Brouwer’s fixed-point theorem in real-cohesive homotopy type theory

MICHAEL SHULMAN†

Department of Mathematics, University of San Diego, San Diego, CA 92110, USA
Email: shulman@sandiego.edu

Received 9 April 2016; revised 30 March 2017

We combine homotopy type theory with axiomatic cohesion, expressing the latter internally with a version of ‘adjoint logic’ in which the discretization and codiscretization modalities are characterized using a judgemental formalism of ‘crisp variables.’ This yields type theories that we call ‘spatial’ and ‘cohesive,’ in which the types can be viewed as having independent topological and homotopical structure. These type theories can then be used to study formally the process by which topology gives rise to homotopy theory (the ‘fundamental $\infty$-groupoid’ or ‘shape’), disentangling the ‘identifications’ of homotopy type theory from the ‘continuous paths’ of topology. In a further refinement called ‘real-cohesion,’ the shape is determined by continuous maps from the real numbers, as in classical algebraic topology. This enables us to reproduce formally some of the classical applications of homotopy theory to topology. As an example, we prove Brouwer’s fixed-point theorem.

1. Introduction

1.1. On spaces, types and $\infty$-groupoids

Homotopy type theory (Univalent Foundations Program 2013) is an emerging field that connects homotopy theory and higher category theory with constructive type theory. The homotopy-theoretic semantics for type theory (Awodey and Warren 2009; Kapulkin and Lumsdaine 2012) enables us to view the types in type theory as homotopical objects. This motivates new rules and axioms for type theory, such as Voevodsky’s univalence axiom (UA) and higher inductive types (HITs), which allow us to do synthetic homotopy theory. That is, we can construct homotopical objects in type theory and prove theorems about them there, such as calculating homotopy groups of spheres (Univalent Foundations Program 2013, Chapter 8). The homotopy-theoretic semantics imply that such formal theorems automatically yield proofs of corresponding classical results.

However, it is important to understand that at least on the surface, a theorem in synthetic homotopy theory is a different statement about different objects than a theorem

† This material is based on research sponsored by The United States Air Force Research Laboratory under agreement number FA9550-15-1-0053. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the United States Air Force Research Laboratory, the U.S. Government, or Carnegie Mellon University.
of the same name in classical homotopy theory. For instance, in the synthetic theorem that \( \pi_1(S^1) = \mathbb{Z} \) (Licata and Shulman 2013):

— \( S^1 \) is a higher inductive type, freely generated by a basepoint and a loop.
— \( \pi_1(X) \) is the 0-truncation of the loop space \( \Omega(X) \), where the 0-truncation is a higher inductive type that ‘kills all information above dimension 0.’
— The loop space \( \Omega(X) \) involves ‘paths’ that are an essentially undefined primitive notion, like ‘point’ and ‘line’ in axiomatic geometry. They are given meaning by the rules governing them (namely, those of Martin-Löf’s intensional identity type).

This should be contrasted with the meanings of the same words in the theorem of classical homotopy theory that is denoted \( \pi_1(S^1) = \mathbb{Z} \):

— \( S^1 \) is the topological space \( \{(x, y) \in \mathbb{R} \times \mathbb{R} | x^2 + y^2 = 1\} \).
— \( \pi_1(X) \) is the set of path-components of the loop space \( \Omega(X) \).
— The loop space \( \Omega(X) \) (and the notion of path component) involves paths that are defined as continuous functions \([0, 1] \to X\) out of the topological unit interval.

**Remark 1.1.** In contrast to Univalent Foundations Program (2013), I will use \( S^1 \) for the topological circle \( \{(x, y) \in \mathbb{R} \times \mathbb{R} | x^2 + y^2 = 1\} \), hoping that the font will serve as a mnemonic for its relationship to the real numbers \( \mathbb{R} \). This leaves \( S^1 \) for the higher inductive one.

The connection between these worlds that justifies using the same terminology in both cases can be described as follows:

1. In homotopy theory, we study objects that may be called homotopy spaces or \( \infty \)-groupoids, which have objects, identifications between those objects, higher identifications between identifications, and so on.
2. A topological space \( X \) gives rise to an \( \infty \)-groupoid, traditionally called its fundamental \( \infty \)-groupoid, whose objects are the points of \( X \), whose identifications are continuous paths in \( X \), whose higher identifications are continuous homotopies in \( X \), and so on.
3. Type theory admits semantics in \( \infty \)-groupoids.
4. Finally, the fundamental \( \infty \)-groupoid of the topological circle \( S^1 \) is the same \( \infty \)-groupoid that furnishes semantics for the higher inductive circle \( S^1 \) in type theory (and similarly for paths, truncations, and so on).

A classical algebraic topologist would not bring up \( \infty \)-groupoids, of course, but just talk about topological spaces directly. However, I find it clarifying to do so, because it enables us to distinguish ways in which topological spaces and \( \infty \)-groupoids behave differently. Specifically, a topological space can be defined up to homeomorphism, whereas an \( \infty \)-groupoid is (at least the way I use the word) only ever defined up to homotopy equivalence. For instance, as topological spaces, a cylinder is distinct from a Möbius strip (e.g. one is orientable and one is not); but they have the same fundamental \( \infty \)-groupoid (which is, in fact, also the same as that of \( S^1 \)).
This is important because the homotopy-theoretic semantics of type theory lands in ∞-groupoids, and not in topological spaces.† Thus, inside of type theory, we should think of types as ∞-groupoids and not as topological spaces. In particular, although it is common in homotopy type theory to use terminology borrowed from topology such as ‘path’ and ‘circle,’ these words have a priori nothing to do with their topological versions which can also be defined inside of type theory. That is, since type theory is rich enough to encode all of mathematics, we can define in it the real numbers \( \mathbb{R} \) and thus also the topological circle \( S^1 = \left\{ (x, y) : \mathbb{R} \times \mathbb{R} \mid x^2 + y^2 = 1 \right\} \); but this type is quite different from the higher inductive circle \( S^1 \). For instance, the former is a set (0-truncated) with infinitely many distinct points, while the latter is not a set and has ‘only one point’ (technically, it is 0-connected).

This frequently causes confusion among newcomers to homotopy type theory, who struggle to understand the meaning of ‘path’ because it both is, and is not, like the topological concept after which it is named. For this reason among others, in this paper, I will say identification or equality rather than ‘path’ when speaking of the synthetic notion (i.e. elements of identity types).

More importantly, however, this means there is something missing from homotopy type theory, because there is more to algebraic topology than the study of ∞-groupoids in their own right. The process by which they arise from topological spaces (the fundamental ∞-groupoid) is also important. This often becomes clear when we are concerned with applications of algebraic topology to fields such as geometry and physics, which have no intrinsic interest in homotopy theory. As a simple example, consider the standard homotopy-theoretic proof of Brouwer’s fixed-point theorem.

**Theorem 1.2.** Let \( \mathbb{D}^2 \) denote the topological disc \( \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 \leq 1\} \). Then any continuous map \( f : \mathbb{D}^2 \to \mathbb{D}^2 \) has a fixed point.

**Proof.** Suppose \( f : \mathbb{D}^2 \to \mathbb{D}^2 \) is continuous with no fixed point. For any point \( z \in \mathbb{D}^2 \), draw the ray from \( f(z) \) to \( z \) (which is well-defined since they are distinct) and keep going until you hit the boundary of \( \mathbb{D}^2 \) (see Figure 1). Call that point of intersection \( r(z) \); then \( r \) defines a continuous map from \( \mathbb{D}^2 \) to its boundary, which is the topological circle \( S^1 \). Moreover, if \( z \in S^1 \) to begin with, then there is no need to keep going once we get from \( f(z) \) to \( z \), so that \( r(z) = z \) in this case. Thus, \( S^1 \) would be a retract of \( \mathbb{D}^2 \). It follows that \( \pi_1(S^1) \) would be a retract of \( \pi_1(\mathbb{D}^2) \); but the latter is trivial while the former is not.

Note how crucial it is that \( S^1 \) is a topological space, with points that can be specified exactly and not just up to homotopy. Indeed, the statement of the theorem makes no reference to homotopy theory.

---

† To be completely precise, at least with current technology (Kapulkin and Lumsdaine 2012), it lands in simplicial sets, which are a different model for ∞-groupoids. Like a topological space and unlike an ∞-groupoid, a simplicial set can be defined up to isomorphism rather than equivalence; but the notion of isomorphism for simplicial sets is totally different from the notion of isomorphism for topological spaces. Moreover, at most a very small amount of this extra strictness of simplicial sets beyond ∞-groupoids is visible to type theory, and conjecturally none at all.
Fig. 1. A retraction from a self-map.

With present technology, this theorem is completely inaccessible to homotopy type theory. Of course, the use of proof by contradiction requires the law of excluded middle, but this can be assumed, as discussed by Univalent Foundations Program (2013, Chapter 3). The real problem is that inside of type theory, we have no way to relate $S^1$ to $S^1$, so that we cannot make the jump into homotopy theory in the last step. (We could, of course, repeat the classical definition of $\pi_1$ as a quotient of a set of continuous paths, and likewise the classical proof that $\pi_1(S^1) \cong \mathbb{Z}$, but then we would have gained nothing from homotopy type theory.)

What we need is some way to define the fundamental $\infty$-groupoid inside of type theory. The most natural way to interpret this is that given a topological space $X$ in type theory, we want a type whose synthetic paths (i.e. elements of its identity type) are the topological paths in $X$ (i.e. continuous maps $[0, 1] \to X$), and so on. It is natural to try to construct such a thing as a higher inductive type, but there are at least two problems with this. First, it seems that we would need infinitely many constructors to handle paths of all dimension, which at present we do not know how to do in the finitary syntax of type theory. And second, if we succeed in doing it, it is not immediately clear how we would show that this construction takes $S^1$ to $S^1$ without essentially copying again the classical proof, thereby negating any benefit from synthetic homotopy theory.

1.2. Combining synthetic topology and synthetic homotopy theory

In this paper, we will take a different approach: we make the topology synthetic, as well as the homotopy theory. That is, in addition to carrying a structure of synthetic $\infty$-groupoid, with identifications and higher identifications, every type also carries a structure of synthetic topological space, with some sort of ‘cohesion’ attaching nearby elements together. (I am being deliberately vague here; I don’t want to demand that a synthetic topology is necessarily specified in terms of open sets the way a classical topology usually is.)
It is well known that type theory admits topological semantics, in which types are interpreted by topological spaces or other similar objects. Indeed, this has been known for much longer than the homotopical interpretations. Moreover, just as type theory can be enhanced with homotopically motivated axioms such as univalence, it can be enhanced with topologically motivated axioms such as Browerian continuity principles or an axiomatic `Sierpinski space,’ leading to synthetic topology (Bauer and Lešnik 2012; Escardó 2004a,b; Taylor 2010).

What we will do is to combine these two kinds of interpretation, so that each type will have both an \(\infty\)-groupoid structure and a topological structure. In contrast to the situation in classical algebraic topology, where the same structure is viewed at different times either as topological or homotopical, in our world the topological and homotopical structures will be unrelated. The objects of our motivating model will be topological \(\infty\)-groupoids, which can be thought of as \(\infty\)-groupoids together with topologies on their points, identifications, higher identifications, and so on, making all of the operations continuous. Here are some examples to get the reader’s intuition up to speed.

— An ordinary topological space yields a topological \(\infty\)-groupoid with no non-trivial identifications, in the same way that a set yields an ordinary \(\infty\)-groupoid.
— An ordinary \(\infty\)-groupoid can be given the discrete topology (in all dimensions, i.e. on points, identifications, and so on).
— An ordinary \(\infty\)-groupoid can also be given the indiscrete topology.
— Recall that an ordinary group \(G\) can be ‘delooped’ to an \(\infty\)-groupoid \(K(G,1)\), with one point \(b\) and \(G\) forming the identifications from \(b\) to itself. If \(G\) is a topological group, then we have an analogous \(K(G,1)\) that is a topological \(\infty\)-groupoid, but remembering the topology on \(G\) as the space of identifications.

Note that here it starts to matter even more that we distinguish between paths and equalities: a topological \(\infty\)-groupoid has both paths and equalities; and it can also have equalities between paths, and paths between equalities.

At this point, we are presented with several problems.

1. How should we define precisely what we mean by a ‘topological \(\infty\)-groupoid’ (in set-theoretic foundations, to serve as semantics for type theory)?
2. What new rules or axioms can we bring into type theory, motivated by such a model, that will enable us to relate \(S^1\) to \(S^1\)? Continuity principles and the Sierpinski space are not well-adapted to saying interesting things about types that are not 0-truncated.
3. The classical proof of Brouwer’s fixed-point theorem uses the law of excluded middle. But unlike homotopical interpretations, topological interpretations are generally incompatible with excluded middle; how can we resolve this apparent contradiction?

Fortunately, all three have the same solution. Let us start with question 3. The problem with excluded middle in topological models is that it may not hold continuously: e.g. a

---

\(\dagger\) This is different from what a classical algebraic topologist means by ‘\(BG\)’ for a topological group \(G\). The latter is the ordinary \(\infty\)-groupoid with one object whose identifications are points of \(G\), whose 2-identifications are paths in \(G\), etc.—i.e. the delooping of the fundamental \(\infty\)-groupoid of \(G\).

\(\ddagger\) Modulo dealing correctly with endpoints, equalities between paths and paths between equalities are actually the same, but are different from paths between paths and from equalities between equalities.
subspace \( U \subseteq X \) and its complement \( X \setminus U \) will together contain all the points of \( X \), but the map from their coproduct \( U + (X \setminus U) \) to \( X \) does not have a continuous section because the topology on the domain is different. Thus, we cannot expect to have the full law of excluded middle in a topological model.

However, we can recover a form of excluded middle if we enhance our model and our type theory with the ability to talk about discontinuous functions in addition to continuous ones. Then it will be consistent to say something like ‘discontinuously, for all \( P \) we have \( P \lor \neg P \).’ There are at least three ways to add discontinuous functions to a world where everything is continuous: we could add a new basic notion of ‘discontinuous map;’ we could add an operation that retopologizes a space discretely (since every function out of a discrete space is continuous), or we could dually add an operation that retopologizes a space indiscretely (since every function into an indiscrete space is continuous). In fact, we will do all three; we will denote the two retopologization operations by \( \flat \) and \( \sharp \), respectively. Categorically, \( \flat \) is a coreflection and \( \sharp \) is a reflection; in type theory, we call them modalities.

Now let us move on to question 2. Here our answer comes from Lawvere (2007) (see also Lawvere and Menni 2015), who has proposed an axiomatization of ‘categories of cohesive spaces’ based entirely on adjoint functors and their properties. This is convenient because it is easy to generalize to homotopy theory (regarded as \((\infty,1)\)-category theory), since we have well-behaved notions of adjoint \(\infty\)-functors. Lawvere has considered many axioms, but the basic setup is a string of four adjoint functors:

\[
\begin{array}{cccc}
& p! & p^* & p_* & p^! \\
\downarrow & \downarrow & \downarrow & \downarrow & \\
sets & & & & \cohesive\ spaces
\end{array}
\]

in which \( p^* \) (hence also \( p^! \)) is fully faithful. The idea is that \( p_* \) finds the underlying set of a space, \( p^* \) equips a set with the discrete topology and \( p^! \) equips it with the indiscrete topology. (We will return to the meaning of \( p! \) below.) It follows that \( p^* p_* \) is a coreflection into the subcategory of discrete spaces while \( p^! p_* \) is a reflection into the subcategory of indiscrete spaces; thus, they coincide with the modalities \( \flat \) and \( \sharp \) mentioned above. We can \(\infty\)-ize this whole setup by simply replacing categories with \((\infty,1)\)-categories, ‘sets’ with ‘\(\infty\)-groupoids,’ and adjoint functors with adjoint \(\infty\)-functors; this has been studied extensively by Schreiber (2013).

Finally, there is a topos-theoretic side to Lawvere’s work which informs our answer to question 1. If the category of cohesive spaces is to be a topos, then the existence of this adjoint string says that it must be a local and locally connected topos. And we know how to construct local and locally connected toposes, by taking categories of sheaves on Grothendieck sites with special properties (see e.g. Johnstone 2002, C3.3.10 and C3.6.3(d)). Thus, it stands to reason that by taking \(\infty\)-sheaves on similar sorts of sites, we should be able to construct cohesive \((\infty,1)\)-toposes, and this is in fact the case, as shown by Schreiber (2013). The objects of the particular cohesive \((\infty,1)\)-topos we are interested in are known as topological \(\infty\)-groupoids or topological stacks.
1.3. Real-cohesive homotopy type theory

At this point, we may appear to have wandered rather far from our original motivating problem, namely that we need an ‘internal’ way to construct the fundamental $\infty$-groupoid. In fact, however, we have snuck up behind it and are about ready to pounce. The secret lies with Lawvere’s fourth functor $p!$. In his 1-categorical context, it constructs the set of connected components of a space, since a map from any space to a discrete one must be constant on every component. However, when we generalize to the $(\infty, 1)$-categorical context, the analogous $p!$ is the fundamental $\infty$-groupoid functor! Unfortunately, to explain how this comes about would take too much space here. Some insight may be gained from the proof of Theorem 9.5 below; for more details, see Schreiber (2013, Proposition 4.3.32) or Carchedi (2016, Section 3).

Just as $p^*p_*$ and $p_!p^!$ induce the operations $\flat$ and $\sharp$ on the type theory of cohesive spaces, the composite $p^*p_!$ induces a third modality, a reflection into discrete spaces that we call the shape and denote $\flat$. (See Remark 9.7 for discussion of the name ‘shape.’) With $\flat$ in our type theory, we can at least state the desired relation between the circles: we should have $\flat S^1 = S^1$. The existence of $\flat$ alone does not suffice to prove this; however, what we need is an additional axiom guaranteeing that $\flat$ is in fact constructed out of continuous paths indexed by intervals in $\mathbb{R}$, as we expect. To be precise, we assert that $\flat$ is the ‘internal localization’ at $\mathbb{R}$, as in Schreiber (2013, Proposition 3.9.4) and Dugger (2001, Proposition 8.3).

At last, we have a type theory in which we can reproduce the classical proof of Brouwer’s fixed-point theorem, using synthetic homotopy theory in the appropriate place. We perform the topological part of the proof in type theory, using classical axioms such as our modified form of excluded middle where appropriate. Then, at the end, we apply $\flat$, use our calculation that $\flat S^1 = S^1$, and then appeal to the synthetic proof of $\pi_1(S^1) = \mathbb{Z}$ to reach a contradiction.

The type theory formed by the addition of $\flat$ to $\sharp$ and $\sharp$ is called cohesive homotopy type theory, and was already sketched by Schreiber and Shulman (2012). Our theory is a further enhancement of this which I call real-cohesive homotopy type theory, due to the special role played by the real numbers in defining $\flat$ using continuous paths. As we will see, the simple axiom of real-cohesion has many striking consequences, especially combined with our modified classical axioms. In addition to Brouwer’s fixed-point theorem, it implies versions of the Intermediate Value Theorem, and characterizes the internal functions $\mathbb{R} \to \mathbb{R}$ as being precisely those that are $\varepsilon$-$\delta$ continuous in the usual sense. The present paper is a formal development of just enough real-cohesive homotopy type theory to enable the proof of Brouwer’s fixed-point theorem, which we give in Section 12.

1.4. The problem of comonadic modalities

It should be noted before proceeding that there are in fact many subtly different ways to represent the modalities $\flat$, $\flat$, and $\sharp$ in type theory. The problems in trying to do this arise from the fact that everything in ordinary type theory happens in an arbitrary context,
which means that all type-theoretic operations must correspond to category-theoretic operations that apply in each slice category and are stable under pullback. For instance, the naïve type-theoretic notion of ‘reflective subuniverse’ (Univalent Foundations Program 2013, Definition 7.7.1) corresponds categorically not to a mere reflective subcategory, but to a ‘reflective subfibration’ of the self-indexing.

For $\sharp$, this is not a problem, since it is a left exact reflector (being a composite of two right adjoints), and any left-exact-reflective subcategory admits a canonical extension to a reflective subfibration. For $\flat$, which is a reflector but not left exact, it is more of an issue; however, as long as $f$ can be defined by an internal localization—which categorically means that our topos is not just locally connected but stably locally connected (see Johnstone 2011)—it can be extended to a reflective subfibration. (The category-theoretic side of such modalities will be studied elsewhere; presently, we are concerned with their type-theoretic manifestation, with the category theory providing only motivation. Internally, in type theory, modalities of this sort were sketched briefly by Univalent Foundations Program (2013, Section 7.7) and will be studied in detail by Rijke, Shulman, and Spitters (2017).)

The real problem arises with the modality $\flat$, which is a coreflector, and cannot be extended to a ‘coreflective subfibration.’ In fact, one can prove a ‘no-go theorem’ (Theorem 4.1) that the only coreflective subfibrations are those of the form $\square_U(A) = A \times U$ for a subterminal object $U$, and $\flat$ is certainly not of this form. Thus, we must find some way to represent $\flat$ that prevents it from being applied in arbitrary contexts.

Schreiber and Shulman (2012) did this by using $\sharp\text{Type}$. We cannot have $\flat$ as an operation $\text{Type} \to \text{Type}$ on the universe itself, but we can have it as an operation $\sharp\text{Type} \to \sharp\text{Type}$, i.e. a ‘discontinuous’ self-map of the universe. This choice had the advantage of being formalizable in existing proof assistants, but the disadvantage that it required developing large amounts of theory of $\sharp\text{Type}$ as ‘the external category of types’ (which was mostly omitted by Schreiber and Shulman 2012). Even worse, there is no way to ‘escape’ from $\sharp\text{Type}$, but it should be the case that if we have a particular type defined in the empty context, such as $\mathbb{N}$, we can define $\flat\mathbb{N}$ as an actual type, rather than just an element of $\sharp\text{Type}$. Finally, in Schreiber and Shulman (2012), all the modalities were obtained by asserting axioms, but for good formal behaviour of a type theory, it is preferable to use rules rather than axioms, since axioms interfere with canonicity.

For these reasons, in this paper, we take a different approach, based on the ‘judgemental reconstruction’ of modal logic by Pfenning and Davies (2001). We modify our base type theory by introducing a new sort of hypothesis $x : A$, which we call crisp, intended to mark whether dependence on a variable is continuous or discontinuous. Based on this, we can then introduce $\flat$ and $\sharp$ using the usual sort of rules for type constructors: formation, introduction, elimination and computation.

Our rules for $\flat$ are almost exactly those of Pfenning and Davies (2001) for $\square$. If $B$ is a type depending only on crisp hypotheses, we can form $\flat B$, and similarly if a term $M : B$ depends only on crisp hypotheses, we can form $M^* : \flat B$, while an element of $\flat B$ can always be assumed to be of this form. However, although our $\sharp$ on its own behaves like the $\bigcirc$ of Pfenning and Davies (2001), its rules are quite different (in particular, we have no judgement ‘$M \simeq A$’ like in Pfenning and Davies (2001)) and make it automatically right
adjoint to $\mathcal{B}$. The resulting type theory, which requires no axioms yet, we call spatial type theory; we expect that it corresponds semantically to local toposes, which have $p^* \dashv p_* \dashv p_!$ but no $p!$. It includes as a fragment the ‘modal logic of local toposes’ of Awodey and Birkedal (2003) and Awodey, Birkedal, and Scott (1999).

In particular, the rules of $\mathcal{B}$ stipulate that it can only be applied in a context of purely discontinuous dependence, circumventing the no-go theorem. In addition to resolving all the above problems with $\sharp Type$, this has the additional advantage that we can apply $\mathcal{B}$ not only in a totally empty context, but in a context of other discrete spaces. Categorically speaking, this means we treat the topos of cohesive spaces as indexed over the base topos via $p^*$, which is exactly the right thing to do for any geometric morphism (see Johnstone 2002, B3.1.2 and Schreiber 2013, Theorem 3.4.20).

**Remark 1.3.** One might argue that it would be better to include a separate kind of type representing objects in the base category, and type constructors representing the functors $p^*$, $p_*$, etc. rather than the monads and comonads they induce. This would be in line with the ‘judgemental deconstruction’ of Reed (2009) that decomposes $\Box$ and $\bigcirc$ into pairs of adjoint functors. It could be that this is indeed better; in fact, the current rules for $\mathcal{B}$ and $\sharp$ were deduced from a generalization of Reed (2009) developed by Licata and Shulman (2016).

However, for now, I have chosen to go with a simpler type theory in which there is only one kind of type. Learning to keep track of the two kinds of variables in spatial type theory is hard enough. Moreover, the theory of Licata and Shulman (2016) is general enough to include this case as well: unlike $\Box$ and $\bigcirc$, our modalities $\mathcal{B}$ and $\sharp$ are type-theoretically sensible in their own right without needing to be deconstructed. There is also a philosophical attractiveness to the position that all types have topological structure, even if that structure happens to be discrete or codiscrete.

**Remark 1.4.** It is natural to wonder why we introduce crisp hypotheses $x : A$, when semantically they are essentially just ordinary hypotheses $u : \mathcal{B}A$. Why not simply restrict the formation of $\mathcal{B}B$ and $M^\mathcal{B}$ to contexts containing only variables of the form $u : \mathcal{B}A$?

This might be simpler in some respects, but it would be bizarre in others. The official type-theoretic reason is that, as noted by Pfenning and Davies (2001, Section 4.2), ‘substitution would no longer be an admissible rule.’ In less fancy language, that means that we would not be able to determine the validity of an expression by plain syntactic analysis, or else we would have to introduce a complicated calculus of explicit substitutions.

For instance, if $M^\mathcal{B}$ could be formed in a context of variables $u : \mathcal{B}A$, then we could define a function $f := (\lambda x. x^\mathcal{B}) : \mathcal{B}B \to \mathcal{B}\mathcal{B}B$, since a variable $x : \mathcal{B}B$ has the correct form to allow us to write $x^\mathcal{B} : \mathcal{B}\mathcal{B}B$. Now, suppose we also had a function $g : A \to \mathcal{B}B$. Then the composite $f \circ g$ would reduce to $\lambda a. g(a)^\mathcal{B}$; but $g(a)$ does not have the correct form to allow us to apply $(-)^\mathcal{B}$ to it! So if we allowed $g(a)^\mathcal{B}$ as a valid expression, we would be unable to determine its validity by inspection, but would have to ‘guess’ that it was obtained by substituting $g(a)$ into the valid $x^\mathcal{B}$. Otherwise, we would have to write instead $x^\mathcal{B}[g(a)/x]$, where the substitution $[g(a)/x]$ is not, as usual, a defined operation on expressions, but a basic part of the grammar of expressions (an ‘explicit substitution’).
The formalism of crisp variables avoids both horns of this dilemma. In this setup, \( f : \mathcal{B} \mathcal{B} \to \mathcal{B} \mathcal{B} \) must be defined as \( \lambda x. \text{let } u^\mathcal{B} := x \text{ in } u^{\mathcal{B}^{\mathcal{B}}} \) (see Section 4), and when we compose it with \( g \) we get \( \lambda a. \text{let } u^\mathcal{B} := \mathcal{G}(a) \text{ in } u^{\mathcal{B}^{\mathcal{B}}} \). Now we are no longer applying \((-)^\mathcal{B}\) directly to \( \mathcal{G}(a) \), but to a crisp variable that it has been destructed into; thus, the syntactic invariant that indicates a correct application of \((-)^\mathcal{B}\) is preserved.

With all that said, once the basic lemmas about \( \mathcal{B} \) are proven using crisp variables, in most cases, we will be able to blur the distinction between crisp elements of \( A \) and elements of \( \mathcal{B}A \), at least when working informally.

Now we have to choose how to represent \( \mathcal{S} \). One possibility would be to add further new judgement forms so that it could be characterized similarly to \( \mathcal{B} \) and \( \mathcal{S} \). The general theory of Licata and Shulman (2016) immediately suggests how to do this, but the judgmental structure of the resulting theory would be substantially more complicated. Moreover, semantically it seems likely to correspond to indexing the base topos over the cohesive topos by means of \( p^\mathcal{B} \), which is at least an odd thing to do. We would also need to take special care that the rules did not force \( \mathcal{S} \) to be left exact (whereas our rules for \( \mathcal{B} \) and \( \mathcal{S} \) do automatically make both of them left exact), since in our desired models, it is not.

For these reasons, I have chosen to introduce \( \mathcal{S} \) axiomatically. However, rather than asserting the modality itself as an axiom, we assert a set of ‘generators’ for it, enabling the modality itself to be constructed by a higher inductive localization. This has some technical advantages, and it also leads naturally into the stronger axiom of real-cohesion, where we can simply assert that \( \mathbb{R} \) alone is a sufficient generator for \( \mathcal{S} \). It seems highly unlikely that the special position of \( \mathbb{R} \) in real-cohesion could be obtained by rules rather than an axiom, so we lose nothing in that regard by making \( \mathcal{S} \) axiomatic already.

It should be clear from this discussion, however, that there is nothing sacred about the particular choices made in this paper. Most of the length of this paper is devoted to proving from our rules and axioms that \( \mathcal{B}, \mathcal{S} \) and \( \mathcal{S} \) satisfy the correct relationships (which Schreiber and Shulman (2012) just assumed to be true). Specifically, we need to know that \( \mathcal{S} \) and \( \mathcal{B} \) are reflectors (Theorems 3.6 and 9.14) and \( \mathcal{B} \) is a coreflector (Corollary 6.15), that \( \mathcal{S} \) is left exact (Theorem 3.9), that \( \mathcal{B} \vdash \mathcal{S} \vdash \mathcal{S} \) (Corollary 6.26 and Theorem 9.15), that \( \mathcal{S} \) preserves finite products and reflects into the same subcategory that \( \mathcal{B} \) coreflects into, that the images of \( \mathcal{B} \) and \( \mathcal{S} \) are equivalent (Corollary 6.23) by an equivalence that identifies \( \mathcal{B} \) with \( \mathcal{S} \) (Theorem 6.22), and that \( \mathcal{S} \) is generated by \( \mathbb{R} \) (Axiom \( \mathbb{R}^\mathcal{S} \)). Once these facts are established somehow, the contentful theorems (such as \( \mathcal{S}S^1 = S^1 \) and Brouwer’s fixed-point theorem) are essentially independent of the underlying type-theoretic machinery. I had to fix some particular choice of that machinery in order to write the paper, but this choice made may very well turn out not to be the optimal one.

Moreover, the behaviour of ‘spatial type theory’ as a type theory has not yet been studied. It looks somewhat reasonable, in that \( \mathcal{S} \) and \( \mathcal{B} \) have introduction and elimination rules of roughly the usual sort. But for it to be really respectable as a type theory, one should prove various standard theorems about it, such as normalization and canonicity, and this has not been done. (Licata and Shulman (2016) did this for a more general class of ‘adjoint logic’ type theories, but only under the fairly severe simplifying assumptions.
of no dependent types and only one variable in the context. Licata, Shulman, and Riley (2017) allows multi-variable contexts, but still no dependent types, and has a rather fiddlier adequacy theorem.) Eventually, one might hope to implement such a type theory in a proof assistant.

1.5. Outline of the paper

This paper has three parts; but rather than coming in sequential order, one of the parts is interleaved through the other two. The first part, consisting of Sections 2–7, develops spatial type theory. It begins with the judgemental structure (Section 2), then moves on to ♯ (Section 3) and then ♭ (Sections 4–6). In Section 7, we sketch briefly the intended categorical models of this theory, and use them to motivate the axioms to be introduced in the third part.

The second, interleaved, part, consisting of Sections 2.1, 3.1, 3.2, 6.4 and 8.3, studies classical axioms that can be added to spatial type theory without destroying its topological content, such as the modified law of excluded middle mentioned above. It is interleaved because the formulation and proof of various classical principles requires more and more of the base type theory, but at the same time provides useful motivation for the same. In Section 2.1, we state the restricted law of excluded middle, and decide that in order to state a similarly restricted axiom of choice we need ♯. After introducing ♯, in Section 3.1, we revisit excluded middle in some other guises made possible by ♯, and in Section 3.2, we formulate the axiom of choice, but note that there ought to be simpler versions available if we had ♭. Thus, after introducing ♭, in Section 6.4, we discuss these versions and their applications. Finally, after stating the axioms of real-cohesion, in Section 8.3, we show that these axioms strengthen our classicality axioms, and state one further such axiom.

The third part, consisting of Sections 8–12, adds additional axioms to spatial type theory that make it cohesive and real-cohesive, and applies them to prove Brouwer’s fixed-point theorem. In Section 8, we state the cohesion and real-cohesion axioms in generator form and deduce some consequences, and in Section 9, we construct 𝐽 as a higher inductive type. Section 10 relates our axioms back to the motivating topos-theoretic notions. In Section 11, we note some implications of real-cohesion for synthetic topology that do not require any higher homotopy, including some versions of the Intermediate Value Theorem and a version of ‘Brouwer’s theorem’ that all functions \( \mathbb{R} \to \mathbb{R} \) are continuous. Finally, in Section 12, we bring everything together to prove Brouwer’s fixed-point theorem by mimicking the classical proof, invoking synthetic homotopy theory and our modified classicality axioms in the appropriate places. We also similarly prove a constructive variant asserting only the existence of approximate fixed points, but not requiring any classicality axioms.

1.6. On axioms and notation

Because we are developing a hierarchy of type theories (spatial < cohesive < real cohesive), and using classical axioms in some places, we keep careful track of which axioms are necessary for which theorems. In addition to our classicality axioms (LEM, AC and T) and
(real-)cohesion axioms \((C_0, C_1, C_2\text{ and } R^{\flat})\), we track uses of the full \(UA\). We do need \(UA\) for our proof of Brouwer's fixed-point theorem, but most of the theory does not require it, and hence should be valid in a 1-topos model in addition to an \((\infty, 1)\)-topos model. We will, however, use without comment the axioms of function extensionality, propositional resizing and univalence restricted to propositions ('propositional extensionality'), since these are all valid in a 1-topos.

We will generally adhere to the notation of Univalent Foundations Program (2013). For example, we write \(x = y\) for the identity type, \(M \equiv N\) for a judgemental equality, and \(a \equiv P\) if \(a\) is currently being defined to equal \(P\). However, as remarked above, we will call elements of \(x = y\) ‘identifications’ or ‘equalities’ rather than ‘paths,’ and we write \(S^1\) for the higher inductive circle to reserve \(S^1\) for the topological one. We will also call \((-1)\)-truncated types simply propositions rather than ‘mere propositions,’ as is common outside of Univalent Foundations Program (2013). Similarly, we pronounce the \((-1)\)-truncated existential (as in Univalent Foundations Program (2013), \(\exists_{x:A} P(x)\) means \(\sum_{(x:A)} P(x)\)) simply as ‘there exists,’ rather than the ‘there merely exists’ of Univalent Foundations Program (2013). This also requires avoiding other terminology such as ‘propositional equality’ and ‘propositional uniqueness rule,’ since the notions in question are not \((-1)\)-truncated; for this purpose, we introduce the new adjective typal (to contrast with judgemental).

1.7. Vistas

Brouwer's fixed-point theorem is not itself of central importance; it serves mainly as a convenient test case. The real point is the development of spatial, cohesive and real-cohesive type theory. Some possibilities for further applications include the following.

— Real-cohesion should also imply other applications of homotopy theory to topology, such as the Lefschetz fixed-point theorem and the hairy ball theorem.

— In cohesive homotopy type theory, a topological group appears as an ordinary internal (0-truncated) group, which can be delooped as in Licata and Finster (2014). These deloopings are classifying spaces for topological principal bundles up to isomorphism (rather than the up-to-homotopy-equivalence classifying spaces of classical algebraic topology); the classifying maps live in ‘continuous cohomology theories.’

— Matrix groups and their deloopings are important in classical algebraic topology, but as \(\infty\)-groupoids, they are hard to define in ordinary homotopy type theory. In real-cohesive homotopy type theory, we can obtain them as shapes of ordinary set-level definitions. The same is true for other classical objects such as higher Hopf fibrations.

— There are other local toposes than cohesive and real-cohesive ones, such as Sierpinski cones, the topological topos of Johnstone (1979), and the relative realizability topos of Awodey, Birkedal, and Scott (1999). Each of these should motivate different axioms.

— Similarly, there are cohesive toposes other than real-cohesive ones. Some, which play an important role in Schreiber (2013), encode smoothness rather than continuity; these satisfy a version of Axiom \(R^{\flat}\) with \(\mathbb{R}\) replaced by a type of ‘smooth reals.’ Other
interesting examples include the global homotopy theory of Rezk (2014), the ∞-topos of simplicial ∞-groupoids and the tangent ∞-toposes of Lurie (2014).

1.8. On theft and honest toil

One of the advantages of type theory is that it enables us to ‘add structure by failing to rule it out.’ For instance, because type theory admits topological models, if we define (say) a group in type theory, as long as we did not use any principles that are invalid topologically, we have automatically defined a topological group as well. Homotopy type theory applies the same effect to ∞-groupoids: as long as we do not use any principles like UIP, everything we do in type theory is automatically homotopical. This is especially valuable because working directly with ∞-groupoids can be combinatorially complicated and require a lot of background in algebraic topology and category theory.

Spatial and cohesive type theory extend this advantage even further: not only does type theory apply automatically to spaces and to ∞-groupoids, it also applies to spatial ∞-groupoids, which have both structures at once. Moreover, even if we did not have any categorical models in mind to specify what a ‘spatial ∞-groupoid’ might mean, we could still undertake to study them in type theory by simply combining the axioms and principles that pertain to spatial models (such as continuity principles and/or modalities) and homotopical models (such as univalence and HITs). Except for Sections 7 and 10, the present paper can be read from such a perspective.

It is true that mathematical honesty may demand that such models eventually be produced, to ensure relative consistency and allow our ‘synthetic’ results to be translated into classical theorems. However, once someone has undertaken that toil (in our case, the toil includes Gepner and Kock 2012; Kapulkin and Lumsdaine 2012; Lumsdaine and Shulman 2017; Lumsdaine and Warren 2015; Lurie 2009; Schreiber 2013; Shulman 2011a,b, as well as future work that remains to be done), we can continue to work in the type theory without needing to think about or understand the model. This principle should also apply to other possible enhancements of spatial type theory.

2. Judgements of spatial type theory

As explained in the introduction, we take the point of view that all types have spatial structure, independently of their ∞-groupoid structure, and concomitantly all ordinary type-theoretic constructions are continuous. Now, it does happen sometimes in mathematics that we want to talk about things that are not continuous. Our main target in this paper—Brouwer’s fixed-point theorem—is a prime example: fixed points of functions cannot be selected continuously with respect to the space of functions. For this reason, we augment our type theory with technology that enables us to talk about possibly discontinuous constructions as well as continuous ones.

¶

We do have to be careful that this happens in the right way. For instance, the naïve definition of ‘group’ does not give the correct ∞-groupoidal notion, but there is a different one that does.
Recall that the basic judgement of type theory is \( a : A \) for some specific type \( A \), expressing that \( a \) is a point of the type \( A \). This judgement is used in two ways: for some specific expression \( a \), we can judge it to be true, or for some variable \( x \), we can \emph{suppose} it to be true. For instance, in a statement like 'whenever \( x : A \) then \( f(x) : B \)' we see both uses: \( x \) is a variable hypothesized to be of type \( A \), and under this hypothesis, \( f(x) \) is an expression judged to have type \( B \). It is a distinguishing feature of type theory that whenever \( x : A \) is a hypothesis, \( x \) must be a variable; it makes no sense to say 'suppose that \( f(x) : B \)' (in contrast to the behaviour of the membership predicate \( \in \) of set theory).

In a world where types are spaces, all ordinary type-theoretic constructions are continuous; thus, a statement like 'whenever \( x : A \) then \( f(x) : B \)' means that \( f(x) \) depends \emph{continuously} on \( x \). We now augment our type theory with new features to express the possibility that this dependence may \emph{not} be continuous. In general, we might imagine doing this either by modifying the hypothesis \( x : A \) or the conclusion \( f(x) : B \). Since continuous functions are also discontinuous, the variance of implication means that if we modify \( x : A \), we would have to replace it by a \emph{stronger} judgement (i.e. one that implies \( x : A \)), whereas if we modify the conclusion \( f(x) : B \), we would have to replace it by a \emph{weaker} judgement (i.e. one that is implied by \( f(x) : B \)).

We take the first route. Following Pfenning and Davies (2001), we denote the stronger judgement by

\[
x :: A
\]

When we hypothesize \( x :: A \), this means that we allow ourselves to perform constructions and proofs using \( x \) that may not respect the topology of \( A \) (but they still must respect the topology of any other ordinary hypotheses \( y : C \)). In other words, \emph{having} \( x :: A \) is \emph{more} powerful than \emph{having} \( x : A \); and thus, oppositely, \emph{proving} something with a hypothesis of \( x :: A \) says \emph{less} than proving it with a hypothesis \( x : A \).

We refer to \( x :: A \) as a \emph{crisp} hypothesis, and \( x \) as a crisp variable. (Pfenning and Davies (2001), whose focus is on logic, use the word \emph{valid}.) A crisp variable can always be used as an ordinary one, but also (potentially) in other ways. If we do not want to give a name to the variable, we may say that \( A \) holds \emph{crisply}. If necessary, for contrast, we will refer to an ordinary hypothesis or variable \( x : A \) as \emph{cohesive}.

For consistency, we require that all variables appearing in the type of a crisp hypothesis are themselves crisp. Moreover, a crisp variable can only be substituted by expressions involving only other crisp variables. If \( M : A \) is an expression other than a variable, we will say that \( M \) is crisp, and sometimes write \( M :: A \), if the only \emph{variables} it contains are crisp. Thus, as for Pfenning and Davies (2001), the notion of \emph{crisp conclusion} is not

\footnote{
We will use 'discontinuous' to mean 'not necessarily continuous,' by analogy with other phrases such as 'non-commutative ring.'
}

\footnote{
The word 'crisp' is chosen as having a connotation somewhat similar to 'discrete' (i.e. not flabby or deformable) and being relatively free of other mathematical meanings. Its one existing mathematical usage that I am aware of is in fuzzy-set theory, where it refers to ordinary (non-fuzzy) sets; this is actually closely related to our usage, since many categories of fuzzy sets are quasitoposes (Wyler 1991, Chapter 8) in which the crisp sets are what we would call the codiscrete objects, so that they also 'forget about the cohesion.'
}
a basic part of the theory, but rather is defined to mean an ordinary conclusion that depends only upon crisp hypotheses.

All the ordinary rules of type theory (\(\prod\)-types, \(\Sigma\)-types, \(=\)-types, \(W\)-types, HITs) are imported into our theory only in the world of cohesive variables. Applications of these rules can involve dependence on ‘unaffected’ crisp variables; but the variables that are manipulated in the rules, as well as their conclusions, are always cohesive. (We will eventually prove, however, that in some cases this can be worked around.)

At a formal level, what we are doing is separating our context into two pieces: first a crisp one, then a cohesive one, maintaining the restriction that the type of each variable can only contain variables occurring to its left. We would therefore write our judgements as \(\Delta \vdash \Gamma \vdash J\). The previous paragraph means that the ordinary rules of type theory leave the crisp context \(\Delta\) untouched; for example, the rules for \(\prod\)-types are shown in Figure 2. In what follows, we will attempt to describe all the rules of spatial type theory in words, but we will also include some formal syntax of this sort for precision. After introducing the rules, however, we will reason almost entirely informally in the style of Univalent Foundations Program (2013).

We will have occasion to use various axioms (including axioms from Univalent Foundations Program (2013) such as function extensionality and univalence), so it is worth noting how these interact with our split contexts. Ordinarily in type theory, an axiom means an assumed element of some type. There are two ways of thinking about such axioms in type theory: they can be premise-free rules added to the theory, or they can be simply additional assumptions in the context that are never discharged. According to the first approach, an axiomatic \(a : A\) in our theory would be given by the rule

\[
\Delta \vdash \Gamma \vdash a : A
\]

Note that because this rule is valid even if \(\Gamma\) is empty, it implies unavoidably that the term \(a : A\) is crisp. The second approach is more flexible: we can add assumptions to either the crisp context \(\Delta\) or the cohesive one \(\Gamma\). However, in this paper, whenever we speak of an axiom, it is to be understood that it is crisp.

We will also consider some ‘axioms’ that are not stated as assumed elements of a fixed type, e.g. because they quantify over crisp variables. These can, however, always be formulated as ‘unjustified rules’; an example will occur momentarily. Moreover, once we

\[
\frac{\Delta \vdash \Gamma \vdash B : Type}{\Delta \vdash \Gamma \vdash \prod_{x : A} B : Type}
\]

\[
\frac{\Delta \vdash \Gamma \vdash b : B}{\Delta \vdash \Gamma \vdash \lambda x. b : \prod_{x : A} B}
\]

\[
\frac{\Delta \vdash \Gamma \vdash f : \prod_{x : A} B}{\Delta \vdash \Gamma \vdash f \equiv (\lambda x. f(x))}
\]

\[
\frac{\Delta \vdash \Gamma \vdash a : A}{\Delta \vdash \Gamma \vdash \lambda x. b(a) \equiv b[a/x]}
\]

\[
\frac{\Delta \vdash \Gamma \vdash \prod_{x : A} B}{\Delta \vdash \Gamma \vdash \prod_{x : A} B}
\]
introduce $\mathfrak{b}$, all such axioms can be reformulated in the usual way (modulo potential issues with universes, which we ignore).

2.1. The law of excluded middle

With the tool of crisp hypotheses in hand, we can express a version of the law of excluded middle that does not contradict topological models.

**Axiom LEM.** (Crisp excluded middle) For any crisp $P :: \text{Prop}$, we have $P \lor \neg P$.

**Remark 2.1.** In ordinary mathematics, we blur the distinction between a *hypothetical conclusion* (or entailment) and an *implication* (or function). That is, when we write ‘if $x : A$ then $f(x) : B(x)$’, we might mean that we have constructed $f(x) : B(x)$ under the hypothesis of $x : A$, or we might mean that we have an element $f : \prod_{x : A} B(x)$. These two statements are intimately connected by the introduction and elimination rules for $\prod$, so no confusion results from failing to distinguish between them.

However, when the hypothesis is crisp, we have as yet no type-former corresponding to $\prod$: we cannot abstract over a crisp variable. Thus, statements such as crisp excluded middle must be read as hypothetical conclusions. In type-theoretic language, we have to read Axiom LEM as an ‘unjustified rule’:

$$
\Delta \mid \cdot \vdash P : \text{Prop} \\
\Delta \mid \cdot \vdash \text{lem}_P : P \lor \neg P
$$

rather than an assumed element of some fixed type. (We write ‘$\Delta \mid \cdot$’ to indicate that the cohesive context is empty.)

Fortunately, in Sections 3 and 4, we will introduce new type formers that will allow us to mostly go back to ignoring this distinction. See Remark 1.4 for an explanation of why we do this in two steps.

To understand why this law of excluded middle is sensible, we need to think a bit about what a *predicate* $P : A \to \text{Prop}$ looks like topologically. In the classical category of topological spaces, the monomorphisms are the continuous injections; they need *not* necessarily be subspace inclusions. Therefore, given a subobject (i.e. a mono) $m : [P] \hookrightarrow A$, there is a difference between saying that every *point* of $A$ is in $[P]$ (which is to say that the injection $m$ is actually a bijection) and saying that $m$ is a homeomorphism. We regard the judgements $P(x)$ for all $x :: A'$ and $P(x)$ for all $x : A'$ as expressing this same dichotomy in our type theory. The latter says that $P(x)$ holds for all $x$ continuously as a function of $x$, so that our monomorphism $m$ must have a section, and hence be a homeomorphism.

Now, if $P : A \to \text{Prop}$, its pointwise negation $\neg P : A \to \text{Prop}$ is, essentially by definition, the *largest subobject of $A$ disjoint from $P$*. Topologically, this means the corresponding mono $[\neg P] \hookrightarrow A$ should contain exactly those points of $A$ that are not in $[P]$, but also that $[\neg P] \hookrightarrow A$ *must be a subspace inclusion*—for the inclusion of the subspace determined by those points is certainly disjoint from $[P]$ and hence must be contained in $[\neg P]$ as a subobject of $A$. 

It should be clear now that the ordinary LEM cannot hold, for it would assert that for any mono \( m : [P] \rightarrow A \), the space \( A \) is the disjoint union of the space \([P]\) and the subspace determined by its complement. This is not true in general even if \([P]\) is a subspace inclusion. However, since \([P]\) and \([-P]\) together do contain all the points of \( A \), it is sensible to assert that for any \( x : A \), we have \( P(x) \lor \neg P(x) \) crisply, and this is what the crisp law of excluded middle gives us.

**Remark 2.2.** It is natural to wonder how we can express internally in type theory the property that a subobject is a subspace inclusion. The above analysis suggests that the double negation \( \neg\neg P : A \rightarrow \text{Prop} \) should be the subspace containing the same points as \( P \), so that \( P \) would itself be a subspace if and only if \( \prod_{(x : A)} \neg\neg P(x) \rightarrow P(x) \). In Section 3, we will introduce another way to express this property, and in Section 3.1, we will prove (using our axioms) that they are equivalent.

**Remark 2.3.** The assertion that every subspace has a complementary subspace (such that their disjoint union is the whole space) translates to \( \prod_{(P : \text{Prop})} \neg P \lor \neg \neg P \). Thus, the latter assertion, which is sometimes called de Morgan’s law (since it is equivalent to the one direction of de Morgan’s laws that is not constructively valid, \( \neg(P \land Q) \rightarrow (\neg P \lor \neg Q) \)), is not acceptable for us either.

We can try to perform a similar analysis of the axiom of choice. There is one subtlety: we must realize that the ‘surjections’ of type theory correspond topologically to regular epimorphisms—which is to say, quotient maps—rather than arbitrary epimorphisms. One way to see that this must be so is to recall that any surjection between sets in type theory can be proven to be a quotient of its domain by some equivalence relation, which is true of regular epis of spaces but not ordinary ones. Another way is to recall that in type theory, we have a (surjection, embedding) factorization system, and topologically the left class corresponding to the monomorphisms is the quotient maps.

Now supposing that \( Y \rightarrow X \) is a quotient map, of course, there may not be any continuous section of it. However, if the only source of non-classicality is topology, it should have a discontinuous section. Thus, we may hope to be able to formulate this principle in our type theory. However, if we inspect the usual formulation of the axiom of choice from Univalent Foundations Program (2013, Theorem 3.2.2):

\[
\left( \prod_{x : X} \| Y(x) \| \right) \rightarrow \| \prod_{x : X} Y(x) \|
\]

it is not clear how to do this with our current tools. What we want is for the section \( f : \prod_{(x : X)} Y(x) \) on the right to be discontinuous. But at present, we can only indicate discontinuity by a conclusion \( f(x) : Y(x) \) with a crisp hypothesis \( x :: X \), and we cannot apply a \( \| - \| \) outside such a judgement. (The outer \( \| - \| \) is, of course, necessary for consistency with univalence.)

In the next section, therefore, we introduce a new type former \( \# \) that ‘internalizes’ such judgements, allowing us to solve this problem (and many others).
3. The ♯ modality and codiscreteness

Topologically, the function-type $\prod_{(x:X)} Y(x)$ represents the space of continuous functions; but as we saw in the last section, we sometimes want to internalize the notion of a discontinuous function. Since every function out of a discrete topological space is continuous, and every function into a codiscrete space is continuous, to make a discontinuous function continuous all we have to do is retopologize its domain discretely or its codomain codiscretely. We represent these 'retopologizing' constructions by modalities called $♭$ and $♯$, which 'reify' judgements involving crisp hypotheses.

In this section, we study ♯ first, as it is somewhat simpler and stands on its own better. Described in English, the rules are as follows; a type-theoretic presentation is shown in Figure 3.

— For any $A : \text{Type}$ there is a type $♯A$.
— If $a : A$, then $a♯ : ♯A$.
— All crisp variables appearing in $a$ can become cohesive ones in $a♯$. In other words, when we write $(-)♯$, inside the $(-)$ we are free to use all variables introduced outside of it as if they were crisp. In yet other words, when we 'hit a $(-)♯$ while parsing an expression, all variables currently in the context become crisp. This expresses the fact that all functions into a codiscrete space are continuous.
— In the same way, all crisp variables appearing in a type $A$ can become cohesive ones in $♯A$. This expresses the fact that the space of codiscrete types is itself codiscrete (which may not be obvious, but is true; we will explain it topologically in Remark 3.14).
— From any crisp element $x :: ♯A$, we can extract an element $x♯ : A$. Of course, since $x$ is crisp, $x♯$ is also crisp.
— A computation rule, which says $(a♯)♯ ≡ a$. This requires $(a♯)♯$ to be well-typed, which means that only crisp variables in the ambient context can occur in $a$. (The above rule says that when writing $a♯$, cohesive variables in the ambient context can be used as crisp inside $a$; but such variables remain cohesive in the term $a♯$, potentially preventing us from writing $(a♯)♯$.)
— A uniqueness rule, which says $(a♯)♯ ≡ a$. This likewise requires both sides to be well-typed, but now the potential problem lies with the right-hand side.†‡ In writing $(a♯)♯$, the cohesive variables in the context might be treated as crisp inside $a♯$ and hence inside $a$, so that $a$ itself might not be well-typed outside of $(-)♯$.

Remark 3.1. ♯ is a negative type former. This means, roughly, that its introduction rule(s) are chosen to match its elimination rule(s) rather than vice versa. For instance, function types are negative because to introduce a function, we use $\lambda$-abstraction, which essentially means we have to say what will happen when we apply the eliminator (application) on any input. In the case of ♯, the elimination rule says that we can get to $A$ if we have a crisp element of $♯A$; thus, to introduce an element of $♯A$, it suffices to give an element of $A$ under the assumption that everything is crisp.

†‡ Note that if we view this rule as an 'expansion' $a \rightarrow_η (a♯)♯$ rather than a reduction, as is common for uniqueness rules, then it shares with the computation rule the property that it can be applied to any well-typed input $a : ♯A$. 
Negative type formers tend to require (judgemental) uniqueness rules (whereas for positive ones, a typal uniqueness rule\(^8\) is usually provable). This holds true for ♯.

As our first application, we can make ♯ into a functor: given \(f : A \to B\), we define \(♯f : ♯A \to ♯B\) by

\[
♯f(u) :≡ f(u♯)♯.
\]

Note how the type-checking works: inside the \((-)♯\), the variable \(u :: ♯A\) is crisp, so we can write \(u♯ : A\) and apply \(f\) to it. We can check functoriality: if also \(g : B \to C\), then

\[
♯g(♯f(u)) ≡ g((f(u♯))♯)
\]

\[
≡ g(f(u♯))♯
\]

\[
≡ ♯(gf)(u).
\]

In addition, for any \(A : Type\), we have a map \((-)♯ : A → ♯A\), and we can check that this is natural:

\[
♯f(x♯) ≡ f((x♯)♯)
\]

\[
≡ f(x)♯.
\]

We will often omit parentheses around iterated sub- and superscripted ♯s (and, later, ⬆️s), maintaining only their order. Thus, for instance, \(g((f(u♯))♯)♯\) becomes instead \(g(f(u♯))♯♯\), and \(f((x♯))♯\) becomes \(f(x♯)♯\). This avoids proliferation of parentheses.

Now, recall that our intent was that ♯A would be \(A\) retopologized codiscretely. Thus, a type should be called codiscrete just when this operation does nothing to it.

**Definition 3.2.** A type \(A : Type\) is codiscrete if \((-)♯ : A → ♯A\) is an equivalence.

The rules for ♯ transfer over to theorems about codiscrete types. For instance, we have:

**Theorem 3.3.** If \(P\) is codiscrete, then when constructing an element of \(P\) we may assume that all variables in the context are crisp.

**Proof.** Since \(P\) is codiscrete, we have an inverse \(r : ♯P → P\). Thus, it suffices to construct an element of ♯\(P\). But now we can apply \((-)♯\), which allows us to assume all variables are crisp.

\(^8\) Recall that we say ‘typal’ to mean that a rule is witnessed by an inhabitant of the identity type rather than being a judgemental equality, avoiding the older term ‘propositional’ since it may not be a ‘proposition’ in the \((-1)\)-truncated sense.
**Theorem 3.4.** If \( P : \# A \to \text{Type} \) is such that each \( P(v) \) is codiscrete, and we have \( f : \prod_{(x : A)} P(x^\#) \), then we have \( g : \prod_{(v : \# A)} P(v) \) such that \( g(x^\#) = f(x) \) for all \( x : A \). We refer to this principle as \( \# \)-induction.

**Proof.** Since each \( P(v) \) is codiscrete, we have inverses \( r_v : \# P(v) \to P(v) \). Thus, to construct \( g \), it will suffice to construct for each \( v : \# A \) an element of \( \# P(v) \), and for this it suffices to construct an element of \( P(v) \) assuming a crisp \( v : \# A \). Now, of course, we have \( v^\# : A \), and hence \( f(v^\#) : P(v^\#) \equiv P(v) \), as desired. In symbols,

\[
g(v) \equiv r_v(f(v^\#)).
\]

Finally, we have

\[
g(x^\#) \equiv r_v(f(x^\#)) \equiv r_v(f(x)) = f(x)
\]

using the computation rule for \( \# \) and the fact that \( r_v \) is an inverse of \((-)^\#\).

We now show that the codiscrete types form a *reflective subuniverse* in the sense of Univalent Foundations Program (2013, Section 7.7).

**Theorem 3.5.** For any \( A : \text{Type} \), the type \( \# A \) is codiscrete.

**Proof.** Given \( z : \#\# A \), to define an element of \( \# A \), it suffices to define an element of \( A \) assuming \( z \) is crisp. But then we have \( z^\# : A \). In other words,

\[
(\lambda z. z^\#) : \#\# A \to \# A.
\]

Note that we cannot reduce \( z^{\#\#} \) to \( z^\# \) using the uniqueness rule, since \( z^\# \) is not well-typed for an arbitrary (cohesive) \( z \).

We now show that this map is inverse to \((-)^\# : \# A \to \#\# A \). In one direction, if \( z \) is \( y^\# \) for \( y : \# A \), we have

\[
y^{\#\#} \equiv y^\# \equiv y
\]

using the computation rule followed by the uniqueness rule. Note that we could not do this computation in the other order, because \( y^{\#\#} \) is not well-typed on its own: \( y \) is not crisp, so neither is \( y^\# \). In the other direction, for any \( z : \#\# A \), we have

\[
z^{\#\#} \equiv z^\# \equiv z
\]

using the uniqueness rule twice; the first use is valid because \( z^\# \) is well-typed inside the outer \((-)^\#\).

**Theorem 3.6.** If \( B : \text{Type} \) is codiscrete, then for any \( A : \text{Type} \), precomposition with \((-)^\# : A \to \# A \) is an equivalence

\[
(\# A \to B) \simeq (A \to B).
\]

More generally, if \( B : \# A \to \text{Type} \) has each \( B(v) \) codiscrete, then precomposition with \((-)^\# \) is an equivalence

\[
\left( \prod_{(v : \# A)} B(v) \right) \simeq \left( \prod_{(x : A)} B(x^\#) \right).
\]
Proof. We prove the more general statement. Since each $B(v)$ is codiscrete, postcomposition with $(-)^\flat : B(v) \to \sharp B(v)$ is an equivalence. Thus, by the 2-out-of-3 property of equivalences, it will suffice to show that the precomposition map

$$\left(\prod_{(v : \sharp A)} \sharp B(v)\right) \to \left(\prod_{(x : A)} \sharp B(x^\sharp)\right)$$

is an equivalence. Now in the opposite direction, given $f : \prod_{(x : A)} \sharp B(x^\sharp)$, we can construct $g : \prod_{(v : \sharp A)} \sharp B(v)$ by

$$g(v) := f(v)^\sharp.$$

Note that this has the form of the left-hand side of the uniqueness rule, but that rule does not apply: ‘$f(v^\sharp)$’ would not be well-typed since $v$ is not a crisp variable. On the other hand, it has the correct type $\sharp B(v)$ because the uniqueness rule says $v^\sharp^\sharp \equiv v$ since $v$ is crisp inside the $(-)^\sharp$, hence $f(v^\sharp) : \sharp B(v^\sharp^\sharp) \equiv \sharp B(v)$.

If we precompose $g$ with $(-)^\sharp : A \to \sharp A$, we get

$$g(y^\sharp) \equiv f(y^\sharp)^\sharp \equiv f(y)^\sharp \equiv f(y),$$

where now we have been able to apply the uniqueness rule since $f(y)$ is well-typed on its own. On the other hand, if we start with $h : \sharp A \to \sharp B$, precompose it with $(-)^\sharp$ and then extend the result back to $\sharp A$ as above, we get the function sending $v : \sharp A$ to

$$h(v^\sharp)^\sharp \equiv h(v)^\sharp \equiv h(v),$$

using the uniqueness rule twice. Applying function extensionality on both sides completes the proof.

Combining this with Theorem 3.4, by Univalent Foundations Program (2013, Theorem 7.7.4), we see that the codiscrete types are actually a modality in the sense of Univalent Foundations Program (2013, Section 7.7). However, in the present paper, we are using the word ‘modality’ more generally to include coreflectors such as $\flat$ in addition to reflectors such as $\sharp$, so we will refer to modalities in the sense of Univalent Foundations Program (2013, Section 7.7) as monadic modalities.

The fact that $\sharp$ is a monadic modality formally implies many useful consequences. In particular, essentially all properties of the n-truncation from Univalent Foundations Program (2013, Chapter 7) that do not refer to more than one value of $n$ are true for all monadic modalities, and hence in particular for $\sharp$. Monadic modalities will be studied further by Rijke, Shulman, and Spitters (2017); here, we list some of the main results, specialized to $\sharp$.

1. A type $A$ is codiscrete if and only if $(-)^\sharp : A \to \sharp A$ admits a retraction.
2. If $A, B, C$ are codiscrete, and we have $x, y : A$ and functions $f : B \to A$ and $g : C \to A$,

then the following types are also codiscrete:

$$1 \quad A \times B \quad x =_A y \quad \text{fib}_f(x) \quad B \times A \quad C \quad A \simeq B \quad \text{is-n-type}(A).$$

3. If $A$ is any type and $P : A \to \text{Type}$ is such that each $P(x)$ is codiscrete, then $\prod_{(x : A)} P(x)$ is codiscrete. If, in addition, $A$ is codiscrete, then $\sum_{(x : A)} P(x)$ is also codiscrete.
4. For any $A, B$, the canonical map $\sharp(A \times B) \to \sharp A \times \sharp B$ is an equivalence.
5. If $A$ is a proposition, so is $\#A$.

In fact, $\#$ is even a left exact monadic modality, i.e. it preserves pullbacks. To show this, we begin with an ‘encode–decode’ characterization of the identity types of $\#A$.

**Theorem 3.7.** For any $x, y : A$, we have an equivalence $(x^\sharp = y^\sharp) \simeq \#(x = y)$ such that the following triangle commutes:

$$
\begin{array}{ccc}
(x = y) & \xrightarrow{\text{ap}_{(-)\sharp}} & (x^\sharp = y^\sharp) \\
\downarrow & \simeq & \downarrow \\
\#(x = y) & \xrightarrow{(-)^\sharp} & \#
\end{array}
$$

(1)

**Proof.** We define $\text{code} : \#A \to \#A \to \text{Type}$ by

$$
\text{code}(u, v) \equiv \#(u^\sharp = v^\sharp).
$$

Here, we use the fact that variables may be assumed crisp inside $\#(-)$ as well as inside $(-)^\sharp$. Note that for $x, y : A$, we have $\text{code}(x^\sharp, y^\sharp) \equiv \#(x = y)$.

We also have $r : \prod_{(u \equiv v)} \text{code}(u, u)$. For by Theorem 3.5, $\text{code}(u, u)$ is codiscrete; thus, by $\#$-induction the goal reduces to $\prod_{(x : A)} \#(x = x)$, which is inhabited by $\lambda x. \text{refl}_x^\sharp$.

Now, we define

$$
\text{encode} : \prod_{u, v : \#A} (u = v) \to \text{code}(u, v)
$$
in the usual way, $\text{encode}(u, v, p) \equiv p_*(r(u))$. To define

$$
\text{decode} : \prod_{u, v : \#A} \text{code}(u, v) \to (u = v),
$$
we proceed as follows. Since $\#A$ is codiscrete, so is $u = v$, and hence so are $\text{code}(u, v) \to (u = v)$ and $\prod_{(u \equiv v)} \text{code}(u, v) \to (u = v)$. Thus, we can apply $\#$-induction twice, reducing the goal to $\prod_{(x, y : A)} \text{code}(x^\sharp, y^\sharp) \to (x^\sharp = y^\sharp)$. But $\text{code}(x^\sharp, y^\sharp) \equiv \#(x = y)$, so we can use $\#$-induction again followed by $\text{ap}_{(-)^\sharp}$.

It now suffices to show that $\text{encode} \circ \text{decode}$ is the identity, since for each $u$ that will exhibit $\sum_{(v : \#A)} \text{code}(u, v)$ as a retract of the contractible $\sum_{(v : \#A)} (u = v)$, hence itself contractible. Now the goal $\text{encode}(u, v, \text{decode}(u, v, c)) = c$ is an equality in $\text{code}(u, v)$, which is codiscrete by Theorem 3.5. Thus, the goal is also codiscrete, so we can do $\#$-induction again, replacing $u$ and $v$ by $x^\sharp$ and $y^\sharp$, respectively, and then $c : \#(x = y)$ by $p^\sharp$ for $p : x = y$, so that the goal becomes $(\text{ap}_{(-)^\sharp}(p))_*(\text{refl}_x^\sharp) = p^\sharp$. Finally, an Id-induction on $p$ completes the proof of equivalence.

To see that Equation (1) commutes, we do Id-induction on $p$ again, then compute

$$
\text{encode}(x^\sharp, x^\sharp, \text{ap}_{x^\sharp}(\text{refl}_x^\sharp)) \equiv \text{encode}(x^\sharp, x^\sharp, \text{refl}_x^\sharp) \equiv (\text{refl}_x^\sharp)_*(\text{refl}_x^\sharp) \equiv \text{refl}_x^\sharp.
$$

We remark in passing that $\#$ is ‘functorial on homotopies’ as well. Given $f, g : A \to B$ and $H : f \sim g$, i.e. $H : \prod_{(x : A)} f(x) = g(x)$, for any $u : \#A$, we have $H(u^\sharp) : \#(f(u^\sharp) = g(u^\sharp))$, which by Theorem 3.7 can be decoded to give $f(u^\sharp) = g(u^\sharp)$.
M. Shulman

878

i.e. \(\sharp f(u) = \sharp g(u)\). We could also go on to construct higher dimensional aspects of a ‘coherent \(\infty\)-functor’ structure on \(\sharp\).

Returning to left-exactness of \(\sharp\), this is actually implied formally by the conclusion of Theorem 3.7 (see HoTT Project 2015; Rijke, Shulman, and Spitters 2017). However, using the special properties of \(\sharp\) we can give a more direct proof, beginning with a sense in which \(\sharp\) ‘preserves \(\sum\)'s.’

**Lemma 3.8.** For any \(A : \text{Type}\) and \(B : A \to \text{Type}\), we have

\[
\sharp \left( \sum_{x : A} B(x) \right) \simeq \sum_{u : \sharp A} \sharp B(u_{\sharp}).
\]

(The right-hand side makes sense because inside \(\sharp(\_\_)\) we may use \(u\) as crisp.)

**Proof.** The right-hand side is codiscrete, so we can construct a map from left to right by \(\sharp\)-induction, sending \((a, b)\) to \((a_{\sharp}, b_{\sharp})\). (Note that if \(u\) is \(a_{\sharp}\), then \(\sharp B(a_{\sharp}) \equiv \sharp B(a)\), so this is well-typed.) In the other direction, it suffices to construct an element of \(\sum_{(x : A)} B(x)\) supposing crisp hypotheses \(u :: \sharp A\) and \(v :: \sharp B(u_{\sharp})\); but then we have \((u_{\sharp}, v_{\sharp}) : \sum_{(x : A)} B(x)\). Finally, since both sides are codiscrete, both round-trip composites can be compared to identities with further \(\sharp\)-inductions.

**Theorem 3.9.** \(\sharp\) preserves pullbacks.

**Proof.** Given \(f : A \to C\) and \(g : B \to C\), recall that the pullback \(A \times_C B\) is \(\sum_{(x : A)} \sum_{(y : B)} (f(x) = g(y))\). By Lemma 3.8, we have

\[
\sharp(A \times_C B) \simeq \sum_{(u : \sharp A)} \sum_{(v : \sharp B)} \sharp(f(u_{\sharp}) = g(v_{\sharp})),
\]

while

\[
\sharp A \times_{\sharp C} \sharp B \simeq \sum_{(u : \sharp A)} \sum_{(v : \sharp B)} (\sharp f(u) = \sharp g(v)).
\]

Thus, it will suffice to prove that for all \(u : \sharp A\) and \(v : \sharp B\), we have

\[
\sharp(f(u_{\sharp}) = g(v_{\sharp})) \simeq (\# f(u) = \# g(v)).
\]

But by definition of the functoriality of \(\#\), the right-hand side is \(f(u_{\sharp})_{\sharp} = g(v_{\sharp})_{\sharp}\). Thus, the result follows from Theorem 3.7.

Left exact monadic modalities also have many nice features, for most of which we refer to Rijke, Shulman, and Spitters (2017) (and which are formalized in HoTT Project (2015)). One particularly important one is the following:

**Corollary 3.10.** \(\sharp\) preserves \(n\)-types for all \(n\).

**Proof.** We induct on \(n\). The base case \(n = -2\) (i.e. \(\#1 \simeq 1\)) is true for any monadic modality. Thus, suppose \(\#\) preserves \(n\)-types for some \(n\), let \(A\) be an \((n + 1)\)-type, and let \(u, v : \# A\). Then, \((u = v)\) is codiscrete, hence so is the proposition \((u = v)\) is an
n-type,’ so we can use \( \# \)-induction on \( u \) and \( v \). But by Theorem 3.7, for \( x, y : A \), we have \((x^\# = y^\#) \simeq \#(x = y)\), which is an \( n \)-type by the inductive hypothesis.

Rijke, Shulman, and Spitters (2017) and HoTT Project (2015) show that if a left-exact monadic modality is also accessible in a technical sense (defined there; see also Section 9), then the universe of modal types is modal. We can prove this for \( \# \) without accessibility, because of our formation rule for \( \#A \). The marker \( \{UA\} \) means that we need the full univalence axiom (although of course for the subuniverse of propositions we would need only propositional extensionality).

**Theorem \( \{UA\} \) 3.11.** The universe \( \text{Codisc} : \equiv \sum_{(A : \text{Type})} \text{iscodiscrete}(A) \) of codiscrete types is codiscrete.

**Proof.** Suppose \( X : \#\text{Codisc} \); we want to construct an element of \( \text{Codisc} \) (to define a retraction). The first component of it will be \( \# \) of some type, and for the purpose of constructing that type we may assume \( X \) is crisp. Thus, we have \( X^\# : \text{Codisc} \), which we can destruct further as \((A, c)\) for \( A : \text{Type} \) and \( c : \text{iscodiscrete}(A) \). Now we use \( A \) to be the type required. Explicitly, from \( X : \#\text{Codisc} \), we have constructed

\[ \#(\text{pr}_1(X^\#)) : \text{Type}. \]

By Theorem 3.5, this type is codiscrete, so gives an element of \( \text{Codisc} \).

To show that this is a retraction, if \( X \) is of the form \( Y^\# \), then we have

\[ \#(\text{pr}_1(Y^\#)) \equiv \#(\text{pr}_1(Y)) \]

which is equivalent to \( \text{pr}_1(Y) \), since by \( \text{pr}_2(Y) \) the latter is codiscrete. This completes the proof since codiscreteness is a proposition.

It follows that we can interpret the formal syntax of ordinary type theory (with only one kind of context) entirely into the codiscrete types, with \( \text{Codisc} \) playing the role of the universe, obtaining a model with \( \Pi \)-types, \( \Sigma \)-types, a unit type and identity types. We can also obtain some (higher) inductive types in this model, at least up to homotopy (in the sense of Univalent Foundations Program 2013, Section 5.5), simply by applying \( \# \) to their ordinary versions; this includes binary sums, the empty type, the natural numbers, \( n \)-truncations, suspensions and more general colimits.

This does not appear to work for general \( W \)-types, however; if \( W \) is generated by a constructor \( \sup : \prod_{(a : A)}(B(a) \to W) \to W \), there is not even any obvious way to define a function \( \prod_{(a : A)}(B(a) \to \#W) \to \#W \). It may be tempting to write ‘\( \lambda a. \lambda b. \sup(a, \lambda y. b(y)^\#)^\# \)’, but this is ill-typed: although the outer \( (\#)^\# \) makes all variables in the current context crisp, the inner \( \lambda y \) binds a new, cohesive variable \( y \), so that \( b(y) \) is not crisp and we cannot apply \( (\#)^\# \) to it.

We could work around this if \( \# \) were accessible, by adding the appropriate localization constructors to any HIT. Accessibility is a perfectly reasonable axiom, since it holds in all known examples. But in this paper, we have no need for it, and moreover, we want the base spatial type theory of this part (as opposed to the real-cohesive extension of it in the next part) to be free of axioms.
As an example of translating type theory into the codiscrete universe, let us define the Dedekind real numbers codiscretely. Since the types \( \mathbb{Z} \) and \( \mathbb{Q} \) are abstractly equivalent to \( \mathbb{N} \), and \( \sharp \mathbb{N} \) is the natural numbers in the codiscrete world, it follows that the codiscrete versions of \( \mathbb{Z} \) and \( \mathbb{Q} \) will be equivalent to \( \#\mathbb{Z} \) and \( \#\mathbb{Q} \), respectively. Now if we expand out the usual definition of the Dedekind reals \( \mathbb{R} \) into the basic type constructors, we obtain the type shown first in Figure 4. Thus, their codiscrete version should be the type \( \mathbb{R}' \) shown second in Figure 4. In contrast to \( \mathbb{Z} \) and \( \mathbb{Q} \), it is not at all obvious that \( \mathbb{R}' \) has anything to do with \( \#\mathbb{R} \); in the former, all intermediate constructions are discontinuous, whereas in the latter we do everything continuously and then at the very end forget the resulting topology.

We will see later on that they are in fact equivalent (Theorem 6.34), but this is a very special property of \( \mathbb{R} \). In most cases, the 'codiscrete version' of a type \( A \) is very different from \( \#A \). For instance, \( \#(\mathbb{R} \to \mathbb{R}) \) should be the set of continuous functions \( \mathbb{R} \to \mathbb{R} \) topologized codiscretely, whereas \( \mathbb{R}' \to \mathbb{R}' \) is the set of discontinuous functions.

Remark 3.12. The overall conclusion of this section so far can be summarized as ‘\( \# \) is a left-exact monadic modality’ (plus Theorem 3.11, which could be obtained from accessibility). This is a purely cohesive statement, i.e. it can be stated without any need for crisp variables. In Schreiber and Shulman (2012), using ordinary type theory without crispness, we took this statement as the axiomatic definition of \( \# \). The benefit of instead deriving it from rules involving crisp variables is that, as we will see in Section 4, \( \# \) then automatically has the correct relationship to \( \flat \).

3.1. Excluded middle revisited

With \( \# \) in hand, we can reformulate the crisp law of excluded middle (Axiom LEM) as a single term (modulo universe polymorphism).

Lemma [LEM] 3.13 (The sharp law of excluded middle). \( \prod_{(P:\text{Prop})} \#(P \lor \neg P) \).
Proof. Assume $P : \text{Prop}$. By the introduction rule for $\sharp$, it suffices to prove $P \lor \neg P$ assuming a crisp $P :: \text{Prop}$. But this is exactly Axiom LEM.

Let us now return to the topological viewpoint, where codiscrete types have the codiscrete topology. More generally, if $P : A \to \text{Type}$ and each $P(x)$ is codiscrete, then $\sum_{(x:A)} P(x)$ must have the ‘final topology’ induced by the first projection, i.e. the coarsest possible topology such that the first projection is continuous. To see this, note that sections $\prod_{(x:A)} P(x)$ are equivalent to sections $\prod_{(x:A)} \sharp P(x)$, and hence to conclusions $f(x) : P(x)$ under crisp hypothesis $x :: A$; but by our topological gloss, these are supposed to be simply discontinuous sections. Thus, the sections $\prod_{(x:A)} P(x)$ take no account of any topology on $P$, so it must be that $P$ has the final topology.

Remark 3.14. This topological gloss explains why the universe of codiscrete types must itself be codiscrete. A map into $A$ whose domain has the final topology is uniquely specified by a map into the underlying set (or $\infty$-groupoid) of $A$, i.e. it takes no account of the topology of $A$. Thus, a family of codiscrete types over $A$ should be the same as such a family over $\sharp A$, which is to say that any map $P : A \to \text{Codisc}$ must factor through $\sharp A$; thus, Codisc must be codiscrete.

In particular, to say that a predicate $P : A \to \text{Prop}$ is pointwise codiscrete must mean that it is a subspace, with the induced topology. We have already seen in Section 2.1 a different type-theoretic property that ought also to characterize subspaces, namely that $P$ is $\neg\neg$-closed. It turns out that one simple assumption beyond Axiom LEM guarantees that these two properties coincide. (See also Lawvere and Menni 2015, Corollary 4.5.)

Axiom $\sharp \emptyset$. $\emptyset$ is codiscrete.

Theorem $\{\text{LEM}, \sharp \emptyset\}$ 3.15.

(i) A proposition $P$ is codiscrete if and only if $\neg\neg P \to P$.
(ii) For any proposition $P$, we have $\sharp P \simeq \neg\neg P$.
(iii) For any type $A$, we have $\sharp \| A \| \simeq \neg\neg A$.

Proof. We prove (i) first. Since $P \to \neg\neg P$ always, we have $\neg\neg P \to P$ if and only if $\neg\neg P = P$. Now since $\emptyset$ is codiscrete, so is $\neg\neg P$ for any $P$. Thus, if $\neg\neg P = P$, then $P$ is also codiscrete.

Conversely, suppose $P$ is codiscrete, and suppose that $\neg\neg P$; we must show $P$. By crisp excluded middle, we have $\sharp (P \lor \neg P)$. Since our goal $P$ is codiscrete, by $\sharp$-induction we have $P \lor \neg P$. But since $\neg\neg P$, it must be that $P$. (Note that each of the two assumptions Axiom LEM and Axiom $\sharp \emptyset$ is used in exactly one direction of this equivalence.)

Now (ii) follows because $\sharp$ and $\neg\neg$ both reflect a proposition into the sub-poset of codiscrete propositions. Likewise, (iii) follows because $\sharp \| - \|$ and $\neg\neg$ both reflect an arbitrary type into that same poset (since $\neg\neg A$ is always a proposition).

In particular, in this case, the statement of Lemma 3.13 is equivalent to

$$\prod_{P : \text{Prop}} \neg\neg (P \lor \neg P),$$
which is a provable statement in type theory. Of course, we used Axiom LEM in proving Theorem 3.15, so this does not, in any way, absolve us from having to assume Axiom LEM. But it is nice to see that the cohesive version of excluded middle implied by our Axiom LEM is just the one that is already provable in constructive type theory.

3.2. The axiom of choice

We can now revisit the axiom of choice. First, however, let us consider all the ways of saying that a family \( P : A \to \text{Type} \) is ‘inhabited,’ of which we now have four. Here is a list of them with their intended topological meanings:

1. \( \prod_{(x:A)} P(x) \) says that \( \text{pr}_1 : (\sum_{(x:A)} P(x)) \to A \) has a continuous section.
2. \( \prod_{(x:A)} \# P(x) \) says that this projection has a discontinuous section.
3. \( \prod_{(x:A)} \Vert P(x) \Vert \) says that this projection is a quotient map.
4. \( \prod_{(x:A)} \# \Vert P(x) \Vert \) says that this projection is surjective on points.

(Assuming Axiom LEM, this is equivalent to \( \prod_{(x:A)} \neg \neg P(x) \).)

These topological glosses are mainly for our intuition, but in some cases, we can prove precise versions of them in type theory. For instance, statement (i) makes sense internally and is obviously true. Regarding (ii), we can prove:

**Lemma 3.16.** \( \prod_{(x:A)} \# P(x) \) if and only if \( \# \text{pr}_1 : \#(\sum_{(x:A)} P(x)) \to \# A \) has a section.

**Proof.** By \( \# \)-induction, \( \prod_{(x:A)} \# P(x) \) is equivalent to \( \prod_{(x:A)} \# P(x) \). This, in turn, is equivalent to saying that \( \text{pr}_1 : (\sum_{(x:A)} \# P(x)) \to \# A \) has a section. But by Lemma 3.8, \( \#(\sum_{(x:A)} P(x)) \simeq \sum_{(x:A)} \# P(x) \), and it is easy to see that the projections match up. \( \square \)

As for (iii), if \( A \) and \( P \) are sets, we can consider it justified by Univalent Foundations Program (2013, Theorem 10.1.5) (surjections of sets are regular epimorphisms). The general case requires a more refined notion of ‘quotient map,’ but is also true when suitably formulated (see Kraus 2016; Rijke 2017; van Doorn 2016).

Finally, regarding (iv), we can say the following.

**Lemma 3.17.** \( \prod_{(x:A)} \# \Vert P(x) \Vert \) if and only if \( \# \text{pr}_1 : \#(\sum_{(x:A)} P(x)) \to \# A \) is ‘codiscretely surjective,’ i.e. for all \( x : \# A \), we have \( \# \Vert \text{fib}_{\text{pr}_1}(x) \Vert \).

Note that ‘codiscretely surjective’ is what we get by interpreting ‘surjective’ in the universe of codiscrete types, as discussed at the end of Section 3. Thus, it is the appropriate way to say that a map of codiscrete types ‘is surjective’ in the untotopologized world.

**Proof.** By \( \# \)-induction, \( \prod_{(x:A)} \# \Vert P(x) \Vert \) is equivalent to \( \prod_{(x:A)} \# \Vert P(x) \Vert \). Thus, it will suffice to show that for any \( x : \# A \), we have \( \# \Vert P(x) \Vert \leftrightarrow \# \Vert \text{fib}_{\text{pr}_1}(x) \Vert \).

Suppose first that \( q : \# \Vert P(x) \Vert \); we will show \( \# \Vert \text{fib}_{\text{pr}_1}(x) \Vert \). We apply \( (-)^\sharp \), making our goal \( \# \Vert \text{fib}_{\text{pr}_1}(x) \Vert \) and our hypotheses crisp, allowing us to form \( q^\sharp : \# \Vert P(x) \Vert \). Now by functoriality of \( \# \), it suffices to assume \( p : P(x) \) and prove \( \text{fib}_{\text{pr}_1}(x) \). But then \( (x^\sharp, p)^\sharp : \#(\sum_{(x:A)} P(x)) \), and

\[ \# \text{pr}_1((x^\sharp, p)^\sharp) \equiv \text{pr}_1(x^\sharp, p)^\sharp \equiv x^\sharp \equiv x. \]
Next, suppose that \( z : \| \text{fib}_{\#pr_1}(x) \| \); we will show \( \# \| P(x_\#) \| \). Since \( \# \) takes values in codiscrete types, it suffices to show \( \# \# \| P(x_\#) \| \). Thus, we can apply \((-)\#\) to make our hypotheses crisp while keeping the goal \( \# \| P(x_\#) \| \). Since \( \# \) takes values in codiscrete types, it suffices to show \( \# \# \| P(x_\#) \| \). Thus, we can apply \((-)\#\) to make our hypotheses crisp while keeping the goal \( \# \| P(x_\#) \| \). Now we can form \( z_\# : \| \text{fib}_{\#pr_1}(x) \| \), and since \( \# \| P(x_\#) \| \) is a proposition, we can destruct \( z_\# \) to obtain \( q : \#(\sum_{x : A} P(x)) \) and \( \#pr_1(q) = x \). Next, by \#-induction, we can assume \( q \) is \((y,p_\#)\) for some \( y : A \) and \( p : P(y) \). But \( \#pr_1((y,p_\#)) \equiv pr_1(y,p_\#) \equiv y_\# \), so we have \( y_\# = x \). And since \( x \) is crisp, we have \( x \equiv x_\# \), so \( y_\# = x_\# \). But by Theorem 3.7, this type is equivalent to \( \#(y = x_\#) \). Thus, with another \#-induction, we obtain \( s : y = x_\#, \) so we can transport \( p \) along \( s \) to get an element of \( P(x_\#) \), and finally map it into \( \# \| P(x_\#) \| \).

**Corollary 3.18.** If \( f : A \to B \) is an embedding and \( \prod_{x : B} \#\text{fib}_f(x) \), then \( \#f \) is an equivalence.

**Proof.** Since \( f \) is an embedding, so is \( \#f \). Moreover, \( \#f \) has codiscrete fibres. Thus, Lemma 3.17 is sufficient to ensure that it is an equivalence. 

Now, since we cannot expect every surjection to have a continuous section, the conclusion of our axiom of choice should be \( \prod_{x : A} \#P(x) \). Therefore, it ought to suffice as a hypothesis that it be surjective on points, i.e. \( \prod_{x : A} \# \| P(x) \| \). Of course, the conclusion must be further \((−1)\)-truncated outside the \( \prod \), to be consistent with univalence. This might lead us to state the axiom of choice as

\[
\left( \prod_{x : A} \# \| P(x) \| \right) \to \left| \prod_{x : A} \#P(x) \right|.
\]

However, I believe this is not yet quite right either. Remember that the entire statement is also parameterized by \( A \) and \( P \). Thus, Equation (2) is actually asserting that one subobject of \( \sum_{(A : \text{Type})} (A \to \text{Type}) \) is contained in another. But the first subobject is codiscrete—i.e. has the subspace topology—whereas the second is not, at least not by definition: \( \| - \| \) does not preserve codiscreteness. Moreover, the reverse implication always holds; thus, Equation (2) is actually asserting that these two subobjects agree, and hence that the one on the right is a subspace even though we have not forced it to be. This seems unreasonable (although I do not currently have a concrete counterexample). Thus, instead we take the following:

**Axiom AC.** (Sharp axiom of choice) For any set \( A \) and type family \( P : A \to \text{Type} \), we have

\[
\left( \prod_{x : A} \# \| P(x) \| \right) \to \# \left| \prod_{x : A} \#P(x) \right|.
\]

This is admittedly quite a mouthful, and one is left wondering whether it is good for anything. Are there simpler special cases? What about, for instance, the countable axiom of choice, where \( A = \mathbb{N} \); surely there should not be any topology on \( \mathbb{N} \) to get in the way. Thus, we now turn to making sense of what it means to have a discrete topology.
4. The \( \flat \) modality

4.1. A no-go theorem

Discreteness is encoded by the modality \( \flat \) that is dual to \( \sharp \). Whereas \( \sharp \) is a reflector into the subcategory of codiscrete types, \( \flat \) is a coreflector into the subcategory of discrete types. However, unlike \( \sharp \), the behaviour of \( \flat \) cannot be stated as ordinary type-theoretic rules or axioms, because the resulting pullback stability would cause it to degenerate. This is the conclusion of the following 'no-go theorem.'

**Theorem 4.1.** Suppose we have the following data:

1. A predicate \( \text{in}_\Box : \text{Type} \to \text{Prop} \) that is invariant under equivalence, i.e. \( (A \simeq B) \to \text{in}_\Box(A) \to \text{in}_\Box(B) \). (This condition is, of course, automatic with univalence.)
2. An operation \( \Box : \text{Type} \to \text{Type} \), such that \( \text{in}_\Box(\Box(A)) \) for all \( A \).
3. For each \( A : \text{Type} \), a function \( \varepsilon_A : \Box A \to A \).
4. If \( \text{in}_\Box(B) \), then postcomposition with \( \varepsilon_A \) is an equivalence \( (B \to \Box A) \simeq (B \to A) \).

Then there exists \( U : \text{Prop} \) such that for all \( A \), we have

a. \( \text{in}_\Box(A) \leftrightarrow (A \to U) \) and

b. \( \Box A \simeq (A \times U) \).

**Proof.** First, we observe that if \( A \) is a proposition, so is \( \Box A \). For if \( x, y : \Box A \), then we have two functions \( \text{const}_x, \text{const}_y : \Box A \to \Box A \) whose composites with \( \varepsilon_A \) are equal. Hence, \( \text{const}_x = \text{const}_y \), and thus (evaluating them at \( x \), say) \( x = y \).

Second, we observe that if \( \varepsilon_A : \Box A \to A \) has a section \( s : A \to \Box A \), then \( \text{in}_\Box(A) \). It suffices to show that \( s \circ \varepsilon_A = \text{id}_{\Box A} \); but these are again two maps \( \Box A \to \Box A \) having the same composite with \( \varepsilon_A \).

Third, we can extend \( \Box \) to a functor such that \( \varepsilon \) is a natural transformation from \( \Box \) to the identity. (This is also the dual of facts we observed in Section 3 for \( \sharp \).)

Now let \( U := \Box 1 \). First, we prove statement a. On the one hand, if \( \text{in}_\Box(A) \), then \( A \to 1 \) factors through \( U \). On the other hand, suppose \( f : A \to U \); we will construct a section \( s : A \to \Box A \). For any \( x : A \), we have \( \text{const}_x : 1 \to A \), and hence by functoriality \( \Box \text{const}_x : \Box 1 \to \Box A \). But \( f : A \to \Box 1 \), so we have \( \Box \text{const}_x(f(x)) : \Box A \); we define this to be \( s(x) \). Now by naturality of \( \varepsilon \), we have

\[
\varepsilon_A(s(x)) = \varepsilon_A(\Box \text{const}_x(f(x))) = \text{const}_x(\varepsilon_f(x)) = x.
\]

Thus, \( \text{in}_\Box(A) \); this completes the proof of statement a.

Now we prove statement b. By statement a, we have \( \text{in}_\Box(A \times U) \); thus, the projection \( \text{pr}_1 : A \times U \to A \) factors through \( \varepsilon_A \) by some \( g : A \times U \to \Box A \). On the other hand, since \( \text{in}_\Box(\Box A) \), we have \( f : \Box A \to U \), and thus \( (\varepsilon_A, f) : \Box A \to A \times U \). The composite in one direction is

\[
(\varepsilon_A, f) \circ g = (\varepsilon_A \circ g, f \circ g) = (\text{pr}_1, \text{pr}_2) = \text{id}_{A \times U},
\]

where \( \varepsilon_A \circ g = \text{id}_A \) by definition of \( g \), while \( f \circ g = \text{id}_U \) since \( U \) is a proposition. The composite in the other direction is a map \( \Box A \to \Box U \) whose composite with \( \varepsilon_A \) is \( \varepsilon_A \), hence it must be the identity. This shows statement b. \( \square \)
We certainly do not expect \( \flat \) to be of the form \( - \times U \), so this means that we cannot describe \( \flat \) as ordinary type-theoretic rules or axioms. The central point of Theorem 4.1 is that the universal property expressed by assumption 4 can be applied in an arbitrary context; this is what enables us to construct a section \( A \to \Box A \) from a map \( A \to \Box I \). Thus, in order to describe a more general coreflection, we need to restrict the context in which it and its universal property can be applied. In our current setup, an obvious restriction to try is that the context should include only crisp variables. This is what we will do.

### 4.2. The rules for \( \flat \)

We now describe our rules for \( \flat \) in English; a type-theoretic presentation is shown in Figure 5.

— For any crisp type \( A :: \text{Type} \), there is a type \( \flat A \).
— For any crisp \( a :: A \), we have \( a^{\flat} :: \flat A \).
— An element of \( \flat A \) may always be assumed to be of the form \( u^{\flat} \) for some crisp variable \( u :: A \). This is called \( \flat \)-induction. Syntactically, if \( C \) is a type depending on \( x :: \flat A \), and \( N : C[u^{\flat}/x] \) is an expression depending on a crisp variable \( u :: A \), while we have an element \( M :: \flat A \), we have an induced element

\[
(\text{let } u^{\flat} := M^{\flat} \text{ in } N) \equiv N[M/u] : C[M^{\flat}/x].
\]

Here, the notation \( C[M/x] \) indicates that the expression \( M \) is substituted for the variable \( x \) in \( C \). Univalent Foundations Program (2013) used function notation when discussing induction principles, but for \( \flat \) we cannot, since we do not have a basic notion of ‘function of a crisp variable’ (indeed, one of the purposes of \( \flat \) is to provide such a notion); thus, we have to talk about substituting into expressions rather than evaluating functions.
— The expected computation rule: \( (\text{let } u^{\flat} := M^{\flat} \text{ in } N) \equiv N[M/u] \) if \( M :: A \) is crisp.

It is crucial that we cannot apply \( \flat \) to types containing cohesive variables. This reflects the fact that, by Theorem 4.1, ‘retopologizing discretely’ is not a continuous operation on the space of spaces. (It is possible to do this for ‘retopologizing codiscretely,’ so that \( \sharp \) requires no such restriction. Indeed, by Theorem 3.11, the type of codiscrete types is codiscrete, so any function into it is automatically continuous.)
Remark 4.2. Strictly speaking, the syntax ‘let \( u \downarrow \downarrow := M \) in \( N \)’ should also notate the type family \( C \), since that is not generally uniquely inferrable. However, when working informally, this will not matter much.

Remark 4.3. Dually to \( \# \) (see Remark 3.1), \( \downarrow \downarrow \) is a positive type former. This means, roughly, that its elimination rule is chosen to match its introduction rule(s). For instance, the coproduct \( A + B \) is positive because to define a (dependent) function out of it, we use case analysis, which essentially means we have to specify the values of that function on all the ways to construct an element of \( A + B \) (namely via \( \text{inl} \) or \( \text{inr} \)). In the case of \( \downarrow \downarrow \), the introduction rule says that the way to get an element of \( \downarrow \downarrow A \) is to have a crisp element of \( A \); thus, the elimination rule says that to define a function out of \( \downarrow \downarrow A \), it suffices to assume we have a crisp element of \( A \).

In contrast to negative type formers, whose uniqueness principles tend to be judgemental rules, positive type formers tend to have uniqueness principles that are only typal, but can be proven from their elimination and computation rules. As we will now see, this is also the case for \( \downarrow \downarrow \).

Lemma 4.4 (Uniqueness principle for \( \downarrow \downarrow \)). Let \( A :: \text{Type} \), \( C :: \downarrow \downarrow A \to \text{Type} \), and \( f :: \prod_{(x :: \downarrow \downarrow A)} C(x) \). For any \( x :: \downarrow \downarrow A \), we have

\[
\text{(let } u \downarrow \downarrow := x \text{ in } f(u \downarrow \downarrow)) = f(x).
\]

Proof. Let \( P(x) \equiv (\text{let } u \downarrow \downarrow := x \text{ in } f(u \downarrow \downarrow)) = f(x) \); then \( P : \downarrow \downarrow A \to \text{Type} \). Thus, in proving \( \prod_{(x :: \downarrow \downarrow A)} P(x) \), we can use \( \downarrow \downarrow \)-induction. But when \( x \) is \( v \downarrow \downarrow \), we have \( P(v \downarrow \downarrow) \equiv (f(v \downarrow \downarrow) = f(v \downarrow \downarrow)) \), using the computation rule for \( \downarrow \downarrow \) on the left-hand side, and this is trivial by reflexivity. In symbols, we have

\[
\text{(let } v \downarrow \downarrow := x \text{ in } \text{refl}_{f(v \downarrow \downarrow)}) : (\text{let } u \downarrow \downarrow := x \text{ in } f(u \downarrow \downarrow)) = f(x). \quad \square
\]

As with \( \# \), we can make \( \downarrow \downarrow \) into a functor, at least on crisp functions: given \( A, B :: \text{Type} \) and \( f :: A \to B \), we define \( \downarrow \downarrow f :: \downarrow \downarrow A \to \downarrow \downarrow B \) by

\[
\downarrow \downarrow f(x) \equiv (\text{let } u \downarrow \downarrow := x \text{ in } f(u \downarrow \downarrow)). \quad (4)
\]

We might be tempted to write instead \( \text{(let } u \downarrow \downarrow := x \text{ in } f(u \downarrow \downarrow)) \), but this is invalid because the inside of \( - \downarrow \downarrow \) cannot contain cohesive variables such as \( x \). For the same reason, we require not only \( A \) and \( B \), but also \( f \), to be crisp. In particular, we cannot deduce from this a map \( (A \to B) \to (\downarrow \downarrow A \to \downarrow \downarrow B) \). The best we can do is a map \( \downarrow \downarrow (A \to B) \to \downarrow \downarrow (\downarrow \downarrow A \to \downarrow \downarrow B) \), defined by \( \downarrow \downarrow \)-induction followed by Equation (4).

When we try to verify the functoriality of this operation, we see that we need some additional lemmas. Given \( A, B, C :: \text{Type} \) and \( f :: A \to B \) and \( g :: B \to C \), in addition to Equation (4), we have

\[
\downarrow \downarrow g(y) \equiv (\text{let } v \downarrow \downarrow := y \text{ in } g(v \downarrow \downarrow)), \quad (5)
\]

\footnote{Well, at least in the usual sense that this adjective is used for dependent type theories. In the sequent calculus of Licata and Shulman (2016), \( \downarrow \downarrow \) is (conjecturally) positive in the precise sense of ‘focusing,’ but our presentation of \( \downarrow \downarrow \) breaks some of this positivity. However, this often happens to positive types when translating from sequent calculus into dependent type theory.}
and thus for any \( x : \mathcal{A} \), we have

\[
\mathcal{B} g(\mathcal{B} f(x)) \equiv (\text{let } v^\mathcal{B} \equiv (\text{let } u^\mathcal{B} \equiv x \text{ in } f(u)^\mathcal{B}) \text{ in } g(v)^\mathcal{B})
\]

but on the right-hand side, what are we to do with this ‘nested let’? We have to apply the following lemma, called a (typal) commuting conversion.

**Lemma 4.5 (Commuting let with itself).** Whenever both sides typecheck, we have

\[
\text{let } v^\mathcal{B} \equiv (\text{let } u^\mathcal{B} \equiv M \text{ in } N) \text{ in } P = (\text{let } u^\mathcal{B} \equiv M \text{ in } \text{let } v^\mathcal{B} \equiv N \text{ in } P)
\]

Moreover, if the left-hand side typechecks, so does the right-hand side, while if the right-hand side typechecks and \( u \) does not occur in \( P \), then the left-hand side typechecks.

**Proof.** One proof is to do \( \mathcal{B} \)-induction on \( M \): when \( M = w^\mathcal{B} \) for \( w :: \mathcal{A} \), both sides of the goal reduce to \( (\text{let } v^\mathcal{B} \equiv N[w/u] \text{ in } P) \), so we can apply reflexivity. Another is to apply the uniqueness principle in reverse, followed by the computation rule:

\[
\text{let } v^\mathcal{B} \equiv (\text{let } u^\mathcal{B} \equiv M \text{ in } N) \text{ in } P
= \text{let } w^\mathcal{B} \equiv M \text{ in } \text{let } v^\mathcal{B} \equiv (\text{let } u^\mathcal{B} \equiv w^\mathcal{B} \text{ in } N) \text{ in } P
\equiv \text{let } w^\mathcal{B} \equiv M \text{ in } \text{let } v^\mathcal{B} \equiv N[w/u] \text{ in } P
\equiv \text{let } u^\mathcal{B} \equiv M \text{ in } \text{let } v^\mathcal{B} \equiv N \text{ in } P.
\]

The second proof makes the typechecking conditions clearer, as these rewriting steps can always be performed forwards, whereas to perform them backwards we need to know that \( u \) does not occur in \( P \). \( \square \)

**Remark 4.6.** In Remark 2.1, we mentioned the difference between implication and entailment, whose type-theoretic syntaxes are \( \Gamma \vdash f : A \to B \) and \( \Gamma, x : A \vdash f(x) : B \), respectively. Both of these, however, are single ‘hypothetical judgements,’ whose hypotheses consist of a context of variables (either ‘\( \Gamma \)’ or ‘\( \Gamma, x : A \)’). By contrast, Lemma 4.5 cannot be expressed as a single hypothetical judgement, because \( M, N \) and \( P \) must be ‘metavariables’ denoting expressions rather than ordinary variables. This should be clear from the condition ‘\( u \) does not occur in \( P \),’ which would be non-sensical if \( P \) were a variable. In type-theoretic language, Lemma 4.5 is a derivable rule rather than a single judgement, because among its hypotheses are other hypothetical judgements; Figure 6 shows it written out in type-theoretic syntax. We will try to avoid such ‘meta-statements’ as much as possible, but in this section and the next, we will need a few more of them.

With Lemma 4.5, we can now verify the functoriality of \( \mathcal{B} \):

\[
\mathcal{B} g(\mathcal{B} f(x)) \equiv \text{let } v^\mathcal{B} \equiv (\text{let } u^\mathcal{B} \equiv x \text{ in } f(u)^\mathcal{B}) \text{ in } g(v)^\mathcal{B}
= \text{let } u^\mathcal{B} \equiv x \text{ in } (\text{let } v^\mathcal{B} \equiv f(u)^\mathcal{B} \text{ in } g(v)^\mathcal{B})
\equiv \text{let } u^\mathcal{B} \equiv x \text{ in } g(f(u))^\mathcal{B}
\equiv \mathcal{B}(g \circ f)(x).
\]
Functoriality on identities is easier, using the uniqueness principle:
\[ \eta(\text{id}_A)(x) \equiv (\text{let } u^\flat := x \text{ in } u^\flat) = x. \]

Note that unlike the functoriality of \( \varepsilon \), these are typal rather than judgemental equalities. Thus, they should actually be viewed as the first level of a ‘coherent \( \infty \)-functor’; later on we will construct the next level by showing that \( \eta \) is also ‘functorial on homotopies.’

Continuing with the analogy to \( \varepsilon \), in place of the ‘unit’ \( (-)^\sharp : A \to \# A \), we have a ‘counit’ \( (-)_\# : \# A \to A \), which is defined for any (crisp!) \( A : \text{Type} \) by
\[ x_\# := (\text{let } u^\flat := x \text{ in } u). \]

The computation rule tells us that for \( v :: A \), we have
\[ v_\# := (\text{let } u^\flat := v^\flat \text{ in } u) \equiv v. \]  

We expect \( (-)_\# \) to be a natural transformation (and eventually a ‘coherent \( \infty \)-natural transformation’) from \( \eta \) to the identity functor. To verify this, let \( A, B :: \text{Type} \) and \( f :: A \to B \) and fix any \( x :: \# A \); then we would like to compute
\begin{align*}
(\eta f(x))_\# &= (\text{let } u^\flat := x \text{ in } f(u^\flat))_\#
= \text{let } u^\flat := x \text{ in } f(u)^\flat, \quad \text{(Lemma 4.5)}
\equiv \text{let } u^\flat := x \text{ in } f(u)
= f(\text{let } u^\flat := x \text{ in } u)
\equiv f(x_\#).
\end{align*}

To fill in the step marked ?, we need another commuting conversion.

**Lemma 4.7 (Commuting let with function application).** Suppose given types \( A, B \) and \( C \), where \( A \) is crisp, \( B \) may depend on \( x :: \# A \) and \( C \) may depend on \( y :: B \). Also, suppose we have expressions \( M :: \# A \) and \( N :: B[u^\flat/x] \) and \( f :: \prod_{x :: \# A} \prod_{y :: B} C \), where \( N \) may depend on \( u :: A \). Then, we may conclude
\[ f(M, (\text{let } u^\flat := M \text{ in } N)) = (\text{let } u^\flat := M \text{ in } f(u^\flat, N)). \]

Like Lemma 4.5, this is formally a ‘derivable rule’; in Figure 6, we have written it out in type-theoretic syntax.
Proof. As before, \( \flat \)-induction on \( M \) reduces both sides to \( f(w^\flat, N[w/u]) \); or we can expand and contract:

\[
\begin{align*}
f(M, (\text{let } \upsilon^\flat := M \text{ in } N)) \\
= \text{let } \upsilon^\flat := M \text{ in } f(v^\flat, (\text{let } \upsilon^\flat := v^\flat \text{ in } N)) \\
\equiv \text{let } \upsilon^\flat := M \text{ in } f(v^\flat, N[v/u]) \\
\equiv \text{let } \upsilon^\flat := M \text{ in } f(u^\flat, N).
\end{align*}
\]

Analogous commuting conversions can be proven for the constructors and eliminators of all other ordinary types; we leave them to the reader. However, there is one conversion we need that is more subtle, relating to \( \flat \) itself. To see how this arises, consider the obvious dual claim to Equation (6), that \( x^{\flat \flat} = x \). Note that \( x^{\flat \flat} \) only makes sense when \( x \) is a crisp element of \( \flat A \), since we can only apply \( (\cdot)^\flat \) to crisp elements. In this case, we would like to write

\[
x^{\flat \flat} \equiv (\text{let } \upsilon^\flat := x \text{ in } u^\flat) = (\text{let } \upsilon^\flat := x \text{ in } u^\flat) = x,
\]

using the uniqueness principle (Lemma 4.4) at the end; but in the middle, we need to be able to commute \( (\cdot)^\flat \) past \text{let}. However, if we try to prove this as we did for the other commuting conversions, we run into the problem that (as remarked above) \( x : \flat A \) must be crisp, whereas \( \flat \)-induction is only stated for types \( C \) depending on a cohesive variable \( x : \sharp A \). Thus, we now take a brief digression to investigate crisp induction principles.

5. Crisp induction principles

Here is the lemma we need about \( \flat \). Note that its proof involves \( \sharp \) as well!

Lemma 5.1 (Crisp \( \flat \)-induction). Let \( C \) be a crisp type depending on a crisp variable \( x : \flat A \), and \( N : C[u^\flat/x] \) a crisp expression depending on a crisp variable \( u : A \). If we have a crisp element \( M : \flat A \), then we have an element of \( C[M/x] \), which we denote

\[
(\text{let } \upsilon^\flat := M \text{ in } N) : C[M/x].
\]

Moreover, if \( M : A \), then

\[
(\text{let } \upsilon^\flat := M^\flat \text{ in } N) \equiv N[M/u].
\]

Proof. If we have only a cohesive variable \( x : \sharp A \), we can still write \( \sharp C : \text{Type} \), since cohesive variables can become crisp inside \( \sharp (-) \). Similarly, if \( u : A \), we have \( N^\sharp : \sharp C[u^\flat/x] \). Thus, since \( M : \flat A \), we have \( (\text{let } \upsilon^\flat := M \text{ in } N^\sharp) : \sharp C[M/x] \). And since this expression is crisp, we have \( (\text{let } \upsilon^\flat := M \text{ in } N^\sharp)_\sharp : C[M/x] \), which we define to be \( \text{let } \upsilon^\flat := M \text{ in } N \). Finally, if \( M : A \), then

\[
(\text{let } \upsilon^\flat := M^\flat \text{ in } N^\sharp)_\sharp \equiv N[M/u]^\sharp \equiv N[M/u].
\]

This proof is somewhat tricky, so we include a derivation tree for it; see Figure 7. The crispness restriction on \( C \) and \( N \) can be worked around in the usual ‘Frobenius’ way:
\[
\begin{align*}
\Delta, x :: bA & \mid \vdash C : \text{Type} \\
\Delta & \mid x : bA \vdash \emptyset C : \text{Type} & \Delta, u :: A & \mid \vdash N : C[u^b/x] \\
\Delta & \mid \vdash M : bA & \Delta, u :: A & \mid \vdash N^x : \emptyset C[u^b/x] \\
\Delta & \mid \vdash (\text{let } u^b := M \text{ in } N^z) : \emptyset C[M/x]
\end{align*}
\]

\[\text{Fig. 7. A derivation tree for Lemma 5.1.}\]

\[
\begin{align*}
\Delta, x :: bA & \mid \vdash C : \text{Type} & \Delta, u :: A & \mid \vdash N : C[u^b/x] \\
\Delta & \mid \vdash M : bA & \Delta & \mid \Gamma (\text{let } u^b := M \text{ in } N) : C[M/x] \\
(\text{let } u^b := M \text{ in } N) & \equiv N[M/u]
\end{align*}
\]

\[\text{Fig. 8. Crisp } \flat\text{-induction and commuting let with } \flat.\]

if \( C \) and \( N \) depend on some cohesive variable \( y : B \), we can form \( C' := \prod_{(y : B)} C \) and \( N' := \lambda y. N \) and apply Lemma 5.1 to them instead. By induction over metatheoretic natural numbers, we can incorporate an arbitrary cohesive context.

Now, we can prove the needed commuting conversion.

**Lemma 5.2 (Commuting let with \((-)^\flat\)).** Given crisp expressions \( M :: \flat A \) and \( N :: B[u^b/x] \), where \( B \) depends on a crisp variable \( x :: \flat A \) and \( N \) depends on \( u :: A \), we have

\[
(\text{let } u^b := M \text{ in } N) = (\text{let } u^b := M \text{ in } N^b).
\]

**Proof.** By Lemma 5.1, we can replace \( M \) by \( w^b \) for some \( w :: A \). Now both sides reduce to \( N^b[w/u] \).

At last, the computation (7) makes sense. Later on, we will also have use for the corresponding type-level commuting conversion:

**Lemma 5.3 (Commuting let with \( \flat \)).** Given crisp expressions \( M :: \flat A \) and \( N :: \text{Type} \), where \( N \) may depend on \( u :: A \), we have

\[
\flat (\text{let } u^b := M \text{ in } N) = (\text{let } u^b := M \text{ in } \flat N).
\]

The idea used to prove Lemma 5.1 can be applied to other positive types as well. For instance, since we assert the rules of ordinary type theory only for cohesive variables, the
The computation rule follows from those for \texttt{case} and \$\sharp\$ combined. \hfill \square

A more or less equivalent way to state Theorem 5.4 is that \$\sharp\$ preserves coproducts. This is sensible from a category-theoretic point of view, since we will soon see that \$\sharp\$ is left adjoint to \$\sharp\$.

\textbf{Corollary 5.5.} \(\sharp A + \sharp B \simeq \sharp(A + B)\) for any crisp types \(A\) and \(B\).
Proof. To map from left to right, we do ordinary case analysis and then use the functoriality of \( \blacklozenge \). From right to left, we do \( \blacklozenge \)-induction, then crisp case analysis, then apply \((-)\) in both cases. We leave the details to the reader, along with the proof that these are inverses.

We can apply the same method to other positive types such as \( \emptyset \) and \( \mathbb{N} \), concluding in particular that \( \blacklozenge \emptyset \simeq \emptyset \) and \( \blacklozenge \mathbb{N} \simeq \mathbb{N} \). It also applies to identity types:

**Theorem 5.6 (Crisp Id-induction).** Suppose \( C :: \text{Type} \) depends on \( u, v :: B \) and \( p :: u = v \), and that we have \( d :: C[u/v, \text{refl}_u/p] \) depending on \( u :: B \). Then, for any \( u, v :: B \) and \( p :: u = v \), we have \( J^p_{u,v,p,C}(u.d; u,v,p) : C \), such that \( J^p_{u,v,p,C}(u.d; u,u, \text{refl}_u) \equiv d \).

Proof. As before, if we have cohesive variables \( x, y : B \) and \( q : x = y \), we can write \( \#C[x/u,y/v,q/p] \). Now we apply ordinary Id-induction: to inhabit \( \#C[x/u,y/v,q/p] \) for all \( x, y : A \) and \( q : x = y \), it suffices to inhabit \( \#C[x/u,x/v,\text{refl}_x/p] \) for all \( x : A \). But such an inhabitant is supplied by \( d[x/u]^\pi \) (note that \( u \) is crisp in \( d \), but inside \((-)^\pi \) we can assume \( x \) to be crisp).

We have shown that for all \( x, y : A \) and \( q : x = y \), we have

\[
J_{x,y,q,\#C[x/u,y/v,q/p]}(x.d[x/u]^\sharp; x,y,q) : \#C[x/u,y/v,q/p].
\]

Now, if we in fact have crisp assumptions \( u, v :: A \) and \( p :: u = v \), then this becomes a crisp conclusion

\[
J_{x,y,q,\#C[x/u,y/v,q/p]}(x.d[x/u]^\sharp; u,v,p) : \#C
\]

(since \( C \) and \( d \) are also crisp). Thus, we can apply \((-)^\# \) to obtain an element of \( C \). Finally, if we substitute \( (u,u, \text{refl}_u) \) for \( (u,v,p) \), we have

\[
J_{x,y,q,\#C(x,y,q)}(x.d[x/u]^\sharp; u,u, \text{refl}_u)^\# \equiv d^\# \equiv d.
\]

In particular, we can transport in a crisp type along a crisp identification: if \( C :: \text{Type} \) depends on \( z :: A \) and we have \( a, b :: A \) and \( p :: a = b \) and \( c :: C[a/z] \), then we have \( p^*(c) :: C[b/z] \). We repeat that the important novelty here is that the definition of \( C \) might use the crispness of \( z \). Note that since crisp Id-induction is defined using an ordinary Id-induction followed by \((-)^\# \), the uniqueness rule for \( \# \) implies \( p^*(c)^\# = p^*(c^\#) \).

With ‘crisp transport,’ we can state and prove crisp induction principles for some higher inductive types as well, and particularly for homotopy colimits.

**Theorem 5.7 (Crisp coeq-induction).** Let \( A, B :: \text{Type} \) and \( f, g :: A \rightarrow B \) with homotopy coequalizer \( q :: B \rightarrow \text{coeq}(f,g) \) and \( p :: \prod_{(u:A)} q(f(u)) = q(g(u)) \). Let \( C :: \text{Type} \) depend on \( z :: \text{coeq}(f,g) \), let \( N :: C[q(v)/z] \) depend on \( v :: B \), and let \( P :: p(u)*_{(N[q(f(u))/z])} = N[q(g(u))/z] \) depend on \( u :: A \). Then, for any \( M :: \text{coeq}(f,g) \), we have an element of \( C[M/z] \), which computes to \( N \) on \( q \) and to \( P \) on \( f,g \).

Proof. For a cohesive variable \( z :: \text{coeq}(f,g) \), we can write \( \#C :: \text{Type} \), and apply ordinary coeq-induction. We also have \( N^\# :: \#C[q(v)/z] \) depending on a cohesive \( v :: B \). As remarked above, we have \( p(u)*_{(N[q(f(u))/z])} = p(u)*_{(N[q(f(u))/z])} \). Now both
\(p(u) \ast (N[q(f(u))/z])^\sharp\) and \(N[q(g(u))/z]^\sharp\) belong to \(♯C[q(g(u))/z]\), so by the proof of Theorem 3.7, we have

\[
\left( p(u) \ast (N[q(f(u))/z])^\sharp = N[q(g(u))/z]^\sharp \right)
\]

\[
≃ ♯ \left( p(u) \ast (N[q(f(u))/z])^\sharp = N[q(g(u))/z]^\sharp \right)
\]

\[
≡ ♯ \left( p(u) \ast (N(q(f(u))/z)) = N(q(g(u))/z) \right)
\]

which is the type of \(P^\sharp\). Thus, we have the inputs to ordinary coeq-induction, after which we can apply \((-\ast)_z\) as usual. 

Similarly, we have crisp induction for \(S^1\), for pushouts, and so on. The case of truncations requires a little more thought.

**Theorem 5.8 (Crisp truncation-induction).** Let \(n :: \mathbb{N}\) and \(A :: \text{Type}\), let \(C\) be a crisp \(n\)-type depending on \(z :: \|A\|_n\), and let \(N :: C[x/n/z]\) depend on \(x :: A\). For any \(M :: \|A\|_n\), we have an element of \(C[M/z]\), which computes to \(N\) on \(|−|_n\).

**Proof.** For a cohesive variable \(z :: \|A\|_n\), we have \(♯C :: \text{Type}\), to which we apply ordinary truncation-induction. We have \(N^\sharp :: ♯C[x/n/z]\) for a cohesive \(x :: A\); so it remains to show that \(♯C\) is an \(n\)-type. But in Corollary 3.10, we showed that \(♯\) preserves \(n\)-types, and it is straightforward to generalize this (using crisp \(\mathbb{N}\)-induction) to a derivable rule in which \(n\) is crisp and the input \(n\)-type \(C\) depends on crisp variables (that become cohesive in \(♯C\)). Thus, ordinary truncation-induction applies, after which we use \((-\ast)_z\) as usual.

In the sequent calculus of Licata and Shulman (2016), these crisp induction principles are taken as given, rather than derived from \(♯\). This produces a better-behaved calculus in a proof-theoretic sense. However, not all (higher) inductive types can consistently admit a crisp induction principle; we will see a counterexample in Remark 9.13. For this reason, I prefer to derive all such principles from \(♯\).

### 6. Discreteness and \(♭\) revisited

#### 6.1. Identifications in \(♭\)

Just as we showed in Theorem 3.7 that \(♯\) is left exact using an encode–decode argument, we can do the same for \(♭\).

**Theorem 6.1.** For any \(x, y :: A\), we have \((x^♭ = y^♭) \simeq ♭(x = y)\).

**Proof.** We define \(\text{code} :: ♭A \to ♭A \to \text{Type}\) by

\[
\text{code}(u, v) := (\text{let } x^♭ := u \text{ in } (\text{let } y^♭ := v \text{ in } ♭(x = y))).
\]

Then, for any \(x, y :: A\), \(\text{code}(x^♭, y^♭) \equiv ♭(x = y)\). Thus, it will suffice to prove that for any \(u, v :: ♭A\), we have \(\text{code}(u, v) \simeq (u = v)\).

First, note that for any \(u :: ♭A\), we have

\[
r(u) := (\text{let } x^♭ := u \text{ in } (\text{refl}_x)^♭) : \text{code}(u, u).
\]
As usual, we then define encode : \( \prod_{(u,v) : \mathcal{B}A} (u = v) \to \text{code}(u,v) \) by encode\( (p) \equiv p \cdot r(u) \).

Second, to construct decode : \( \prod_{(u,v) : \mathcal{B}A} \text{code}(u,v) \to (u = v) \), if we destruct \( u \) and \( v \) as \( x^\flat \) and \( y^\flat \), respectively with \( x, y : A \), we obtain \( c : \mathcal{B}(x = y) \) with a goal of \( (x^\flat = y^\flat) \).

Then, we can further destruct \( c \) as a crisp \( p : x = y \). Now our assumptions of \( x, y \) and \( p \) suffice to apply crisp Id-induction, Theorem 6.1; this reduces our goal to \( (x^\flat = y^\flat) \) for a given \( x : A \). But now, of course, we can use \( \text{refl}_x^\flat \). In symbols, we have

\[
\text{decode}_{u,v}(c) \equiv (\text{let } x^\flat \equiv u \text{ in } (\text{let } y^\flat \equiv v \text{ in } (\text{let } p^\flat \equiv c \text{ in } J^\flat_{x,y,p}(x^\flat = y^\flat)(x,\text{refl}_x^\flat, x, y, p)))).
\]

Now as remarked in Theorem 3.7, it suffices to show that for any \( u, v : \mathcal{B} A \) and \( c : \text{code}(u,v) \), we have \( \text{encode}(\text{decode}(c)) = c \). Of course, we destruct \( u, v, c \) as \( x, y : A \) and \( p : x = y \), and then apply crisp Id-induction; the goal then becomes \( (\text{refl}_x^\flat)_*(\text{refl}_x^\flat) = \text{refl}_x^\flat \), which is true.

**Corollary 6.2.** For any \( u, v : \mathcal{B} A \), we have \( (u = v) \simeq \mathcal{B}(u_\flat = v_\flat) \).

**Proof.** We showed above that any crisp element \( u : \mathcal{B} A \) is equal to \( u_\flat^\flat \), so this follows from Theorem 6.1.

Note that \( u \) and \( v \) in the statement of Corollary 6.2 must be crisp for the right-hand side to make sense. However, inspecting the proof of Theorem 6.1, we extract a more general statement that applies even to cohesive variables \( x, y : \mathcal{B} A \), namely

\[
(x = y) \simeq (\text{let } x^\flat \equiv u \text{ in } (\text{let } y^\flat \equiv v \text{ in } \mathcal{B}(x = y))).
\]

Theorem 6.1 has many useful consequences. For instance, it implies the functoriality of \( \mathcal{B} \) on homotopies. That is, suppose that \( f, g : A \to B \) and \( H : f \sim g \) is a crisp homotopy, i.e. \( H : \prod_{x : A} f(x) = g(x) ; \) we would like to conclude \( \mathcal{B} f \sim \mathcal{B} g \). Introducing \( x : \mathcal{B} A \) and using \( \mathcal{B} \)-induction, this reduces to saying that for any \( u : A \), we have \( f(u)^\flat = g(u)^\flat \). This looks like it should be immediate from \( H(u) : f(u) = g(u) \), but since \( (-)^\flat \) is not a function, we cannot write ‘\( \text{ap}_{(-)^\flat}(H(u)) \).’ Instead we use the following.

**Corollary 6.3.** For any \( u, v : B \) and \( p : u = v \), we have \( p^\flat = p^\# \) such that \( \text{refl}_u^\flat p^\# = \text{refl}_u^\flat \) and \( \text{ap}_{(-)^\flat}(p^\#) = p \).

**Proof.** We apply the decode from Theorem 6.1 to \( p^\flat : \mathcal{B}(u = v) \). The first equality \( \text{refl}_u^\flat p^\# = \text{refl}_u^\flat \) follows from the definition of decode. For the second, we use crisp Id-induction on \( p \), and observe that

\[
\text{ap}_{(-)^\flat}(\text{refl}_u^\flat)^\flat) = \text{ap}_{(-)^\flat}(\text{refl}_u^\flat) = \text{refl}_u
\]

using the first equality.

Now we can define the action of \( \mathcal{B} \) on a homotopy \( H \) by

\[
\mathcal{B} H(x) \equiv (\text{let } u^\flat \equiv x \text{ in } \mathcal{B}(u)^\flat).
\]

Functionality on homotopies also implies that the functor \( \mathcal{B} \) preserves (crisp) equivalences: if \( e : A \simeq B \), then we have \( \mathcal{B} e : \mathcal{B} A \simeq \mathcal{B} B \).
We can also regard Theorem 6.1 as the corollary of Theorem 5.6 saying analogously to Corollary 5.5 that \( \flat \) preserves identity types. It implies analogous corollaries of Theorems 5.7 and 5.8.

**Corollary 6.4.** For any \( A, B :: \text{Type} \) and \( f, g :: A \to B \), we have \( \flat(\text{coeq}(f, g)) \simeq \text{coeq}(\flat f, \flat g) \).

**Proof.** From left to right, we apply \( \flat \)-induction, crisp \( \text{coeq} \)-induction, and then \( (\cdot)^\flat \) and the structure maps of \( \text{coeq}(\flat f, \flat g) \). From right to left, we apply ordinary \( \text{coeq} \)-induction first, and then \( \flat q : \flat B \to \flat(\text{coeq}(f, g)) \); finally, we need to show that for \( x : \flat A \), we have \( \flat q(\flat f(x)) = \flat q(\flat g(x)) \); but this follows from the above functoriality of \( \flat \) on homotopies. That the composites are identities is proven similarly.

**Corollary 6.5.** \( \flat S^1 \simeq S^1 \).

**Theorem 6.6.** For any \( n :: \mathbb{N} \) and \( A :: \text{Type} \), if \( A \) is crisply an \( n \)-type, so is \( \flat A \).

**Proof.** We do crisp induction (as in Section 5) over \( n :: \mathbb{N} \), beginning with \( n = -2 \). If \( A \) is crisply contractible, we have some \( c :: A \), hence also \( c^\flat : \flat A \). Now, any \( x : \flat A \) can be destructed as \( u^\flat \) for \( u :: A \), and by Theorem 6.1, \( (u^\flat = a^\flat) \simeq \flat(u = a) \). But \( A \) is crisply contractible, so we have \( p :: u = a \), hence \( p^\flat : \flat(u = a) \).

For the induction step, suppose \( A \) is crisply an \((n+1)\)-type. Then, \( x, y : \flat A \) can be destructed as \( u^\flat \) and \( v^\flat \) for \( u, v :: A \), and we have \( (u^\flat = v^\flat) \simeq \flat(u = v) \). But \( u = v \) is crisply an \( n \)-type; hence, by the inductive hypothesis so is \( \flat(u = v) \).

**Corollary 6.7.** For \( n :: \mathbb{N} \) and \( A :: \text{Type} \), we have \( \| \flat A \|_n \simeq \| \flat A \|_n \).

**Proof.** From left to right, we apply \( \flat \)-induction and crisp truncation-induction, followed by \( (\cdot)^\flat \). From right to left, we apply ordinary truncation-induction (using Theorem 6.6) and then functoriality of \( \flat \) on \( |-|_n \). The composites on either side are proven to be identities similarly.

Third, Theorem 6.1 implies \( \flat \) is left exact, using an analogue of Lemma 3.8:

**Lemma 6.8.** For any \( A :: \text{Type} \) and \( B :: A \to \text{Type} \), we have
\[
\flat \left( \sum_{x : A} B(x) \right) \simeq \sum_{x : \flat A} (\text{let } u^\flat := x \text{ in } \flat B(u)) .
\]

**Proof.** From left to right, we first apply \( \flat \)-induction, then destruct to get a crisp \( u :: A \) and \( v :: B(u) \). Then, we can form \( u^\flat \) and \( v^\flat : \flat B(u) \), and \( (u^\flat, v^\flat) \) belongs to the right-hand side by the computation rule for \( \flat \). From right to left, we first destruct into \( x : \flat A \) and \( y : (\text{let } u^\flat := x \text{ in } \flat B(u)) \). Then we apply \( \flat \)-induction to \( x \) to get \( u :: A \), reducing the type of \( y \) to \( \flat B(u) \). Next, we apply \( \flat \)-induction to \( y \) to get \( v :: B(u) \), so we can form \( (u, v)^\flat : \flat \left( \sum_{x : A} B(x) \right) \). We leave it to the reader to check that these are inverses.

**Theorem 6.9.** For any \( A, B :: \text{Type} \), we have \( \flat(A \times B) \simeq \flat A \times \flat B \).
Proof. If $B$ is independent of $u : A$, then (let $u^♭ := x$ in $♭B) = ♭B$ (this can be regarded as an instance of the uniqueness principle for $♭$).

Theorem 6.10. For any $A,B,C ::$ Type with $f :: A \to C$ and $g :: B \to C$, we have $♭(A \times_C B) \simeq ♭A \times ♭C ♭B$.

Proof. We compute

\[
♭(A \times_C B) \equiv ♭\left(\sum_{x :: A} \sum_{y :: B} (f(x) = g(y))\right)
\]

\[
\simeq ∑_{x :: A} let u^♭ := x in ∑_{y :: B} let v^♭ := y in ♭(f(u) = g(v))
\]

\[
\simeq ∑_{x :: A} ∑_{y :: B} ♭(f(x)^♭) = ♭g(y)^♭
\]

\[
\simeq ∑_{x :: A} ∑_{y :: B} (♭f(x) = ♭g(y))
\]

\[
≡ ♭A \times ♭C ♭B.
\]

The first step is the definition of pullbacks and the second is Lemma 6.8. The third step combines several commuting conversions and the fourth is naturality of $(–)^♭$. The fifth step is Corollary 6.2 and the last is the definition of pullbacks.

For completeness, we should also record:

Theorem 6.11. $♭1 \simeq 1$.

Proof. It suffices to show that $♭1$ is contractible. Since we have $tt^♭ : ♭1$, it suffices to show that $♭1$ is a proposition. Now, if $u,v : ♭1$, we can destruct them as $x^♭$ and $y^♭$ for $x :: 1$ and $y :: 1$. Since $1$ is (crisply) contractible, we have $p :: x = y$, whence $p^♭ : ♭(x = y)$. Thus, by Theorem 6.1, we have $x^♭ = y^♭$.

6.2. Discrete types

We now move on to study discrete types, the duals of the codiscrete types from Section 3. The definition should come as no surprise.

Definition 6.12. A crisp type $A ::$ Type is discretete if $(–)^♭ : ⬤A \to A$ is an equivalence.

Remark 6.13. To say that a type is discrete is very different from saying that it is crisp. A discrete type is one whose points are ‘completely unconnected’ topologically, whereas a crisp type is one for which we can ignore the topology relating it to other types. Discreteness is an ‘intrinsic’ aspect of a type, whereas crispness is a more ‘syntactic’ one indicating what we are doing with that type at the moment. Roughly speaking, a type is discrete if we are free to assume that its elements are crisp.

At this point in our development, a type must be crisp before we can even ask whether it is discrete. However, in Section 8, we will introduce a more general notion of discreteness that removes this restriction.

Note that a (crisp and) discrete type $A ::$ Type might also be crisply discrete, i.e. we might have a $\text{crisp}$ witness $d :: \text{isdiscrete}(A)$. This distinction is admittedly somewhat
confusing. In practice, pretty much every discrete type is crisply discrete, and whenever we assume a type to be discrete it must be crisply discrete. Moreover, in Section 8, we will introduce a further axiom ensuring that every discrete type is crisply discrete.

Our first goal is to show that the discrete types form a coreflective subuniverse in an appropriate sense, with $\flat$ as the coreflector. The core of that result is that maps $\flat B \to A$ are equivalent to maps $\flat B \to \flat A$; this should appear sensible since any function out of a discrete space is continuous. We do, however, have to apply a further $\flat$ on both sides, since the type $\flat B \to A$ may inherit a non-trivial topology from $A$.

**Theorem 6.14.** For any $A, B :: \text{Type}$, postcomposition with $(-)_\flat : \flat A \to A$ induces an equivalence

$$\flat (\flat B \to \flat A) \simeq \flat (\flat B \to A).$$

**Proof.** To be precise, the left-to-right map in Equation (8) is defined as follows: given $h : \flat (\flat B \to \flat A)$, we first destruct it into $k :: \flat B \to \flat A$. Now since we have only crisp variables, it suffices to construct a function $\flat B \to A$, for which we have $\lambda(x : \flat B).k(x)_\flat$. In symbols, from $h : \flat (\flat B \to \flat A)$, we construct

$$\text{let } k^\flat := h \text{ in } (\lambda(x : \flat B).k(x))_{\flat} : \flat (\flat B \to A).$$

To define the right-to-left map in Equation (8), suppose given an element of $\flat (\flat B \to A)$, which we immediately destruct into $f :: \flat B \to A$. To construct an element of $\flat (\flat B \to \flat A)$, since we have only crisp variables it suffices to construct a function $\flat B \to \flat A$. Thus, let $x : \flat B$, which we immediately destruct into $u :: B$. But now again it suffices to construct an element of $A$ itself, for which we can take $f(u)$. In symbols, from $g : \flat (\flat B \to A)$, we construct

$$\text{let } f^\flat := g \text{ in } (\lambda(x : \flat B).\text{let } u^\flat := x \text{ in } f(u^\flat))_{\flat} : \flat (\flat B \to \flat A).$$

It is tempting to think that the uniqueness principle for $\flat$ (Lemma 4.4) should simplify ‘let $u^\flat := x \text{ in } f(u^\flat)$’ to $f(x)^\flat$. However, this rule does not apply because we cannot write $f(x)^\flat$, since $x$ is not a crisp variable.

In Figure 10, we verify that both round-trip composites are the identity. We use the fact that since $(-)_\flat$, is defined with let, the commuting conversions allow us to push it inside another let, along with function extensionality to allow this (and the uniqueness principle) to happen inside a $\lambda$-abstraction.

**Corollary 6.15.** If $B :: \text{Type}$ is crisply discrete, then for any $A :: \text{Type}$, postcomposition with $(-)_\flat : \flat A \to A$ induces an equivalence

$$\flat (B \to \flat A) \simeq \flat (B \to A).$$

**Proof.** As remarked after Corollary 6.3, the functor $\flat$ preserves equivalences, and precomposition with the equivalence $(-)_\flat : \flat B \simeq B$ is an equivalence on function types. Thus, this follows immediately from Theorem 6.14.
More generally, we have a dependent universal property:

**Theorem 6.16.** For \( B :: \text{Type} \) and \( A :: B \to \text{Type} \), fibrewise postcomposition with \((-)_{\flat} : \flat B \to \text{Type} \) induces an equivalence

\[
\flat \left( \prod_{x : \flat B} \left( \text{let } u^\flat := x \text{ in } \flat A(u^\flat) \right) \right) \simeq \left( \prod_{x : \flat B} A(x) \right).
\]

Therefore, if \( B :: \text{Type} \) is crisply discrete and \( A :: B \to \text{Type} \), fibrewise postcomposition with \((-)_{\flat} : \flat A(x) \to A(x) \) induces an equivalence

\[
\flat \left( \prod_{x : B} \left( \text{let } u^\flat := s(x) \text{ in } \flat A(u) \right) \right) \simeq \left( \prod_{x : B} A(x) \right),
\]

where \( s : B \to \flat B \) is the inverse of \((-)_{\flat} : \flat B \to B \).

**Proof.** The left-to-right map in Equation (10) is defined like that in Equation (8), taking \( h \) to

\[
\text{let } k^\flat := h \text{ in } (\lambda x : \flat B. \text{let } u^\flat := x \text{ in } k(u^\flat))^{\flat} : \flat \left( \prod_{x : \flat B} A(x) \right).
\]

Its inverse looks exactly the same as the inverse of Equation (8), though now all the functions are dependent:

\[
\text{let } f^\flat := g \text{ in } (\lambda x : \flat B. \text{let } u^\flat := x \text{ in } f(u^\flat))^{\flat} : \flat (\flat B \to \flat A).
\]
The calculations in Figure 10 generalize immediately, and Equation (11) is again obtained by transporting along the equivalence $\gamma B \simeq B$.

**Lemma 6.17.** For $A :: \text{Type}$ to be crisply discrete, it suffices that $(-)_{\gamma} : \gamma A \to A$ have a crisp section.

**Proof.** First, we should clarify what is meant by a ‘crisp section.’ The type of sections of $(-)_{\gamma}$ is $\sum_{(s : A \to \gamma A)} \prod_{(a : A)} s(a)_{\gamma} = a$. Having a crisp element of this type means having a crisp $s :: A \to \gamma A$ and also a crisp homotopy $H :: \prod_{(a : A)} s(a)_{\gamma} = a$.

It remains to show that $s(x_{\gamma}) = x$ for all $x : A$. By $\gamma$-induction, it suffices to show that for any $u :: A$, we have $s(u_{\gamma}) = u$. But $u_{\gamma} \equiv u$, so what we must show is $s(u) = u$. Now, $s(u) : \gamma A$ is crisp, so we also have $s(u) = s(u)_{\gamma}$; thus it suffices to show $s(u)_{\gamma} = u$. But now Corollary 6.3 tells us that this follows from $H(u) :: s(u)_{\gamma} = u$.

**Theorem 6.18.** For any $A :: \text{Type}$, the type $\gamma A$ is crisply discrete.

**Proof.** We define $s : \gamma A \to \gamma \gamma A$ by $s(x) : (\text{let } u^\beta := x \text{ in } u^{\beta \beta})$. Applying $(-)_{\gamma}$ to $s(x)$, we get

$$(\text{let } u^\beta := x \text{ in } u^{\beta \beta})_{\gamma} = (\text{let } u^\beta := x \text{ in } u^{\beta \beta}_{\gamma}) = (\text{let } u^\beta := x \text{ in } u^{\beta}) = x$$

using a commuting conversion and the uniqueness rule.

Thus, $\gamma$ coreflects into the discrete types, dually to how $\sharp$ reflects into the codiscrete types. However, there are several ways in which the duality breaks down. First, there is the outer $\gamma$ on the equivalence in Corollary 6.15.

Second, we have seen that $\sharp$ and $\gamma$ are both left exact, whereas a strict duality would interchange left exactness with right exactness. Note that exactness of reflectors and coreflectors can be a bit subtle and confusing. For instance, a reflector is a left adjoint, so it always preserves colimits, i.e. takes colimits in the ambient category to colimits in the reflective subcategory—but these latter colimits will only coincide with those in the ambient category if the subcategory is closed under them. This is not generally the case for $\sharp$: the coproduct of codiscrete spaces is not generally codiscrete.

Dually, a coreflector is a right adjoint, so it always takes limits in the ambient category to limits in the coreflective subcategory. The content of Theorems 6.9 to 6.11 is thus that the discrete types are closed under finite limits. We record these facts:

**Theorem 6.19.** If $A, B, C :: \text{Type}$ are crisply discrete and $f :: A \to C$ and $g :: B \to C$ and $x, y :: A$, then the following types are crisply discrete:

$A \times B \quad A \times C \quad B \quad (x = y)$

**Proof.** For $A \times B$, we have $A \times B \simeq \gamma A \times \gamma B \simeq \gamma (A \times B)$. The case of $A \times C$, $B$ is similar, and $1$ is exactly Theorem 6.11. For $x = y$, since equivalences lift to identity types, it suffices to show that if $x, y :: \gamma A$, then $(x = y)$ is discrete; but this follows from Theorem 6.18 and Corollary 6.2.
More generally, we have:

**Theorem 6.20.** Suppose $A :: \text{Type}$ is crisply discrete, and that $B :: A \to \text{Type}$ is such that for every $x : \flat A$, we have (let $u^\flat := x$ in isdiscrete$(B(u))$). Then, $\sum_{(x:A)} B(x)$ is crisply discrete.

The hypothesis is morally ‘$B(u)$ is discrete for every $u :: A$,’ but stating it that way would require this theorem to be only a ‘derivable rule’ (see Remark 4.6), since we cannot say ‘for every $u :: A$’ in an ordinary hypothesis.

**Proof.** Using Lemma 6.8, we have

$$\flat \left( \sum_{(x:A)} B(x) \right) \simeq \left( \sum_{(x:\flat A)} \left( \text{let } u^\flat := x \text{ in } \flat B(u) \right) \right)$$

$$\simeq \sum_{(x:\flat A)} B(x)$$

$$\simeq \sum_{(u:A)} B(u).$$

On the other hand, the dual of left exactness of $\sharp$ would be right exactness of $\flat$, which is true. In fact, as remarked in Section 5, this is essentially the content of the crisp induction principles. We record these facts:

**Theorem 6.21.** If $A, B :: \text{Type}$ are crisply discrete and $f, g :: A \to B$ and $n :: \mathbb{N}$, then the following types are crisply discrete:

$$A + B \quad \emptyset \quad \mathbb{N} \quad S^1 \quad \text{coeq}(f, g) \quad \parallel A \parallel_n.$$

Combining the last two, set-coequalizers also preserve crisp discreteness.

6.3. **Adjointness of $\flat$ and $\sharp$**

To complete our study of $\flat$, we consider how it interacts with $\sharp$. Our first observation is that $\flat$ and $\sharp$ ‘eat each other.’

**Theorem 6.22.** For any $A :: \text{Type}$, there are natural equivalences

$$\# \flat A \simeq \# A \quad \text{and} \quad \flat A \simeq \flat \# A.$$

This is sensible according to our topological intuition: $\flat$ and $\sharp$ are supposed to re-equip a space with the discrete or codiscrete topology, respectively, so it should not matter whether we have already modified its topology in the other way.

**Proof.** The first equivalence is defined by

$$\lambda(y : \# \flat A). y_{\flat \sharp} \quad \text{and} \quad \lambda(x : \# A). x_{\sharp \flat}.$$

The round-trip composite starting with $x : \# A$ yields

$$x_{\sharp \flat} \equiv x_{\flat \sharp} \equiv x,$$

while that starting with $y : \# \flat A$ yields

$$y_{\flat \sharp} \equiv y_{\sharp \flat} \equiv y.$$
The second equivalence is defined by
\[ \lambda(x : \flat A). \text{let } u^\flat := x \text{ in } u^\sharp \] and
\[ \lambda(y : \sharp \sharp A). \text{let } v^\sharp := y \text{ in } v^\flat. \]
The round-trip composite starting with \( x : \flat A \) yields
\[
\begin{align*}
\text{let } v^\flat &:= (\text{let } u^\flat := x \text{ in } u^\sharp) \text{ in } v^\sharp \\
&= \text{let } u^\flat := x \text{ in } (\text{let } v^\flat := u^\flat \text{ in } v^\sharp) \\
&\equiv \text{let } u^\flat := x \text{ in } u^\sharp \\
&\equiv \text{let } u^\flat := x \text{ in } u \\
&= u.
\end{align*}
\]
And the round-trip composite starting with \( y : \sharp \sharp A \) yields
\[
\begin{align*}
\text{let } u^\flat &:= (\text{let } v^\flat := y \text{ in } v^\sharp) \text{ in } u^\sharp \\
&= \text{let } v^\flat := y \text{ in } (\text{let } u^\flat := v^\flat \text{ in } u^\sharp) \\
&\equiv \text{let } v^\flat := y \text{ in } v^\sharp \\
&\equiv \text{let } v^\flat := y \text{ in } v \\
&= v.
\end{align*}
\]

**Corollary 6.23.** For any crisply discrete \( A :: \text{Type} \), the map \( A \to \flat \sharp A \) is an equivalence. Dually, for any crisply codiscrete \( A :: \text{Type} \), the map \( \sharp \flat A \to A \) is an equivalence.

We can express this more internally using the type \( \text{Codisc} := \sum_{(A :: \text{Type})} \text{iscodiscrete}(A) \) from Theorem 3.11 and an analogous universe of discrete types, defined as
\[
\text{Disc} := \sum_{A :: \text{Type}} (\text{let } B^\flat := A \text{ in } \text{isdiscrete}(B)).
\]

**Corollary \{UA\} 6.24.** \( \flat \text{Codisc} \simeq \sharp \text{Codisc} \); hence, also \( \sharp \text{Disc} \simeq \text{Codisc} \).

**Proof.** By Lemma 6.8, we have
\[
\flat \text{Codisc} \simeq \sum_{A :: \text{Type}} (\text{let } B^\flat := A \text{ in } \text{iscodiscrete}(B)).
\]
The equivalence \( \text{Disc} \simeq \flat \text{Codisc} \) now comes from the endomaps of \( \flat \text{Type} \) induced by \( \flat \) and \( \sharp \). These are mutually inverse by Corollary 6.23 and Theorem 6.1 and univalence.

Since \( \sharp \) and \( \flat \) preserve \( n \)-types, this equivalence relativizes to the universes of \( n \)-types. In particular, we have \( \text{DiscProp} \simeq \flat \text{CodiscProp} \), which uses only propositional univalence.

Furthermore, this equivalence respects all type formers, suitably reflected or coreflected. For instance, since discrete and codiscrete types are closed under cartesian products, we have product operations \( \text{Disc} \to \text{Disc} \to \text{Disc} \) and \( \text{Codisc} \to \text{Codisc} \to \text{Codisc} \), and these commute with the equivalence of Corollary 6.24. Discrete objects need not be closed under function types, but we have an operation \( \text{Disc} \to \text{Disc} \to \text{Disc} \) defined by
\[
\lambda A. \lambda B. \text{let } C^\flat := A \text{ in } \text{let } D^\flat := B \text{ in } \flat (C \to D),
\]
which agrees with $\lambda A. \lambda B. (A \to B) : \text{Codisc} \to \text{Codisc} \to \text{Codisc}$ under the equivalence of Corollary 6.24; and so on.

Topological intuition also suggests that since all functions between discrete or codiscrete spaces are continuous, both $\flat A \to \flat B$ and $\# A \to \# B$ ought to be the set of all (discontinuous) functions from $A$ to $B$. These two spaces may themselves have different topologies, of course (and in fact they do), but if we make them both discrete or codiscrete we should be able to identify them.

**Theorem 6.25.** For any $A, B :: \text{Type}$, there is an equivalence

$$\flat(\flat A \to \flat B) \simeq \flat(\# A \to \# B).$$

(12)

**Proof.** On the one hand, we have the functorial action of $\#$:

$$\# : (\flat A \to \flat B) \to (\# \flat A \to \# \flat B).$$

We can compose the result with the equivalences $\# \flat A \simeq \# A$ and $\# \flat B \simeq \# B$ on either side, and apply then the functor $\flat$, to obtain a map from left to right in Equation (12). On the other hand, as remarked above, the functorial action of $\flat$ can be expressed (in this case) as

$$\flat : \flat(\# A \to \# B) \to \flat(\# A \to \# B),$$

and we can again compose on either side with the equivalences $\flat \flat A \simeq \flat A$ and $\flat \flat B \simeq \flat B$.

More explicitly, given $f : \flat(\flat A \to \flat B)$, we can write

$$\text{let } h^\flat := f \text{ in } (\lambda (x : \# A). h(x_{\flat^\flat B})_{\flat^\flat B}) : \flat(\# A \to \# B),$$

and given $g : \flat(\# A \to \# B)$, we can write

$$\text{let } k^\flat := g \text{ in } (\lambda (x : \flat A). k(x_{\flat^\flat B})_{\flat^\flat B}) : \flat(\# A \to \# B).$$

Omitting the commuting conversions and reductions that eliminate the variables $h$ and $k$, as in the proof of Theorem 6.14, the computations that show these to be inverses are

$$(\text{let } u^\flat := x_{\flat^\flat B} \text{ in } k(x_{\flat^\flat B})_{\flat^\flat B} \equiv k(x_{\flat^\flat B})_{\flat^\flat B} \equiv k(x_{\flat^\flat B})_{\flat^\flat B} \equiv k(x))$$

$$(\text{let } u^\flat := x \text{ in } h(u_{\flat^\flat B})_{\flat^\flat B} \equiv (\text{let } u^\flat := x \text{ in } h(u^\flat)) = h(x).$$

**Corollary 6.26.** For any $A, B :: \text{Type}$, there is a natural equivalence

$$\flat(\# A \to B) \simeq \flat(A \to \# B).$$

In other words, $\flat$ is ‘crisply left adjoint’ to $\#$.

**Proof.** We have the following chain of equivalences:

$$\flat(\# A \to B) \simeq \flat(\# A \to B) \simeq \flat(\# A \to \# B) \simeq \flat(A \to \# B).$$

In fact, more generally, we can say:

**Theorem 6.27.** For any $A :: \text{Type}$ and $B :: A \to \text{Type}$, there is a natural equivalence

$$\flat \left( \prod_{(u : \# A)} B(u) \right) \simeq \flat \left( \prod_{(x : A)} \# B(x) \right).$$
**Proof.** Given \( f : \blacklastrightarrow \left( \prod_{(u : A)} B(u) \right) \), we define \( g : \blacklastrightarrow \left( \prod_{(x : A)} \sharp B(x) \right) \) by

\[
\text{let } h^\blacklastrightarrow := f \text{ in } (\lambda (x : A). h(x^\blacklastrightarrow)^\sharp)^\blacklastrightarrow
\]

and given \( g \), we define \( f \) by

\[
\text{let } k^\blacklastrightarrow := g \text{ in } (\lambda (u : \blacklastrightarrow A). \text{let } x^\blacklastrightarrow := u \text{ in } k(x^\sharp))^\blacklastrightarrow.
\]

Omitting the outer commuting conversions and reductions again, the computations that show these to be inverses are

\[
(\text{let } x^\blacklastrightarrow := u \text{ in } h(x^\blacklastrightarrow)^\sharp)^\blacklastrightarrow \equiv (\text{let } x^\blacklastrightarrow := u \text{ in } h^\blacklastrightarrow) \equiv h(u)
\]

and

\[
(\text{let } x^\blacklastrightarrow := x^\blacklastrightarrow \text{ in } k(x)^\sharp)^\blacklastrightarrow \equiv k(x)^\sharp \equiv k(x).
\]

Hence, we have both a reflective subcategory (the codiscrete types) and a coreflective subcategory (the discrete types), which by Corollary 6.23 are abstractly equivalent via the reflection and coreflection. Moreover, Theorem 6.22 tells us that modulo this equivalence, the reflector \( \sharp \) agrees with the coreflector \( \blacklastrightarrow \), which is sensible if we consider that both take the same underlying set (or \( \infty \)-groupoid) and equip it with a new topology.

**Remark 6.28.** We now have another formulation of the law of excluded middle. When assumed as an axiom, Lemma 3.13 is a crisp element of \( \prod_{(P : \mathsf{Prop})} \sharp (P \lor \neg P) \), and hence yields an element of \( \blacklastrightarrow \left( \prod_{(P : \mathsf{Prop})} \sharp (P \lor \neg P) \right) \). Thus, by Theorem 6.27, we can obtain a (crisp) element of

\[
\prod_{P : \mathsf{Prop}} (P^\blacklastrightarrow \lor \neg P^\blacklastrightarrow),
\]

a statement which we might call the flat law of excluded middle.

**Remark 6.29.** The overall conclusion of Sections 4–6 can be summarized as ‘\( \blacklastrightarrow \) is a coreflector, and the images of \( \blacklastrightarrow \) and \( \blacktriangleright \) are equivalent by an equivalence that identifies \( \blacklastrightarrow \) with \( \blacktriangle \).’ But in contrast to the analogous statement about \( \blacktriangle \) in Remark 3.12, this cannot be stated purely cohesively, because of the no-go Theorem 4.1. In Schreiber and Shulman (2012), using ordinary type theory without crispness, we stated this axiomatically with \( \blacklastrightarrow \) as an operation on \( \sharp \mathsf{Type} \); here, we instead derive it from the rules governing \( \blacklastrightarrow \) and \( \blacktriangle \) in relation to crisp hypotheses.

6.4. The axiom of choice revisited

Let us now return to the sharp axiom of choice, Axiom AC. Restated, this says that for any set \( A \) and type family \( P : A \to \mathsf{Type} \), we have

\[
\left( \prod_{x : A} \sharp \parallel P(x) \parallel \right) \to \parallel \prod_{x : A} \sharp P(x) \parallel.
\]

Using discreteness, we can derive from this a statement less encumbered by \( \blacktriangleright \)s.
**Theorem [AC] 6.30 (The discrete axiom of choice).** Suppose \( A : \text{Type} \) is a crisply discrete set, that \( P : A \to \text{Type} \) is also crisp, and that \( \prod_{(x:A)} \|P(x)\| \) crisply. Then \( \|\prod_{(x:A)} P(x)\| \).

**Proof.** The hypothesis \( \prod_{(x:A)} \|P(x)\| \) is stronger than that of the sharp axiom of choice, so we can apply the latter, obtaining \( \#\|\prod_{(x:A)} \#P(x)\| \). Since all the parameters are crisp, we can apply \((-)_\sharp\) to get \( \|\prod_{(x:A)} \#P(x)\| \); whereas what we want is \( \|\prod_{(x:A)} P(x)\| \).

By functoriality, we can remove the \( \|\cdot\| \) from both. Now by Theorem 6.27, we get \( \prod_{\mid u\mid A} P(u) \); but \( A \) is discrete, so this gives \( \prod_{(x:A)} P(x) \) as desired. \( \square \)

**Corollary [AC] 6.31 (The crisp countable axiom of choice).** Suppose \( P : \mathbb{N} \to \text{Type} \) is crisp and that \( \prod_{(n:\mathbb{N})} \|P(n)\| \) crisply. Then, \( \|\prod_{(n:\mathbb{N})} P(n)\| \).

We will refer to this statement as Axiom AC\(_{\mathbb{N}}\). As mentioned previously, later on we will assume an axiom ensuring that all propositions are discrete; thus, the crispness of \( \prod_{(n:\mathbb{N})} \|P(n)\| \) will become automatic. However, the crispness of \( P \) is a real restriction that makes this significantly weaker than the ordinary countable axiom of choice.

For example, let \( \mathbb{R}_C \) denote the Cauchy real numbers and \( \mathbb{R} \) the Dedekind real numbers.\(^{11}\) There is an injection \( i : \mathbb{R}_C \to \mathbb{R} \) which is not constructively an isomorphism. It is well-known that ordinary countable choice implies that it is an isomorphism, i.e. that every Dedekind real is a Cauchy real. But with our restricted Axiom AC\(_{\mathbb{N}}\) (or Axiom LEM), all we can get is the following.

**Theorem [AC\(_{\mathbb{N}}\) or LEM] 6.32.** Any crisp Dedekind real \( x : \mathbb{R} \) is a Cauchy real.

**Proof.** By Univalent Foundations Program (2013, Lemma 11.4.1), a Dedekind real \( x : \mathbb{R} \) is a Cauchy real iff there exists \( c : \prod_{(q,r:Q)} (q < r) \to (q < x) + (x < r) \).

First, suppose Axiom LEM. Then since \( x \) is crisp, and \( Q \) and \( (q < r) \) are discrete, for any \( q,r \) with \( q < r \), we have \( (q < x) + \neg (q < x) \). This implies \( (q < x) + (x < r) \) as in Univalent Foundations Program (2013, Corollary 11.4.3).

Now, suppose Axiom AC\(_{\mathbb{N}}\). Then, since \( x \) is crisp, so is the family \( P : S \to \text{Type} \) defined in Univalent Foundations Program (2013, Corollary 11.4.3) as

\[
P(q,r) := \sum_{b:2} \left( b = 0 \to q < x \right) \wedge \left( b = 1 \to x < r \right),
\]

where \( S := \{(q,r) : Q \times Q \mid q < r\} \) is equivalent to \( \mathbb{N} \). Thus, we can get a choice function and extract \( c \) in the same way. \( \square \)

**Corollary [AC\(_{\mathbb{N}}\) or LEM] 6.33.** The induced maps \( \|\mathbb{R}_C\| \to \|\mathbb{R}\| \) and \( \#\mathbb{R}_C \to \#\mathbb{R} \) are equivalences.

\(^{11}\) For constructive definitions of \( \mathbb{R}_C \) and \( \mathbb{R} \), see e.g. Univalent Foundations Program (2013, Chapter 11) or Johnstone (2002, Section D4.7). At the moment, by the ‘Cauchy reals,’ we mean a simple quotient of the set of Cauchy sequences or approximations; in Corollary 8.27, we will show that under suitable axioms, these coincide with the fancier Cauchy reals of Univalent Foundations Program (2013, Section 11.3).
Proof. By Theorem 6.22, the two statements are equivalent, so it suffices to prove the second. By Corollary 3.18, it will suffice to show $\prod_{(x : \mathbb{R})} \neg \text{lib}(x)$. But by Theorem 6.27, this follows from Theorem 6.32.

In other words, $\mathbb{R}_C$ and $\mathbb{R}$ have the same points, but perhaps different topologies. It is natural to wonder what those topologies are! We will return to this in Sections 8 and 11.

Another important question to ask is what their points are. From the perspective of the cohesive part of the type theory, this is a meaningless question: their points are, by definition, the real numbers (Cauchy or Dedekind, respectively). However, recall from Section 3 that we can also construct ‘the set of real numbers’ entirely in the codiscrete world, yielding the type $\mathbb{R}'$ shown in Figure 4 and reproduced in Figure 11. Thus, we can ask whether the set of points of the space of real numbers $\mathbb{R}$ is the same as this ‘codiscrete set of real numbers,’ i.e. is ‘the space of real numbers’ $\mathbb{R}$ really a ‘topology’ on the set of real numbers? The answer is yes; this internalizes Johnstone (2002, C3.6.11).

**Theorem 6.34.** $\mathfrak{h}\mathbb{R}' \simeq \mathfrak{h}\mathbb{R}$, and hence $\mathbb{R}' \simeq \mathbb{R}$.

**Proof.** First of all, notice that since $\text{CodiscProp}$ is codiscrete, we have

$$(\#Q \to \text{CodiscProp}) \simeq (Q \to \text{CodiscProp}).$$

Thus, in $\mathbb{R}'$, we can sum over $L, U : Q \to \text{CodiscProp}$ instead; we write $L', U'$ for their unique extensions to $\#Q$.

Next, by the dependent version of Theorem 3.6, the $\prod_{(q : \#Q)}$'s in $\mathbb{R}'$ can be replaced by $\prod_{(q : Q)}$, with $q$ replaced by $q^\sharp$ inside. Now we can use the facts that $L'(q^\sharp) = L(q)$, $U'(q^\sharp) = U(q)$ and $(q^\sharp < r^\sharp) = \#(q < r)$ (since the inequalities on $\#Q$ extend those of $Q$), and since $\#(q < r)$ is the domain of a function with codiscrete codomain, we can replace it by $(q < r)$. Thus, we obtain the second version shown in Figure 11.

Now note that for $q : \#Q$, we have

$$L'(q) = \#L'(q) = \#L(q^\sharp) = \#L(q^\sharp),$$

and similarly for $R'$. Likewise, for $q : \#Q$ and $r : Q$, we have

$$(q < r^\sharp) = \#(q < r^\sharp) = \#(q^\sharp < r^\sharp) = \#(q < r).$$

Thus, using Lemma 3.8 (and, in the second and third lines, the fact that $\#$ preserves products), together with the fact that $\#\|A\| \simeq \#\|A\|$ (since both are a reflection into codiscrete propositions), we obtain the third version.

Now, since $Q$ is crisply discrete, we have

$$\mathfrak{h}(Q \to \text{CodiscProp}) \simeq \mathfrak{h}(Q \to \mathfrak{h}\text{CodiscProp}) \simeq \mathfrak{h}(Q \to \text{DiscProp})$$

using Corollary 6.24 in the second step. Thus, with Lemma 6.8, we can convert $\mathfrak{h}\mathbb{R}'$ into a sum over $M, N : \mathfrak{h}(Q \to \text{DiscProp})$, obtaining the first expression for $\mathfrak{h}\mathbb{R}'$ in Figure 11. (Since $\mathfrak{h}$ preserves products, we have also distributed it over the $\times$s.)

However, as remarked after Corollary 6.24, the equivalence $\text{Disc} \simeq \mathfrak{h}\text{Codisc}$ respects all suitably (co)reflected type formers. Since discrete types are closed under truncations and contain $\emptyset$, we deduce the second expression for $\mathfrak{h}\mathbb{R}'$ in Figure 11. Here, we abuse
\[ R' := \sum_{(L,U:Q \to \text{CodiscProp})} \#\left( \left\| \sum_{(q:Q)} L(q) \right\| \times \left\| \sum_{(r:Q)} U(r) \right\| \right) \]
\[ \times \left( \prod_{(q:Q)} L(q) \leftrightarrow \#\left( \sum_{(r:Q)} (q < r) \times L(r) \right) \right) \]
\[ \times \left( \prod_{(r:Q)} U(r) \leftrightarrow \#\left( \sum_{(q:Q)} U(q) \times (q < r) \right) \right) \]
\[ \times \left( \prod_{(q:Q)} (L(q) \times U(q)) \to \#\emptyset \right) \times \left( \prod_{(q,r:Q)} (q < r) \to \#(L(q) + U(r)) \right) \]
\[ \simeq \sum_{(L,U:Q \to \text{CodiscProp})} \#\left( \left\| \sum_{(q:Q)} L'(q) \right\| \times \left\| \sum_{(r:Q)} U'(r) \right\| \right) \]
\[ \times \left( \prod_{(q:Q)} L(q) \leftrightarrow \#\left( \sum_{(r:Q)} (q < r) \times L'(r) \right) \right) \]
\[ \times \left( \prod_{(r:Q)} U(r) \leftrightarrow \#\left( \sum_{(q:Q)} U'(q) \times (q < r) \right) \right) \]
\[ \times \left( \prod_{(q:Q)} (L(q) \times U(q)) \to \#\emptyset \right) \times \left( \prod_{(q,r:Q)} (q < r) \to \#(L(q) + U(r)) \right) \]
\[ \simeq \sum_{(L,U:Q \to \text{CodiscProp})} \#\left( \left\| \sum_{(q:Q)} L(q) \right\| \times \left\| \sum_{(r:Q)} U(r) \right\| \right) \]
\[ \times \left( \prod_{(q:Q)} L(q) \leftrightarrow \#\left( \sum_{(r:Q)} (q < r) \times L(r) \right) \right) \]
\[ \times \left( \prod_{(r:Q)} U(r) \leftrightarrow \#\left( \sum_{(q:Q)} U(q) \times (q < r) \right) \right) \]
\[ \times \left( \prod_{(q:Q)} (L(q) \times U(q)) \to \#\emptyset \right) \times \left( \prod_{(q,r:Q)} (q < r) \to \#(L(q) + U(r)) \right) \]

\[ b R' \simeq \sum_{(M,N:\emptyset \to \text{DiscProp})} \text{let } L^b := M \text{ in let } U^b := N \text{ in} \]
\[ b \#\left( \left\| \sum_{(q:Q)} L^b(q) \right\| \times \left\| \sum_{(r:Q)} U^b(r) \right\| \right) \]
\[ \times b \left( \prod_{(q:Q)} L^b(q) \leftrightarrow \#\left( \sum_{(r:Q)} (q < r) \times U^b(q) \right) \right) \]
\[ \times b \left( \prod_{(r:Q)} U^b(r) \leftrightarrow \#\left( \sum_{(q:Q)} U^b(q) \times (q < r) \right) \right) \]
\[ \times b \left( \prod_{(q:Q)} (L^b(q) \times U^b(q)) \to \#\emptyset \right) \times b \left( \prod_{(q,r:Q)} (q < r) \to \#(L^b(q) + U^b(r)) \right) \]
\[ \simeq \sum_{(M,N:\emptyset \to \text{DiscProp})} \text{let } L^b := M \text{ in let } U^b := N \text{ in} \]
\[ b \left( \left\| \sum_{(q:Q)} L^b(q) \right\| \times \left\| \sum_{(r:Q)} U^b(r) \right\| \right) \]
\[ \times b \left( \prod_{(q:Q)} L^b(q) \leftrightarrow \#\left( \sum_{(r:Q)} (q < r) \times L^b(r) \right) \right) \]
\[ \times b \left( \prod_{(r:Q)} U^b(r) \leftrightarrow \#\left( \sum_{(q:Q)} U^b(q) \times (q < r) \right) \right) \]
\[ \times b \left( \prod_{(q:Q)} (L^b(q) \times U^b(q)) \to \#\emptyset \right) \times b \left( \prod_{(q,r:Q)} (q < r) \to \#(L^b(q) + U^b(r)) \right) \]

Fig. 11. The Dedekind reals and their codiscrete version.
7. Topos models of spatial type theory

So far we have described a spatial type theory with crisp variables and two modalities \( \flat \) and \( \sharp \), and also (in Sections 2.1, 3.1, 3.2 and 6.4) some classical axioms that can be added to this theory without (we expect) destroying the intended topological content. However, nothing we have said excludes the possibility that all types are both discrete and codiscrete, with \( \flat \) and \( \sharp \) being the identity. (In this case, our classical axioms would of course apply to all types, trivializing the topology.)

Thus, in order to ‘do some real topology,’ we need a new axiom that gives us a way to ‘access the topology’ internally in type theory. We could motivate such an axiom purely on first principles, but it seems appropriate at this point to instead discuss the various possible models of our theory.

So far, we have been referring mainly to classical topological spaces for intuition. This is fine, but topological spaces do not actually model our theory for several reasons. One reason is that they are not a topos, lacking universes and even a subobject classifier. Indeed, they are not even locally cartesian closed, so do not have \( \prod \)-types.

Another reason is that a classical topology is ‘mere structure’ on a set, whereas our theory demands something more. Consider, for instance, the following definition.

**Definition 7.1.** A type \( A \) is **concrete** if \((–)^\sharp : A \to \sharp A\) is a embedding (i.e. a \((-1\)-truncated map).

In classical topological spaces, the map from a space to its codiscrete reflection is always injective, so that all spaces would be concrete in this sense. However, for us, this is impossible.

**Theorem 7.2.** If \( \text{Prop} \) is concrete, then all propositions are codiscrete.

**Proof.** Suppose \((–)^\sharp : \text{Prop} \to \sharp \text{Prop}\) is an embedding. Then for any \( P, Q : \text{Prop} \), we have \((P = Q) \simeq (P^\sharp = Q^\sharp)\); but the latter is \(\sharp(P = Q)\) by Theorem 3.7. Thus, \( P = Q \) is codiscrete. But taking \( Q = 1 \), we have \((P = 1) = P\), so all propositions are codiscrete. \( \square \)

From a topological point of view, ‘all propositions are codiscrete’ would mean that all continuous injections are subspace embeddings. This seems quite dubious, and in fact, it trivializes almost the entire theory:

**Theorem 7.3.** For any \( n : \mathbb{N} \), an \( n \)-type \( A \) is codiscrete if and only if the map \( \text{const} : A \to (P \to A) \) is an equivalence for all propositions \( P \) such that \( \sharp P \). In particular, therefore, if all propositions are codiscrete, then all \( n \)-types are codiscrete.
Proof. The ‘only if’ direction follows immediately from the universal property of ⊤ (Theorem 3.6), since if \( P : \text{Prop} \) and \( \# P \), then \( (\# P \to A) \simeq A \). For the ‘if’ direction, we induct on \( n \). When \( n = -1 \), let \( A : \text{Prop} \) and suppose \( \text{const} : A \to (P \to A) \) is an equivalence for all \( P : \text{Prop} \) such that \( \# P \). To show \( A \) is codiscrete, we assume \( \# A \) and try to prove \( A \). But then \( A \) is a valid choice of \( P \), and of course \( A \to A \); hence \( A \).

Before embarking on the inductive step, we note that by Theorem 7.3, \( A \to \# A \) is an embedding (i.e. \( A \) is concrete) if and only if \( x = y \) is codiscrete for all \( x, y : A \). On the other hand, for \( x, y : A \) and \( P : \text{Prop} \), we have

\[
(\text{const}_x = \text{const}_y) \simeq (P \to (x = y)).
\]

Thus, \( \text{const} : A \to (P \to A) \) is an embedding if and only if

\[
\text{const} : (x = y) \to (P \to (x = y))
\]
is an equivalence. Therefore, if \( A \) is an \((n + 1)\)-type and \( \text{const} : A \to (P \to A) \) is an equivalence for all \( P : \text{Prop} \) such that \( \# P \), then the inductive hypothesis ensures that \( A \to \# A \) is an embedding; thus, it remains to show that it is surjective.

Let \( z : \# A \) and write \( \eta := (-)^\# : A \to \# A \). Then, \( \text{fib}_\eta(z) \) is a proposition. Moreover, since \( \# \) is left exact, we have \( \# \text{fib}_\eta(z) \simeq \text{fib}_{\# \eta}(z^\#) \). But \( \# \eta \) is an equivalence, so the latter type is contractible. Thus, \( \# \text{fib}_\eta(z) \), so by assumption \( \text{const} : A \to (\text{fib}_\eta(z) \to A) \) is an equivalence. But we have \( \text{pr}_1 : \text{fib}_\eta(z) \to A \), so there is an \( x : A \) such that \( \text{pr}_1 = \text{const}_x \).

We claim \( \eta(x) = z \). Since this is a codiscrete goal, and we have \( \# \text{fib}_\eta(z) \), by \#-induction we may assume that \( \text{fib}_\eta(z) \), i.e. there is a \( y : A \) and \( p : \eta(y) = z \). But \( \text{pr}_1 = \text{const}_x \) yields \( y = x \), and hence \( \eta(x) = z \) as desired.

Remark 7.4. Theorem 7.3 is a general fact about left exact monadic modalities, which generalizes the fact that a subtopos of a 1-topos is determined by its action on monomorphisms (see e.g. Johnstone 2002, A4.3.6). A subtopos of an \((\infty, 1)\)-topos, however, is not completely determined by its action on \( n \)-types for finite \( n \) (see Lurie 2009, Section 6.5).

Therefore, we are forced to think of our ‘topologies’ as some more contentful ‘stuff’ rather than just structure. A good example to think about is the topological topos of Johnstone (1979). Its objects are sets \( X \) equipped with, for every sequence \( x : \mathbb{N} \to X \) and point \( y \in X \), a collection of ‘witnesses that \( (x_n) \) converges to \( y \),’ equipped with natural operations (e.g. there is a specified ‘reflexivity’ witness that a constant sequence converges to the point at which it is constant, etc.) Such an object is codiscrete just when every sequence converges uniquely to every point. Thus, since monomorphisms must be injective on points and witnesses both, such an object is concrete just when there is at most one witness that any sequence converges to any point. But in general, a ‘space’ in this topos can have a sequence that converges to one point in many different ways.

The topological topos is a 1-topos rather than a higher topos, so we can expect it to model all of our type theory except for full univalence. Pending a solution to the general problem of higher topos-theoretic semantics, we may expect that there should be a corresponding ‘topological \((\infty, 1)\)-topos’ that models the whole theory with univalence.
More generally, the spatial type theory of Sections 2–6 should admit models in any local topos. A geometric morphism \( f : \mathcal{F} \to \mathcal{E} \) is called local (see e.g. Johnstone 2002, Section C3.6) if its direct image \( f_* \) admits a further \( \mathcal{E} \)-indexed right adjoint \( f^! \), which is then necessarily fully faithful (and so is \( f^* \)). In this case, \( \mathcal{F} \) inherits a comonad \( f^* f_* \) and a monad \( f_! f_* \) which are adjoint, and ought to extend the internal type theory of \( \mathcal{F} \) to model our \( \flat \) and \( \sharp \) (without univalence). The restricted classical axioms of Sections 2.1, 3.1, 3.2 and 6.4 will hold if their ordinary versions hold in \( \mathcal{E} \). Moreover, if we enhance \( f \) to a local morphism of \((\infty, 1)\)-toposes, then it should be possible to model homotopy type theory with (at least weak) univalent universes.

**Remark 7.5.** At present, the previous paragraph is only conjectural. What needs to be done to make it precise is (1) define an appropriate sort of ‘category with families’ or ‘contextual category’ to handle our two-context type theory with crisp variables and formulate \( \sharp \) and \( \flat \) as algebraic structure on such a gadget, (2) prove that syntax yields an initial one of these and (3) construct such algebraic objects from local geometric morphisms. Of these, (1) should be straightforward, (2) is still only conjectural even for most ordinary type theories (the one complete theorem along these lines is in Streicher 1991), while (3) should be straightforward in the 1-topos case and, I hope, possible in the \( \infty \)-case, at least if we only ask for weak universes. However, none of this is technically necessary for our actual results, which are simply theorems in our formal system; the categorical semantics is for motivation, for relative consistency, and, eventually, for applications to classical mathematics.

As shown by Johnstone (2002, C3.6.3(d)), local toposes are obtained as sheaves on sites with terminal objects admitting no non-trivial covers (local sites), and this was generalized to \((\infty, 1)\)-toposes by Schreiber (2013, Proposition 3.4.18). The topological topos arises in this way: its site has two objects 1 and \( \mathbb{N}_\infty \). More generally, we can consider sheaves on any small full subcategory of topological spaces that contains the one-point space. Any such topos will model spatial type theory with our classical axioms, and if we generalize to higher sheaves, it should have at least weak univalent universes.

**Remark 7.6.** A number of further examples of local geometric morphisms of 1-toposes can be found in Johnstone (2002, C3.6.3), all of which generalize appropriately to \((\infty, 1)\)-toposes. A ‘purely \( \infty \)-categorical’ example is the \((\infty, 1)\)-topos of parameterized spectra, whose objects are pairs \((X, E)\) where \( X \) is a space and \( E \) a spectrum parameterized over \( X \); this is equivalently the category of 1-excisive functors from pointed spaces to unpointed spaces studied in Goodwillie calculus (Goodwillie 2003; Joyal 2008; Lurie 2014). Here, the discrete and codiscrete objects coincide, being those for which \( E = 0 \), and moreover, we have \( \sharp = \flat \). Finally, the (co)discrete objects are also exactly the hypercomplete ones, and hence include all \( n \)-types for finite \( n \).

In particular, therefore, our ‘spatial type theory’ is in fact significantly more general than the name suggests. In the rest of this paper, we will add axioms that bring it closer to our intended intuition that ‘types have topology’; studying other classes of models would of course lead us to different axioms.
Remark 7.7. A different sort of internal logic for local geometric morphisms (along with another class of examples arising from realizability) is considered by Awodey and Birkedal (2003) and Awodey, Birkedal, and Scott (1999). It is a ‘logic over a type theory’ in which the types all belong to the base category but the propositions to the local topos, and there are two propositional modalities. This theory can be roughly identified with the fragment of spatial type theory in which we allow arbitrary crisp contexts (of sets, i.e. 0-truncated types), but require all types in the cohesive context to be propositions. (Since dependence on a proposition is always trivial, we can then assume that our cohesive propositions depend only on the crisp context, as is usual for logic-enriched type theories.) Unfortunately, the meaning of $\flat$ and $\sharp$ in *ibid.* is reversed from ours; but as we will see in Remark 9.21, there is a good reason for our choice.

As remarked in *ibid.*, when restricted to propositions, our $\flat$ (their $\sharp$) satisfies the formal properties of the operator $\square$ in S4 modal logic. Roughly speaking, this is the eventual origin of our term modality for $\flat$ and $\sharp$ (and, later on, $\dashv$).

One further remark is that the operations relating types to propositions in the logic of *ibid.* are not exactly those of spatial type theory. On the one hand, since $\Pi$ and $\Sigma$ cannot be applied to crisp dependence directly, we have to wrap them in $\flat$ first to yield the quantifiers $\exists$ and $\forall$ of *ibid.* Specifically, if $u :: A \vdash P : \text{Prop}$, then by ‘$\forall u, P$,’ we can only mean ‘$\prod_{(x : A)} \text{let } u^\flat := x \text{ in } P$,’ and similarly for $\exists$.

On the other hand, if $A$ is a set and $u :: A$ and $v :: A$, we can form the proposition $u = v$, but this is not what is meant by ‘$u = v$’ in *ibid.* since that is discrete, it must instead be $\flat(u = v)$ (or equivalently, by Theorem 6.1, $(u^\flat = v^\flat)$). That this has the correct universal property (i.e. satisfies ‘Lawvere’s law’ from Lawvere 1970) can be seen from crisp Id-induction (Theorem 5.6), which implies that if $x :: A, y :: A \vdash Q : \text{Prop}$ and $u :: A \vdash Q[u/x, u/y] \text{ true, then } x :: A, y :: A, p :: x = y \vdash Q \text{ true, and hence } x :: A, y :: A \vdash q : \flat(x = y) \vdash Q \text{ true by } \flat$-induction.

Returning to our goal of ‘accessing the topology’ internally in type theory, one natural approach would be to pick one local topos with a ‘topological’ character, or a class of them, and look for special features of that model that are visible internally. For Johnstone’s topological topos, this approach has been pursued by Escardó and Streicher (2016), in the following way. One can define internally the generic convergent sequence $\mathbb{N}_\infty$ to be the type of non-increasing functions $\mathbb{N} \to 2$; in the topological topos, this does in fact yield the space $\mathbb{N}_\infty$. Thus, one can define a convergent sequence in an arbitrary type $X$ to be a map $\mathbb{N}_\infty \to X$. If we wanted to combine this approach with spatial type theory, we could add axioms ensuring that the discrete and/or codiscrete types, as defined using $\flat$ and $\sharp$, can be characterized in terms of $\mathbb{N}_\infty$.†††

This is an interesting direction for future research, but in this paper, we take a different route, corresponding to a different choice of a small full subcategory of topological

†††Note that the notion of ‘indiscrete’ referred to in the title of Escardó and Streicher (2016) is weaker than our ‘codiscrete’; the former asks only that every sequence converges to every point in some way. Theorems 7.2 and 7.3 imply that the universe cannot be codiscrete in our stronger sense without largely trivializing the theory.
spaces. We are interested in applications to theorems about ‘manifold-like’ spaces locally modelled on the real numbers, both because of their intrinsic interest, because homotopy types presented by such spaces play an important role in classical algebraic topology, and because of their importance in applications (e.g. to physics). While manifold-like spaces are generally well-enough behaved that their topology is detected by convergent sequences, so that they embed fully faithfully into the topological topos, the generating object $\mathbb{N}_\infty$ of this topos is not at all manifold-like. In particular, it is not even locally connected, a property that is important for ensuring that a space has a well-behaved fundamental $\infty$-groupoid.

Thus, our motivating model will instead be the topos of sheaves (or the higher topos of higher sheaves) on the full subcategory of topological spaces whose objects are the cartesian spaces $\mathbb{R}^n$. This higher topos was studied by Schreiber (2013, Section 4.3) under the name Euclidean-topological cohesion, its objects being called Euclidean-topological $\infty$-groupoids or continuous $\infty$-groupoids; in other places, they are called topological stacks or topological $\infty$-stacks. In the same way that the topological topos defines ‘topology’ to consist of convergent sequences, this topos defines ‘topology’ to consist of continuous paths (and, more generally, homotopies and higher homotopies between such paths). In particular, just as the former allows spaces in which a sequence converges to a limit in many ways, our topos allows paths that ‘are continuous’ in many ways; the ‘concrete objects’ will be those in which any path is continuous in at most one way.‡‡‡

†††In general, when interpreted in a local topos of sheaves on a local site, our definition of ‘concrete’ reduces to the notion of ‘concrete sheaf’ studied by Dubuc (1979), Dubuc and Español (2006) and Baez and Hoffnung (2011). Its formulation using $\sharp$ was pointed out by Carchedi and appears in Schreiber (2013, Proposition 3.7.5).

‡‡ ‡

We emphasize again that these ‘paths’ are totally different from the terms of identity type (or ‘identifications’) that in homotopy type theory are sometimes called ‘paths.’ For instance, the topological circle $S^1$ contains many continuous paths, but no identifications other than reflexivity (inside type theory, it will be a ‘set’), whereas the homotopical circle $S^1$ contains $\mathbb{Z}$-many identifications but no non-trivial topology (in fact, we have seen already in Theorem 6.21 that it is topologically discrete). A general object of our (higher) topos is a sort of stack, with both topological paths and non-trivial identifications, and moreover, it can have ‘topology on its identifications.’ A good example is the delooping of a topological group such as $O(n)$; this is homotopically a 1-type, but its loop type is $O(n)$ with its topological structure.

Just as $\mathbb{N}_\infty$ plays a central role in the internal theory of the topological topos, we should expect the space of real numbers $\mathbb{R}$ to play an analogously central role for us. We would like to be able to define this space internally, just as $\mathbb{N}_\infty$ can be defined internally using decreasing binary sequences. As remarked in Section 6.4, there are at least two obvious ways to define ‘the real numbers’ internally, so we should inquire whether either of these definitions automatically inherits the correct topology.

The answer is that the Dedekind real numbers do inherit the correct topology, whereas the Cauchy real numbers do not. In fact, the Cauchy real numbers are discrete; we will prove this internally in Theorem 8.26. The correctness of the topology on the Dedekind
reals requires a more involved semantic argument (see Lin (2014) and Example 8.33), but in Theorem 11.5, we will prove a corresponding internal approximation. Both of these proofs use the axiom of real-cohesion to be introduced in the second part of the paper, which enhances our spatial type theory by asserting that the ‘topology’ of types is determined by maps out of the Dedekind real numbers.

Remark 7.8. Before continuing, let us note that having moved away from the idea of types as classical topological spaces, we also need to update our subsidiary intuitions. For example, we noted before that monomorphisms of topological spaces are continuous injections; but in topological toposes, the monomorphisms must be injective not only on points but on ‘topological structure’ (e.g. witnesses to convergence of a sequence or continuity of a path). Such a monomorphism is a subspace inclusion when it is also ‘full’ on topological structure (e.g. for any path lying in the subspace, the witnesses of its continuity in the subspace map isomorphically to the witnesses in the ambient space).

The case of (regular) epimorphisms (corresponding to type-theoretic surjections) is somewhat more subtle and involves the Grothendieck topology with which we equip our site. In Johnstone’s topological topos, a map \( f : A \to B \) is epimorphic if it is surjective on points and any (witness of a) convergent sequence \( (b_n) \to b_\infty \) in \( B \) has a ‘subsequence’ \( (b_{n_k}) \to b_\infty \) that is the image of some (witness of a) convergent sequence in \( A \). More importantly for us, in our topos of Euclidean-topological spaces, \( f : A \to B \) is epimorphic if for any continuous ‘plot’ \( \mathbb{R}^n \to B \), we can cover \( \mathbb{R}^n \) by open balls such that the restriction of our plot to any such ball is in the image of some plot in \( A \). If we think of \( A \) and \( B \) as a sort of manifold locally modelled on \( \mathbb{R}^n \), then the epimorphisms are the continuous maps that admit local sections.

8. Cohesion and real-cohesion

Recall that \( \mathbb{R} \) denotes the Dedekind real numbers (since it is much more important for us than any other kind of real numbers, we dignify it by omitting any subscript). We may call a map \( \mathbb{R} \to A \) a (continuous) path in \( A \). If topology is to be determined by such paths, then a discrete type should be one in which every such path is constant. This is what the following axiom expresses.

Axiom R♭. A crisp type \( A :: \text{Type} \) is discrete if and only if \( \text{const} : A \to (\mathbb{R} \to A) \) is an equivalence.

Remark 8.1. As stated, this ‘axiom’ has to be formulated as an unjustified rule rather than an element of some fixed type, like our original Axiom LEM. However, as we did for Axiom LEM in Remark 6.28, by using \( \flat \), we can reformulate it as an assumed (crisp) element of

\[
\prod_{B :: \text{Type}} \text{let } A^\flat := B \text{ in } (\text{isdiscrete}(A) \leftrightarrow \text{isequiv}(\text{const}_{A,\mathbb{R}})).
\]
8.1. Cohesion

When Axiom R♯ is added to spatial type theory, we call it *real-cohesive type theory* (or real-cohesive homotopy type theory, if homotopical features such as univalence and HITs are included). We now explore the consequences of Axiom R♯ in stages, by introducing a sequence of weaker axioms that use successively more properties of ℝ. In addition to clarifying the exposition, this makes it clear that many results are true much more generally. By *cohesive (homotopy) type theory*, we mean spatial type theory with one or more of these weaker axioms.

We begin with the following, the weakest of the axioms of cohesion.

**Axiom C0.** There is a type family \( R : I \to \text{Type} \) such that a crisp type \( A : \text{Type} \) is discrete if and only if \( \text{const} : A \to (R_i \to A) \) is an equivalence for all \( i : I \).

Since this characterization of discreteness is purely internal, we finally have a notion of ‘discrete’ that applies also to non-crisp types.

**Definition {C0} 8.2.** An arbitrary type \( A : \text{Type} \) (not necessarily crisp) is said to be **discrete** if \( \text{const} : A \to (R_i \to A) \) is an equivalence for all \( i : I \).

**Lemma {C0} 8.3.** Discrete types are an exponential ideal, and even a ‘dependent exponential ideal’: if \( B : A \to \text{Type} \) and each \( B(x) \) is discrete, then \( \prod(x:A) B(x) \) is discrete.

**Proof.**

\[
\left( \prod_{i} (R_i \to A) \right) \simeq \left( \prod_{i} (R_i \to B(x)) \right) \simeq \left( \prod_{i} B(x) \right).
\]

Technically, we also need to verify that this equivalence is the map const, but this is usually easy, so we omit it, both here and in the following lemmas.

**Lemma {C0} 8.4.** Discrete types are closed under \( \sum \): if \( A \) is discrete and \( B : A \to \text{Type} \) has each \( B(x) \) discrete, then \( \sum_{(x:A)} B(x) \) is discrete.

**Proof.** We have \( \left( R_i \to \sum_{(x:A)} B(x) \right) \simeq \left( \sum_{(f:R_i \to A)} \prod_{(r:R_i)} B(f(r)) \right) \). But since \( A \to (R_i \to A) \) is an equivalence, the latter is equivalent to \( \sum_{(x:A)} (R_i \to B(x)) \), which is equivalent to \( \sum_{(x:A)} B(x) \) since each \( B(x) \) is discrete.

**Lemma {C0} 8.5.** If \( A \) is discrete and \( x, y : A \), then \( (x = y) \) is discrete.

**Proof.** Since \( \text{const} : A \to (R_i \to A) \) is an equivalence, it induces an equivalence on identity types:

\[
(x = y) \simeq (\text{const}_x = \text{const}_y).
\]

However, \( (\text{const}_x = \text{const}_y) \) is equivalent to \( R_i \to (x = y) \).

**Lemma {C0} 8.6.** Discrete types are closed under pullbacks.

**Proof.** If \( A, B, C \) are discrete and \( f : A \to C, g : B \to C \), then

\[
A \times_C B = \sum_{(x:C)} \sum_{(y:B)} (f(x) = g(y)),
\]

so this follows from Lemmas 8.4 and 8.5.
8.2. Punctual cohesion

Our next batch of consequences uses the additional assumption that each $R_i$ is inhabited (which is clearly true for $R$).

**Axiom C1.** Axiom C0 holds, and moreover, we have some $r : \prod_{i : I} R_i$.

This makes Definition 8.2 equivalent to a weaker looking definition.

**Lemma \{c1\} 8.7.** A type $A$ is discrete if and only if every function $R_i \to A$ is constant, i.e. for all $i : I$ and $f : R_i \to A$, there is an $a : A$ such that $f(x) = a$ for all $x : R_i$.

**Proof.** The given condition says that each $\text{const} : A \to (R_i \to A)$ has a section; but it always has a retraction, namely evaluation at $r_i$. Thus, having a section is equivalent to being an equivalence.

The primary application of Axiom C1 for us is the following fact.

**Lemma \{c1\} 8.8.** All propositions are discrete.

**Proof.** Since each $R_i$ is inhabited, if $A$ is a proposition and $R_i \to A$, then $A$ is contractible; hence, the condition of Lemma 8.7 holds.

From the $\♭\;\♯$ adjunction, we obtain a partial dual:

**Lemma \{c1\} 8.9.** All crisp propositions are codiscrete.

**Proof.** We are to show that for any $P :: \text{Prop}$, $P$ is codiscrete, i.e. that $\♯P \to P$ holds. Since $P$ and $\♯P$ are both discrete by Lemma 8.8, it will suffice to prove $\♭\♯P \to \♭P$. But by Theorem 6.22, $\♭\♯P \simeq \♭P$.

For instance, this implies that we can dispense with the distinction between ‘crisp and discrete’ and ‘crisply discrete.’

**Lemma \{c1\} 8.10.** If $A :: \text{Type}$ is crisp and discrete, then we may assume it is crisply discrete.

**Proof.** Suppose $A :: \text{Type}$ and that we have some $d : \text{isdiscrete}(A)$. Since $\text{isdiscrete}(A)$ is a proposition, it is discrete, and so we might as well have $d : \♭(\text{isdiscrete}(A))$. Thus, by $\♭$-induction, no matter our goal, we are free to assume $d' :: \text{isdiscrete}(A)$.

Similarly, if a crisp function is an equivalence, it is automatically a crisp equivalence; if a crisp type is an $n$-type, it is crisply an $n$-type; and so on.

As another particular case of Lemma 8.9, we have:

**Theorem \{c1\} 8.11.** $\emptyset$ is codiscrete, i.e. Axiom $\not\emptyset \not\emptyset$ holds.

**Proof.** $\emptyset$ is a crisp proposition.

**Corollary \{LEM.C1\} 8.12.** For any proposition $P$, we have $\not P \simeq \neg\neg P$.

**Proof.** This is just Theorem 3.15, but with Axiom $\not\emptyset \not\emptyset$ replaced by Axiom C1, which implies it (by Theorem 8.11).
In particular, even though Axiom C1 only characterizes discreteness explicitly, in
the presence of Axiom LEM, it implies an even more explicit characterization of
codiscreteness, at least for $n$-types.

**Corollary** \{LEM,C1\} \textbf{8.13}. An $n$-type $A$ is codiscrete if and only if $\text{const} : A \to (P \to A)$ is
an equivalence for all propositions $P$ such that $\neg \neg P$.

**Proof.** Combine Corollary 8.12 and Theorem 7.3. \hfill \Box

**Remark 8.14.** In topos-theoretic language, Axiom $\#\emptyset$ says that the subtopos defined by
$\#$ is dense, while the stronger Lemma 8.9 says that $\#$ is in fact fibrewise dense in the sense
of Johnstone (2002, C1.1.22).

We also obtain a characterization of discrete sets (i.e. 0-types) that makes no reference
to the family $R$, and coincides with that of Penon (1985) and Dubuc and Penon (1986):
they are the sets with decidable equality.

**Lemma** \{LEM,C1\} \textbf{8.15}. A set $A$ is discrete if and only if $\forall x, y : A. (x = y \lor x \neq y)$.

**Proof.** If $A$ is discrete, then given $x, y : A$, we may assume they are crisp. Then since $A$
is a set, $x = y$ is a crisp proposition, hence $x = y \lor x \neq y$ follows from Axiom LEM.

Conversely, suppose $\forall x, y : A. (x = y \lor x \neq y)$. Since $x = y$ and $x \neq y$ are incompatible
propositions, we have $((x = y) \lor (x \neq y)) \simeq ((x = y) + (x \neq y))$, so we can define functions
by cases on equality in $A$. Now for any $f : R_i \to A$, define $g : R_i \to 2$ by

$$
g(r) = \begin{cases} 1_2 & \text{if } f(r) = f(r_i) \\ 0_2 & \text{if } f(r) \neq f(r_i). \end{cases}$$

Since $2$ is discrete, $g$ is constant. But $g(r_i) = 1_2$, so $g(r) = 1_2$ for all $r : R_i$, i.e. $f(r) = f(r_i)$
for all $r$. Hence, $f$ is constant. \hfill \Box

We record a few more useful consequences. First, $\flat$ detects emptiness:

**Corollary** \{C1\} \textbf{8.16}. If $A :: \text{Type}$ and $\neg \flat A$, then $\neg A$.

**Proof.** If $\neg \flat A$, then it may as well hold crisply by Lemma 8.8, i.e. we have $f :: \flat A \to \emptyset$.
Then by Corollary 6.26, we also have $A \to \#\emptyset$, and hence $\neg A$ since $\#\emptyset = \emptyset$. \hfill \Box

Next, any injective continuous function with discrete codomain also has discrete domain,
as we would expect from classical topology.

**Lemma** \{C1\} \textbf{8.17}. Any subobject of a discrete type is discrete.

**Proof.** If $A$ is discrete and $P : A \to \text{Prop}$ is a predicate, then each $P(x)$ is discrete by
Lemma 8.8; hence, the subobject $\sum_{(x : A)} P(x)$ is discrete by Lemma 8.4. \hfill \Box
Corollary \((C1)\) 8.18. If \(m : A \rightarrow B\) is a crisp embedding, then its naturality square for \(\mathfrak{h}\) is a pullback:

\[
\begin{array}{ccc}
\mathfrak{h}A & \rightarrow & A \\
\downarrow & \ & \downarrow \\
\mathfrak{h}B & \rightarrow & B \\
\end{array}
\]

Proof. Since \(\mathfrak{h}\) preserves embeddings, \(\mathfrak{h}A \rightarrow \mathfrak{h}B\) is an embedding, as is the pullback of \(A\) to \(\mathfrak{h}B\). Thus, it will suffice to show that the latter pullback factors through \(\mathfrak{h}A\) as a subobject of \(\mathfrak{h}B\). But since it is a subobject of a discrete type, it is discrete, so this follows from the universal property of \(\mathfrak{h}A\).

We can also characterize the concrete objects (Definition 7.1) more exactly.

Corollary \{(LEM,C1)\} 8.19. A set \(A\) is concrete if and only if it is \(\neg\neg\) -separated, i.e. \(\neg\neg(x = y) \rightarrow (x = y)\) for all \(x, y : A\).

Proof. By definition, \(A\) is concrete if \((-)^{\sharp} : A \rightarrow \sharp A\) is an embedding. But by Theorem 3.7, \((x^{\sharp} = y^{\sharp}) \simeq \sharp(x = y)\), so this holds just when \(x = y\) is codiscrete for all \(x, y : A\). Now apply Corollary 8.12.

For example, since \(\mathbb{R}\) is always \(\neg\neg\) -separated (see, for example, Johnstone 2002, D4.7.6), it follows that \(\mathbb{R}\) is concrete. But in fact, this is true even without LEM.

Theorem \{(C1)\} 8.20. \(\mathbb{R}\) is concrete.

Proof. It will suffice to prove that for any crisp \(A :: \text{Type} \) and \(f, g :: A \rightarrow \mathbb{R}\), if the two composites \(A \rightarrow \mathbb{R} \rightarrow \sharp\mathbb{R}\) are equal, then so are \(f\) and \(g\). (Applying this when \(A = \mathbb{R} \times \sharp\mathbb{R}\) will then show that \(\mathbb{R} \rightarrow \sharp\mathbb{R}\) is monic.) Moreover, by Corollary 6.26, saying that the two composites \(A \rightarrow \mathbb{R} \rightarrow \sharp\mathbb{R}\) are equal is the same as saying that the two composites \(\mathfrak{h}A \rightarrow A \rightarrow \mathbb{R}\) are equal.

Now, a map \(f : A \rightarrow \mathbb{R}\) is determined by two \(\mathbb{Q}\)-indexed families of subobjects of \(A\), say \(L^f(q)\) and \(U^f(q)\) for \(q : \mathbb{Q}\), satisfying the usual axioms, and likewise for \(g\). Thus, our assumption says that \(L^f(q)\) and \(U^f(q)\) agree with \(L^\sharp(q)\) and \(U^\sharp(q)\) when pulled back to \(\mathfrak{h}A\), and we want to show that \(L^f(q) = L^\sharp(q)\) and \(U^f(q) = U^\sharp(q)\) already over \(A\). By Corollary 8.18, the pullback of \(L^f(q)\) to \(\mathfrak{h}A\) is \(\mathfrak{h}L^f(q)\), and likewise for all the others. (Note that \(\mathbb{Q}\), being abstractly isomorphic to \(\mathbb{N}\), is discrete, so we can assume any rational number to be crisp).

Now if \(q < r\), we have \(L^f(q) \cap U^f(r) = \emptyset\), and thus \(\mathfrak{h}L^f(q) \cap \mathfrak{h}U^f(r) = \emptyset\). But \(\mathfrak{h}U^f(r) = \mathfrak{h}U^\sharp(r)\) by assumption, so \(\mathfrak{h}L^f(q) \cap \mathfrak{h}U^\sharp(r) = \emptyset\) as well. Since \(\mathfrak{h}\) preserves pullbacks, by Corollary 8.16, we have \(L^f(q) \cap U^\sharp(r) = \emptyset\). If further \(r < s\), then since \(U^\sharp(r) \cup L^\sharp(s) = A\), we have \(L^f(q) \subseteq L^\sharp(s)\). Finally, for any \(p : \mathbb{Q}\), we have \(L^f(p) = \bigcup_{q < p} L^f(q) \subseteq \bigcup_{q < p} L^\sharp(p) = L^\sharp(p)\) (by taking \(s \equiv p\), and \(r\) between \(q\) and \(p\)). The same argument applies in the other direction and to all the other subsets.

We can also show the inclusion of the discrete retopologization is injective (indeed, bijective) on points—but with a truncation restriction.
Theorem (c1) 8.21. If $B :: \text{Type}$ is a crisp set, then $\♭B \rightarrow B$ is injective.

Proof. Note that $\♭B$ is a set by Theorem 6.6. In the proof of Corollary 6.2, we showed that for any $u, v : \♭B$, we have $(u = v) \simeq \text{code}(u, v)$, where

$$\text{code}(u, v) \equiv \text{let } x^♭ := u \text{ in } (\text{let } y^♭ := v \text{ in } \♭(x = y)).$$

However, if $B$ is a set, then $x = y$ is a proposition, hence discrete by Lemma 8.8. Thus, by $\♭$-induction we can prove that $\text{code}(u, v)$ is equivalent to

$$\text{let } x^♭ := u \text{ in } (\text{let } y^♭ := v \text{ in } (x = y)),$$

which by a couple of commuting conversions is equivalent to

$$(\text{let } x^♭ := u \text{ in } x) = (\text{let } y^♭ := v \text{ in } y),$$

i.e. to $(u_♭ = v_♭)$. Thus, $(-)_♭$ is injective.

The proof also makes clear why we should not expect $\♭B \rightarrow B$ to be an embedding if $B$ is not a set: in that case, we also have to discretify the identity types.

Corollary (c1) 8.22. If $B :: \text{Type}$ is a crisp set, then the composite $\♭B \rightarrow B \rightarrow \#B$ is injective.

Proof. It is equal to the composite $\♭B \cong \♭\#B \rightarrow \#B$, and $\#B$ is also a crisp set.

Corollary (c1) 8.23. Any crisp discrete set is concrete.

We can also show that discrete sets are closed under surjective quotients (see also Johnstone 2002, A4.6.6).

Lemma (c1) 8.24. If $A, B :: \text{Type}$ are crisp, $A$ is discrete, $B$ is a set, and $f :: A \rightarrow B$ is surjective, then $B$ is discrete.

Proof. The kernel pair of $f$ is a subobject of $A \times A$, and is therefore crisply discrete by Lemma 8.17. But $B$ is the set-coequalizer of that kernel pair, hence discrete by Theorem 6.21.

Corollary (c1) 8.25. If $A$ is crisp and discrete, then so is $\|A\|_0$.

Now we can prove the following, which was claimed in Section 7 to hold in our topos model.

Theorem (c1) 8.26. The Cauchy real numbers $\mathbb{R}_C$ are discrete.

Proof. Since $\mathbb{Q}$ and $\mathbb{N}$ are discrete, by Lemma 8.3, the type $\mathbb{N} \rightarrow \mathbb{Q}$ of sequences of rational numbers is discrete. The type $\mathcal{C}$ of Cauchy sequences is a subobject of $\mathbb{N} \rightarrow \mathbb{Q}$, so by Lemma 8.17, it is also discrete. Finally, the Cauchy reals are a surjective image of the set of Cauchy sequences, so by Lemma 8.24, they are also discrete.

Theorem 8.26 is about the usual sort of Cauchy real numbers defined as a quotient of the set of Cauchy sequences. Univalent Foundations Program (2013, Section 11.3) constructs a ‘better’ set of Cauchy real numbers that is constructively Cauchy complete.
However, our classicality axioms suffice to ensure that the usual $\mathbb{R}_C$ is already Cauchy complete, hence coincides with that of Univalent Foundations Program (2013):

**Corollary** \{AC,C1\} 8.27. $\mathbb{R}_C$ is Cauchy-complete.

*Sketch of proof.* Since $\mathbb{R}_C$ is discrete by Theorem 8.26 and the quotient map $C \to \mathbb{R}_C$ is surjective, by Theorem 6.30, there exists a section of it. Thus, for any Cauchy sequence in $\mathbb{R}_C$, there exists a sequence of sequences of rational numbers, and ‘diagonalizing’ this in the usual way we obtain a limit in $\mathbb{R}_C$.

**Corollary** \{AC$_N$ or LEM, C1\} 8.28. $\mathbb{R}_C \simeq \flat \mathbb{R}$.

*Proof.* By Corollary 6.33, $\flat \mathbb{R}_C \simeq \flat \mathbb{R}$. But $\mathbb{R}_C$ is discrete, so $\flat \mathbb{R}_C \simeq \mathbb{R}_C$.

Finally, we can also nail down the exact relationship between the topology on the two types of real numbers.

**Corollary** \{AC,C1\} 8.29. The limited principle of omniscience (LPO) holds: for any $f : \mathbb{N} \to 2$, either there exists an $n : \mathbb{N}$ such that $f(n) = 1_2$, or $f(n) = 0_2$ for all $n$.

*Proof.* Since $\mathbb{N} \to 2$ is discrete by Lemma 8.3, we may assume $f :: \mathbb{N} \to 2$ is crisp. Then ‘$\exists n.f(n) = 1_2$’ is a crisp proposition, so by Axiom LEM we have either $\exists n.f(n) = 1_2$ or $\neg(\exists n.f(n) = 1_2)$. But $\neg(\exists n.f(n) = 1_2)$ is the same as $(\forall n.f(n) = 0_2)$, so we are done.

In particular, this implies that if we define $\mathbb{N}_\infty$ as in Escardó and Streicher (2016) to be the type of non-increasing binary sequences $\mathbb{N} \to 2$, then the canonical map $\mathbb{N} + 1 \to \mathbb{N}_\infty$ is an equivalence. In particular, *every* type in our theory is ‘indiscrete’ in the sense of Escardó and Streicher (2016). Thus, the ‘topology’ of the types in cohesive type theory is very different from the ‘topology’ studied by Escardó and Streicher (2016).

It is well-known in constructive mathematics that LPO implies the following weaker classicality principles.

**Corollary** \{LEM,C1\} 8.30. The lesser limited principle of omniscience (LLPO) holds: for any $f, g : \mathbb{N} \to 2$, if it is not the case that both $\exists n.f(n) = 1_2$ and $\exists n.g(n) = 1_2$, then either $\forall n.f(n) = 0_2$ or $\forall n.g(n) = 0_2$.

**Corollary** \{LEM,C1\} 8.31. Markov’s principle (MP) holds: if $f : \mathbb{N} \to 2$ and $\neg \forall n.f(n) = 0_2$, then $\exists n.f(n) = 1_2$.  

8.3. *Omniscience principles*

We have seen that the ordinary LEM and AC are inconsistent with spatial interpretations, motivating the introduction of $\sharp$ and $\flat$ in order to state our modified Axiom LEM and Axiom AC. However, there are weaker ‘classicality’ principles whose ordinary versions *are* true in our motivating model, and some of them can even be proven from our current axioms, such as the following.

**Theorem** \{LEM,C1\} 8.29. The limited principle of omniscience (LPO) holds: for any $f : \mathbb{N} \to 2$, either there exists an $n : \mathbb{N}$ such that $f(n) = 1_2$, or $f(n) = 0_2$ for all $n$.

*Proof.* Since $\mathbb{N} \to 2$ is discrete by Lemma 8.3, we may assume $f :: \mathbb{N} \to 2$ is crisp. Then ‘$\exists n.f(n) = 1_2$’ is a crisp proposition, so by Axiom LEM we have either $\exists n.f(n) = 1_2$ or $\neg(\exists n.f(n) = 1_2)$. But $\neg(\exists n.f(n) = 1_2)$ is the same as $(\forall n.f(n) = 0_2)$, so we are done.

In particular, this implies that if we define $\mathbb{N}_\infty$ as in Escardó and Streicher (2016) to be the type of non-increasing binary sequences $\mathbb{N} \to 2$, then the canonical map $\mathbb{N} + 1 \to \mathbb{N}_\infty$ is an equivalence. In particular, *every* type in our theory is ‘indiscrete’ in the sense of Escardó and Streicher (2016). Thus, the ‘topology’ of the types in cohesive type theory is very different from the ‘topology’ studied by Escardó and Streicher (2016).

It is well-known in constructive mathematics that LPO implies the following weaker classicality principles.

**Corollary** \{LEM,C1\} 8.30. The lesser limited principle of omniscience (LLPO) holds: for any $f, g : \mathbb{N} \to 2$, if it is not the case that both $\exists n.f(n) = 1_2$ and $\exists n.g(n) = 1_2$, then either $\forall n.f(n) = 0_2$ or $\forall n.g(n) = 0_2$.

**Corollary** \{LEM,C1\} 8.31. Markov’s principle (MP) holds: if $f : \mathbb{N} \to 2$ and $\neg \forall n.f(n) = 0_2$, then $\exists n.f(n) = 1_2$.  

It is also well-known that LPO, LLPO and MP are equivalent to statements about the order and equality of Cauchy real numbers. Specifically:

1. LPO is equivalent to saying that $\mathbb{R}_C$ has decidable equality. (Thus, Theorem 8.29 and Lemma 8.15 give an alternative proof of Theorem 8.26.)
2. LLPO is equivalent to $\forall x : \mathbb{R}_C. (x \leq 0 \lor x \geq 0)$, hence that $\leq$ is a total order on $\mathbb{R}_C$.
3. MP is equivalent to $\forall x : \mathbb{R}_C. (\neg (x \leq 0) \rightarrow x > 0)$.

Traditionally in constructive mathematics, two real numbers $x, y$ are said to be apart, written $x # y$, if $|x - y| > 0$. We have $\neg (x # y) \leftrightarrow (x = y)$, but in general, $x # y$ is stronger than $\neg (x = y)$. However, since $x = y$ is equivalent to $|x - y| \leq 0$, MP implies that $x # y$ and $\neg (x = y)$ coincide for Cauchy reals.

Of course, we are generally more interested in the Dedekind reals $\mathbb{R}$ than the Cauchy ones $\mathbb{R}_C$. Toby Bartels has suggested the following terminology:

1. The analytic LPO claims that $\mathbb{R}$ has decidable equality.
2. The analytic LLPO claims that $\forall x : \mathbb{R}. (x \leq 0 \lor x \geq 0)$.
3. The analytic MP claims that $\forall x : \mathbb{R}. (\neg (x \leq 0) \rightarrow x > 0)$.

Since (assuming Axiom $\mathbb{R}^\flat$) $\mathbb{R}$ is not discrete (its identity map is not constant), the analytic LPO is false. Somewhat more surprisingly, we will prove in Theorem 11.7 that the analytic LLPO is also false. The analytic MP, however, is actually true in our motivating model! I conjecture that it can even be proven from Axiom LEM and Axiom $\mathbb{R}^\flat$, but I have not managed to show this yet. Thus, we assume it as an additional axiom. Actually, I prefer the following axiom, which is equivalent in the presence of Axiom LEM.

**Axiom T.** For any $x : \mathbb{R}$, the proposition $x > 0$ is codiscrete.

**Theorem [LEM.C1.T] 8.32 (Analytic Markov’s Principle).** If $x, y : \mathbb{R}$ satisfy $\neg (x = y)$, then $x # y$. In particular, if $\neg (x \leq 0)$, then $x > 0$.

**Proof.** By Axiom T, $x # y$ is codiscrete, so by Corollary 8.12 it is $\neg \neg$-stable. Thus, we can prove it by contradiction. However, $\neg (x # y)$ means $x = y$, which contradicts the assumption of $\neg (x = y)$. \(\square\)

Axiom T is arguably the most mysterious part of the theory. Topologically, it says that the open subset $(0, \infty)$ of $\mathbb{R}$ is a subspace, i.e. has the induced topology. In other words, it relates the intrinsic ‘topology’ of $\mathbb{R}$, arising from its definition as a type, to the internal ordering relation defined on it. Another way to look at it is that it ensures that the intrinsic ‘topology’ of $\mathbb{R}$ makes it into a topological field, i.e. that the reciprocal is continuous on the subspace of invertible elements.

Some considerations of models may also help to understand Axiom T.

**Example 8.33.** Axiom T holds in our motivating model of sheaves on the category $\mathcal{C}$ of cartesian spaces $\mathbb{R}^n$.

**Sketch of proof.** This requires recalling a bit about the standard proof that $\mathbb{R}$ inherits the correct topology in this model, which goes as follows (see Mac Lane and Moerdijk 1994, Section VI.9; Lin 2014). For any space $X \in \mathcal{C}$, we have $\text{Sh}(\mathcal{C})/X \simeq \text{Sh}(\mathcal{C}/X)$,
which admits a local geometric morphism to $\text{Sh}(X)$. Since local geometric morphisms are orthogonal to grouplike morphisms (Johnstone 2002, C3.6.10), and the space of real numbers is a grouplike locale, to give a real number in $\text{Sh}(C)/X$ is equivalent to giving a real number in $\text{Sh}(X)$, or equivalently a map $X \to \mathbb{R}$. Thus, the sections of the Dedekind real number object over $X$ are the continuous maps $X \to \mathbb{R}$.

Repeating this argument with $\mathbb{R}$ replaced by the space of positive real numbers, which is also grouplike, we find that the object of positive reals in $\text{Sh}(C)$ is the sheaf of positive real-valued functions. To say that this is a codiscrete subobject of the Dedekind realals is to say that a continuous real-valued function factors through the space of positive real numbers just when it is positive at every point, which is obviously true.  

The preceding argument depends crucially on the classicality of the base topos, and in particular, the fact that the locale of formal real numbers (which classifies internal Dedekind real numbers) is spatial. Indeed, we have:

**Example 8.34.** If we start from an arbitrary base topos, and construct a similar topos of sheaves on the category of cartesian locales $R_f^n$ (where $R_f$ denotes the locale of formal reals), then Axiom T can fail.

**Sketch of proof.** I am indebted to Bas Spitters for this argument. There exist toposes (particularly, recursive ones such as the effective topos) in which there are uniformly continuous functions $f : \mathbb{R} \to \mathbb{R}$ such that $f(x) > 0$ for all $x$ but $f$ does not have uniform lower bounds on all finite intervals. Now since the metric space $\mathbb{R}$ is ‘locally compact’ in the sense of Bishop, by Palmgren (2007a), $f$ extends to a locale morphism $\tilde{f} : R_f \to R_f$, and since $\mathbb{R}$ is the space of points of $R_f$, we still have $\tilde{f}(x) > 0$ for all $x$. However, $\tilde{f}$ does not factor through the locale of formal positive reals, since by Palmgren (2007b) if it did then it would have local uniform lower bounds. Thus, we can repeat the proof of Example 8.33 until the last sentence, at which point we find that the ‘obvious’ fact is now false.

Note that the topos of Example 8.34 is local over its base and satisfies Axiom $R^\flat$, by the same argument as before. Thus, (modulo Remark 7.5) Axiom T does not follow from Axiom $R^\flat$ alone; but I do not know a countermodel to it that satisfies both Axiom $R^\flat$ and Axiom LEM.

9. **Shape**

Codiscrete types are defined by the fact that functions into them need not be continuous; while discrete types are defined by the fact that functions out of them need not be continuous. In a sense, this means that all the non-tautological information about codiscrete types is carried by the maps out of them, and likewise the non-trivial information about discrete types is carried by the maps into them. Let us leave the first for another day and concentrate on maps $A \to B$, where $B$ is discrete.

---

88 But see Theorem 10.2 and Corollary 10.3.
In the world of classical topological spaces, if $B$ is discrete, then (at least when $A$ is well-behaved, which at the moment means ‘locally connected’) continuous maps $A \to B$ are the same as functions $\pi_0(A) \to B$, where $\pi_0(A)$ is the set of connected components of $A$. In other words, $\pi_0$ is a reflection into the subcategory of discrete spaces. Since $\pi_0(A)$ is also the ‘first layer’ of the homotopy type, or fundamental $\infty$-groupoid, of $A$, this suggests that mapping into discrete spaces can carry information about the latter construction.

In fact, it turns out that the single generalization from sets to $\infty$-groupoids ensures that mapping into discrete types carries all the information about the fundamental $\infty$-groupoid. At least in good situations, such as our $\infty$-topos of sheaves on the category of cartesian spaces $\mathbb{R}^n$, the fundamental $\infty$-groupoid can be defined as a left adjoint to the inclusion of discrete spaces (see Schreiber 2013, Proposition 4.3.32; Carchedi 2016, Section 3). With Axiom R♭ in our theory, we can get an inkling of how this works internally by considering the topological circle $S^1$.

**Definition 9.1.** The topological circle $S^1$ is the (homotopy) coequalizer of the pair of maps

$$
\begin{array}{ccc}
\mathbb{R} & \xrightarrow{id} & \mathbb{R} \\
+1 & \xrightarrow{f} & \mathbb{R}
\end{array}
$$

We might write this as $S^1 := \mathbb{R}/\mathbb{Z}$. In Theorem 9.19, we will compare this definition to some other possible definitions of the topological circle. For now, let us simply observe that it is at least one reasonable definition—with one caveat, namely that $S^1$ ought to be a set (i.e. have no higher identifications), and with this definition, it is not obvious that this is so. We could, of course, 0-truncate it, but that would defeat the purpose of what we are about to do, and fortunately it turns out to be unnecessary:

**Theorem (UA) 9.2.** Suppose $R$ is a set and $f : R \simeq R$ an equivalence such that for all $r : R$ and $m,n : \mathbb{Z}$, if $f^m(r) = f^n(r)$, then $m = n$. (In other words, the induced action of $\mathbb{Z}$ on $R$ is free.) Then, the homotopy coequalizer of

$$
\begin{array}{ccc}
R & \xrightarrow{id} & R \\
+1 & \xrightarrow{f} & R
\end{array}
$$

is also a set.

**Proof.** Recall that $S^1$ denotes the homotopical circle, an HIT with two constructors $\text{base} : S^1$ and $\text{loop} : \text{base} = \text{base}$. We define $P : S^1 \to \text{Type}$ by $P(\text{base}) : \simeq R$ and $P(\text{loop}) = f$ (modulo univalence). Note that $S^1$ is the homotopy coequalizer of $1 \simeq 1$. Thus, by the flattening lemma (Univalent Foundations Program 2013, Section 6.12), the desired homotopy coequalizer of $id_R$ and $f$ is equivalent to $\sum_{(x:S^1)} P(x)$; so it will suffice to prove that the latter is a set.

Let $x,y : S^1$ and $u : P(x), v : P(y)$; we must prove that $(x,u) = (y,v)$ is a proposition. Now, $(x,u) = (y,v)$ is equivalent to $\sum_{(p : x = y)} (p \cdot u = v)$. Thus, suppose $p,q : x = y$ and $r : p \cdot u = v$ and $s : q \cdot u = v$; we must show $(p,r) = (q,s)$. Since $r$ and $s$ are equalities in a set, it suffices to show $p = q$. But this is also an equality in a set (since $S^1$ is a 1-type), hence a proposition; thus, since $S^1$ is connected, we may assume $x = \text{base}$ and $y = \text{base}$.

Now under the isomorphism $(\text{base} = \text{base}) \simeq \mathbb{Z}$, our $p$ and $q$ get identified with $n,m : \mathbb{Z}$, respectively. Moreover, by definition of $P$, transporting along $p$ or $q$ gets identified with
application of \(f^n\) or \(f^m\), respectively. Thus, from \(r\) and \(s\), we obtain \(p \cdot u = q \cdot u\) and hence \(f^n(u) = f^n(u)\), with \(u : R\), and so the freeness assumption gives \(n = m\) as needed.

**Remark 9.3.** It should be possible to prove Theorem 9.2 using univalence only for propositions (e.g. with Univalent Foundations Program 2013, Theorem 7.2.2). But the preceding proof is easier.

**Corollary \{UA\} 9.4.** The topological circle \(S^1\) is a set.

**Proof.** Since \(\mathbb{R}\) is an abelian group, \((+1) : \mathbb{R} \to \mathbb{R}\) is an equivalence. The freeness condition in Theorem 9.2 is equivalent to saying that the unique ring homomorphism \(\mathbb{Z} \to \mathbb{R}\) is injective, which just says that \(\mathbb{R}\) has characteristic 0.

The following proof contains the crucial idea involved in the adjoint characterization of fundamental \(\infty\)-groupoids.

**Theorem \{R\} 9.5.** For any discrete type \(X\), we have \((S^1 \to X) \cong (S^1 \to X)\).

**Proof.** The universal property of \(S^1\) means that \((S^1 \to X)\) is equivalent to

\[
\sum_{x : X} (x = x). \tag{14}
\]

But by the universal property of a coequalizer, \((S^1 \to X)\) is equivalent to

\[
\sum_{g : \mathbb{R} \to X} (g \circ (+1) = g). \tag{15}
\]

Now, \(g \circ (+1) = g\) is an equality in the type \(\mathbb{R} \to X\), which is equivalent to \(X\) since \(X\) is discrete. The equivalence can be implemented by evaluating at any \(r : \mathbb{R}\), such as 0; thus Equation (15) is equivalent to

\[
\sum_{g : \mathbb{R} \to X} (g(1) = g(0)). \tag{16}
\]

But since \((\mathbb{R} \to X) \cong X\), we may assume \(g\) is constant at some \(x : X\), giving Equation (14).

In particular, the identity map of \(S^1\) corresponds to a non-trivial map \(S^1 \to S^1\), which may be said informally to ‘wrap the topological circle around the homotopical one.’ Since Theorem 9.5 is easily shown to be natural, this map exhibits \(S^1\) as a reflection of \(S^1\) into the discrete types. (Recall from Theorem 6.21 that \(S^1\) is topologically discrete.)

Since \(S^1\) is, or should be, the fundamental \(\infty\)-groupoid of \(S^1\), this leads us to ask: does every type have a reflection into the discrete types? The answer is yes, and it requires only the much weaker Axiom C0.

**Definition \{C0\} 9.6.** For any type \(A\), its shape \(\mathcal{f}A\) is the higher inductive type with the following five constructors:

1. \(\sigma_A : A \to \mathcal{f}A\).
2. \(\kappa_A : \prod_{i : I} (R_i \to \mathcal{f}A) \to \mathcal{f}A\).
3. For all \(i : I\) and \(g : R_i \to \mathcal{f}A\) and \(x : R_i\), an equality \(g(x) = \kappa_A(i, g)\).
4. \( \kappa'_A : \prod_{i : I}(R_i \to \mathcal{f}A) \to \mathcal{f}A \).
5. For all \( i : I \) and \( x : \mathcal{f}A \), an equality \( x = \kappa'_A(i, \text{const}_x) \).

**Remark 9.7.** We use the word ‘shape’ not only because it is nine syllables shorter than ‘fundamental \( \infty \)-groupoid,’ but because the latter has various connotations that we want to be free of. In particular, the shape is a purely internal construction in our ‘real-cohesive type theory,’ although it generally behaves like the fundamental \( \infty \)-groupoid. Moreover, the fundamental \( \infty \)-groupoid is usually notated \( \Pi_{\infty} \), but since \( \Pi \) is rather overworked in type theory already, a different symbol is preferable. The word ‘shape’ comes from shape theory, which also studies generalizations of the fundamental \( \infty \)-groupoid that make sense for less well-behaved spaces. The symbol \( \mathcal{f} \) is not an integral sign (\( \int \)) but an ‘esh,’ the IPA sign for a voiceless postalveolar fricative (English sh).

The definition of \( \mathcal{f}A \) is cooked up precisely to admit a map from \( A \) and to be discrete:

**Lemma 9.8.** \( \mathcal{f}A \) is discrete.

**Proof.** The last four constructors say exactly that \( \mathcal{f}A \to (R_i \to \mathcal{f}A) \) has both a left and a right inverse.

It is unsurprising, therefore, that \( \mathcal{f}A \) has the desired universal property. We express this first as a \( \mathcal{f} \)-induction principle.

**Theorem 9.9.** If \( P : \mathcal{f}A \to \text{Type} \) is a family of discrete types, and we have \( d : \prod_{(x : A)} P(\sigma(x)) \), then we have \( f : \prod_{(y : \mathcal{f}A)} P(y) \) such that \( f(\sigma(x)) = d(x) \) for all \( x : A \).

**Proof.** The ‘basic’ induction principle of \( \mathcal{f}A \) arising from its higher inductive definition says that given \( Q : \mathcal{f}A \to \text{Type} \) together with

1. \( d_\sigma : \prod_{(x : A)} Q(\sigma(x)) \);
2. \( d_\kappa : \prod_{(i : I)} \prod_{(g : R_i \to \mathcal{f}A)} \prod_{(h : [\kappa_{\mathcal{f}A}])} Q(\kappa(i, g)) \);
3. \( d_\varsigma : \prod_{(g : R_i \to \mathcal{f}A)} \prod_{(h : [\kappa_{\mathcal{f}A}])} Q(g(x)) \prod_{(x : A)} (h(x) = _\varsigma d_\varsigma(i, g, h)) \), where \( \varsigma \) is the third constructor of \( \mathcal{f}A \);
4. \( d_{\kappa'} : \prod_{(i : I)} \prod_{(g : R_i \to \mathcal{f}A)} \prod_{(h : [\kappa_{\mathcal{f}A}])} Q(\kappa'(i, g)) \);
5. \( d_{\kappa''} : \prod_{(i : I)} \prod_{(y : \mathcal{f}A)} \prod_{(x : \mathcal{f}A)} (y = _{\kappa''} d_{\kappa''}(i, \text{const}_i, \text{const}_y)) \), where \( \kappa'' \) is the last constructor of \( \mathcal{f}A \).

We have \( f : \prod_{(y : \mathcal{f}A)} P(y) \) such that \( f(\sigma(x)) \equiv d_\sigma(x) \) (plus four other equalities). Thus, given our current hypotheses, it remains to construct \( d_\varsigma, d_{\kappa''}, d_{\kappa'}, \) and \( d_{\kappa'} \).

First, suppose \( g : R_i \to \mathcal{f}A \) and \( h : \prod_{(x : R_i)} Q(g(x)) \), for some \( i \). By the third constructor of \( \mathcal{f}A \), we have \( g(x) = \kappa(i, g) \) for all \( x : R_i \). Thus, transporting \( h(x) \) along these equalities, we get \( h' : R_i \to Q(\kappa(i, g)) \). Thus, since \( Q(\kappa(i, g)) \) is discrete, there is a \( d_\varsigma(i, g, h) : Q(\kappa(i, g)) \) such that \( d_\varsigma(i, g, h, x) = h'(x) = d_\varsigma(i, g, h) \) for all \( x : R_i \), as desired.

Now, suppose \( x : \mathcal{f}A \) and \( y : Q(x) \). By the last constructor of \( \mathcal{f}A \), we have \( x = \kappa(i, \text{const}_x) \).

Thus, transporting \( y \) along this equality, we obtain \( y' : Q(\kappa(i, \text{const}_x)) \). Since \( Q(\kappa(i, \text{const}_x)) \) is discrete, we have \( d_{\kappa''}(i, x, y) = y' = d_{\kappa''}(i, \text{const}_i, \text{const}_y) \) as desired.

**Corollary 9.10.** \( \mathcal{f}R_i \) is contractible for any \( i : I \).
Proof. It is inhabited by $\kappa_R(i, \sigma_R)$, so it remains to show $x = \kappa_R(i, \sigma_R)$ for all $x : \mathcal{R}$. Since $\mathcal{R}$ is discrete, so is this equality type; so by $\mathcal{R}$-induction, it suffices to prove $\sigma_R(x) = \kappa_R(i, \sigma_R)$ for all $x : \mathcal{R}$. But this follows from the third constructor of $\mathcal{R}$.

**Corollary** [R\(_\flat\)] 9.11. \(\mathcal{R}\) is contractible.

**Corollary** [C1] 9.12. For any $A$, the map $\sigma : A \to \mathcal{R}$ is surjective. In particular, if $\|\mathcal{R}\|$, then $\|A\|$.

Proof. Given $y : \mathcal{R}$, define $P(y) : \equiv \|\sum_{(x : A)} \sigma(x) = y\|$. By Lemma 8.8, each $P(y)$ is discrete. Thus, by Theorem 9.9, to prove $\prod_{(y : \mathcal{R})} P(y)$ (i.e. that $\sigma$ is surjective), it suffices to prove $\prod_{(x : A)} P(\sigma(x))$; but this is obvious.

**Remark** 9.13. Recall from Section 4 that many positive type formers have a ‘crisp’ variant of their induction principle, so that we can do case analysis on a $u :: A + B$ or induction on a $u :: \mathbb{N}$. This is not the case for $\mathcal{R}$, however. If it were, then we could do $\mathcal{R}$-induction on $u :: \mathcal{R}$, so that if $B$ were discrete, then any $M : B$ depending on $u :: A$ would factor through $\mathcal{R}$. Specializing to $A : \equiv \mathbb{R}$ and $B : \equiv 2$, any $M : 2$ depending on $u :: \mathbb{R}$ would be constant. However, since $u$ is crisp, by using the crisp LEM, we can define $M : 2$ to be $0_2$ if $u < 0$ and $1_2$ otherwise, which is not at all constant.

Now we have the reflection property:

**Theorem** [C0] 9.14. For any $A : Type$ and any discrete $B : Type$, composition with $\sigma : A \to \mathcal{R}$ induces an equivalence

\[(\mathcal{R} \to B) \simeq (A \to B).\]

Proof. $\mathcal{R}$-induction into the constant family $\lambda x. B$ yields a section of $(\mathcal{R} \cdot \sigma)$. Thus, it suffices to show that given $h, k : \mathcal{R} \to B$, if $h \circ \sigma = k \circ \sigma$, then $h = k$. This follows from another $\mathcal{R}$-induction with $P(x) : \equiv (h(x) = k(x))$, which is discrete by Lemma 8.5.

Moreover, analogously to how $\flat \dashv \sharp$ crisply (Corollary 6.26), we have $\mathcal{R} \dashv \flat$ crisply.

**Theorem** [C0] 9.15. For any $A, B :: Type$, there is a natural equivalence

\[\flat(\mathcal{R} \to B) \simeq \flat(A \to \flat B).\]

Proof. Since $\flat B$ is discrete, by Theorem 9.14, we have a natural equivalence

\[(\mathcal{R} \to \flat B) \simeq (A \to \flat B),\]

and this equivalence is preserved by $\flat$. On the other hand, since $\mathcal{R}$ is discrete, by Corollary 6.15, we have a natural equivalence

\[\flat(\mathcal{R} \to B) \simeq \flat(\mathcal{R} \to B).\]

Composing these two equivalences yields the conclusion.
Theorem 9.14 makes $\mathfrak{f}$ into a monadic modality, like $\mathfrak{♯}$, and its construction as a localization makes it an *accessible* one (see Rijke, Shulman, and Spitters 2017). Unlike $\mathfrak{♯}$, however, $\mathfrak{f}$ is not left exact. To show this, we introduce our final weakening of Axiom $R\flat$.

**Axiom C2.** Axiom C1 holds, and moreover, there exist $i_0 : I$ and $r_0, r_1 : R_{i_0}$ such that $R_{i_0}$ is a set and $r_0 \neq r_1$.

Note that Axiom $R\flat$ implies Axiom C2, since $0 \neq 1$ in $\mathbb{R}$.

**Theorem (C2) 9.16.** There exists a pullback that is not preserved by $\mathfrak{f}$.

**Proof.** The pullback of the inclusions $r_0, r_1 : 1 \to R_{i_0}$ is $\emptyset$. But $\mathfrak{f}1$ and $\mathfrak{f}R_{i_0}$ are both contractible, whereas $\mathfrak{f}\emptyset = \emptyset$ since it is already discrete. \(\square\)

(However, like any monadic modality, $\mathfrak{f}$ does preserve products.)

Viewing $\mathfrak{f}$ as a reflection gives another, more category-theoretic, way to phrase and prove Theorem 9.5.

**Theorem (R\flat) 9.17.** $\mathfrak{f}S^1 = S^1$.

**Proof.** By Theorem 9.14, $\mathfrak{f}$ is a left adjoint to the inclusion of discrete types in all types. Thus, it preserves all colimits, which is to say it takes colimits in the category of all types to colimits in the category of discrete types. Thus, since $S^1$ is the (homotopy) coequalizer of $R \supseteq R$, it follows that $\mathfrak{f}S^1$ is the coequalizer *in the category of discrete types* of $\mathfrak{f}R \supseteq \mathfrak{f}R$, and hence by Corollary 9.11 of $1 \supseteq 1$. Now, $S^1$ is the coequalizer of $1 \supseteq 1$ in the category of *all* types; but by Theorem 6.21, it lies in the subcategory of discrete types and hence is also the coequalizer there. \(\square\)

Similar methods may be used in many other examples. That is, given a ‘cell complex’ presentation of a classical topological space, if we can convert it into both a specification for a HIT and a colimit decomposition of that space that is sufficiently ‘cofibrant,’ then $\mathfrak{f}$ will preserve that colimit and take the space to the HIT. We do have to be careful to avoid the sort of cell complex used in classical algebraic topology where discs are glued along their *boundaries*, because such gluing in a constructive world tends to produce inhomogeneous ‘cusps.’ For instance, identifying the two endpoints of the topological interval $[0, 1] = \{x : \mathbb{R} \mid 0 \leq x \leq 1\}$ would not produce a space equivalent to $S^1$. This is also a known ‘defect’ (if one regards it so) of the topos model we have in mind; see also the remarks after Theorem 11.7. (It is ‘fixed’ in the topological topos of Johnstone (1979, Section 6), but in that model, no left adjoint $\mathfrak{f}$ can exist due to a lack of local connectedness.)

However, in many cases, it should be possible to remedy this by gluing along *open overlaps* instead. What makes this work is the fact that although $\mathbb{R}$ fails to satisfy the trichotomy principle constructively, we do have $(x < z) \to (x < y) \lor (y < z)$ for all

\*\*\*Although, amusingly, its shape would still be $S^1$: it would be the coequalizer of $1 \supseteq [0, 1]$, while $\mathfrak{f}[0, 1]$ is also contractible since $[0, 1]$ is a retract of $\mathbb{R}$. \*\*\*
$x, y, z : \mathbb{R}$. Combined with the following observation, this tells us that gluing along open overlaps produces no cusps.

**Lemma 9.18.** If $f : A \to C$ and $g : B \to C$ are embeddings, then their union as sub-types of $C$ is their pushout under their intersection: $A \cup B \simeq A \sqcup B$.

**Proof.** By definition, $f \cup g : A \cup B \to C$ is the smallest embedding through which both $f$ and $g$ factor, while $A \cap B$ is the pullback $A \times_C B$. If $h : D \to C$ is an embedding through which both $f$ and $g$ factor, then the square

$$
\begin{array}{ccc}
A \cap B & \to & B \\
\downarrow & & \downarrow \\
A & \to & D
\end{array}
$$

commutes since $h$ is an embedding, so there is an induced map $A \sqcup B \to D$. It remains to note that $A \sqcup B \to C$ is an embedding by Rijke (2017, Lemma 2.4). □

As an example, we give two other definitions of $S^1$.

**Theorem 9.19.** The following types are equivalent.

(i) $S^1$ as defined in Definition 9.1, i.e. the coequalizer of the identity and $+1 : \mathbb{R} \to \mathbb{R}$.

(ii) The subset $\{(x, y) \mid x^2 + y^2 = 1\}$ of $\mathbb{R} \times \mathbb{R}$ (the boundary of the topological disc $D^2$).

(iii) The coequalizer of the inclusion of open intervals $(0, \varepsilon) \to (0, 1)$ and the translated inclusion $(\lambda x. 1 - \varepsilon + x) : (0, \varepsilon) \to (0, 1)$, for any $0 < \varepsilon < \frac{1}{2}$.

**Proof.** We can define the functions sin and cos on the Dedekind reals constructively, and prove that $\sin^2 x + \cos^2 x = 1$. Thus, $\lambda x. (\cos(2\pi x), \sin(2\pi x))$ defines a map from $\mathbb{R}$ to $\{(x, y) \mid x^2 + y^2 = 1\}$, which by the periodicity of sin and cos descends to $S^1$. It is then straightforward to verify that this map is both surjective and injective, hence an equivalence since both types are sets. Thus, the types described in (i) and (ii) are equivalent.

Now, let $0 < \varepsilon < \frac{1}{2}$. Every real number is either $< \varepsilon$ or $> -\varepsilon$, so the set $\{(x, y) \mid x^2 + y^2 = 1\}$ in (ii) is the union of its subsets $U$ and $V$ consisting of the points where $y < \varepsilon$ and where $y > -\varepsilon$. Thus, by Lemma 9.18, it is the pushout of $U$ and $V$, each of which is isomorphic to an open interval, under their intersection, which is isomorphic to the disjoint union of two open intervals, included into $U$ and $V$ at their ends. Since (again by Lemma 9.18), the pushout of two open intervals under their intersection is their union, we can rearrange this pushout to become the type described in (iii). □

Similarly, we can obtain the topological sphere $S^2$ by gluing two topological discs $D^2$ (isomorphic to $\mathbb{R}^2$, hence with contractible shape) along a strip $S^1 \times \mathbb{R}$ (with shape $S^1$). Thus, $\bar{f}S^2$ is the homotopical suspension of $S^1$, i.e. the homotopical sphere $S^2$. We can proceed inductively for higher spheres, and so on. In this way, it may be possible to access homotopy types that would be quite difficult to present as HITs (due, for example, to their having infinitely many constructors), by first constructing their topological versions as sets using the Dedekind reals and then applying $\bar{f}$.

**Remark 9.20.** The overall conclusions drawn from Axiom C0 in this section and Section 8 can be summarized as ‘$\bar{f}$ is a reflector into the same subcategory that $\bar{g}$ coreflects into, and
it preserves finite products.’ As with the analogous conclusions about ♯ and ♬ noted in Remarks 3.12 and 6.29, in Schreiber and Shulman (2012), we assumed this axiomatically, while here we have deduced it from Axiom C0 and a higher inductive definition of ♦. This difference from Schreiber and Shulman (2012) is roughly orthogonal to our introduction of ♬ and ♯ using crisp variables; it is not necessitated by anything, but it is a convenient starting point for expressing the stronger Axiom C1, Axiom C2 and Axiom R♭. These axioms were not considered in Schreiber and Shulman (2012); but in Section 10, we will relate the first two to corresponding categorical properties studied by Lawvere (2007), Johnstone (2011) and Lawvere and Menni (2015).

Remark 9.21. The crisp adjointness $\int \dashv ♬$ from Theorem 9.15 finally allows us to justify the choice of the notations ♬ and ♯ (the notation and justification are both due to Schreiber (2013)). Suppose $G$ is a group, meaning a set (0-type) with a group structure (but, like all types in spatial type theory, also possessing intrinsic topology). We can construct a delooping $BG$ (a.k.a. $K(G, 1)$) by the methods of Licata and Finster (2014).

Now, if $X$ is any space, a principal $G$-bundle over $X$ with a flat connection assigns to every (topological) path in $X$ an element of $G$ (‘transport’ along that path), such that concatenating paths corresponds to multiplication in $G$, and so forth. Since paths in $X$ and elements of $G$ are the 1-morphisms in the fundamental $\infty$-groupoid of $X$ and in $BG$, respectively, such a bundle with flat connection can be described categorically as a map $fX \to BG$. But by Theorem 9.15, this is equivalent to a map $X \to ♬BG$. Thus, $♭BG$ is the moduli space of flat $G$-connections, justifying the notation ♬. The notation ♯ is chosen simply as dual to flat; although one can argue that if $Z$ classifies bundles of some sort, then ♯$Z$ classifies analogous bundles that are ‘sharp’ in that their fibres can vary discontinuously. (Unfortunately, as noted in Remark 7.7, in Awodey and Birkedal (2003) and Awodey, Birkedal, and Scott (1999), the symbols ♬ and ♯ were used for the propositional restriction of the same operations, but with reversed meanings.)

10. Axioms of cohesion

The axioms introduced in Sections 8 and 9 form a linear string of implications:

$$\mathsf{R♭} \Rightarrow \mathsf{C2} \Rightarrow \mathsf{C1} \Rightarrow \mathsf{C0}.$$  

Our main goal (Brouwer’s fixed point theorem) requires Axiom $\mathsf{R♭}$ (or at least its consequence Corollary 9.11). However, the weaker versions are much more general assumptions, which are not very specific to the ‘topological’ situation and are satisfied in many other models, so it is interesting that they suffice for most of the general theory. In this somewhat digressive section, we investigate some of their further consequences and their topos-theoretic meaning.

10.1. **Axiom C0 means stable local connectedness**

Combining Corollary 6.26 and Theorem 9.15, we see that Axiom C0 gives us a crisp ‘adjoint triple’ $\int \dashv ♬ \dashv ♯$, in which $\int$ preserves finite products. In topos-theoretic language, the category of types is ‘local and locally connected’ over the category of discrete (or,
equivalently, codiscrete) types, and in addition, the left adjoint preserves binary products. Johnstone (2011) called this additional condition *stable local connectedness*.

Conversely, if a topos is local and stably locally connected, then by combining Johnstone (2002, C3.6.3(d)) with Johnstone (2011, Proposition 1.3), it has a site of definition that is local and locally connected and closed under finite products. We should therefore be able to show that Axiom C0 holds in its internal spatial type theory by taking $I$ to be the discrete set of objects of the site and $R_i$ the corresponding representable. The condition that $A \to (R_i \to A)$ be an equivalence then says, in terms of sheaves, that $A(U) \to A(R_i \times U)$ is an isomorphism (or, in the higher topos case, an equivalence) for any object $U$ of the site (here, we use the fact that the site has finite products). The special case $U = 1$ tells us that $A$ is a constant (pre)sheaf, and the rest of the conditions follow automatically. (We could also take $I$ to be a set of objects that generate the site under finite products, as $R$ does for our site of cartesian spaces $\mathbb{R}^n$.)

Thus, Axiom C0 is exactly a type-theoretic incarnation of stable local connectedness (as an additional property added to a local topos). Schreiber (2013) calls a local and stably locally connected $\infty$-topos *cohesive* (although for Lawvere (2007) the word ‘cohesive’ also includes the categorical version of Axiom C1, below, and an additional axiom called ‘continuity’ that we do not consider here).

In addition to our motivating example (sheaves on cartesian spaces), several other similar examples of local and stably locally connected toposes and $\infty$-toposes can be found in Schreiber (2013). Here are a few more:

— By Johnstone (2011, Lemma 1.1) and Johnstone (2002, C3.6.3(a) and C3.6.17(a)), the topos $\text{Sh}(X)$ of sheaves on a space (or locale) is local and stably locally connected if and only if $X$ has both a focal point and a dense point. For example, in the Sierpinski space, the closed point is focal and the open point is dense.

— The local $\infty$-topos of parameterized spectra mentioned in Remark 7.6 is stably locally connected, with $\mathcal{S} = \mathcal{S}$. In fact, it is also punctually locally connected (see below).

— Rezk (2014) shows that ‘global equivariant homotopy theory’ forms a local and stably locally connected $\infty$-topos.

10.2. *Axiom C1 means punctual local connectedness*

If we additionally assume Axiom C1, then by Lemma 8.17, the discrete objects are closed under subobjects. Johnstone (2011) shows that this is equivalent to the following condition, there called *punctual local connectedness* (Lawvere (2007) calls it the *Nullstellensatz* and Schreiber (2013) calls it *pieces have points*).

**Theorem {C1} 10.1 (Johnstone 2011, Lemma 2.3).** For any $A :: \text{Type}$, the composite $\mathcal{S}A \to A \to \mathcal{J}A$ is surjective.

**Proof.** As in Univalent Foundations Program (2013, Theorem 8.8.1), it suffices to show that $\|\mathcal{S}A\|_0 \to \|\mathcal{J}A\|_0$ is surjective. Thus, since epimorphisms of sets are surjections (Univalent Foundations Program 2013, Lemma 10.1.4), it suffices to show (invoking the universal property of $\|\mathcal{-}\|_0$) that $(\mathcal{J}A \to C) \to (\mathcal{S}A \to C)$ is injective for any set $C$. Moreover, since in the proof of Lemma 10.1.4 from Univalent Foundations Program
(2013), we only need to instantiate this property with one particular set $C$, in fact it suffices to show that
\[ \hat{\varphi}(fA \to C) \to \hat{\varphi}(\varphi A \to C) \]
(17)
is injective for any crisp set $C$. And by Corollary 6.15, we may as well assume $C$ is also discrete (otherwise, we could apply $\hat{\varphi}$ to it).

Now, we can essentially copy the proof of Lemma 2.3 from Johnstone (2011). First, note that Equation (17) is equal to the composite
\[ \hat{\varphi}(fA \to C) \xrightarrow{\sim} \hat{\varphi}(A \to C) \xrightarrow{\varphi} \hat{\varphi}(\varphi A \to C) \xrightarrow{\sim} \hat{\varphi}(\varphi A \to C), \]
where the first equivalence is Theorem 9.14 and the last is because $C$ is discrete. Thus, it will suffice to show that the middle map is injective. But this map is equal to the composite
\[ \hat{\varphi}(A \to C) \to \hat{\varphi}(A \to \#C) \xrightarrow{\sim} \hat{\varphi}(\varphi A \to C) \xrightarrow{\sim} \hat{\varphi}(\varphi A \to C), \]
where the middle equivalence is Corollary 6.26 and the last is because $C$ is discrete. But $C \to \#C$ is injective by Corollary 8.23.

Thus, Axiom C1 implies punctual local connectedness. Conversely, by Johnstone (2011, Proposition 1.4), any punctually locally connected topos has a site of definition that is local and locally connected and in which every object has a global point. Clearly, in this case, if we take $I$ and $R_i$ as described above, then Axiom C1 will hold; thus, it is exactly a type-theoretic incarnation of punctual local connectedness.

In the different terminology of Lawvere and Menni (2015), Theorem 10.1 (along with our previous results) means that the category of types is ‘pre-cohesive’ over that of (co)discrete types. In fact, it is ‘stably pre-cohesive,’ because all of our theorems admit an arbitrary crisp context.

10.3. Axiom C2 means contractible codiscreteness

Following Lawvere and Menni (2015) further, we investigate the shapes of codiscrete types, for which purpose we bring back Axiom C2 (page 924).

**Theorem** {LEM,C2} 10.2. If $A$ is codiscrete, then $\mathcal{J}A$ is a proposition.

**Proof.** Let $x, y : \mathcal{J}A$; we must show $x = y$. By $\mathcal{J}$-induction, we may assume that $x = \eta(u)$ and $y = \eta(v)$ for $u, v : A$. Let $i_0$ and $r_0, r_1 : R_{i_0}$ be as in Axiom C2. Since they are axioms, they are crisp, and so we have $r_0^\# : \hat{\varphi}R_{i_0}$ and $r_1^\# : \hat{\varphi}R_{i_0}$. By Theorem 6.1, $(r_0^\# = r_1^\#)$ is equivalent to $\hat{\varphi}(r_0 = r_1)$, i.e. to $\hat{\varphi}\emptyset$ and hence to $\emptyset$. In other words, we have $(r_0^\# \neq r_1^\#)$.

Now, since $\hat{\varphi}R_{i_0}$ is a crisp discrete set, by the flat LEM it has decidable equality. Thus, there is a map $f : \hat{\varphi}R_{i_0} \to A$ with $f(r_0^\#) = u$ and $f(r_1^\#) = v$. But the inclusion $\hat{\varphi}R_{i_0} \to R_{i_0}$ is inverted by $\hat{\varphi}$ and hence also by $\emptyset$, so since $A$ is codiscrete our $f$ extends to a map $g : R_{i_0} \to A$ with $g(r_0) = u$ and $g(r_1) = v$. Now the composite $g \circ \eta : R_{i_0} \to \mathcal{J}A$ has $g(r_0) = x$ and $g(r_1) = y$; but as $\mathcal{J}A$ is discrete, $g$ is constant, so $x = y$. \qed

**Corollary** {LEM,C2} 10.3. For any crisp $B$, we have $\mathcal{J}\#B = \|B\|$. 

Proof. For any \( B \), we have \( B \to \int \sharp B \), hence a map \( \| B \| \to \int \sharp B \). But \( \♭ B \simeq \♭ \sharp B \to \int \sharp B \) is surjective by Theorem 10.1, and factors through \( \| B \| \). Hence, our map \( \| B \| \to \int \sharp B \) is also surjective, thus an equivalence.

Lawvere and Menni (2015) call the conclusion of Theorem 10.2 connected codiscreteness, and Corollary 10.3 (proven there as Lemma 7.3) explicit connected codiscreteness. In a homotopical context, however, it seems better to say contractible codiscreteness. As before, since we can work in an arbitrary crisp context, we automatically have ‘stable contractible codiscreteness.’

Remark 10.4. Unfortunately, the phrase ‘\( X \) is contractible’ is used by Lawvere and Menni (2015) to mean ‘\( \| (Y \to X) \|_0 \simeq 1 \) for all \( Y \),’ which conflicts with its standard usage in homotopy type theory to mean ‘\( X \simeq 1 \).’ (They do not mention the 0-truncation since they are working with 1-toposes.) The intuition behind their definition is that the sets \( \| (Y \to X) \|_0 \) are the hom-sets of a ‘strong homotopy category’ of types, analogous to the strong homotopy category of topological spaces obtained by identifying homotopic maps, so this definition says that \( X \) is terminal in that category. Since \( f \) is a functor from this strong homotopy category to the category of discrete spaces and their 0-truncated hom-sets, and it preserves the terminal object, if \( X \) is contractible in the sense of \textit{ibid.} then \( [X \simeq 1] \), i.e. \( [X] \) is contractible in the sense of homotopy type theory. Thus, if we want to keep ‘\( X \) is contractible’ to mean \( X \simeq 1 \), we could use a phrase like strongly spatially contractible for the ‘contractibility’ of Lawvere and Menni (2015).

Conversely, if a punctually locally connected topos satisfies Theorem 10.2, then we have an object, namely \( \sharp 2 \), for which \( \int \sharp 2 \) is contractible, but which has two unequal elements \( 0_\sharp \) and \( 1_\sharp \). (By Theorem 3.7, we have \( (0_\sharp = 1_\sharp) \simeq \sharp (0_2 = 1_2) \simeq \sharp \emptyset \), which is empty by Theorem 8.11.) Thus, we can make Axiom C2 hold by adding \( \sharp 2 \) to the collection of \( R_i \)’s; so Axiom C2 is exactly a type-theoretic incarnation of (punctual local connectedness and) contractible codiscreteness.

We could now copy the proof of Lawvere and Menni (2015, Corollary 6.5) to show that every type \( A \) embeds into a type \( B \) such that \( \int B \) is contractible, there called sufficient cohesion. I conjecture that Axiom R♭ and Axiom AC should together also imply the property called continuity in Lawvere (2007) and Menni (2014)—that the canonical map \( \int (A^S) \to (\int A)^S \) is an equivalence for any crisp set \( A \) and any crisp discrete set \( S \)—but I have not managed to prove this yet.

Axiom R♭ does not seem to have been studied yet in a topos-theoretic context. Of course, it is much more restrictive even than Axiom C2; rather than defining a general class of toposes for investigation, it singles out an important characteristic of a single topos (or a small group of toposes) that we are already interested in.

11. Synthetic topology

As a warm-up for Brouwer’s fixed-point theorem, in this section, we develop some aspects of synthetic topology that stay in the world of sets (0-types). Here, we can already see the
It is well-known that a continuous choice of \( x \) non-trivially. Taylor (2010) calls this property the power of Axiom R. Combining this with Axiom LEM, and also Axiom T from Section 8.3, we can prove a version of the Intermediate Value Theorem (which is a sort of one-dimensional analogue of Brouwer’s fixed-point theorem).

Theorem 11.1 (Discontinuous IVT). Let \( f : \mathbb{R} \to \mathbb{R} \) be a crisp function and \( c : \mathbb{R} \) a crisp real number. If there exist \( a, b : \mathbb{R} \) such that \( f(a) < c < f(b) \), then there exists an \( x : \mathbb{R} \) such that \( f(x) = c \).

Proof. Since the hypothesis of \( a \) and \( b \) is a crisp proposition, it is discrete. Thus, all our hypotheses may as well be crisp, so we may prove \( \exists x : \mathbb{R} \mid f(x) = c \) instead, which by Corollary 8.12 is equivalent to \( \neg \neg \sum_{x : \mathbb{R}} f(x) = c \). Thus, suppose for contradiction \( \neg \sum_{x : \mathbb{R}} f(x) = c \), i.e. for all \( x : \mathbb{R} \), we have \( f(x) \neq c \). By Theorem 8.32, for all \( x \), we have \( f(x) \neq c \), hence either \( f(x) < c \) or \( f(x) > c \). Let \( U = \{ x \mid f(x) < c \} \) and \( V = \{ x \mid f(x) > c \} \); then \( U \cup V = \mathbb{R} \) and \( U \cap V = \emptyset \). By Theorem 11.1, then, either \( U = \mathbb{R} \) or \( V = \mathbb{R} \), which is a contradiction since \( a \in U \) and \( b \in V \).

The crispness of the hypotheses \( f \) and \( c \) means that the point \( x \) does not vary continuously with them. It is well-known that a continuous choice of \( x \) is impossible, and that this is what prevents the classical version of IVT from holding constructively (see, for instance, Taylor 2010). Our technology of spatial type theory enables us to state a ‘discontinuous’ IVT, even in a world where all types are spaces, and our axioms of classicality and real-cohesion allow us to prove it.

A more common way to constructivize the IVT is to strengthen the hypothesis (e.g. that \( f \) ‘does not hover’ or is ‘locally non-constant’) or weaken the conclusion (e.g. we only find an approximate solution). We can also use Axiom R to prove such a form of IVT, without the need for Axiom LEM or Axiom T.

First, we show that \( \mathbb{R} \) is connected in a different sense. The following proof also introduces an idea that we will use repeatedly. Since \( f \) is a left adjoint, it takes colimits of types to colimits in the subcategory of discrete types. Moreover, since crisp discrete types are also coreflective, they are closed under colimits, so \( f \) preserves crisp colimits of types. Similarly, using Corollary 8.25, crisp discrete sets are also closed under set-colimits (i.e. 0-truncations of homotopy colimits). Thus, if we write \( f_0 A \) for \( \|Af\|_0 \), then the functor \( f_0 \) reflects sets into discrete sets, so it preserves crisp set-colimits. We can then use facts about (set-)colimits of the shapes of types to conclude things about the types themselves.
**Theorem (R)} 11.3.** Let $U, V :: \mathbb{R} \to \text{Prop}$ be crisp subsets of $\mathbb{R}$ with $U \cup V = \mathbb{R}$. If $\| U \|$ and $\| V \|$, then also $\| U \cap V \|$.

**Proof.** By Lemma 9.18, the assumption ensures that $\mathbb{R}$ is the pushout of $U$ and $V$ over $U \cap V$. Thus, the contractible type $\int \mathbb{R}$ is the pushout of $\int U$ and $\int V$ under $\int (U \cap V)$, and hence the contractible $\int_0 \mathbb{R}$ is the set-pushout of $\int_0 U$ and $\int_0 V$ under $\int_0 (U \cap V)$. But $\| U \|$ and $\| V \|$, hence also $\| \int_0 U \|$ and $\| \int_0 V \|$.

Now, if $x : \int_0 U$ and $y : \int_0 V$, their images in the contractible type $\int_0 \mathbb{R}$ must be equal. Using the explicit construction of this set-pushout as the set-quotient of an equivalence relation of zigzags on $\int_0 U + \int_0 V$, we see that we must, in particular, have $\| \int_0 (U \cap V) \|$. Finally, since $\int (U \cap V) \to \int_0 (U \cap V)$ is 0-connected and hence surjective, while $(U \cap V) \to \int (U \cap V)$ is surjective by Corollary 9.12, we also have $\| U \cap V \|$. \hfill \Box

Taylor (2010) calls the above property *overt connectedness*. Note that we have only proven it for crisp subsets.

The 0-truncations in the above proof could be omitted, but if we left them out, then instead of the set-based construction of set-pushouts we would need to use the van Kampen theorem (Univalent Foundations Program 2013, Section 8.7). The former needs only the fact that the sets form a πωW-pretopos (Univalent Foundations Program 2013, Theorem 10.1.11), which requires only function extensionality and propositional extensionality; whereas, the latter in general requires full univalence. Similar remarks apply to the proof of Theorem 11.5 below, but in Theorem 12.6, we will need to resort to univalence and van Kampen. (Actually, based on intuition from classical topology, it is natural to conjecture that $\int U$ is automatically a set for any subset $U \subseteq \mathbb{R}$, but I do not know how to prove this.)

**Theorem (R)} 11.4 (Approximate IVT).** Let $f :: \mathbb{R} \to \mathbb{R}$ be a crisp function and $c :: \mathbb{R}$ a crisp real number. Then for any $\varepsilon > 0$, if there exist $a, b : \mathbb{R}$ such that $f(a) < c < f(b)$, then there exists an $x : \mathbb{R}$ such that $|f(x) - c| < \varepsilon$.

**Proof.** It suffices to consider the case when $\varepsilon : \mathbb{Q}^+$ is a positive rational number. Then, since $\mathbb{Q}^+$ is discrete, we may assume $\varepsilon$ is also crisp. Therefore, the sets $U := \{ x : \mathbb{R} \mid f(x) < c + \varepsilon \}$ and $V := \{ x : \mathbb{R} \mid f(x) > c - \varepsilon \}$ are also crisp. We have $U \cup V = \mathbb{R}$ by locatedness of Dedekind reals, while by assumption $a \in U$ and $b \in V$. Thus, Theorem 11.3 supplies the desired $x$. \hfill \Box

Note that although Theorem 11.4 eliminates classicality axioms, we still have only crisp dependence on $f$ and $a$. In fact, the approximate IVT (for continuous functions) is constructively provable without any additional assumptions; the most common proofs use countable choice or uniform continuity of $f$, but Frank (2017) has recently given a proof avoiding both of these. So there is no topological reason for this crispness restriction, but I do not know whether it can be removed using our current methods.

On the other hand, Theorems 11.2 and 11.4 are statements about all functions $\mathbb{R} \to \mathbb{R}$. No explicit continuity hypothesis is required, because in our synthetic world ‘all functions are continuous.’ In fact, with a little more work we can actually prove explicitly that all
functions are continuous, thereby showing that the axioms of real-cohesion really do nail down precisely the ‘topology’ of $\mathbb{R}$ to be the intended one.

**Theorem (R5,T) 11.5.** Every crisp function $f :: \mathbb{R} \to \mathbb{R}$ is $\varepsilon$-$\delta$ continuous at every crisp real number $a :: \mathbb{R}$.

*Proof.* We will need to assume given crisp $b_1 < a$ and $b_2 > a$ such that $f(b_1) \neq f(a)$ and $f(b_2) \neq f(a)$. This can always be ensured by modifying $f$ without changing it near $a$. For instance, $f(a + 2) - f(a)$ is either $> \frac{1}{2}$ or $< 1$ (and this is crisp since it is a proposition); if the former we can take $b_2 = a + 2$, while if the latter we can subtract $\max(0, x - a - 1)$ from $f$ before taking $b_2 = a + 2$.

As before, we may assume $\varepsilon :: \mathbb{Q}^+$ is a crisp positive rational number. We may also assume $\varepsilon$ is less than both $|f(b_1) - f(a)|$ and $|f(b_2) - f(a)|$. Define

$$U = \{x : \mathbb{R} | |f(x) - f(a)| < \varepsilon/2\};$$

$$V = \{x : \mathbb{R} | |f(x) - f(a)| > \varepsilon/4\}.$$

Then, $U \cup V = \mathbb{R}$, so that $\mathbb{R}$ is the pushout of $U$ and $V$ under $U \cap V$. Let $R_1 = (-\infty, a]$ and $R_2 = [a, \infty)$, and $U_i = U \cap R_i$, $V_i = V \cap R_i$ for $i = 1, 2$, so that $R_i = U_i \cup V_i$ and $R_i$ is the pushout of $U_i$ and $V_i$ under $U_i \cap V_i$. Note that $a \in U_i$ and $b_i \in V_i$ for $i = 1, 2$.

Consider the fibres of the map $U \to f_0 U$; these are the ‘spatially connected components’ of $U$. Let $w$ be the image of $a$ in $f_0 U$, and $W$ the preimage of $w$ in $U$; thus, $W$ is the ‘component of $U$ containing $a$.’ Similarly, define $w_i$ and $W_i$ for $i = 1, 2$ to represent the component of $U_i$ containing $a$. Then, the function $f_0 U_i \to f_0 U$ maps $w_i$ to $w$, so that $W_i$ is a subset of $W$. Similarly, let $Z_i$ be the component of $V_i$ containing $b_i$, with image $z_i$ in $f_0 V_i$, for $i = 1, 2$.

Now as remarked above, $f_0$ preserves crisp set-colimits. Thus, the contractible $f_0 R$ is the set-pushout of $f_0 U$ and $f_0 V$ under $f_0(U \cap V)$, and similarly for $i = 1, 2$. In particular, $w_i$ and $z_i$ are identified in the set-pushout of $f_0 U_i$ and $f_0 V_i$ under $f_0(U_i \cap V_i)$, and so $w_i$ must be equal to the image of some element of $f_0(U_i \cup V_i)$. Since $X \to f_0 X$ is surjective by Corollary 9.12, $W_i$ must contain an element of $U_i \cap V_i$. Let $y_i$ be such an element. Since $y_i \in V_i$, we have, in particular, $f(y_i) \neq f(a)$, and hence $y_i \neq a$. By Theorem 8.32, $y_i \neq a$; so since $y_i \in R_i$, we have $y_i < a < y_2$. Finally, since $y_i \in U_i$, also $y_i \in U$.

Let $\delta = \min(y_2 - a, a - y_1)$; I claim this is the $\delta$ to our $\varepsilon$. Thus, suppose $x : \mathbb{R}$ and $|x - a| < \delta$, hence $x \in (y_1, y_2)$. (Note that $x$ is not crisp.) Either $|f(x) - f(a)| < \varepsilon$ or $|f(x) - f(a)| > \varepsilon/2$; the first is what we want, so it suffices to show that the second leads to a contradiction.

Thus, suppose $|f(x) - f(a)| > \varepsilon/2$. Then since any $p \in U$ satisfies $|f(p) - f(a)| < \varepsilon/2$ by definition of $U$, we have $f(p) \neq f(x)$, hence $p \neq x$. By Theorem 8.32, therefore, $p \neq x$, and thus either $p < x$ or $x < p$. In other words, $\prod_{\mathbb{P} \cap U}(p < x) \lor (x < p)$, which is equivalent to $\prod_{\mathbb{P} \cup U}(p < x) \lor (x < p)$ since $p$ cannot be both greater and less than $x$. It follows that we can define a map $q : U \to 2$ by $q(p) = 0_2$ if $p < x$ and $q(p) = 1_2$ if $x < p$. In particular, since $y_1 < x$ and $x < y_2$, we have $q(y_1) = 0_2$ and $q(y_2) = 1_2$. But $y_1$ and $y_2$ are also both in $W$, i.e. they are identified in $f_0 U$. Since $2$ is a discrete set, $q : U \to 2$ must
factor through \( f_0 U \), and so \( q(y_1) = q(y_2) \), a contradiction. So it must be that the other case holds: \( |f(x) - f(a)| < \epsilon \), as desired. \( \square \)

The statement that all functions \( R \to R \) are \( \epsilon, \delta \)-continuous is sometimes called ‘Brouwer’s theorem,’ because Brouwer proved it in his ‘intuitionistic mathematics’ using choice sequences. Here, we have proved a version of it using instead our Axiom R\( \mathfrak{B} \) and Axiom T. (Both uses of Axiom T could also be replaced by the assumption that \( f \) is ‘strongly extensional,’ i.e. \( f(x) \neq f(y) \) implies \( x \neq y \).

Theorem 11.5 provides an almost complete answer to the question of ‘what is the topology of \( R \),’ with ‘topology’ interpreted in the sense relevant to our intended model, namely ‘functions out of \( R \).’

**Corollary \{LEM,R\( \mathfrak{B},T \}\} 11.6.** The canonical function \( \gamma(R \to R) \to (\gamma R \to \gamma R) \) is an injection, and coincides with the inclusion of the \( \epsilon,\delta \)-continuous functions.

**Proof.** To show it is an injection, suppose given \( f, g : R \to R \) such that \( \gamma f = \gamma g \) as functions \( \gamma R \to \gamma R \), or equivalently \( \# f = \# g \) as functions \( \# R \to \# R \). We must show that for any \( x : R \), we have \( f(x) = g(x) \); but since \( R \) is concrete by Theorem 8.20, it suffices to show \( f(x)^{\#} = g(x)^{\#} \), which follows from \( \# f = \# g \).

Now, Theorem 11.5 implies that this injection lands inside the \( \epsilon,\delta \)-continuous functions. Conversely, suppose \( f : \gamma R \to \gamma R \) is \( \epsilon,\delta \)-continuous. By Axiom LEM applied in the universe of discrete types, \( f \) is locally uniformly continuous therein (meaning that it is uniformly continuous on all finite intervals). Since it suffices to consider intervals with rational endpoints, while \( \mathbb{Q} \) is discrete and discrete types are closed under all logical operations, \( f \) is also locally uniformly continuous in the world of all types. Therefore, so is the composite \( \mathbb{Q} \to \gamma R \to \gamma R \to R \). But it can be proved constructively that any locally uniformly continuous function \( \mathbb{Q} \to R \) extends uniquely to \( R \); see Troelstra and van Dalen (1988, Theorem 5.6.2). And such an extension restricts to \( f \) on \( \gamma R \), since the two agree on \( \mathbb{Q} \) and both are continuous. \( \square \)

It would be even better to characterize the ‘topology’ of \( R \to R \) itself and so on, or equivalently the functions \( R^n \to R \) for all \( n : \mathbb{N} \), but I do not know how to do this without additional axioms. However, the general method of proof of Theorem 11.5 does have various further applications. Here, we mention only one more: the promised falsity of the analytic LLPO from Section 8.3.

**Theorem \{LEM,R\( \mathfrak{B} \}\} 11.7.** It is not the case that for all \( x : R \), we have \( x \leq 0 \lor x \geq 0 \).

**Proof.** By Corollary 8.16, it suffices to show \( \neg \gamma (\forall x : R. (x \leq 0 \lor x \geq 0)) \). For contradiction, suppose \( \gamma (\forall x : R. (x \leq 0 \lor x \geq 0)) \), hence \( \forall x : R. (x \leq 0 \lor x \geq 0) \) crisply.

Let \( f : R \to R \) be such that \( f(0) = 0 \) and \( f(x) = x \sin(\frac{x}{\pi}) \) for any \( x \neq 0 \). Such a function can be defined constructively without too much trouble, and since it needs no hypotheses, it is crisp. Let \( U = \{ x : R \mid x \geq 0 \land f(x) \geq 0 \} \) and \( V = \{ x : R \mid x \leq 0 \land f(x) \leq 0 \} \); then \( U \) and \( V \) are also crisp. Our assumption implies that \( U \cup V = [0, \infty) \).

Classically, \( U \) would be the disjoint union of \( \{0\} \) with the intervals \( \left[ \frac{1}{n+1}, \frac{1}{n} \right] \) for even \( n \), and \( V \) the similar union for odd \( n \). (Here and subsequently, we adopt the convention
that when \( n = 0 \), the interval \( \left[ \frac{1}{n+1}, \frac{1}{n} \right] \) means \([1, \infty)\). In fact, Axiom LEM and Axiom C1 suffice to make this true, by the following argument.

Let \( \mathbb{N}_\infty \) be the type of functions \( \mathbb{N} \to 2 \) that take the value \( 1_2 \) at most once (this is equivalent to the type of non-increasing sequences). Now, for any even \( n \) and any \( x \in U \), we have \( (x < \frac{1}{n+1}) \lor (x > \frac{1}{n}) \); but since \( x \notin \left( \frac{1}{n}, \frac{1}{n+1} \right) \), it follows that \( (x \leq \frac{1}{n}) \lor (x > \frac{1}{n}) \). Similarly, \( (x \geq \frac{1}{n+1}) \lor (x < \frac{1}{n}) \), and so the proposition \( x \in \left[ \frac{1}{n+1}, \frac{1}{n} \right] \) is decidable (given that \( x \in U \)). Thus, we can define \( p : U \to \mathbb{N}_\infty \) by

\[
p(x)_n = \begin{cases} 1_2 & x \in \left[ \frac{1}{2n+1}, \frac{1}{2n} \right] \\ 0_2 & x \notin \left[ \frac{1}{2n+1}, \frac{1}{2n} \right]. \end{cases}
\]

If we write \( n : \mathbb{N}_\infty \) for the characteristic sequence of \( n : \mathbb{N} \), and \( \infty : \mathbb{N}_\infty \) for the constantly-02 sequence, then \( p^{-1}(n) = \left[ \frac{1}{2n+1}, \frac{1}{2n} \right] \) and \( p^{-1}(\infty) = \{0\} \). On the other hand, as for any function, \( U \simeq \sum_{n : \mathbb{N}_\infty} p^{-1}(n) \). But by LPO (Theorem 8.29), \( \mathbb{N}_\infty = \mathbb{N} + 1 \), so we have

\[
U \simeq \{0\} + \sum_{n : \mathbb{N}} [\left[ \frac{1}{2n+1}, \frac{1}{2n} \right]] \quad \text{and similarly} \quad V \simeq \{0\} + \sum_{n : \mathbb{N}} [\left[ \frac{1}{2n+2}, \frac{1}{2n+1} \right]].
\]

Now as remarked above, \( f \) preserves coproducts of crisp types (such as the summands of \( U \) and \( V \)). Similarly, for any \( P : \mathbb{N} \to \text{Type} \), by the generalization of Univalent Foundations Program (2013, Theorem 7.3.9) to monadic modalities, we have

\[
f(\sum_{n : \mathbb{N}} P(n)) \simeq \sum_{n : \mathbb{N}} f(P(n)) \quad \text{(since this is already discrete, as \( \mathbb{N} \) is)}.
\]

Thus, \( fU \simeq 1 + \sum_{n : \mathbb{N}} [\left[ \frac{1}{2n+1}, \frac{1}{2n} \right]] \) and similarly for \( V \). But any closed interval \([a, b]\) is a retract of \( \mathbb{R} \); hence, \( f[a, b] \) is a retract of \( f\mathbb{R} \) and thus contractible. Therefore, \( fU \simeq 1 + \mathbb{N} \) and also \( fV \simeq 1 + \mathbb{N} \).

Recalling that \([0, \infty) = U \cup V \), it follows that \([0, \infty)\) is the pushout of \( U \) and \( V \) under their intersection. However, this intersection is just \{0\} \( \cup \left\{ \frac{1}{n} \mid n : \mathbb{N} \right\} \), which is another copy of \( 1 + \mathbb{N} \). Since \( f \) preserves this pushout, the contractible type \( f[0, \infty) \) is the pushout of \( 1 + \mathbb{N} \) and \( 1 + \mathbb{N} \) under \( 1 + \mathbb{N} \). Moreover, the two maps \( 1 + \mathbb{N} \to 1 + \mathbb{N} \) in this pushout can be identified with the identity and with ‘+1’ acting on \( \mathbb{N} \), respectively (arising from the inclusions of \( \frac{1}{n} \) as the left endpoint of one interval and the right endpoint of another). But the pushout of these two maps is \( 2 \) rather than \( 1 \), since there is nothing to identify the elements of \( 1 \) (which correspond to \([0\) in \([0, \infty)\)) with anything else. This is a contradiction, so our assumption \( f(\forall x : \mathbb{R}, (x < 0 \lor x \geq 0)) \) is false.

\[\square\]

Theorem 11.7 explains why classical ‘closed’ cell complexes must be avoided in real-cohesion, as mentioned in Section 9. Namely, the analytic LLPO claims that the evident map \((-\infty, 0] + [0, \infty) \to \mathbb{R} \) is surjective, which would be exactly what we need in order to be able to glue (say) the endpoints of a closed interval together seamlessly; its failure means that such a gluing will not give the desired answer. (Conversely, therefore, if one wants a ‘topological topos’ in which classical cell complexes \textit{do} work, one has to give up on local connectedness, as done by Johnstone (1979).)
12. Brouwer’s fixed-point theorem

Finally, we are ready to attack Brouwer’s fixed-point theorem. We confine ourselves to the simplest version, which is about self-maps of the disc:

\[ \mathbb{D}^2 \equiv \{(x, y) : \mathbb{R} \times \mathbb{R} \mid x^2 + y^2 \leq 1\} . \]

Recall from Theorem 9.19 that the boundary of the disc

\[ \partial \mathbb{D}^2 \equiv \{(x, y) : \mathbb{R} \times \mathbb{R} \mid x^2 + y^2 = 1\} , \]

is an equivalent definition of the topological circle \( S^1 \); it is the only one we will use in this section. Recall also from Theorem 9.17 that \( \mathbb{S}^1 \simeq S^1 \). We also have:

**Lemma (R\text{S})** 12.1. \( \mathbb{I} \mathbb{D}^2 \) is contractible.

**Proof.** Since \( \mathbb{I} \mathbb{R} \) is contractible and \( \mathbb{I} \) preserves finite products, \( \mathbb{I} (\mathbb{R} \times \mathbb{R}) \) is also contractible. But \( \mathbb{D}^2 \) is a retract of \( \mathbb{R} \times \mathbb{R} \), by the function

\[
(x, y) \mapsto \left( \frac{x}{\max(1, \sqrt{x^2 + y^2})}, \frac{y}{\max(1, \sqrt{x^2 + y^2})} \right).
\]

Hence, \( \mathbb{I} \mathbb{D}^2 \) is a retract of \( \mathbb{I} (\mathbb{R} \times \mathbb{R}) \), and thus also contractible. \( \square \)

In particular, therefore, we have:

**Lemma (UA,R\text{S})** 12.2. \( S^1 \) is not a retract of \( \mathbb{D}^2 \).

**Proof.** If it were, \( \mathbb{I} S^1 \) would be a retract of \( \mathbb{I} \mathbb{D}^2 \) and hence contractible. But \( \mathbb{I} S^1 = S^1 \), which is not contractible since its loop space is \( \mathbb{Z} \) (Licata and Shulman 2013). \( \square \)

So much for the homotopy-theoretic part of the proof; we proceed to the topological part. This might be done in many ways, but we choose to be explicit and calculational to make it obvious that the argument is constructive. Here is also where the mysterious Axiom T from Section 8.3 reappears.

**Lemma (LEM,C1,T)** 12.3. If there is a map \( f : \mathbb{D}^2 \to \mathbb{D}^2 \) with no fixed point, then \( S^1 \) is a retract of \( \mathbb{D}^2 \).

**Proof.** Our assumption is that for all \( z : \mathbb{D}^2 \), we have \( f(z) \neq z \). Write \( z = (x, y) \) and \( f(z) = (u, v) \); then \( f(z) \neq z \) means \( (x - u)^2 + (y - v)^2 \neq 0 \), hence by Theorem 8.32 (using Axiom T) \( (x - u)^2 + (y - v)^2 > 0 \).

Now the line through \( f(z) \) and \( z \) can be parameterized as

\[
(tx + (1 - t)u, ty + (1 - t)v) ,
\]

with \( t = 0 \) being \( f(z) \) and \( t = 1 \) being \( z \). We are interested in the intersections of this line with \( S^1 \), which are given by solving a quadratic equation:

\[
(tx + (1 - t)u)^2 + (ty + (1 - t)v)^2 = 1 \iff ((x - u)t + u)^2 + ((y - v)t + v)^2 = 1 \iff ((x - u)^2 + (y - v)^2)t^2 + 2((x - u)u + (y - v)v)t + (u^2 + v^2 - 1) = 0 .
\]
The standard quadratic formula yields the solutions
\[ t = \frac{-B \pm \sqrt{D}}{A}, \]
where
\[ B = ((x - u)u + (y - v)v), \]
\[ D = ((x - u)u + (y - v)v)^2 - ((x - u)^2 + (y - v)^2)(u^2 + v^2 - 1), \]
\[ A = (x - u)^2 + (y - v)^2. \]

To ensure that this yields real solutions constructively, we need to know that \( A \neq 0 \) and \( D \geq 0 \). As we saw above, the former follows from our assumption that \( f(z) \neq z \). For the latter, we can compute
\[ D = (x - u)^2 + (y - v)^2 - (vx - uy)^2. \]
Now, \((x - u)^2 + (y - v)^2\) is the squared length of one diagonal of the parallelogram spanned by \( z \) and \( f(z) \) regarded as vectors originating at the origin, while \((vx - uy)^2\) is the squared area of that parallelogram. The area of a parallelogram is at most half the product of the lengths of its diagonals; thus, \( D \geq 0 \) as long as the other diagonal of the parallelogram in question is \( \leq 2 \). But this other diagonal is the vector sum \( z + f(z) \), which has magnitude \( \leq 2 \) since both \( z \) and \( f(z) \) lie in \( \mathbb{D}^2 \).

Thus, we constructively have two real solutions to our quadratic equation for \( t \). We choose the one in which the \( \pm \) sign is +. Substituting this value of \( t \) into Equation (18) of our line, we obtain a formula for a point in \( \mathbb{R}^2 \) that is a function of \((x, y)\) and \((u, v)\); hence, recalling that \((u, v) = f(x, y)\) of \( z = (x, y) : \mathbb{D}^2 \) only. Moreover, by construction, this point always lies in \( S^1 \). Let us denote it \( r(z) \); thus, we have \( r : \mathbb{D}^2 \to S^1 \).

It remains to show that \( r \) is a retraction of the obvious inclusion, i.e. that if \( z = (x, y) \) lies in \( S^1 \), then \( r(z) = z \). For this, it suffices to show that if \( x^2 + y^2 = 1 \), then our chosen value of \( t \) becomes equal to 1. This is equivalent to asking that \( A + B = \sqrt{D} \), and this in turn is equivalent to asking that \( A + B \geq 0 \) and that \((A + B)^2 - D = 0 \). For the first, we compute
\[ A + B = x^2 + y^2 - ux - vy \]
\[ = 1 - (ux + vy). \]
But \( ux + vy \) is the dot product of \( z \) and \( f(z) \), which is \( \leq 1 \) since both have magnitude \( \leq 1 \). Finally, for the second, we compute
\[ (A + B)^2 - D = ((x - u)^2 + (y - v)^2)(x^2 + y^2 - 1), \]
which clearly vanishes if \( x^2 + y^2 = 1 \).

Combining Lemmas 12.2 and 12.3, we see that an arbitrary \( f : \mathbb{D}^2 \to \mathbb{D}^2 \) cannot fail to have a fixed point. To deduce a positive statement from this, we invoke Axiom LEM.

**Theorem 12.4 (Sharp Brouwer’s fixed-point theorem).** For any function \( f : \mathbb{D}^2 \to \mathbb{D}^2 \), we have \( \| \sum_{(x : \mathbb{D}^2)} f(x) - x \| \geq 0 \). That is, any self-map of \( \mathbb{D}^2 \) sharply has a fixed point.
Proof. By Corollary 8.12, the claim is equivalent to \( \neg \sum_{(x : D^2)} f(x) = x \). But this is what we have just proven.

Equivalently, we can say:

**Corollary {UA,LEM,R5,T} 12.5 (Crisp Brouwer’s fixed-point theorem).** For any crisp function \( f :: D^2 \to D^2 \), there exists \( x : D^2 \) such that \( f(x) = x \).

Proof. If \( f \) is crisp, then so is the fixed point asserted by Theorem 12.4, so we can strip the \( \# \) from it.

Topologically, these formulations both say the same thing: every continuous self-map of \( D^2 \) has a fixed point, but such a fixed point cannot be selected continuously. Since the existential in both theorems is propositionally truncated, even if the \( \# \) or crispness could be omitted we would not be asserting the existence of a single continuous function selecting fixed points. However, according to Remark 7.8, we would be asserting that such a function exists locally, i.e. that any given function has a neighbourhood of functions on which fixed points can be selected continuously; and this is just as impossible. Thus, the versions of the theorem that we have proven really are the strongest we can expect.

On the other hand, just as we did with the Intermediate Value Theorem in Section 11, we can eliminate Axiom LEM and Axiom T by constructing an approximate fixed point. Our proof of this will also involve more synthetic homotopy theory: in addition to \( \pi_1(S^1) = \mathbb{Z} \), we will use the van Kampen theorem for pushouts.

**Theorem {UA,R5} 12.6 (Approximate Brouwer’s fixed-point theorem).** For any crisp function \( f :: D^2 \to D^2 \) and \( \varepsilon > 0 \), there exists an \( x : D^2 \) such that \( |f(x) - x| < \varepsilon \).

Proof. As before, we may assume that \( \varepsilon \) is rational, and hence that it is crisp. We may also assume \( \varepsilon < 1 \). Define

\[
g(x) = \begin{cases} 
  f(x), & \text{if } |f(x)| > 1 - \varepsilon \\
  \frac{\min(|f(x)|, 1 - \varepsilon/2)}{|f(x)|}, & \text{if } |f(x)| < 1 - \varepsilon/2.
\end{cases}
\]

The case division is constructively valid by Lemma 9.18, since one of the two cases must be true, and if both are true, then the two definitions agree. And since \( \varepsilon < 1 \), if \( |f(x)| > 1 - \varepsilon \), then \( |f(x)| > 0 \), so that \( \frac{\min(|f(x)|, 1 - \varepsilon/2)}{|f(x)|} \) makes sense.

Now, this definition ensures that \( |f(x) - g(x)| < \varepsilon/2 \), so it suffices to find an \( x : D^2 \) such that \( |g(x) - x| < \varepsilon/2 \). Moreover, we have \( |g(x) - x| > \varepsilon/4 \) for all \( x : S^1 \).

Let \( U = \{ x : D^2 \mid |g(x) - x| > \varepsilon/4 \} \) and \( V = \{ x : D^2 \mid |g(x) - x| < \varepsilon/2 \} \). Then, \( U \cup V = D^2 \), exhibiting \( D^2 \) as a crisp pushout of \( U \) and \( V \) under their intersection. Therefore, the contractible type \( J D^2 \) is also the pushout of \( JU \) and \( JV \) under \( J(U \cap V) \).

However, \( S^1 \) is contained in \( U \) by the construction of \( g \), and the construction of the retraction \( r \) in Lemma 12.3 works for all \( x : U \). Thus, \( S^1 \) is a retract of \( U \), and so \( JS^1 \) is a retract of \( JU \). But since \( JS^1 = S^1 \), it contains a non-identity identification, and hence so does \( JU \). On the other hand, this identification must be mapped to the identity in the contractible type \( J D^2 \).
Now, recall that the van Kampen theorem (Univalent Foundations Program 2013, Section 8.7) identifies the 0-truncated equality types of a (homotopy) pushout as a certain set-quotient. Inspecting this construction, we see that the only way two identifications in $fU$ can become equal in $fU \sqcup (f(U \cap V) \sqcup fV$ if we have $\|fV\|$. Hence, by Corollary 9.12, we also have $\|V\|$.

As in Theorem 11.4, there seems no topological reason for the crispness of $f$, but I do not know how to remove it. Note that although the technique is similar to that in Theorems 11.3 and 11.5, here we need to use $f$ rather than $f_0$, and likewise we need the van Kampen theorem for pushouts that are not sets (whose proof also involves the full UA).

Acknowledgements

This paper owes a lot to many people. My understanding of cohesive $(\infty, 1)$-toposes developed over the course of many discussions with Urs Schreiber, who also helped greatly with understanding the categorical semantics of modalities (this joint work is represented in Schreiber and Shulman (2012), which this paper draws heavily on). Other participants at the nForum and n-Category Café have also been very helpful, particularly Zhen Lin. Andrej Bauer and Martín Escardó patiently explained to me the relationship between different classicality axioms and some existing theories of synthetic topology, and Bas Spitters contributed the proof of Example 8.34. Urs Schreiber, Bas Spitters, Egbert Rijke and Martín Escardó also gave useful feedback on drafts. Finally, Dan Licata explained Pfenning and Davies (2001) to me, did most of the work on our joint paper (Licata and Shulman 2016) generalizing Reed (2009) that led to the spatial type theory presented here, and has been generally indispensable in teaching me to sound at least vaguely like a type theorist.

References

Awodey, S. and Birkedal, L. (2003). Elementary axioms for local maps of toposes. *Journal of Pure and Applied Algebra* 177 (3) 215–230.
Awodey, S., Birkedal, L. and Scott, D. S. (1999). Local realizability toposes and a modal logic for computability: (Extended Abstract). *Electronic Notes in Theoretical Computer Science* 23 (1) 13–26.
Awodey, S. and Warren, M.A. (2009). Homotopy theoretic models of identity types. *Mathematical Proceedings of the Cambridge Philosophical Society* 146 (45) 45–55.
Baez, J.C. and Hoffnung, A.E. (2011). Convenient categories of smooth spaces. *Transactions of the American Mathematical Society* 363 (11) 5789–5825.
Bauer, A. and Lešnik, D. (2012). Metric spaces in synthetic topology. *Annals of Pure and Applied Logic* 163 (2) 87–100.
Carchedi, D. (2016). On the homotopy type of higher orbifolds and Haefliger classifying spaces. *Advances in Mathematics*, 294, 756–818. arXiv:1504.02394.
Dubuc, E.J. (1979). Concrete quasitopoi. In: Fourman, M., Mulvey, C. and Scott, D. (eds.) *Applications of Sheaves (Proceedings, Durham 1977)*. Lecture Notes in Mathematics, Springer-Verlag, 239–254.
Dubuc, E.J. and Español, L. (2006). Quasitopoi over a base category. arXiv:math/0612727.

Dubuc, E.J. and Penon, J. (1986). Objets compacts dans les topos. Journal of the Australian Mathematical Society (Series A) 40 (2) 203–217.

Dugger, D. (2001). Universal homotopy theories. Advances in Mathematics 164 (1) 144–176.

Escarfó, M. (2004a). Synthetic topology of data types and classical spaces. In: Electronic Notes in Theoretical Computer Science. Available at http://www.cs.bham.ac.uk/~mhe/papers/entcs87.pdf. Elsevier, p. 2004.

Escarfó, M. (2004b). Topology via higher-order intuitionistic logic. Unfinished draft. Available at http://www.cs.bham.ac.uk/~mhe/papers/index.html.

Escarfó, M.H. and Streicher, T. (2016). The intrinsic topology of Martin-Löf universes. Annals of Pure and Applied Logic 167 (9), 794–805.

Frank, M. (2017). Interpolating between choices for the approximate intermediate value theorem. arXiv:1701.02227.

Gepner, D. and Kock, J. (2012). Univalence in locally cartesian closed (∞,1)-categories. arXiv:1208.1749.

Goodwillie, T.G. (2003). Calculus. III. Taylor series. Geometry & Topology 7 645–711.

HoTT Project (2015). The homotopy type theory coq library. Available at http://github.com/HoTT/HoTT/.

Johnstone, P.T. (1979). On a topological topos. Proceedings of the London Mathematical Society (3) 38 (2) 237–271.

Johnstone, P.T. (2002). Sketches of an Elephant: A Topos Theory Compendium: Volumes 1 and 2. Oxford Logic Guides, vol. 43, Oxford Science Publications.

Johnstone, P.T. (2011). Remarks on punctual local connectedness. Theory and Applications of Categories 25 (3) 51–63.

Joyal, A. (2008). Notes on logoi. Available at http://www.math.uchicago.edu/~may/IMA/Joyal.pdf.

Kapulkin, C. and Lumsdaine, P.L. (2012). The simplicial model of univalent foundations (after Voevodsky). arXiv:1211.2851.

Kraus, N. (2016). Constructions with non-recursive higher inductive types. In: LICS’16.

Lawvere, F.W. (1970). Equality in hyperdoctrines and comprehension schema as an adjoint functor. In: Applications of Categorical Algebra, Providence, R.I.: Amer. Math. Soc., pp. 1–14. Lawvere, F.W. (2007). Axiomatic cohesion. Theory and Applications of Categories 19 (3) 41–49.

Lawvere, F.W. and Menni, M. (2015). Internal choice holds in the discrete part of any cohesive topos satisfying stable connected codiscreteness. Theory Applications of Categories 30 (26) 909–932.

Licata, D. and Finster, E. (2014). Eilenberg–MacLane spaces in homotopy type theory. In: LICS.

Available at http://dlicata.web.wesleyan.edu/pubs/lf14em/lf14em.pdf.

Licata, D.R. and Shulman, M. (2013). Calculating the fundamental group of the circle in homotopy type theory. In: LICS’13. eprint: arXiv:1301.3443.

Licata, D. and Shulman, M. (2016). Adjoint logic with a 2-category of modes. In: LFCS ’16.

Available at http://dlicata.web.wesleyan.edu/pubs/ls15adjoint/ls15adjoint.pdf.

Licata, D.R., Shulman, M. and Riley, M. (2017). A fibrational framework for substructural and modal logics. To appear in FSCD ’17.

Lin, Z. (2014). Answer to MathOverflow question ‘The real numbers object in Sh(Top).’ Available at http://mathoverflow.net/a/186165/49.

Lumsdaine, P.L. and Shulman, M. (2017). Semantics of higher inductive types. arXiv:1705.07088.

Lumsdaine, P.L. and Warren, M.A. (2015). The local universes model: An overlooked coherence construction for dependent type theories. ACM Transactions on Computational Logic 16 (3) 23:1–23:31. arXiv:1411.1736.
Lurie, J. (2009). *Higher Topos Theory*. Annals of Mathematics Studies, vol. 170, Princeton University Press. arXiv:math.CT/0608040.

Lurie, J. (2014). Higher algebra. Available at http://www.math.harvard.edu/~lurie/.

Mac Lane, S. and Moerdijk, I. (1994). *Sheaves in Geometry and Logic: A First Introduction to Topos Theory*. Universitext. Corrected reprint of the 1992 edition. New York: Springer-Verlag.

Menni, M. (2014). Continuous cohesion over sets. *Theory and Applications of Categories* **29** (20) 542–568.

Palmgren, E. (2007a). A constructive and functorial embedding of locally compact metric spaces into locales. *Topology and its Applications* **154** (9) 1854–1880.

Palmgren, E. (2007b). *Resolution of the Uniform Lower Bound Problem in Constructive Analysis*. Tech. rep. 11. Uppsala University Department of Mathematics.

Penon, J. (1985). De l’infinitésimal au local (Thèse de Doctorat d’État). In: *Diagrammes S13*. Available at http://www.numdam.org/item?id=DIA_1985_1_10_0, pp. 1–191.

Pfenning, F. and Davies, R. (2001). A judgmental reconstruction of modal logic. *Mathematical Structures in Computer Science* **11** (4) 511–540.

Reed, J. (2009). A judgmental deconstruction of modal logic. Available at http://www.cs.cmu.edu/~jcreed/papers/jdml.pdf.

Rezk, C. (2014). Global homotopy theory and cohesion. Available at http://www.math.uiuc.edu/~rezk/global-cohesion.pdf.

Rijke, E. (2017). The join construction. arXiv:1701.07538.

Rijke, E., Shulman, M. and Spitters, B. (2017). Modalities in homotopy type theory. arXiv:1706.07526.

Schreiber, U. (2013). Differential cohomology in a cohesive (\infty,1)-topos. Available at http://ncatlab.org/schreiber/show/differential+cohomology+\in+a+cohesive+topos; arXiv:1310.7930.

Schreiber, U. and Shulman, M. (2012). Quantum gauge field theory in cohesive homotopy type theory. In: *QPL’12*. Available at http://ncatlab.org/schreiber/files/QFTinCohesiveHoTT.pdf.

Shulman, M. (2011a). Internalizing the external, or the joys of codiscreteness. Available at https://golem.ph.utexas.edu/category/2011/11/internalizing_the_external_or.html.

Shulman, M. (2011b). Reflective subfibrations, factorization systems, and stable units. Available at https://golem.ph.utexas.edu/category/2011/12/reflective_subfibrations_facto.html.

Streicher, T. (1991). *Semantics of Type Theory: Correctness, Completeness, and Independence Results*. Progress in Theoretical Computer Science, Birkhäuser.

Taylor, P. (2010). A lambda calculus for real analysis. *Journal of Logic & Analysis* **2** (5) 1–115.

Troelstra, A.S. and van Dalen, D. (1988). *Constructivism in Mathematics. Vol. I*. Studies in Logic and the Foundations of Mathematics, vol. 121. Amsterdam: North-Holland Publishing Co., pp. xx+342+XIV.

Univalent Foundations Program (2013). *Homotopy Type Theory: Univalent Foundations of Mathematics*. first edition. Available at http://homotopytypetheory.org/book/.

van Doorn, F. (2016). Constructing the propositional truncation using non-recursive HITs. In: *Certified Programs and Proofs ’16*. arXiv:1512.02274.

Wiry, O. (1991). *Lecture Notes on Topoi and Quasitopoi*. World Scientific.