A proposition is the (homotopy) type of its proofs

Steve Awodey[*]

January 8, 2016

> There are, at first blush, two kinds of construction involved: constructions of proofs of some proposition and constructions of objects of some type. But I will argue that, from the point of view of foundations of mathematics, there is no difference between the two notions. A proposition may be regarded as a type of object, namely, the type of its proofs. Conversely, a type A may be regarded as a proposition, namely, the proposition whose proofs are the objects of type A. So a proposition A is true just in case there is an object of type A.
>
> W.W. Tait [21]

1

Overview

Homotopy type theory is a new field devoted to a recently discovered connection between Logic and Topology — more specifically, between constructive type theory, which was originally invented as a constructive foundation for mathematics and now has many applications in the theory of programming languages and formal proof verification, and homotopy theory, a branch of algebraic topology devoted to the study of continuous deformations of geometric spaces and mappings. The basis of homotopy type theory is an interpretation of the system of intensional type theory into abstract homotopy theory. As a result of this interpretation, one can construct new kinds of models of constructive logic and study that system semantically, e.g. proving consistency and independence results. Conversely, constructive type theory can also be used as a formal calculus to reason about abstract homotopy. This is particularly interesting in light of the fact that the type theory used underlies several computational proof assistants, such as Coq and Agda; this allows one to use those systems to reason formally about homotopy theory and fully verify the correctness of definitions and proofs using these computer proof systems. Potentially, this could provide a useful tool for mathematicians working in fields like homotopy theory and higher category theory. Finally, new logical principles and constructions based on homotopical and higher categorical intuitions can be added to the system, providing a way to formalize many classical spaces and sophisticated mathematical constructions. Examples include the so-called *higher inductive types* and the *univalence axiom* of Voevodsky.

More broadly, *univalent foundations* is an ambitious new program for foundations of mathematics, proposed by Voevodsky, which is based roughly on homotopy type theory and intended to capture a very broad range of mathematics (I am reluctant to use the phrase "All of Mathematics", but there is nothing in particular that could not, in principle, be done). The new univalence axiom, which roughly speaking implies that isomorphic structures can be identified, and the general point of view that it promotes sharpen the expressiveness of the system and make it more powerful, so that new concepts can be isolated and new constructions can be carried out, and others that were previously ill-behaved (such as quotients) can be better controlled. The system is not only more expressive and powerful than previous type- and set-theoretic systems of foundations; it also has two further, distinct novelties: it is still amenable to computer formalizations, and it captures a conception of mathematics that is distinctly "structural". These

two seemingly unrelated aspects, one practical, the other philosophical, are in fact connected in a rather subtle way. The structural character of the system, which the univalence axiom requires and indeed strengthens, permits the use of a new "synthetic" style of foundational axiomatics which is quite different from conventional axiomatic foundations. One might call the conventional, set-theoretic style of foundations an "analytic" (or perhaps "bottom-up") approach, which "analyses" mathematical objects into constituent *material* (e.g. sets or numbers), or at least constructs appropriate "surrogate objects" from such material — think of real numbers as Dedekind cuts of rationals. By contrast, the "synthetic" (or "top-down") approach permitted by univalent foundations is based on describing the fundamental *structure* of mathematical objects in terms of their universal properties, which in type theory are given by rules of inference determining directly how the new objects map to and from all other ones. This fundamental shift in foundational methodology has the practical effect of simplifying and shortening many proofs by taking advantage of a more axiomatic approach, as opposed to the more laborious analytic constructions.[1] Indeed, in a relatively short time, a large amount of classical mathematics has already been developed in this new system: basic homotopy theory, category theory, real analysis, the cumulative hierarchy of set theory, and many other topics. The proofs of some very sophisticated, high-level theorems have now been fully formalized and verified by computer proof assistants — a foundational achievement that would be very difficult to match using conventional, "analytic" style foundational methods.

Indeed, this combination of a synthetic foundational methodology and a powerful computational implementation has the potential to give new life, and a new twist, to the old idea of reducing mathematics to a purely formal calculus. Explicit formalizations that were once too tedious or complicated to be done by hand can now be accomplished in practice with a combination of synthetic methods and computer assistance. This new formal reduction of mathematics raises again the epistemological question of whether, and in what sense, the type-theoretic basis of the formal system is purely "logical", and what this means about mathematics and the nature of a priori knowledge. That is a question of significant philosophical interest, but it is perhaps better pursued independently, once the mathematical issues related to the formalization itself are more settled.

---

[1]In a related context, it has been said that such an approach has "all the advantages of theft over honest toil", but the issue of how to *justify* the rules for new constructions can be separated from that of their expedience.

1 Type theory

In its current form, constructive type theory is the result of contributions made by several different people, working both independently and in collaboration. Without wanting to give an exhaustive history (for one such, see [12]), it may be said that essential early contributions were made by H. Curry, W. Howard, F.W. Lawvere, P. Martin-Löf, D.S. Scott, and W.W. Tait.

Informally, the basic system consists of the following ingredients:

- **Types**: $X, Y, \ldots, A \times B$, $A \to B, \ldots$, including both primitive types and type-forming operations, which construct new types from given ones, such as the product type $A \times B$ and the function type $A \to B$.

- **Terms**: $a : A$, $b : B$, $\ldots$, including variables $x : A$ for all types, primitive terms $b : B$, and term-forming operations like $\langle a, b \rangle : A \times B$ and $\lambda x.b(x) : A \to B$ associated to the type-forming operations.

One essential novelty is the use of so-called *dependent types*, which are regarded as "parametrized" types or *type families indexed over a type*.

- **Dependent Types**: $x : A \vdash B(x)$ means that $B(x)$ is a type for each $x : A$, and thus it can be thought of as a function from A to types. Moreover, one can have iterated dependencies, such as:

  $x : A \vdash B(x)$

  $x : A, y : B(x) \vdash C(x, y)$

  $x : A, y : B(x), z : C(x, y) \vdash D(x, y, z)$

  etc.

- **Dependent Type Constructors**: There are special type constructors for dependent types, such as the sum $\sum_{x:A} B(x)$ and product $\prod_{x:A} B(x)$ operations. Associated to these are term constructors that act on dependent terms $x : A \vdash b(x) : B(x)$, such as $\lambda x.b(x) : \prod_{x:A} B(x)$.

- **Equations:** As in an algebraic theory, there are then equations $s = t : A$ between terms of the same type, such as $(\lambda x.b(x))(a) = b(a) : B(a)$.

4

The entire system of constructive type theory is a formal calculus of such typed terms and equations, usually presented as a deductive system by formal rules of inference. For one modern presentation, see the appendix to [22]. This style of type theory is somewhat different from the Frege-Russell style systems of which it is a descendant. It was originally intended as a foundation for *constructive* mathematics, and it has a distinctly "predicative" character — for instance, it is usually regarded as open-ended with respect to the addition of new type- and term-forming operations, such as universes, so that one does not make use of the notion of "all types" in the way that set-theory admits statements about "all sets" via its first-order logical formulation. Type theory is now used widely in the theory of programming languages and as the basis of computerized proof systems, in virtue of its good computational properties.

Propositions as types

The system of type theory has a curious dual interpretation:

- On the one hand, there is the interpretation as **mathematical** objects: the types are some sort of constructive "sets", and the terms are the "elements" of these sets, which are being built up according to the stated rules of construction.

- But there is also a second, **logical** interpretation: the types are "propositions" about mathematical objects, and their terms are "proofs" of the corresponding propositions, which are being derived in a deductive system.

This is known as the *Curry-Howard correspondence*, and it can be displayed as follows:

0	1	$A + B$	$A \times B$	$A \to B$	$\sum_{x:A} B(x)$	$\prod_{x:A} B(x)$
$\bot$	$\top$	$A \vee B$	$A \wedge B$	$A \Rightarrow B$	$\exists_{x:A} B(x)$	$\forall_{x:A} B(x)$

For instance, regarded as propositions, A and B have a conjunction $A \wedge B$, a proof of which corresponds to a pair of proofs a of A and b of B (via the $\wedge$-introduction and elimination rules), and so the terms of $A \wedge B$, regarded as a type, are just pairs $\langle a, b \rangle : A \times B$ where $a : A$ and $b : B$. Similarly, a proof of the implication $A \Rightarrow B$ is a function f that, when applied to a proof $a : A$ returns a proof

$f(a) : B$ (*modus ponens*), and so $f : A \to B$. The interpretation of the existential quantifer $\exists_{x:A} B(x)$ mixes the two points of view: a proof of $\exists_{x:A} B(x)$ consists of a term $a : A$ and a proof $b : B(a)$; so in particular, when it can be proved, one always has an instance a of an existential statement. In classical logic, by contrast, one can use "proof by contradiction" to establish an existential statement without knowing an instance of it, but this is not possible here. This gives the system has a distinctly constructive character (which can be specified in terms of certain good proof-theoretic properties). This is one reason it is useful for computational applications.

Identity types

Under the logical interpretation above we now have:

- **propositional logic**: 0, 1, $A + B$, $A \times B$, $A \to B$,

- **predicate logic**: $B(x)$, $C(x, y)$, with the **quantifiers** $\prod$ and $\sum$.

It would therefore be natural to add a primitive relation representing equality of terms $x = y$ as a *type*. On the logical side, this would represent the proposition "x is identical to y". But what would it be *mathematically*? How are we to continue the above table:

0	1	$A + B$	$A \times B$	$A \to B$	$\sum_{x:A} B(x)$	$\prod_{x:A} B(x)$	?
$\bot$	$\top$	$A \vee B$	$A \wedge B$	$A \Rightarrow B$	$\exists_{x:A} B(x)$	$\forall_{x:A} B(x)$	$x = y$

We shall add to the system a new, primitive type of *identity* between any terms $a, b : A$ of the same type A:
$$\mathsf{Id}_A(a, b).$$

The *mathematical* interpretation of this identity type is what leads to the homotopical interpretation of type theory. Before we can explain that, however, we must first consider the rules for the identity types.

The **introduction** rule says that $a : A$ is always identical to itself:

$$r(a) : \mathsf{Id}_A(a, a)$$

The **elimination** rule is a form of what may be called "Lawvere's Law":[2]

$$\frac{c : \mathsf{Id}_A(a, b) \qquad x : A \ \vdash\ d(x) : R\big(x, x, r(x)\big)}{J(a, b, c, d) : R(a, b, c)}$$

That may look a bit forbidding when seen for the first time. Schematically, it is saying something like:

$$a = b\ \&\ R(x, x)\ \Rightarrow\ R(a, b).$$

Omitting the proof terms, this characterizes identity by saying that it is the *least* (or better: *initial*) reflexive relation.

The rules for identity types are such that if a and b are *syntactically equal* as terms, $a = b : A$, then they are also *identical* in the sense that there is a term $p : \mathsf{Id}_A(a, b)$. But the converse is not true: distinct terms $a \neq b$ may still be propositionally identical $p : \mathsf{Id}_A(a, b)$. This is a kind of *intensionality* in the system, in that terms that are identified by the propositions of the system may nonetheless remain distinct syntactically, e.g. different polynomial expressions may determine the same function. Allowing such syntactic distinctions to remain (rather than including a "reflection rule" of the form $p : \mathsf{Id}_A(a, b) \Rightarrow a = b$, as is done in "extensional type theory"), gives the system its good computational and proof-theoretic properties. It also gives rise to a structure of great combinatorial complexity.

Although only the syntactically equal terms $a = b : A$ are fully interchangeable everywhere, propositionally identical ones $p : \mathsf{Id}_A(a, b)$ are still interchangeable *salva veritate* in the following sense: assume we are given a type family $x : A \vdash B(x)$ (regarded, if you like, as a "predicate" on A), an identity $p : \mathsf{Id}_A(a, b)$ in A, and a term $u : B(a)$ (a "proof of $B(a)$"). Then consider the following derivation, using the identity rules.

$$\frac{u : B(a) \qquad \dfrac{p : \mathsf{Id}_A(a, b) \qquad \dfrac{\dfrac{\dfrac{x : A \vdash B(x)}{x : A, y : B(x) \vdash y : B(x)}}{x : A \vdash \lambda y.y : B(x) \to B(x)}}{p_* : B(a) \to B(b)}}{p_* u : B(b)}$$

[2]See [14] for a closely related principle.

Here $p_* = J(a, b, p, \lambda y.y)$. The resulting term $p_* u : B(b)$ (which is a derived "proof of $B(b)$") is called the *transport* of u along p. Logically, this just says

$$a = b \ \& \ B(a) \Rightarrow B(b),$$

i.e. that a type family over A must respect the identity relation on A. As we shall see below, the homotopy interpretation provides a different view of transport; namely, it corresponds to the familiar *lifting property* used in the definition of a "fibration of spaces":

$$
\begin{array}{lll}
B & u \dashrightarrow p_* u & \qquad (1) \\
\downarrow & & \\
A & a \xrightarrow{\ p\ } b &
\end{array}
$$

2   The homotopy interpretation

Given any terms $a, b : A$, we can form the identity type $\mathsf{Id}_A(a, b)$ and then consider its terms, if any, say $p, q : \mathsf{Id}_A(a, b)$. Logically, p and q are "proofs" that a and b are identical, or more abstractly, "reasons" or "evidence" that this is so. Can p and q be different? It was once thought that such identity proofs might themselves always be identical, in the sense that there should always be some $\alpha : \mathsf{Id}_{\mathsf{Id}_A(a,b)}(p, q)$; however, as it turns out, this need not be so. Indeed, there may be many distinct (i.e. non-identical) terms of an identity type, or none at all. Understanding the structure of such iterated identity types is one result of the homotopical interpretation.

Suppose we have terms of ascending identity types:

$$
\begin{aligned}
&a, \ b : A \\
&p, \ q : \mathsf{Id}_A(a, b) \\
&\alpha, \ \beta : \mathsf{Id}_{\mathsf{Id}_A(a,b)}(p, q) \\
&\qquad \ldots : \mathsf{Id}_{\mathsf{Id}_{\mathsf{Id}\ldots}}(\ldots)
\end{aligned}
$$

Then we can consider the following informal interpretation:

$$
\begin{array}{rcl}
\text{Types} & \rightsquigarrow & \text{Topological spaces} \\
\text{Terms} & \rightsquigarrow & \text{Continuous maps} \\
a : A & \rightsquigarrow & \text{Points } a \in A \\
p : \mathsf{Id}_A(a, b) & \rightsquigarrow & \text{Paths } p \text{ from } a \text{ to } b \\
\alpha : \mathsf{Id}_{\mathsf{Id}_A(a,b)}(p, q) & \rightsquigarrow & \text{Homotopies } \alpha \text{ from } p \text{ to } q \\
& \vdots &
\end{array}
$$

So for instance A may be a space with points a and b, and then an identity term $p : \mathsf{Id}_A(a, b)$ is interpreted as a path in A from a to b, i.e. a continuous function $p : [0, 1] \to A$ with $p0 = a$ and $p1 = b$. If $q : \mathsf{Id}_A(a, b)$ is another such path from a to b, a higher identity term $\alpha : \mathsf{Id}_{\mathsf{Id}_A(a,b)}(p, q)$ is then interpreted as a homotopy from p to q, i.e. a "continuous deformation" of p into q, described formally as a continuous function $\alpha : [0, 1] \times [0, 1] \to A$ with the expected behavior on the boundary of the square $[0, 1] \times [0, 1]$. Higher identity terms are likewise interpreted as higher homotopies.

Note that, depending on the choice of space A and points $a, b \in A$ and paths p, q, it may be that there are no homotopies from p to q because, for example, those paths may go around a hole in A in two different ways, so that there is no continuous way to deform one into the other. Or there may be many different homotopies between them, for instance wrapping different numbers of times around the surface of a ball. Depending on the space, this can become quite a complicated structure of paths, deformations, higher-dimensional deformations, etc. — indeed, the investigation of this structure is what homotopy theory is all about.

One could say that the basic idea of the homotopy interpretation is just to extend the well-known topological interpretation of the *simply-typed* λ-calculus [3, 6] (which interprets types as spaces and terms as continuous functions) to the *dependently typed* λ-calculus with Id-types. The essential new idea is then simply this:

> *An identity term* $p : \mathsf{Id}_A(a, b)$ *is a path in the space* A *from the point* a *to the point* b.

Everything else essentially follows from this one idea: the dependent types $x : A \vdash B(x)$ are then forced by the rules of the type theory to be interpreted as *fibrations*,

in the topological sense, since one can show from the rules for identity types that the associated map $B \to A$ of spaces must have the *lifting property* indicated in diagram (1) above (a slightly more intricate example shows that one can "lift" not only the endpoint, but also the entire path, and even a homotopy). The total Id-types $\sum_{x,y:A} Id_A(x, y)$ are naturally interpreted as *path spaces* A^I, and the maps $f, g : A \to B$ that are identical as terms of function type $A \to B$ are just those that are *homotopic* $f \sim g$.

   The homotopy interpretation was first proposed by the present author and worked out formally (with a student) in terms of Quillen model categories—a modern, axiomatic setting for abstract homotopy theory that encompasses not only the classical homotopy theory of spaces and their combinatorial models like simplicial sets, but also other, more exotic notions of homotopy (see [8]). The interpretation was shown to be *complete* in the logical sense by Gambino and Garner [10].[3] These results show that intensional type theory can in a certain sense be regarded as a "logic of homotopy", in that the system can be faithfully represented homotopically, and then used to reason formally about spaces, continuous maps, homotopies, and so on. The next thing one might ask is, how much general homotopy theory can be expressed in this way? It turns out that a surprising amount can be captured under this interpretation, as we shall now proceed to indicate.

The fundamental groupoid of a type

Like path spaces in topology, identity types endow each type with the structure of a *groupoid*: a category in which every arrow has an inverse.
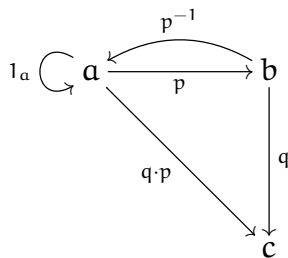
   The familiar *laws of identity*, namely reflexivity, symmetry, and transitivity are provable in type theory, and their proof terms therefore act on identity terms,

---

[3]There is a technical question related to the selection of path objects and diagonal fillers as interpretations of Id_A-types and elimination J-terms in a "coherent" way, i.e. respecting substitution of terms for variables; various solutions have been given, including [25, 23, 24, 19, 2].
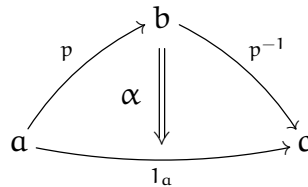
providing the *groupoid operations* of unit, inverse, and composition:

$$r : \mathsf{Id}(a, a) \qquad \text{reflexivity} \qquad a \longrightarrow a$$

$$s : \mathsf{Id}(a, b) \to \mathsf{Id}(b, a) \qquad \text{symmetry} \qquad a \rightleftarrows b$$

$$t : \mathsf{Id}(a, b) \times \mathsf{Id}(b, c) \to \mathsf{Id}(a, c) \qquad \text{transitivity}$$

The *groupoid laws* of units, inverses, and associativity also hold "up to homotopy", i.e. up to the existence of a higher identity term. This means that instead of e.g. $p^{-1} \cdot p = 1_a$, we have a higher identity term:

$$\alpha : \mathsf{Id}_{\mathsf{Id}} \left(p^{-1} \cdot p, 1_a \right)$$

as indicated in:

Indeed, this is just the same situation that one encounters in defining the fundamental group of a space in classical homotopy theory, where one shows e.g. that composition of paths is associative up to homotopy by reparametrization of the composites. In fact, in virtue of the homotopy interpretation, the classical case is really just an instance of the more general, type theoretic one.

Inspired by this occurence of type theoretic groupoids, Hofmann and Streicher [11] discovered an interpretation of the entire system of type theory into the category of all groupoids, which was a precursor of the homotopy interpretation. It was used, for instance, to establie the above mentioned fact that identity types may have elements that are not themselves identical.

The identity structure of a general type may actually be much richer than that of just a groupoid; as in homotopy theory, there may be non-trivial higher identities, representing higher homotopies between homotopies, and this structure may go on to higher and higher identities without ever becoming degenerate.
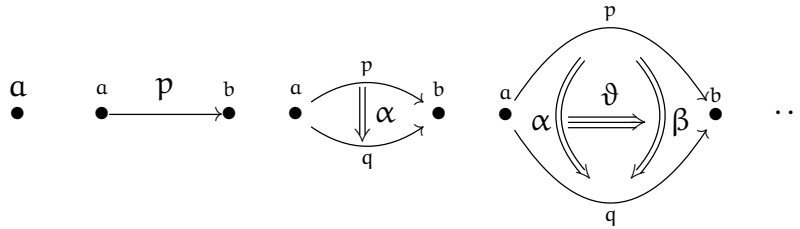
The resulting structure is that of an ω-*groupoid*, which is something that has also appeared elsewhere in mathematics — twice! As already mentioned, such "infinite-dimensional groupoids" also occur in homotopy theory, where the fundamental ω-groupoid of a space is an algebraic invariant that respects the homotopy type (according to Grothendieck's famous "homotopy hypothesis" these groupoids contain all the essential information of the space up to homotopy); but also in category theory, one has considered the idea of an ω-*category*, with not only objects and arrows between them, but also 2-arrows between arrows, 3-arrows between 2-arrows, and so on. It is indeed remarkable that the same notion has now appeared again in logic, as exactly the structure of iterated identity in type theory.[4]

Homotopy levels

One of the most useful new discoveries is that the system of all types is naturally stratified into "homotopy levels" by a hierarchy of definable conditions.[5]

At the lowest level are those types that are *contractible* in the following sense.

$$X \text{ is contractible} =_{\text{def}} \sum_{x:X} \prod_{y:X} \mathsf{Id}_X(x, y).$$

Under the logical reading, this condition says that X is a "singleton", in that there is an element $x : X$ such that everything $y : X$ is identical to it. So roughly, these are the types that have just one element, up to homotopy.

The next level consists of the *propositions*, defined as those types whose identity types are always contractible,

$$X \text{ is a proposition} =_{\text{def}} \prod_{x,y:X} \mathsf{Contr}(\mathsf{Id}_X(x, y)).$$

---

[4]See [17] and [9] for details.
[5]This concept is due to Voevodsky, cf. [24]. Also see [22], ch. 7.

It is not hard to see that such types are contractible *if* they are inhabited—thus they are like "truth values", either false (i.e. empty) or true (i.e. contractible), and then essentially uniquely so. In other words, the elements of a proposition contain no further information, other than the mere inhabitation of the proposition, which we interpret to mean that it holds.

At the next level are the *sets*, which are types whose identity relation is always a proposition:

$$X \text{ is a set} =_{\text{def}} \prod_{x,y:X} \mathsf{Prop}(\mathsf{Id}_X(x,y))$$

These types have the familiar, set-like behavior that the identity proofs, when they exist, are unique (again "up to homotopy").

Next come the types whose identity types are sets, which may be called *groupoids*, because they are like the algebraic groupoids just discussed:

$$X \text{ is a groupoid} =_{\text{def}} \prod_{x,y:X} \mathsf{Set}(\mathsf{Id}_X(x,y))$$

These types may have distinct identity proofs between elements, but all higher identity proofs are degenerate.

The general pattern is now clear:

$$X \text{ has homotopy level } n+1 =_{\text{def}} \prod_{x,y:X} \mathsf{H}_n(\mathsf{Id}_X(x,y))$$

Thus the types X of homotopy level $n+1$ (for which we write $\mathsf{H}_{n+1}(X)$) are the types whose identity relation is of homotopy level n; these types correspond to the higher-dimensional groupoids of category theory, when we think of identity terms as higher-dimensional arrows. To start the numbering, we may set the contractible types to be level 0.

The homotopy level of a type is the height at which the tower of iterated identity types becomes degenerate; under the homotopy interpretation this corresponds (up to a shift in numbering) to the notion of a space being a *homotopy n-type*, which is usually defined as the greatest n such that the n-th homotopy group is non-trivial. In each case, it is a measure of the complexity of the type/space — in the former case in terms of higher identities, and in the latter in terms of higher homotopies.

The recognition that types have these different degrees of complexity allows for a more refined version of the propositions-as-types idea, according to which
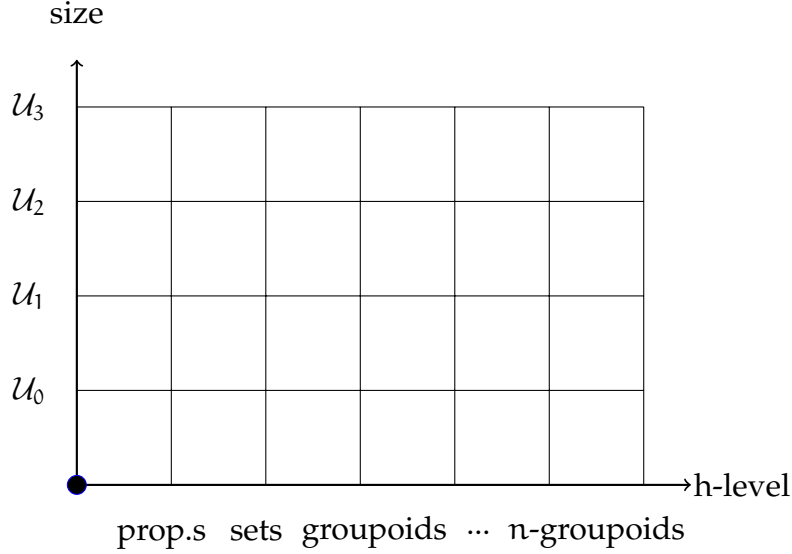
size

$\mathcal{U}_3$

$\mathcal{U}_2$

$\mathcal{U}_1$

$\mathcal{U}_0$

h-level

prop.s   sets   groupoids $\cdots$ n-groupoids

Figure 1: The 2D hierarchy of types

only those types that are "propositions" in the sense of the homotopy levels are read as bare *assertions*, while others are regarded more discriminately as *structured objects* of various kinds. Accordingly, a type family $x : A \vdash B(x)$ such that all values $B(x)$ are propositions can be regarded as a simple "predicate" (or a "relation" depending on the arity), while a family of sets, groupoids, etc. is viewed more accurately as a structure on A.

The stratification of types by homotopy levels gives us a new view of the mathematical universe, which is now seen to be arranged not only into the familiar, one-dimensional hierarchy of *size*, determined by a system of universes $\mathcal{U}_0, \mathcal{U}_1, \mathcal{U}_2, \ldots$, but also into a hierarchy of *homotopy levels*, which form a second dimension independent of the first (see Fig. 1).

3   Higher inductive types

The recognition and use of the notion of homotopy level of a type has made the entire system of type theory more expressive and powerful, for example by allowing greater control over the introduction of new type constructions. One such construction that was formerly problematic but is now better behaved is the construc-

tion of the *quotient type* $A/\sim$ of a type A by an equivalence relation $x, y : A \vdash x \sim y$. When $x \sim y$ is known to be a proposition for all $x, y : A$, then the quotient $A/\sim$ will be a set, and the introduction and elimination rules can be determined without difficulty. Such "set quotients" can be constructed, roughly speaking, as equivalence classes [24]; or they can be introduced axiomatically [22], essentially by stating rules that say that the identity type of $A/\sim$ is a *relation* (i.e. a family of propositions) that is freely generated by the equivalence relation $x \sim y$.

The latter, axiomatic approach is a special case of the very powerful construction method of *higher inductive types*, which are a systematic way of introducing new types with stipulated points, paths, higher paths, etc.. In order to explain this further, let us first recall how type theory deals with *ordinary* inductive types, like the natural numbers. The natural numbers $\mathbb{N}$ can be implemented as an inductive type via rules that may be represented schematically as:

$$\mathbb{N} := \left\{ \begin{array}{l} 0 : \mathbb{N} \\ s : \mathbb{N} \to \mathbb{N} \end{array} \right.$$

The terms 0 and s are the *introduction rules* for this type. The recursion property of $\mathbb{N}$ is captured by an *elimination rule*:

$$\frac{a : X \qquad f : X \to X}{\mathsf{rec}(a, f) : \mathbb{N} \to X}$$

which says that given any structure of the same kind as $\mathbb{N}$, there is a map $\mathsf{rec}(a, f)$ to it from $\mathbb{N}$, which furthermore preserves the structure, as stated by the following *computation rules*:

$$\mathsf{rec}(a, f)(0) = a \,,$$
$$\mathsf{rec}(a, f)(sn) = f(\mathsf{rec}(a, f)(n)) \,.$$

The map $\mathsf{rec}(a, f) : \mathbb{N} \to X$ is actually required to be the *unique* one satisfying the computation rules, a condition that can be ensured either with a further computation rule or by reformulating the elimination rule as a more general *induction principle* rather than a recursion principle (cf. [7]).

In more algebraic terms, one would say that $(\mathbb{N}, 0, s)$ is the *free structure* of this kind. We remark that it can be shown on the basis of these rules, and without further assumptions, that $\mathbb{N}$ *is a set* in the sense of the hierarchy of homotopy levels.

The circle

We now want to use the same method of specifying a new type by introduction and elimination rules (which amount to specifying the mappings to and from other types), but now with generating data that may include also elements of identity types, in addition to elements of the type itself and operations on it. A simple example is the following.

The homotopical circle $\mathbb{S}$ can be given as an inductive type involving one "base point" and one "higher-dimensional" generator:

$$\mathbb{S} := \begin{cases} \mathsf{base} : \mathbb{S} \\ \mathsf{loop} : \mathsf{Id}_\mathbb{S}(\mathsf{base}, \mathsf{base}) \end{cases}$$

The element $\mathsf{loop} : \mathsf{Id}_\mathbb{S}(\mathsf{base}, \mathsf{base})$ can therefore be regarded as a "loop" at the basepoint $\mathsf{base} : \mathbb{S}$, i.e. a path that starts and ends at base. The corresponding recursion property of $\mathbb{S}$ is then given by the following elimination rule,

$$\frac{a : X \qquad p : \mathsf{Id}_X(a, a)}{\mathsf{rec}(a, p) : \mathbb{S} \to X}$$

with computation rules,

$$\mathsf{rec}(a, p)(\mathsf{base}) = a \,,$$
$$\mathsf{rec}(a, p)_!(\mathsf{loop}) = p \,.$$

There is an obvious analogy to the rules for $\mathbb{N}$.[6] The map $\mathsf{rec}(a, p) : \mathbb{S} \to X$ is then moreover required to be unique up to homotopy, which again is achieved either with additional computation rules or a generalized elimination rule in the form of "circle induction" rather than "circle recursion" (see [22],[20]).

Conceptually, these rules suffice to make the structure $(\mathbb{S}, \mathsf{base}, \mathsf{loop})$ into the "free type with a point and a loop". To see that it actually behaves as it should to be the homotopical circle, one can verify that it has the correct homotopy groups (cf. [16]):

Theorem 1 (Shulman 2011). *The type-theoretic circle $\mathbb{S}$ has the following homotopy groups:*

$$\pi_n(\mathbb{S}) = \begin{cases} \mathbb{Z}, & \text{if } n = 1, \\ 0, & \text{if } n \neq 1. \end{cases}$$

---

[6]A map $f : A \to B$ induces a map on identities, taking each $p : \mathsf{Id}_A(a, b)$ to a term in $\mathsf{Id}_A(fa, fb)$ which we here write $f_!p$ (see [22], ch. 2).

The homotopy groups $\pi_n(X, x)$ for any type X and basepoint $x : X$ can be defined as usual in terms of loops at x in X, i.e. identity elements $\mathsf{Id}_X(x, x)$, "modulo homotopy", i.e. modulo higher identities. The proof of the above theorem can be given entirely within the system of type theory, and it combines methods from classical homotopy theory with ones from constructive type theory in a novel way, using Voevodsky's univalence axiom (a sketch is given in Section 5 below). The entire development has been fully formalized [16].

The interval

The homotopical interval $\mathbb{I}$ is also a higher inductive type, this time generated by the basic data:

$$\mathbb{I} := \left\{ \begin{array}{l} 0, 1 : \mathbb{I} \\ \mathsf{path} : \mathsf{Id}_{\mathbb{I}}(0, 1) \end{array} \right.$$

Thus $\mathsf{path} : \mathsf{Id}_{\mathbb{I}}(0, 1)$ represents a path from 0 to 1 in $\mathbb{I}$. The elimination and computation rules are analogous to those for the circle, but now with separate endpoints 0 and 1. So given any path $p : \mathsf{Id}_X(a, b)$ between points a and b in any type X, there is a unique (up to homotopy) map $\mathbb{I} \to X$ taking 0 to a, 1 to b, and path to p. This specification makes the structure $(\mathbb{I}, 0, 1, \mathsf{path})$ the "free type with a path".

In terms of this example, we can plainly compare the methodology behind the use of higher inductive types in homotopy type theory with the conventional approach of classical topology:

> In classical topology, we start with the *interval* and use it to define the notion of a *path*.
> In homotopy type theory, we start with the notion of a *path*, and use it to define the *interval*.

The notion of a *path*, recall, is a primitive one in our system, namely a term of identity type. In terms of these, one can then determine the interval $\mathbb{I}$ via its mappings, rather than the other way around.

Constructing higher inductive types

The higher inductive types mentioned so far were introduced *axiomatically*, by stating their basic rules. Can we instead *construct* them, similarly to the way that quotients can also be constructed from equivalence classes? One possible way to

do this is by what is sometimes called an "impredicative encoding", which is a construction that involves a quantification over "all types".

Consider first some related examples that are not *higher* inductive types, but which can be determined by such impredicative encodings. First let p and q be propositions and consider:

$$p \vee q \ =_{\text{def}} \ \forall_x \big[(p \Rightarrow x) \wedge (q \Rightarrow x) \Rightarrow x \big]$$

where the quantifier $\forall_x$ is over *all propositions* x (in a given universe). Among propositions, this type has the correct behavior to be the disjunction of p and q.

Next let A and B be sets, and consider:

$$A + B \ =_{\text{def}} \ \prod_X \big[(A \rightarrow X) \times (B \rightarrow X) \rightarrow X \big]$$

where the product $\prod_X$ is over *all sets* X (again, in a given universe). In order for this to actually be the coproduct among all sets, this specification requires an additional *coherence condition* saying that the transformations

$$\alpha_X : \big((A \rightarrow X) \times (B \rightarrow X) \big) \longrightarrow X$$

are *natural in* X, in a straightforward sense that we will not spell out here.

The same general idea can be used for the interval and the circle: for a type X, define the *path space* $I(X)$ and the *(free) loop space* $L(X)$ by:

$$I(X) \ =_{\text{def}} \ \sum_{x,y:X} \text{Id}_X(x, y)$$

$$L(X) \ =_{\text{def}} \ \sum_{x:X} \text{Id}_X(x, x)$$

Morally, we expect that $I(X) = \mathbb{I} \rightarrow X$ and $L(X) = \mathbb{S} \rightarrow X$, so as a first approximation we set:

$$\mathbb{I} \ =_{\text{def}} \ \prod_X \big[I(X) \rightarrow X \big]$$

$$\mathbb{S} \ =_{\text{def}} \ \prod_X \big[L(X) \rightarrow X \big]$$

where the product $\prod_X$ is now over all *groupoids* X (types of homotopy level 3, in a given universe). In order to get the correct elimination rules for these types,

we again add a further *coherence condition*, now involving higher-order naturality, which again we will not spell out here.[7]

The possibility of a "logical construction of the circle" and similar constructions of some other higher inductive types are current work in progress. At present they require either a general assumption of "impredicativity", or more specialized "resizing rules", or some other device to handle the shift in universes involved in the quantification over "all types".

Many basic spaces and constructions can be introduced directly as higher inductive types. These include, for example:

- higher spheres $\mathbb{S}^n$, mapping cylinders, tori, cell complexes,

- suspensions ΣA, homotopy pushouts,

- truncations, such as connected components $\pi_0(A)$ and "bracket" types $[A]$ (cf. [5]),

- (higher) homotopy groups π_n, Eilenberg-MacLane spaces $K(G, n)$, Postnikov systems,

- a Quillen model structure on the system of all types,

- quotients by equivalence relations and more general quotients,

- free algebras, algebras presented by generators an relations,

- the real numbers, the surreal numbers,

- the cumulative hierarchy of Zermel-Fraenkel sets.

The use of higher inductive types is a topic that is curently under very active investigation (see e.g. [18]).

---

[7]See [4]. The displayed formula for the circle was first considered by M. Shulman.

19

4 Univalence

Voevodsky has proposed a new foundational axiom to be added to type theory: the *univalence axiom*. It is motivated by the homotopy interpretation and makes precise the informal mathematical practice of "identifying" isomorphic objects. Especially when combined with higher inductive types, this new axiom is a powerful addition to the system. Although it is formally incompatible with the naive interpretation of type theory according to which all types are sets, it is provably consistent with the homotopical interpretation [13]. Its status as a constructive principle is still unsettled, however, and that question is the focus of much current research.

Isomorphism, equivalence, and invariance

In type theory, the notion of a *type isomorphism* $A \cong B$ is definable as usual; namely, the statement

there are $f : A \to B$ and $g : B \to A$ with $gf(x) = x$ and $fg(y) = y$

is formalized by the type of isomorphisms,

$$\mathrm{Iso}(A, B) \ =_{\mathrm{def}} \ \sum_{f:A\to B} \sum_{g:B\to A} \Big(\prod_{x:A} \mathsf{Id}_A(gf(x), x) \times \prod_{y:B} \mathsf{Id}_B(fg(y), y) \Big).$$

Under the logical reading, this type expresses exactly the preceding informal statement. The types A and B are isomorphic just if this type is inhabited by a term, which is then exactly an isomorphism between A and B. Here we see the propositions-as-types idea at work: a proof of the proposition $A \cong B$ is the same thing as a term of the type $\mathrm{Iso}(A, B)$, namely, an isomorphism.

There is also a more refined notion of *equivalence* of types $A \simeq B$ which adds a further "coherence" condition relating the identity terms of $\mathsf{Id}_A(gf(x), x)$ and $\mathsf{Id}_B(fg(y), y)$ via f and g (see [22], chapter 4). Since every isomorphism can be "promoted" to an equivalence, the latter condition is no "stronger" logically; nonetheless, it is worth the extra trouble to consider, because being an equivalence $f : A \simeq B$ is always a propositional condition, whereas being an isomorphism $f : \mathrm{Iso}(A, B)$ need not be one. Under the homotopy interpretation, the type $A \simeq B$ consists of the *homotopy equivalences* of spaces. The notion of type equivalence also subsumes

categorical equivalence (for groupoids), *isomorphism* (for sets), and *logical equivalence* (for propositions).

Now, it is an important fact about type theory that all "definable properties" $P(X)$ of types X (formally, any type expression with a type variable X) can be shown to respect type equivalence, in the sense that $A \simeq B$ and $P(A)$ imply $P(B)$; indeed, if $A \simeq B$ then $P(A) \simeq P(B)$. Briefly, we may say that all type-theoretic properties and concepts are *invariant*.[8] It therefore follows that equivalent types $A \simeq B$ are *indiscernable* within the system. Thus it is natural to ask how equivalence is related to the *identity* of the types A and B.

The univalence axiom

To reason internally about identity of types A and B, we need to add to the basic system a *type universe $\mathcal{U}$*, with an identity type,

$$\mathsf{Id}_{\mathcal{U}}(A, B)$$

expressing the relation of identity of types. The usual rules for identity then imply that identity of types implies their equivalence (because equivalence is reflexive), and so there is a comparison map,

$$\mathsf{Id}_{\mathcal{U}}(A, B) \to (A \simeq B).$$

The *univalence axiom* asserts that this map is itself an equivalence:

$$\mathsf{Id}_{\mathcal{U}}(A, B) \simeq (A \simeq B) \tag{UA}$$

So UA can be read: *identity is equivalent to equivalence.* It internally identifies those types that are equivalent, and therefore indiscernable. Indeed, since UA is an equivalence, there is a map coming back:

$$\mathsf{Id}_{\mathcal{U}}(A, B) \longleftarrow (A \simeq B).$$

Regarded as a map of types, we write this as $\mathsf{ua} : (A \simeq B) \to \mathsf{Id}_{\mathcal{U}}(A, B)$, which maps equivalences of types to identities between them. Read logically, ua is a proof of the statement that *equivalent types are identical*; thus in particular, isomorphic sets, groups, etc., are also identified.

---

[8]This of course does not hold in set theory. For example, consider the sets $\{\emptyset\}$ and $\{\{\emptyset\}\}$, which are isomorphic but are distinguished by the property $P(X) = (\exists x, y)\, x \in y \in X$.

Note that in the extended system with a universe $\mathcal{U}$, the univalence axiom is just what is needed to maintain the above-mentioned *invariance* of all "properties" $P(X)$:
$$A \simeq B \text{ implies } P(A) \simeq P(B),$$
for we can take $P(X) = \mathsf{Id}_{\mathcal{U}}(A, X)$ to see that equivalent types must be identified.

Note also that UA implies that $\mathcal{U}$, in particular, is not a set: for there are two distinct isomorphisms $2 \cong 2$, and these therefore correspond by UA to two distinct identity terms in $\mathsf{Id}_{\mathcal{U}}(2, 2)$.

Finally, we mention that the computational character of UA is still an open question. The system of type theory without it has some desirable properties, like the so-called "strong normalization" of terms, which implies the decidability of the syntactic equality relation of terms $a = b : A$. Adding a new axiom like univalence is likely to disrupt this property, but does it completely destroy the constructive character of the system? This is one of the open questions currently under active investigation.

5 Synthetic reasoning

In homotopy type theory, what is called the "synthetic" style of reasoning involves making use of the *primitive geometric element* introduced by taking the notion of a *path* as basic, rather than reducing it to maps from the real interval $[0, 1]$. This method is especially powerful in combination with the univalence axiom. By way of example, let us sketch the proof of the above-mentioned theorem that the fundamental group of the circle $\mathbb{S}$ is the integers $\mathbb{Z}$.

Computing $\pi_1\mathbb{S}$

To compute the fundamental group of the circle $\mathbb{S}$, just as in the classical proof, we shall make use of the "universal cover" (see Fig. 2). As a covering space and therefore a fibration, the universal cover will be a dependent type over $\mathbb{S}$, which, in the presence of a universe $\mathcal{U}$, is simply a map

$$\mathsf{cov} : \mathbb{S} \longrightarrow \mathcal{U}.$$

We can define such a type family using the recursion property of the circle; indeed, we just need the following data:
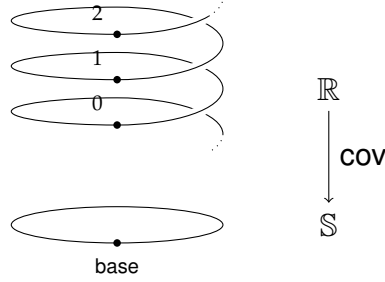
Figure 2: The winding map in classical topology

- a point $A : \mathcal{U}$

- a loop $p : \mathsf{Id}_{\mathcal{U}}(A, A)$

For the point A we shall take the integers $\mathbb{Z}$. By univalence, to give a loop $p :$ $\mathsf{Id}_{\mathcal{U}}(A, A)$ in $\mathcal{U}$, it suffices to give an equivalence $q : \mathbb{Z} \simeq \mathbb{Z}$. But since $\mathbb{Z}$ is a set, equivalences are just isomorphisms; so for q we can take the successor function $\mathsf{succ} : \mathbb{Z} \cong \mathbb{Z}$.

Definition (Universal cover of $\mathbb{S}$). The type family $\mathsf{cov} : \mathbb{S} \longrightarrow \mathcal{U}$ is given by circle recursion with

$$\mathsf{cov}(\mathsf{base}) = \mathbb{Z},$$
$$\mathsf{cov}(\mathsf{loop}) = \mathsf{ua}(\mathsf{succ}).$$

As in classical homotopy theory, we then use the universal cover to define the "winding number" of any path $p : \mathsf{Id}_{\mathbb{S}}(\mathsf{base}, \mathsf{base})$ by $\mathsf{wind}(p) = p_*(0)$, where p_* is the transport operation along p. This gives a map from the type $\Omega(\mathbb{S})$ of (based) loops in $\mathbb{S}$ to the integers,

$$\mathsf{wind} : \Omega(\mathbb{S}) \longrightarrow \mathbb{Z},$$

which can be shown to be inverse to the map $\mathbb{Z} \longrightarrow \Omega(\mathbb{S})$ defined by composing loop with itself a given number of times i,

$$i \mapsto \mathsf{loop}^i.$$

This proof can be formalized in a very efficient way, and the result is no longer than a conventional, "unformal" mathematical proof (see [16]). This is a real

advance over the traditional "analytic" style of formalization, which would require defining the circle as a subspace of the Euclidean plane $\mathbb{R}^2$, defining homotopies via continuous maps from the unit interval $[0, 1]$, using reparametrizations of paths to define their composition, defining the real numbers $\mathbb{R}$ and the winding map via trigonometric functions, and so on. To be sure, those are worthwhile mathematical objects and constructions in their own right! But by avoiding the need for them in this case, the synthetic approach seems to be somehow closer to the real "essence" of the homotopical fact being proved.

The cumulative hierarchy of sets

As a final example of synthetic reasoning in the full system of homotopy type theory with higher inductive types and univalence, we consider a somewhat "experimental" construction which gives the cumulative hierarchy of sets; see [22] for the details.

Given a universe $\mathcal{U}$, we make the *cumulative hierarchy* V of sets in $\mathcal{U}$ as a higher inductive type with the following generating constructors (we shall write $x =_V y$ for $\mathsf{Id}_V(x, y)$ for easier comparison with more familiar treatments):

1. For any type $A : \mathcal{U}$ and any map $f : A \to V$, there is a "set",

$$\mathsf{set}(A, f) : V.$$

   We think of $\mathsf{set}(A, f)$ as the image of A under f, i.e. the classical set

$$\{f(a) \mid a \in A\}.$$

2. For all $A : \mathcal{U}$ and $f : A \to V$ and $B : \mathcal{U}$ and $g : B \to V$ such that

$$\left(\forall a : A\; \exists b : B\; f(a) =_V g(b)\right) \wedge \left(\forall b : B\; \exists a : A\; f(a) =_V g(b)\right),$$

   we put in a path in V from $\mathsf{set}(A, f)$ to $\mathsf{set}(B, g)$.

3. The "set-truncation" constructor: for all $x, y : V$ and all $p, q : x =_V y$, we add a (higher) path from p to q.

In (2) we used the "logical" notation $\exists$ and $\forall$, etc., to indicate that we are working with the "propositional truncations" of the corresponding type theoretical operations Σ and Π (cf. [22] ch. 3).

Next, the membership relation $x \in y$ is defined for elements of V by

$$(x \in \mathsf{set}(A, f)) \quad =_{\mathrm{def}} \quad (\exists a : A. \, x =_V f(a)) .$$

One can then show entirely within the system that the resulting structure $(V, \in)$ satisfies *almost* all of the axioms of Aczel's constructive set theory CZF [1] (e.g. Strong Collection is missing).

Finally, assuming the usual axiom of choice just for those types that are *sets* in the sense of the homotopy levels, it then follows that $(V, \in)$ is a model of the full system of ZFC set theory (cf. [22], ch. 10.5). The proofs of these results make essential use of the univalence axiom and have been fully formalized in the Coq proof assistant (cf. [15]).

The system just mentioned is an interesting hybrid of classical set theory and constructive type theory that not only contains a model of ZFC but also many types of higher homotopy level that do not behave like classical sets. This provides a new, more refined view of one possible relationship between classical and constructive foundations. Whereas constructive foundations are usually regarded as incompatible with classical logic, in this system the classical sets form a subsystem of constructive type theory consisting of certain objects distinguished by a natural, intrinsic, structural property—namely that their identity relation is always a proposition.

References

[1] Peter Aczel. The type theoretic interpretation of constructive set theory. In A. MacIntyre, L. Pacholski, and J. Paris, editors, *Logic Colloquium '77*, volume 96 of *Studies in Logic and the Foundations of Mathematics*, pages 55–66. North-Holland, Amsterdam, 1978.

[2] S. Awodey. Natural models of homotopy type theory. arXiv:1406.3219.

[3] S. Awodey. Topological representation of the lambda-calculus. *Mathematical Structures in Computer Science*, 1(10):81–96, 2000.

[4] S. Awodey. Impredicative encodings of higher inductive types, 2015. In preparation.

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

[6] S. Awodey and C. Butz. Topological completeness for higher-order logic. *Journal of Symbolic Logic*, 3(65):1168–1182, 2000.

[7] S. Awodey, N. Gambino, and K. Sojakova. Inductive types in Homotopy Type Theory. In *Logic in Computer Science (LICS 2012)*, pages 95–104. IEEE Computer Society, 2012.

[8] S. Awodey and M. Warren. Homotopy-theoretic models of identity types. *Mathematical Proceedings of the Cambridge Philosophical Society*, 146(1):45–55, 2009.

[9] B. van den Berg and R. Garner. Types are weak ω-groupoids. *Journal of the London Mathematical Society*, 102(2):370–394, 2011.

[10] N. Gambino and R. Garner. The identity type weak factorisation system. *Theoretical Computer Science*, 409(3):94—109, 2008.

[11] M. Hofmann and T. Streicher. The groupoid interpretation of type theory. In *Twenty-five years of constructive type theory 1995*, volume 36 of *Oxford Logic Guides*, pages 83–111. Oxford Univ. Press, 1998.

[12] F. D. Kamareddine, T. Laan, and R. P. Nederpelt. *A modern perspective on type theory: from its origins until today*. Springer, 2004.

[13] C. Kapulkin, P. Lumsdaine, and V. Voevodsky. The simplicial model of univalent foundations. arXiv:1211.2851v1, 2012.

[14] F.W. Lawvere. Equality in hyperdoctrines and comprehension schema as an adjoint functor. In A. Heller, editor, *Proceedings of the AMS Symposium on Pure Mathematics*, volume XVII, pages 1–14, 1970.

[15] J. Ledent. The cumulative hierarchy of sets, 2014. Post on the Homotopy Type Theory blog.

[16] D. Licata and M. Shulman. Calculating the fundamental group of the circle in Homotopy Type Theory. In *Logic in Computer Science (LICS 2013)*, pages 223–232. IEEE Computer Society, 2013.

[17] P. Lumsdaine. Weak ω-categories from intensional type theory. *Logical Methods in Computer Science*, 6:1–19, 2010.

[18] P. Lumsdaine. Higher inductive types: a tour of the menagerie, 2011. Post on the Homotopy Type Theory blog.

[19] P. Lumsdaine and M. Warren. The local universes model: An overlooked coherence theorem. *ACM Transactions on Computational Logic*, 2015.

[20] K. Sojakova. Higher inductive types as homotopy-initial algebras. Technical Report CMU-CS-14-101, Carnegie Mellon University, 2014. Available at `http://reports-archive.adm.cs.cmu.edu/`.

[21] W.W. Tait. The law of excluded middle and the axiom of choice. In A. George, editor, *Mathematics and Mind*, pages 45–70. Oxford University Press, 1994.

[22] The Univalent Foundations Program, Institute for Advanced Study. *Homotopy Type Theory - Univalent Foundations of Mathematics*. Univalent Foundations Project, 2013.

[23] B. van den Berg and R. Garner. Topological and simplicial models of identity types. *ACM Transactions on Computational Logic*, 13(1), 2012.

[24] V. Voevodsky. Notes on type systems, 2009. Available from the author's web page.

[25] M. Warren. *Homotopy-theoretic aspects of constructive type theory*. PhD thesis, Carnegie Mellon University, 2008.