Monoids up to Coherent Homotopy in Two-Level Type Theory

Hugo Moeneclaey,
ENS Paris-Saclay
hmoenecl@ens-paris-saclay.fr

Under the supervision of:

Peter LeFanu Lumsdaine,
Stockholm University
p.l.lumsdaine@math.su.se

October 2018 - March 2019

Abstract

We present a formalization of monoids up to coherent homotopy in Agda. In order to achieve this we postulate a structure of type theory with two equalities with a notion of fibrant type. Then we build an operad ∞Mon and define monoids up to coherent homotopy as its algebras. We prove that this notion is invariant under equivalences between fibrant types, and that loop spaces are such monoids.

Contents

1 Introduction **2**

2 Two-level type theory **5**
2.1 Introduction to two-level type theory 5
2.2 Definition of the homotopical structure on types 6
2.3 Fibrations and equivalences 8
2.4 Lifting properties 10
2.5 The cocylinder factorization for fibrant types 11

3 Operads and their algebras **11**
3.1 Introduction to operads 11
3.2 A type theoretic definition of operads 12
3.3 Algebras for an operad 16
3.4 Toward an operad for monoids up to coherent homotopy 17

4 Cofibrant operads **17**
4.1 Homotopical structure on operads 18
4.2 The cocylinder factorisation for operads 18
4.3 Endomorphism operad of a map between types 19
4.4 The case of trivial fibrations 20
4.5 Invariance under equivalence of algebras for a cofibrant operad . 21

5 The operad ∞Mon **21**
5.1 Non-functorial operads with partial compositions 22
5.2 Definition of labelled trees 25
5.3 A rewriting on labelled trees 27
5.4 Definition of ∞Mon 29
5.5 Sections of strongly contractible morphisms over ∞Mon 30

6 Properties of ∞Mon **33**
6.1 Cofibrations and pseudo-cofibrations 33
6.2 ∞Mon is cofibrant 34
6.3 Loop spaces are ∞Mon-algebras 35

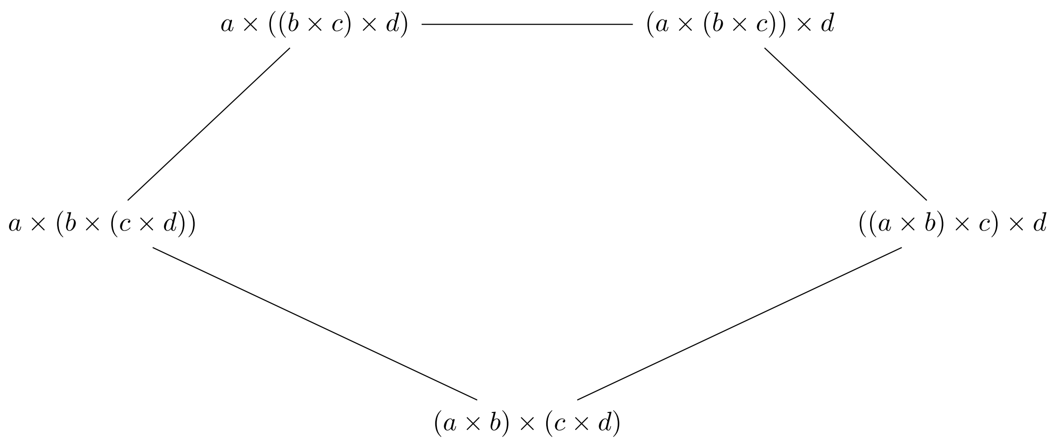
7 Conclusion **38**

1 Introduction

Martin Löf's Type Theory [11] is a foundational system for mathematics which replaces the primitive notion of sets by the notion of types. It has a computational interpretation in the sense that a program can be extracted from any proof in type theory. Most modern proof assistant (Coq, Agda, Lean, ...) implement a variant of it. In this system given any type X and any $x, y : X$, one can form the identity type $\mathrm{Id}_X(x, y)$, which is inhabited if and only if x and y are equal. But this construction can be iterated, for example given $p, q : \mathrm{Id}_X(x, y)$, one can form the type $\mathrm{Id}_{\mathrm{Id}_X(x,y)}(p, q)$, and so on. In Extensional Type Theory [12, 13], all iterated identity types are assumed to be inhabited, so that equalities behave as in informal mathematics. This property is called the *uniqueness of identity proofs*.

Homotopy Type Theory [15] is a variant of type theory in which types can be interpreted as spaces up to homotopy equivalences, i.e. up to continuous deformations. An inhabitant of an identity type is then interpreted as a path in the relevant space. The main feature of Homotopy Type Theory is the axiom of *univalence*, which roughly states that isomorphic types are equal. Univalence contradicts uniqueness of identity proofs, but it implies that types behave like spaces, so that for example some homotopy groups of spheres have been computed from this axiom by Brunerie, Licata and Shulman [6, 9, 10].

One could try to define monoids in Homotopy Type Theory as the data of a type A together with a binary operation $\_ \times \_ : A \to A \to A$ and a unit $1 : A$ such that for all $a, b, c : A$ we have a path from $(a \times b) \times c$ to $a \times (b \times c)$, and similarly for the unit laws. But this definition is not satisfying because such types do not behave as the usual monoids. In fact when we consider $a, b, c, d : A$ we have the following diagram:



where lines are paths induced by the associativity of $\_ \times \_$. In our context we should require a filling of the diagram, that is a suitable path between paths, so that the multiplications of 4 elements form a contractible space (i.e. a space that can be continuously deformed to a point). Similarly for 5 elements one should require a path between paths between paths, and so on. There should be similar higher homotopies for the unit laws. Such monoids are called *monoids up to coherent homotopy* or equivalently *∞-monoids*. Our main goal is to define them in type theory.

Monoids up to coherent homotopy are well-known in algebraic topology. It is possible to define them using *operads*, as introduced by May [14] (other approaches are possible e.g. the one by Boardman and Vogt [4]). An operad is a well-behaved algebraic theory, and we can define its algebras. Operads can be generalised to other setting than set, so that for example topological operads have spaces of operations, and their algebras are spaces. An important result is that there exists a topological operad whose algebras are precisely monoids up to coherent homotopy.

For any space X and point x in X, we can define its *loop space* as the space of path in X from x to x. An ∞-monoid is called *group-like* if the induced structure of monoid on its connected components is in fact a group structure. Now we use this vocabulary to list the main results in the classical theory of ∞-monoids. We give precise references to the book by Boardman and Vogt [4], which develops a general theory of structures up to coherent homotopy.

1. Loop spaces are ∞-monoids, with the concatenation of paths as multiplication [4, Prp 3.25]. They are group-like because any path has an inverse up to homotopy induced by traversing the path backward.

2. If X is an ∞-monoid and f is a homotopy equivalence between X and a type Y, then Y has an induced structure of ∞-monoid [4, Thm 4.37].

3. If X is an ∞-monoid, then its structure is induced by a topological monoid Y and a homotopy equivalence between X and Y [4, Thm 4.37].

4. If X is a group-like ∞-monoid, then its structure is induced by a loop space Y and a homotopy equivalence between X and Y [4, Thm 6.30].

So to summarize we know that ∞-monoids are precisely spaces homotopically equivalent to a topological monoid, and group-like ∞-monoids are precisely spaces homotopically equivalent to a loop space. In this report we present a formalization of 1 and 2.

At this point it should be noted that the correspondence between goup-like ∞-monoids and loop spaces suggests to define ∞-*groups* in Homotopy Type Theory as spaces equivalent to a loop space. This is the approach taken by Buchholtz, van Doorn and Rijke [7].

The straightforward approach to our problem is to define operads in Homotopy Type Theory. But operads themselves obey a form of associativity, so a definition of operads using paths as witnesses for this equality should itself be stated up to coherent homotopy. To get out of this circularity we use a type theory with two equalities, similar to Voevodsky's Homotopy Type System [16]. This method, called two-level type theory, has already been used to solve similar problems by Annenkov, Capriotti and Kraus [1], and by Boulier and Tabareau [5]. So given a type X and $x, y : X$, one has two types: a type $x \equiv y$ of witnesses that x and y are equal, which obeys uniqueness of identity proofs, and a type $x \rightsquigarrow y$ of paths between x and y, which is meant to obey univalence. We call $\_ \equiv \_$ the strict equality. Then asscociativity for operads is stated using the strict equality, and associativity for monoids up to coherent homotopy is stated using paths.

In Section 2 we present the method from [1] to make both equalities cohabit and we give the details of our variant. In Section 3 we give a type-theoretic definition of operads. In Section 4 we give an auxiliary result which will be used to show our notion of monoids up to coherent homotopy invariant under equivalences between fibrant types. In Section 5 we give a definition of the operad ∞Mon of monoids up to coherent homotopy and show its key property. In Section 6 we show how to deduce from this key property that ∞Mon-algebras are invariant under equivalences between fibrant types, and that loop spaces are ∞Mon-algebras.

Remark. *We want to give a computer-checked account of the results presented here. To do this we use the Agda proof assistant. Our formalisation can be found at* `https://github.com/hmoeneclaey/operads`.

All results in Sections 2, 3, 4 and 6 have been fully formalized, but some combinatorial results from Section 5 are missing.

Notations and conventions. *Assume given a type X and a family of types P indexed by X. The type of maps associating to $x : X$ and element in $P(x)$ is* *denoted by:*

$$(x : X) \to P(x)$$

The type of $x : X$ *together with* $p : P(x)$ *is denoted by* $\Sigma_X P$ *or:*

$$\Sigma(x : X).\, P(x)$$

An element in $\Sigma_X P$ *is sometimes denoted* (x, p)*.*

We write the applications of functions in the mathematical way, so that $f(x)$ *is* f *applied to* x*. If* f *takes two arguments, we write* $f(x, y)$ *instead of* $f(x)(y)$*.*

We write in an informal type-theoretic style, so that when we say "for all $x : X$ *we have* $P(x)$*" we mean that* $(x : X) \to P(x)$ *is inhabited. Similarly when we say "there exists* $x : X$ *such that* $P(x)$*" we mean that* $\Sigma(x : X).\, P(x)$ *is inhabited.*

If we refer to a mathematical concept which we did not explicitly define, we mean its translation using the strict equality $\_ \equiv \_$*. So for example the pullback of* $f : X \to Z$ *by* $g : Y \to Z$ *is defined as the projection from*

$$\Sigma(x : X)(y : Y).\, f(x) \equiv g(y)$$

to Y*.*

2 Two-level type theory

We present a variant of the two-level type theory defined by Annenkov, Capriotti and Kraus [1]. It is inspired by Cubical Type Theory [8].

2.1 Introduction to two-level type theory

The main purpose of two-level type theory is to allow internalization of arguments involving the usual equality to Homotopy Type Theory. In order to do so a strict equality is introduced. On the other hand we still need a homotopical equality, interpreted as the type of paths between two points. How is it possible to have both equalities cohabiting harmoniously?

We use intuitions from the theory of model categories. A model category has three classes of maps called fibrations, cofibrations and weak equivalences. They are assumed to obey a certain number of axioms, which we do not list. From a model category $\mathcal{C}$, it is possible to build a relatively explicit description of its homotopy category, which is the category $\mathcal{C}$ where weak equivalences have been formally inverted. A model category $\mathcal{C}$ can be viewed as a nice presentation of its homotopy category.

In any model category we can define the classes of *fibrant* and *cofibrant* objects. Usually maps out of cofibrant objects to fibrant objects are well-behaved homotopically. The key intuition we use is that it is more convenient to have non-fibrant and non-cofibrant objects at our disposal, even if we are ultimately not interested in them. In our case we will axiomatize a universe of fibrant objects, which is meant to satisfy univalence, sitting in a larger universe which does not.

The reader should note that although we use these intuitions, universes in two-level type theory do not carry the structure of a model category. In fact fibrant replacements contradict the univalence of the fibrant universes [1].

2.2 Definition of the homotopical structure on types

We use Agda for implementation, so we have by default a (non-cumulative) hierarchy of universes Set_k for k an external natural number. We will often write Set without index when our construction is valid in any Set_k.

We use the default equality of Agda as strict equality. So for X a type and $x, y : X$, we have a strict equality type denoted $x \equiv y$, with the usual elimination principle. It is assumed to obey axiom K and function extensionnality, so that it behaves more or less as the usual mathematical equality. Axiom K is stronger than uniqueness of identity proofs, but we use it only because it is implemented by default in Agda. Uniqueness of identity proofs is sufficient to formalize all our results.

First a preliminary definition.

Definition 2.1. *We say two types X and Y are isomorphic when there exists $f : X \to Y$ and $g : Y \to X$ such that:*

- *For all $x : X$, we have $g(f(x)) \equiv x$.*
- *For all $y : Y$, we have $f(g(y)) \equiv y$.*

Note that isomorphic types do not need to belong to the same universe.

Now we define a hierarchy of fibrant universes.

Definition 2.2. *A hierarchy of fibrant universes in $(\mathrm{Set}_k)_{k\in\mathbb{N}}$ is a family of predicates $\mathrm{Fib} : \mathrm{Set}_k \to \mathrm{Set}_k$ such that (when $\mathrm{Fib}(X)$ is inhabited we say that X is fibrant):*

- *$\top$ is fibrant, where $\top$ is a terminal object in Set.*
- *If X is fibrant and $P : X \to \mathrm{Set}$ is a family of fibrant types then*

$$\Sigma(x : X).\, P(x)$$

  and

$$(x : X) \to P(x)$$

  are fibrant.
- *If X is fibrant and X is isomorphic to Y, then Y is fibrant.*

For now we still do not have a notion of homotopical equality. A possible option is to axiomatize it as a constructor $\mathrm{Path} : (A : \mathrm{Set}) \to A \to A \to \mathrm{Set}$, as in [1] or [5]. Instead we will axiomatize an interval $\mathbb{I}$, inspired by cubical type theory. The intuition is that $\mathbb{I}$ is the simplicial interval Δ^1, or similarly the real segment $[0, 1]$.

Definition 2.3. *An interval is a type* $\mathbb{I}$ *in* Set_0 *together with two inhabitants* $0 : \mathbb{I}$ *and* $1 : \mathbb{I}$ *such that:*

- *If* $P : \mathbb{I} \to \mathrm{Set}$ *is a family of fibrant types,* $x : P(0)$ *and* $y : P(1)$*, then the type of functions* $f : (i : \mathbb{I}) \to P(i)$ *such that* $f(0) \equiv x$ *and* $f(1) \equiv y$ *is fibrant.*

- *If* X *is fibrant and* $C : (\mathbb{I} \to X) \to \mathrm{Set}$ *is a family of fibrant type, then given* $d : (x : X) \to C(\lambda i.x)$ *we have an inhabitant* $J(d) : (p : \mathbb{I} \to X) \to C(p)$*. Moreover* $J(d)(\lambda i.x)$ *is definitionally equal to* $d(x)$*.*

We now give the definition of path types.

Definition 2.4. *For* X *a type and* $x, y : X$*, we define the type of paths from* x *to* y *as the type of functions* $f : \mathbb{I} \to X$ *such that* $f(0) \equiv x$ *and* $f(1) \equiv y$*. It is denoted by* $x \rightsquigarrow y$*.*

For $x : X$*, we denote by* hrefl_x *the inhabitant of* $x \rightsquigarrow x$ *with underlying function* $\lambda i.x$*.*

A key point in the definition of $\mathbb{I}$ is that C is assumed to be a family of *fibrant* types. This will guarantee that $x \rightsquigarrow y$ does not imply $x \equiv y$, because strict equality types are not fibrant.

Remarks. *At this point a few remarks should be made.*

- *If* X *is fibrant and* $x, y : X$*, then* $x \rightsquigarrow y$ *is fibrant. In fact we included strict equalities in the first axiom for* $\mathbb{I}$ *in order to guarantee that.*

- *The type* $\mathbb{I}$ *is not assumed fibrant.*

- *With our definition we can define paths in non-fibrant types. This is in contrast with [1]. But these non-fibrant path types are very poorly behaved, for example a path* $x \rightsquigarrow y$ *does not imply a path* $y \rightsquigarrow x$*. This does justify the oriented notation.*

- *The elimination principle for paths is stated only for paths in fibrant types. This limitation can probably be removed, the important point being that we only eliminate into fibrant types.*

- *We have assumed definitional computation rule for* J*. We did this for convenience, but everything could be done with the strict equalities:*

  $$J(d)(\lambda i.x) \equiv d(x)$$

  We use the Agda rewriting feature, so than we can postulate definitional equalities.

Moreover we will assume that $\mathbb{I}$ has two connections $\_ \vee \_ : \mathbb{I} \to \mathbb{I} \to \mathbb{I}$ and $\_ \wedge \_ : \mathbb{I} \to \mathbb{I} \to \mathbb{I}$. If $\mathbb{I}$ is interpreted as the real segment $[0, 1]$, then $i \vee j$ is interpreted as the maximum of i and j and $i \wedge j$ as their minimum.

Definition 2.5. *Connections for an interval* $\mathbb{I}$ *are functions* $\_\vee\_ : \mathbb{I} \to \mathbb{I} \to \mathbb{I}$ *and* $\_\wedge\_ : \mathbb{I} \to \mathbb{I} \to \mathbb{I}$ *such that for any* $i : \mathbb{I}$ *we have:*

- $i \vee 1 \equiv 1 \vee i \equiv 1$ *and* $i \vee 0 \equiv 0 \vee i \equiv i$.
- $i \wedge 1 \equiv 1 \wedge i \equiv i$ *and* $i \wedge 0 \equiv 0 \wedge i \equiv 0$.

Moreover we require $\_\vee\_$ *associative, and* $\_\wedge\_$ *distributing on the left of* $\_\vee\_$*, i.e. for any* $i, j, k : \mathbb{I}$ *we have:*

- $i \vee (j \vee k) \equiv (i \vee j) \vee k$.
- $i \wedge (j \vee k) \equiv (i \wedge j) \vee (i \wedge k)$.

We have used as little as we could on the connections, explaining this somewhat unnatural list.

Assumption 2.6. *We assume given a hierarchy of fibrant universes and an interval with connections. We call them collectively the homotopical structure of the universes.*

2.3 Fibrations and equivalences

Now we define a some homotopical concepts.

Definition 2.7. *A map* $f : X \to Y$ *is said to be an equivalence if there exists* $g_1, g_2 : Y \to X$ *such that:*

- *For all* $x : X$ *we have* $x \rightsquigarrow g_1(f(x))$.
- *For all* $y : Y$ *we have* $y \rightsquigarrow f(g_2(y))$.

We use different left and right inverses so that the type of witnesses that a map is an equivalence is contractible. This definition is well-behaved only for maps between fibrant types. For example, the composite of two equivalences between non-fibrant types is not necessarily an equivalence.

Definition 2.8. *The fibre of a map* $f : X \to Y$ *over* $y : Y$ *is the type:*

$$\Sigma(x : X).\, f(x) \equiv y$$

It is denoted by $\mathrm{fibre}_f(y)$.

It is straightforward to show that any function $f : X \to Y$ is isomorphic to the projection:

$$\Sigma(y : Y).\, \mathrm{fibre}_f(y) \to Y.$$

Similarly the fibre of the projection:

$$\Sigma(y : Y).\, P(y) \to Y$$

over $y : Y$ is isomorphic to $P(y)$. This gives a correspondence between maps to Y and families $P : Y \to \mathrm{Set}$.

Definition 2.9. *A map $f : X \to Y$ is called a fibration if its fibres are fibrant. The type Y is then called the base of the fibration f.*

So fibrations with base Y corresponds to families of fibrant types over Y.

Definition 2.10. *A type X is called contractible if there is $x : X$ such that for all $y : X$ we have $x \rightsquigarrow y$.*

Once again, this notion is well-behaved only for fibrant types.

Definition 2.11. *A fibration $f : X \to Y$ is said trivial if its fibres are contractible.*

Now we state some basic properties of these notions. The proofs can be found in the formalization. We will use these lemmas without references in the rest of the text.

Lemma 2.12. *Equivalences between fibrant types obey the two-out-of-three property.* □

Lemma 2.13. *A trivial fibration is an equivalence.* □

Lemma 2.14. *A map between fibrant types is a trivial fibration if and only if it is a fibration and an equivalence.* □

Lemma 2.15. *The pullback of a fibration (respectively trivial fibration) is a fibration (respectively trivial fibration).* □

Lemma 2.16. *Assume Y fibrant and $f : X \to Y$ a fibration. Then X is fibrant.* □

The next lemma is often called the contractibility of singletons.

Lemma 2.17. *Assume X is fibrant and $x : X$. Then the type:*

$$\Sigma(y : X).\, x \rightsquigarrow y$$

is contractible. □

The next lemma shows that our definition of contractibility is reasonable, because it implies trivial higher equalities on the type.

Lemma 2.18. *Assume given X is a contractible fibrant type and $x, y : X$. Then $x \rightsquigarrow y$ is contractible.* □

Lemma 2.19. *Assume given $f : X \to Y$ an equivalence between fibrant types. Then for any $y : Y$, the type:*

$$\Sigma(x : X).\, f(x) \rightsquigarrow y$$

is contractible. □

The last lemma is a bit technical, an informal proof can be found in [15, Thm 4.4.5].

2.4 Lifting properties

In this section we introduce some consideration on lifting properties. They are basic tools in the theory of model categories, and we will use them to define suitable notions of cofibrations.

Definition 2.20. *Assume given two maps* $u : A \to B$ *and* $p : X \to Y$*. We says that* u *has the left lifting property against* p *(or equivalently that* p *has the right lifting property against* u*) if for any commutative square:*

$$\begin{array}{ccc} A & \longrightarrow & X \\ {\scriptstyle u}\big\downarrow & \nearrow & \big\downarrow{\scriptstyle p} \\ B & \longrightarrow & Y \end{array}$$

there exists a dotted arrow making the two triangles commute.

If $u : A \to B$ has the left lifting property against $p : X \to Y$, then we say that any local section of p over A can be extended to a local section of p over B. The next lemma is straightforward to prove.

Lemma 2.21. *Assume given two maps* $u : A \to B$ *and* $p : X \to Y$*. If* p *has the right lifting property against* u*, then so does any pullback of* p*.* $\square$

Now we define two useful constructions on maps.

Definition 2.22. *Assume given two maps* $u : A \to B$ *and* $p : X \to Y$*. Then the induced map of type:*

$$(B \to X) \longrightarrow (B \to Y) \times_{A \to Y} (A \to X)$$

is called the pullback-exponential of u *and* p *and is denoted* $\langle u/p \rangle$*.*

Definition 2.23. *Assume given two maps* $u : A \to B$ *and* $v : A' \to B'$*. Then the induced map of type:*

$$A \times B' \coprod_{A \times A'} B \times A' \longrightarrow (B \times B')$$

is called the pushout-product of u *and* v *and is denoted* $u \,\square\, v$*.*

Note that the pushout in the last lemma is formalized as a strict quotient. Now we justify the interest of the pullback-exponential construction.

Lemma 2.24. *Assume given two maps* u *and* p*. Then* $\langle u/p \rangle$ *has a section if and only if* u *has the left lifting property against* p*.* $\square$

We say that two maps are isomorphic if there are isomorphisms between their codomains and domains such that the induced square commutes. We will use without mention the fact that all the classes of maps defined in this report are stable under isomorphisms.

The next lemma is proved by a straightforward but lengthy computation, and shows the duality between pullback-exponentials and pushout-products.

Lemma 2.25. *Assume given maps* u*,* v *and* p*. Then* $\langle u \,\square\, v/p \rangle$ *is isomorphic to* $\langle u/\langle v/p \rangle \rangle$*.* $\square$

2.5 The cocylinder factorization for fibrant types

We present a useful factorization of maps.

Lemma 2.26. *Assume given a map f between fibrant types. Then it factors as a section of a trivial fibration followed by a fibration. Moreover if f is an equivalence then the fibration is trivial.*

Proof. Assume given a map between fibrant types $f : X \to Y$. Then we have the factorisation of f through:

$$\Sigma(x : X)(y : Y).\, f(x) \rightsquigarrow y$$

as $\lambda x.\,(x, f(x), \mathrm{hrefl}_{f(x)})$ followed by the projection to Y.

The map $\lambda x.\,(x, f(x), \mathrm{hrefl}_{f(x)})$ is a section of the projection:

$$\Big(\Sigma(x : X)(y : Y).\, f(x) \rightsquigarrow y\Big) \longrightarrow X$$

whose fibre over $x : X$ is isomorphic to:

$$\Sigma(y : Y).\, f(x) \rightsquigarrow y$$

but this type is clearly fibrant and it is contractible by Lemma 2.17. Therefore $\lambda x.\,(x, f(x), \mathrm{hrefl}_{f(x)})$ is a section of a trivial fibration.

The fibre of the projection:

$$\Big(\Sigma(x : X)(y : Y).\, f(x) \rightsquigarrow y\Big) \longrightarrow Y$$

over $y : Y$ is isomorphic to:

$$\Sigma(x : X).\, f(x) \rightsquigarrow y$$

This type is fibrant and if f is an equivalence then it is contractible by Lemma 2.19. □

3 Operads and their algebras

In this section we give a type-theoretic definition of operads. We do not use the homotopical structure of the universes, so that our definition makes sense in any type theory obeying uniqueness of identity proofs and function extensionnality.

3.1 Introduction to operads

We give an overview of operads from the classical point of view (i.e. in a set-theoretic foundations).

Intuitively operads are certain well-behaved algebraic theories. These theories are called linear because they can be defined by equations with variables occurring exactly once on each side.

First we present non-symmetric operads in sets. In this case an operad is a family of sets $(\mathcal{P}(n))_{n\in\mathbb{N}}$, indexed by the natural numbers. Intuitively $\mathcal{P}(n)$ is the set of n-ary operations in the represented algebraic theory. These sets should be equipped with:

- An element in $\mathcal{P}(1)$ called the identity (intuitively the operation $x \mapsto x$).
- A composition of operations denoted γ, which, given $c \in \mathcal{P}(n)$ and $d_i \in \mathcal{P}(k_i)$ for $1 \leq i \leq n$, outputs an element $\gamma(c; d_1, \cdots, c_n)$ in $\mathcal{P}(\Sigma_{i=1}^{n} k_i)$. Intuitively, given ψ an n-ary operation and given φ_i a k_i-operation for $1 \leq i \leq n$, their composition $\gamma(\psi; \varphi_1, \cdots, \varphi_n)$ corresponds to:

$$(x_{i,j})_{1\leq i\leq n, 1\leq j\leq k_i} \mapsto \psi(\varphi_1(x_{1,1}, \cdots, x_{1,k_1}), \cdots, \varphi_n(x_{n,1}, \cdots, x_{n,k_n}))$$

Moreover this structure should obey some axioms suggested by the intuitive interpretation. Note that if for example $\psi \in \mathcal{P}(2)$ is interpreted as $(x, y) \mapsto \psi(x, y)$, there is no way to consider $(x, y) \mapsto \psi(y, x)$, this is justify the name of non-symmetric operads: the order of the inputs is fixed. In this report we only consider non-symmetric operads, so we call them operads.

For any operad $\mathcal{P}$ we have a notion of $\mathcal{P}$-algebra structure on a set X. Such a structure consists of an actual operation $X^n \to X$ for each element in $\mathcal{P}(n)$, respecting identity and compositions in some sense. We say that $\mathcal{P}$ acts on X.

A first example is the operad Mon corresponding to the theory of monoids. Then Mon(n) a singleton for any $n \in \mathbb{N}$, because precisely one n-ary operation preserving the order of its inputs can be derived from the axioms of monoids. It is equipped with the obvious compositions and identity. An algebra for this operad is a monoid in the usual sense.

An attractive feature of the definition of operad is that it can be adapted straightforwardly to any monoidal category other than the category of sets by requiring that $\mathcal{P}$ is a family of objects in the category, and that all functions are morphisms. Operads in the category of topological spaces can be used to define monoids up to coherent homotopy. Our goal is to adapt this idea with types instead of topological spaces, so that we can define monoids up to coherent homotopy in type theory. So we first need to define operads in two-level type theory, and then we will give an operad which we call ∞Mon, and give some results supporting the claim that ∞Mon-algebras are precisely monoids up to coherent homotopy.

3.2 A type theoretic definition of operads

The main problem one encounter when defining operads in type theory as functions $\mathcal{P} : \mathbb{N} \to \mathrm{Set}$ with extra structure, is that the axioms of operads are not well-typed, so transport along strict equalities in $\mathbb{N}$ needs to be used. Instead we define operads as functors from the groupoid of finite totally ordered sets to the category of types. So we replace transport by the action of a functor on isomorphisms of finite totally ordered sets, on which we have a more direct control.

Definition 3.1. *For any* $n : \mathbb{N}$*, we define* $\mathrm{Fin}(n)$ *as the type of* $k : \mathbb{N}$ *such that* $k < n$*.*

In the implementation we give an inductive definition of the family:

$$\mathrm{Fin} : \mathbb{N} \to \mathrm{Set}_0$$

In fact we only need some canonical finite totally ordered sets, no matter how they are defined. Then the most convenient way to define a finite totally ordered set is as a type A together with an isomorphism between A and a canonical finite set. The order on A is induced by the order on the canonical finite set.

Definition 3.2. *The groupoid* FOSet *of (small) finite totally ordered sets is defined as follows:*

- *An object consists of* $A : \mathrm{Set}_0$ *together with* $n : \mathbb{N}$ *and an isomorphism from* A *to* $\mathrm{Fin}(n)$*. This induced an order on* A*.*
- *A morphism is an order-preserving isomorphism between the underlying types.*

Note that there exists at most one morphism between two objects in FOSet.

We will often omit the coercion of an object in FOSet to its underlying type, and of a morphism in FOSet to its underlying function. Moreover for $A, A' : \mathrm{FOSet}$ we will denote by $A \cong A'$ the type of isomorphisms from A to A' in FOSet.

The next few lemmas are easy to prove, but are necessary to state our definition of operads.

Lemma 3.3. *Assume* $A : \mathrm{FOSet}$ *and* $B : A \to \mathrm{FOSet}$*. Then* $\Sigma_A B$ *is in* FOSet*.*

In the proof of the last lemma we need to construct an order on $\Sigma_A B$. This is done by saying that for any $a, a' : A$ and $b : B(a)$ and $b' : B(a')$, we have that $(a, b) < (a', b')$ if and only if $a < a'$ or $a \equiv a'$ and $b < b'$.

Lemma 3.4. *Assume given:*

- $A, A' : \mathrm{FOSet}$.
- $B : A \to \mathrm{FOSet}$ *and* $B' : A' \to \mathrm{FOSet}$.
- $f : A \cong A'$.
- $F : (a : A) \to B(a) \cong B'(f(a))$.

Then we have:

$$(f, F) : \Sigma_A B \cong \Sigma_{A'} B'$$

Lemma 3.5. *Assume given* $A : \mathrm{FOSet}$*, then we have:*

$$\eta'_A : A \cong A \times \mathrm{Fin}(1)$$

Lemma 3.6. *Assume given* $B : \mathrm{Fin}(1) \to \mathrm{FOSet}$*, then we have:*

$$\eta_B : B(0) \cong \Sigma_{\mathrm{Fin}(1)} B$$

Lemma 3.7. *Assume given:*

- $A : \mathrm{FOSet}$.
- $B : A \to \mathrm{FOSet}$.
- $C : \Sigma_A B \to \mathrm{FOSet}$

Then we have:

$$\psi_{A,B,C} : \Sigma(a : A)(b : B(a)).\, C(a, b) \cong \Sigma_{\Sigma_A B} C$$

We are now ready to state the main definition of this section.

Definition 3.8. *An operad consists of:*

- *A functor* $\mathcal{P}$ *from* FOSet *to* Set.
- *An element* $\mathrm{id}_{\mathcal{P}} : \mathcal{P}(\mathrm{Fin}(1))$.
- *Given* $A : \mathrm{FOSet}$ *and* $B : A \to \mathrm{FOSet}$*, a function:*

$$\gamma_{\mathcal{P}} : \mathcal{P}(A) \to \Big((a : A) \to \mathcal{P}(B(a))\Big) \to \mathcal{P}(\Sigma_A B)$$

such that:

1. *Assume given:*
   - $A : \mathrm{FOSet}$ *and* $B, B' : A \to \mathrm{FOSet}$.
   - $F : (a : A) \to B(a) \cong B'(a)$.
   - $c : \mathcal{P}(A)$ *and* $d : (a : A) \to \mathcal{P}(B(a))$.

   Then we have:

$$\mathcal{P}(\lambda x.x, F)(\gamma_{\mathcal{P}}(c, d)) \equiv \gamma_{\mathcal{P}}\Big(c, \lambda a.\, \mathcal{P}(F(a))(d(a))\Big)$$

2. *Assume given:*
   - $A, A' : \mathrm{FOSet}$ *and* $f : A \cong A'$.
   - $B' : A' \to \mathrm{FOSet}$.
   - $c : \mathcal{P}(A)$ *and* $d : (a' : A') \to \mathcal{P}(B'(a'))$

   Then we have:

$$\mathcal{P}(f, \lambda x.x)(\gamma_{\mathcal{P}}(c, d \circ f)) \equiv \gamma_{\mathcal{P}}(\mathcal{P}(f)(c), d)$$

   Note that $(f, \lambda x.x) : \Sigma(a : A).\, B'(f(a)) \cong \Sigma_{A'} B'$.

3. *Given* $A : \mathrm{FOSet}$ *and* $c : \mathcal{P}(A)$ *we have:*

$$\gamma_{\mathcal{P}}(c, \lambda\_.\mathrm{id}) \equiv \mathcal{P}(\eta'_A)(c)$$

4. *Given* $B : \mathrm{Fin}(1) \to \mathrm{FOSet}$ *and* $d : (x : \mathrm{Fin}(1)) \to B(x)$ *we have:*

$$\gamma_{\mathcal{P}}(\mathrm{id}, d) \equiv \mathcal{P}(\eta_B)(d(0))$$

5. *Assume given:*

   - $A : \mathrm{FOSet}$ *and* $c : \mathcal{P}(A)$.
   - $B : A \to \mathrm{FOSet}$ *and* $d : (a : A) \to \mathcal{P}(B(a))$.
   - $C : \Sigma_A B \to \mathrm{FOSet}$
   - $e : (x : \Sigma_A B) \to \mathcal{P}(C(x))$.

   Then we have:

$$\gamma_{\mathcal{P}}(\gamma_{\mathcal{P}}(c, d), e) \equiv \mathcal{P}(\psi_{A,B,C})(\gamma_{\mathcal{P}}(c, \lambda a.\, \gamma_{\mathcal{P}}(d(a), \lambda b.\, e(a, b)))$$

Now we explain the meaning of the five equations in this definition.

Equations 1 and 2 mean that $\gamma_{\mathcal{P}}$ is natural in the only reasonable sense. It is not stated as one equation to avoid the use of a transport along a strict equality, using the decomposition of any function:

$$(f, F) : \Sigma_A B \to \Sigma_{A'} B'$$

as:

$$\Sigma_A B \xrightarrow{(f, \lambda x.x)} \Sigma(a : A).\, B'(f(a)) \xrightarrow{(\lambda x.x, F)} \Sigma_{A'} B'$$

Equations 3 and 4 roughly state that $\mathrm{id}_{\mathcal{P}}$ is a unit on the right and left for $\gamma_{\mathcal{P}}$, and equation 5 that $\gamma_{\mathcal{P}}$ is associative.

To build a usual operad $\mathcal{Q} : \mathbb{N} \to \mathrm{Set}$ out of such an operad $\mathcal{P}$, one should define $\mathcal{Q}(n)$ as $\mathcal{P}(\mathrm{Fin}(n))$ for n:$\mathbb{N}$. So we will use the notation $\mathcal{P}(n)$ for $\mathcal{P}(\mathrm{Fin}(n))$ with $n : \mathbb{N}$.

We define morphisms between operads.

Definition 3.9. *Let* $\mathcal{P}$ *and* $\mathcal{Q}$ *be operads. A morphism from* $\mathcal{P}$ *to* $\mathcal{Q}$ *is a natural transformation* α *of the underlying functors such that:*

1. $\alpha_{\mathrm{Fin}(1)}(\mathrm{id}_{\mathcal{P}}) \equiv \mathrm{id}_{\mathcal{Q}}$.

2. *Assume given:*

   - $A : \mathrm{FOSet}$ *and* $B : A \to \mathrm{FOSet}$.
   - $c : \mathcal{P}(A)$ *and* $d : (a : A) \to \mathcal{P}(B(a))$

   Then we have:

$$\alpha_{\Sigma_A B}(\gamma_{\mathcal{P}}(c, d)) \equiv \gamma_{\mathcal{Q}}(\alpha_A(c), \lambda a.\, \alpha_{B(a)}(d(a)))$$

Definition 3.10. *We define an operad* Mon*. For any* $A : \mathrm{FOSet}$*, the type* $\mathrm{Mon}(A)$ *is* $\top$ *the terminal object of* Set*. The identity and compositions can be defined in a unique way. This operad is terminal.*

Now we define the pullback of operads. Together with the previous definition this shows that the category of operads has all finite limits.

Lemma 3.11. *Assume given two morphisms of operads:*

$$\alpha : \mathcal{P} \to \mathcal{R}$$

and

$$\beta : \mathcal{Q} \to \mathcal{R}$$

Then the pullback of α *and* β *seen as natural transformations can be endowed with the structure of a pullback of operads.*

Proof. We denote by $\mathcal{P} \times_{\mathcal{R}} \mathcal{Q}$ the pullback of α and β seen as natural transformations. Then the operad structure on $\mathcal{P} \times_{\mathcal{R}} \mathcal{Q}$ can be defined on each component, and it is straightforward to show that this defines a pullback of operads. □

3.3 Algebras for an operad

Recall that we want a notion of algebra for an operad. For X to be a $\mathcal{P}$-algebra, we need to associate to any $c : \mathcal{P}(A)$ an element of $(A \to X) \to X$, i.e. an A-ary operation on X. This should be done in a way compatible with $\mathrm{id}_{\mathcal{P}}$ and $\gamma_{\mathcal{P}}$. This motivates the following two definitions.

Definition 3.12. *Assume given* $X : \mathrm{Set}$*. Then we define an operad* $\mathcal{E}nd_X$*.*

- *For* $A : \mathrm{FOSet}$ *we define* $\mathcal{E}nd_X(A)$ *as* $(A \to X) \to X$*. This has an obvious functor structure.*
- *We define* $\mathrm{id}_{\mathcal{E}nd_X} : \mathcal{E}nd_X(1)$ *as* $\lambda f.\, f(0)$*.*
- *Assume given* $A : \mathrm{FOSet}$ *and* $B : A \to \mathrm{FOSet}$*, together with:*

  $$c : (A \to X) \to X$$

  and

  $$d : (a : A) \to (B(a) \to X) \to X$$

  Then we define $\gamma_{\mathcal{E}nd_X}(c, d)$ *as:*

  $$\lambda f.\, c(\lambda a.\, d(a, \lambda b.\, f(a, b)))$$

The composition takes an A-ary operation c and for any $a : A$ a $B(a)$-ary operation $d(a)$, and outputs a $\Sigma_A B$-ary operation in the natural way. A pleasing feature of our definition of operads is that $\mathcal{E}nd_X$ obeys the axioms of operads definitionally. Note that $\mathcal{E}nd_X$ is not functorial in X, because of variance problems.

Now we can give the definition of algebras for an operad.

Definition 3.13. *Assume given* $X : \mathrm{Set}$ *and an operad* $\mathcal{P}$*. Then a* $\mathcal{P}$*-algebra structure on* X *is defined as a morphism of operads from* $\mathcal{P}$ *to* $\mathcal{E}nd_X$*.*

This match our intuition, indeed for $A : \mathrm{FOSet}$ and $c : \mathcal{P}(A)$, a $\mathcal{P}$-algebra structure on X gives an element in $(A \to X) \to X$, i.e. a A-ary operation. The fact that those operations respect the composition of $\mathcal{P}$ comes from the fact that we ask for a morphism of operads from $\mathcal{P}$ to $\mathcal{E}nd_X$. For example the Mon-algebras are the strict monoids.

We define morphisms between $\mathcal{P}$-algebras.

Definition 3.14. *A morphism between two* $\mathcal{P}$*-algebras* (X, ϵ_X) *and* (Y, ϵ_Y) *is a map* $f : X \to Y$ *such that for any* $A : \mathrm{FOSet}$ *and* $c : \mathcal{P}(A)$ *we have:*

$$f \circ \epsilon_X(c) \equiv \lambda h.\, \epsilon_Y(c)(f \circ h)$$

3.4 Toward an operad for monoids up to coherent homotopy

Recall that we want to define an operad ∞Mon whose algebras are monoids up to coherent homotopy. We use our new vocabulary to formulate its desired properties:

1. For any $A : \mathrm{FOSet}$, the type $\infty\mathrm{Mon}(A)$ should be contractible, because there is a unique way up to homotopy to multiply A elements in a given order.

2. For any $x : X$ with X fibrant, the type $x \rightsquigarrow x$ should be an ∞Mon-algebra. We say that fibrant loop spaces are ∞Mon-algebras.

3. If two fibrant types are equivalent and one of them is an ∞Mon-algebra, then so is the other. We say that ∞Mon-algebra are invariant under equivalences.

4 Cofibrant operads

We want ∞Mon-algebras to be invariant under equivalences. In this section we give a general class of operads called cofibrant, and we show that algebras for cofibrant operads are invariant under equivalences. We will later show that ∞Mon is cofibrant.

We mostly follow an article by Berger and Moerdijk [2], in which they prove (in a set-theoretic foundation) that the category of operads in a suitable monoidal model category is itself a model category. So for example we can study the homotopy theory of topological operads! Their results require some adaptations to our context, of course because we use different foundations but also because types do not form a model category.

4.1 Homotopical structure on operads

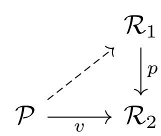
These first definitions are very natural.

Definition 4.1. *A morphism of operads is called a fibration (respectively trivial fibration, equivalence) if its underlying natural transformation is a family of fibrations (respectively trivial fibrations, equivalences).*

Similarly an operad $\mathcal{P}$ is called fibrant if $\mathcal{P}(A)$ is fibrant for all $A : \mathrm{FOSet}$.

Now we define cofibrant operads.

Definition 4.2. *Let $\mathcal{P}$ be an operad. Then $\mathcal{P}$ is called cofibrant if for all trivial fibration of operads p from $\mathcal{R}_1$ to $\mathcal{R}_2$ with $\mathcal{R}_2$ fibrant and for all morphism of operads v from $\mathcal{P}$ to $\mathcal{R}_2$, there exists a morphism of operads from $\mathcal{P}$ to $\mathcal{R}_1$ making the following triangle commutes:*



Remark. *There always exists a natural transformation making this triangle commutes, the point is that it should be a morphism of operads.*

4.2 The cocylinder factorisation for operads

In this section we show that cofibrant operads have a weak lifting property against equivalences between fibrant operads.

Lemma 4.3. *Assume given a morphism α between fibrant operads. Then it factors as a section of a trivial fibration followed by a fibration. Moreover if α is an equivalence, the fibration is trivial.*

Proof. We use the factorisation for types of Lemma 2.26 pointwise. One shows by direct calculations that the obtained natural transformation carries an operad structure such that all involved natural transformations are morphisms of operads. □

Lemma 4.4. *Let $\mathcal{P}$ be a cofibrant operad and assume given an equivalence between fibrant operads $\mathcal{R}_1$ and $\mathcal{R}_2$. Then given a morphism from $\mathcal{P}$ to $\mathcal{R}_2$, one can build a morphism from $\mathcal{P}$ to $\mathcal{R}_1$.*

Proof. This is immediate using Lemma 4.3. □

Note that the triangle obtained by the lifting of Lemma 4.4 does not commute strictly, but only up to homotopy.

4.3 Endomorphism operad of a map between types

We want to show that if X and Y are equivalent fibrant types, $\mathcal{P}$ is a cofibrant operad and Y is a $\mathcal{P}$-algebra then so is X. Unfolding the definition of algebras, one need to build a morphism of operads from $\mathcal{P}$ to $\mathcal{E}nd_X$ using a morphism from $\mathcal{P}$ to $\mathcal{E}nd_Y$. So we have to provide a link between the operads $\mathcal{E}nd_X$ and $\mathcal{E}nd_Y$ from a map $f : X \to Y$. This will be done using the next construction.

Lemma 4.5. *Assume given a map $f : X \to Y$. Then we can define an operad $\mathcal{E}nd_f$ as follows:*

- *For $A : \mathrm{FOSet}$, we define $\mathcal{E}nd_f(A)$ as:*

$$\Sigma(c_1 : \mathcal{E}nd_X(A))(c_2 : \mathcal{E}nd_Y(A)).\, f \circ c_1 \equiv \lambda h.\, c_2(f \circ h)$$

  This carry an obvious structure of functor inherited from the ones of $\mathcal{E}nd_X$ and $\mathcal{E}nd_Y$.

- *We define $\mathrm{id}_{\mathcal{E}nd_f} : \mathcal{E}nd_f(1)$ as $(\mathrm{id}_{\mathcal{E}nd_X}, \mathrm{id}_{\mathcal{E}nd_Y})$ together with the obvious proof of equality.*

- *Composition is defined component-wise using $\gamma_{\mathcal{E}nd_X}$ and $\gamma_{\mathcal{E}nd_Y}$.*

Then the projections from $\mathcal{E}nd_f$ to $\mathcal{E}nd_X$ and $\mathcal{E}nd_Y$ are morphisms of operads.

Proof. A straightforward calculation shows that this defines an operad, using the uniqueness of identity proofs. The projections are morphisms of operads by definition. □

We call $\mathcal{E}nd_f$ the endomorphism operad of f. Up to isomorphism, an element in $\mathcal{E}nd_f(n)$ consists of two n-ary operations $\psi : X^n \to X$ and $\varphi : Y^n \to Y$, such that for all $x_1, \cdots, x_n : X$ we have:

$$f(\psi(x_1, \cdots, x_n)) \equiv \varphi(f(x_1), \cdots, f(x_n))$$

This is reminiscent of the definition of morphism between algebras. In fact one can easily show that given (X, ϵ_X) and (Y, ϵ_Y) two $\mathcal{P}$-algebras, $f : X \to Y$ is a morphism of $\mathcal{P}$-algebras if and only if ϵ_X and ϵ_Y factor through $\mathcal{E}nd_f$.

Definition 4.6. *Assume given two types X and Y. Then for $A : \mathrm{FOSet}$ we define $\mathcal{E}nd_{X,Y}(A)$ as:*

$$(A \to X) \to Y$$

This defines a functor from FOSet to types.

Note that $\mathcal{E}nd_{X,Y}$ does not have an operad structure.

Lemma 4.7. *Assume given a map $f : X \to Y$.*

Then the following square is a pullback of natural transformations:

$$\begin{array}{ccc} \mathcal{E}nd_f & \xrightarrow{\pi_X} & \mathcal{E}nd_X \\ {\scriptstyle \pi_Y}\downarrow & & \downarrow{\scriptstyle f_*} \\ \mathcal{E}nd_Y & \xrightarrow[f^*]{} & \mathcal{E}nd_{X,Y} \end{array}$$

where:

- $f_*(c_1)$ *is* $f \circ c_1$.
- $f^*(c_2)$ *is* $\lambda h.\, c_2(f \circ h)$.
- π_X *and* π_Y *are the projections.* □

Now we have a way to link $\mathcal{E}nd_X$ and $\mathcal{E}nd_Y$ using $f : X \to Y$.

4.4 The case of trivial fibrations

In this section we want to show that given a trivial fibration between fibrant types $f : X \to Y$ and a cofibrant operad $\mathcal{P}$, we have that X is a $\mathcal{P}$-algebra if and only if Y is a $\mathcal{P}$-algebra. To do this we use the endomorphism operad of f.

Lemma 4.8. *Assume given a trivial fibration between fibrant types* $f : X \to Y$*. In the pullback diagram of Lemma 4.7:*

$$\begin{array}{ccc} \mathcal{E}nd_f & \xrightarrow{\pi_X} & \mathcal{E}nd_X \\ {\scriptstyle \pi_Y}\downarrow & & \downarrow{\scriptstyle f_*} \\ \mathcal{E}nd_Y & \xrightarrow[f^*]{} & \mathcal{E}nd_{X,Y} \end{array}$$

we have that:

- π_Y *is a trivial fibration of operad with fibrant base.*
- π_X *is an equivalence between fibrant operads.*

Proof. First we see by induction on the size of $A : \mathrm{FOSet}$ that for any fibrant type X, the type $A \to X$ is fibrant. From this it is immediate to conclude that $\mathcal{E}nd_X$, $\mathcal{E}nd_Y$ and $\mathcal{E}nd_{X,Y}$ are fibrant.

- The fibre of f_* over $h : (A \to X) \to Y$ is strictly isomorphic to:

  $$(x : A \to X) \to \mathrm{fibre}_f(h(x))$$

  This type is fibrant and contractible because f is a trivial fibration and $A \to X$ is fibrant. Then π_Y is a trivial fibration because it is the pullback of f_*.

- We know that $\mathcal{E}nd_X$ is fibrant and since π_Y is a fibration and $\mathcal{E}nd_Y$ is fibrant, $\mathcal{E}nd_f$ is fibrant. So in order to show that π_X is an equivalence, it is enough to show that f_*, π_Y and f^* are, by the two-out-of-three property of equivalences between fibrant types. We already know that f_* and π_Y are equivalences, and so is f^* by direct calculation. □

So we have a strong link between $\mathcal{E}nd_X$ and $\mathcal{E}nd_Y$ when we have a trivial fibration between fibrant types $f : X \to Y$. We now use this link.

Lemma 4.9. *Assume given a trivial fibration between fibrant types $f : X \to Y$ and a cofibrant operad $\mathcal{P}$.*

If Y is a $\mathcal{P}$-algebra, so is X.

Proof. We have a morphism ϵ_Y from $\mathcal{P}$ to $\mathcal{E}nd_Y$. But since π_Y is a trivial fibration of operads with fibrant base by Lemma 4.8, we can lift ϵ_Y to a morphism from $\mathcal{P}$ to $\mathcal{E}nd_f$. So we have a morphism form $\mathcal{P}$ to $\mathcal{E}nd_X$ by composition with π_X. □

Lemma 4.10. *Assume given a trivial fibration between fibrant types $f : X \to Y$ and a cofibrant operad $\mathcal{P}$.*

If X is a $\mathcal{P}$-algebra, so is Y.

Proof. We have a morphism ϵ_X from $\mathcal{P}$ to $\mathcal{E}nd_X$. But since π_X is an equivalence between fibrant operads by Lemma 4.8, we can lift ϵ_X to a morphism from $\mathcal{P}$ to $\mathcal{E}nd_f$ using Lemma 4.4. So we have a morphism from $\mathcal{P}$ to $\mathcal{E}nd_Y$ by composition with π_Y. □

Remark. *Lemma 4.9 produces an algebra structure such that f is a morphism of algebras, but not Lemma 4.10. This comes from the fact that the lifting of Lemma 4.4 does not commute strictly. Nevertheless it commutes up to homotopy, and in Lemma 4.10, f is a morphism of algebra up to homotopy, in a sense we do not make precise.*

4.5 Invariance under equivalence of algebras for a cofibrant operad

We are now ready to prove the main theorem of this section.

Theorem 4.11. *Assume given a cofibrant operad $\mathcal{P}$ and an equivalence between two fibrant types X and Y. Then if Y has a $\mathcal{P}$-algebra structure, so does X.*

Proof. We use the factorisation of Lemma 2.26 on the equivalence between X and Y in order to obtain two trivial fibrations, and then we use Lemmas 4.9 and 4.10 to conclude. □

5 The operad ∞Mon

In this section we give a definition of the operad ∞Mon. We use ideas from another article by Berger and Moerdijk [3], where a cofibrant replacement is constructed for operads in a suitable model category. We apply a variant of this construction to the operad for strict monoids.

In fact our construction can probably be generalized to any operad, again following [3], so that any algebraic notion defined by an operad can be defined up to coherent homotopy.

5.1 Non-functorial operads with partial compositions

In this section we give an alternate definition of operad, which will be used to define ∞Mon. It is based on two ideas:

- On one hand, it was already explained that operads can be defined as families indexed by $\mathbb{N}$ rather than functors from the groupoid of finite totally ordered sets.
- On the other hand, rather than axiomatizing the composition of an n-ary operation with n operations, one can axiomatize the so-called partial compositions of a n-ary operation c with another operation. There are n such partial compositions, one for each input of c.

Definition 5.1. *A non-functorial operad with partial compositions consists of:*

- *A family of types* $\mathcal{P} : \mathbb{N} \to \mathrm{Set}$.
- *An identity* $\mathrm{id}_{\mathcal{P}} : \mathcal{P}(1)$.
- *Some partial compositions* $\_ \circ_k \_ : \mathcal{P}(m) \to \mathcal{P}(n) \to \mathcal{P}(m+n-1)$ *for any* $m, n : \mathbb{N}$ *and* $k : \mathrm{Fin}(m)$.

such that:

- *For all* $n : \mathbb{N}$, $k : \mathrm{Fin}(n)$ *and* $c : \mathcal{P}(n)$, *we have:*

$$c \circ_k \mathrm{id}_{\mathcal{P}} \equiv c$$

- *For all* $n : \mathbb{N}$ *and* $c : \mathcal{P}(n)$ *we have:*

$$\mathrm{id}_{\mathcal{P}} \circ_0 c \equiv c$$

- *For all* $l, m, n : \mathbb{N}$ *and* $k_1 : \mathrm{Fin}(l)$ *and* $k_2 : \mathrm{Fin}(l+m-1)$ *together with* $c_1 : \mathcal{P}(l)$, $c_2 : \mathcal{P}(m)$ *and* $c_3 : \mathcal{P}(n)$ *we have:*

  1. *If* $k_2 < k_1$ *then:*

  $$(c_1 \circ_{k_1} c_2) \circ_{k_2} c_3 \equiv (c_1 \circ_{k_2} c_3) \circ_{k_1+n-1} c_2$$

  2. *If* $k_1 \leq k_2 < k_1 + m$, *then:*

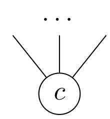
  $$(c_1 \circ_{k_1} c_2) \circ_{k_2} c_3 \equiv c_1 \circ_{k_2} (c_2 \circ_{k_2-k_1} c_3)$$

  3. *If* $k_1 + m \leq k_2$ *then:*

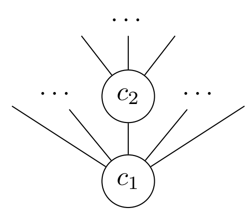
  $$(c_1 \circ_{k_1} c_2) \circ_{k_2} c_3 \equiv (c_1 \circ_{k_2-m+1} c_3) \circ_{k_1} c_2$$

Note that we have omitted all the transports along strict equalities in $\mathbb{N}$. We will often write $\_ \circ \_$ instead of $\_ \circ_k \_$, omitting the index.

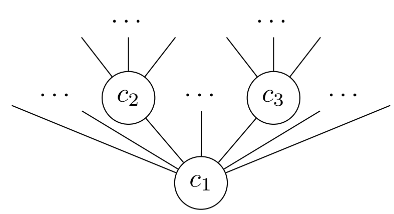
We draw $c : \mathcal{P}(n)$ as:

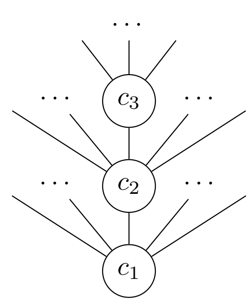


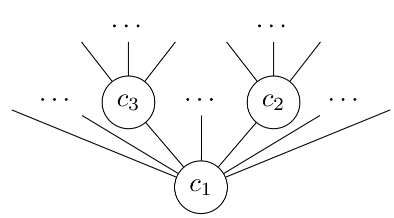
and we draw the partial composite $c_1 \circ c_2$ as:



The following drawings can be interpreted as partial composites in two ways:

1. 

2. 

3. 

Then the equalities 1, 2 and 3 state that the two interpretations give the same result, so that we are indeed allowed to write partial compositions this way. Now we give the straightforward definition of morphisms between non-functorial operads with partial compositions.

Definition 5.2. *A morphism between two non-functorial operads with partial compositions $\mathcal{P}$ and $\mathcal{Q}$ is a family of map $\alpha_n : \mathcal{P}(n) \to \mathcal{Q}(n)$ such that:*

- $\alpha_1(\mathrm{id}_{\mathcal{P}}) \equiv \mathrm{id}_{\mathcal{Q}}$
- *For all* $m, n : \mathbb{N}$, $k : \mathrm{Fin}(m)$ *and* $c_1 : \mathcal{P}(m)$, $c_2 : \mathcal{P}(n)$, *we have:*

$$\alpha_{m+n-1}(c_1 \circ_k c_2) \equiv \alpha_m(c_1) \circ_k \alpha_n(c_2)$$

The proof of the next lemma is very sketchy, but the result is well-known.

Lemma 5.3. *The category of non-functorial operads with partial compositions is equivalent to the category of operads.*

Proof. There is a clear notion of non-functorial operads with total composition, which are families $\mathcal{P} : \mathbb{N} \to \mathrm{Set}$ such that:

- We have $\mathrm{id} : \mathcal{P}(1)$.
- Assume given:
  - $n : \mathbb{N}$ and $s_k : \mathbb{N}$ for all $k : \mathrm{Fin}(n)$
  - $c : \mathcal{P}(n)$ and $c_k : \mathcal{P}(s_k)$ for all $k : \mathrm{Fin}(n)$

  Then we have:

$$\gamma(c; c_0, \cdots, c_{n-1}) : \mathcal{P}(s_0 + \cdots + s_{n-1})$$

Of course γ and id are supposed to respect the obvious associativity and unit axioms.

There is an equivalence between non-functorial operads with total composition and operads, essentially defined from the equivalence between the groupoid of finite totally ordered sets and the category with $\mathbb{N}$ as object and only identities as morphisms.

Now we just need to give an equivalence between non-functorial operads with partial and total compositions. To do this we define partial compositions from a total one, and vice-versa.

- From a total composition γ we can define $c_1 \circ_k c_2$ as:

$$\gamma(c_1; \mathrm{id}, \cdots, \mathrm{id}, c_2, \mathrm{id}, \cdots, \mathrm{id})$$

  with c_2 at the k-th place.

- From partial compositions $\_ \circ \_$ we can define $\gamma(c; c_0, \cdots, c_{n-1})$ as:

$$(\cdots((c \circ_{n-1} c_{n-1}) \circ_{n-2} c_{n-2}) \cdots) \circ_0 c_0$$

A straightforward but lengthy calculation shows that this construction preserves the axioms of operads, and that it interacts well with morphisms. ☐

From now on we will call non-functorial operads with partial composition simply operads. This is justified by Lemma 5.3.

5.2 Definition of labelled trees

We define two operads, using the equivalent definition from the last section.

Definition 5.4. *We define the inductive type* Tree *with the following constructors:*

- leaf : Tree
- cons : $(n : \mathbb{N}) \to (\mathrm{Fin}(n) \to \mathrm{Tree}) \to \mathrm{Tree}$

Definition 5.5. *We define* Tree(n) *as the type of trees with n leaves. This has an operad structure induced by the grafting of trees.*

We define the internal vertices of a tree as its cons *constructors which are not at the root.*

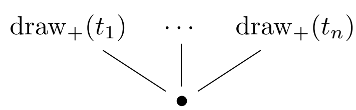
Definition 5.6. *We define a labelled tree as a tree together with a map from its internal vertices to* $\mathbb{I}$*. We denote the type of labelled trees by* LTree.

For any $n : \mathbb{N}$*, we define* LTree(n) *as the type of labelled trees with n leaves.*

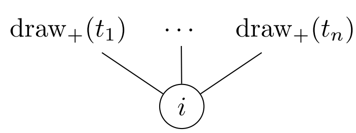
We will often omit the labeling of a tree in LTree from notations.

For t : LTree, we can define its graphical representation draw(t) as follows:

- draw(leaf) is $\emptyset$.
- draw(cons$(t_1, ..., t_n)$) is:



- draw$_+$(leaf) is a blank space. Note draw$_+$ is only applied to subtrees, so there is no tree drawn as a blank space.
- draw$_+$(cons$(t_1, ..., t_n)$) is:



  where i is the label of the corresponding internal vertex.

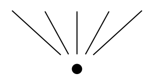
Remark. *The three labelled trees:*

should not be confused. They correspond to cons$(1, \lambda_.\,\mathrm{leaf})$*,* leaf *and* cons$(0, \lambda())$ *(where* $\lambda()$ *is the unique function out of* Fin(0) *to anything).*

Definition 5.7. *We denote by μ_n the element:*

$$\operatorname{cons}(n, \lambda\_.\operatorname{leaf}) : \operatorname{LTree}(n)$$

For example μ_5 is the tree:

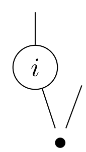


Note that for any $n : \mathbb{N}$, the tree μ_n has no internal vertex.

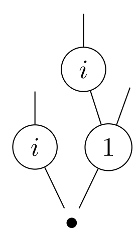
Lemma 5.8. *Assume given two labelled trees t_1 and t_2 different from* leaf*, then there is a new internal vertex in $t_1 \circ t_2$ coming from the root of t_2.*

The family of types $\operatorname{LTree}(n)$ for $n : \mathbb{N}$ carries an operad structure, with partial compositions defined using the grafting of trees with any new internal vertex labelled by 1.

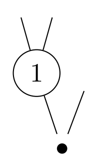
So for example if we compose:



with itself at its right leaf, we obtain:



Similarly $\mu_2 \circ_0 \mu_2$ is:



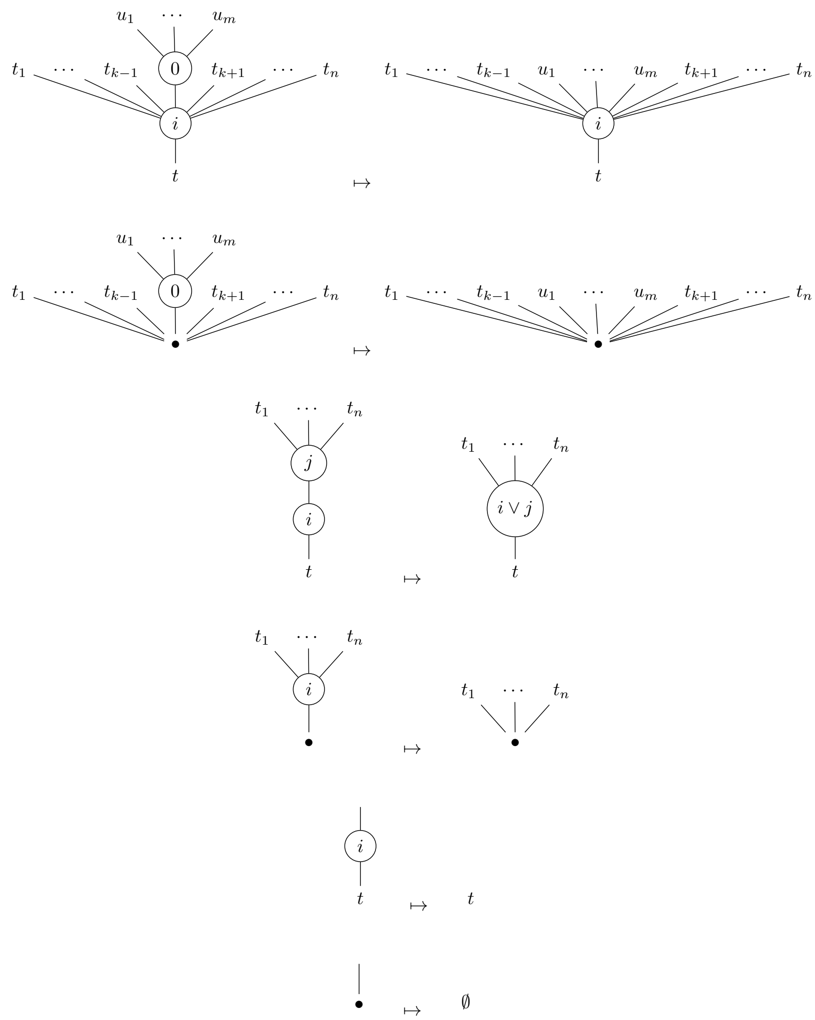
5.3 A rewriting on labelled trees

We call a binary relation $\_R\_$ on a type X a rewriting on this type. The intuition is that $x\,R\,y$ means that x rewrites in one step to y.

In this section we define a rewriting on labelled trees, and state some of its properties.

Remark. *There are no proofs in this section. They consist essentially of a verification for all the possible cases. It is here that we use all our assumptions on the connections on* $\mathbb{I}$.

Definition 5.9. *For* $n : \mathbb{N}$*, we define a rewriting* $\_ \mapsto \_$ *on* $\mathrm{LTree}(n)$ *as follows:*



Lemma 5.10. *Assume given* $t_1, t_2 : \mathrm{LTree}$.

- *If* $t_1 \mapsto t_1'$ *then* $t_1 \circ t_2 \mapsto t_1' \circ t_2$.
- *If* $t_2 \mapsto t_2'$ *then* $t_1 \circ t_2 \mapsto t_1 \circ t_2'$.

In the language of rewritings, the next lemma states that $\_ \mapsto \_$ is strongly confluent.

Lemma 5.11. *For* $t, t' : \mathrm{LTree}$*, we write* $t \mapsto_{\leq 1} t'$ *to say that* $t \mapsto t'$ *or* $t \equiv t'$.

Assume given $t, t_1, t_2 : \mathrm{LTree}$ *such that* $t \mapsto t_1$ *and* $t \mapsto t_2$*. Then there exists* $t_3 : \mathrm{LTree}$ *such that* $t_1 \mapsto_{\leq 1} t_3$ *and* $t_2 \mapsto_{\leq 1} t_3$.

Lemma 5.12. *Assume* $0 \not\equiv 1$ *in* $\mathbb{I}$*. For all* $t_1, t_2 : \mathrm{LTree}$ *such that* $t_1 \circ t_2 \mapsto t_3$ *one of the following is true:*

- *Either* $t_1 \mapsto t_1'$ *with* $t_3 \equiv t_1' \circ t_2$.
- *Otherwise* $t_2 \mapsto t_2'$ *with* $t_3 \equiv t_1 \circ t_2'$.

The condition $0 \not\equiv 1$ in $\mathbb{I}$ is necessary for the result to hold, otherwise:

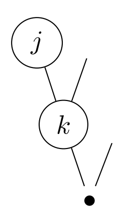
$$\mu_2 \circ \mu_2 \mapsto \mu_3$$

but μ_2 does not rewrite, i.e. there is no t such that $\mu_2 \mapsto t$. It is reasonable to assume $0 \not\equiv 1$ in $\mathbb{I}$, for example it is implied by the univalence of the fibrant universes. Nevertheless if we assume $0 \equiv 1$, then homotopy equivalences are isomorphisms, and loop spaces are singletons, so all our theorems become trivial. Therefore it is somewhat unsatisfying that we have to require this hypothesis here.

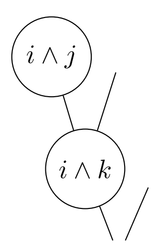
Next we introduce a construction on labelled trees using the connection $\_ \wedge \_$.

Definition 5.13. *For any labelled tree* t *and* $i : \mathbb{I}$*, we define a labelled tree* $i \wedge t$ *by applying* $i \wedge \_ : \mathbb{I} \to \mathbb{I}$ *on the label of each internal vertex of* t.

So fo example if we denote by t the tree:



then for $i : \mathbb{I}$ the tree $i \wedge t$ is:



Lemma 5.14. *Assume given* $t, t' : \mathrm{LTree}$ *such that* $t \mapsto t'$*. Then for any* $i : \mathbb{I}$*, we have* $i \wedge t \mapsto i \wedge t'$*.*

5.4 Definition of ∞Mon

Given any type X and any binary relation R on X, it is possible to define the type-theoretic quotient of X by R. It is isomorphic to the quotient of X by the equivalence relation generated by R. These quotient types are not present by default in Agda, so we postulate them.

Now we can define the operad ∞Mon as a quotient of LTree.

Definition 5.15. *The quotient of* $\mathrm{LTree}(n)$ *by* $\_ \mapsto \_$ *is denoted by* $\infty\mathrm{Mon}(n)$*. For* $t : \mathrm{LTree}(n)$ *we denote by* $[t]$ *its image in* $\infty\mathrm{Mon}(n)$*.*

Recall that our goal is to define monoids up to coherent homotopy as ∞Mon-algebras. So for example, the image of $[\mu_n]$ in an ∞Mon-algebra will be called the canonical multiplication of n elements. We define composition using 1 as a label, whereas we quotient every trees with labels 0, so that in any ∞Mon-algebra we have that:

- On one hand there is a path from a canonical multiplication to any composition of canonical multiplications with the suitable number of inputs (this will be proven in Lemma 5.17).
- On the other hand if $0 \not\equiv 1$ (as implied by univalence), such composition is not strictly equal to the canonical multiplication.

Remark. *Note that* Tree*,* LTree *and* ∞Mon *are not fibrant. This is not a problem because we want to build* ∞Mon*-algebras, which are maps out of* ∞Mon*.*

Now we show that the partial compositions in LTree induces partial compositions in ∞Mon.

Lemma 5.16. *The operad structure on* LTree *induces an operad structure on* ∞Mon*.*

Proof. The fact that partial compositions of labelled trees factors through quotient is a direct consequence of Lemma 5.10. □

Now we prove the first important property of ∞Mon.

Lemma 5.17. *For any* $n : \mathbb{N}$*, the type* $\infty\mathrm{Mon}(n)$ *is contractible.*

Proof. The type $\infty\mathrm{Mon}(n)$ is inhabited by $[\mu_n]$. Assume given $[t] : \infty\mathrm{Mon}(n)$, then we need a path from $[\mu_n]$ to $[t]$. We define $p_t : \mathbb{I} \to \infty\mathrm{Mon}(n)$ as $\lambda i.\, [i \wedge t]$. But $p_t(0) \equiv [\mu_n]$ and $p_t(1) \equiv [t]$, so this gives the desired path.

Now we need to show that if $t \mapsto t'$, we have $p_t \equiv p_{t'}$, so that we indeed have a proof of contractibility of $\infty\mathrm{Mon}(n)$. But this is a direct consequence of Lemma 5.14. □

5.5 Sections of strongly contractible morphisms over ∞Mon

In this section we will show the key property of ∞Mon, which will be sufficient to imply the other results we want. First we need to define strongly contractible morphisms.

Definition 5.18. *We define $\delta : \partial\,\mathbb{I} \to \mathbb{I}$ as the inclusion of the type with two points in $\mathbb{I}$ given by the endpoints 0 and 1.*

For $k : \mathbb{N}$ we define $\delta_k : \partial\,\mathbb{I}^k \to \mathbb{I}^k$ as the iterated pushout-product of $\delta : \partial\,\mathbb{I} \to \mathbb{I}$, i.e.

- *$\delta_0 : \partial\,\mathbb{I}^0 \to \mathbb{I}^0$ is the unique map form $\bot$ to $\top$.*
- *$\delta_{k+1} : \partial\,\mathbb{I}^{k+1} \to \mathbb{I}^{k+1}$ is the map $\delta_k \,\Box\, \delta$.*

Intuitively δ_k in the inclusion of the border in the k-dimensional cube.

Definition 5.19. *A map between types is called strongly contractible if it has the right lifting property against $\delta_k : \partial\,\mathbb{I}^k \to \mathbb{I}^k$ for all $k : \mathbb{N}$.*

A morphism of operads is called strongly contractible if its underlying natural transformation is a family of strongly contractible maps.

It is easy to check that a strongly contractible map is contractible (i.e. it has contractible fibres), but the converse is not true without some fibrancy hypothesis. Firstly it is possible to fill any cube in the fibres of a strongly contractible map, whereas it might not be possible to fill any cube in a non-fibrant contractible type. Secondly the lifting property of strongly contractible maps guarantee a correspondence between its fibres over points linked by a path. This correspondence is not here for a contractible map with non-fibrant base. We will see later that a fibration with fibrant base is contractible if and only if it is strongly contractible.

Now we give the key property of ∞Mon. Its proof is quite involved, but the result can be efficiently used as a black box.

Theorem 5.20. *Let $\mathcal{P}$ be an operad. Then any strongly contractible morphism of operads from $\mathcal{P}$ to ∞Mon has a section which is a morphism of operads.*

Proof. Let us denote by β the given morphism from $\mathcal{P}$ to ∞Mon.

For $n, k : \mathbb{N}$ we denote by $\mathrm{LTree}_k(n)$ the type of labelled trees with n leaves and at most k internal vertices. We denote by LTree_k the type of trees with at most k internal vertices. Note that $\_ \mapsto \_$ strictly decreases the number of internal edges of a tree, so it restricts to a rewriting on $\mathrm{LTree}_k(n)$ which we denote by $\_ \mapsto \_$ as well.

By induction on $k : \mathbb{N}$ we define $\alpha_k : (n : \mathbb{N}) \to \mathrm{LTree}_k(n) \to \mathcal{P}(n)$ such that:

0. α_{k+1} extends α_k.
1. For all $t : \mathrm{LTree}_k$, we have $\beta(\alpha_k(t)) \equiv [t]$.

2. For any $t_1, t_2 : \mathrm{LTree}$ such that $t_1 \circ t_2 : \mathrm{LTree}_k$, we have

$$\alpha_k(t_1 \circ t_2) \equiv \alpha_k(t_1) \circ \alpha_k(t_1)$$

   Moreover $\alpha_k(\mathrm{id}_{\mathrm{LTree}}) \equiv \mathrm{id}_{\mathcal{P}}$.

3. For any $t_1, t_2 : \mathrm{LTree}_k$, if $t_1 \mapsto t_2$ then $\alpha_k(t_1) \equiv \alpha_k(t_2)$.

From this we can conclude. Indeed, we first define α from LTree to $\mathcal{P}$ as the union of the α_k. Point 2 implies that α is a morphism of operads and point 3 implies that α factors through ∞Mon. Then point 1 shows that the factored morphism is indeed a section of β.

When k is 0, we just need to define the image of μ_n in $\mathcal{P}(n)$ for all $n : \mathbb{N}$, and the image of leaf. This is done using the fact that the fibres of β are inhabited by the left lifting property against $\delta_0 : \partial\, \mathbb{I}^0 \to \mathbb{I}^0$. We have to be careful to define the lifting of μ_1 and leaf to be $\mathrm{id}_{\mathcal{P}}$.

Assume given such an α_k for $k : \mathbb{N}$, we want to extend it as α_{k+1}. Assume given $t : \mathrm{Tree}$ with $k+1$ internal vertices and n leaves, seen as a map $t : \mathbb{I}^{k+1} \to \mathrm{LTree}_{k+1}(n)$. We want to define α_{k+1} on $t(x)$ for all $x : \mathbb{I}^{k+1}$.

If the unlabelled tree t has a unary vertex, then we know $t(x) \mapsto t'(x)$ for any $x : \mathbb{I}^{k+1}$ by a reduction eliminating this unary vertex. So we define $\alpha_{k+1}(t(x))$ as $\alpha_k(t'(x))$. This does not depend on the choice of reduction by Lemma 5.11 and hypothesis 3.

Otherwise the tree has no unary vertex. We define $v_t : \partial\, \mathbb{I}^k \to \mathcal{P}(n)$. Assume given $x : \partial\, \mathbb{I}^k$, then if one of its component is 0 we have $t(x) \mapsto t'(x)$. So we define $v_t(x)$ as $\alpha_k(t'(x))$. Otherwise one of the component of x is 1, and $t(x)$ is $t_1(x) \circ t_2(x)$. We define $v_t(x)$ as $\alpha_k(t_1(x)) \circ \alpha_k(t_2(x))$. To show that this defines a map $v_t : \partial\, \mathbb{I}^{k+1} \to \mathcal{P}(n)$, we need to show that both definitions agree when two components of x are equals to 0 or 1. We have three cases:

- If there are two 0s in $x : \mathbb{I}^{k+1}$, then we have $t(x) \mapsto t_1(x)$ and $t(x) \mapsto t_2(x)$. We need to check that $\alpha_k(t_1(x)) \equiv \alpha_k(t_2(x))$, but this is the case by Lemma 5.11 and hypothesis 1 on α_k.

- If there is a 0 and a 1 in $x : \mathbb{I}^{k+1}$, then $t(x) \equiv t_1(x) \circ t_2(x)$ and $t(x) \mapsto t_3(x)$. But by Lemma 5.12, we have for example that $t_3(x) \equiv t'_1(x) \circ t_2(x)$ with $t_1(x) \mapsto t'_1(x)$. Then we have:

$$\alpha_k(t_1(x)) \circ \alpha_k(t_2(x)) \equiv \alpha_k(t'_1(x)) \circ \alpha_k(t_2(x))$$
$$\equiv \alpha_k(t'_1(x) \circ t_2(x)) \equiv \alpha_k(t_3(x))$$

- If there are two 1s in $x : \mathbb{I}^{k+1}$, then we have for example:

$$t(x) \equiv t_1(x) \circ (t_2(x) \circ t_3(x)) \equiv (t_1(x) \circ t_2(x)) \circ t_3(x)$$

  But then:

$$\alpha_k(t_1(x)) \circ \alpha_k\Big(t_2(x) \circ t_3(x)\Big) \equiv \alpha_k(t_1(x)) \circ \Big(\alpha_k(t_2(x)) \circ \alpha_k(t_3(x))\Big)$$

$$\equiv \Big(\alpha_k(t_1(x)) \circ \alpha_k(t_2(x))\Big) \circ \alpha_k(t_3(x)) \equiv \alpha_k\Big(t_1(x) \circ t_2(x)\Big) \circ \alpha_k(t_3(x))$$

by hypothesis 3 on α_k and the operad equations in $\mathcal{P}$.

Then we we check that we have a commutative square:

$$\begin{array}{ccc} \partial\,\mathbb{I}^{k+1} & \xrightarrow{v_t} & \mathcal{P}(n) \\ {\scriptstyle\delta_k}\big\downarrow & & \big\downarrow{\scriptstyle\beta} \\ \mathbb{I}^k & \xrightarrow[\lambda x.[t(x)]]{} & \infty\mathrm{Mon}(n) \end{array}$$

Indeed either $v_t(x)$ has been defined as $\alpha_k(t'(x))$ with $t(x) \mapsto t'(x)$, in which case we can conclude by hypothesis 1 on α_k. Otherwise $t(x) \equiv t_1(x) \circ t_2(x)$ and:

$$\beta(v_t(x)) \equiv \beta\Big(\alpha_k(t_1(x)) \circ \alpha_k(t_2(x))\Big) \equiv \beta(\alpha_k(t_1)) \circ \beta(\alpha_k(t_2))$$

$$\equiv [t_1(x)] \circ [t_2(x)] \equiv [t_1(x) \circ t_2(x)] \equiv [t(x)]$$

so the square commutes.

But since β is strongly contractible, we can extend v_t to $V_t : \mathbb{I}^k \to \mathcal{P}(n)$ making the two triangles commute. Then we define $\alpha_{k+1}(t(x))$ as $V_t(x)$. So we have an extension α_{k+1} of α_k to LTree_{k+1}.

Now we check that α_{k+1} satisfies the desired properties.

1. Assume t is an unlabelled tree with $k+1$ internal vertices, and $x : \mathbb{I}^{k+1}$. Then $\beta(\alpha_{k+1}(t(x))) \equiv \beta(V_t(x)) \equiv [t(x)]$

2. Assume t is a labelled tree with $k+1$ internal vertices and $t \equiv t_1 \circ t_2$. Then if t does not contain a unary vertex, by definition:

   $$\alpha_{k+1}(t) \equiv \alpha_k(t_1) \circ \alpha_k(t_2)$$

   Otherwise $t_1 \circ t_2 \mapsto t_3$ through a unary rule, and $\alpha_{k+1}(t) \equiv \alpha_k(t_3)$. By Lemma 5.12 we have for example $t_1 \mapsto t_1'$ and $t_3 \equiv t_1' \circ t_2$. Then:

   $$\alpha_k(t_3) \equiv \alpha_k(t_1' \circ t_2) \equiv \alpha_k(t_1') \circ \alpha_k(t_2)$$

   by hypothesis 2 on α_k. But then by hypothesis 3 on α_k we have:

   $$\alpha_k(t_1') \circ \alpha_k(t_2) \equiv \alpha_k(t_1) \circ \alpha_k(t_2)$$

   so we can conclude.

3. Assume that t is a tree with $k+1$ internal vertices. Assume $t \mapsto t'$ through a rule eliminating a unary vertex. Then $\alpha_{k+1}(t) \equiv \alpha_{k+1}(t')$ by definition. Otherwise we conclude using Lemma 5.11 and hypothesis 3 for α_k.

□

The last theorem means that if we can extend any section over $\partial\,\mathbb{I}^k$ of a morphism β from $\mathcal{P}$ to ∞Mon to a section over $\mathbb{I}^k$, then we can obtain a global section of β which is a morphism of operads. To sum up the situation, there is as little as possible strict equalities between compositions in ∞Mon, so it is easy to build morphism out of it.

6 Properties of ∞Mon

In this section we show that ∞Mon is cofibrant and that it acts on loop spaces. In fact we only use Theorem 5.20, so that any operad obeying this result will be cofibrant and acts on loop spaces.

6.1 Cofibrations and pseudo-cofibrations

In algebraic topology cofibrations are well-behaved inclusions of subspace. In this section we introduce two reasonable definitions of cofibrations in our context. It seems likely that the two definitions are incomparable without further hypothesis, and both are useful. We will show that the maps $\delta_k : \partial\,\mathbb{I}^k \to \mathbb{I}^k$ are examples of both.

Definition 6.1. *A map is called a pseudo-cofibration if its pullback-exponential with any fibration is again a fibration.*

Definition 6.2. *A map is called a cofibration if its pullback-exponential with any trivial fibration with fibrant base is again a trivial fibration with fibrant base.*

Recall that a section of the pullback-exponential of two maps $u : A \to B$ and $p : X \to Y$ shows that any local section of p over A can be extended to a local section over B. So assuming well-behaved pullback-exponentials with $u : A \to B$ has something to do with the ability to extends maps out of A to B, justifying the intuition that a (pseudo-)cofibration is a well-behaved inclusion.

We give two auxiliary lemmas on cofibrations and pseudo-cofibrations. The proofs are the same in both cases, and can be extended to any class of maps defined in a similar fashion.

Lemma 6.3. *The unique map $\bot \to \top$ is a cofibration and a pseudo-cofibration.*

Proof. One can check that for any map p, the map $\langle(\bot \to \top)/p\rangle$ is isomorphic to p. □

Lemma 6.4. *The pushout-product of two cofibrations (respectively two pseudo-cofibrations) is again a cofibration (respectively a pseudo-cofibration).*

Proof. This is a direct consequence of Lemma 2.25. □

Now we show that $\delta_k : \partial\,\mathbb{I}^k \to \mathbb{I}^k$ is a cofibration and a pseudo-cofibration. For these results we will need to use the details of the homotopical structure of our universe. First we give an auxiliary lemma, which is easy to prove.

Lemma 6.5. *Assume given a map $f : X \to Y$, then the map $\langle\delta/f\rangle$ is isomorphic to the obvious map from:*

$$\Sigma(q : \mathbb{I} \to Y).\,(i : \mathbb{I}) \to \mathrm{fibre}_f(q(i))$$

to

$$\Sigma(q : \mathbb{I} \to Y).\,\mathrm{fibre}_f(q(0)) \times \mathrm{fibre}_f(q(1))$$

Moreover the fibre of this map over $q : \mathbb{I} \to Y$, $x : \mathrm{fibre}_f(q(0))$ and $y : \mathrm{fibre}_f(q(1))$ is isomorphic to the type:

$$\Sigma\big(p : (i : \mathbb{I}) \to \mathrm{fibre}_f(q(i))\big).\, (p(0) \equiv x) \times (p(1) \equiv y) \qquad \square$$

Lemma 6.6. *For $k : \mathbb{N}$, the map $\delta_k : \partial\, \mathbb{I}^k \to \mathbb{I}^k$ is a pseudo-cofibration.*

Proof. By Lemmas 6.3 and 6.4, it is enough to prove that $\delta : \partial\, \mathbb{I} \to \mathbb{I}$ is a pseudo-cofibration. Assume given a fibration $f : X \to Y$.

By Lemma 6.5, it is enough to prove that for any $q : \mathbb{I} \to Y$, $x : \mathrm{fibre}_f(q(0))$ and $y : \mathrm{fibre}_f(q(1))$, the type:

$$\Sigma\big(p : (i : \mathbb{I}) \to \mathrm{fibre}_f(q(i))\big).\, (p(0) \equiv x) \times (p(1) \equiv y)$$

is fibrant. This is true by the definition of the interval. $\square$

Lemma 6.7. *For $k : \mathbb{N}$, the map $\delta_k : \partial\, \mathbb{I}^k \to \mathbb{I}^k$ is a cofibration.*

Proof. By Lemmas 6.3 and 6.4, it is enough to prove that $\delta : \partial\, \mathbb{I} \to \mathbb{I}$ is a cofibration. Let $f : X \to Y$ be a trivial fibration with fibrant base.

First we prove that the base of $\langle \delta / f \rangle$ is fibrant. By Lemma 6.5, it is isomorphic to:

$$\Sigma(p : \mathbb{I} \to Y).\, \mathrm{fibre}_f(p(0)) \times \mathrm{fibre}_f(p(1))$$

so it is enough to show $\mathbb{I} \to Y$ fibrant in order to conclude. But $\mathbb{I} \to Y$ is isomorphic to:

$$\Sigma(x, y : Y).\, x \rightsquigarrow y$$

which is assumed fibrant.

Now we prove that $\langle \delta / f \rangle$ is a trivial fibration. By Lemma 6.5, it is enough to show that for any path $q : \mathbb{I} \to Y$ and any $x : \mathrm{fibre}_f(q(1))$ and $y : \mathrm{fibre}_f(q(0))$, the type:

$$\Sigma\big(p : (i : \mathbb{I}) \to \mathrm{fibre}_f(q(i))\big).\, (p(0) \equiv x) \times (p(1) \equiv y)$$

is contractible. By lemma 6.6, we know that it is fibrant. But since contractibility of a fibrant type is a fibrant type, we can use path elimination and assume that q is $\lambda i.z$ for $z : Y$. In this case we need to show that for any $x, y : \mathrm{fibre}_f(z)$ the type $x \rightsquigarrow y$ is contractible, but this is true by Lemma 2.18. $\square$

6.2 ∞Mon is cofibrant

In this section we show that ∞Mon is cofibrant.

Lemma 6.8. *The pullback of a strongly contractible morphism of operads is strongly contractible.*

Proof. Since both strongly contractible morphisms and pullbacks of operads are defined pointwise, it is enough to show the property for maps between types. Being strongly contractible for maps is defined by a right lifting property, so we can conclude using Lemma 2.21. $\square$

Lemma 6.9. *A trivial fibration with fibrant base is strongly contractible.*

Proof. Assume given a trivial fibration with fibrant base p. Then by Lemma 2.24 it is enough to show that $\langle \delta_k / p \rangle$ has a section, but this map is a trivial fibration by Lemma 6.7. $\square$

Theorem 6.10. $\infty\mathrm{Mon}$ *is cofibrant.*

Proof. Assume given two operads $\mathcal{R}_1$ and $\mathcal{R}_2$ with $\mathcal{R}_2$ fibrant, together with a trivial fibration of operads β from $\mathcal{R}_1$ to $\mathcal{R}_2$ and a morphism α from $\infty\mathrm{Mon}$ to $\mathcal{R}_2$. Then we consider the pullback square:

$$\begin{array}{ccc} \infty\mathrm{Mon} \times_{\mathcal{R}_2} \mathcal{R}_1 & \xrightarrow{\epsilon} & \mathcal{R}_1 \\ {\scriptstyle \pi}\downarrow & & \downarrow{\scriptstyle \beta} \\ \infty\mathrm{Mon} & \xrightarrow[\alpha]{} & \mathcal{R}_2 \end{array}$$

We know that β is strongly contractible by Lemma 6.9, and then so is π by Lemma 6.8. So by Theorem 5.20 the morphism π has a section denoted u and $\epsilon \circ u$ is a lifting of α through β. $\square$

6.3 Loop spaces are $\infty\mathrm{Mon}$-algebras

Recall that a fibrant loop space is a type $x \rightsquigarrow x$ with $x : X$ for X fibrant. In this section we show that $\infty\mathrm{Mon}$ acts on fibrant loop spaces. The intuitive idea is that we define the canonical multiplications of paths by path induction as hrefl, and then we build all the coherence conditions inductively on their dimension, defining them by path induction as hrefl. We make this vague argument precise.

First we derive from Theorem 5.20 a way to build maps rather than sections out of $\infty\mathrm{Mon}$. We need an auxiliary definition.

Definition 6.11. *A type X is called strongly contractible if for any $k : \mathbb{N}$ and map $u : \partial\,\mathbb{I}^k \to X$ there exists a dotted arrow making the following triangle commutes:*

$$\begin{array}{ccc} \partial\,\mathbb{I}^k & \xrightarrow{u} & X \\ {\scriptstyle \delta_k}\downarrow & \nearrow & \\ \mathbb{I}^k & & \end{array}$$

An operad $\mathcal{P}$ is called strongly contractible if $\mathcal{P}(A)$ is strongly contractible for any $A : \mathrm{FOSet}$.

We show how to build morphisms of operads out of $\infty\mathrm{Mon}$.

Lemma 6.12. *Assume $\mathcal{P}$ is a strongly contractible operad. Then there is a morphism of operads from $\infty\mathrm{Mon}$ to $\mathcal{P}$.*

Proof. We consider the morphism of operads $\infty\mathrm{Mon} \times \mathcal{P} \to \infty\mathrm{Mon}$. It can be checked that it is strongly contractible, so by Theorem 5.20 it has a section. But the composite of this section with the projection to $\mathcal{P}$ gives the required morphism. $\square$

Now it is enough to build a strongly contractible operad $\mathcal{P}$ with a morphism from $\mathcal{P}$ to $\mathcal{E}nd_{x \rightsquigarrow x}$ for $x : X$ with X fibrant. Recall that we want to define all coherences as hrefl by path induction, but it is not possible to define a function out of $x \rightsquigarrow x$ in such a way. So it is natural consider functions defined on strings of composable paths, and we define the type of such strings. We need some auxiliary definitions on finite totally ordered sets.

Definition 6.13. *We denote the smallest (respectively greatest) element of* $A :$ FOSet *as* min *(respectively* max*).*

Definition 6.14. *Assume given* $A : \mathrm{FOSet}$*, we define:*

- $S(A) : \mathrm{FOSet}$ *as* $A \amalg \top$ *seen as a finitely ordered set with the unique element of* $\top$ *larger than all elements of* A*.*
- *We define* $\mathrm{inc}_0 : A \to S(A)$ *as the obvious inclusion.*
- *For* $a : A$*, we define* $\mathrm{inc}_1(a) : S(A)$ *as the successor of* a *if* $a < \max$*, and we define* $\mathrm{inc}_1(\max)$ *as the unique element of* $\top$*.*

Definition 6.15. *Assume given a type* X *and* $A : \mathrm{FOSet}$*, then we define the type* $A - \mathrm{Path}_X$ *as:*

$$\Sigma(f : S(A) \to X).\,(a : A) \to f(\mathrm{inc}_0(a)) \rightsquigarrow f(\mathrm{inc}_1(a))$$

We have maps:

$$c_0, c_1 : A - \mathrm{Path}_X \to X$$

with $c_0(f, \_)$ *defined as* $f(\min)$ *and* $c_1(f, \_)$ *defined as* $f(\max)$*.*

Moreover for any $x : X$ *we define:*

$$\mathrm{hrefl}_x^A : A - \mathrm{Path}_X$$

as $(\lambda\_.x, \lambda\_.\mathrm{hrefl}_x)$*.*

So $A - \mathrm{Path}_X$ is the type of strings of A composable paths, c_0 gives the first point of the first path in the string and c_1 give the endpoint of the last path in the string. We state the generalized version of path induction for strings of composable paths.

Lemma 6.16. *Assume given* C *a family of fibrant types indexed by* $A - \mathrm{Path}_X$ *for* X *fibrant and* $A : \mathrm{FOSet}$*.*

Assume given $d : (x : X) \to C(\mathrm{hrefl}_x^A)$*, then we have:*

$$J_A(d) : (p : A - \mathrm{Path}_X) \to C(p)$$

Moreover for any $x : X$ *we have* $J_A(\mathrm{hrefl}_x^A) \equiv d(x)$*.* □

We are ready to build the desired operad. We want to consider functions defined on strings of composable paths using path induction, and only those. This justify the following definition.

Definition 6.17. *For any type X and A : FOSet, we define $\mathcal{P}ath_X(A)$ as the type of functions:*

$$\varphi : (p : A - \mathrm{Path}_X) \to c_0(p) \rightsquigarrow c_1(p)$$

such that for all $x : X$ we have $\varphi(\mathrm{hrefl}^A_x) \equiv \mathrm{hrefl}_x$.

Lemma 6.18. *The functor $\mathcal{P}ath_X$ carries an operad structure. Moreover for any $x : X$ there is a morphism of operads from $\mathcal{P}ath_X$ to $\mathcal{E}nd_{x \rightsquigarrow x}$.*

Proof. First we define an operad structure on the functor sending A : FOSet to:

$$(p : A - \mathrm{Path}_X) \to c_0(p) \rightsquigarrow c_1(p)$$

To define composition we use the same idea as for $\mathcal{E}nd_{x \rightsquigarrow x}$, except that we have to be careful about the endpoints of paths. We omit the precise proof. Then it is easy to check that this induces an operad structure on $\mathcal{P}ath_X$.

There are maps from $A \to x \rightsquigarrow x$ to $A - \mathrm{Path}_X$ given by $\lambda f. (\lambda \_. x, f)$. Restrictions along these maps give a morphism of operads from $\mathcal{P}ath_X$ to $\mathcal{E}nd_{x \rightsquigarrow x}$. □

Now we show that $\mathcal{P}ath_X$ is strongly contractible. It is reasonable to expect so because there should be up to homotopy a unique map:

$$\varphi : (p : A - \mathrm{Path}_X) \to c_0(p) \rightsquigarrow c_1(p)$$

such that $\varphi(\mathrm{hrefl}^A_x) \equiv \mathrm{hrefl}_x$ for any $x : X$, which is the one defined using Lemma 6.16.

We prove the generalisation of the desired lifting property for any pseudo-cofibration.

Lemma 6.19. *Assume given a fibrant type X, a pseudo-cofibration $u : A \to B$ and C : FOSet together with a map $\varphi : A \to \mathcal{P}ath_X(C)$. Then there exists a dotted arrow making the following triangle commutes:*

$$\begin{array}{ccc} A & \xrightarrow{\varphi} & \mathcal{P}ath_X(C) \\ {\scriptstyle u}\downarrow & \nearrow & \\ B & & \end{array}$$

Proof. Assume given for any $a : A$ a map:

$$\varphi_a : (p : C - \mathrm{Path}_X) \to c_0(p) \rightsquigarrow c_1(p)$$

such that for all $x : X$ we have $\varphi_a(\mathrm{hrefl}^C_x) \equiv \mathrm{hrefl}_x$. First we want to extend φ to B. It is enough to define for any $p : C - \mathrm{Path}_X$ a map:

$$\psi(p) : B \to c_0(p) \rightsquigarrow c_1(p)$$

such that for any $a : A$ we have $\psi(p, u(a)) \equiv \varphi_a(p)$.

But since the path types in X are fibrant, and $u : A \to B$ is a pseudo-cofibration, the type of maps extending $\lambda a.\varphi_a(p)$ to B is fibrant. So by Lemma 6.16 it is enough to define ψ on hrefl^C_x for any $x : X$. In this case we just need to extend the constant function with value hrefl_x on A to B. We do this using the constant function.

Now we need to show that the for any $b : B$ the obtained maps:

$$\varphi_b : (p : C - \mathrm{Path}_X) \to c_0(p) \rightsquigarrow c_1(p)$$

have value hrefl_x on hrefl^C_x for any $x : X$. But this is true by the computation rule of Lemma 6.16. □

Now we are ready to show that fibrant loop spaces are ∞Mon-algebras.

Theorem 6.20. *Any fibrant loop spaces is an* ∞Mon*-algebra.*

Proof. By Lemma 6.6 we know that $\delta_k : \partial\,\mathbb{I}^k \to \mathbb{I}^k$ is a pseudo-cofibration for any $k : \mathbb{N}$, so by Lemma 6.19 the operad $\mathcal{P}ath_X$ is strongly contractible. Therefore we have a morphism from ∞Mon to $\mathcal{P}ath_X$ by Lemma 6.12, which gives the desired algebra structure by composing with the morphism from $\mathcal{P}ath_X$ to $\mathcal{E}nd_{x \rightsquigarrow x}$ of Lemma 6.18. □

We reassemble the pieces.

Theorem 6.21. *The operad* ∞Mon *acts on any fibrant type equivalent to a fibrant loop space.*

Proof. By Theorem 6.20 we know that ∞Mon acts on $x \rightsquigarrow x$ for $x : X$ and X fibrant. But we also know that it is cofibrant by Theorem 6.10, and hence that its algebras are invariant under equivalences between fibrant types by Theorem 4.11. □

This supports the claim that ∞Mon-algebras are monoids up to coherent homotopy.

7 Conclusion

We defined ∞-monoids in two-level type theory, and show some basic properties about them. We expect that these results can be transferred to some geometric settings (at least in simplicial and cubical sets). We believe this work illustrate how convenient two-level type theory can be, by allowing to internalize model-categoric methods to type theory. To my knowledge our proof that loop spaces are ∞-monoids is new. It is based on the type-theoretical idea that every coherence is built as a reflexivity using path induction, and it would be interesting to see how it can be formulated in the usual geometric langage.

The most natural and interesting way to extend our results would be to prove that any group-like ∞-monoid is equivalent to a loop space. This would require univalence of the universes of fibrant types. A straightforward approach of this problem would require to develop geometric realization of simplicial types, which is an interesting project in its own right.

Our work also has some limitations, most notably the choice of using two-level type theory. Indeed plain homotopy type theory can be interpreted in a large class of models called higher topoi, whereas it is not known whether two-level type theory admits such a rich semantic. This makes our proof less interesting to geometers. For type theorists, the problem of defining ∞-monoids (and more generally higher algebraic structures) in plain homotopy type theory is still wide open, and we do not claim any progress toward this goal.

References

[1] Danil Annenkov, Paolo Capriotti, and Nicolai Kraus. Two-level type theory and applications. *arXiv preprint arXiv:1705.03307*, 2017.

[2] Clemens Berger and Ieke Moerdijk. Axiomatic homotopy theory for operads. *Commentarii Mathematici Helvetici*, 78(4):805–831, 2003.

[3] Clemens Berger and Ieke Moerdijk. The Boardman–Vogt resolution of operads in monoidal model categories. *Topology*, 45(5):807–849, 2006.

[4] Michael Boardman and Rainer Vogt. *Homotopy invariant algebraic structures on topological spaces*. Springer, 1973.

[5] Simon Boulier and Nicolas Tabareau. Model structure on the universe in a two level type theory. 2016. `https://hal.archives-ouvertes.fr/hal-01579822`.

[6] Guillaume Brunerie. On the homotopy groups of spheres in homotopy type theory. *arXiv preprint arXiv:1606.05916*, 2016.

[7] Ulrik Buchholtz, Floris van Doorn, and Egbert Rijke. Higher groups in homotopy type theory. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 205–214. ACM, 2018.

[8] Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. Cubical type theory: a constructive interpretation of the univalence axiom. *arXiv preprint arXiv:1611.02108*, 2016.

[9] Daniel Licata and Guillaume Brunerie. $\pi_n(S^n)$ in homotopy type theory. In *International Conference on Certified Programs and Proofs*, pages 1–16. Springer, 2013.

[10] Daniel Licata and Michael Shulman. Calculating the fundamental group of the circle in homotopy type theory. In *2013 28th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 223–232. IEEE, 2013.

[11] Per Martin-Löf. An intuitionistic theory of types: Predicative part. In *Studies in Logic and the Foundations of Mathematics*, volume 80, pages 73–118. Elsevier, 1975.

[12] Per Martin-Löf. Constructive mathematics and computer programming. In *Studies in Logic and the Foundations of Mathematics*, volume 104, pages 153–175. Elsevier, 1982.

[13] Per Martin-Löf and Giovanni Sambin. *Intuitionistic type theory*. Bibliopolis Naples, 1984.

[14] Peter May. *The geometry of iterated loop spaces*. Springer, 1972.

[15] The univalent foundations program. *Homotopy type theory: univalent foundations of mathematics*. Institute for Advanced Study, 2013. `https://homotopytypetheory.org/book`.

[16] Vladimir Voevodsky. A simple type system with two identity types. 2013. `https://www.math.ias.edu/vladimir/sites/math.ias.edu.vladimir/files/HTS.pdf`.