
let x be an A	$\dfrac{x : A}{\dots}$	let $x \in A$
assume P	$\dfrac{h : P}{\dots}$	assume P
Definition *xyz* is $t : A$.	$\mathsf{xyz} \overset{\text{def}}{\equiv} t : A$	$\mathsf{xyz} \overset{\text{def}}{=} t$
Theorem P. *Proof.* t. $\square$	$t : P$	$t \in P$
Description	Type	Space
[pair/triple/... of] $x : B$, $E[x]$	$\sum_{x:B} E[x]$	the total space E
$x : B$ such that $P[x]$	$\sum_{x:B} P[x]$	subspace $\{ x \in B \mid P[x] \text{ inhabited} \}$
function A to B	$A \to B$ or $\prod_{x:A} B$	function space $A \to B$
for all $x : B$, $E[x]$	$\prod_{x:B} E[x]$ or $\forall x : B.\, E[x]$	sections of $E \to B$
A is a proposition	$\mathtt{isaprop}(A)$	A is contractible-if-inhabited
there exists a unique A	$\mathtt{iscontr}(A)$	A is contractible
there exists a unique $x : B$ such that $P[x]$	$\mathtt{iscontr}(\sum_{x:B} P[x])$ or $\exists! x : B.\, P[x]$	
there exists A	$\lVert A \rVert$ (propositional truncation)	A is inhabited
there exists $x : B$ such that $P[x]$	$\left\lVert \sum_{x:B} P[x] \right\rVert$ or $\exists x : B.\, P[x]$	
A is a set	$\mathtt{isaset}(A)$	A is a homotopy 0-type
$t_1 : A$ and $t_2 : A$ are identified	$t_1 =_A t_2$	path space t_1 to t_2
(if A is a set) $t_1 : A$ and $t_2 : A$ are equal	$t_1 =_A t_2$	path space t_1 to t_2
either A or B	$A \amalg B$	disjoint union of A and B
A or B	$A \vee B$ or $\lVert A \amalg B \rVert$	
$f : A \to B$ is an equivalence	$\mathtt{isweq}(f)$ or $\prod_{y:B} \exists! x : A.\, f(x) =_B y$	f is a homotopy equivalence
A and B are equivalent	$A \simeq B$ or $\sum_{f:\,A \to B} \mathtt{isweq}(f)$	A and B are homotopy equivalent

broken.

2.3. ROUGH Univalent Categories

Beyond the foundation of mathematics in which we work, the content of this dissertation is about non-associative generalizations of categories. Namely, Munch-Maccagnoni's *duploids* [21]. Before we can talk about univalent duploids, we ought understand categories in the context of univalent foundations.[6]

The authoritative paper on categories in univalent foundations is by Ahrens et al. [1], whose formalisation in Rocq is today part of UniMath. An expanded version of that paper is included as Chapter 9 of the *HoTT Book* [24]. The development has since diverged from the paper: we use the names "`category`" and "`univalent_category`" for what used to be "precategory" and "category" respectively.

Intuition

It is well-known that in a category $\mathcal{C}$, the correct notion of "sameness" between two objects $a, b : \mathcal{C}$ is *isomorphism* $a \cong b$, and not (necessarily) the foundations' notion of "equality" $a = b$. In set-based foundations, equality is overly restrictive: there may be objects $a \cong b$ that are not equal,[7] and there may be isomorphisms $a \cong a$ that are not the identity.[8] In univalent foundations, the type of identifications $a =_{\mathsf{ob}\,\mathcal{C}} b$ may very well correspond to isomorphisms $a \cong b$. A *univalent category* is one where they do.

An informative use-case of univalence is in relation to weak equivalences between categories. It is a theorem that for a univalent category $\mathcal{C}$ and a category $\mathcal{D}$, a fully-faithful and essentially surjective functor $F : \mathcal{C} \to \mathcal{D}$ is an adjoint equivalence. The analogous statement in set-based foundations amounts to the axiom of choice. More generally, objects determined by a universal property in a univalent category $\mathcal{C}$ become literally unique, as do left- and right- adjoints of a fixed functor $F : \mathcal{C} \to \mathcal{D}$ to another category $\mathcal{D}$. In other words, universal properties become mere properties and, when they hold, contractible. Thus, univalence has the practical benefit of being able to talk about *the* object (functor, category, etc.) satisfying a universal property, with the representative being canonical and choice-free.[9]

[6] TODO: This paragraph is slightly out of place now that I have moved things around.
[7] Say, the sets $\{0, 1\}$ and $\{1, 2\}$.
[8] Say, the set $\mathbb{B}$ under boolean negation.
[9] TODO: How much does this even matter lol

When the univalence axiom holds, many naturally-arising categories are univalent. For example, for a univalent universe $\mathcal{U}$, the category $\mathbf{Set}_{\mathcal{U}}$ (or simply $\mathbf{Set}$) of $\mathcal{U}$-small sets $A, B : \mathtt{hSet}_{\mathcal{U}}$ and functions $A \to B$ is univalent (`HSET_univalent_category`). As an extension, it was shown by Coquand et al. [5] that the univalence axiom extends to a "large class of algebraic structures." This implies that categories of set-based algebraic structures such as monoids and rings are univalent too.

Every category $\mathcal{C}$ is weakly equivalent to a univalent category $\mathfrak{R}\mathcal{C}$. This is in the sense that there is a fully-faithful and essentially surjective functor $\mathfrak{r}_{\mathcal{C}} : \mathcal{C} \to \mathfrak{R}\mathcal{C}$. We call the category $\mathfrak{R}\mathcal{C}$ the *Rezk completion* of $\mathcal{C}$. One way to construct the Rezk completion is as the image of the Yoneda embedding $\mathsf{y} : \mathcal{C} \to [\mathcal{C}^{\mathsf{op}}, \mathbf{Set}_{\mathcal{U}}]$ [1].

The univalence of a category is sensitive to how the category is constructed. A relevant example is the *Kleisli category* of a monad

$$(\mathcal{C} \xrightarrow{T} \mathcal{C}, \quad \mathsf{Id}_{\mathcal{C}} \overset{\eta}{\Rightarrow} T, \quad T^2 \overset{\mu}{\Rightarrow} T)$$

on a category $\mathcal{C}$. One way to construct the Kleisli category of T is as follows. The objects are those of $\mathcal{C}$, and, for objects $a, b : \mathcal{C}$, the morphisms are the Kleisli arrows $a \to Tb$ in $\mathcal{C}$. Identities and composition are through η and μ respectively. Even when $\mathcal{C}$ is univalent, this construction of the Kleisli category is **not** generally univalent (see Example 2.11). Let us therefore call this non-univalent Kleisli category $\mathbf{Kleisli}_{\cancel{\mathsf{univ}}}(\mathcal{C}, T)$. There is a univalent construction of the Kleisli category, however, as the subcategory of free T-algebras in the Eilenberg-Moore category of T (`univalent_kleisli_cat`). We shall call this category $\mathbf{Kleisli}_{\mathsf{univ}}(\mathcal{C}, T)$, or simply $\mathbf{Kleisli}(\mathcal{C}, T)$. Explicitly, this amounts to considering a suitable subcategory of $\mathcal{C}$ whose objects are of the form Ta for some $a : \mathcal{C}$.

Example 2.11 (Levy [16])**.** Consider the state monad $Ta \overset{\text{def}}{\equiv} (\mathbb{N} \Rightarrow a \times \mathbb{N})$ on $\mathbf{Set}$. Then the Kleisli arrows $a \to Tb$ of T correspond to set-functions $a \times \mathbb{N} \to b \times \mathbb{N}$. From this it is easy to see that there is an isomorphism $\mathtt{unit} \cong \mathtt{bool}$ in $\mathbf{Kleisli}_{\cancel{\mathsf{univ}}}(\mathbf{Set}, T)$: this is simply an isomorphism $\mathtt{unit} \times \mathbb{N} \cong \mathtt{bool} \times \mathbb{N}$ in $\mathbf{Set}$. However, the type `unit` is certainly not identified with `bool`!

Definitions

TODO: I might move these to the unital magmoids section and define categories/functors/natural transformations in parallel with the respective things for unital magmoids?

Example 2.12. Let $\mathcal{C}$ and $\mathcal{D}$ be categories with functors $F, G \colon \mathcal{C} \to \mathcal{D}$ between them. Then the type of natural transformations $F \Rightarrow G$ is a set because $\mathcal{D}$ has hom-sets. Thus, there is a category $[\mathcal{C}, \mathcal{D}]$ (`functor_category`) whose objects are functors $F, G \colon \mathcal{C} \to \mathcal{D}$ and whose morphisms are natural transformations $\alpha \colon F \Rightarrow G$. If $\mathcal{D}$ is univalent, then the functor category $[\mathcal{C}, \mathcal{D}]$ is univalent. That is, the identifications $F = G$ between two functors $F, G \colon \mathcal{C} \to \mathcal{D}$ correspond to the natural isomorphisms $F \cong G$ between them (`is_univalent_functor_category`).

Non-example 2.13. Let $\mathcal{U}$ be a universe. There is in general no category $\mathbf{Cat}_{\mathcal{U}}$ of $\mathcal{U}$-small categories and functors between them. This is because, in general, the functors $\mathcal{C} \to \mathcal{D}$ between categories $\mathcal{C}$ and $\mathcal{D}$ do not form a set.[10] One must instead consider $\mathbf{Cat}_{\mathcal{U}}$ as a bicategory. Another approach, more reminiscent of set-based foundations, is to restrict ourselves to *strict categories*, or $\mathtt{hSet}_{\mathcal{U}}$-small categories when $\mathcal{U}$-small. These are categories $\mathcal{C}$ whose objects $\mathsf{ob}\,\mathcal{C}$ form a set. Then we may perfectly well consider the category $\mathbf{Cat}_{\mathtt{hSet}_{\mathcal{U}}}$ of strict $\mathcal{U}$-small categories.

TODO: narrative to bring in Rezk completion?

Definition 2.14. The *Rezk completion* of a category $\mathcal{C}$ is a univalent category $\mathfrak{R}\mathcal{C}$ and a functor $\mathfrak{r}_{\mathcal{C}} \colon \mathcal{C} \to \mathfrak{R}\mathcal{C}$ which is fully-faithful and essentially surjective on objects.

Remark 2.15. The Rezk completion of categories $\mathfrak{R}$ enjoys the following universal property as a consequence of $\mathfrak{r}_{\mathcal{C}}$ being fully-faithful and essentially surjective. Let $\mathcal{C}$ be a category, $\mathcal{D}$ be a univalent category, and $F \colon \mathcal{C} \to \mathcal{D}$ be a functor. Then there exists a unique functor $\hat{F} \colon \mathfrak{R}(\mathcal{C}) \to \mathcal{D}$ such that $\mathfrak{r}_{\mathcal{C}} \mathbin{⨾} \hat{F} = F$. Graphically, this is the following commutative diagram in the bicategory $\mathbf{Cat}$:[11]

$$\begin{tikzcd} \mathcal{C} \arrow[r, "\mathfrak{r}_{\mathcal{C}}"] \arrow[dr, "F"'] & \mathfrak{R}\mathcal{C} \arrow[d, dashed, "\exists! \hat{F}"] \\ & \mathcal{D}\,. \end{tikzcd}$$

Another way to say this is that $\mathfrak{R}$ extends to a pseudofunctor $\mathfrak{R} \colon \mathbf{Cat} \to \mathbf{Cat}_{\mathsf{univ}}$ which is the left-biadjoint of the forgetful pseudofunctor $U \colon \mathbf{Cat}_{\mathsf{univ}} \to \mathbf{Cat}$, witnessing $\mathbf{Cat}_{\mathsf{univ}}$ as a reflective sub-bicategory of $\mathbf{Cat}$.

[10] Consider the two natural isomorphisms from $\mathbf{1} \xrightarrow{\mathtt{bool}} \mathbf{Set}$ to itself, when $\mathbf{Set}$ is univalent.

[11] This $\mathbf{Cat}$ should be $\mathbf{Cat}_{\mathcal{U}}$ for a universe $\mathcal{U}$ in which both $\mathcal{C}$ and $\mathfrak{R}\mathcal{C}$ are small. We make no assumption here about the size of $\mathfrak{R}\mathcal{C}$ in relation to $\mathcal{C}$. (NOTE: In the UniMath construction via the Yoneda embedding, $\mathcal{C}$ is large even when $\mathcal{C}$ is small. There is a HIT variant too which is the same size. I don't know how much I care to include these references.)

3. ROUGH Unital Magmoids

3.1. Definitions

Definition 3.1 (`category` [1])**.** A *category* $\mathcal{C}$ consists of the following:

1. A type of *objects*, written $\mathsf{ob}\,\mathcal{C}$ or just $\mathcal{C}$.

2. For each pair of objects $a, b : \mathcal{C}$, a type of *morphisms* from a to b, also known as a *hom-type*, written $\mathcal{C}[\![a, b]\!]$ or $a \to b$.

3. For each object $a : \mathcal{C}$, a distinguished *identity* morphism $\mathsf{id}_a : \mathcal{C}[\![a, a]\!]$.

4. A *composition* operator $\mathbin{;}_{\mathcal{C}}$ or simply $\mathbin{;}$ assigning to each pair of morphisms $a \xrightarrow{f} b \xrightarrow{g} c$ a composite $a \xrightarrow{f \mathbin{;} g} c$ such that:

   a) Composition is *unital*: for $a \xrightarrow{f} b$ we have $\mathsf{id}_a \mathbin{;} f = f$ and $f \mathbin{;} \mathsf{id}_b = f$.

   b) Composition is *associative*: for $a \xrightarrow{f} b \xrightarrow{g} c \xrightarrow{h} d$ we have $f \mathbin{;} (g \mathbin{;} h) = (f \mathbin{;} g) \mathbin{;} h$.

5. $\mathcal{C}$ *has hom-sets*: each hom-type $\mathcal{C}[\![a, b]\!]$ is a set.

Most of Definition 3.1 will be familiar to a mathematician who has come across categories. Indeed, putting aside the hom-set law for now (Definition 3.1.5), one obtains a typical set-theoretic definition by substituting "class" for "type" in the definition of objects and morphisms. The hom-set law simply makes identitification $f =_{a \to b} g$ of morphisms $f, g \colon a \to b$ be a proposition. We shall therefore find it convenient to use the language of "equality" for identifications between morphisms.

Definition 3.2. A *unital magmoid* or *non-associative category* [17] (`unital_magmoid`) is a generalization of a category that need not be associative (Definition 3.1.4.b). More generally, a *reflexive graph*[1] (`rxgraph`) is like a category but omits composition (Definition 3.1.4) entirely and need not have hom-sets (Definition 3.1.5). These can

[1] Our "graph" is similar to what others may call a "quiver" or "directed multigraph."

further be generalized to *magmoids* and *graphs*[1] respectively by omitting identities (Definition 3.1.3, 3.1.4.a). These are summarized in Table 3.1.

Table 3.1.: Attributes of category-like structures.

Attribute	Structure				
	Category	Unital Magmoid	Magmoid	Reflexive Graph	Graph
Objects	✓	✓	✓	✓	✓
Morphisms	✓	✓	✓	✓	✓
Hom-sets	✓	✓	✓	✗	✗
Identities	✓	✓	✗	✓	✗
Composition	✓	✓	✓	✗	✗
Unital	✓	✓	N/A	N/A	N/A
Associative	✓	✗	✗	N/A	N/A

We import most definitions of structures in and on categories for (unital) magmoids and (reflexive) graphs directly, as much as they are well-defined. I spell out these definitions here in their entirety. The failure of associativity causes the imported notions to become weaker, however. We shall see this for isomorphisms (Definition 3.9) and natural transformations (Examples 3.16).

Definition 3.3. Let $\mathcal{U}$ be a universe. A graph (or category, unital magmoid, etc.) $\mathcal{G}$ is *locally $\mathcal{U}$-small* if for all $a, b : \mathcal{G}$ the hom-type $\mathcal{G}[\![a, b]\!]$ is $\mathcal{U}$-small. A graph $\mathcal{G}$ is *$\mathcal{U}$-small* when it is locally small and its type of objects $\mathsf{ob}\,\mathcal{G}$ is $\mathcal{U}$-small. A graph is *large* with respect to some universes $\mathcal{U}, \mathcal{U}', \ldots$ when it is not small in any of them. Following Convention 2.5, we shall often omit the universe when describing graphs as "small" or "large."

Example 3.4. For a category $\mathcal{C}$, its *opposite category* $\mathcal{C}^{\mathsf{op}}$ is the category with the same objects, and with the directions of all morphisms inverted:

$$\mathsf{ob}\,\mathcal{C}^{\mathsf{op}} \overset{\text{def}}{\equiv} \mathsf{ob}\,\mathcal{C}$$
$$\mathcal{C}^{\mathsf{op}}[\![a, b]\!] \overset{\text{def}}{\equiv} \mathcal{C}[\![b, a]\!]\,.$$

The identity id_a for $a : \mathcal{C}^{\mathsf{op}}$ is that of $\mathcal{C}$, and composition of two morphisms $f : \mathcal{C}^{\mathsf{op}}[\![a, b]\!]$ and $g : \mathcal{C}^{\mathsf{op}}[\![b, c]\!]$ is given by composition in $\mathcal{C}$:

$$f \mathbin{⨟}_{\mathcal{C}^{\mathsf{op}}} g \overset{\text{def}}{\equiv} g \mathbin{⨟}_{\mathcal{C}} f\,.$$

The category laws of $\mathcal{C}^{\mathsf{op}}$ all follow from those of $\mathcal{C}$.

Definition 3.5. The opposite construction straightforwardly generalizes to graphs, reflexive graphs, magmoids, and unital magmoids. Thus we may speak of the *opposite graph* $\mathcal{G}^{\mathsf{op}}$ of a graph $\mathcal{G}$, the *opposite unital magmoid* $\mathcal{M}^{\mathsf{op}}$ of a unital magmoid $\mathcal{M}$, etc.

Definition 3.6 (`is_z_isomorphism`, `z_iso`)**.** Let $\mathcal{C}$ be a category. A morphism $f\colon a \to b$ between objects $a, b : \mathcal{C}$ *has an inverse* if there is a morphism $f^{-1}\colon b \to a$ such that

$$f \mathbin{⨾} f^{-1} = \mathsf{id}_a \quad \text{and} \quad f^{-1} \mathbin{⨾} f = \mathsf{id}_b\,. \tag{3.1}$$

In this case, we say that f is an *isomorphism*, written $f : a \cong b$.

Remark 3.7. It is well known that inverses of morphisms are unique in a category. That is, if we have a morphism $f\colon a \to b$ in a category $\mathcal{C}$ and any two morphisms $p, q\colon b \to a$ satisfying Equation 3.1, then we have $p = q$. I spell out the proof, because it depends on associativity:

$$p = p \mathbin{⨾} (f \mathbin{⨾} q) = (p \mathbin{⨾} f) \mathbin{⨾} q = q\,.$$

In combination with the hom-set law, this makes the type "f has an inverse" be a proposition in a category (`isaprop_is_z_isomorphism`).

Definition 3.8 (`is_univalent` [1])**.** Let $\mathcal{C}$ be a category. Then there is a canonical map from identifications to isomorphisms

$$\begin{aligned} \mathtt{id\_to\_iso} &: \prod_{a,b:\mathsf{ob}\,\mathcal{C}} (a =_{\mathsf{ob}\,\mathcal{C}} b) \to (a \cong b) \\ \mathtt{id\_to\_iso}\; a\; a\; \mathsf{refl}_a &\overset{\text{def}}{\equiv} \mathsf{id}_a \end{aligned} \tag{3.2}$$

defined by identification induction (reminiscent of Equation 2.7). The category $\mathcal{C}$ is *univalent* if `id_to_iso` a b is an equivalence for all $a, b : \mathcal{C}$.

Definition 3.9. In a unital magmoid, we shall still use the term isomorphism and its notation $\cong$ to mean a morphism that has a specified inverse, which is no longer unique. These isomorphisms in unital magmoids are inferior to those in categories, and so we do not generalize the definition of "univalent category" to unital magmoids directly. Instead, we shall define a stronger notion of isomorphism and use that for the univalence condition of unital magmoids (see Definition 3.26), which simplifies to Definition 3.8 when the unital magmoid is a category.

Definition 3.10 (`functor`)**.** A *functor* (or *category functor* for emphasis) F between categories $\mathcal{C}$ and $\mathcal{D}$, written $\mathcal{C} \to \mathcal{D}$, consists of:

1. A mapping on objects $F_0 : \mathsf{ob}\,\mathcal{C} \to \mathsf{ob}\,\mathcal{D}$.

2. For all objects $a, b : \mathsf{ob}\,\mathcal{C}$, a mapping on morphisms $F_1 : \mathcal{C}[\![a, b]\!] \to \mathcal{C}[\![F_0\, a, F_0\, b]\!]$.

3. F preserves:

   a) Identities:
   $$F_1 \left(a \xrightarrow{\mathsf{id}_a} a \right) = F_0\ a \xrightarrow{\mathsf{id}_{F_0 a}} F_0\, a$$
   for all $a : \mathcal{C}$.

   b) Composition:
   $$F_1 \left(a \xrightarrow{f} b \xrightarrow{g} c \right) = F_0\, a \xrightarrow{F_1\, f} F_0\, b \xrightarrow{F_1\, g} F_0\, c$$
   for all $a \xrightarrow{f} b \xrightarrow{g} c$ in $\mathcal{C}$.

Both F_0 and F_1 may simply be written F.

Definition 3.11. Functors of (unital) magmoids and (reflexive) graphs consist of the same data as category functors, but omit the identity and composition laws where they are not well defined. Specifically, a *unital magmoid functor* is the same as a category functor, a *reflexive graph functor* (`rxfunctor`) need only preserve identities, a *magmoid functor* need only preserve composition, and a *graph functor* need preserve neither.

Notation 3.12. Between (reflexive) graphs $\mathcal{G}, \mathcal{H}$, we shall write $\mathcal{G} \rightharpoonup \mathcal{H}$ for graph functors in general, and $\mathcal{G} \dashrightarrow \mathcal{H}$ for reflexive graph functors. Between unital mamoids $\mathcal{M}, \mathcal{N}$, we write unital magmoid functors $\mathcal{M} \to \mathcal{N}$ the same as category functors.

Remark 3.13. The functor laws (Definition 3.10.3) for a graph functor $\mathcal{G} \rightharpoonup \mathcal{H}$ are mere properties when $\mathcal{H}$ has hom-sets (Definition 3.1.5). For example, when $\mathcal{H}$ is a unital magmoid.

Definition 3.14 (`nat_trans`)**.** A *natural transformation* α between category functors $F, G : \mathcal{C} \to \mathcal{D}$, written $\alpha : F \Rightarrow G$, consists of:

1. For each object $a : \mathcal{C}$, a morphism $\alpha_a : Fa \to Ga$.

2. *Naturality*: for all objects $a, b : \mathcal{C}$ and morphisms $f\colon a \to b$, the equation $\alpha_a \fatsemi Gf = Ff \fatsemi \alpha_b$ holds. That is, the *naturality square*

$$\begin{tikzcd} Fa \arrow[r, "Ff"] \arrow[d, "\alpha_a"'] & Fb \arrow[d, "\alpha_b"] \\ Ga \arrow[r, "Gf"] & Gb \end{tikzcd} \tag{3.3}$$

in $\mathcal{D}$ commutes.

Definition 3.15. Natural transformations $F \Rightarrow G$ are perfectly well defined for any sort of graph functors $F, G\colon \mathcal{G} \rightharpoonup \mathcal{M}$, so long as the codomain $\mathcal{M}$ of the functors is at least a magmoid. In the absence of naturality, or when composition is undefined, we call the family of morphisms an *unnatural transformation* and write $F \rightleftharpoons G$ for its type.

Examples 3.16. Let $\mathcal{G}$ be a graph, $\mathcal{M}$ a unital magmoid, and $F\colon \mathcal{G} \rightharpoonup \mathcal{M}$ a graph functor. There is a natural transformation $\mathsf{id}_F\colon F \Rightarrow F$ given componentwise by $(\mathsf{id}_F)_a \overset{\text{def}}{\equiv} \mathsf{id}_{Fa}$, where naturality follows by unitality (Definition 3.1.4.a). For graph functors $F, G, H\colon \mathcal{G} \rightharpoonup \mathcal{M}$ and natural transformations $\alpha\colon F \Rightarrow G$, and $\beta\colon G \Rightarrow H$, we may define their vertical composite $(\alpha \fatsemi \beta)_a \overset{\text{def}}{\equiv} \alpha_a \fatsemi \beta_a$. However, this is only an unnatural transformation $\alpha \fatsemi \beta\colon F \rightleftharpoons H$ in general, due to the non-associativity of $\mathcal{M}$. When $\mathcal{M}$ is a category, the vertical composite $\alpha \fatsemi \beta$ is natural.

3.2. Polarities

Many theorems about categories which fail to generalize to unital magmoids can be salvaged by hypothesizing associativity of certain composites.

Definition 3.17 (`is_thunkable`, `is_negative`, etc. [21])**.** We say that a path of morphisms

$$a_0 \xrightarrow{f_1} a_1 \xrightarrow{f_2} a_2 \to \cdots \to a_{n-1} \xrightarrow{f_n} a_n \tag{3.4}$$

in a unital magmoid *associates* when all ways of parenthesising their composite $f_1 \fatsemi f_2 \fatsemi \cdots \fatsemi f_n$ ($= \mathsf{id}_{a_0}$ if empty) are equal. In the simplest non-trivial case, a triple of morphisms $a \xrightarrow{f} b \xrightarrow{g} c \xrightarrow{h} d$ associates if and only if we have $f \fatsemi (g \fatsemi h) = (f \fatsemi g) \fatsemi h$.

We say that:

- f is *thunkable* when all triples (f, g, h) associate.

- h is *linear* when all triples (f, g, h) associate.
- g is *intermediate* when all triples (f, g, h) associate.
- b is (*semantically*) *negative* when all morphisms into b are thunkable.
- c is (*semantically*) *positive* when all morphisms out of c are linear.

This is depicted pictorially in Figure 3.1.

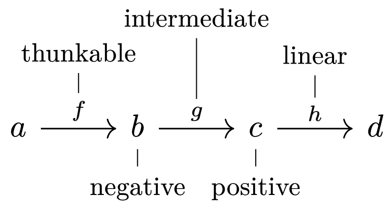

Figure 3.1.: The triple (f, g, h) associates whenever any of the labelled properties holds.

Notation 3.18. When a path of morphisms as in Diagram 3.4 is known to associate, we will occasionally write its composite without parentheses as $f_1 \fatsemi f_2 \fatsemi \cdots \fatsemi f_n$. Occasionally, following Notation 2.9, we will use the diagram $a_0 \xrightarrow{f_1} a_1 \xrightarrow{f_2} a_2 \to \cdots \to a_{n-1} \xrightarrow{f_n} a_n$ itself to denote the composite.

Remark 3.19. It is easy to see that thunkability/linearity and negativity/positivity are dual, while intermediateness is self-dual. This is in the usual sense that being thunkable in some unital magmoid $\mathcal{C}$ is the same as being thunkable in $\mathcal{C}^{\mathsf{op}}$, and being negative in $\mathcal{C}$ is the same as being positive in $\mathcal{C}^{\mathsf{op}}$, up to inversion of the identification $f \fatsemi (g \fatsemi h) = (f \fatsemi g) \fatsemi h$.

Example 3.20. In any unital magmoid, the identity morphisms are thunkable, linear, and intermediate by unitality (Definition 3.1.4.a). Composites of thunkable, linear, or intermediate morphisms are also thunkable, linear, or intermediate respectively.

Definition 3.21 (`linear_category`, `positive_category`, etc. [21])**.** Let $\mathcal{M}$ be a unital magmoid. The following are subcategories of $\mathcal{M}$; that is, sub- unital magmoids of $\mathcal{M}$ which are associative.

- The *linear subcategory* $\mathcal{M}_l$, restricting to linear morphisms.
- The *thunkable subcategory* $\mathcal{M}_t$, restricting to thunkable morphisms.

$$
\begin{tikzcd}
\mathcal{M}^{\oplus} \arrow[r, hook, two heads] & \mathcal{M}_l & \\
\mathcal{M}_t^{\oplus} \arrow[u, hook] \arrow[r, hook, two heads] \arrow[dr, hook, two heads, bend right] & \mathcal{M}_{lt} \arrow[u, hook] \arrow[d, hook] & \mathcal{M}_l^{\ominus} \arrow[l, hook', two heads] \arrow[ul, hook, two heads, bend right] \arrow[d, hook] \\
& \mathcal{M}_t & \mathcal{M}^{\ominus} \arrow[l, hook', two heads]
\end{tikzcd}
$$

Figure 3.2.: Inclusion functors between the subcategories of a unital magmoid $\mathcal{M}$. The double arrow head ($\twoheadrightarrow$) means full, and the hook ($\hookrightarrow$) means faithful.

- The *intermediate subcategory* $\mathcal{M}_i$, restricting to intermediate morphisms.
- The *negative subcategory* $\mathcal{M}^{\ominus}$, restricting to semantically negative objects.
- The *positive subcategory* $\mathcal{M}^{\oplus}$, restricting to semantically positive objects.
- Combinations of the above, notably:
  - The *linear-and-thunkable subcategory* $\mathcal{M}_{lt}$, restricting to linear-and-thunkable morphisms.
  - The *linear-and-thunkable-and-intermediate subcategory* $\mathcal{M}_{lti}$.
  - The *negative-linear subcategory* $\mathcal{M}_l^{\ominus}$, the full subcategory of $\mathcal{M}_l$ of semantically negative objects.
  - The *positive-thunkable subcategory* $\mathcal{M}_t^{\oplus}$, the full subcategory of $\mathcal{M}_t$ of semantically negative objects.

By the property of polarization, all morphisms of $\mathcal{M}^{\ominus}$ and $\mathcal{M}_l^{\ominus}$ are thunkable and intermediate, and all morphisms of $\mathcal{M}^{\oplus}$ and $\mathcal{M}_t^{\oplus}$ are linear and intermediate. Figure 3.2 depicts some inclusion functors between the subcategories.

Notation 3.22. Let $a, b : \mathcal{M}$ be objects in a unital magmoid $\mathcal{M}$. We may write the subscripts from Definition 3.21 onto the binary operators of morphisms $a \to b$ or isomorphisms $a \cong b$ to denote that the morphism or isomorphism resides in the corresponding subcategory of the unital magmoid. For example, $a \to_l b$ denotes a linear morphism, and $a \cong_{lt} b$ denotes a *linear-and-thunkable isomorphism*: an isomorphism which is linear-and-thunkable, and has a linear-and-thunkable inverse.

3.3. Univalence

Lemma 3.23 (i_iso_interpose)**.** *In a unital magmoid, for an intermediate isomorphism* $p : b \cong_i b'$*, the equation* $(f \mathbin{⨾} p) \mathbin{⨾} (p^{-1} \mathbin{⨾} g) = f \mathbin{⨾} g$ *holds for all* $a \xrightarrow{f} b \xrightarrow{g} c$*.*

Proof. The path $a \xrightarrow{f} b \xrightarrow{p}_i b' \xrightarrow{p^{-1}}_i b \xrightarrow{g} c$ associates by intermediateness of p and p^{-1}. Thus we have

$$\begin{aligned}\left(a \xrightarrow{f} b \xrightarrow{p}_i b'\right) ⨾ \left(b' \xrightarrow{p^{-1}}_i b \xrightarrow{g} c\right) &= a \xrightarrow{f} \left(b \xrightarrow{p}_i b' \xrightarrow{p^{-1}}_i b\right) \xrightarrow{g} c \\ &= a \xrightarrow{f} b \xrightarrow{g} c\,.\end{aligned}$$

□

Lemma 3.24. *In a unital magmoid, if* $p : b \cong b'$ *is an isomorphism with thunkable inverse* $p^{-1} \colon b' \to_t b$*, and* p *satisfies* $(f \mathbin{⨾} p) \mathbin{⨾} (p^{-1} \mathbin{⨾} g) = f \mathbin{⨾} g$ *for all* $a \xrightarrow{f} b \xrightarrow{g} c$*, then* $p \colon b \to b'$ *is intermediate.*

Proof. Fix $f \colon a \to b$ and $g \colon b' \to c$. We need to show $f \mathbin{⨾} (p \mathbin{⨾} g) = (f \mathbin{⨾} p) \mathbin{⨾} g$. Indeed, we have

$$\begin{aligned}a \xrightarrow{f} \left(b \xrightarrow{p} b' \xrightarrow{g} c\right) &= \left(a \xrightarrow{f} b \xrightarrow{p} b'\right) ⨾ \left(b' \xrightarrow{p^{-1}}_t b \xrightarrow{p} b' \xrightarrow{g} c\right) \\ &= \left(a \xrightarrow{f} b \xrightarrow{p} b'\right) \xrightarrow{g} c\,.\end{aligned}$$

□

Lemma 3.25. *In a unital magmoid, the linear-and-thunkable-and-intermediate isomorphisms* $a \cong_{lti} b$ *correspond to the indiscernibilities* $a \asymp b$ *as defined by Ahrens et al. [2].*

Proof. (sketch) This follows from the previous lemmas (3.23, 3.24, TODO Yoneda). Note that the structure of unital magmoids does not differ from that of a category. □

Definition 3.26. A unital magmoid $\mathcal{M}$ is *univalent* if the linear-and-thunkable-and-intermediate subcategory $\mathcal{M}_{lti}$ is univalent (Definition 3.8).

4. ROUGH Duploids

4.1. Definitions

Definition 4.1 (`preduploid` [21, 4, 17]). A *preduploid* (or *single-sorted preduploid* for emphasis) is a unital magmoid such that each object is semantically negative or semantically positive.

Definition 4.2 (`duploid` [21, 4, 17]). A *duploid* (or *single-sorted duploid*) is a preduploid $\mathcal{D}$ equipped with *polarity shifts*:

1. (`has_negative_shifts`) For every object $a : \mathcal{D}$,
   a) its *upshift*, a semantically negative object written $\Uparrow a : \mathcal{D}$,
   b) a linear morphism $\texttt{force}_a : \Uparrow a \to a$,
   c) a linear inverse for $\texttt{force}_a$ called $\texttt{delay}_a : a \to \Uparrow a$.
2. (`has_positive_shifts`) Dually, for every object $a : \mathcal{D}$,
   a) its *downshift*, a semantically positive object written $\Downarrow a : \mathcal{D}$,
   b) a thunkable morphism $\texttt{wrap}_a : a \to \Downarrow a$,
   c) a thunkable inverse for $\texttt{wrap}_a$ called $\texttt{unwrap}_a : \Downarrow a \to a$.

Lemma 4.3 (`negative_shift_axioms_iff_has_negative_shifts_up` [17]). *The polarity shifts of a Definition 4.2 are equivalent to the following universal properties. [TODO : I do not actually use this definition at the moment, except in the formalization to prove some properties.]*

1. *The* negative shift *of an object $a : \mathcal{D}$ is an object $\Uparrow a : \mathcal{D}$ and a linear and monic morphism $\texttt{force}_a : \Uparrow a \to a$, such that for all morphisms $f : a \to b$ there exists a unique thunkable morphism $\mathsf{lift}^{\ominus} f : a \to \Uparrow b$ satisfying $\mathsf{lift}^{\ominus} f \mathbin{⨟} \texttt{force}_b = f$.*

2. *The* positive shift *of an object $a : \mathcal{D}$ is an object $\Downarrow a : \mathcal{D}$ and a thunkable and epi morphism* $\mathtt{wrap}_a : a \to \Downarrow a$*, such that for all morphisms $f \colon a \to b$ there exists a unique linear morphism* $\mathsf{lift}^{\oplus} f \colon \Downarrow a \to b$ *satisfying* $\mathtt{wrap}_b \mathbin{⨾} \mathsf{lift}^{\oplus} f = f$*.*

These are depicted in the commutative diagrams:

$$\begin{tikzcd} a \arrow[d, dashed, "\exists!\,\mathsf{lift}^{\ominus} f"'] \arrow[dr, "f"] & \\ \Uparrow b \arrow[r, "\mathtt{force}_b"'] & b \end{tikzcd} \qquad \begin{tikzcd} a \arrow[r, "\mathtt{wrap}_a"] \arrow[dr, "f"'] & \Downarrow a \arrow[d, dashed, "\exists!\,\mathsf{lift}^{\oplus} f"] \\ & b \end{tikzcd}$$

Proof. For negative shifts (and dually positive shifts), the data $\Uparrow$ and $\mathtt{force}$ are the same as Definition 4.2. We leave it to the reader to verify that $\Uparrow a$ is negative, that $\mathtt{delay}_a \overset{\text{def}}{\equiv} \mathsf{lift}^{\ominus}\, \mathsf{id}_a$ is a linear inverse for $\mathtt{force}_a$, and in the other direction that $\mathsf{lift}^{\ominus} f \overset{\text{def}}{\equiv} f \mathbin{⨾} \mathtt{delay}_a$ satisfies the universal property. □

Remark 4.4. [TODO: this is pending a move and rewrite in a better location] The above definitions make the polarities in a (pre)duploid be a *property* on unital magmoids, rather than *structure*. This differs from the original construction [21] which defined (pre)duploids in terms of separate negative and positive categories, with additional mixed composition operators, or as explicit structure. I therefore call the (pre)duploids from the original definition *two-sorted (pre)duploids* to distinguish them from the single-sorted (pre)duploids defined above. It is not always immediate how the theorems of single-sorted (pre)duploids transfer over to their single-sorted counterparts, and some of this dissertation consists of developing this theory.

We subsume two-sorted (pre)duploids into the theory of single-sorted (pre)duploids by considering them as extra structure on a single-sorted (pre)duploid. A *split preduploid* (`split_preduploid`) is a unital magmoid $\mathcal{D}$ equipped with a chosen *polarity mapping* $\omega \colon \mathsf{ob}\, \mathcal{D} \to \{\ominus, \oplus\}$. We say that an object a is *syntactically negative* if $\omega(a) = \ominus$ and *syntactically positive* if $\omega(a) = \oplus$, and require that every **syntactically** negative/positive object is also **semantically** negative/positive respectively. If $\mathcal{D}$ is a single-sorted duploid and a split preduploid, then $\mathcal{D}$ is a *split duploid* (`split_duploid`) when $\Uparrow a$ and $\Downarrow a$ are also syntactically negative and positive respectively. The original two-sorted duploids are precisely the split duploids where $\Uparrow$ and $\Downarrow$ are trivial (identity-on-objects and `wrap`/`unwrap`/`force`/`delay` identities) for syntactically negative and syntactically positive objects respectively.

4.2. Univalence

Lemma 4.5 (is_positive_of_lt_iso). *In a unital magmoid $\mathcal{M}$ with a linear-and-thunkable isomorphism $p : a \cong_{lt} b$, the object a is positive if and only if b is positive, and a is negative if and only if b is negative.*

Proof. Without loss of generality, we need only prove that a being positive implies that b is positive. By definition, this is that for any object c, any morphism $f \colon b \to c$ is linear. We have

$$\begin{aligned} b \xrightarrow{f} c &= \left(b \xrightarrow{p^{-1}}_{lt} a \xrightarrow{p}_{lt} b \right) \xrightarrow{f} c \\ &= b \xrightarrow{p^{-1}}_{lt} \left(a \xrightarrow{p}_{lt} b \xrightarrow{f} c \right) . \end{aligned}$$

This is linear because p^{-1} is linear and a is positive. □

Lemma 4.6 (lt_iso_is_intermediate_from_polarized). *In a unital magmoid $\mathcal{M}$, if an object $a : \mathcal{M}$ is positive or negative, then all linear-and-thunkable isomorphisms $p : a \cong_{lt} b$ are intermediate.*

Proof. If a is negative, then p is automatically intermediate. If a is positive, then b must be positive too by Lemma 4.5, and so p is automatically intermediate. □

Corollary 4.7 (preduploid_lt_iso_is_intermediate). *Every linear-and-thunkable isomorphism $p : a \cong_{lt} b$ in a preduploid is intermediate.*

Proof. This follows from Lemma 4.6 since a is positive or negative by the preduploid property (Definition 4.1). □

Definition 4.8 (is_duploid_univalent). A duploid or preduploid $\mathcal{D}$ is *univalent* if its linear-and-thunkable subcategory $\mathcal{D}_{lt}$ is univalent.

Remark 4.9 (unital_magmoid_univalent_iff_duploid_univalent). By Corollary 4.7, this coincides with univalence of $\mathcal{D}$ as a unital magmoid.

Lemma 4.10 (is_rxgraph_univalent_from_isaprop_edges_from [23, Chapter 11]). *A category $\mathcal{C}$ is univalent if and only if for all $a : \mathcal{C}$ the type $\sum_{b:\mathcal{C}} a \cong b$, called the type of* fans *of a, is a proposition. Equivalently, if the type of fans is contractible with centre $\langle a, \mathsf{id}_a \rangle : \sum_{b:\mathcal{C}} a \cong b$.*

Proof. Observe that

$$\texttt{id\_to\_edge}\ a : \prod_{b:\mathcal{C}} (a =_{\mathsf{ob}\,\mathcal{C}} b) \to (a \cong b)$$

may be curried (totalfun) to a function

$$\texttt{totalfun}(\texttt{id\_to\_edge}\ a) : \left(\sum_{b:\mathcal{C}} a =_{\mathsf{ob}\,\mathcal{C}} b\right) \to \left(\sum_{b:\mathcal{C}} a \cong b\right).$$

We have that $\texttt{id\_to\_edge}\ a\ b$ is an equivalence if and only if $\texttt{totalfun}(\texttt{id\_to\_edge}\ a)$ is. Moreover, we have that the codomain of $\texttt{totalfun}(\texttt{id\_to\_edge}\ a)$, the singleton type $\sum_{b:\mathcal{C}} a = b$, is contractible. Thus, in the "only if" direction, we have that the fans $\sum_{b:\mathcal{C}} a =_{\mathsf{ob}\,\mathcal{C}} b$, equivalent to the singleton type, are contractible. Finally, in the "if" direction, we have that $\texttt{totalfun}(\texttt{id\_to\_edge}\ a)$ is an equivalence because both its domain and codomain are contractible. □

Remark 4.11. The formalization and Rijke [23] make a more general statement about "reflexive (pre)graphs" and "identity systems" respectively, but we do not develop the language for those statements in this dissertation.

Lemma 4.12 (is_duploid_univalent_from_positive_thunkable_and_negative_-linear_categories)**.** *A preduploid $\mathcal{D}$ is univalent if and only if the subcategories $\mathcal{D}_t^{\oplus}$ and $\mathcal{D}_l^{\ominus}$ are univalent.*

Proof. The "only if" direction follows immediately from the fully-faithfulness of the inclusion functors $\mathcal{M}_t^{\oplus} \hookrightarrow\!\!\!\rightarrow \mathcal{M}_{lt} \leftarrow\!\!\!\hookleftarrow \mathcal{M}_l^{\ominus}$.

For the "if" direction, we apply Lemma 4.10 and show that for all $a : \mathcal{D}$ the type of fans $\sum_{b:\mathcal{D}} a \cong_{lt} b$ is a proposition. By the preduploid property, assume that a is positive (dually, negative). Then fans $\sum_{b:\mathcal{D}} a \cong_{lt} b$ are equivalent to fans $\sum_{b:\mathcal{D}_t^{\oplus}} a \cong_t^{\oplus} b$ (dually, $\sum_{b:\mathcal{D}_l^{\ominus}} a \cong_l^{\ominus} b$) by Lemma 4.5. The latter type is a proposition by assumption. □

5. ROUGH Adjunctions and Duploids

5.1. ROUGH The Oblique Duploid

Definition 5.1 (adjunction [1]). An *adjunction* between categories $\mathcal{P}$ and $\mathcal{N}$ consists of two functors $L\colon \mathcal{P} \to \mathcal{N}$ and $R\colon \mathcal{N} \to \mathcal{P}$, along with two natural transformations: the *unit* $\eta\colon \mathsf{Id}_{\mathcal{P}} \Rightarrow RL$ and *counit* $\varepsilon\colon LR \Rightarrow \mathsf{Id}_{\mathcal{N}}$. These must satisfy the "triangle equations"

$$\begin{aligned} L\eta_p \mathbin{⨟} \varepsilon_{Fp} &= \mathsf{id}_{Fp} && \text{for all } p : \mathcal{P}\text{, and} \\ \eta_{Gn} \mathbin{⨟} G\varepsilon_n &= \mathsf{id}_{Gn} && \text{for all } n : \mathcal{N}. \end{aligned}$$

The adjunction itself may be written as any one of

$$L \dashv R \quad \text{or} \quad L \dashv_{(\eta,\varepsilon)} R \quad \text{or}$$

$$L \dashv_{(\eta,\varepsilon)} R : \mathcal{N} \to \mathcal{P} \quad \text{or} \quad \begin{tikzcd} \mathcal{P} \arrow[r, bend left, "L"] \arrow[r, phantom, "\perp"] & \mathcal{N} \arrow[l, bend left, "R"] \end{tikzcd} .$$

Remark 5.2 (natural_hom_weq, natural_bi_hom_weq). It is well-known that another way to formulate an adjunction $L \dashv_{(\eta,\varepsilon)} R : \mathcal{N} \to \mathcal{P}$ is as an equivalence of sets

$$\mathcal{N}[\![Lp, n]\!] \overset{\varphi_{p,n}}{\simeq} \mathcal{P}[\![p, Rn]\!] \qquad \text{natural in } p : \mathcal{P}^{\mathsf{op}} \text{ and } n : \mathcal{N}. \tag{5.1}$$

Explicitly, φ may be given in terms of η and ε as

$$\varphi_{p,n}\left(Lp \xrightarrow{f} n\right) \overset{\text{def}}{\equiv} p \xrightarrow{\eta_p} RLp \xrightarrow{Rf} Rn \tag{5.2}$$

$$\varphi^{-1}_{p,n}\left(p \xrightarrow{g} Rn\right) \overset{\text{def}}{\equiv} Lp \xrightarrow{Lg} LRn \xrightarrow{\varepsilon_n} n\,. \tag{5.3}$$

Following Levy [14], we may call the morphisms on either side of φ (5.1) the *oblique morphisms*, adopting the following definition.

Definition 5.3 (oblique_mor)**.** Let $\varphi_{p,n} : \mathcal{N}[\![Lp, n]\!] \cong \mathcal{P}[\![p, Rn]\!]$ be the natural isomorphism witnessing an adjunction $L \dashv R$. An *oblique morphism* f in $L \dashv R$ from $p : \mathcal{P}$ to $n : \mathcal{N}$, written $f : \mathcal{O}_{L\dashv R}[\![p, n]\!]$, or simply $\mathcal{O}[\![p, n]\!]$, is a pair of *mates*: a morphism $f^\flat : \mathcal{N}[\![Lp, n]\!]$ and a morphism $f^\sharp : \mathcal{P}[\![p, Rn]\!]$ such that $\varphi_{p,n}(f^\flat) = f^\sharp$.

Convention 5.4. We will draw diagrams involving oblique morphisms in two columns, one on the left for the $\mathcal{N}$ mate and another on the right for the $\mathcal{P}$ mate. For example, $f : \mathcal{O}[\![p, n]\!]$ on its own may be depicted as

$$Lp \xrightarrow{f^\flat} n \qquad\qquad p \xrightarrow{f^\sharp} Rn \ .$$

Remark 5.5 (oblique_compose_negative, …). $\mathcal{O}$ extends to a category functor $\mathcal{P}^{\mathsf{op}} \times \mathcal{N} \to \mathbf{Set}$ by pre- and post-composition in the respective categories, as

$$\begin{tikzcd} Lp \arrow[d, "Lf"'] & \\ Lq \arrow[r, "g^\flat"] & n \arrow[d, "h"] \\ & m \end{tikzcd} \qquad \begin{tikzcd} p \arrow[d, "f"'] & \\ q \arrow[r, "g^\sharp"] & Rn \arrow[d, "Rh"] \\ & Rm \end{tikzcd},$$

with the two sides being mates precisely by naturality of φ.

Remark 5.6 (isweq_oblique_mor_negative, …). The equation $\varphi_{p,n}(f^\flat) = f^\sharp$ for an oblique morphism $f : \mathcal{O}[\![p, n]\!]$ means that each mate is uniquely determined by the other. The projections $\flat$ and $\sharp$ thus have the obvious inverses

$$\flat^{-1} : \mathcal{N}[\![Lp, n]\!] \to \mathcal{O}[\![p, n]\!] \quad \text{and} \quad \sharp^{-1} : \mathcal{P}[\![p, Rn]\!] \to \mathcal{O}[\![p, n]\!]$$

such that

$$\begin{aligned} \left(g^{\flat^{-1}}\right)^\sharp &=_{\mathcal{P}[\![n, Rp]\!]} \varphi_{p,n}(g) \quad \text{for } g : \mathcal{N}[\![Ln, p]\!], \text{ and} \\ \left(h^{\sharp^{-1}}\right)^\flat &=_{\mathcal{N}[\![Lp, n]\!]} \varphi_{p,n}^{-1}(h) \quad \text{for } h : \mathcal{P}[\![n, Rp]\!]. \end{aligned}$$

This way we have

$$\mathcal{N}[\![Lp, n]\!] \overset{\flat}{\simeq} \mathcal{O}[\![p, n]\!] \overset{\sharp}{\simeq} \mathcal{P}[\![p, Rn]\!] \qquad \text{natural in } p : \mathcal{P}^{\mathsf{op}} \text{ and } n : \mathcal{N}.$$

NOTE: The oblique duploid formalization considerably diverges from this description. The formalization defines it more directly as Guillaume described it, but ported to oblique morphisms, rather than taking a bias. The description of composition and shifts here is actually a description of the envelope duploid.

Definition 5.7 (`oblique_ob`, `oblique_mor'`)**.** Let $L \dashv R : \mathcal{N} \to \mathcal{P}$ be an adjunction. The *oblique duploid-to-be* $\mathcal{O}[L \dashv R]$ over $L \dashv R$ has its graph structure—objects and morphisms—defined as follows.

- The objects $\mathsf{ob}\,\mathcal{O}[L \dashv R] \stackrel{\text{def}}{\equiv} \mathcal{N} \amalg \mathcal{P}$ are the objects from either $\mathcal{N}$ or $\mathcal{P}$. These are written respectively $\iota^{\ominus}_{\mathcal{O}}\, n$ (or simply $\iota^{\ominus}\, n$) for $n : \mathcal{N}$, and $\iota^{\oplus}_{\mathcal{O}}\, p$ (or simply $\iota^{\oplus}\, p$) for $p : \mathcal{P}$. The inclusions $\iota^{\ominus}$ and $\iota^{\oplus}$ may be omitted, particularly in diagrams.

- For an object $a : \mathsf{ob}\,\mathcal{O}[L \dashv R]$, define the *negative* and *positive projections* $\pi^{\ominus}_{\mathcal{O}}\, a : \mathcal{N}$ and $\pi^{\oplus}_{\mathcal{O}}\, a : \mathcal{P}$, or simply $\pi^{\ominus}\, a$ and $\pi^{\oplus}\, a$, as

$$\pi^{\ominus}(\iota^{\ominus}\, n) \stackrel{\text{def}}{\equiv} n \qquad \pi^{\oplus}(\iota^{\ominus}\, n) \stackrel{\text{def}}{\equiv} Ln$$
$$\pi^{\ominus}(\iota^{\oplus}\, p) \stackrel{\text{def}}{\equiv} Rp \qquad \pi^{\oplus}(\iota^{\oplus}\, p) \stackrel{\text{def}}{\equiv} p\,.$$

- The morphisms $\mathcal{O}[L \dashv R][\![a, b]\!]$ are given by the oblique morphisms $\mathcal{O}_{L \dashv R}[\![\pi^{\oplus}\, a, \pi^{\ominus}\, b]\!]$ between the positive and negative projections.

Definition 5.8 (`oblique_negative_identity`, …)**.** For an object $a : \mathcal{O}[L \dashv R]$ of the oblique duploid-to-be, its *cross morphism* $\mathsf{cross}_{\mathcal{O}}\, a$, or simply $\mathsf{cross}\, a$, is an oblique morphism $\mathsf{cross}\, a : \mathcal{O}_{L \dashv R}[\![\pi^{\oplus}\, a, \pi^{\ominus}\, a]\!]$, defined as

$$\mathsf{cross}(\iota^{\oplus}\, p)^{\flat} \stackrel{\text{def}}{\equiv} \mathsf{id}_{Lp}$$
$$\mathsf{cross}(\iota^{\ominus}\, n)^{\sharp} \stackrel{\text{def}}{\equiv} \mathsf{id}_{Rn}\,.$$

That is, the cross morphism has components

$$Lp \xrightarrow{\mathsf{id}_{Lp}} Lp \qquad p \xrightarrow{\eta_p} RLp \qquad \text{when } a \equiv \iota^{\oplus}\, p\text{, and}$$

$$LRn \xrightarrow{\varepsilon_n} n \qquad Rn \xrightarrow{\mathsf{id}_{Rn}} Rn \qquad \text{when } a \equiv \iota^{\ominus}\, n.$$

Definition 5.9 (`oblique_identity`)**.** The oblique duploid-to-be $\mathcal{O}[L \dashv R]$ has reflexive graph structure (Definition 3.1.3)—that is, identities—given by the cross morphisms. That is, for $a : \mathcal{O}[L \dashv R]$ define $\mathsf{id}_a \stackrel{\text{def}}{\equiv} \mathsf{cross}\, a$.

How should we compose the morphisms of the oblique duploid-to-be? Given morphisms $f : \mathcal{O}[\![\pi^{\oplus}\, a, \pi^{\ominus}\, b]\!]$ and $g : \mathcal{O}[\![\pi^{\oplus}\, b, \pi^{\ominus}\, c]\!]$, we must find an oblique morphism $f \mathbin{;} g : \mathcal{O}[\![\pi^{\oplus}\, a, \pi^{\ominus}\, c]\!]$. Since composition shall need to be unital, it must satisfy $\mathsf{cross}\, b \mathbin{;} g = g$ and $f \mathbin{;} \mathsf{cross}\, b = f$. Let us therefore depict f, g, and $\mathsf{cross}\, b$ on a diagram:

$$\begin{tikzcd}
L(\pi^{\oplus}\, a) \arrow[dr, "f^{\flat}"] & \\
L(\pi^{\oplus}\, b) \arrow[r, "(\mathsf{cross}\, b)^{\flat}"] \arrow[dr, "g^{\flat}"'] & \pi^{\ominus}\, b \\
& \pi^{\ominus}\, c
\end{tikzcd}
\qquad
\begin{tikzcd}
\pi^{\oplus}\, a \arrow[dr, "f^{\sharp}"] & \\
\pi^{\oplus}\, b \arrow[r, "(\mathsf{cross}\, b)^{\sharp}"] \arrow[dr, "g^{\sharp}"'] & R(\pi^{\ominus}\, b) \\
& R(\pi^{\ominus}\, c)
\end{tikzcd}$$

We need to find a composite in either category: either $(f \mathbin{;} g)^{\flat} : \mathcal{N}[\![L(\pi^{\oplus}\, a), \pi^{\ominus}\, c]\!]$ or $(f \mathbin{;} g)^{\sharp} : \mathcal{P}[\![\pi^{\oplus}\, a, R(\pi^{\ominus}\, c)]\!]$. If we had a morphism in the opposite direction to $\mathsf{cross}\, b$ in either category, so one $h : \mathcal{N}[\![\pi^{\ominus}\, b, L(\pi^{\oplus}\, b)]\!]$ or $k : \mathcal{P}[\![R(\pi^{\ominus}\, b), \pi^{\oplus}\, b]\!]$, then we could simply compose it between f and g as $f^{\flat} \mathbin{;} h \mathbin{;} g^{\flat}$ or $f^{\sharp} \mathbin{;} k \mathbin{;} g^{\sharp}$ respectively. We do have such a morphism: recall from Definition 5.8 that either $(\mathsf{cross}\, b)^{\flat}$ or $(\mathsf{cross}\, b)^{\sharp}$ is the identity, and thus an isomorphism.

Definition 5.10 (`oblique_compose`)**.** The oblique duploid-to-be $\mathcal{O}[L \dashv R]$ has magmoid structure (Definition 3.1.4) defined as follows. Let $f : \mathcal{O}[L \dashv R][\![a, b]\!]$ and $g : \mathcal{O}[L \dashv R][\![b, c]\!]$ be morphisms in the oblique duploid. Define $f \mathbin{;} g : \mathcal{O}[L \dashv R][\![a, c]\!]$ by discrimination on b, as

$$(f \mathbin{;} g)^{\sharp} \stackrel{\text{def}}{\equiv} \left(\pi^{\oplus}\, a \xrightarrow{f^{\sharp}} R(\pi^{\ominus}\, b) \xrightarrow{((\mathsf{cross}\, b)^{\sharp})^{-1}} \pi^{\oplus}\, b \xrightarrow{g^{\sharp}} R(\pi^{\ominus}\, c)\right) \qquad \text{if } b \equiv \iota^{\ominus}\, n$$
$$(f \mathbin{;} g)^{\flat} \stackrel{\text{def}}{\equiv} \left(L(\pi^{\oplus}\, a) \xrightarrow{f^{\flat}} \pi^{\ominus}\, b \xrightarrow{((\mathsf{cross}\, b)^{\flat})^{-1}} L(\pi^{\oplus}\, b) \xrightarrow{g^{\flat}} \pi^{\ominus}\, c\right) \qquad \text{if } b \equiv \iota^{\oplus}\, p\,.$$

The other mate is

$$(f \mathbin{;} g)^{\flat} \stackrel{\text{def}}{\equiv} \left(L(\pi^{\oplus}\, a) \xrightarrow{Lf^{\sharp}} LR(\pi^{\ominus}\, b) \xrightarrow{L((\mathsf{cross}\, b)^{\sharp})^{-1}} L(\pi^{\oplus}\, b) \xrightarrow{g^{\flat}} \pi^{\ominus}\, c\right) \qquad \text{if } b \equiv \iota^{\ominus}\, n$$
$$(f \mathbin{;} g)^{\sharp} \stackrel{\text{def}}{\equiv} \left(\pi^{\oplus}\, a \xrightarrow{f^{\flat}} R(\pi^{\ominus}\, b) \xrightarrow{R((\mathsf{cross}\, b)^{\flat})^{-1}} RL(\pi^{\oplus}\, b) \xrightarrow{Rg^{\flat}} R(\pi^{\ominus}\, c)\right) \qquad \text{if } b \equiv \iota^{\oplus}\, p\,.$$

Remark 5.11. The above definition "forgets" that the mentioned isomorphisms $(\mathsf{cross}\, b)^{\sharp}$ and $(\mathsf{cross}\, b)^{\flat}$ are the identity, in anticipation of the generalization in Section 5.2. If we remember this fact, we can instead obtain, for $a \xrightarrow{f} b \xrightarrow{g} c$, the definitions

$$
\begin{aligned}
(f \mathbin{⨟} g)^\sharp &\stackrel{\text{def}}{\equiv} \left(\pi^\oplus\, a \xrightarrow{f^\sharp} Rn \xrightarrow{g^\sharp} R(\pi^\ominus\, c)\right) && \text{if } b \equiv \iota^\ominus\, n \\
(f \mathbin{⨟} g)^\flat &\stackrel{\text{def}}{\equiv} \left(L(\pi^\oplus\, a) \xrightarrow{f^\flat} Lp \xrightarrow{g^\flat} \pi^\ominus\, c\right) && \text{if } b \equiv \iota^\oplus\, p\,.
\end{aligned}
$$

This is the actual definition of `oblique_compose` used in the formalization, and the original construction by Munch-Maccagnoni [21].

Lemma 5.12 (`oblique_left_id`, …)**.** *The oblique duploid-to-be $\mathcal{O}[L \dashv R]$ with composition as in Definition 5.10 is unital (Definition 3.1.4.a), and thus a unital magmoid.*

Proof. Let $f\colon a \to b$ in $\mathcal{O}[L \dashv R]$. Let us show that $\mathsf{id}_a \mathbin{⨟} f = f$. Suppose $a \equiv \iota^\oplus\, p$; then we have

$$
\begin{aligned}
(\mathsf{cross}\, a \mathbin{⨟} f)^\flat &\equiv (\mathsf{cross}\, a)^\flat \mathbin{⨟} ((\mathsf{cross}\, a)^\flat)^{-1} \mathbin{⨟} f^\flat \\
&= f^\flat\,.
\end{aligned}
$$

Thus we have $\mathsf{cross}\, a \mathbin{⨟} f = f$ by injectivity of the equivalence $\flat$. The cases for $a \equiv \iota^\ominus\, n$ and the two cases for post-composition are all analogous. □

Lemma 5.13 (`oblique_negative_is_negative`, …)**.** *In the oblique duploid-to-be $\mathcal{O}[L \dashv R]$, the objects $\iota^\ominus\, n$ for $n : \mathcal{N}$ are semantically negative, and those $\iota^\oplus\, p$ for $p : \mathcal{P}$ are semantically positive (Definition 3.17).*

Proof. All objects $\iota^\ominus\, n$ are negative; that is, all triples $a \xrightarrow{f} x \xrightarrow{g} c \xrightarrow{h} d$ with $x \equiv \iota^\ominus\, n$ associate. We leave it to the reader to compute $\left(a \xrightarrow{f} \left(x \xrightarrow{g} c \xrightarrow{h} d\right)\right)^\sharp = \left(\left(a \xrightarrow{f} x \xrightarrow{g} c\right) \xrightarrow{h} d\right)^\sharp$, by discriminating on c.[1] The other polarity is symmetric. □

Corollary 5.14 (`oblique_preduploid`)**.** *The oblique duploid-to-be $\mathcal{O}[L \dashv R]$ is a preduploid.*

Definition 5.15 (`oblique_negative_shift_data`, …)**.** The oblique duploid-to-be $\mathcal{O}[L \dashv R]$ has polarity shift structure (Definition 4.2) given on objects by

$$
{\Uparrow} a \stackrel{\text{def}}{\equiv} \iota^\ominus(\pi^\ominus\, a) \quad \text{and} \quad {\Downarrow} a \stackrel{\text{def}}{\equiv} \iota^\oplus(\pi^\oplus\, a)\,.
$$

[1] Bonus points for forgetting $(\mathsf{cross}\, x)^\sharp \equiv \mathsf{id}_{Rn}$.

For the morphism $\texttt{force}_a : \mathcal{O}[L \dashv R][\![\Uparrow a, a]\!]$, compute its type

$$\begin{aligned}\mathcal{O}[L \dashv R][\![\Uparrow a, a]\!] &\equiv \mathcal{O}[\![\pi^{\oplus}(\Uparrow a), \pi^{\ominus}\, a]\!] \\ &\equiv \mathcal{O}[\![\pi^{\oplus}(\Uparrow a), \pi^{\ominus}(\Uparrow a)]\!]\,,\end{aligned}$$

and give $\texttt{force}_a \stackrel{\text{def}}{\equiv} \mathsf{cross}\Uparrow a : \mathcal{O}[\![\pi^{\oplus}(\Uparrow a), \pi^{\ominus}(\Uparrow a)]\!]$. For its inverse-to-be $\texttt{delay}_a$: $\mathcal{O}[L \dashv R][\![a, \Uparrow a]\!]$, instead compute

$$\begin{aligned}\mathcal{O}[L \dashv R][\![a, \Uparrow a]\!] &\equiv \mathcal{O}[\![\pi^{\oplus}\, a, \pi^{\ominus} \Uparrow a]\!] \\ &\equiv \mathcal{O}[\![\pi^{\oplus}\, a, \pi^{\ominus}\, a]\!]\end{aligned}$$

and give $\texttt{delay}_a \stackrel{\text{def}}{\equiv} \mathsf{cross}\, a : \mathcal{O}[\![\pi^{\oplus}\, a, \pi^{\ominus}\, a]\!]$.

Dually, define $\texttt{wrap}_a \stackrel{\text{def}}{\equiv} \mathsf{cross}\Downarrow a$ of the type $\mathcal{O}[L \dashv R][\![a, \Downarrow a]\!] \equiv \mathcal{O}[\![\pi^{\oplus}(\Downarrow a), \pi^{\ominus}(\Downarrow a)]\!]$, and $\texttt{unwrap}_a \stackrel{\text{def}}{\equiv} \mathsf{cross}\, a$ of the type $\mathcal{O}[L \dashv R][\![\Downarrow a, a]\!] \equiv \mathcal{O}[\![\pi^{\oplus}\, a, \pi^{\ominus}\, a]\!]$.

Lemma 5.16 (`oblique_negative_shift_axioms`, …). *The polarity shift structure of Definition 5.15 for the oblique duploid-to-be $\mathcal{O}[L \dashv R]$ satisfies the axioms of polarity shifts (Definition 4.2).*

Proof. The polarities of the shifts follow from Lemma 5.13. We leave it to the reader to check the remainder of the axioms. □

Corollary 5.17 (`oblique_duploid`). *The oblique duploid-to-be $\mathcal{O}[L \dashv R]$ is a duploid. Hence, we now call it the* oblique duploid.

The oblique duploid fails to be univalent.

Definition 5.18. Let $L \dashv R : \mathcal{N} \to \mathcal{P}$ be an adjunction. Define the polarity mapping $\omega\colon \mathsf{ob}\,\mathcal{O}[L \dashv R] \to \{\ominus, \oplus\}$ as

$$\begin{aligned}\omega(\iota^{\oplus}\, p) &\stackrel{\text{def}}{\equiv} \oplus \\ \omega(\iota^{\ominus}\, n) &\stackrel{\text{def}}{\equiv} \ominus\,.\end{aligned}$$

Lemma 5.19 (`neg_is_univalent_oblique_from_positive_and_negative`). *Let $L \dashv R : \mathcal{N} \to \mathcal{P}$ be an adjunction, and $a : \mathcal{O}[L \dashv R]$ an object of the oblique duploid which is both semantically positive and semantically negative. Then the oblique duploid $\mathcal{O}[L \dashv R]$ is not univalent.*

Proof. If a is both positive and negative, then there is a linear-and-thunkable isomorphism

$$\Uparrow a \overset{\mathtt{force}_a}{\cong_{lt}} a \overset{\mathtt{wrap}_a}{\cong_{lt}} \Downarrow a\,.$$

If $\mathcal{O}[L \dashv R]$ is univalent, then we have $\Uparrow a =_{\mathsf{ob}\,\mathcal{O}[L \dashv R]} \Downarrow a$ by the above. Thus, in particular, the polarity mapping gives

$$\omega(\Uparrow a) \equiv \ominus =_{\{\ominus,\oplus\}} \oplus \equiv \omega(\Downarrow a)\,,$$

which is a contradiction. Therefore $\mathcal{O}[L \dashv R]$ is not univalent. □

5.2. ROUGH The Envelope Duploid

As anticipated in Remark 5.11, we obtain a perfectly serviceable version of the oblique duploid by treating the cross morphisms as merely isomorphisms.

Definition 5.20 (`envelope_preob`)**.** Let $L \dashv R : \mathcal{N} \to \mathcal{P}$ be an adjunction. An *envelope pre-object* in $L \dashv R$ is a triple of a positive object $p : \mathcal{P}$, a negative object $n : \mathcal{N}$, and an oblique morphism $f : \mathcal{O}_{L\dashv R}[\![p, n]\!]$ between them.

Notation 5.21. For an envelope pre-object a in $L \dashv R$, write $\pi^{\oplus}_{\mathcal{E}}\, a : \mathcal{P}$ (or simply $\pi^{\oplus}\, a$) for its positive object, $\pi^{\ominus}_{\mathcal{E}}\, a : \mathcal{N}$ (or simply $\pi^{\ominus}\, a$) for its negative object, and $\mathsf{cross}_{\mathcal{E}}\, a : \mathcal{O}_{L\dashv R}[\![\pi^{\oplus}\, a, \pi^{\ominus}\, a]\!]$ (or simply $\mathsf{cross}\, a$).

Notation 5.22. We may write any envelope pre-object a as simply its oblique morphism:

$$\pi^{\oplus}_{\mathcal{E}}\, a \xrightarrow{\mathsf{cross}_{\mathcal{E}}\, a} \pi^{\ominus}_{\mathcal{E}}\, a\,.$$

Definition 5.23 (`envelope_polarization_choice`)**.** Let a be an envelope pre-object in $L \dashv R$. A *polarization choice* for a is a (two-sided) inverse for either mate of $\mathsf{cross}\, a : \mathcal{O}_{L\dashv R}[\![\pi^{\oplus}\, a, \pi^{\ominus}\, a]\!]$. That is, either

$$\begin{array}{llll} \text{an inverse} & R(\pi^{\ominus}\, a) \xrightarrow{((\mathsf{cross}\, a)^{\sharp})^{-1}} \pi^{\oplus}\, a & \text{for } (\mathsf{cross}\, a)^{\sharp}\,, \text{ or} \\ \text{an inverse} & \pi^{\ominus}\, a \xrightarrow{((\mathsf{cross}\, a)^{\flat})^{-1}} L(\pi^{\oplus}\, a) & \text{for } (\mathsf{cross}\, a)^{\flat}\,. \end{array}$$

We call the former a *negative polarization choice*, and the latter a *positive polarization choice*.

Lemma 5.24 (`weq_oblique_ob_to_split_envelope_ob`). *Let $L \dashv R : \mathcal{N} \to \mathcal{P}$ be an adjunction between univalent categories $\mathcal{N}$ and $\mathcal{P}$. The objects of the oblique duploid $\mathcal{O}[L \dashv R]$ are equivalent to envelope pre-objects in $L \dashv R$ equipped with a polarization choice.*

Proof. In the forward direction, we obtain for any object $a : \mathcal{O}[L \dashv R]$ an envelope pre-object $\hat{a} \overset{\text{def}}{\equiv} \left(\pi^{\oplus}_{\mathcal{O}}\, a \xrightarrow{\mathsf{cross}_{\mathcal{O}}\, p} \pi^{\ominus}_{\mathcal{O}}\, a\right)$. For negative $a \equiv \iota^{\ominus}_{\mathcal{O}}\, n$, the pre-object $\hat{a}$ has a negative polarization choice id_{Rn}; for positive $a \equiv \iota^{\oplus}_{\mathcal{O}}\, p$, it has a positive polarization choice id_{Lp}. To show that $\hat{a}$ gives an equivalence, we must show that for each envelope pre-object b in $L \dashv R$ equipped with a polarization choice h, there exists a unique pair of an oblique duploid object $a : \mathcal{O}[L \dashv R]$ satisfying $\hat{a} = \langle b, h \rangle$. We obtain this by induction on either $\langle (\mathsf{cross}_{\mathcal{E}}\, b)^{\flat}, h \rangle : L(\pi^{\oplus}\, b) \cong \pi^{\ominus}\, b$ or $\langle (\mathsf{cross}_{\mathcal{E}}\, b)^{\sharp}, h \rangle : \pi^{\oplus}\, b \cong R(\pi^{\ominus}\, b)$. [TODO: or rather, contractibility of the fans involving them. I will likely axe this proof, as it is not strictly necessary.] □

Definition 5.25. The *envelope duploid-to-be* $\mathcal{E}[L \dashv R]$ has objects, morphisms, and identities given as follows.

- The objects $\mathsf{ob}\,\mathcal{E}[L \dashv R]$ are envelope pre-objects a in $L \dashv R$ such that there exists a polarization choice for a. That is, $(\mathsf{cross}\, a)^{\sharp}$ or $(\mathsf{cross}\, a)^{\flat}$ is an isomorphism.
- Morphisms $\mathcal{E}[L \dashv R][\![a, b]\!]$ are the oblique morphisms $\mathcal{O}_{L \dashv R}[\![\pi^{\oplus}\, a, \pi^{\ominus}\, b]\!]$.
- For an object $a : \mathcal{E}[L \dashv R]$, its identity is given by its cross morphism $\mathsf{id}_a \overset{\text{def}}{\equiv} \mathsf{cross}\, a$.

Theorem 5.26 (`envelope_compose_irrel`). *Let $f : \mathcal{E}[L \dashv R][\![a, b]\!]$ and $g : \mathcal{E}[L \dashv R][\![b, c]\!]$ be morphisms in the envelope duploid-to-be, such that both polarization choices $((\mathsf{cross}\, b)^{\sharp})^{-1}$ and $((\mathsf{cross}\, b)^{\flat})^{-1}$ of b exist. Then the two ways of composing $f \mathbin{⨾} g : \mathcal{E}[L \dashv R][\![a, c]\!]$, as in the oblique duploid (Definition 5.10), are equal. That is, the following morphisms are mates:*

$$
\begin{aligned}
(f \mathbin{⨾} g)^{\flat} &\overset{\text{def}}{\equiv} \left(L(\pi^{\oplus}\, a) \xrightarrow{L f^{\sharp}} LR(\pi^{\ominus}\, b) \xrightarrow{L((\mathsf{cross}\, b)^{\sharp})^{-1}} L(\pi^{\oplus}\, b) \xrightarrow{g^{\flat}} \pi^{\ominus}\, c \right) \\
(f \mathbin{⨾} g)^{\sharp} &\overset{\text{def}}{\equiv} \left(\pi^{\oplus}\, a \xrightarrow{f^{\flat}} R(\pi^{\ominus}\, b) \xrightarrow{R((\mathsf{cross}\, b)^{\flat})^{-1}} RL(\pi^{\oplus}\, b) \xrightarrow{R g^{\flat}} R(\pi^{\ominus}\, c) \right).
\end{aligned}
$$

Proof. Let $\varphi_{p,n} : \mathcal{N}[\![Lp, n]\!] \cong \mathcal{P}[\![p, Rn]\!]$ be the natural isomorphism witnessing an adjunction $L \dashv R$. We need only check that

$$
R(\pi^{\ominus}\, b) \xrightarrow{R(((\mathsf{cross}\, b)^{\flat})^{-1})} RL(\pi^{\oplus}\, b) \quad \text{and} \quad LR(\pi^{\ominus}\, b) \xrightarrow{L(((\mathsf{cross}\, b)^{\sharp})^{-1})} L(\pi^{\oplus}\, b)
$$

are mates, since it follows by naturality of φ that $\varphi_{(\pi^{\oplus} a),(\pi^{\ominus} a)}\left((f \mathbin{⨟} g)^{\flat}\right) = (f \mathbin{⨟} g)^{\sharp}$. We must show

$$R(((\mathsf{cross}\, b)^{\flat})^{-1}) = \varphi_{R(\pi^{\ominus} b),L(\pi^{\oplus} b)}(L(((\mathsf{cross}\, b)^{\sharp})^{-1}))\,.$$

It suffices to consider

$$\begin{aligned}&\varphi^{-1}_{(\pi^{\oplus} a),L(\pi^{\oplus} a)}\left((\mathsf{cross}\, b)^{\sharp} \mathbin{⨟} R(((\mathsf{cross}\, b)^{\flat})^{-1})\right)\\ &\quad = \varphi^{-1}_{(\pi^{\oplus} a),L(\pi^{\oplus} a)}\left((\mathsf{cross}\, b)^{\sharp} \mathbin{⨟} \varphi_{R(\pi^{\ominus} b),L(\pi^{\oplus} b)}(L(((\mathsf{cross}\, b)^{\sharp})^{-1}))\right)\end{aligned}$$

by isomorphism of $(\mathsf{cross}\, b)^{\sharp}$ and φ. Both sides are equal to $\mathsf{id}_{\pi^{\oplus} a}$ by naturality of φ and isomorphism $(\mathsf{cross}\, b)^{\sharp}$. □

Definition 5.27 (`envelope_compose`)**.** The envelope duploid-to-be has composition given by discriminating on the polarization choice. That is, let $f : \mathcal{E}[L \dashv R][\![a, b]\!]$ and $g : \mathcal{E}[L \dashv R][\![b, c]\!]$ be morphisms in the envelope duploid-to-be. Then a composite $(f \mathbin{⨟} g) : \mathcal{E}[L \dashv R][\![a, b]\!]$ exists, given by

$$\begin{aligned}(f \mathbin{⨟} g)^{\sharp} &\overset{\text{def}}{\equiv} \left(\pi^{\oplus} a \xrightarrow{f^{\sharp}} R(\pi^{\ominus} b) \xrightarrow{((\mathsf{cross}\, b)^{\sharp})^{-1}} \pi^{\oplus} b \xrightarrow{g^{\sharp}} R(\pi^{\ominus} c)\right) && \text{if there is a } ((\mathsf{cross}\, b)^{\sharp})^{-1}\\ (f \mathbin{⨟} g)^{\flat} &\overset{\text{def}}{\equiv} \left(L(\pi^{\oplus} a) \xrightarrow{f^{\flat}} \pi^{\ominus} b \xrightarrow{((\mathsf{cross}\, b)^{\flat})^{-1}} L(\pi^{\oplus} b) \xrightarrow{g^{\flat}} \pi^{\ominus} c\right) && \text{if there is a } ((\mathsf{cross}\, b)^{\flat})^{-1}\,.\end{aligned}$$

By Theorem 5.26 the choice of polarization is irrelevant, thus the above definition is unique.

Lemma 5.28 (`is_negative_of_envelope_chosen_negative`, …)**.** *Objects $a : \mathcal{E}[L \dashv R]$ in the envelope duploid-to-be with a positive polarization choice are positive, and those with a negative polarization choice are negative.*

Proof. Same as Lemma 5.13. □

Corollary 5.29 (`envelope_preduploid`)**.** *The envelope duploid-to-be $\mathcal{E}[L \dashv R]$ is a preduploid.*

Definition 5.30 (`envelope_negative_shift_data`, …)**.** The envelope duploid-to-be $\mathcal{E}[L \dashv R]$ has polarity shift structure given the same way as the oblique duploid (Definition 5.15).

Lemma 5.31 (`envelope_negative_shift_axioms`, …)**.** *The polarity shift structure of Definition 5.30 satisfies the axioms of polarity shifts.*

Proof. Same as Lemma 5.16. □

Corollary 5.32 (`envelope_duploid`). *The envelope duploid-to-be $\mathcal{E}[L \dashv R]$ is a duploid. Hence, we now call it the* envelope duploid.

5.3. TODO The Equalizing Requirement

Lemma 5.33. *Let $\mathcal{P}$ and $\mathcal{N}$ be univalent categories, and $L \dashv R : \mathcal{P} \to \mathcal{N}$ a fully equalizing adjunction between them. The following categories characterize the subcategories of the envelope duploid $\mathcal{E}[L \dashv R]$.*

- *The positive thunkable category $\mathcal{E}[L \dashv R]^{\oplus}_{t}$ is identified with $\mathcal{P}$.*
- *The negative linear category $\mathcal{E}[L \dashv R]^{\ominus}_{l}$ is identified with $\mathcal{N}$.*
- *The positive subcategory $\mathcal{E}[L \dashv R]^{\oplus}$ is identified with the non-univalent Kleisli category $\mathbf{Kleisli}_{\cancel{\mathsf{univ}}}(\mathcal{P}, RL)$.*
- *The negative subcategory $\mathcal{E}[L \dashv R]^{\ominus}$ is identified with the non-univalent Co-Kleisli category $\mathbf{CoKleisli}_{\cancel{\mathsf{univ}}}(\mathcal{N}, LR) \equiv \mathbf{Kleisli}_{\cancel{\mathsf{univ}}}(\mathcal{N}^{\mathsf{op}}, (LR)^{\mathsf{op}})$.*

6. TODO Related Work

Bibliography

[1] B. Ahrens, K. Kapulkin, and M. Shulman, "Univalent categories and the rezk completion," *Mathematical Structures in Computer Science*, vol. 25, no. 5, pp. 1010–1039, 2015. DOI: 10.1017/S0960129514000486

[2] B. Ahrens, P. R. North, M. Shulman, and D. Tsementzis, *The Univalence Principle* (Memoirs of the American Mathematical Society). 2025, vol. 305. DOI: 10.1090/memo/1541 arXiv: 2102.06275.

[3] A. Church, "A formulation of the simple theory of types," *Journal of Symbolic Logic*, vol. 5, no. 2, pp. 56–68, 1940. DOI: 10.2307/2266170

[4] P. Clairambault and G. Munch-Maccagnoni, "Duploid situations in concurrent games," in *12th Workshop on Games for Logic and Programming Languages (GaLoP XII)*, Juha Kontinen, Marina Lenisa, Uppsala, Sweden, Apr. 2017. [Online]. Available: https://inria.hal.science/hal-01991555

[5] T. Coquand and N. A. Danielsson, "Isomorphism is equality," *Indagationes Mathematicae*, vol. 24, no. 4, pp. 1105–1120, 2013, In memory of N.G. (Dick) de Bruijn (1918–2012), ISSN: 0019-3577. DOI: 10.1016/j.indag.2013.09.002 [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0019357713000694

[6] M. H. Escardó, *A self-contained, brief and complete formulation of voevodsky's univalence axiom*, 2018. arXiv: 1803.02294 [math.LO]. [Online]. Available: https://arxiv.org/abs/1803.02294

[7] C. Führmann, "Direct models of the computational lambda-calculus," *Electronic Notes in Theoretical Computer Science*, vol. 20, pp. 245–292, 1999, MFPS XV, Mathematical Foundations of Progamming Semantics, Fifteenth Conference, ISSN: 1571-0661. DOI: 10.1016/S1571-0661(04)80078-1 [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1571066104800781

[8] D. R. Grayson, "An introduction to univalent foundations for mathematicians," *Bull. Amer. Math. Soc.*, vol. 55, pp. 427–450, 2018, ISSN: 1088-9485. DOI: 10.1090/bull/1616

[9] D. R. Grayson et al., *Unimath/unimath: V20250923*, version v20250923, Sep. 2025. DOI: 10.5281/zenodo.17186647

[10] M. Hofmann and T. Streicher, "The groupoid model refutes uniqueness of identity proofs," in *Proceedings Ninth Annual IEEE Symposium on Logic in Computer Science*, 1994, pp. 208–212. DOI: 10.1109/LICS.1994.316071

[11] C. Kapulkin, P. L. Lumsdaine, and V. Voevodsky, *Univalence in simplicial sets*, 2018. arXiv: 1203.2553 [math.AT]. [Online]. Available: https://arxiv.org/abs/1203.2553

[12] K. Kapulkin and P. L. Lumsdaine, "The simplicial model of univalent foundations (after voevodsky)," *Journal of the European Mathematical Society*, vol. 23, no. 6, pp. 2071–2126, Mar. 2021, ISSN: 1435-9863. DOI: 10.4171/jems/1050

[13] J. Lambek and P. J. Scott, *Introduction to higher order categorical logic*. USA: Cambridge University Press, 1986, ISBN: 0521246652.

[14] P. B. Levy, "Adjunction models for call-by-push-value with stacks," *Electronic Notes in Theoretical Computer Science*, vol. 69, pp. 248–271, 2003, CTCS'02, Category Theory and Computer Science, ISSN: 1571-0661. DOI: 10.1016/S1571-0661(04)80568-1

[15] P. B. Levy, "Call-by-push-value: A subsuming paradigm," in *Typed Lambda Calculi and Applications*, J.-Y. Girard, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 228–243, ISBN: 978-3-540-48959-7.

[16] P. B. Levy, "Contextual isomorphisms," *SIGPLAN Not.*, vol. 52, no. 1, pp. 400–414, Jan. 2017, ISSN: 0362-1340. DOI: 10.1145/3093333.3009898

[17] É. Mangel, P.-A. Melliès, and G. Munch-Maccagnoni, "Classical notions of computation and the hasegawa-thielecke theorem," *Proc. ACM Program. Lang.*, vol. 10, no. POPL, Jan. 2026. DOI: 10.1145/3776715

[18] P. Martin-Löf, "Constructive mathematics and computer programming," in *Logic, methodology and philosophy of science, VI (Hannover, 1979)*, ser. Stud. Logic Found. Math. Vol. 104, North-Holland, Amsterdam, 1982, pp. 153–175. DOI: 10.1016/S0049-237X(09)70189-2

[19] E. Moggi, “Computational lambda-calculus and monads,” in *[1989] Proceedings. Fourth Annual Symposium on Logic in Computer Science*, 1989, pp. 14–23. DOI: 10.1109/LICS.1989.39155

[20] E. Moggi, “Notions of computation and monads,” *Information and Computation*, vol. 93, no. 1, pp. 55–92, 1991, Selections from 1989 IEEE Symposium on Logic in Computer Science, ISSN: 0890-5401. DOI: 10.1016/0890-5401(91)90052-4

[21] G. Munch-Maccagnoni, “Models of a non-associative composition,” in *Foundations of Software Science and Computation Structures*, A. Muscholl, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 396–410, ISBN: 978-3-642-54830-7. DOI: 10.1007/978-3-642-54830-7_26

[22] G. Munch-Maccagnoni, “Syntax and Models of a non-Associative Composition of Programs and Proofs,” Theses, Université Paris-Diderot - Paris VII, Dec. 2013. [Online]. Available: https://theses.hal.science/tel-00918642

[23] E. Rijke, *Introduction to Homotopy Type Theory* (Cambridge Studies in Advanced Mathematics). Cambridge University Press, 2025.

[24] T. Univalent Foundations Program, *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study, 2013. [Online]. Available: https://homotopytypetheory.org/book

Index

A. Counterexample Unital Magmoids

In this section I include some small examples of unital magmoids, to show that certain properties obtained in categories and preduploids do not generalize to unital magmoids.

Example A.1. There is a unital magmoid **Lini** (`lini`) that has a linear-and-thunkable isomorphism $p : b \cong_{lt} d$ which is not intermediate. **Lini** is generated by four objects: $a, b, c, d : \mathbf{Lini}$ and the morphisms

$$a \xrightarrow{f} b \xrightarrow{g} c$$
$$b \xrightarrow{p}_{lt} d \xrightarrow{q}_{lt} b$$

such that $p \mathbin{⨾} q = \mathsf{id}_b$ and $q \mathbin{⨾} p = \mathsf{id}_a$. In other words, p and q form a linear-and-thunkable isomorphism $p : b \cong_{lt} d$ with inverse $p^{-1} = q$. This is depicted in the diagram

$$\begin{tikzcd} a \arrow[r, "f"] & b \arrow[r, "g"] \arrow[d, bend right, "p"'] & c \\ & d \arrow[u, bend right, "q"'] & \end{tikzcd} .$$

The unital magmoid **Lini** can be described explicitly by setting

$$\mathbf{Lini}[\![a, c]\!] \overset{\text{def}}{\equiv} \{[f \mathbin{;} g], [[f \mathbin{;} p] \mathbin{;} [q \mathbin{;} g]]\} \quad (\text{or } \texttt{bool}).$$

All other hom-sets are either `unit` or `empty`, as needed to satisfy the existence of f, g, p, q, identities, and composites. The element $[f \mathbin{;} g] : \mathbf{Lini}[\![a, c]\!]$ is just notation for, say `false` : `bool`. Composition is uniquely determined by unitality and the equations

$$f \mathbin{⨾} g \overset{\text{def}}{\equiv} [f \mathbin{;} g] \quad \text{and} \quad (f \mathbin{⨾} p) \mathbin{⨾} (q \mathbin{⨾} g) \overset{\text{def}}{\equiv} [[f \mathbin{;} p] \mathbin{;} [q \mathbin{;} g]] .$$

It may be computed that p is a linear-and-thunkable isomorphism $b \cong_{lt} d$. Moreover, p does not satisfy $(f \mathbin{⨾} p) \mathbin{⨾} (q \mathbin{⨾} g) = f \mathbin{⨾} g$, and (therefore) is not intermediate.

Example A.2. There is a finite unital magmoid **Inu** (inu) with a morphism $f\colon a \to b$ that has two inverses. **Inu** is generated by the three morphisms

$$\begin{tikzcd} a \arrow[r, "f"] & b \arrow[l, bend right, "p"'] \arrow[l, bend left, "q"] \end{tikzcd}$$

and the equations

$$f \fatsemi p = \mathsf{id}_a = f \fatsemi q$$
$$p \fatsemi f = \mathsf{id}_b = q \fatsemi f\,.$$

This may be constructed explicitly by setting

$$\mathbf{Inu}[\![b,a]\!] \overset{\text{def}}{\equiv} \{p,q\} \quad (\text{or } \texttt{bool}),$$

and all other hom-sets to `unit`.

Additionally, it can be computed that f is linear and thunkable but not intermediate, and that p and q are both intermediate but not linear or thunkable.