On the Hierarchy of Univalent Universes: $\mathcal{U}_n$ is not n-Truncated

NICOLAI KRAUS CHRISTIAN SATTLER

November 2013

Abstract

In recent years, it has become clear that types in intensional Martin-Löf Type Theory can be seen as spaces, alternatively to the traditional view as sets or propositions. This observation motivated Voevodsky's *univalence axiom* and the development of a whole branch of mathematics, known as *Univalent Foundations* and *Homotopy Type Theory (HoTT)*.

One of the most basic consequences of univalence is that the type-theoretic universe $\mathcal{U}_0$ does not have unique identity proofs. We show a generalization of this result: universe $\mathcal{U}_n$ is not *n-truncated*, meaning that it has a non-trivial homotopical structure above dimension n. Our solution also answers the related (and so far open) problem of the Univalent Foundations Program in Princeton (2012/2013) of constructing a type that *strictly* has some high truncation level without using *higher inductive types*.

Further, we present a construction for the dual notion, *connectedness*. Given a type, we construct (in plain Martin-Löf Type Theory without HoTT-axioms) a type that is equivalent to the original one above a given dimension n. We show that it is trivial on all lower dimensions in the sense of a predicate we define, and that this predicate is equivalent to saying that the type is n-connected if the theory supports *truncations*.

We have fully formalized and verified our results within the dependently typed language and proof assistant Agda.

1 Introduction

Martin-Löf Type Theory (MLTT), as introduced and pushed forward by Martin-Löf [25–28], is a branch of mathematical logic with many applications in computer science, especially in the theory of programming languages. This form of type theory is powerful enough to serve as a framework for the formalization of huge parts of (constructive) mathematics. It serves as the basis of the dependently typed programming language Agda [29]. Better known is the proof assistant Coq, based on a variant of MLTT, the *Calculus of Inductive Constructions* [10]. Coq acquired much of its publicity in the mathematical community when it was utilized by Werner and Gonthier to formalize a proof of the famous Four color theorem [11]. For more recent work in Coq, we want to mention the Feit-Thompson odd order theorem [12] and the ForMath project [9].

An important aspect of type theory (intensional type theory, to be precise) is computation: *terms* are identified with their *normal forms*. For concrete im-

plementations, such as Agda and Coq, this means that we have an automatic simplification of expressions. Such a simplification would, in a proof on paper, have to be done manually by the mathematician, and we believe that the computational behavior of type theory can be seen as one of its main features that make it valuable for the mathematical community.

While the mathematical community seems to appreciate the existence of proof assistants in principle, their practical usage is still mostly restricted to those subjects that are close to logic, or, as in the case of the Four color theorem, those cases that require a case analysis so vast that it is unfeasible to do it by hand. Two reasons for that restriction are certainly the vast overhead that formalizations often require, and certain behaviors of type theory that are not understood sufficiently.

However, some years ago, progress in the semantics of MLTT lead to a development that has improved the situation with respect to both of these issues. Traditionally, a number of different views on types existed, including types as sets (Russel [31]) or propositions (Curry and Howard [17]); see [30] for a discussion. In addition to these, Awodey and Warren [5] and, independently, Voevodsky [36] found out that types may also be regarded as, roughly speaking, topological spaces. This new interpretation has helped to explain a lot of the behavior of MLTT regarding *equality types*. Voevodsky noted that in his *Simplicial Set model* (presentation by Streicher [34], and Kapulkin, Lumsdaine and Voevodsky [18], extending [19]) another interesting property is fulfilled: *equivalences* correspond to *equalities* of types. Motivated thereby, he introduced the *Univalence Axiom*. This axiom implies that isomorphic structures are actually equal and can directly be substituted for each other. This seems to be a key concept if we want type theory to be usable by working mathematicians as a tool for formal verification, or even for actually finding proofs, as mathematicians tend to identify isomorphic structures in informal proofs all the time. Hoping that type theory would finally be more accessible for mathematicians outside of the logic spectrum as he used to be himself, Voevodsky continued working on his *Univalent Foundations program*.

Soon, the ideas of Awodey, Warren and Voevodsky attracted many researchers from fields that were considered very different from type theory, such as higher dimensional category theory and abstract topology, fascinated by the surprising connection that allowed to transfer intuition, or even results, from one field into another. Traditional type theorists got interested because of the striking consequences of the univalence axiom, some of which had been considered feasible (but hard to realize) before. These direct consequences of univalence include function extensionality (considered, e. g., in [1]) and an extensional universe [15]. The homotopical view later induced the idea of *higher inductive types*, yielding very well-behaved quotient types (as previously considered in [2, 13, 14]) as a special case. In particular, the wish for properties that previously led to the development of *Observational Type Theory* [3, 4] are naturally satisfied, or conjectured to be satisfied, in type theory with the univalence axiom. Due to the homotopical nature of the type theory of interest, it became known as *Homotopy Type Theory* (HoTT), often used synonymously with *Univalent Foundations*.

The steady growth of interest culminated in the year-long special program on Univalent Foundations at the Institute of Advanced Study in Princeton 2012/2013, co-organized by Awodey, Coquand and Voevodsky, with around 60 participants, long- and short-term visitors. This was also where *Homotopy Type*

Theory: Univalent Foundations of Mathematics [35] was collaboratively written, in the community often referred to as "the HoTT book" or even as "the book", which will serve as our main reference in this article.

Especially during the program in Princeton, but also before and after, a lot of progress was made. In particular, the formalization of classical homotopy-theoretical theorems was pushed forward. The formalized part of homotopy theory includes the calculation of some homotopy groups of spheres, the van Kampen theorem, the Freudenthal suspension theorem, a restricted form of Whitehead's theorem, the Blakers-Massey theorem, and others, mostly reported in [35]. Noteworthy is that, due to the abstractness of HoTT, the proven results are not only true in the simplicial sets model (or, using the realization functor, the category of CW-complexes), but also in other appropriate models of weak ω-categories.

To explain how HoTT achieves these things, we have to go back a couple of steps, or equivalently, about one or two decades. One particularly interesting (and crucial) concept in MLTT is equality. Type theory knows two different forms of equality: first, there is the so-called *definitional* or *judgmental* equality, based on what we have just described: terms are identified if they behave identically from the computational point of view, meaning that they have the same normal form. In a more abstract sense, judgmental equality is a meta-theoretic concept of MLTT that is used for type checking. In intensional type theory, judgmental equality, and thus type checking, is decidable, a demand that corresponds to the very basic usage of proof assistants: if we have a proof a for a "proposition" A, the system should be able to check automatically whether a is indeed a *correct* proof of A. Judgmental equality in concrete implementations usually consists of β-equality and, optionally, some forms of η-equality. In contrast, extensional type theory does not distinguish between judgmental and propositional type theory, making type checking undecidable and therefore not suitable for the purposes that HoTT wants to serve. Consequently, HoTT is based on intensional MLTT.

As we want judgmental equality to be decidable, it is clear that this is a very strict notion of equality. Often, two mathematical objects are equal, but proving so can be arbitrarily hard. The corresponding terms in type theory will generally not be judgmentally equal, but *propositionally* equal: for any two terms a and b of the same type A, there is the type $\mathsf{Id}_A(a,b)$ of proofs that a and b are propositionally equal (later, we will often just write $a =_A b$ or even $a = b$). Propositional equality is thus an *internal* concept, making the formulation of mathematical theorems involving equality possible.

For some time, it was unknown whether *uniqueness of identity proofs* (UIP) is derivable, i. e. whether, given p and q of type $\mathsf{Id}_A(a,b)$, one can construct an inhabitant of the type $\mathsf{Id}_{\mathsf{Id}_A(a,b)}(p,q)$. This question was answered negatively by Hofmann and Streicher, who observed that type theory can be interpreted in the category of groupoids [15]. They also speculated that there might be models using higher groupoids, and even ω-groupoids, but were lacking an appropriate framework for the construction of such an interpretation.

UIP was often considered desirable: it was believed that a proof that a equals b should be the mere information thereof, without containing additional data. The homotopical view does not only show why UIP can not be derived nevertheless but also helps to explain what its absence means. A type can be seen as a topological space, and an equality proof can be understood as a *path* in

this space; but paths are, in general, not unique. However, there might be a path between paths, traditionally called a *homotopy*, and higher homotopies between homotopies, and so on, giving a space the structure of a *weak ω-groupoid*. As Lumsdaine [24] and, independently, van den Berg and Garner [7] explained, types do indeed carry the structure of a weak ω-groupoid.

In his PhD thesis, Warren [37] generalizes the Hofmann-Streicher groupoid model. Instead of ordinary groupoids, he uses *strict ω-groupoids* to model MLTT. He thereby proves that, for any n, the principle UIP_n can not be derived, where UIP_n is (the judgmental version of) the statement that, for any type A, iterating the process of taking two points and considering their path space $n-1$ times always leads to a type with unique identity proofs. In particular, he shows that having UIP_m for all types is strictly stronger than UIP_n if $m < n$.

Voevodsky's model in simplicial sets [36] can be understood as a further improvement of Warren's construction. Instead of strict ω groupoids, Voevodsky uses *Kan simplicial sets*, also known as *weak ω-groupoids*; and this is the model that motivated him to formulate the univalence axiom.

One of the most basic and well-known implications of the univalence axiom is that the first type universe, written $\mathcal{U}_0$, does not have unique identity proofs. This is due to the fact that, for an example, the type $\mathbf{2}$ of boolean values is isomorphic to itself in two different ways. These two isomorphisms give rise to two different inhabitants of $\mathsf{Id}_{\mathcal{U}_1}(\mathbf{2}, \mathbf{2})$. In the language of HoTT, this means that $\mathcal{U}_0$ is not a *set*, i. e., is not a 0-*type*. In general, a type A is called an $n+1$-*type* (or $n+1$-*truncated*) if, for all $a, b : A$, the type $\mathsf{Id}_A(a, b)$ is an n-type. Reading through the argument that $\mathcal{U}_0$ is not a 0-type, it seems plausible to assume that, as we go up the hierarchy of universes, we get types that can be shown to be not n-truncated for higher and higher n, meaning that they have a more and more complicated homotopical structure.

However, this turns out to be fairly involved. As sketched above, the type $\mathbf{2}$ is sufficient to see that $\mathcal{U}_0$ is not a 0-type. To go further, one idea that was suggested several times at the special year program in Princeton (first by Finster and Lumsdaine, as far as we know) was to consider the type of types that are *merely equal* to $\mathbf{2}$, written $\Sigma_{X:\mathcal{U}_0} \|X =_{\mathcal{U}_0} \mathbf{2}\|_{-1}$, where $\|-\|_{-1}$ is the *propositional truncation*. Technically, propositional truncation is a certain *higher inductive type*, but it can be encoded in a suitable way so that the construction can be expressed in pure MLTT with univalence, as required. The idea of $\Sigma_{X:\mathcal{U}_0} \|X =_{\mathcal{U}_0} \mathbf{2}\|_{-1}$ is to take $\mathbf{2}$, but "wrap" it once, defining something like the *subuniverse* of $\mathcal{U}_0$ that only contains $\mathbf{2}$. This operation shifts the non-trivial proof of $\mathbf{2} = \mathbf{2}$ by one level. Finster and Lumsdaine used the construction to show that $\mathcal{U}_1$ is not a 1-type. It seems plausible that the "wrapping" could be repeated in order to get the corresponding statements for higher universes, but this becomes difficult very quickly, and it is unclear whether the strategy could be used to prove the general statement. In this article, we will not need to consider truncations at all until we want to discuss *connectedness*.

Another argument for that fact that $\mathcal{U}_1$ is not a 1-type was given by Coquand, using the type of $\mathbb{Z}/2\mathbb{Z}$-sets: a set X, together with an endomorphism on X, and a proof that this endomorphism is self-inverse. It is not clear how a generalization of this construction could be used for higher cases, but in fact, the base case of our construction in this article is similar.

When the attempts to construct a type of non-trivial higher structure, or prove that universes have this property, remained inconclusive, Awodey added

the question to the internal list of open problems of the special year program.

In this article, we solve the problem by constructing a family of types, parametrized over $n \geq -1$, that are "strict" $n+1$-types ($n+1$-, but not n-truncated). It will then turn out that the "subuniverse" of $\mathcal{U}_n$ which only contains n-types is already such a strict $n+1$-type itself. Our construction also shows immediately that the universe $\mathcal{U}_n$ is not an n-type. From a homotopical point of view, this means that the first n homotopy groups of $\mathcal{U}_n$ are non-trivial. We say that they are *non-trivial in a high dimension* or *have a high truncation level*, even though both expressions are slightly inaccurate. Consequently, in MLTT with univalence, (the internal version of) $\neg\mathrm{UIP}_n$ can be shown internally by using sufficiently high universe levels.

After constructing these types that are non-trivial on a high dimension, we discuss types that are trivial on low dimensions: given a type and a number n we construct, in plain MLTT, a type that is trivial on dimension n and below, while the higher dimensions remain untouched. In HoTT, such a type is called *n-connected*, a notion that involves certain *higher inductive types*. We define a version of n-connectedness without referring to higher inductive types which is fulfilled by the type we constructed and which is equivalent to the usual n-connectedness in HoTT.

Combining both of our constructions, we present a family M_n of types, indexed over the natural numbers and constructed in MLTT from univalence alone, such that M_n is trivial on all dimensions except from dimension n. We also discuss how our constructions can be understood as a usage of higher inductive types in a theory that does actually not support them.

The first-named author has presented two of the main results of this article at the IAS in Princeton in April 2013. We have completely formalized our construction in Agda [20] (see Section 2.4).

Contents In Section 2, we introduce some basic notions and properties of MLTT and HoTT. The theory that we work in is standard intensional MLTT with a hierarchy of universes, and we assume all universes to be univalent. In particular, this section introduces all concepts of HoTT that we need, and no prior knowledge of HoTT is required. On the other hand, the reader with background in HoTT can skip that section completely, maybe apart from Subsection 2.3 where we present a proof that $\mathcal{U}_1$ is not a 1-type. We use Section 3 to develop some simple, but very useful theory on pointed types and the interaction of dependent sums, dependent products (which we synonymously call dependent functions), and loop spaces. Section 4 contains two of our main results: we show that universe $\mathcal{U}_n$ is not n-truncated, and we construct a type that is "strictly" of truncation level $n+1$, i. e. in particular not of level n. We devote Section 5 to *connectedness*, which is, in some sense, the dual notion of truncatedness. Intuitively, a type is n-truncated if it is trivial on dimension n and below. We define a notion of n-connectedness in plain MLTT without using higher inductive types, and we show that our notion is (in HoTT) equivalent to the standard definition of n-connectedness. For any inhabited type A and number n, we can construct a type of which we show that it is n-connected in our sense but which is equivalent to the original type on levels above n. Finally, in Section 6, we combine the two constructions, and discuss how they are motivated by higher inductive types.

2 Preliminaries

We work in Martin-Löf Type Theory with the univalence axiom, as introduced by Voevodsky [36]. Apart from the univalence axiom, our type theory is standard intensional MLTT. Here, we give a very brief description of the needed fragment; for a comprehensive formal presentation see [35, Appendix A.1 or A.2]. For the univalence axiom, we give a slightly more detailed introduction. An extensive explanation can, again, be found in [35, chapter 2.10]. In the main part of this article, we do not assume that our theory has any other features that are often considered in Homotopy Type Theory; however, we will discuss *higher inductive types* in Sections 5 and 6. Regarding notation, we stick very close to our main reference [35].

2.1 Martin-Löf Type Theory

MLTT knows the following concepts, which we will all use:

Universes There is a hierarchy $\mathcal{U}_0, \mathcal{U}_1, \mathcal{U}_2, \ldots$ of universes, where $\mathcal{U}_0$ is the first universe and, for all n, we have $\mathcal{U}_n : \mathcal{U}_{n+1}$. This hierarchy is assumed to be *cumulative*: if $m < n$ and $A : \mathcal{U}_m$, then $A : \mathcal{U}_n$. If we just write $\mathcal{U}$, the corresponding statement or derivation is to be understood for *any* universe.

Dependent Sums Given types $A : \mathcal{U}$ and $B : A \to \mathcal{U}$, there is a type $\Sigma_A B : \mathcal{U}$, the inhabitants of which are pairs (a, b) with $a : A$ and $b : B(a)$. We also write $\Sigma_{a:A} B(a)$ if it increases the readability.

Dependent Products If $A : \mathcal{U}$ and $B : A \to \mathcal{U}$ are as above, $\Pi_A B : \mathcal{U}$, or synonymously $\Pi_{a:A} B(a)$, is the type of dependent functions. An inhabitant is a function f such that $f(a) : B(a)$ for all $a : A$. If B does not depend on A, we write $A \to B$ instead of $\Pi_A B$.

Finite Types and Natural Numbers There are the empty type $\mathbf{0}$, the unit type *unit* with canonical inhabitant $\star$, the boolean type $\mathbf{2}$ with canonical inhabitants $0_\mathbf{2}$ and $1_\mathbf{2}$, and the natural numbers $\mathbb{N}$ in the lowest universe $\mathcal{U}_0$ (and thereby in every universe). We write $\mathsf{swap} : \mathbf{2} \to \mathbf{2}$ for the negation function.

Identity Types For a type $A : \mathcal{U}$ and inhabitants $a, b : A$, there is the type $\mathsf{Id}_A(a, b) : \mathcal{U}$ of *equality proofs* between a and b, or *paths* from the *point* a to the *point* b. The reason for the latter terminology is the interpretation of types as spaces. The equality type itself is also called *path space*. The canonical inhabitants are given by $\mathsf{refl}_a : \mathsf{Id}_A(a, a)$ for $a : A$. The elimination rule (named J) says that, given a type family $P : (\Sigma_{a,b:A} \mathsf{Id}_A(a, b)) \to \mathcal{U}$, it is sufficient to construct $P(a, a, \mathsf{refl}_a)$ for any a in order to show $P(a, b, p)$ for any a, b and p. Applying this principle is often paraphrased as *path induction*. Following [35], we write $a =_A b$ for this type. We omit the type A if it can easily be inferred.

As first described by Hofmann and Streicher [15], the identity type carries the structure of a *groupoid*, i. e. a category in which every morphism is an isomorphism. Every path $p : a =_A b$ has an inverse $p^{-1} : b =_A a$, and all paths $p : a =_A b$ and $q : b =_A c$ have a composition $p \cdot q : a =_A c$, with refl_a acting as

the identity element of $a =_A a$. These operations can be constructed using path induction. The groupoid laws between them hold again up to to propositional equality. Finally, the non-dependent version of the eliminator deserves to be mentioned, in HoTT usually called *transportation* along a path, and in general often referred to as *substitution*: if $P : A \to \mathcal{U}_k$ is a family of types over A and there are $u : a =_A b$ as well as $t : P(a)$, then there is $u_*(t) : P(b)$.

As "$=$" is thus reserved for propositional equality, we use "$\equiv$" for judgmental (i.e., definitional) equality. In particular, we write "$:\equiv$" if we are giving a definition. Whenever we say that two terms are equal, propositional equality is meant.

2.2 Homotopy Type Theory

Let us recall some basic notions of HoTT. All of the following, except the univalence axiom below, are definitions that are only made within the previously described setting. They do not depend on any additional assumptions.

Truncation Levels (n-Types) A type A is *contractible* if there is a point $a : A$ such that all points are equal to a,

$$\mathsf{isContr}(A) :\equiv \Sigma_{a:A}\Pi_{b:A} a = b.$$

For $n \geq -2$, the statement that A is an *n-type*, is *n-truncated*, or has *truncation level n*, is defined as follows. For $n \equiv -2$, we take $\mathsf{isContr}(A)$ as the definition. Otherwise, the meaning is that all path spaces over elements of A are of truncation level one lower,

$$\begin{aligned} \mathsf{is}\text{-}(-2)\text{-}\mathsf{type}(A) &:\equiv \mathsf{isContr}(A) \\ \mathsf{is}\text{-}(n+1)\text{-}\mathsf{type}(A) &:\equiv \Pi_{x,y:A}\mathsf{is}\text{-}n\text{-}\mathsf{type}(x =_A y) \end{aligned}$$

For $n \equiv -1$ and $n \equiv 0$, there are very common synonyms to "n-type" and "n-truncated". By a standard lemma, A is a -1-type if and only if all its inhabitants are equal, i. e. if $\Pi_{a,b:A} a = b$ is inhabited. Such a type is called a *proposition* and has the property of being *propositional*. Further, a 0-type is a type with unique identity proofs; those types are called *sets*. For $n \equiv -2, -1, 0$, instead of $\mathsf{is}\text{-}n\text{-}\mathsf{type}(A)$, we will therefore write $\mathsf{isContr}(A), \mathsf{isProp}(A), \mathsf{isSet}(A)$, respectively.

Equivalences Let A, B be types and $f : A \to B$ be a non-dependent function. There are several equivalent ways to characterize f as an *equivalence*. One is by requiring that every fiber of f is contractible. Another possibility is asking for an inverse of f with a *coherent* pair of proofs, making f a so-called "half adjoint equivalence". An extensive treatment of equivalences can be found in [35, chapter 4]. Here, we only use the result that, for any non-dependent function f, there is a term $\mathsf{isequiv}_{A,B}(f)$ with the following properties:

- $\mathsf{isequiv}_{A,B}(f)$ is a proposition (a -1-type).
- $\mathsf{isequiv}_{A,B}(f)$ is inhabited if and only if f has an "inverse", that is a map $g : B \to A$ together with a proof of $\Pi_{a:A} g(f(a)) = a$ and a proof of $\Pi_{b:B} f(g(b)) = b$.

We write $A \simeq B$ for the type $\Sigma_{f:A\to B}\mathsf{isequiv}_{A,B}(f)$. If this type is inhabited, we say that A and B are equivalent. For example, the identity id_A is an equivalence, and so is the function swap defined above. We write e_{id} and $\mathsf{e}_{\mathsf{swap}}$ for the (up to propositional equality) unique inhabitants of $\mathsf{isequiv}(\mathsf{id}_A)$ and $\mathsf{isequiv}(\mathsf{swap})$, respectively. The pair $(\mathsf{id}_A, \mathsf{e}_{\mathsf{id}})$ of type $A \simeq A$ is called the *identity equivalence*.

Analogous to equality on some fixed type, equivalence induces a groupoid structure on universes.

Pointed types If A is a type and $a : A$ one of its elements, then (A, a) is called a *pointed type* with *underlying type* A and *basepoint* a. Let us write $\mathcal{U}_\bullet$ for the type of pointed types with underlying type living in $\mathcal{U}$ [35, Definition 2.1.7], that is,

$$\mathcal{U}_\bullet :\equiv \Sigma_{A:\mathcal{U}} A.$$

We call $\mathcal{U}_\bullet$ the *universe of pointed types* as this matches the intuition. Note, however, that it is really just a defined type, rather than a primitive of the theory as the universes $\mathcal{U}_k$ are. If (A, a) and (B, b) are pointed types, a *pointed function* consists of a map $f : A \to B$ and a proof of $f(a) = b$, showing that the basepoint is preserved. If additionally f is an equivalence, we speak of a *pointed equivalence*.

We call a pointed type n-truncated (or an n-type, or say that it has truncation level n) if its underlying type has that property.

Loop spaces Let (A, a) be a pointed type. Its *loop space* [35, chapter 2.1] the pointed type

$$\Omega(A, a) :\equiv ((a =_A a), \mathsf{refl}_a),$$

the elements of which are called *loops*. As Ω is thus an endomorphism on $\mathcal{U}_\bullet$, it can be composed with itself. This gives us the *n-fold iterated loop space*

$$\Omega^0(A, a) \equiv (A, a)$$
$$\Omega^{n+1}(A, a) \equiv \Omega^n(\Omega(A, a)).$$

To gain intuition for $\Omega^{n+1}(A, a)$, we can unfold the definition. This shows immediately that the underlying type is $\mathsf{refl}_a^n = \mathsf{refl}_a^n$, while the point is the canonical inhabitant of the underlying type, namely refl_a^{n+1}.

Remark. Note that we consider addition on $\mathbb{N}$ to be defined by recursion on the *second* argument to allow the presentation to follow the traditional convention of writing $n + 1$ instead of $1 + n$. Some of the equalities that we claim to hold judgmentally depend on this assumption.

Univalence For types A and B, there is a canonical map

$$\mathsf{idtoeqv} : (A = B) \to (A \simeq B),$$

defined by path induction, where $\mathsf{refl}_A : A = A$ is mapped to the identity equivalence $(\mathsf{id}_A, \mathsf{e}_{\mathsf{id}})$. This allows us to formulate the core axiom of HoTT:

Univalence Axiom: The map $\mathsf{idtoeqv}$ is an equivalence.

It implies that the type of paths between types is equivalent to the type of their equivalences,

$$(A =_{\mathcal{U}} B) \simeq (A \simeq B).$$

As is standard, we assume the univalence axiom for every universe $\mathcal{U}_k$, i. e. all universes are assumed *univalent*.

Adding the univalence axiom to the theory destroys *canonicity*: there are closed terms of type $\mathbb{N}$ that are not in normal form. It is conjectured that some form of canonicity relative to propositional equality can be recovered. While many formalizations of mathematical theorems have already been performed, many of them could have been easier if there had been an appropriate computation rule for the univalence axiom. Finding a computational meaning, i. e. reduction rules for univalence, is arguable one of the most important current open problems in HoTT.

It is a well-known and immediate consequence of the univalence axiom that the smallest universe is not a set (a 0-type). The standard proof goes as follows. Suppose $\mathsf{isSet}(U_0)$. Then, by definition of isSet, we have $\mathsf{isProp}(\mathbf{2} = \mathbf{2})$. By univalence, we may replace $\mathbf{2} = \mathbf{2}$ by $\mathbf{2} \simeq \mathbf{2}$. However, there are two distinct automorphisms on $\mathbf{2}$, yielding a contradiction. In formulae:

$$\begin{aligned}
\mathsf{isSet}(\mathcal{U}_0) &\implies \mathsf{isProp}(\mathbf{2} = \mathbf{2}) \\
&\implies \mathsf{isProp}(\mathbf{2} \simeq \mathbf{2}) \\
&\implies (\mathsf{id}_{\mathbf{2}}, \mathsf{e}_{\mathsf{id}}) = (\mathsf{swap}, \mathsf{e}_{\mathsf{swap}}) \\
&\implies \mathsf{id}_{\mathbf{2}} = \mathsf{swap} \\
&\implies \mathsf{id}_{\mathbf{2}}(1_{\mathbf{2}}) = \mathsf{swap}(1_{\mathbf{2}}) \\
&\implies 1_{\mathbf{2}} = 0_{\mathbf{2}} \\
&\implies \bot.
\end{aligned}$$

Intuitively, it may appear that the reason why $\mathcal{U}_0$ is not a set is that an inhabitant of it, namely $\mathbf{2}$, is already not a proposition. However, possibly somewhat surprisingly, this simple idea does not generalize and the proof of $\neg\mathsf{is\text{-}1\text{-}type}(\mathcal{U}_1)$ already requires significantly more thought.

To prove the general version $\mathsf{is}\text{-}n\text{-}\mathsf{type}(\mathcal{U}_n)$ for any chosen n, we will develop some theory about pointed types. An important ingredient will be our *local-global looping principle*, allowing us to freely switch between (higher) loops in the universe and families of loops that are indexed over some type.

We will make frequent use of the following basic properties. Most of them are directly stated in [35], and the remaining ones are easy to prove using standard techniques.

Properties 1. (P1) *Truncation level properties are propositional*: the type $\mathsf{is}\text{-}n\text{-}\mathsf{type}(A)$ is a proposition. [35, Theorem 7.1.10]

(P2) *Truncation levels are upwards closed*: if $n \leq n'$, then $\mathsf{is}\text{-}n\text{-}\mathsf{type}(A)$ implies $\mathsf{is}\text{-}n'\text{-}\mathsf{type}(A)$. [35, Theorem 7.1.7]

(P3) Σ *preserves truncation level*: for a type B that may depend on some type A, if A and $B(a)$ with $a : A$ are n-types, then so is $\Sigma_{a:A} B(a)$. In particular, the product of two n-types is an n-type. [35, Theorem 7.1.8]

(P4) Π *preserves truncation level*: if $B(a)$ is an n-type for every a, then $\Pi_A B$ is also an n-type. [35, Theorem 7.1.9]

(P5) *Strong function extensionality*: for any two functions $f, g : \Pi_A B$, there is a canonical map from $f = g$ to $\Pi_{a:A}(f(a) = g(a))$ (usually called happly), defined by path induction. This map is an equivalence. [35, Chapter 4.9]

(P6) *Functions can be applied on equalities*: if we have $f : A \to B$, then $a = b$ implies $f(a) = f(b)$. This function is called ap (**a**ction on **p**aths) and will be used frequently without explicit mention. [35, Lemma 2.2.1]

(P7) *Equivalences preserve path spaces*: if $f : X \to Y$ is an equivalence and $x_1, x_2 : X$, then $x_1 =_X x_2 \simeq f(x_1) =_Y f(x_2)$. [35, Theorem 2.11.1]

(P8) *Transporting paths along path is composition*: given $t : a = b$ and $p : a = a$, the term $t_*(p) : b = b$ is equal to $t^{-1} \cdot p \cdot t$. [35, Theorem 2.11.5]

(P9) *Paths between pairs are pairs of paths*: if (x_1, y_1) and (x_2, y_2) are both of type $\Sigma_X Y$, then $(x_1, y_1) = (x_2, y_2) \simeq \Sigma_{u:x_1=x_2} u_*(y_1) = y_2$. In the case of a non-dependent product $X \times Y$, the latter type simplifies to $(x_1 = x_2) \times (y_1 = y_2)$. A special application of this rule concerns pointed types: Univalence tells us that, for pointed types X and Y, the type of pointed equivalences between them is equivalent to the type of equalities $X = Y$. [35, Theorem 2.7.2]

(P10) *Neutral contractible components*: if Y depends on X and $Y(x)$ is contractible for all x, then $\Sigma XY \simeq X$. [35, Lemma 3.11.9 (i)]

(P11) *Neutral contractible exponents*: if Y depends on X, where X is some contractible type with inhabitant $x_0 : X$, then $\Pi_X Y \simeq Y(x_0)$. Similarly, we have $\Sigma_X Y \simeq Y(x_0)$. [35, Lemma 3.11.9 (ii)]

2.3 The second universe is not a 1-type

In this subsection, we want to present the proof that $\mathcal{U}_1$ is not 1-truncated. We will not reuse this result later as it will easily follow from more general constructions. However, the approach we take for this special case contains some of the key ideas and could therefore be supportive for understanding the later developments.

Let us first try to prove $\neg\mathsf{is\text{-}1\text{-}type}(\mathcal{U}_1)$ in a similar way as we have proved $\neg\mathsf{isSet}(\mathcal{U}_0)$ in Subsection 2.2 above, choosing two inhabitants of $\mathcal{U}_1$ that seem homotopically complicated enough:

$$\mathsf{is\text{-}1\text{-}type}(\mathcal{U}_1) \implies \mathsf{isSet}(\mathcal{U}_0 = \mathcal{U}_0)$$

(by univalence)

$$\implies \mathsf{isSet}(\mathcal{U}_0 \simeq \mathcal{U}_0)$$

(choose two inhabitants of $\mathcal{U}_0 \simeq \mathcal{U}_0$)

$$\implies \mathsf{isSet}((\mathsf{id}_{\mathcal{U}_0}, \mathsf{e}_{\mathsf{id}}) = (\mathsf{id}_{\mathcal{U}_0}, \mathsf{e}_{\mathsf{id}}))$$
$$\implies \ldots ?$$

In the attempt above, in the very first step, we have to choose two inhabitants of $\mathcal{U}_1$ with sufficiently complicated equality type. We have chosen $\mathcal{U}_0$ as we have already seen before that $\mathcal{U}_0$ is not a set.

The problem is that we seem unable to derive a contradiction from the assumption $\mathsf{isSet}(\mathcal{U}_0 \simeq \mathcal{U}_0)$. In fact, an expected meta-theoretic result is that the identity is the only definable non-constant endofunction on $\mathcal{U}_0$. Because of this, it is to be assumed that not even $\mathsf{isContr}(\mathcal{U}_0 \simeq \mathcal{U}_0)$ implies a contradiction. To the best of our knowledge, no one has rigorously proven this form of *parametricity* in the presence of univalence so far, but it is commonly believed to hold.

The problematic step in the above attempt is the first one, where we need to choose something in $\mathcal{U}_1$ and take $\mathcal{U}_0$. We have to choose something better behaved. We use the type of loops in $\mathcal{U}_0$,

$$L :\equiv \Sigma_{X:\mathcal{U}_0} X = X.$$

Showing that the second universe is not a groupoid proceeds as follows:

$$\mathsf{is\text{-}1\text{-}type}(\mathcal{U}_1) \implies \mathsf{isSet}(L = L)$$

(by univalence)

$$\implies \mathsf{isSet}(L \simeq L)$$

(choose the identity)

$$\implies \mathsf{isProp}((\mathsf{id}_L, \mathsf{e}_{\mathsf{id}}) = (\mathsf{id}_L, \mathsf{e}_{\mathsf{id}}))$$

Here, we have the type of paths between pairs. By P9, this corresponds to a pair of paths. What we use is that the second component will be trivial: e_{id} lives in a propositional type, and its path type will thus be contractible. This implies that the type of paths between two equivalences is equivalent to the type of paths between the underlying functions.

$$\implies \mathsf{isProp}(\mathsf{id}_L = \mathsf{id}_L)$$

(by strong functional extensionality)

$$\implies \mathsf{isProp}(\Pi_{a:L} a = a)$$

(unfold the definition of L and curry for readability)

$$\implies \mathsf{isProp}(\Pi_{X:\mathcal{U}_0} \Pi_{p:X=X} (X, p) = (X, p))$$

(by P9, paths between pairs are pairs of paths)

$$\implies \mathsf{isProp}\,(\Pi_{X:\mathcal{U}_0}\Pi_{p:X=X}\Sigma_{q:X=X}\ q_*(p) = p)$$

By P8, transportation of a path along a path can be written as path composition: $q_*(p) = q^{-1} \cdot p \cdot q$. Making this replacement and precomposing with q, we get $\mathsf{isProp}(K)$ where

$$K \ :\equiv\ \Pi_{X:\mathcal{U}_0}\Pi_{p:X=X}\Sigma_{q:X=X}\ p \cdot q = q \cdot p.$$

Two inhabitants of K are

$$\alpha :\equiv \lambda X.\lambda p.(\mathsf{refl}_X, u),$$
$$\beta :\equiv \lambda X.\lambda p.(p, \mathsf{refl}_{p \cdot p})$$

where u is a proof of $p \cdot \mathsf{refl}_X = \mathsf{refl}_X \cdot p$. Since K is propositional, we may conclude $\alpha =_K \beta$ and consequently $\pi_1(\alpha(\mathbf{2})) = \pi_1(\beta(\mathbf{2}))$, which evaluates to

$$\lambda p.\mathsf{refl}_{\mathbf{2}} =_{\mathbf{2}=\mathbf{2}\to\mathbf{2}=\mathbf{2}} \lambda p.p$$

and, after replacing $\mathbf{2} = \mathbf{2}$ by $\mathbf{2} \simeq \mathbf{2}$ and applying on $(\mathsf{swap}, \mathsf{e}_{\mathsf{swap}})$, implies the same contradiction as we got in the proof of $\neg\mathsf{isSet}(\mathcal{U}_0)$.

In the general case, we consider higher loops in higher universes. The core obstacle in translating the above proof is the step where $q_*(p) = p$ is observed to hold for $q :\equiv \mathsf{refl}$ and $q :\equiv p$ by virtue of $q_*(p) = q^{-1} \cdot p \cdot q$. In general, it is not so clear how a uniform presentation of transportation along higher loops would look like. However, it turns out that this obstacle can be effectively bypassed for higher dimensions as we will see below.

2.4 A note on our formalized proof

Supplementing this article, we have formalized all of our results in the programming language and proof assistant *Agda* [29], making use of the HoTT community's Agda library [16]. The formalization [20] and all of its dependencies are available in browser-viewable format and as plain source code on the first-named author's academic homepage. All proofs have been verified to type check in Agda version 2.3.3.2.

Deserving mention is a subtle difference between the type theory commonly used to develop HoTT [35], also used in this article, and the one that Agda implements. While HoTT universes are *cumulative*, i.e. $A : \mathcal{U}$ and $\mathcal{U} : \mathcal{U}'$ imply $A : \mathcal{U}'$, Agda requires explicit *lifting*. A sour consequence of this is the following: recall that the univalence axiom implies $(A = B) =_{\mathcal{U}_{k+1}} (A \simeq B)$ for types $A, B : \mathcal{U}_k$. Note however that this cannot be stated in Agda, the reason being that $A = B$ lives in $\mathcal{U}_{k+1}$ while $A \simeq B$ lives in $\mathcal{U}_k$. We can still state this proposition by first lifting $A \simeq B$ to the universe $\mathcal{U}_{k+1}$. To avoid the readability of our code being impacted by manifold instances of lifting, we have strived to represent type (pointed type) equality by (pointed) equivalence wherever possible in the formalization.

3 Pointed Types

Pointed types, as defined in [35], are a simple but helpful concept. Their properties can usually easily be formulated in terms of ordinary types. For our presentation we will develop some of their theory explicitly in this section, aiming to provide an elegant way of expressing how Ω interacts with Σ and Π.

3.1 Dependent Sums and Loops

We will first treat the interaction of Σ and Ω. Let us begin by recalling the following definition:

Definition 3.1 (pointed predicate [35, Definition 5.8.1])**.** For a pointed type $\mathfrak{A} \equiv (A, a)$, a *pointed predicate* is a type family $P : A \to \mathcal{U}$ where the type over the basepoint is again pointed:

$$\mathrm{Pred}^\bullet_{\mathfrak{A}} :\equiv \Sigma_{P:A\to\mathcal{U}} P(a).$$

Extending the notion of truncatedness from types to predicates, we say that the pointed predicate (P, p) is n-truncated if P is a family of n-types.

Remark. Note that a pointed type can always be seen as a pointed predicate over the trivial pointed type $(\mathbf{1}, \star)$.

Let (P, p) be a pointed predicate over some pointed type (A, a). There is an induced type family $\tilde{P}$ over $\Omega(A, a)$, given by $\tilde{P}(q) :\equiv q_*(p) =_{P(a)} p$. The type over the basepoint is $P(\mathsf{refl}_a) \equiv (p = p)$ and therefore trivially inhabited by reflexivity. This allows us to define a fibered version of Ω:

Definition 3.2 ($\tilde{\Omega}$)**.** For a pointed type $\mathfrak{A} \equiv (A, a)$, we define

$$\tilde{\Omega} : \mathrm{Pred}^\bullet_{\mathfrak{A}} \to \mathrm{Pred}^\bullet_{\Omega\mathfrak{A}}$$
$$\tilde{\Omega}(P, p) :\equiv \left(\lambda q.q_*(p) =_{P(a)} p\ ,\ \mathsf{refl}_p\right).$$

Consequently, Ω and $\tilde{\Omega}$ together form the following endofunction:

$$\langle \Omega, \tilde{\Omega}\rangle : \Sigma_{\mathfrak{A}:\mathcal{U}_\bullet} \mathrm{Pred}^\bullet_{\mathfrak{A}} \to \Sigma_{\mathfrak{A}:\mathcal{U}_\bullet} \mathrm{Pred}^\bullet_{\mathfrak{A}}$$
$$\langle \Omega, \tilde{\Omega}\rangle(\mathfrak{A}, \mathfrak{P}) \equiv (\Omega\mathfrak{A}, \tilde{\Omega}\mathfrak{P})$$

Given a pair of a pointed type and a pointed predicate, it is straightforward to construct a pointed type corresponding to the dependent sum.

Definition 3.3 ($\Sigma^\bullet$)**.** We define the operator $\Sigma^\bullet$ in the following way:

$$\Sigma^\bullet : (\Sigma_{\mathfrak{A}:\mathcal{U}_\bullet} \mathrm{Pred}^\bullet_{\mathfrak{A}}) \to \mathcal{U}_\bullet$$
$$\Sigma^\bullet((A, a), (P, p)) :\equiv (\Sigma_A P\ ,\ (a, p))$$

We write $\Sigma^\bullet_{\mathfrak{A}}\mathfrak{P}$ synonymously for $\Sigma^\bullet(\mathfrak{A}, \mathfrak{P})$.

We are now ready to formulate precisely how dependent sums and loop spaces interact:

Lemma 3.4. *The operators $\Sigma^\bullet$ and Ω commute in the following sense:*

$$\Omega \circ \Sigma^\bullet = \Sigma^\bullet \circ \langle \Omega, \tilde{\Omega}\rangle.$$

Proof. Let $\mathfrak{A} \equiv (A, a)$ be a pointed type with a pointed predicate $\mathfrak{P} \equiv (P, p)$. By function extensionality (P5), it is enough to show that both sides of the equation are equal if applied to $(\mathfrak{A}, \mathfrak{P})$. Let us calculate:

$$\begin{aligned}
& & & (\Omega \circ \Sigma^\bullet)(\mathfrak{A}, \mathfrak{P}) \\
& \text{(by definition of } \Sigma^\bullet) & \equiv & \ \Omega\left(\Sigma_A P\ ,\ (a, p)\right) \\
& \text{(by definition of } \Omega) & \equiv & \ \left((a,p) = (a,p)\ ,\ \mathsf{refl}_{(a,p)}\right) \\
& \text{(by P9)} & = & \ (\Sigma_{q:a=a} q_*(p) = p\ ,\ (\mathsf{refl}_a, \mathsf{refl}_p)) \\
& \text{(with } \tilde{P} \text{ as above)} & \equiv & \ \left(\Sigma_{q:a=a} \tilde{P}(q)\ ,\ (\mathsf{refl}_a, \mathsf{refl}_p)\right) \\
& \text{(by definition of } \Sigma^\bullet) & \equiv & \ \Sigma^\bullet_{(a=a,\mathsf{refl}_a)} (\lambda q.q_*(p) =_{P(a)} p, \mathsf{refl}_p) \\
& \text{(by definition of } \Omega \text{ and } \tilde{\Omega}) & \equiv & \ (\Sigma^\bullet \circ \langle \Omega, \tilde{\Omega} \rangle)(\mathfrak{A}, \mathfrak{P}).
\end{aligned}$$

□

3.2 Dependent Products and Loops

The situation is similar, and even simpler, if we want to examine the interaction of Π and Ω. Given a family of pointed types over some (ordinary) type A, there is a straightforward way to construct a pointed type out of the given data corresponding to the dependent product.

Definition 3.5 ($\Pi^\bullet$)**.** We define the operator $\Pi^\bullet$ by:

$$\begin{aligned}
\Pi^\bullet &: (\Sigma_{A:\mathcal{U}}(A \to \mathcal{U}_\bullet)) \to \mathcal{U}_\bullet \\
\Pi^\bullet(A, \mathfrak{F}) &:\equiv (\Pi_A \pi_1 \circ \mathfrak{F}\ ,\ \pi_2 \circ \mathfrak{F})
\end{aligned}$$

We use the notations $\Pi^\bullet_{a:A} \mathfrak{F}(a)$ and $\Pi^\bullet_A \mathfrak{F}$ synonymously with $\Pi^\bullet(A, \mathfrak{F})$.

With this at hand, we are ready to prove:

Lemma 3.6. *Ω and $\Pi^\bullet$ commute in the following sense: given a type A and a family $\mathfrak{F}$ of pointed types over A, we have*

$$\Omega(\Pi^\bullet(A, \mathfrak{F})) = \Pi^\bullet(A, \Omega \circ \mathfrak{F}).$$

Proof. Let us do the following calculation:

$$\begin{aligned}
& & & \Omega(\Pi^\bullet(A, \mathfrak{F})) \\
& \text{(by definition of } \Pi^\bullet) & \equiv & \ \Omega\left(\Pi_A \pi_1 \circ \mathfrak{F}\ ,\ \pi_2 \circ \mathfrak{F}\right) \\
& \text{(by definition of } \Omega) & \equiv & \ (\pi_2 \circ \mathfrak{F} = \pi_2 \circ \mathfrak{F}\ ,\ \mathsf{refl}) \\
& \text{(by P5)} & = & \ \Pi_{a:A}\left(\pi_2(\mathfrak{F}(a)) = \pi_2(\mathfrak{F}(a))\ ,\ \lambda a.\mathsf{refl}\right) \\
& \text{(by definition of } \Pi^\bullet) & \equiv & \ \Pi^\bullet(A, \Omega \circ \mathfrak{F})
\end{aligned}$$

□

4 Homotopically Complicated Types

In this section, we will prove two main results of this article: first, in MLTT with the univalence axiom, we can construct a type that *strictly* has truncation level n; and second, the universe $\mathcal{U}_n$ is not n-truncated.

We begin with a lemma that tells us how a truncated Σ-component can be neutralized by Ω.

Lemma 4.1. *Let n be a natural number. Further, let $\mathfrak{A}$ be a pointed type and $\mathfrak{P}$ be a pointed predicate over $\mathfrak{A}$ of truncation level $n-2$. Then,*

$$\Omega^n(\Sigma^\bullet_{\mathfrak{A}}\mathfrak{P}) = \Omega^n(\mathfrak{A}).$$

Proof. We do induction on n. For the base case $n \equiv 0$, the statement is exactly given by P10. For the induction case, we have the following chain of equalities:

$$\begin{aligned}
& \Omega^{n+1}(\Sigma^\bullet_{\mathfrak{A}}\mathfrak{P}) \\
\equiv\ & \Omega^n(\Omega(\Sigma^\bullet_{\mathfrak{A}}\mathfrak{P})) \\
(\text{by Lemma 3.4}) \quad =\ & \Omega^n(\Sigma^\bullet_{\Omega\mathfrak{A}}\tilde{\Omega}\mathfrak{P}) \\
(\text{by induction hypothesis}) \quad =\ & \Omega^n(\Omega(\mathfrak{A})) \\
\equiv\ & \Omega^{n+1}(\mathfrak{A})
\end{aligned}$$

For the second to last step, note that if $\mathfrak{P}$ is $n-1$-truncated, then $\tilde{\Omega}\mathfrak{P}$ is $n-2$-truncated. □

We are now ready to prove our local-global looping principle, stating that a loop in the universe is the same as a family of loops in its underlying type:

Lemma 4.2 (local-global looping)**.** *Let A be a type and n be a natural number. Then,*

$$\Omega^{n+2}(\mathcal{U}, A) = \Pi^\bullet_{a:A}\Omega^{n+1}(A, a).$$

Proof. The proof is again done by a calculation, utilizing most of the theory we have developed so far:

$$\begin{aligned}
& \Omega^{n+2}(\mathcal{U}, A) \\
(\text{by definition}) \quad \equiv\ & \Omega^{n+1}\left(A = A\ ,\ \mathsf{refl}_A\right) \\
(\text{by univalence and P9}) \quad =\ & \Omega^{n+1}\left(A \simeq A\ ,\ (\mathsf{id}_A, \mathsf{e}_{\mathsf{id}})\right) \\
(\text{by definition of } \simeq) \quad \equiv\ & \Omega^{n+1}\left(\Sigma_{f:A\to A}\mathsf{isequiv}(f)\ ,\ (\mathsf{id}_A, \mathsf{e}_{\mathsf{id}})\right) \\
(\text{by definition of } \Sigma^\bullet) \quad \equiv\ & \Omega^{n+1}(\Sigma^\bullet_{(A\to A, \mathsf{id}_A)}(\mathsf{isequiv}, \mathsf{e}_{\mathsf{id}})) \\
(\text{by Lemma 4.1}) \quad =\ & \Omega^{n+1}(A \to A, \mathsf{id}_A) \\
(\text{by definition of } \Pi^\bullet) \quad \equiv\ & \Omega^{n+1}(\Pi^\bullet_{a:A}(A, a)) \\
(\text{by Lemma 3.6}) \quad =\ & \Pi^\bullet_{a:A}\Omega^{n+1}(A, a)
\end{aligned}$$

□

It will be useful to consider the restriction of a universe to its n-types:

Definition 4.3 ($\mathcal{U}^{\leq n}$, see [35, Chapter 7.1])**.** For a universe $\mathcal{U}$ and an integer $n \geq -2$, we define $\mathcal{U}^{\leq n}$ as the "subuniverse" of n-types, that is,

$$\mathcal{U}^{\leq n} :\equiv \Sigma_{A:\mathcal{U}}\mathsf{is}\text{-}n\text{-}\mathsf{type}(A).$$

Similar as for $\mathcal{U}_\bullet$, we call $\mathcal{U}^{\leq n}$ a *universe*, but it is important to note that it is a defined type, not a primitive of the theory.

The following two simple and well-known observations will be useful:

Lemma 4.4 ([35, Theorem 7.2.9]). *For $n \geq -1$, a type A is an n-type if and only if for every a in A, the loop space $\Omega^{n+1}(A, a)$ is contractible.*

(sketch). The statement is clear for $n \equiv -1$. Assume $n \geq 0$. By definition, A is an n-type if and only if, for all $a, b : A$, the type $a = b$ is an $n-1$-type. By the induction hypothesis, this is (for all a, b) the case if and only if $\Omega^n(a = b, p)$ is contractible for all $p : a = b$. By path induction on p, this is equivalent to requiring $\Omega^n(a = a, \mathsf{refl}_a)$ to be contractible for all a in A. □

Lemma 4.5 ([35, Theorem 7.1.11]). *For any $n \geq -2$ and universe $\mathcal{U}$, the type $\mathcal{U}^{\leq n}$ is $n+1$-truncated.*

Proof. For $n \equiv -2$, note that every contractible type is equivalent to the unit type. Assume $n \geq -1$. By Lemma 4.4, it suffices to show that, for any $(X, h) : \mathcal{U}^{\leq n}$, the loop space $\Omega^{n+2}(\mathcal{U}^{\leq n}, (X, h))$ is contractible. But $(\mathcal{U}^{\leq n}, (X, h))$ is judgmentally equal to $\Sigma^{\bullet}_{(\mathcal{U}, X)}(\mathsf{is}\text{-}n\text{-}\mathsf{type}, h)$. By Lemma 4.1 showing that $\Omega^{n+2}(\mathcal{U}, X)$ is contractible for any n-type X is therefore enough. This is easy to see for $n \equiv -1$. For $n \geq 0$ we apply Lemma 4.2, requiring us to show that $\Pi^{\bullet}_{x:X}\Omega^{n+1}(X, x)$ is contractible, and this is the case by Lemma 4.4. □

For $n \in \mathbb{N}$, let us write $P_n(X)$ for the type of $n+1$-loops that live in the universe $\mathcal{U}_n^{\leq n}$ and have basepoint X. More precisely, we abbreviate

$$P_n : \mathcal{U}_n^{\leq n} \to \mathcal{U}_{\bullet_{n+1}}$$
$$P_n(X) :\equiv \Omega^{n+1}(\mathcal{U}_n^{\leq n}, X).$$

Homotopically, these loops $P_n(X)$ are rather tame:

Corollary 4.6 (of Lemma 4.5). *P_n is a family of sets, that is,*

$$\Pi_{\mathcal{U}_n^{\leq n}} \mathsf{isSet} \circ P_n.$$

An $n+1$-loop consists of a basepoint X and the actual loop around X. The type of $n+1$-*loops in universe* $\mathcal{U}_n^{\leq n}$ is therefore given by

$$\mathsf{Loop}_n : \mathcal{U}_{n+1}$$
$$\mathsf{Loop}_n :\equiv \Sigma_{\mathcal{U}_n^{\leq n}} \pi_1 \circ P_n.$$

For technical reasons, we choose to define $\mathsf{Loop}_{-1} :\equiv \mathbf{2}$ in the lowest universe $\mathcal{U}_0$.

This type is also fairly tame homotopically:

Lemma 4.7. *For all natural numbers n, the type Loop_{n-1} is n-truncated, that is, we can construct*

$$h_n : \mathsf{is}\text{-}n\text{-}\mathsf{type}(\mathsf{Loop}_{n-1}).$$

Proof. The claim is clearly fulfilled for $n \equiv 0$, so let us assume $n \geq 1$. By P3, it is enough to examine the two parts of the dependent sum separately. The required property for the first part is given by Lemma 4.5. Further, the second component is a family of sets by Corollary 4.6, which suffices by P2. □

For a pointed type (A, a), we say that an element $b : A$ is *trivial* if it is equal to the basepoint, i. e. if we have a proof of $a = b$.

Lemma 4.8. *For all $n \geq 0$, the type $\Omega^{n+1}\left(\mathcal{U}_n\ ,\ \mathsf{Loop}_{n-1}\right)$ has a non-trivial inhabitant. The same is true for $\Omega^{n+1}\left(\mathcal{U}_n^{\leq n}\ ,\ (\mathsf{Loop}_{n-1}, h_n)\right)$.*

Proof. Observe that the pointed type $\left(\mathcal{U}_n^{\leq n}\ ,\ (\mathsf{Loop}_{n-1}, h_n)\right)$ can be written as (and is judgmentally equal to) the expression $\Sigma^\bullet_{(\mathcal{U}_n, \mathsf{Loop}_{n-1})}(\mathsf{is}\text{-}n\text{-}\mathsf{type}, h_n)$. As the predicate $\mathsf{is}\text{-}n\text{-}\mathsf{type}$ is propositional, Lemma 4.1 implies the equivalence of the two loop spaces of this lemma, and we may restrict ourselves to showing that the claim holds for $\Omega^{n+1}\left(\mathcal{U}_n\ ,\ \mathsf{Loop}_{n-1}\right)$.

We do induction on n. For $n \equiv -1$, we have to provide a non-trivial inhabitant of $2 = 2$. This is swap, which is different from the trivial inhabitant $\mathsf{refl}_{\mathbf{2}}$ (see Section 2.2).

Assume $n \equiv m + 1$ and calculate:

$$\begin{aligned}
& & \Omega^{m+2}(\mathcal{U}_{m+1}, \mathsf{Loop}_m) \\
\text{(by 4.2 - local-global)} & = & \Pi^\bullet_{(X,q):\mathsf{Loop}_m} \Omega^{m+1}(\mathsf{Loop}_m, (X, q)) \\
\text{(by definition of } \Sigma^\bullet) & \equiv & \Pi^\bullet_{(X,q):\mathsf{Loop}_n} \Omega^{m+1}(\Sigma^\bullet_{(\mathcal{U}_m^{\leq m}, X)}(\pi_1 \circ P_m, q)) \\
\text{(by 3.4 - } \Omega, \Sigma^\bullet \text{ commute)} & \equiv & \Pi^\bullet_{(X,q):\mathsf{Loop}_m} \Sigma^\bullet_{\Omega^{m+1}(\mathcal{U}_m^{\leq m}, X)} \tilde{\Omega}^{m+1}(\pi_1 \circ P_m, q)
\end{aligned}$$

The underlying type of this last pointed type has the following inhabitant:

$$\xi :\equiv \lambda(X, q).(q, d_q)$$

where d_q is defined as follows:

- for $m \equiv 0$, the type of d_q is $q_*(q) = q$, which is inhabited by P8 and the fact that equality carries a groupoid structure. Note that this case corresponds to the special case we already handled, and that we are not able to obviate this additional coherence condition here.
- for $m \geq 1$, the type of d_q is contractible by Corollary 4.6 and the definition of $\tilde{\Omega}$, providing a canonical choice for d_q.

Once again, let us view Loop_{m-1}, together with h_m, as an inhabitant of $\mathcal{U}_m^{\leq m}$. Note that $P_m(\mathsf{Loop}_{m-1}, h_m)$ is, by definition, judgmentally equal to $\Omega^{m+1}\left(\mathcal{U}_m^{\leq m}\ ,\ (\mathsf{Loop}_{m-1}, h_m)\right)$. By the induction hypothesis, we can construct a non-trivial inhabitant $\tilde{q}$ of the underlying type, so that we have

$$\left((\mathsf{Loop}_{m-1}, h_m)\ ,\ \tilde{q}\right) : \mathsf{Loop}_m. \tag{1}$$

If ξ was trivial, the term $\pi_1(\xi(X, q)) \equiv q$ would be trivial in $P_n(X)$ for any $(X, q) : \mathsf{Loop}_m$. But this is invalidated by (1). □

This allows us to prove:

Theorem 4.9. *In Martin-Löf Type Theory with a hierarchy of univalent universes $\mathcal{U}_0, \mathcal{U}_1, \mathcal{U}_2, \ldots$, the universe $\mathcal{U}_n$ is not an n-type. Formally, for any natural number n, the type*

$$\neg\mathsf{is}\text{-}n\text{-}\mathsf{type}\left(\mathcal{U}_n\right)$$

is inhabited.

Remark. It is a subtle question whether this result can be stated in the form

$$\Pi_{n:\mathbb{N}} \neg \mathsf{is}\text{-}n\text{-}\mathsf{type}\,(\mathcal{U}_n)\,. \tag{2}$$

If the theory provides an internal type of universe levels that can be eliminated into from the natural numbers, then all our constructions can be carried out uniformly over a natural number inhabitant n in the context. Otherwise, the constructions are to be viewed parameterized over a natural number n in the meta theory, yielding a derivation in the theory only when instantiated with a fixed n.

of the theorem. If $\mathcal{U}_n$ was an n-type, then $\Omega^{n+1}\left(\mathcal{U}_n\ ,\ \mathsf{Loop}_{n-1}\right)$ would be propositional, contradicting Lemma 4.8. □

At the same time, we have solved the question of constructing a "strict" n-type that was discussed several times at the UF special year in Princeton:

Theorem 4.10. *For a given $n \geq -2$, there is (in the settings of Theorem 4.9) a type that is an $n+1$-type but not an n-type. In particular, for $n \geq -1$, the type Loop_n has this property. Further, for $n \geq 0$, the universe of n-types at level n, namely $\mathcal{U}_n^{\leq n}$, is such a strict $n+1$-type.*

Proof. For $n \equiv -2$, the empty type proves the statement. The claim for $\mathcal{U}_n^{\leq n}$ follows in the same way as Theorem 4.9, combined with Lemma 4.5. $\mathsf{Loop}_{-1} \equiv \mathbf{2}$ is clearly strictly a set. For $n \geq 0$, Lemma 4.7 shows that Loop_n is $n+1$-truncated. To see that it is not n-truncated, observe that the first component is $\mathcal{U}_n^{\leq n}$ and therefore not n-truncated while the second component is always inhabited. □

5 Connectedness

5.1 Truncations via Higher Inductive Types and via Universal Properties

As mentioned in the introduction, a (closed) type in MLTT can be interpreted as a *Kan complex* in the simplicial set model, and, by the *realization functor* to topological spaces, as a *CW-complex*. Under this interpretation, "ordinary" small inductive types (such as the natural numbers) correspond to the construction of a very simple CW-complex, consisting only of 0-cells (points).

In order to do more advanced synthetic homotopy theory in HoTT, it seems therefore desirable and reasonable to add *higher inductive types* (HITs) to the theory [35, Chapter 6]. These do not only allow the construction of 0-cells, as usual inductive types do, but also the construction of higher cells. Thereby, more complicated CW-complexes can be built. However, the general principle is not fully understood yet, especially when it comes to the computational aspect. At the moment, it is unclear how HITs can be added to the theory without causing similar problems as the univalence axiom. On the other hand, it seems plausible that such a computational interpretation, should it be found for the univalence axiom, could also be done similarly for HITs.

An easy example of a higher inductive type is the circle $\mathbb{S}^1$, defined by

$$\mathsf{base} : \mathbb{S}^1$$
$$\mathsf{loop} : \mathsf{base} =_{\mathbb{S}^1} \mathsf{base}.$$

Note that there is also the proof $\mathsf{refl}_{\mathsf{base}} : \mathsf{base} =_{\mathbb{S}^1} \mathsf{base}$, even if it is not part of the constructors of $\mathbb{S}^1$, and similarly, there are loop^{-1}, $\mathsf{loop} \cdot \mathsf{loop}$, and so on. It was first shown by Shulman [32] and later by Licata [21] (see also [23]) that the *fundamental group* [35, Definition 8.0.1] of $\mathbb{S}^1$ is $\mathbb{Z}$, which immediately implies that $\mathbb{S}^1$ is not a set. Their proofs are a translation of the (obviously much older) proof of the corresponding fact in homotopy theory and a usage of a HoTT-specific technique, the *encode-decode method*, respectively.

As HITs allow us to add paths at an arbitrarily high level to a type, it is clear that they can be used to construct types that are not n-truncated for a given n - after all, "being not an n-type" is by definition the same as "not having only trivial paths at level n". In $\mathbb{S}^1$, a path is added on the first level, and similarly, the higher spheres $\mathbb{S}^n$ can be constructed by adding paths on higher levels. For a discussion of the spheres, see [35, Chapter 8].

The only thing that is actually needed is showing that the path generated by the highest-level constructor of the sphere is indeed non-trivial. However, even for this seemingly simple property is not amenable to an instant argument. While $\mathbb{S}^n$ has $\mathbb{Z}$ as n-th homotopy group, which immediately implies that it is not an $n-1$-type, calculating it in HoTT via the long exact sequence requires some effort. It was performed by Brunerie and Licata [22].

On the other hand, the construction of Loop_n that we present in this article can be understood as a way to use HITs even if the theory does not support them. Recall that our type Loop_n was (after unfolding the definition of P_n) defined as

$$\mathsf{Loop}_n :\equiv \pi_1(\Sigma^\bullet_{X:\mathcal{U}_n^{\leq n}} \Omega^{n+1}(\mathcal{U}_n^{\leq n}, X)).$$

If HITs are available, Loop_n is equivalent to the function type

$$\mathbb{S}^n \to \mathcal{U}_n^{\leq n}.$$

Even if we do not have $\mathbb{S}^n$ available in the theory, we can thus still talk about how the sphere could be mapped into another type. Another certain class of higher inductive types is important for a further construction we want to present. The notion of connectedness (as defined in [35, Chapter 7.5]) is from a certain point of view dual to that of truncation levels. If a type A is n-truncated, it means that A is trivial *above* dimension n. In contrast, a type is n-connected if it is trivial *below* dimension n in the topological model. However, this property is tricky to express in type theory: if we directly state that a type is trivial at some dimension, it immediately implies that it is trivial at all higher dimensions as well (see P2). The way Homotopy Type Theory deals with this problem is the following: in order to express that a type is n-connected, the type is first "artificially" made trivial above dimension n, and then required to be contractible.

This is done using a special sort of HITs, namely *truncations* [35, Chapter 7.3]. Given a type A, the n-truncation is, roughly speaking, constructed by adding paths between any two "objects" at dimension $n+1$ of A, thus "cutting off" all possibly interesting homotopical structure above dimension n. We want

to give a formal definition that can be stated without developing the theory of HITs:

Definition 5.1 (truncation, c. f. [35, Theorem 7.3.2])**.** For a given type $A : \mathcal{U}$ and $n \geq -2$, the *n-truncation* is characterized by the following rules:

- *Formation rule:* $\|A\|_n$ is an n-type in $\mathcal{U}$.
- *Introduction rule:* there is a map $|-|_n : A \to \|A\|_n$.
- *Elimination rule:* for a universe $\mathcal{V}$ and a family $(P, h) : \|A\|_n \to \mathcal{V}^{\leq n}$ of n-types, any dependent function $g : \Pi_A P \circ |-|_n$ can be lifted to a function $\overline{g} : \Pi_{\|A\|_n} P$.
- *Computation rule:* for any g as in the rule above and $a : A$, we have the judgmental β-rule $\overline{g}(|a|_n) \equiv g(a)$.

We say that a theory has truncations if the truncation exists for any type A and any $n \geq -2$.

The definition we need is then:

Definition 5.2 (connectedness [35, Definition 7.5.1])**.** A type A is n-connected (for some $n \geq -2$) if $\|A\|_n$ is contractible,

$$\mathsf{is}\text{-}n\text{-}\mathsf{connected}(A) :\equiv \mathsf{isContr}(\|A\|_n).$$

Even though the definition of connectedness requires HITs, we can do surprisingly much without them. We say that a statement holds in MLTT if it only requires plain Martin-Löf Type Theory, without univalence or higher inductive types. The setting that we have considered so far, namely MLTT with a hierarchy of univalent universes, is MLTT+UA. If a lemma additionally requires truncations, we say that it holds in MLTT+UA+TRUNC.

We will use the fact that truncations can be characterized by their *universal properties* in MLTT. This allows us to formulate the statement that a type is n-connected* in MLTT, with the name being justified by the fact that we can prove $\mathsf{is}\text{-}n\text{-}\mathsf{connected} = \mathsf{is}\text{-}n\text{-}\mathsf{connected}^\star$ in MLTT+UA+TRUNC.

Given a type A with an inhabitant a, we will (in MLTT) construct the n-connected version of A with basepoint a. In MLTT+UA, that type has the same loop spaces as A above dimension n and is n-connected*. For $n \equiv -1$, this corresponds to constructing the connected component of a in the ordinary topological sense.

The construction of this "n-connected version" can also be done fairly easily in MLTT+UA+TRUNC, which is very likely to be known, although we are unable to find it in the standard literature.

We first want to specify what it means in plain MLTT to have the universal property of a truncation.

Definition 5.3. Let $\mathcal{U}$ and $\mathcal{V}$ be two universes, A be a type in $\mathcal{U}$, and $n \geq -2$ be a number. Let X be an n-type in $\mathcal{U}$ and c a function from A to X. We say that (X, c) (or just c) has the *universal property of the n-truncation of A with respect to* $\mathcal{V}$ if, for any n-type Y in $\mathcal{V}$, the function types $X \to Y$ and $A \to Y$ are equivalent, and the equivalence is given by composition with c:

$$\mathsf{up}^{A:\mathcal{U}}_{\mathcal{V}^{\leq n}}(c) :\equiv \Pi_{(Y,k):\mathcal{V}^{\leq n}} \mathsf{isequiv}(\lambda f : X \to Y. f \circ c).$$

Let us now define a type with the property that any of its inhabitant can serve as an n-truncation. The crucial point will consist of perspicaciously inserting the assumption that certain truncations exist deep into connectedness construction types, taking care not to change their expected properties.

Definition 5.4. Given two universes $\mathcal{U}$ and $\mathcal{V}$, a type A in $\mathcal{U}$ and $n \geq -2$ in MLTT, define the *type of n-truncations* of A to be the type of maps $c : A \to X$ for some n-type $X : \mathcal{U}$ which satisfy the universal property:

$$\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A) :\equiv \Sigma_{(X,h):\mathcal{U}^{\leq n}} \Sigma_{c:A\to X}\ \mathsf{up}^{A:\mathcal{U}}_{\mathcal{V}^{\leq n}}(c).$$

Note that this type is not in universe $\mathcal{U}$ or $\mathcal{V}$, but it inhabits every universe that $\mathcal{U}$ and $\mathcal{V}$ both inhabit. Given $t \equiv (X, h, c, u) : \mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$, we write $\mathsf{type}(t)$ for the component X (i.e., $\mathsf{type} \equiv \pi_1$) and $\mathsf{cons}(t)$ for the component c (i.e., $\mathsf{cons} \equiv \pi_1 \circ \pi_2$).

Remark. In MLTT+UA+TRUNC, the type $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$ is inhabited by the n-truncation of A, namely $\|A\|_n$ [35, Lemma 7.3.3]. For MLTT or even MLTT+UA, which we consider here, we strongly believe that an inhabitant of $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$ can, in general, not be constructed: the n-truncation is not *definable*.

Our intention is to use an (assumed) inhabitant of $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$ in the same way as $\|A\|_n$ could be used if it was part of the theory. While this turns out to be possible, there are a couple of obstacles:

First, the only assumption we make is that $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$ includes the encoding of a universal property. By [35, Lemma 7.3.3], the truncation $\|A\|_n$ has this universal property with respect to n-types of any universe. Being unable to polymorphically quantify over all universes (see Remark 4), we have to restrict ourselves to a fixed elimination unverse $\mathcal{V}$ in in $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$. We need to be careful though: it is not sufficient for our purposes to require the universal property with respect to all n-types in the same universe as A, as this would prevent us from performing *large elimination*. For that reason, we require the universal property for all n-types that live in some fixed universe $\mathcal{V}$, which for most constructions will have to be larger than $\mathcal{U}$.

Second, the truncation $\|A\|_n$, defined as a HIT, has important judgmental computation rules, i. e. equations that hold judgmentally. Such rules have the potential to make the theory strictly stronger: for example, an interval type with judgmental β-rule implies function extensionality [33]. It is therefore not a priori clear that the universal property that we have at hand suffices for everything we want to do. However, even disregarding that, judgmental computational rules offer in many cases a huge simplification. Indeed, our formalization of the proof is considerably more tedious than the analogous proof using $\|A\|_n$ would be.

Third, note that in the definition of $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$, we only ask for a non-dependent universal property, while (in MLTT+UA+TRUNC) $\|A\|_n$ has the dependent version of this property. While we could encode the dependent universal property in the definition as well, we do not need to: as we will see, the non-dependent universal property implies the dependent one.

5.2 Consequences of the Universal Property

Our next goal is to prove a couple of properties of $\mathsf{up}^{A:\mathcal{U}}_{\mathcal{V}^{\leq n}}$ and $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}$ in MLTT. For the lemmata in this subsection, let $\mathcal{U}$ and $\mathcal{V}$ be two universes where $\mathcal{V}$ is at least

as big as $\mathcal{U}$, i. e. every type in $\mathcal{U}$ is also of type $\mathcal{V}$. Further, let $A : \mathcal{U}$ be a type and $n \geq -2$ a number.

Lemma 5.5. *A map $e : A \to X$ for some n-type $X : \mathcal{U}$ has the universal property if and only if it has the dependent universal property,*

$$\mathsf{up}^{A:\mathcal{U}}_{\mathcal{V}^{\leq n}}(e) \longleftrightarrow \mathsf{dup}^{A:\mathcal{U}}_{\mathcal{V}^{\leq n}}(e),$$

where the latter expression is defined as

$$\mathsf{dup}^{A:\mathcal{U}}_{\mathcal{V}^{\leq n}}(e) :\equiv \Pi_{\langle Y,h\rangle : X \to \mathcal{V}^{\leq n}} \mathsf{isequiv}(\lambda f : \Pi_X Y. f \circ e).$$

Proof. The direction "if" is trivial. For the other direction, we need the following statement: For a function $u : C \to D$ between types C, D and $E : D \to \mathcal{U}'$ a type family in any universe $\mathcal{U}'$, dependent functions from $c : C$ to $E(u(c))$ are equivalent to liftings of u through $\pi_1 : \Sigma_D E$:

$$\Sigma_{s:C\to\Sigma_D E}\ \pi_1 \circ s = u \ \simeq\ \Pi_C\ E \circ u. \tag{3}$$

Note that for $C \equiv D$ and $u \equiv \mathsf{id}$ this just says that $\Pi_C E$ is the type of sections of the first projection. The equivalence 3 could be slightly strengthened by making D (and E) dependent on C, but we do not need this generality here. An easy way to prove (3) is combining a couple of equivalences. First,

$$\Sigma_{s:C\to\Sigma_D E}\ \pi_1 \circ s = u \ \simeq\ \Pi_{c:C}\Sigma_{p:\Sigma_D E}\ \pi_1(p) = u(c) \tag{4}$$

holds by strong function extensionality P5 combined with the (∞,∞)-version of the *axiom of choice*, which is by no means an axiom, but always fulfilled in MLTT [35, Theorem 2.15.7]. Further, we have

$$\begin{aligned}\Sigma_{p:\Sigma_D E}\ \pi_1(p) = u(c) \ &\simeq\ \Sigma_{d:D}\ E(d) \times (d = u(c)) \\ &\simeq\ \Sigma_{\Sigma_{d:D} d = u(c)}\ E \circ \pi_1 \\ &\simeq\ E(u(c))\end{aligned} \tag{5}$$

The third step follows from contractibility of $\Sigma_{d:D} d = d_0$ and P11. The equivalences (4) and (5) together prove (3).

To prove the "only if" direction of the lemma, assume $e : A \to X$ satisfies the non-dependent universal property. Let $\langle Y, h\rangle : X \to \mathcal{V}^{\leq n}$ be a family of n-types over X. Set $\tilde{Y} :\equiv \Sigma_X Y$; by P3, this is an n-type in $\mathcal{V}$. By $\mathsf{up}^{A:\mathcal{U}}_{\mathcal{V}^{\leq n}}(e)$, the map $\lambda f. f \circ e$ induces an equivalence

$$(X \to \tilde{Y}) \simeq (A \to \tilde{Y}). \tag{6}$$

Fix $s : X \to \tilde{Y}$. We then have $\pi_1 \circ s : X \to X$. By the assumed universal property, composition with e is an equivalence $(X \to X) \simeq (A \to X)$. As equivalences preserve path spaces (P7), we have $(\pi_1 \circ s = \mathsf{id}_X) \simeq (\pi_1 \circ s \circ e = e)$. This immediately gives us

$$\Sigma_{s:X\to\tilde{Y}} \pi_1 \circ s = \mathsf{id}_X \ \simeq\ \Sigma_{s:X\to\tilde{Y}} \pi_1 \circ s \circ e = e. \tag{7}$$

In general, for any types C, D, a type family $P : D \to \mathcal{U}'$ and an equivalence $h : C \to D$, we have that $\Sigma_C P \circ h \simeq \Sigma_D P$. Applying this rule with $C \equiv X \to \tilde{Y}$, $D \equiv A \to \tilde{Y}$, $P(s) \equiv \pi_1 \circ s = e$ and the equivalence (6) for h, we get

$$\Sigma_{s:X\to\tilde{Y}} \pi_1 \circ s \circ e = e \ \simeq\ \Sigma_{s:A\to\tilde{Y}} \pi_1 \circ s = e. \tag{8}$$

Finally, we are ready to prove $\Pi_X Y \simeq \Pi_A Y \circ e$. We start with $\Pi_X Y$ and apply (3) with $u \equiv \mathsf{id}_X$ to transform it into $\Sigma_{s:X\to\tilde{Y}} \pi_1 \circ s = \mathsf{id}_X$. Using first (7) and then (8), this type is equivalent to $\Sigma_{s:A\to\tilde{Y}} \pi_1 \circ s = e$. Equivalence (3), this time with $u \equiv e$, transforms that type into $\Pi_A Y \circ e$ as required.

Going through the proof again and only checking what the function part of the constructed equivalence does, it is easy to see that $f : \Pi_X Y$ is mapped to $f \circ e : \Pi_A Y \circ e$, proving $\mathsf{dup}_{\mathcal{V}^{\leq n}}^{A:\mathcal{U}}(e)$. □

Another interesting point is that the type $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$ can be shown to be propositional already in MLTT+UA, implying that it is contractible in MLTT+UA+TRUNC by Remark 5.1.

Lemma 5.6. *In MLTT+UA, the type $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$ is propositional.*

Proof. This is a consequence of the fact that $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$ is characterized via a universal property. Given two inhabitants $A \xrightarrow{e_1} X_1$ and $A \xrightarrow{e_2} X_2$, we need to show that they are equal. Note that the universal property itself is propositional, and thus, we do not have to construct a path between the two witnesses of this property. Such an equality proof therefore just corresponds to an equivalence $e : X_1 \simeq X_2$ and a proof of $e \circ e_1 = e_2$. The universal property of e_1 tells us that $X_1 \to X_2$ and $A \to X_2$ are equivalent. As the latter type is inhabited by e_2, we find an inhabitant e of the former, and we get the property $e \circ e_1 = e_2$ for free. The construction of the inverse of e is completely analogous, and the proof that their composition is the identity proceeds by elimination using the universal properties of e_1 and e_2. □

Corollary 5.7. *Write $\mathcal{T}_{\mathcal{U},\mathcal{V}}$ for the type $\Pi_{(n,A):\mathbb{N}_{-2}\times\mathcal{U}} \mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$. Then, $\mathcal{T}_{\mathcal{U},\mathcal{V}}$ is propositional by P4.*

Not surprisingly, if a type has a truncation, then so does a type equivalent to it.

Lemma 5.8. *Assume we have a third universe $\mathcal{U}'$ which is at least as big as $\mathcal{U}$ and at most as big as $\mathcal{V}$ (any type in $\mathcal{U}$ is a type in $\mathcal{U}'$, and any type in $\mathcal{U}'$ is a type in $\mathcal{V}$). Assume further we have types $A : \mathcal{U}$ and $B : \mathcal{U}'$ as well as (for some n) an inhabitant $t_A : \mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$. Then, there exists $t_B : \mathcal{T}^n_{\mathcal{U}',\mathcal{V}}(B)$ and it satisfies $\mathsf{type}(t_A) \simeq \mathsf{type}(t_B)$.*

Proof. Write $t_A \equiv (X, h, c, u)$ with $(X, h) : \mathcal{U}^{\leq n}$, $c : A \to X$ and $u : \mathsf{up}_{\mathcal{V}^{\leq n}}^{A:\mathcal{U}}(c)$, as well as $(f, e) : B \simeq A$ for the (inverse of the) equivalence between A and B. As $\mathcal{U}'$ is at least as big as $\mathcal{U}$, we have $(X, h) : \mathcal{U}'^{\leq n}$. It is easy to construct an inhabitant $u' : \mathsf{up}_{\mathcal{V}^{\leq n}}^{B:\mathcal{U}'}(c \circ f)$: all we have to do is using u together with the fact that composition with the equivalence f is an equivalence itself. Thus, we have $(X, h, c \circ f, u') : \mathcal{T}^n_{\mathcal{U}',\mathcal{V}}(B)$ with the required property. □

Corollary 5.9. *Under the assumptions of Lemma 5.8 above and, in addition, given $t_B : \mathcal{T}^n_{\mathcal{U}',\mathcal{V}}(B)$, we have the equivalence $\mathsf{type}(t_A) \simeq \mathsf{type}(t_B)$, as inhabitants of $\mathcal{T}^n_{\mathcal{U}',\mathcal{V}}(B)$ are unique by Lemma 5.6.*

Assuming $\mathcal{T}_{\mathcal{U},\mathcal{V}}$ is inhabited at the necessary points, we need information on how our notion of truncation interacts with dependent sum types and path spaces. The corresponding lemmata are well-known for the truncation with judgmental computation rule (i.e., that are part of the theory) of [35]. Their

proofs can be modified to only use the universal property, allowing us to transfer them to our setting. As explained in Remark 5.1, reasoning about reduction behaviour propositionally is tedious, particularly so whenever it occurs in a type. The second lemma is the reason why we need to parametrize $\mathcal{T}_{\mathcal{U},\mathcal{V}}$ over *two* universes, enabling us to perform the correct elimination steps. Still, the proofs of the lemmata follow well-known ideas [35]. We only give rough sketches. Rigorous proofs can be found, as for everything else in this article, in our formalization [20].

The following lemma can be understood as "flattening" (cf. [35, Section 6.12]) for truncations.

Lemma 5.10. *Let $t_A : \mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$ and $\langle P, h_P\rangle : \mathsf{type}(t_A) \to \mathcal{U}^{\leq n}$ (i.e., P is a family of types over $\mathsf{type}(t_A)$ and h_P is a proof that it is a family of n-types). Given $t_\Sigma : \mathcal{T}^n_{\mathcal{U},\mathcal{V}}\left(\Sigma_A P \circ \mathsf{cons}(t_\Sigma)\right)$, we have*

$$\mathsf{type}(t_\Sigma) \simeq \Sigma_{\mathsf{type}(t_A)} P. \tag{9}$$

Proof. We can directly define functions

$$\begin{aligned} f &: \Sigma_A P \circ \mathsf{cons}(t_A) \;\to\; \Sigma_{\mathsf{type}(t_A)} P & f(a,x) &:\equiv (\mathsf{cons}(t_A)(a), p) \\ g &: \Pi_{a:A}\left(P(\mathsf{cons}(t_A)(a)) \to \mathsf{type}(t_\Sigma)\right) & g(a) \quad &:\equiv \lambda p.\mathsf{cons}(t_\Sigma)(a,p) \end{aligned}$$

After applying the universal property of $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}\left(\Sigma_A P \circ \mathsf{cons}(t_\Sigma)\right)$, the function f gives us a map from the left-hand side to the right-hand side of the equivalence 9. For the other direction, we need to use the dependent universal property of $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(A)$ together with g and uncurry the constructed function.

To prove that the two maps are inverses of each other, we use the (dependent) universal properties again. Both directions are straightforward. □

In the statement of the next lemma, note that $\mathcal{T}^m_{\mathcal{U},\mathcal{U}}(X)$ is always implied by $\mathcal{T}^n_{\mathcal{U},\mathcal{V}}(X)$ (if $\mathcal{U} : \mathcal{V}$) and is therefore a more minimalistic assumption.

Lemma 5.11. *Let $\mathcal{U}$, A, n be as before, but let $\mathcal{V}$ be a universe larger than $\mathcal{U}$ in the sense that we have $\mathcal{U} : \mathcal{V}$. Assume we have $t_A : \mathcal{T}^{n+1}_{\mathcal{U},\mathcal{V}}(A)$ as well as $t_P : \Pi_{a_1,a_2:A}\mathcal{T}^n_{\mathcal{U},\mathcal{U}}(a_1 = a_2)$. Then, for any given $a_1, a_2 : A$, the equivalence*

$$\mathsf{type}(t_P\, a_1\, a_2) \;\simeq\; \mathsf{cons}(t_A)(a_1) =_{\mathsf{type}(t_A)} \mathsf{cons}(t_A)(a_2) \tag{10}$$

holds.

Proof. The proof is analogous to the one in [35, Theorem 7.3.12], but more tedious for lack of judgmental computation rules. In the application of the encode-decode method, it is necessary to use the dependent universal property of t_A to construct a map into $\mathcal{U}^{\leq n}$, a type that does not inhabit $\mathcal{U}$, and that is why we need to ask for the universal property with respect to types in $\mathcal{V}$ instead. □

5.3 Construction of n-connected Types

With the previous lemmata at hand, we are able to perform the discussed construction.

Definition 5.12. In MLTT, given a pointed type (A, a) and $n \geq -1$, we write $\lfloor A, a \rfloor^n$ for the *n-connected version* of (A, a), defined by

$$\lfloor A, a \rfloor^n :\equiv \Sigma_{b:A} \Pi_{t:\mathcal{T}^{n-1}_{\mathcal{U},\mathcal{U}}(a =_A b)} \mathsf{type}(t).$$

It has the canonical inhabitant $(a, \overline{a})$ where $\overline{a} :\equiv \lambda t.\mathsf{cons}(t)(\mathsf{refl}_a)$. Note that $\lfloor A, a \rfloor^n$ is not a type in $\mathcal{U}$ but a type in $\mathcal{V}$ for any $\mathcal{V}$ satisfying $\mathcal{U} : \mathcal{V}$.

In the above definition, we include the case $n \equiv -1$, but note that it trivial in the sense that $\lfloor X \rfloor^{-1} \simeq X$. The construction is interesting already for $n \equiv 0$ though, which corresponds to the *connected component* of a.

Lemma 5.13. *In MLTT, $\lfloor A, a \rfloor^n$ is, on dimension $n+1$ and above, equivalent to A, in the sense that*

$$\Omega^{n+1}(A, a) \;=\; \Omega^{n+1}(\lfloor A, a \rfloor^n, (a, \overline{a})).$$

Proof. Write $(\lfloor A, a \rfloor^n, \overline{a})$ as a pointed dependent sum with an $n-1$-truncated pointed predicate over (A, a). The statement then follows from Lemma 4.1. □

Note that the formulation of the above lemma is justified by the fact that the n-th dimension of a type is entirely described by the n-th loop spaces. If $a = b$, then the types $a = a$ and $a = b$ are equivalent, and therefore, examining the elements of the loop space is enough to determine the whole structure of a type above some dimension. This principle was used in the proof of Lemma 4.4.

Let us now discuss the connectedness properties of the type $\lfloor X \rfloor^n$ for appropriate X and n.

Definition 5.14. In plain MLTT, given universes $\mathcal{U}$ and $\mathcal{V}$ as well as $C : \mathcal{V}$, we define the following notion of n-connectedness:

$$\mathsf{is}\text{-}n\text{-}\mathsf{connected}^\star_{\mathcal{U},\mathcal{V}}(C) :\equiv \Pi_{t:\mathcal{T}_{\mathcal{U},\mathcal{V}}} \Pi_{t_C:\mathcal{T}^n_{\mathcal{V},\mathcal{V}}(C)} \mathsf{isContr}(\mathsf{type}(t_C))$$

The notation $\mathcal{T}_{\mathcal{U},\mathcal{V}}$ is used as introduced in Corollary 5.7.

Lemma 5.15. *In MLTT+UA, for any pointed type $A : \mathcal{U}$ (with a point $a : A$) and universe $\mathcal{V}$ that contains $\mathcal{U}$, i. e. $\mathcal{U} : \mathcal{V}$, the type $\lfloor A, a \rfloor^n$ is n-connected in the above sense, i.e.*

$$\mathsf{is}\text{-}n\text{-}\mathsf{connected}^\star_{\mathcal{U},\mathcal{V}}(\lfloor A, a \rfloor^n).$$

Proof. Unfolding the definitions, given

$$\begin{aligned} t \quad &: \mathcal{T}_{\mathcal{U},\mathcal{V}}, \\ t_C &: \mathcal{T}^n_{\mathcal{V},\mathcal{V}} \left(\Sigma_{b:A} \Pi_{s:\mathcal{T}^{n-1}_{\mathcal{U},\mathcal{U}}(a =_A b)} \mathsf{type}(s) \right), \end{aligned} \tag{11}$$

we need to show that $\mathsf{type}(t_C)$ is contractible.

We perform a sequence of steps, each transforming $\mathsf{type}(t_C)$ into an equivalent type. Heavy use of Corollary 5.9 is made. This is especially true for the very first step: by Corollary 5.7 and P11, the remaining occurrence of Π in (11) can be removed, as $t(n-1, a =_A b)$ can be viewed as an inhabitant of $\mathcal{T}^{n-1}_{\mathcal{U},\mathcal{U}}(a =_A b)$. Consequently it is enough to prove the type

$$\mathsf{type}\big(t\,(n, \Sigma_{b:A} \mathsf{type}(t(n-1, a =_A b)))\big) \tag{12}$$

contractible. Next, we apply Lemma 5.11 which provides an equivalence

$$\mathsf{type}(t(n-1, a =_A b)) \;\simeq\; P(\mathsf{cons}(t(n,A))(b)) \tag{13}$$

where

$$\begin{aligned} P &: \mathsf{type}(t(n,A)) \to \mathcal{U} \\ P(x) &:\equiv \mathsf{cons}(t(n,A))(a) =_{\mathsf{type}(t(n,A))} x, \end{aligned}$$

transforming the type (12) into

$$\mathsf{type}\left(t\left(n, \Sigma_A P \circ \mathsf{cons}(t(n,A))\right)\right). \tag{14}$$

Clearly, P is a family of $n-1$-types. We may therefore apply Lemma 5.10, telling us that the type (14) is equivalent to

$$\Sigma_{\mathsf{type}(t(n,A))} P, \tag{15}$$

and this *path-from* type is trivially contractible. □

The name of the entity in Definition 5.14 is justified in that we took the standard definition of connectivity and replaced the use of truncation with an inhabitant of our type of truncations before quantifying over it. Unsurprisingly, it turns out to be equivalent to the standard notion of connectedness (Definition 5.2) if available.

Lemma 5.16. *In MLTT+UA+TRUNC, for any $\mathcal{U}$ and $\mathcal{V}$, we have*

$$\Pi_{A:\mathcal{U}} \Pi_{n:\mathbb{N}} \;\mathsf{is}\text{-}n\text{-}\mathsf{connected}^\star_{\mathcal{U},\mathcal{V}}(A) \simeq \mathsf{is}\text{-}n\text{-}\mathsf{connected}(A).$$

Proof. In MLTT+UA+TRUNC, the function type $\mathcal{T}_{\mathcal{U},\mathcal{V}}$ has a canonical inhabitant, as mentioned in Remark 5.1. Together with Lemma 5.6, this shows that $\mathcal{T}_{\mathcal{U},\mathcal{V}}$ is contractible, and property P11 validates the stated equivalence. □

To conclude this section, let us make two short remarks, both of which refer to MLTT+UA+TRUNC. The first is an immediate consequence of the lemmata and definitions above: for a given pointed type (A, a) and a given $n \geq -1$, the type

$$\Sigma_{b:A} \left\| b = a \right\|_{n-1}$$

is n-connected and has, above dimension n, the same properties as A (with basepoint a, of course).

The second remark is that n-connectedness can be defined without referring to n-truncations at all, but only by using propositional truncations [35, Exercise 7.6]. Those could be seen as somewhat more "elementary" in the sense that they, or notions similar to them, have been considered a long time before HoTT was developed; see [8], [6]. Instead of saying that a type is n-connected if its n-truncation is contractible, we could ask for it to be "merely" inhabited (in the sense of [35]), and all its path spaces to be $n-1$-connected:

$$\begin{aligned} \mathsf{is}\text{-}{-2}\text{-}\mathsf{connected}'(A) &:\equiv \mathbf{1} \\ \mathsf{is}\text{-}n+1\text{-}\mathsf{connected}'(A) &:\equiv \left\| A \right\|_{-1} \times \Pi_{x,y:A} \mathsf{is}\text{-}n\text{-}\mathsf{connected}'(x = y) \end{aligned}$$

This notion of connectedness is equivalent to the one stated in Definition 5.2, as we have shown in our formalization. We think that, generally in HoTT, there might be cases where this notion is more convenient to use, as it supports direct induction on n. Unfortunately, it does not seem suitable for a modification that gives a strong enough definition of connectedness in MLTT+UA as does our Definition 5.14.

6 Conclusion

In Section 4, we have seen that $\mathcal{U}_n$ is not an n-type and that $\mathcal{U}_{n+1}$ contains a "strict" $n+1$-type, namely $\mathcal{U}_n^{\leq n}$. Combining that result with the construction of the previous section, we immediately get:

Theorem 6.1. *In MLTT+UA, for a given natural number n, we can construct a type, defined by*

$$\begin{aligned} M_n &: \quad \mathcal{U}_{n+2} \\ M_n &:\equiv \left\lfloor \mathcal{U}_n^{\leq n}, (\mathsf{Loop}_{n-1}, h_n) \right\rfloor^n \end{aligned}$$

(with h_n as in Lemma 4.7) such that, purely in MLTT+UA, the following propositions are provable:

$$\begin{aligned} &M_n \text{ is } n+1\text{-truncated:} && \mathsf{is}\text{-}n+1\text{-}\mathsf{type}(M_n) \\ &M_n \text{ is not } n\text{-truncated:} && \neg\mathsf{is}\text{-}n\text{-}\mathsf{type}(M_n) \\ &M_n \text{ is } n\text{-connected:} && \mathsf{is}\text{-}n\text{-}\mathsf{connected}^{\star}_{\mathcal{U}_{n+2},\mathcal{U}_{n+3}}(M_n) \end{aligned}$$

Note that these properties also imply that, in MLTT+UA+TRUNC, the n-th homotopy group [35, Definition 8.0.1] of M_n is non-trivial, and all other homotopy groups are trivial.

Proof. The first two statement follows by Lemma 5.13, where the first uses additionally Lemma 4.5, the second Lemma 4.8. The third part is an application of Lemma 5.15 with $\mathcal{U} \equiv \mathcal{U}_{n+2}$ and $\mathcal{V} \equiv \mathcal{U}_{n+3}$. □

A related result with much stronger assumptions was shown before using *Eilenberg-Mac Lane Spaces* [35, Theorem 8.10.3]: in MLTT with univalence and not just truncations, but general higher inductive types, it is possible to construct a type $K(G,n)$ that is an n-type such that the n-th homotopy group equals some abelian group G and all the others are trivial. That construction uses higher inductive types not just to truncate, but also to produce the actual non-trivial higher paths. Again, this is not too surprising - the property of $K(G,n)$ point directly to that usage of HITs. We have shown that, even without them, we can get quite close.

Acknowledgments First of all, we would like to thank the participants, particularly the organizers, of the Univalent Foundations Program in Princeton 2012/2013 for letting us know of the problem of constructing a "strict" n-type without higher inductive types, and for many beneficial discussions. We especially thank Thierry Coquand and Thorsten Altenkirch for their stimulating comments and for encouraging us to continue pursuing these results.

References

[1] Thorsten Altenkirch. Extensional equality in intensional type theory. In *In LICS 99*, pages 412–420. IEEE Computer Society Press, 1999.

[2] Thorsten Altenkirch, Thomas Anberrée, and Nuo Li. Definable Quotients in Type Theory. 2011.

[3] Thorsten Altenkirch and Conor McBride. Towards observational type theory. Manuscript, available online, February 2006.

[4] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra. Observational equality, now! In *PLPV '07: Proceedings of the 2007 workshop on Programming languages meets program verification*, pages 57–68, New York, NY, USA, 2007. ACM.

[5] Steve Awodey and Michael A Warren. Homotopy theoretic models of identity types. Technical Report arXiv:0709.0248, Sep 2007.

[6] Steven Awodey and Andrej Bauer. Propositions as [types]. *Journal of Logic and Computation*, 14(4):447–471, 2004.

[7] Benno van den Berg and Richard Garner. Types are weak omega-groupoids. Technical report, Dec 2008.

[8] R. L. Constable, S. F. Allen, H. M. Bromley, W. R. Cleaveland, J. F. Cremer, R. W. Harper, D. J. Howe, T. B. Knoblock, N. P. Mendler, P. Panangaden, J. T. Sasaki, and S. F. Smith. *Implementing Mathematics with the Nurpl Proof Development System*. Prentice-Hall, NJ, 1986.

[9] Thierry Coquand. Formath (research program), 2013.

[10] Thierry Coquand and Gerard Huet. The calculus of constructions. *Inf. Comput.*, 76(2-3):95–120, February 1988.

[11] Georges Gonthier. Formal Proof – The Four-Color Theorem. *Notices of the American Mathematical Society*, 55(11):1382–1393, December 2008.

[12] Georges Gonthier, Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, François Garillot, Stéphane Le Roux, Assia Mahboubi, Russell O'Connor, Sidi Ould Biha, Ioana Pasca, Laurence Rideau, Alexey Solovyev, Enrico Tassi, and Laurent Thery. A machine-checked proof of the odd order theorem. In *Interactive Theorem Proving*, 2013.

[13] Martin Hofmann. Quotient types via coequalizers in martin-löf type theory. 1990.

[14] Martin Hofmann. *Extensional concepts in intensional type theory*. PhD thesis, July 1995.

[15] Martin Hofmann and Thomas Streicher. The groupoid interpretation of type theory. In *In Venice Festschrift*, pages 83–111. Oxford University Press, 1996.

[16] Community for UF and HoTT. Hott agda library, 2013.

[17] William A. Howard. The formulas-as-types notion of construction. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism*, pages 479–490. Academic Press, 1980.

[18] Chris Kapulkin, Peter LeFanu Lumsdaine, and Vladimir Voevodsky. The simplicial model of univalent foundations. 2012.

[19] Chris Kapulkin, Peter LeFanu Lumsdaine, and Vladimir Voevodsky. Univalence in simplicial sets. 2012.

[20] Nicolai Kraus and Christian Sattler. Universe n is not an n-type (Agda formalization), 2013. Available at the first-named author's institutional webpage.

[21] Daniel Licata. A simpler proof that pi1 of s1 is (blog post), 2012.

[22] Daniel Licata. Homotopy theory in type theory: Progress report (blog post), 2013.

[23] Daniel R. Licata and Michael Shulman. Calculating the fundamental group of the circle in homotopy type theory. In *LICS*, pages 223–232, 2013.

[24] Peter Lefanu Lumsdaine. Weak omega-categories from intensional type theory. In *Proceedings of the 9th International Conference on Typed Lambda Calculi and Applications*, TLCA '09, pages 172–187, Berlin, Heidelberg, 2009. Springer-Verlag.

[25] Per Martin-Löf. An intuitionistic theory of types: predicative part. In H.E. Rose and J.C. Shepherdson, editors, *Logic Colloquium '73, Proceedings of the Logic Colloquium*, volume 80 of *Studies in Logic and the Foundations of Mathematics*, pages 73–118. North-Holland, 1975.

[26] Per Martin-Löf. Constructive mathematics and computer programming. In L. Jonathan Cohen, Jerzy ÅĄoÅŻ, Helmut Pfeiffer, and Klaus-Peter Podewski, editors, *Logic, Methodology and Philosophy of Science VI, Proceedings of the Sixth International Congress of Logic, Methodology and Philosophy of Science, Hannover 1979*, volume 104 of *Studies in Logic and the Foundations of Mathematics*, pages 153–175. North-Holland, 1982.

[27] Per Martin-Löf. *Intuitionistic type theory*, volume 1 of *Studies in Proof Theory*. Bibliopolis, 1984.

[28] Per Martin-Löf. An intuitionistic theory of types. In Giovanni Sambin and Jan M. Smith, editors, *Twenty-five years of constructive type theory (Venice, 1995)*, volume 36 of *Oxford Logic Guides*, pages 127–172. Oxford University Press, 1998.

[29] Ulf Norell. *Towards a practical programming language based on dependent type theory*. PhD thesis, Department of Computer Science and Engineering, Chalmers University of Technology and Göteborg University, Göteborg, Sweden, 2007.

[30] Pelayo, Álvaro and Warren, Michael A. Homotopy type theory and voevodsky's univalent foundations. *CoRR*, abs/1210.5658, 2012.

[31] B. Russell. *The Principles of Mathematics*. 1903.

[32] Michael Shulman. A formal proof that pi1 of S1 is Z (blog post), 2011.

[33] Michael Shulman. An interval type implies function extensionality (blog post), 2011.

[34] Thomas Streicher. A Model of Type Theory in Simplicial Sets. September 2011.

[35] The Univalent Foundations Program UF. *Homotopy type theory: Univalent foundations of mathematics*. first edition, 2013. Available online at homotopytypetheory.org/book.

[36] V. Voevodsky. Univalent Foundations Project. *a modified version of an NSF grant application*, 2010.

[37] Michael A. Warren. The strict ω-groupoid interpretation of type theory. Hart, Bradd (ed.) et al., Models, logics, and higher-dimensional categories: A tribute to the work of Mihály Makkai. Proceedings of a conference, CRM, Montréal, Canada, June 18–20, 2009. Providence, RI: American Mathematical Society (AMS). CRM Proceedings and Lecture Notes 53, 291-340 (2011)., 2011.