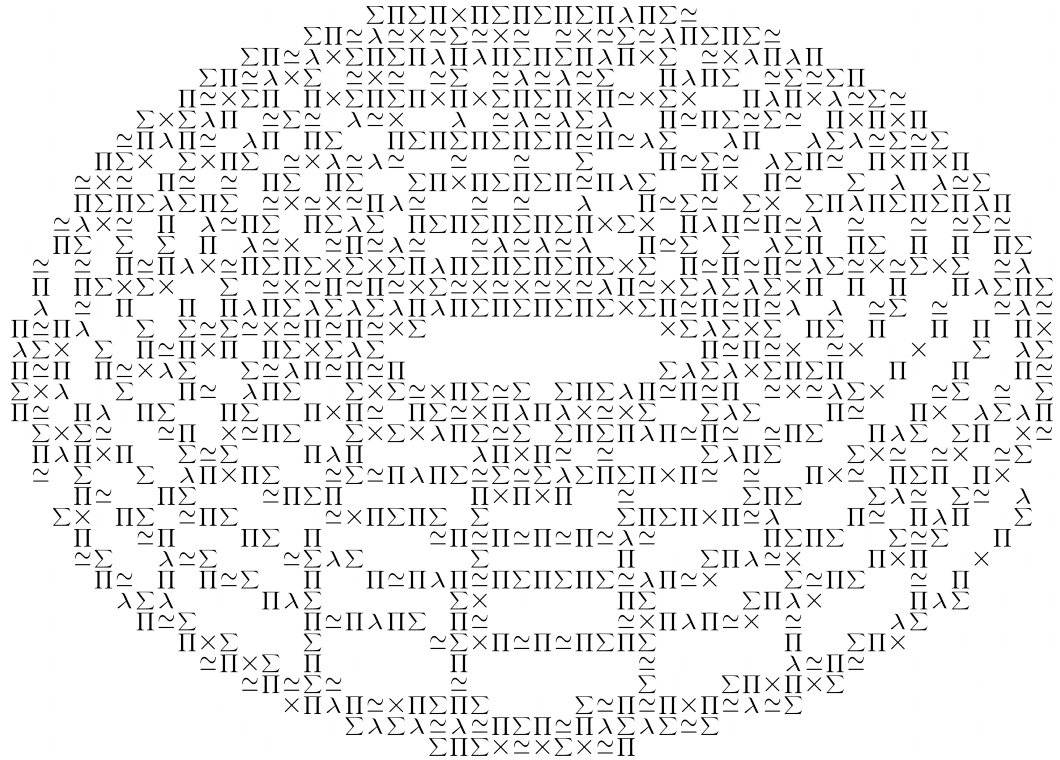
Homotopy Type Theory

Univalent Foundations of Mathematics

The Univalent Foundations Program
Institute for Advanced Study



"Homotopy Type Theory: Univalent Foundations of Mathematics"

Book version: `first-edition-1174-g29279f5`

MSC 2010 classification: `03-02`, `55-02`, `03B15`

This book is freely available at `http://homotopytypetheory.org/book/`.

Acknowledgment

Apart from the generous support from the Institute for Advanced Study, some contributors to the book were partially or fully supported by the following agencies and grants:

- Association of Members of the Institute for Advanced Study: a grant to the Institute for Advanced Study
- Agencija za raziskovalno dejavnost Republike Slovenije: P1–0294, N1–0011.
- Air Force Office of Scientific Research: FA9550-11-1-0143, and FA9550-12-1-0370.

  > This material is based in part upon work supported by the AFOSR under the above awards. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the AFOSR.

- Engineering and Physical Sciences Research Council: EP/G034109/1, EP/G03298X/1.
- European Union's 7th Framework Programme under grant agreement nr. 243847 (ForMath).
- National Science Foundation: DMS-1001191, DMS-1100938, CCF-1116703, and DMS-1128155.

  > This material is based in part upon work supported by the National Science Foundation under the above awards. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

- The Simonyi Fund: a grant to the Institute for Advanced Study

Contents

Part II

Mathematics

Chapter 8

Homotopy theory

In this chapter, we develop some homotopy theory within type theory. We use the *synthetic approach* to homotopy theory introduced in Chapter 2: Spaces, points, paths, and homotopies are basic notions, which are represented by types and elements of types, particularly the identity type. The algebraic structure of paths and homotopies is represented by the natural ∞-groupoid structure on types, which is generated by the rules for the identity type. Using higher inductive types, as introduced in Chapter 6, we can describe spaces directly by their universal properties.

There are several interesting aspects of this synthetic approach. First, it combines advantages of concrete models (such as topological spaces or simplicial sets) with advantages of abstract categorical frameworks for homotopy theory (such as Quillen model categories). On the one hand, our proofs feel elementary, and refer concretely to points, paths, and homotopies in types. On the other hand, our approach nevertheless abstracts away from any concrete presentation of these objects — for example, associativity of path concatenation is proved by path induction, rather than by reparametrization of maps $[0,1] \to X$ or by horn-filling conditions. Type theory seems to be a very convenient way to study the abstract homotopy theory of ∞-groupoids: by using the rules for the identity type, we can avoid the complicated combinatorics involved in many definitions of ∞-groupoids, and explicate only as much of the structure as is needed in any particular proof.

The abstract nature of type theory means that our proofs apply automatically in a variety of settings. In particular, as mentioned previously, homotopy type theory has one interpretation in Kan simplicial sets, which is one model for the homotopy theory of ∞-groupoids. Thus, our proofs apply to this model, and transferring them along the geometric realization functor from simplicial sets to topological spaces gives proofs of corresponding theorems in classical homotopy theory. However, though the details are work in progress, we can also interpret type theory in a wide variety of other categories that look like the category of ∞-groupoids, such as $(\infty,1)$-toposes. Thus, proving a result in type theory will show that it holds in these settings as well. This sort of extra generality is well-known as a property of ordinary categorical logic: univalent foundations extends it to homotopy theory as well.

Second, our synthetic approach has suggested new type-theoretic methods and proofs. Some of our proofs are fairly direct transcriptions of classical proofs. Others have a more type-theoretic feel, and consist mainly of calculations with ∞-groupoid operations, in a style that is very similar to how computer scientists use type theory to reason about computer programs. One thing that seems to have permitted these new proofs is the fact that type theory emphasizes different aspects of homotopy theory than other approaches: while tools like path induction and the

universal properties of higher inductives are available in a setting like Kan simplicial sets, type theory elevates their importance, because they are the *only* primitive tools available for working with these types. Focusing on these tools had led to new descriptions of familiar constructions such as the universal cover of the circle and the Hopf fibration, using just the recursion principles for higher inductive types. These descriptions are very direct, and many of the proofs in this chapter involve computational calculations with such fibrations. Another new aspect of our proofs is that they are constructive (assuming univalence and higher inductives types are constructive); we describe an application of this to homotopy groups of spheres in §8.10.

Third, our synthetic approach is very amenable to computer-checked proofs in proof assistants such as COQ and AGDA. Almost all of the proofs described in this chapter have been computer-checked, and many of these proofs were first given in a proof assistant, and then "unformalized" for this book. The computer-checked proofs are comparable in length and effort to the informal proofs presented here, and in some cases they are even shorter and easier to do.

Before turning to the presentation of our results, we briefly review some basic concepts and theorems from homotopy theory for the benefit of the reader who is not familiar with them. We also give an overview of the results proved in this chapter.

Homotopy theory is a branch of algebraic topology, and uses tools from abstract algebra, such as group theory, to investigate properties of spaces. One question homotopy theorists investigate is how to tell whether two spaces are the same, where "the same" means *homotopy equivalence* (continuous maps back and forth that compose to the identity up to homotopy—this gives the opportunity to "correct" maps that don't exactly compose to the identity). One common way to tell whether two spaces are the same is to calculate *algebraic invariants* associated with a space, which include its *homotopy groups* and *homology* and *cohomology groups*. Equivalent spaces have isomorphic homotopy/(co)homology groups, so if two spaces have different groups, then they are not equivalent. Thus, these algebraic invariants provide global information about a space, which can be used to tell spaces apart, and complements the local information provided by notions such as continuity. For example, the torus locally looks like the 2-sphere, but it has a global difference, because it has a hole in it, and this difference is visible in the homotopy groups of these two spaces.

The simplest example of a homotopy group is the *fundamental group* of a space, which is written $\pi_1(X, x_0)$: Given a space X and a point x_0 in it, one can make a group whose elements are loops at x_0 (continuous paths from x_0 to x_0), considered up to homotopy, with the group operations given by the identity path (standing still), path concatenation, and path reversal. For example, the fundamental group of the 2-sphere is trivial, but the fundamental group of the torus is not, which shows that the sphere and the torus are not homotopy equivalent. The intuition is that every loop on the sphere is homotopic to the identity, because its inside can be filled in. In contrast, a loop on the torus that goes through the donut's hole is not homotopic to the identity, so there are non-trivial elements in the fundamental group.

The *higher homotopy groups* provide additional information about a space. Fix a point x_0 in X, and consider the constant path refl_{x_0}. Then the homotopy classes of homotopies between refl_{x_0} and itself form a group $\pi_2(X, x_0)$, which tells us something about the two-dimensional structure of the space. Then $\pi_3(X, x_0)$ is the group of homotopy classes of homotopies between homotopies, and so on. One of the basic problems of algebraic topology is *calculating the homotopy groups of a space* X, which means giving a group isomorphism between $\pi_k(X, x_0)$ and some more direct description of a group (e.g., by a multiplication table or presentation). Somewhat surprisingly, this is a very difficult question, even for spaces as simple as the spheres. As can be seen

from Table 8.1, some patterns emerge in the higher homotopy groups of spheres, but there is no general formula, and many homotopy groups of spheres are currently still unknown.

	$\mathbb{S}^0$	$\mathbb{S}^1$	$\mathbb{S}^2$	$\mathbb{S}^3$	$\mathbb{S}^4$	$\mathbb{S}^5$	$\mathbb{S}^6$	$\mathbb{S}^7$	$\mathbb{S}^8$
π_1	0	$\mathbb{Z}$	0	0	0	0	0	0	0
π_2	0	0	$\mathbb{Z}$	0	0	0	0	0	0
π_3	0	0	$\mathbb{Z}$	$\mathbb{Z}$	0	0	0	0	0
π_4	0	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0	0	0
π_5	0	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0	0
π_6	0	0	$\mathbb{Z}_{12}$	$\mathbb{Z}_{12}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0
π_7	0	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}\times\mathbb{Z}_{12}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0
π_8	0	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_2^2$	$\mathbb{Z}_{24}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$
π_9	0	0	$\mathbb{Z}_3$	$\mathbb{Z}_3$	$\mathbb{Z}_2^2$	$\mathbb{Z}_2$	$\mathbb{Z}_{24}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$
π_{10}	0	0	$\mathbb{Z}_{15}$	$\mathbb{Z}_{15}$	$\mathbb{Z}_{24}\times\mathbb{Z}_3$	$\mathbb{Z}_2$	0	$\mathbb{Z}_{24}$	$\mathbb{Z}_2$
π_{11}	0	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_{15}$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	$\mathbb{Z}_{24}$
π_{12}	0	0	$\mathbb{Z}_2^2$	$\mathbb{Z}_2^2$	$\mathbb{Z}_2$	$\mathbb{Z}_{30}$	$\mathbb{Z}_2$	0	0
π_{13}	0	0	$\mathbb{Z}_{12}\times\mathbb{Z}_2$	$\mathbb{Z}_{12}\times\mathbb{Z}_2$	$\mathbb{Z}_2^3$	$\mathbb{Z}_2$	$\mathbb{Z}_{60}$	$\mathbb{Z}_2$	0

Table 8.1: Homotopy groups of spheres [Wik13]. The k^{th} homotopy group π_k of the n-dimensional sphere $\mathbb{S}^n$ is isomorphic to the group listed in each entry, where $\mathbb{Z}$ is the additive group of integers, and $\mathbb{Z}_m$ is the cyclic group of order m.

One way of understanding this complexity is through the correspondence between spaces and ∞-groupoids introduced in Chapter 2. As discussed in §6.4, the 2-sphere is presented by a higher inductive type with one point and one 2-dimensional loop. Thus, one might wonder why $\pi_3(\mathbb{S}^2)$ is $\mathbb{Z}$, when the type $\mathbb{S}^2$ has no generators creating 3-dimensional cells. It turns out that the generating element of $\pi_3(\mathbb{S}^2)$ is constructed using the interchange law described in the proof of Theorem 2.1.6: the algebraic structure of an ∞-groupoid includes non-trivial interactions between levels, and these interactions create elements of higher homotopy groups.

Type theory provides a natural setting for investigating this structure, as we can easily define the higher homotopy groups. Recall from Definition 2.1.8 that for $n : \mathbb{N}$, the n-fold iterated loop space of a pointed type (A, a) is defined recursively by:

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

This gives a *space* (i.e. a type) of n-dimensional loops, which itself has higher homotopies. We obtain the set of n-dimensional loops by truncation (this was also defined as an example in §6.11):

Definition 8.0.1 (Homotopy Groups)**.** Given $n \geq 1$ and (A, a) a pointed type, we define the **homotopy groups** of A at a by

$$\pi_n(A, a) :\equiv \left\| \Omega^n(A, a) \right\|_0$$

Since $n \geq 1$, the path concatenation and inversion operations on $\Omega^n(A)$ induce operations on $\pi_n(A)$ making it into a group in a straightforward way. If $n \geq 2$, then the group $\pi_n(A)$ is abelian,

by the Eckmann–Hilton argument (Theorem 2.1.6). It is convenient to also write $\pi_0(A) :\equiv \|A\|_0$, but this case behaves somewhat differently: not only is it not a group, it is defined without reference to any basepoint in A.

This definition is a suitable one for investigating homotopy groups because the (higher) inductive definition of a type X presents X as a free type, analogous to a free ∞-groupoid, and this presentation *determines* but does not *explicitly describe* the higher identity types of X. The identity types are populated by both the generators (loop, for the circle) and the results of applying to them all of the groupoid operations (identity, composition, inverses, associativity, interchange, ...). Thus, the higher-inductive presentation of a space allows us to pose the question "what does the identity type of X really turn out to be?" though it can take some significant mathematics to answer it. This is a higher-dimensional generalization of a familiar fact in type theory: characterizing the identity type of X can take some work, even if X is an ordinary inductive type, such as the natural numbers or booleans. For example, the theorem that $0_{\mathbf{2}}$ is different from $1_{\mathbf{2}}$ does not follow immediately from the definition; see §2.12.

The univalence axiom plays an essential role in calculating homotopy groups (without univalence, type theory is compatible with an interpretation where all paths, including, for example, the loop on the circle, are reflexivity). We will see this in the calculation of the fundamental group of the circle below: the map from $\Omega(\mathbb{S}^1)$ to $\mathbb{Z}$ is defined by mapping a loop on the circle to an automorphism of the set $\mathbb{Z}$, so that, for example, $\mathsf{loop} \cdot \mathsf{loop}^{-1}$ is sent to $\mathsf{successor} \cdot \mathsf{predecessor}$ (where successor and predecessor are automorphisms of $\mathbb{Z}$ viewed, by univalence, as paths in the universe), and then applying the automorphism to 0. Univalence produces non-trivial paths in the universe, and this is used to extract information from paths in higher inductive types.

In this chapter, we first calculate some homotopy groups of spheres, including $\pi_k(\mathbb{S}^1)$ (§8.1), $\pi_k(\mathbb{S}^n)$ for $k < n$ (§§8.2 and 8.3), $\pi_2(\mathbb{S}^2)$ and $\pi_3(\mathbb{S}^2)$ by way of the Hopf fibration (§8.5) and a long-exact-sequence argument (§8.4), and $\pi_n(\mathbb{S}^n)$ by way of the Freudenthal suspension theorem (§8.6). Next, we discuss the van Kampen theorem (§8.7), which characterizes the fundamental group of a pushout, and the status of Whitehead's principle (when is a map that induces an equivalence on all homotopy groups an equivalence?) (§8.8). Finally, we include brief summaries of additional results that are not included in the book, such as $\pi_{n+1}(\mathbb{S}^n)$ for $n \geq 3$, the Blakers–Massey theorem, and a construction of Eilenberg–Mac Lane spaces (§8.10). Prerequisites for this chapter include Chapters 1, 2, 6 and 7 as well as parts of Chapter 3.

8.1 $\pi_1(S^1)$

In this section, our goal is to show that $\pi_1(\mathbb{S}^1) = \mathbb{Z}$. In fact, we will show that the loop space $\Omega(\mathbb{S}^1)$ is equivalent to $\mathbb{Z}$. This is a stronger statement, because $\pi_1(\mathbb{S}^1) = \left\|\Omega(\mathbb{S}^1)\right\|_0$ by definition; so if $\Omega(\mathbb{S}^1) = \mathbb{Z}$, then $\left\|\Omega(\mathbb{S}^1)\right\|_0 = \|\mathbb{Z}\|_0$ by congruence, and $\mathbb{Z}$ is a set by definition (being a set-quotient; see Remarks 6.10.7 and 6.10.11), so $\|\mathbb{Z}\|_0 = \mathbb{Z}$. Moreover, knowing that $\Omega(\mathbb{S}^1)$ is a set will imply that $\pi_n(\mathbb{S}^1)$ is trivial for $n > 1$, so we will actually have calculated *all* the homotopy groups of $\mathbb{S}^1$.

8.1.1 Getting started

It is not too hard to define functions in both directions between $\Omega(\mathbb{S}^1)$ and $\mathbb{Z}$. By specializing Corollary 6.10.13 to $\mathsf{loop} : \mathsf{base} = \mathsf{base}$, we have a function $\mathsf{loop}^- : \mathbb{Z} \to (\mathsf{base} = \mathsf{base})$ defined

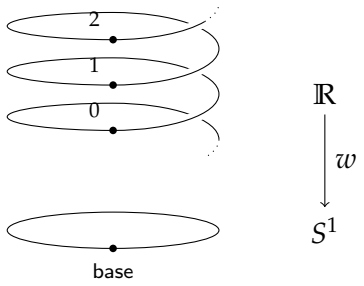


Figure 8.1: The winding map in classical topology

(loosely speaking) by

$$\mathsf{loop}^n = \begin{cases} \underbrace{\mathsf{loop} \cdot \mathsf{loop} \cdot \cdots \cdot \mathsf{loop}}_{n} & \text{if } n > 0, \\ \underbrace{\mathsf{loop}^{-1} \cdot \mathsf{loop}^{-1} \cdot \cdots \cdot \mathsf{loop}^{-1}}_{-n} & \text{if } n < 0, \\ \mathsf{refl}_{\mathsf{base}} & \text{if } n = 0. \end{cases}$$

Defining a function $g : \Omega(\mathbb{S}^1) \to \mathbb{Z}$ in the other direction is a bit trickier. Note that the successor function $\mathsf{succ} : \mathbb{Z} \to \mathbb{Z}$ is an equivalence, and hence induces a path $\mathsf{ua}(\mathsf{succ}) : \mathbb{Z} = \mathbb{Z}$ in the universe $\mathcal{U}$. Thus, the recursion principle of $\mathbb{S}^1$ induces a map $c : \mathbb{S}^1 \to \mathcal{U}$ by $c(\mathsf{base}) :\equiv \mathbb{Z}$ and $\mathsf{ap}_c(\mathsf{loop}) := \mathsf{ua}(\mathsf{succ})$. Then we have $\mathsf{ap}_c : (\mathsf{base} = \mathsf{base}) \to (\mathbb{Z} = \mathbb{Z})$, and we can define $g(p) :\equiv \mathsf{transport}^{X \mapsto X}(\mathsf{ap}_c(p), 0)$.

With these definitions, we can even prove that $g(\mathsf{loop}^n) = n$ for any $n : \mathbb{Z}$, using the induction principle Lemma 6.10.12 for n. (We will prove something more general a little later on.) However, the other equality $\mathsf{loop}^{g(p)} = p$ is significantly harder. The obvious thing to try is path induction, but path induction does not apply to loops such as $p : (\mathsf{base} = \mathsf{base})$ that have *both* endpoints fixed! A new idea is required, one which can be explained both in terms of classical homotopy theory and in terms of type theory. We begin with the former.

8.1.2 The classical proof

In classical homotopy theory, there is a standard proof of $\pi_1(\mathbb{S}^1) = \mathbb{Z}$ using universal covering spaces. Our proof can be regarded as a type-theoretic version of this proof, with covering spaces appearing here as fibrations whose fibers are sets. Recall that *fibrations* over a space B in homotopy theory correspond to type families $B \to \mathcal{U}$ in type theory. In particular, for a point $x_0 : B$, the type family $(x \mapsto (x_0 = x))$ corresponds to the *path fibration* $P_{x_0}B \to B$, in which the points of $P_{x_0}B$ are paths in B starting at x_0, and the map to B selects the other endpoint of such a path. This total space $P_{x_0}B$ is contractible, since we can "retract" any path to its initial endpoint x_0 — we have seen the type-theoretic version of this as Lemma 3.11.8. Moreover, the fiber over x_0 is the loop space $\Omega(B, x_0)$ — in type theory this is obvious by definition of the loop space.

Now in classical homotopy theory, where $\mathbb{S}^1$ is regarded as a topological space, we may proceed as follows. Consider the "winding" map $w : \mathbb{R} \to \mathbb{S}^1$, which looks like a helix projecting down onto the circle (see Figure 8.1). This map w sends each point on the helix to the point on the circle that it is "sitting above". It is a fibration, and the fiber over each point is isomorphic to the integers. If we lift the path that goes counterclockwise around the loop on the bottom,

we go up one level in the helix, incrementing the integer in the fiber. Similarly, going clockwise around the loop on the bottom corresponds to going down one level in the helix, decrementing this count. This fibration is called the *universal cover* of the circle.

Now a basic fact in classical homotopy theory is that a map $E_1 \to E_2$ of fibrations over B which is a homotopy equivalence between E_1 and E_2 induces a homotopy equivalence on all fibers. (We have already seen the type-theoretic version of this as well in Theorem 4.7.7.) Since $\mathbb{R}$ and $P_{\mathsf{base}}S^1$ are both contractible topological spaces, they are homotopy equivalent, and thus their fibers $\mathbb{Z}$ and $\Omega(\mathbb{S}^1)$ over the basepoint are also homotopy equivalent.

8.1.3 The universal cover in type theory

Let us consider how we might express the preceding proof in type theory. We have already remarked that the path fibration of $\mathbb{S}^1$ is represented by the type family $(x \mapsto (\mathsf{base} = x))$. We have also already seen a good candidate for the universal cover of $\mathbb{S}^1$: it's none other than the type family $c : \mathbb{S}^1 \to \mathcal{U}$ which we defined in §8.1.1! By definition, the fiber of this family over base is $\mathbb{Z}$, while the effect of transporting around loop is to add one — thus it behaves just as we would expect from Figure 8.1.

However, since we don't know yet that this family behaves like a universal cover is supposed to (for instance, that its total space is simply connected), we use a different name for it. For reference, therefore, we repeat the definition.

Definition 8.1.1 (Universal Cover of $\mathbb{S}^1$)**.** Define $\mathsf{code} : \mathbb{S}^1 \to \mathcal{U}$ by circle-recursion, with

$$\begin{aligned} \mathsf{code}(\mathsf{base}) &:\equiv \mathbb{Z} \\ \mathsf{ap}_{\mathsf{code}}(\mathsf{loop}) &:= \mathsf{ua}(\mathsf{succ}). \end{aligned}$$

We emphasize briefly the definition of this family, since it is so different from how one usually defines covering spaces in classical homotopy theory. To define a function by circle recursion, we need to find a point and a loop in the codomain. In this case, the codomain is $\mathcal{U}$, and the point we choose is $\mathbb{Z}$, corresponding to our expectation that the fiber of the universal cover should be the integers. The loop we choose is the successor/predecessor isomorphism on $\mathbb{Z}$, which corresponds to the fact that going around the loop in the base goes up one level on the helix. Univalence is necessary for this part of the proof, because we need to convert a *non-trivial* equivalence on $\mathbb{Z}$ into an identity.

We call this the fibration of "codes", because its elements are combinatorial data that act as codes for paths on the circle: the integer n codes for the path which loops around the circle n times.

From this definition, it is simple to calculate that transporting with code takes loop to the successor function, and loop^{-1} to the predecessor function:

Lemma 8.1.2. $\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}, x) = x + 1$ *and* $\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}^{-1}, x) = x - 1$.

Proof. For the first equation, we calculate as follows:

$$\begin{aligned} \mathsf{transport}^{\mathsf{code}}(\mathsf{loop}, x) &= \mathsf{transport}^{A \mapsto A}((\mathsf{code}(\mathsf{loop})), x) && \text{(by Lemma 2.3.10)} \\ &= \mathsf{transport}^{A \mapsto A}(\mathsf{ua}(\mathsf{succ}), x) && \text{(by computation for } \mathsf{rec}_{\mathbb{S}^1}) \\ &= x + 1. && \text{(by computation for } \mathsf{ua}) \end{aligned}$$

The second equation follows from the first, because $\mathsf{transport}^B(p, -)$ and $\mathsf{transport}^B(p^{-1}, -)$ are always inverses, so $\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}^{-1}, -)$ must be the inverse of succ. □

We can now see what was wrong with our first approach: we defined f and g only on the fibers $\Omega(\mathbb{S}^1)$ and $\mathbb{Z}$, when we should have defined a whole morphism *of fibrations* over $\mathbb{S}^1$. In type theory, this means we should have defined functions having types

$$\prod_{x:\mathbb{S}^1} ((\mathsf{base} = x) \to \mathsf{code}(x)) \qquad \text{and/or} \tag{8.1.3}$$

$$\prod_{x:\mathbb{S}^1} (\mathsf{code}(x) \to (\mathsf{base} = x)) \tag{8.1.4}$$

instead of only the special cases of these when x is base. This is also an instance of a common observation in type theory: when attempting to prove something about particular inhabitants of some inductive type, it is often easier to generalize the statement so that it refers to *all* inhabitants of that type, which we can then prove by induction. Looked at in this way, the proof of $\Omega(\mathbb{S}^1) = \mathbb{Z}$ fits into the same pattern as the characterization of the identity types of coproducts and natural numbers in §§2.12 and 2.13.

At this point, there are two ways to finish the proof. We can continue mimicking the classical argument by constructing (8.1.3) or (8.1.4) (it doesn't matter which), proving that a homotopy equivalence between total spaces induces an equivalence on fibers, and then that the total space of the universal cover is contractible. The first type-theoretic proof of $\Omega(\mathbb{S}^1) = \mathbb{Z}$ followed this pattern; we call it the *homotopy-theoretic* proof.

Later, however, we discovered that there is an alternative proof, which has a more type-theoretic feel and more closely follows the proofs in §§2.12 and 2.13. In this proof, we directly construct both (8.1.3) and (8.1.4), and prove that they are mutually inverse by calculation. We will call this the *encode-decode* proof, because we call the functions (8.1.3) and (8.1.4) *encode* and *decode* respectively. Both proofs use the same construction of the cover given above. Where the classical proof induces an equivalence on fibers from an equivalence between total spaces, the encode-decode proof constructs the inverse map (*decode*) explicitly as a map between fibers. And where the classical proof uses contractibility, the encode-decode proof uses path induction, circle induction, and integer induction. These are the same tools used to prove contractibility—indeed, path induction *is* essentially contractibility of the path fibration composed with $\mathsf{transport}$—but they are applied in a different way.

Since this is a book about homotopy type theory, we present the encode-decode proof first. A homotopy theorist who gets lost is encouraged to skip to the homotopy-theoretic proof (§8.1.5).

8.1.4 The encode-decode proof

We begin with the function (8.1.3) that maps paths to codes:

Definition 8.1.5. Define $\mathsf{encode} : \prod_{(x:\mathbb{S}^1)} (\mathsf{base} = x) \to \mathsf{code}(x)$ by

$$\mathsf{encode}\ p :\equiv \mathsf{transport}^{\mathsf{code}}(p, 0)$$

(we leave the argument x implicit).

Encode is defined by lifting a path into the universal cover, which determines an equivalence, and then applying the resulting equivalence to 0. The interesting thing about this function is that it computes a concrete number from a loop on the circle, when this loop is represented using the abstract groupoidal framework of homotopy type theory. To gain an intuition for how it does this, observe that by the above lemmas, $\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}, x)$ is the successor map and

$\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}^{-1}, x)$ is the predecessor map. Further, transport is functorial (Chapter 2), so $\mathsf{transport}^{\mathsf{code}}(\mathsf{loop} \cdot \mathsf{loop}, -)$ is

$$(\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}, -)) \circ (\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}, -))$$

and so on. Thus, when p is a composition like

$$\mathsf{loop} \cdot \mathsf{loop}^{-1} \cdot \mathsf{loop} \cdot \cdots$$

$\mathsf{transport}^{\mathsf{code}}(p, -)$ will compute a composition of functions like

$$\mathsf{succ} \circ \mathsf{pred} \circ \mathsf{succ} \circ \cdots$$

Applying this composition of functions to 0 will compute the *winding number* of the path—how many times it goes around the circle, with orientation marked by whether it is positive or negative, after inverses have been canceled. Thus, the computational behavior of encode follows from the reduction rules for higher-inductive types and univalence, and the action of transport on compositions and inverses.

Note that the instance $\mathsf{encode}' :\equiv \mathsf{encode}_{\mathsf{base}}$ has type $(\mathsf{base} = \mathsf{base}) \to \mathbb{Z}$. This will be one half of our desired equivalence; indeed, it is exactly the function g defined in §8.1.1.

Similarly, the function (8.1.4) is a generalization of the function loop^- from §8.1.1.

Definition 8.1.6. Define $\mathsf{decode} : \prod_{(x:\mathbb{S}^1)} \mathsf{code}(x) \to (\mathsf{base} = x)$ by circle induction on x. It suffices to give a function $\mathsf{code}(\mathsf{base}) \to (\mathsf{base} = \mathsf{base})$, for which we use loop^-, and to show that loop^- respects the loop.

Proof. To show that loop^- respects the loop, it suffices to give a path from loop^- to itself that lies over loop. By the definition of dependent paths, this means a path from

$$\mathsf{transport}^{(x' \mapsto \mathsf{code}(x') \to (\mathsf{base} = x'))}(\mathsf{loop}, \mathsf{loop}^-)$$

to loop^-. We define such a path as follows:

$$\begin{aligned}
&\mathsf{transport}^{(x' \mapsto \mathsf{code}(x') \to (\mathsf{base} = x'))}(\mathsf{loop}, \mathsf{loop}^-) \\
&\quad = \mathsf{transport}^{x' \mapsto (\mathsf{base} = x')}(\mathsf{loop}) \circ \mathsf{loop}^- \circ \mathsf{transport}^{\mathsf{code}}(\mathsf{loop}^{-1}) \\
&\quad = (- \cdot \mathsf{loop}) \circ (\mathsf{loop}^-) \circ \mathsf{transport}^{\mathsf{code}}(\mathsf{loop}^{-1}) \\
&\quad = (- \cdot \mathsf{loop}) \circ (\mathsf{loop}^-) \circ \mathsf{pred} \\
&\quad = (n \mapsto \mathsf{loop}^{n-1} \cdot \mathsf{loop}).
\end{aligned}$$

On the first line, we apply the characterization of transport when the outer connective of the fibration is $\to$, which reduces the transport to pre- and post-composition with transport at the domain and codomain types. On the second line, we apply the characterization of transport when the type family is $x \mapsto \mathsf{base} = x$, which is post-composition of paths. On the third line, we use the action of code on loop^{-1} from Lemma 8.1.2. And on the fourth line, we simply reduce the function composition. Thus, it suffices to show that for all n, $\mathsf{loop}^{n-1} \cdot \mathsf{loop} = \mathsf{loop}^n$. This is an easy application of Lemma 6.10.12, using the groupoid laws. □

We can now show that encode and decode are quasi-inverses. What used to be the difficult direction is now easy!

Lemma 8.1.7. *For all* $x : \mathbb{S}^1$ *and* $p : \mathsf{base} = x$, $\mathsf{decode}_x(\mathsf{encode}_x(p)) = p$.

Proof. By path induction, it suffices to show that $\mathsf{decode}_{\mathsf{base}}(\mathsf{encode}_{\mathsf{base}}(\mathsf{refl}_{\mathsf{base}})) = \mathsf{refl}_{\mathsf{base}}$. But $\mathsf{encode}_{\mathsf{base}}(\mathsf{refl}_{\mathsf{base}}) \equiv \mathsf{transport}^{\mathsf{code}}(\mathsf{refl}_{\mathsf{base}}, 0) \equiv 0$, and $\mathsf{decode}_{\mathsf{base}}(0) \equiv \mathsf{loop}^0 \equiv \mathsf{refl}_{\mathsf{base}}$. □

The other direction is not much harder.

Lemma 8.1.8. *For all* $x : \mathbb{S}^1$ *and* $c : \mathsf{code}(x)$, *we have* $\mathsf{encode}_x(\mathsf{decode}_x(c)) = c$.

Proof. The proof is by circle induction. It suffices to show the case for base, because the case for loop is a path between paths in $\mathbb{Z}$, which is immediate because $\mathbb{Z}$ is a set.

Thus, it suffices to show, for all $n : \mathbb{Z}$, that

$$\mathsf{encode}'(\mathsf{loop}^n) = n.$$

The proof is by induction, using Lemma 6.10.12.

- In the case for 0, the result is true by definition.
- In the case for $n + 1$,

$$\begin{aligned}
\mathsf{encode}'(\mathsf{loop}^{n+1}) &= \mathsf{encode}'(\mathsf{loop}^n \cdot \mathsf{loop}) && \text{(by definition of } \mathsf{loop}^-) \\
&= \mathsf{transport}^{\mathsf{code}}((\mathsf{loop}^n \cdot \mathsf{loop}), 0) && \text{(by definition of encode)} \\
&= \mathsf{transport}^{\mathsf{code}}(\mathsf{loop}, (\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}^n, 0))) && \text{(by functoriality)} \\
&= (\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}^n, 0)) + 1 && \text{(by Lemma 8.1.2)} \\
&= n + 1. && \text{(by the inductive hypothesis)}
\end{aligned}$$

- The case for negatives is analogous. □

Finally, we conclude the theorem.

Theorem 8.1.9. *There is a family of equivalences* $\prod_{(x:\mathbb{S}^1)}((\mathsf{base} = x) \simeq \mathsf{code}(x))$.

Proof. The maps encode and decode are quasi-inverses by Lemmas 8.1.7 and 8.1.8. □

Instantiating at base gives

Corollary 8.1.10. $\Omega(\mathbb{S}^1, \mathsf{base}) \simeq \mathbb{Z}$.

A simple induction shows that this equivalence takes addition to composition, so that $\Omega(\mathbb{S}^1) = \mathbb{Z}$ as groups.

Corollary 8.1.11. $\pi_1(\mathbb{S}^1) = \mathbb{Z}$, *while* $\pi_n(\mathbb{S}^1) = 0$ *for* $n > 1$.

Proof. For $n = 1$, we sketched the proof from Corollary 8.1.10 above. For $n > 1$, we have $\left\|\Omega^n(\mathbb{S}^1)\right\|_0 = \left\|\Omega^{n-1}(\Omega\mathbb{S}^1)\right\|_0 = \left\|\Omega^{n-1}(\mathbb{Z})\right\|_0$. And since $\mathbb{Z}$ is a set, $\Omega^{n-1}(\mathbb{Z})$ is contractible, so this is trivial. □

8.1.5 The homotopy-theoretic proof

In §8.1.3, we defined the putative universal cover $\mathsf{code} : \mathbb{S}^1 \to \mathcal{U}$ in type theory, and in §8.1.5 we defined a map $\mathsf{encode} : \prod_{(x:\mathbb{S}^1)}(\mathsf{base} = x) \to \mathsf{code}(x)$ from the path fibration to the universal cover. What remains for the classical proof is to show that this map induces an equivalence on total spaces because both are contractible, and to deduce from this that it must be an equivalence on each fiber.

In Lemma 3.11.8 we saw that the total space $\sum_{(x:\mathbb{S}^1)}(\mathsf{base} = x)$ is contractible. For the other, we have:

Lemma 8.1.12. *The type $\sum_{(x:\mathbb{S}^1)} \mathsf{code}(x)$ is contractible.*

Proof. We apply the flattening lemma (Lemma 6.12.2) with the following values:

- $A :\equiv \mathbf{1}$ and $B :\equiv \mathbf{1}$, with f and g the obvious functions. Thus, the base higher inductive type W in the flattening lemma is equivalent to $\mathbb{S}^1$.
- $C : A \to \mathcal{U}$ is constant at $\mathbb{Z}$.
- $D : \prod_{(b:B)}(\mathbb{Z} \simeq \mathbb{Z})$ is constant at succ.

Then the type family $P : \mathbb{S}^1 \to \mathcal{U}$ defined in the flattening lemma is equivalent to $\mathsf{code} : \mathbb{S}^1 \to \mathcal{U}$. Thus, the flattening lemma tells us that $\sum_{(x:\mathbb{S}^1)} \mathsf{code}(x)$ is equivalent to a higher inductive type with the following generators, which we denote R:

- A function $\mathsf{c} : \mathbb{Z} \to R$.
- For each $z : \mathbb{Z}$, a path $\mathsf{p}_z : \mathsf{c}(z) = \mathsf{c}(\mathsf{succ}(z))$.

We might call this type the **homotopical reals**; it plays the same role as the topological space $\mathbb{R}$ in the classical proof.

Thus, it remains to show that R is contractible. As center of contraction we choose $\mathsf{c}(0)$; we must now show that $x = \mathsf{c}(0)$ for all $x : R$. We do this by induction on R. Firstly, when x is $\mathsf{c}(z)$, we must give a path $q_z : \mathsf{c}(0) = \mathsf{c}(z)$, which we can do by induction on $z : \mathbb{Z}$, using Lemma 6.10.12:

$$\begin{aligned} q_0 &:\equiv \mathsf{refl}_{\mathsf{c}(0)} \\ q_{n+1} &:\equiv q_n \cdot \mathsf{p}_n && \text{for } n \geq 0 \\ q_{n-1} &:\equiv q_n \cdot {\mathsf{p}_{n-1}}^{-1} && \text{for } n \leq 0. \end{aligned}$$

Secondly, we must show that for any $z : \mathbb{Z}$, the path q_z is transported along p_z to q_{z+1}. By transport of paths, this means we want $q_z \cdot \mathsf{p}_z = q_{z+1}$. This is easy by induction on z, using the definition of q_z. This completes the proof that R is contractible, and thus so is $\sum_{(x:\mathbb{S}^1)} \mathsf{code}(x)$. □

Corollary 8.1.13. *The map induced by* encode*:*

$$\textstyle\sum_{(x:\mathbb{S}^1)}(\mathsf{base} = x) \to \sum_{(x:\mathbb{S}^1)}\mathsf{code}(x)$$

is an equivalence.

Proof. Both types are contractible. □

Theorem 8.1.14. $\Omega(\mathbb{S}^1, \mathsf{base}) \simeq \mathbb{Z}$.

Proof. Apply Theorem 4.7.7 to encode, using Corollary 8.1.13. □

In essence, the two proofs are not very different: the encode-decode one may be seen as a "reduction" or "unpackaging" of the homotopy-theoretic one. Each has its advantages; the interplay between the two points of view is part of the interest of the subject.

8.1.6 The universal cover as an identity system

Note that the fibration $\mathsf{code} : \mathbb{S}^1 \to \mathcal{U}$ together with $0 : \mathsf{code}(\mathsf{base})$ is a *pointed predicate* in the sense of Definition 5.8.1. From this point of view, we can see that the encode-decode proof in §8.1.4 consists of proving that code satisfies Theorem 5.8.2(iii), while the homotopy-theoretic proof in §8.1.5 consists of proving that it satisfies Theorem 5.8.2(iv). This suggests a third approach.

Theorem 8.1.15. *The pair* $(\mathsf{code}, 0)$ *is an identity system at* $\mathsf{base} : \mathbb{S}^1$ *in the sense of Definition 5.8.1.*

Proof. Let $D : \prod_{(x:\mathbb{S}^1)} \mathsf{code}(x) \to \mathcal{U}$ and $d : D(\mathsf{base}, 0)$ be given; we want to define a function $f : \prod_{(x:\mathbb{S}^1)} \prod_{(c:\mathsf{code}(x))} D(x, c)$. By circle induction, it suffices to specify $f(\mathsf{base}) : \prod_{(c:\mathsf{code}(\mathsf{base}))} D(\mathsf{base}, c)$ and verify that $\mathsf{loop}_*(f(\mathsf{base})) = f(\mathsf{base})$.

Of course, $\mathsf{code}(\mathsf{base}) \equiv \mathbb{Z}$. By Lemma 8.1.2 and induction on n, we may obtain a path $p_n : \mathsf{transport}^{\mathsf{code}}(\mathsf{loop}^n, 0) = n$ for any integer n. Therefore, by paths in Σ-types, we have a path $\mathsf{pair}^=(\mathsf{loop}^n, p_n) : (\mathsf{base}, 0) = (\mathsf{base}, n)$ in $\sum_{(x:\mathbb{S}^1)} \mathsf{code}(x)$. Transporting d along this path in the fibration $\widehat{D} : (\sum_{(x:\mathbb{S}^1)} \mathsf{code}(x)) \to \mathcal{U}$ associated to D, we obtain an element of $D(\mathsf{base}, n)$ for any $n : \mathbb{Z}$. We define this element to be $f(\mathsf{base})(n)$:

$$f(\mathsf{base})(n) :\equiv \mathsf{transport}^{\widehat{D}}(\mathsf{pair}^=(\mathsf{loop}^n, p_n), d).$$

Now we need $\mathsf{transport}^{\lambda x.\, \prod_{(c:\mathsf{code}(x))} D(x,c)}(\mathsf{loop}, f(\mathsf{base})) = f(\mathsf{base})$. By Lemma 2.9.7, this means we need to show that for any $n : \mathbb{Z}$,

$$\mathsf{transport}^{\widehat{D}}(\mathsf{pair}^=(\mathsf{loop}, \mathsf{refl}_{\mathsf{loop}_*(n)}), f(\mathsf{base})(n)) =_{D(\mathsf{base}, \mathsf{loop}_*(n))} f(\mathsf{base})(\mathsf{loop}_*(n)).$$

Now we have a path $q : \mathsf{loop}_*(n) = n + 1$, so transporting along this, it suffices to show

$$\mathsf{transport}^{D(\mathsf{base})}(q, \mathsf{transport}^{\widehat{D}}(\mathsf{pair}^=(\mathsf{loop}, \mathsf{refl}_{\mathsf{loop}_*(n)}), f(\mathsf{base})(n))) \\ =_{D(\mathsf{base}, n+1)} \mathsf{transport}^{D(\mathsf{base})}(q, f(\mathsf{base})(\mathsf{loop}_*(n))).$$

By a couple of lemmas about transport and dependent application, this is equivalent to

$$\mathsf{transport}^{\widehat{D}}(\mathsf{pair}^=(\mathsf{loop}, q), f(\mathsf{base})(n)) =_{D(\mathsf{base}, n+1)} f(\mathsf{base})(n+1).$$

However, expanding out the definition of $f(\mathsf{base})$, we have

$$\begin{aligned} \mathsf{transport}^{\widehat{D}}(\mathsf{pair}^=(\mathsf{loop}, q), f(\mathsf{base})(n)) &= \mathsf{transport}^{\widehat{D}}(\mathsf{pair}^=(\mathsf{loop}, q), \mathsf{transport}^{\widehat{D}}(\mathsf{pair}^=(\mathsf{loop}^n, p_n), d)) \\ &= \mathsf{transport}^{\widehat{D}}(\mathsf{pair}^=(\mathsf{loop}^n, p_n) \cdot \mathsf{pair}^=(\mathsf{loop}, q), d) \\ &= \mathsf{transport}^{\widehat{D}}(\mathsf{pair}^=(\mathsf{loop}^{n+1}, p_{n+1}), d) \\ &= f(\mathsf{base})(n+1). \end{aligned}$$

We have used the functoriality of transport, the characterization of composition in Σ-types (which was an exercise for the reader), and a lemma relating p_n and q to p_{n+1} which we leave it to the reader to state and prove.

This completes the construction of $f : \prod_{(x:\mathbb{S}^1)} \prod_{(c:\mathsf{code}(x))} D(x, c)$. Since

$$f(\mathsf{base}, 0) \equiv \mathsf{pair}^=(\mathsf{loop}^0, p_0)_*(d) = \mathsf{refl}_{\mathsf{base}*}(d) = d,$$

we have shown that $(\mathsf{code}, 0)$ is an identity system. □

Corollary 8.1.16. *For any $x : \mathbb{S}^1$, we have $(\mathsf{base} = x) \simeq \mathsf{code}(x)$.*

Proof. By Theorem 5.8.2. □

Of course, this proof also contains essentially the same elements as the previous two. Roughly, we can say that it unifies the proofs of Definition 8.1.6 and Lemma 8.1.8, performing the requisite inductive argument only once in a generic case.

Remark 8.1.17. Note that all of the above proofs that $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$ use the univalence axiom in an essential way. This is unavoidable: univalence or something like it is *necessary* in order to prove $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$. In the absence of univalence, it is consistent to assume the statement "all types are sets" (a.k.a. "uniqueness of identity proofs" or "Axiom K", as discussed in §7.2), and this statement implies instead that $\pi_1(\mathbb{S}^1) \simeq \mathbf{1}$. In fact, the (non)triviality of $\pi_1(\mathbb{S}^1)$ detects exactly whether all types are sets: the proof of Lemma 6.4.1 showed conversely that if $\mathsf{loop} = \mathsf{refl}_{\mathsf{base}}$ then all types are sets.

8.2 Connectedness of suspensions

Recall from §7.5 that a type A is called **n-connected** if $\|A\|_n$ is contractible. The aim of this section is to prove that the operation of suspension from §6.5 increases connectedness.

Theorem 8.2.1. *If A is n-connected then the suspension of A is $(n+1)$-connected.*

Proof. We remarked in §6.8 that the suspension of A is the pushout $\mathbf{1} \sqcup^A \mathbf{1}$, so we need to prove that the following type is contractible:

$$\left\| \mathbf{1} \sqcup^A \mathbf{1} \right\|_{n+1}.$$

By Theorem 7.4.12 we know that $\left\| \mathbf{1} \sqcup^A \mathbf{1} \right\|_{n+1}$ is a pushout in $(n+1)$-Type of the diagram

$$\begin{array}{ccc} \|A\|_{n+1} & \longrightarrow & \|\mathbf{1}\|_{n+1} \\ \downarrow & & \\ \|\mathbf{1}\|_{n+1} & & \end{array}.$$

Given that $\|\mathbf{1}\|_{n+1} = \mathbf{1}$, the type $\left\| \mathbf{1} \sqcup^A \mathbf{1} \right\|_{n+1}$ is also a pushout of the following diagram in $(n+1)$-Type (because both diagrams are equal)

$$\mathscr{D} = \begin{array}{ccc} \|A\|_{n+1} & \longrightarrow & \mathbf{1} \\ \downarrow & & \\ \mathbf{1} & & \end{array}.$$

We will now prove that $\mathbf{1}$ is also a pushout of $\mathscr{D}$ in $(n+1)$-Type. Let E be an $(n+1)$-truncated type; we need to prove that the following map is an equivalence

$$\begin{cases} (\mathbf{1} \to E) & \longrightarrow & \mathsf{cocone}_{\mathscr{D}}(E) \\ y & \longmapsto & (y, y, \lambda u.\, \mathsf{refl}_{y(\star)}) \end{cases}.$$

where we recall that $\mathsf{cocone}_{\mathscr{D}}(E)$ is the type

$$\sum_{(f : \mathbf{1} \to E)} \sum_{(g : \mathbf{1} \to E)} (\|A\|_{n+1} \to (f(\star) =_E g(\star))).$$

The map $\left\{\begin{array}{rcl}(\mathbf{1}\to E) & \longrightarrow & E\\ f & \longmapsto & f(\star)\end{array}\right.$ is an equivalence, hence we also have

$$\mathsf{cocone}_{\mathscr{D}}(E) = \sum_{(x:E)}\sum_{(y:E)}(\|A\|_{n+1}\to(x =_E y)).$$

Now A is n-connected hence so is $\|A\|_{n+1}$ because $\left\|\|A\|_{n+1}\right\|_n = \|A\|_n = \mathbf{1}$, and $(x =_E y)$ is n-truncated because E is $(n+1)$-truncated. Hence by Corollary 7.5.9 the following map is an equivalence

$$\left\{\begin{array}{rcl}(x =_E y) & \longrightarrow & (\|A\|_{n+1}\to(x =_E y))\\ p & \longmapsto & \lambda z.\, p\end{array}\right.$$

Hence we have

$$\mathsf{cocone}_{\mathscr{D}}(E) = \sum_{(x:E)}\sum_{(y:E)}(x =_E y).$$

But the following map is an equivalence

$$\left\{\begin{array}{rcl}E & \longrightarrow & \sum_{(x:E)}\sum_{(y:E)}(x =_E y)\\ x & \longmapsto & (x, x, \mathsf{refl}_x)\end{array}\right.\ .$$

Hence

$$\mathsf{cocone}_{\mathscr{D}}(E) = E.$$

Finally we get an equivalence

$$(\mathbf{1}\to E)\simeq\mathsf{cocone}_{\mathscr{D}}(E)$$

We can now unfold the definitions in order to get the explicit expression of this map, and we see easily that this is exactly the map we had at the beginning.

Hence we proved that $\mathbf{1}$ is a pushout of $\mathscr{D}$ in $(n+1)$-Type. Using uniqueness of pushouts we get that $\left\|\mathbf{1}\sqcup^A\mathbf{1}\right\|_{n+1} = \mathbf{1}$ which proves that the suspension of A is $(n+1)$-connected. □

Corollary 8.2.2. *For all $n:\mathbb{N}$, the sphere $\mathbb{S}^n$ is $(n-1)$-connected.*

Proof. We prove this by induction on n. For $n=0$ we have to prove that $\mathbb{S}^0$ is merely inhabited, which is clear. Let $n:\mathbb{N}$ be such that $\mathbb{S}^n$ is $(n-1)$-connected. By definition $\mathbb{S}^{n+1}$ is the suspension of $\mathbb{S}^n$, hence by the previous lemma $\mathbb{S}^{n+1}$ is n-connected. □

8.3 $\pi_{k\leq n}$ of an n-connected space and $\pi_{k<n}(\mathbb{S}^n)$

Let (A,a) be a pointed type and $n:\mathbb{N}$. Recall from Example 6.11.4 that if $n>0$ the set $\pi_n(A,a)$ has a group structure, and if $n>1$ the group is abelian.

We can now say something about homotopy groups of n-truncated and n-connected types.

Lemma 8.3.1. *If A is n-truncated and $a:A$, then $\pi_k(A,a)=\mathbf{1}$ for all $k>n$.*

Proof. The loop space of an n-type is an $(n-1)$-type, hence $\Omega^k(A,a)$ is an $(n-k)$-type, and we have $(n-k)\leq -1$ so $\Omega^k(A,a)$ is a mere proposition. But $\Omega^k(A,a)$ is inhabited, so it is actually contractible and $\pi_k(A,a) = \left\|\Omega^k(A,a)\right\|_0 = \|\mathbf{1}\|_0 = \mathbf{1}$. □

Lemma 8.3.2. *If A is n-connected and $a:A$, then $\pi_k(A,a)=\mathbf{1}$ for all $k\leq n$.*

Proof. We have the following sequence of equalities:

$$\pi_k(A,a) = \left\|\Omega^k(A,a)\right\|_0 = \Omega^k(\|(A,a)\|_k) = \Omega^k(\|\|(A,a)\|_n\|_k) = \Omega^k(\|\mathbf{1}\|_k) = \Omega^k(\mathbf{1}) = \mathbf{1}.$$

The third equality uses the fact that $k \leq n$ in order to use that $\|-\|_k \circ \|-\|_n = \|-\|_k$ and the fourth equality uses the fact that A is n-connected. □

Corollary 8.3.3. $\pi_k(\mathbb{S}^n) = \mathbf{1}$ *for* $k < n$.

Proof. The sphere $\mathbb{S}^n$ is $(n-1)$-connected by Corollary 8.2.2, so we can apply Lemma 8.3.2. □

8.4 Fiber sequences and the long exact sequence

If the codomain of a function $f : X \to Y$ is equipped with a basepoint $y_0 : Y$, then we refer to the fiber $F :\equiv \mathsf{fib}_f(y_0)$ of f over y_0 as **the fiber of** f. (If Y is connected, then F is determined up to mere equivalence; see Exercise 8.5.) We now show that if X is also pointed and f preserves basepoints, then there is a relation between the homotopy groups of F, X, and Y in the form of a *long exact sequence*. We derive this by way of the *fiber sequence* associated to such an f.

Definition 8.4.1. A **pointed map** between pointed types (X, x_0) and (Y, y_0) is a map $f : X \to Y$ together with a path $f_0 : f(x_0) = y_0$.

For any pointed types (X, x_0) and (Y, y_0), there is a pointed map $(\lambda x.\, y_0) : X \to Y$ which is constant at the basepoint. We call this the **zero map** and sometimes write it as $0 : X \to Y$.

Recall that every pointed type (X, x_0) has a loop space $\Omega(X, x_0)$. We now note that this operation is functorial on pointed maps.

Definition 8.4.2. Given a pointed map between pointed types $f : X \to Y$, we define a pointed map $\Omega f : \Omega X \to \Omega Y$ by

$$(\Omega f)(p) :\equiv {f_0}^{-1} \cdot f(p) \cdot f_0.$$

The path $(\Omega f)_0 : (\Omega f)(\mathsf{refl}_{x_0}) = \mathsf{refl}_{y_0}$, which exhibits Ωf as a pointed map, is the obvious path of type

$${f_0}^{-1} \cdot f(\mathsf{refl}_{x_0}) \cdot f_0 = \mathsf{refl}_{y_0}.$$

There is another functor on pointed maps, which takes $f : X \to Y$ to $\mathsf{pr}_1 : \mathsf{fib}_f(y_0) \to X$. When f is pointed, we always consider $\mathsf{fib}_f(y_0)$ to be pointed with basepoint (x_0, f_0), in which case pr_1 is also a pointed map, with witness $(\mathsf{pr}_1)_0 :\equiv \mathsf{refl}_{x_0}$. Thus, this operation can be iterated.

Definition 8.4.3. The **fiber sequence** of a pointed map $f : X \to Y$ is the infinite sequence of pointed types and pointed maps

$$\ldots \xrightarrow{f^{(n+1)}} X^{(n+1)} \xrightarrow{f^{(n)}} X^{(n)} \xrightarrow{f^{(n-1)}} \ldots \longrightarrow X^{(2)} \xrightarrow{f^{(1)}} X^{(1)} \xrightarrow{f^{(0)}} X^{(0)}$$

defined recursively by

$$X^{(0)} :\equiv Y \qquad X^{(1)} :\equiv X \qquad f^{(0)} :\equiv f$$

and

$$\begin{aligned} X^{(n+1)} &:\equiv \mathsf{fib}_{f^{(n-1)}}(x_0^{(n-1)}) \\ f^{(n)} &:\equiv \mathsf{pr}_1 \qquad\qquad\qquad : X^{(n+1)} \to X^{(n)}. \end{aligned}$$

where $x_0^{(n)}$ denotes the basepoint of $X^{(n)}$, chosen recursively as above.

Thus, any adjacent pair of maps in this fiber sequence is of the form

$$X^{(n+1)} \equiv \mathsf{fib}_{f^{(n-1)}}(x_0^{(n-1)}) \xrightarrow{f^{(n)} \equiv \mathsf{pr}_1} X^{(n)} \xrightarrow{f^{(n-1)}} X^{(n-1)}.$$

In particular, we have $f^{(n-1)} \circ f^{(n)} = 0$. We now observe that the types occurring in this sequence are the iterated loop spaces of the base space Y, the total space X, and the fiber $F :\equiv \mathsf{fib}_f(y_0)$, and similarly for the maps.

Lemma 8.4.4. *Let $f : X \to Y$ be a pointed map of pointed spaces. Then:*

(i) The fiber of $f^{(1)} :\equiv \mathsf{pr}_1 : \mathsf{fib}_f(y_0) \to X$ is equivalent to ΩY.

(ii) Similarly, the fiber of $f^{(2)} : \Omega Y \to \mathsf{fib}_f(y_0)$ is equivalent to ΩX.

(iii) Under these equivalences, the pointed map $f^{(3)} : \Omega X \to \Omega Y$ is identified with the pointed map $\Omega f \circ (-)^{-1}$.

Proof. For (i), we have

$$\begin{aligned}
\mathsf{fib}_{f^{(1)}}(x_0) &:\equiv \sum_{z:\mathsf{fib}_f(y_0)} (\mathsf{pr}_1(z) = x_0) \\
&\simeq \sum_{(x:A)} \sum_{(p:f(x)=y_0)} (x = x_0) && \text{(by Exercise 2.10)} \\
&\simeq (f(x_0) = y_0) && \text{(as } \textstyle\sum_{(x:A)}(x = x_0) \text{ is contractible)} \\
&\simeq (y_0 = y_0) && \text{(by } (f_0 \cdot -)\text{)} \\
&\equiv \Omega Y.
\end{aligned}$$

Tracing through, we see that this equivalence sends $((x, p), q)$ to ${f_0}^{-1} \cdot f\left(q^{-1}\right) \cdot p$, while its inverse sends $r : y_0 = y_0$ to $((x_0, f_0 \cdot r), \mathsf{refl}_{x_0})$. In particular, the basepoint $((x_0, f_0), \mathsf{refl}_{x_0})$ of $\mathsf{fib}_{f^{(1)}}(x_0)$ is sent to ${f_0}^{-1} \cdot f\left({\mathsf{refl}_{x_0}}^{-1}\right) \cdot f_0$, which equals refl_{y_0}. Hence this equivalence is a pointed map (see Exercise 8.7). Moreover, under this equivalence, $f^{(2)}$ is identified with $\lambda r.\,(x_0, f_0 \cdot r) : \Omega Y \to \mathsf{fib}_f(y_0)$.

Item (ii) follows immediately by applying (i) to $f^{(1)}$ in place of f. Since $(f^{(1)})_0 :\equiv \mathsf{refl}_{x_0}$, under this equivalence $f^{(3)}$ is identified with the map $\Omega X \to \mathsf{fib}_{f^{(1)}}(x_0)$ defined by $s \mapsto ((x_0, f_0), s)$. Thus, when we compose with the previous equivalence $\mathsf{fib}_{f^{(1)}}(x_0) \simeq \Omega Y$, we see that s maps to ${f_0}^{-1} \cdot f\left(s^{-1}\right) \cdot f_0$, which is by definition $(\Omega f)(s^{-1})$. We omit the proof that this is an equality of pointed maps rather than just of functions. □

Thus, the fiber sequence of $f : X \to Y$ can be pictured as:

$$\ldots \longrightarrow \Omega^2 X \xrightarrow{\Omega^2 f} \Omega^2 Y \xrightarrow{-\Omega\partial} \Omega F \xrightarrow{-\Omega i} \Omega X \xrightarrow{-\Omega f} \Omega Y \xrightarrow{\partial} F \xrightarrow{i} X \xrightarrow{f} Y.$$

where the minus signs denote composition with path inversion $(-)^{-1}$. Note that by Exercise 8.6, we have

$$\Omega\left(\Omega f \circ (-)^{-1}\right) \circ (-)^{-1} = \Omega^2 f \circ (-)^{-1} \circ (-)^{-1} = \Omega^2 f.$$

Thus, there are minus signs on the k-fold loop maps whenever k is odd.

From this fiber sequence we will deduce an *exact sequence of pointed sets*. Let A and B be sets and $f : A \to B$ a function, and recall from Definition 7.6.3 the definition of the *image* $\mathsf{im}(f)$, which can be regarded as a subset of B:

$$\mathsf{im}(f) :\equiv \{\, b : B \mid \exists (a : A).\, f(a) = b \,\}.$$

If A and B are moreover pointed with basepoints a_0 and b_0, and f is a pointed map, we define the **kernel** of f to be the following subset of A:

$$\ker(f) :\equiv \{\, x : A \mid f(x) = b_0 \,\}.$$

Of course, this is just the fiber of f over the basepoint b_0; it is a subset of A because B is a set.

Note that any group is a pointed set, with its unit element as basepoint, and any group homomorphism is a pointed map. In this case, the kernel and image agree with the usual notions from group theory.

Definition 8.4.5. An **exact sequence of pointed sets** is a (possibly bounded) sequence of pointed sets and pointed maps:

$$\ldots \longrightarrow A^{(n+1)} \xrightarrow{f^{(n)}} A^{(n)} \xrightarrow{f^{(n-1)}} A^{(n-1)} \longrightarrow \ldots$$

such that for every n, the image of $f^{(n)}$ is equal, as a subset of $A^{(n)}$, to the kernel of $f^{(n-1)}$. In other words, for all $a : A^{(n)}$ we have

$$(f^{(n-1)}(a) = a_0^{(n-1)}) \iff \exists (b : A^{(n+1)}).\, (f^{(n)}(b) = a).$$

where $a_0^{(n)}$ denotes the basepoint of $A^{(n)}$.

Usually, most or all of the pointed sets in an exact sequence are groups, and often abelian groups. When we speak of an **exact sequence of groups**, it is assumed moreover that the maps are group homomorphisms and not just pointed maps.

Theorem 8.4.6. *Let $f : X \to Y$ be a pointed map between pointed spaces with fiber $F :\equiv \mathsf{fib}_f(y_0)$. Then we have the following long exact sequence, which consists of groups except for the last three terms, and abelian groups except for the last six.*

$$\begin{array}{ccccc}
\vdots & & \vdots & & \vdots \\
\pi_k(F) & \longrightarrow & \pi_k(X) & \longrightarrow & \pi_k(Y) \\
\vdots & & \vdots & & \vdots \\
\pi_2(F) & \longrightarrow & \pi_2(X) & \longrightarrow & \pi_2(Y) \\
\pi_1(F) & \longrightarrow & \pi_1(X) & \longrightarrow & \pi_1(Y) \\
\pi_0(F) & \longrightarrow & \pi_0(X) & \longrightarrow & \pi_0(Y)
\end{array}$$

Proof. We begin by showing that the 0-truncation of a fiber sequence is an exact sequence of pointed sets. Thus, we need to show that for any adjacent pair of maps in a fiber sequence:

$$\mathsf{fib}_f(z_0) \xrightarrow{g} W \xrightarrow{f} Z$$

with $g :\equiv \mathsf{pr}_1$, the sequence

$$\left\|\mathsf{fib}_f(z_0)\right\|_0 \xrightarrow{\|g\|_0} \|W\|_0 \xrightarrow{\|f\|_0} \|Z\|_0$$

is exact, i.e. that $\mathsf{im}(\|g\|_0) \subseteq \ker(\|f\|_0)$ and $\ker(\|f\|_0) \subseteq \mathsf{im}(\|g\|_0)$.

The first inclusion is equivalent to $\|g\|_0 \circ \|f\|_0 = 0$, which holds by functoriality of $\|-\|_0$ and the fact that $g \circ f = 0$. For the second, we assume $w' : \|W\|_0$ and $p' : \|f\|_0(w') = |z_0|_0$ and show there merely exists $t : \mathsf{fib}_f(z_0)$ such that $g(t) = w'$. Since our goal is a mere proposition, we can assume that w' is of the form $|w|_0$ for some $w : W$. Now by Theorem 7.3.12, $p' : |f(w)|_0 = |z_0|_0$ yields $p'' : \|f(w) = z_0\|_{-1}$, so by a further truncation induction we may assume some $p : f(w) = z_0$. But now we have $|(w, p)|_0 : \left|\mathsf{fib}_f(z_0)\right|_0$ whose image under $\|g\|_0$ is $|w|_0 \equiv w'$, as desired.

Thus, applying $\|-\|_0$ to the fiber sequence of f, we obtain a long exact sequence involving the pointed sets $\pi_k(F)$, $\pi_k(X)$, and $\pi_k(Y)$ in the desired order. And of course, π_k is a group for $k \geq 1$, being the 0-truncation of a loop space, and an abelian group for $k \geq 2$ by the Eckmann–Hilton argument (Theorem 2.1.6). Moreover, Lemma 8.4.4 allows us to identify the maps $\pi_k(F) \to \pi_k(X)$ and $\pi_k(X) \to \pi_k(Y)$ in this exact sequence as $(-1)^k\pi_k(i)$ and $(-1)^k\pi_k(f)$ respectively.

More generally, every map in this long exact sequence except the last three is of the form $\|\Omega h\|_0$ or $\|-\Omega h\|_0$ for some h. In the former case it is a group homomorphism, while in the latter case it is a homomorphism if the groups are abelian; otherwise it is an "anti-homomorphism". However, the kernel and image of a group homomorphism are unchanged when we replace it by its negative, and hence so is the exactness of any sequence involving it. Thus, we can modify our long exact sequence to obtain one involving $\pi_k(i)$ and $\pi_k(f)$ directly and in which all the maps are group homomorphisms (except the last three). □

The usual properties of exact sequences of abelian groups can be proved as usual. In particular we have:

Lemma 8.4.7. *Suppose given an exact sequence of abelian groups:*

$$K \longrightarrow G \xrightarrow{f} H \longrightarrow Q.$$

(i) If $K = 0$, then f is injective.
(ii) If $Q = 0$, then f is surjective.
(iii) If $K = Q = 0$, then f is an isomorphism.

Proof. Since the kernel of f is the image of $K \to G$, if $K = 0$ then the kernel of f is $\{0\}$; hence f is injective because it's a group morphism. Similarly, since the image of f is the kernel of $H \to Q$, if $Q = 0$ then the image of f is all of H, so f is surjective. Finally, (iii) follows from (i) and (ii) by Theorem 4.6.3. □

As an immediate application, we can now quantify in what way n-connectedness of a map is stronger than inducing an equivalence on n-truncations.

Corollary 8.4.8. *Let $f : A \to B$ be n-connected and $a : A$, and define $b :\equiv f(a)$. Then:*

(i) If $k \leq n$, then $\pi_k(f) : \pi_k(A,a) \to \pi_k(B,b)$ is an isomorphism.
(ii) If $k = n+1$, then $\pi_k(f) : \pi_k(A,a) \to \pi_k(B,b)$ is surjective.

Proof. For $k = 0$, part (i) follows from Lemma 7.5.14, noticing that $\pi_0(f) \equiv \|f\|_0$. For $k = 0$ part (ii) follows from Exercise 7.18, noticing that a function is surjective iff it's (-1)-connected, by Lemma 7.5.2. For $k > 0$ we have as part of the long exact sequence an exact sequence

$$\pi_k(\mathsf{fib}_f(b)) \longrightarrow \pi_k(A,a) \xrightarrow{f} \pi_k(B,b) \longrightarrow \pi_{k-1}(\mathsf{fib}_f(b)).$$

Now since f is n-connected, $\left\|\mathsf{fib}_f(b)\right\|_n$ is contractible. Therefore, if $k \leq n$, then $\pi_k(\mathsf{fib}_f(b)) = \left\|\Omega^k(\mathsf{fib}_f(b))\right\|_0 = \Omega^k(\left\|\mathsf{fib}_f(b)\right\|_k)$ is also contractible. Thus, $\pi_k(f)$ is an isomorphism for $k \leq n$ by Lemma 8.4.7(iii), while for $k = n+1$ it is surjective by Lemma 8.4.7(ii). □

In §8.8 we will see that the converse of Corollary 8.4.8 also holds.

8.5 The Hopf fibration

In this section we will define the **Hopf fibration**.

Theorem 8.5.1 (Hopf Fibration)**.** *There is a fibration H over $\mathbb{S}^2$ whose fiber over the basepoint is $\mathbb{S}^1$ and whose total space is $\mathbb{S}^3$.*

The Hopf fibration will allow us to compute several homotopy groups of spheres. Indeed, it yields the following long exact sequence of homotopy groups (see §8.4):

$$\begin{array}{ccccc}
\pi_k(\mathbb{S}^1) & \longrightarrow & \pi_k(\mathbb{S}^3) & \longrightarrow & \pi_k(\mathbb{S}^2) \\
\vdots & \swarrow & \vdots & \swarrow & \vdots \\
\pi_2(\mathbb{S}^1) & \longrightarrow & \pi_2(\mathbb{S}^3) & \longrightarrow & \pi_2(\mathbb{S}^2) \\
\pi_1(\mathbb{S}^1) & \longrightarrow & \pi_1(\mathbb{S}^3) & \longrightarrow & \pi_1(\mathbb{S}^2)
\end{array}$$

We've already computed all $\pi_n(\mathbb{S}^1)$, and $\pi_k(\mathbb{S}^n)$ for $k < n$, so this becomes the following:

$$\begin{array}{ccccc}
0 & \longrightarrow & \pi_k(\mathbb{S}^3) & \longrightarrow & \pi_k(\mathbb{S}^2) \\
\vdots & \swarrow & \vdots & \swarrow & \vdots \\
0 & \longrightarrow & \pi_3(\mathbb{S}^3) & \longrightarrow & \pi_3(\mathbb{S}^2) \\
0 & \longrightarrow & 0 & \longrightarrow & \pi_2(\mathbb{S}^2) \\
\mathbb{Z} & \longrightarrow & 0 & \longrightarrow & 0
\end{array}$$

In particular we get the following result:

Corollary 8.5.2. *We have $\pi_2(\mathbb{S}^2) \simeq \mathbb{Z}$ and $\pi_k(\mathbb{S}^3) \simeq \pi_k(\mathbb{S}^2)$ for every $k \geq 3$ (where the map is induced by the Hopf fibration, seen as a map from the total space $\mathbb{S}^3$ to the base space $\mathbb{S}^2$).*

In fact, we can say more: the fiber sequence of the Hopf fibration will show that $\Omega^3(\mathbb{S}^3)$ is the fiber of a map from $\Omega^3(\mathbb{S}^2)$ to $\Omega^2(\mathbb{S}^1)$. Since $\Omega^2(\mathbb{S}^1)$ is contractible, we have $\Omega^3(\mathbb{S}^3) \simeq \Omega^3(\mathbb{S}^2)$. In classical homotopy theory, this fact would be a consequence of Corollary 8.5.2 and Whitehead's theorem, but Whitehead's theorem is not necessarily valid in homotopy type theory (see §8.8). We will not use the more precise version here though.

8.5.1 Fibrations over pushouts

We first start with a lemma explaining how to construct fibrations over pushouts.

Lemma 8.5.3. *Let $\mathscr{D} = (Y \xleftarrow{j} X \xrightarrow{k} Z)$ be a span and assume that we have*

- *Two fibrations $E_Y : Y \to \mathcal{U}$ and $E_Z : Z \to \mathcal{U}$.*
- *An equivalence e_X between $E_Y \circ j : X \to \mathcal{U}$ and $E_Z \circ k : X \to \mathcal{U}$, i.e.*

$$e_X : \prod_{x:X} E_Y(j(x)) \simeq E_Z(k(x)).$$

Then we can construct a fibration $E : Y \sqcup^X Z \to \mathcal{U}$ such that

- *For all $y : Y$, $E(\mathsf{inl}(y)) \equiv E_Y(y)$.*
- *For all $z : Z$, $E(\mathsf{inr}(z)) \equiv E_Z(z)$.*
- *For all $x : X$, $E(\mathsf{glue}(x)) = \mathsf{ua}(e_X(x))$ (note that both sides of the equation are paths in $\mathcal{U}$ from $E_Y(j(x))$ to $E_Z(k(x))$).*

Moreover, the total space of this fibration fits in the following pushout square:

$$\begin{array}{ccc} \sum_{(x:X)} E_Y(j(x)) & \xrightarrow{\mathsf{id}\times e_X}_{\sim} & \sum_{(x:X)} E_Z(k(x)) \xrightarrow{k\times\mathsf{id}} \sum_{(z:Z)} E_Z(z) \\ \downarrow{\scriptstyle j\times\mathsf{id}} & & \downarrow{\scriptstyle \mathsf{inr}} \\ \sum_{(y:Y)} E_Y(y) & \xrightarrow[\mathsf{inl}]{} & \sum_{(t:Y\sqcup^X Z)} E(t) \end{array}$$

Proof. We define E by the recursion principle of the pushout $Y \sqcup^X Z$. For that, we need to specify the value of E on elements of the form $\mathsf{inl}(y)$, $\mathsf{inr}(z)$ and the action of E on paths $\mathsf{glue}(x)$, so we can just choose the following values:

$$\begin{aligned} E(\mathsf{inl}(y)) &:\equiv E_Y(y), \\ E(\mathsf{inr}(z)) &:\equiv E_Z(z), \\ E(\mathsf{glue}(x)) &:= \mathsf{ua}(e_X(x)). \end{aligned}$$

To see that the total space of this fibration is a pushout, we apply the flattening lemma (Lemma 6.12.2) with the following values:

- $A :\equiv Y + Z$, $B :\equiv X$ and $f, g : B \to A$ are defined by $f(x) :\equiv \mathsf{inl}(j(x))$, $g(x) :\equiv \mathsf{inr}(k(x))$,
- the type family $C : A \to \mathcal{U}$ is defined by

$$C(\mathsf{inl}(y)) :\equiv E_Y(y) \qquad \text{and} \qquad C(\mathsf{inr}(z)) :\equiv E_Z(z),$$

- the family of equivalences $D : \prod_{(b:B)} C(f(b)) \simeq C(g(b))$ is defined to be e_X.

The base higher inductive type W in the flattening lemma is equivalent to the pushout $Y \sqcup^X Z$ and the type family $P : Y \sqcup^X Z \to \mathcal{U}$ is equivalent to the E defined above.

Thus the flattening lemma tells us that $\sum_{(t:Y\sqcup^X Z)} E(t)$ is equivalent to the higher inductive type $E^{\mathrm{tot}'}$ with the following generators:

- a function $\mathsf{z} : \sum_{(a:Y+Z)} C(a) \to E^{\mathrm{tot}'}$,
- for each $x : X$ and $t : E_Y(j(x))$, a path $\mathsf{z}(\mathsf{inl}(j(x)), t) = \mathsf{z}(\mathsf{inr}(k(x)), e_X(t))$.

Using the flattening lemma again or a direct computation, it is easy to see that $\sum_{(a:Y+Z)} C(a) \simeq \sum_{(y:Y)} E_Y(y) + \sum_{(z:Z)} E_Z(z)$, hence $E^{\mathrm{tot}'}$ is equivalent to the higher inductive type E^{tot} with the following generators:

- a function $\mathsf{inl} : \sum_{(y:Y)} E_Y(y) \to E^{\mathrm{tot}}$,
- a function $\mathsf{inr} : \sum_{(z:Z)} E_Z(z) \to E^{\mathrm{tot}}$,
- for each $(x, t) : \sum_{(x:X)} E_Y(j(x))$ a path $\mathsf{glue}(x, t) : \mathsf{inl}(j(x), t) = \mathsf{inr}(k(x), e_X(t))$.

Thus the total space of E is the pushout of the total spaces of E_Y and E_Z, as required. □

8.5.2 The Hopf construction

Definition 8.5.4. An **H-space** consists of

- a type A,
- a base point $e : A$,
- a binary operation $\mu : A \times A \to A$, and
- for every $a : A$, equalities $\mu(e, a) = a$ and $\mu(a, e) = a$.

Lemma 8.5.5. *Let A be a connected H-space. Then for every $a : A$, the maps $\mu(a, -) : A \to A$ and $\mu(-, a) : A \to A$ are equivalences.*

Proof. Let us prove that for every $a : A$ the map $\mu(a, -)$ is an equivalence. The other statement is symmetric. The statement that $\mu(a, -)$ is an equivalence corresponds to a type family $P : A \to \mathsf{Prop}$ and proving it corresponds to finding a section of this type family.

The type Prop is a set (Theorem 7.1.11) hence we can define a new type family $P' : \|A\|_0 \to \mathsf{Prop}$ by $P'(|a|_0) :\equiv P(a)$. But A is connected by assumption, hence $\|A\|_0$ is contractible. This implies that in order to find a section of P', it is enough to find a point in the fiber of P' over $|e|_0$. But we have $P'(|e|_0) = P(e)$ which is inhabited because $\mu(e, -)$ is equal to the identity map by definition of an H-space, hence is an equivalence.

We have proved that for every $x : \|A\|_0$ the proposition $P'(x)$ is true, hence in particular for every $a : A$ the proposition $P(a)$ is true because $P(a)$ is $P'(|a|_0)$. □

Definition 8.5.6. Let A be a connected H-space. We define a fibration over ΣA using Lemma 8.5.3.

Given that ΣA is the pushout $\mathbf{1} \sqcup^A \mathbf{1}$, we can define a fibration over ΣA by specifying

- two fibrations over $\mathbf{1}$ (i.e. two types F_1 and F_2), and
- a family $e : A \to (F_1 \simeq F_2)$ of equivalences between F_1 and F_2, one for every element of A.

We take A for F_1 and F_2, and for $a : A$ we take the equivalence $\mu(a, -)$ for $e(a)$.

According to Lemma 8.5.3, we have the following diagram:

$$\begin{array}{ccccc} A & \xleftarrow{\mathsf{pr}_2} & A \times A & \xrightarrow{\mu} & A \\ \downarrow & & \downarrow{\scriptstyle \mathsf{pr}_1} & & \downarrow \\ 1 & \longleftarrow & A & \longrightarrow & 1 \end{array}$$

and the fibration we just constructed is a fibration over ΣA whose total space is the pushout of the top line.

Moreover, with $f(x, y) :\equiv (\mu(x, y), y)$ we have the following diagram:

$$\begin{array}{ccccc} A & \xleftarrow{\mathsf{pr}_2} & A \times A & \xrightarrow{\mu} & A \\ {\scriptstyle \mathsf{id}}\downarrow & & \downarrow{\scriptstyle f} & & \downarrow{\scriptstyle \mathsf{id}} \\ A & \xleftarrow[\mathsf{pr}_2]{} & A \times A & \xrightarrow[\mathsf{pr}_1]{} & A \end{array}$$

The diagram commutes and the three vertical maps are equivalences, the inverse of f being the function g defined by

$$g(u, v) :\equiv (\mu(-, v)^{-1}(u), v).$$

This shows that the two lines are equivalent (hence equal) spans, so the total space of the fibration we constructed is equivalent to the pushout of the bottom line. And by definition, this latter pushout is the *join* of A with itself (see §6.8). We have proven:

Lemma 8.5.7. *Given a connected H-space A, there is a fibration, called the **Hopf construction**, over ΣA with fiber A and total space $A * A$.*

8.5.3 The Hopf fibration

We will first construct a structure of H-space on the circle $\mathbb{S}^1$, hence by Lemma 8.5.7 we will get a fibration over $\mathbb{S}^2$ with fiber $\mathbb{S}^1$ and total space $\mathbb{S}^1 * \mathbb{S}^1$. We will then prove that this join is equivalent to $\mathbb{S}^3$.

Lemma 8.5.8. *There is an H-space structure on the circle $\mathbb{S}^1$.*

Proof. For the base point of the H-space structure we choose base. Now we need to define the multiplication operation $\mu : \mathbb{S}^1 \times \mathbb{S}^1 \to \mathbb{S}^1$. We will define the curried form $\widetilde{\mu} : \mathbb{S}^1 \to (\mathbb{S}^1 \to \mathbb{S}^1)$ of μ by recursion on $\mathbb{S}^1$:

$$\widetilde{\mu}(\mathsf{base}) :\equiv \mathsf{id}_{\mathbb{S}^1}, \qquad \text{and} \qquad \widetilde{\mu}(\mathsf{loop}) := \mathsf{funext}(h).$$

where $h : \prod_{(x:\mathbb{S}^1)}(x = x)$ is the function defined in Lemma 6.4.2, which has the property that $h(\mathsf{base}) :\equiv \mathsf{loop}$.

Now we just have to prove that $\mu(x, \mathsf{base}) = \mu(\mathsf{base}, x) = x$ for every $x : \mathbb{S}^1$. By definition, if $x : \mathbb{S}^1$ we have $\mu(\mathsf{base}, x) = \widetilde{\mu}(\mathsf{base})(x) = \mathsf{id}_{\mathbb{S}^1}(x) = x$. For the equality $\mu(x, \mathsf{base}) = x$ we do it by induction on $x : \mathbb{S}^1$:

- If x is base then $\mu(\mathsf{base}, \mathsf{base}) = \mathsf{base}$ by definition, so we have $\mathsf{refl}_{\mathsf{base}} : \mu(\mathsf{base}, \mathsf{base}) = \mathsf{base}$.

- When x varies along loop, we need to prove that

$$\mathsf{refl}_{\mathsf{base}} \cdot \mathsf{ap}_{\lambda x.\, x}(\mathsf{loop}) = \mathsf{ap}_{\lambda x.\, \mu(x,\mathsf{base})}(\mathsf{loop}) \cdot \mathsf{refl}_{\mathsf{base}}.$$

  The left-hand side is equal to loop, and for the right-hand side we have:

$$\begin{aligned}
\mathsf{ap}_{\lambda x.\, \mu(x,\mathsf{base})}(\mathsf{loop}) \cdot \mathsf{refl}_{\mathsf{base}} &= \mathsf{ap}_{\lambda x.\, (\widetilde{\mu}(x))(\mathsf{base})}(\mathsf{loop}) \\
&= \mathsf{happly}(\mathsf{ap}_{\lambda x.\, (\widetilde{\mu}(x))}(\mathsf{loop}), \mathsf{base}) \\
&= \mathsf{happly}(\mathsf{funext}(h), \mathsf{base}) \\
&= h(\mathsf{base}) \\
&= \mathsf{loop}.
\end{aligned}$$

□

Now recall from §6.8 that the *join* $A * B$ of types A and B is the pushout of the diagram

$$A \xleftarrow{\mathsf{pr}_1} A \times B \xrightarrow{\mathsf{pr}_2} B.$$

Lemma 8.5.9. *The operation of join is associative: if A, B and C are three types then we have an equivalence $(A * B) * C \simeq A * (B * C)$.*

Proof. We define a map $f : (A * B) * C \to A * (B * C)$ by induction. We first need to define $f \circ \mathsf{inl} : A * B \to A * (B * C)$ which will be done by induction, then $f \circ \mathsf{inr} : C \to A * (B * C)$, and then $\mathsf{ap}_f \circ \mathsf{glue} : \prod_{(t:(A*B)\times C)} f(\mathsf{inl}(\mathsf{pr}_1(t))) = f(\mathsf{inr}(\mathsf{pr}_2(t)))$ which will be done by induction on the first component of t:

$$\begin{aligned}
(f \circ \mathsf{inl})(\mathsf{inl}(a)) &:\equiv \mathsf{inl}(a), \\
(f \circ \mathsf{inl})(\mathsf{inr}(b)) &:\equiv \mathsf{inr}(\mathsf{inl}(b)), \\
\mathsf{ap}_{f\circ\mathsf{inl}}(\mathsf{glue}(a,b)) &:= \mathsf{glue}(a, \mathsf{inl}(b)), \\
f(\mathsf{inr}(c)) &:\equiv \mathsf{inr}(\mathsf{inr}(c)), \\
\mathsf{ap}_f(\mathsf{glue}(\mathsf{inl}(a), c)) &:= \mathsf{glue}(a, \mathsf{inr}(c)), \\
\mathsf{ap}_f(\mathsf{glue}(\mathsf{inr}(b), c)) &:= \mathsf{ap}_{\mathsf{inr}}(\mathsf{glue}(b,c)), \\
\mathsf{apd}_{\lambda x.\, \mathsf{ap}_f(\mathsf{glue}(x,c))}(\mathsf{glue}(a,b)) &:= \text{“}\mathsf{apd}_{\lambda x.\, \mathsf{glue}(a,x)}(\mathsf{glue}(b,c))\text{”}.
\end{aligned}$$

For the last equation, note that the right-hand side is of type

$$\mathsf{transport}^{\lambda x.\, \mathsf{inl}(a)=\mathsf{inr}(x)}(\mathsf{glue}(b,c), \mathsf{glue}(a, \mathsf{inl}(b))) = \mathsf{glue}(a, \mathsf{inr}(c))$$

whereas it is supposed to be of type

$$\mathsf{transport}^{\lambda x.\, f(\mathsf{inl}(x))=f(\mathsf{inr}(c))}(\mathsf{glue}(a,b), \mathsf{ap}_f(\mathsf{glue}(\mathsf{inl}(a), c))) = \mathsf{ap}_f(\mathsf{glue}(\mathsf{inr}(b), c)).$$

But by the previous clauses in the definition, both of these types are equivalent to the following type:

$$\mathsf{glue}(a, \mathsf{inr}(c)) = \mathsf{glue}(a, \mathsf{inl}(b)) \cdot \mathsf{ap}_{\mathsf{inr}}(\mathsf{glue}(b,c)),$$

and so we can coerce by an equivalence to obtain the necessary element. Similarly, we can define a map $g : A * (B * C) \to (A * B) * C$, and checking that f and g are inverse to each other is a long and tedious but essentially straightforward computation. □

A more conceptual proof sketch is as follows.

Proof. Let us consider the following diagram where the maps are the obvious projections:

$$\begin{array}{ccccc} A & \longleftarrow & A \times C & \longrightarrow & A \times C \\ \uparrow & & \uparrow & & \uparrow \\ A \times B & \longleftarrow & A \times B \times C & \longrightarrow & A \times C \\ \downarrow & & \downarrow & & \downarrow \\ B & \longleftarrow & B \times C & \longrightarrow & C \end{array}$$

Taking the colimit of the columns gives the following diagram, whose colimit is $(A * B) * C$:

$$A * B \longleftarrow (A * B) \times C \longrightarrow C$$

On the other hand, taking the colimit of the lines gives a diagram whose colimit is $A * (B * C)$.

Hence using a Fubini-like theorem for colimits (that we haven't proved) we have an equivalence $(A * B) * C \simeq A * (B * C)$. The proof of this Fubini theorem for colimits still requires the long and tedious computation, though. □

Lemma 8.5.10. *For any type A, there is an equivalence $\Sigma A \simeq \mathbf{2} * A$.*

Proof. It is easy to define the two maps back and forth and to prove that they are inverse to each other. The details are left as an exercise to the reader. □

We can now construct the Hopf fibration:

Theorem 8.5.11. *There is a fibration over $\mathbb{S}^2$ of fiber $\mathbb{S}^1$ and total space $\mathbb{S}^3$.*

Proof. We proved that $\mathbb{S}^1$ has a structure of H-space (cf Lemma 8.5.8) hence by Lemma 8.5.7 there is a fibration over $\mathbb{S}^2$ of fiber $\mathbb{S}^1$ and total space $\mathbb{S}^1 * \mathbb{S}^1$. But by the two previous results and Lemma 6.5.1 we have:

$$\mathbb{S}^1 * \mathbb{S}^1 = (\Sigma\mathbf{2}) * \mathbb{S}^1 = (\mathbf{2} * \mathbf{2}) * \mathbb{S}^1 = \mathbf{2} * (\mathbf{2} * \mathbb{S}^1) = \Sigma(\Sigma\mathbb{S}^1) = \mathbb{S}^3.$$ □

8.6 The Freudenthal suspension theorem

Before proving the Freudenthal suspension theorem, we need some auxiliary lemmas about connectedness. In Chapter 7 we proved a number of facts about n-connected maps and n-types for fixed n; here we are now interested in what happens when we vary n. For instance, in Lemma 7.5.7 we showed that n-connected maps are characterized by an "induction principle" relative to families of n-types. If we want to "induct along" an n-connected map into a family of k-types for $k > n$, we don't immediately know that there is a function by such an induction principle, but the following lemma says that at least our ignorance can be quantified.

Lemma 8.6.1. *If $f : A \to B$ is n-connected and $P : B \to k$-Type is a family of k-types for $k \geq n$, then the induced function*

$$(- \circ f) : \Big(\prod_{b:B} P(b)\Big) \to \Big(\prod_{a:A} P(f(a))\Big)$$

is $(k - n - 2)$-truncated.

Proof. We induct on the natural number $k-n$. When $k=n$, this is Lemma 7.5.7. For the inductive step, suppose f is n-connected and P is a family of $(k+1)$-types. To show that $(-\circ f)$ is $(k-n-1)$-truncated, let $\ell : \prod_{(a:A)} P(f(a))$; then we have

$$\mathsf{fib}_{(-\circ f)}(\ell) \simeq \sum_{(g:\prod_{(b:B)} P(b))} \prod_{(a:A)} g(f(a)) = \ell(a).$$

Let (g,p) and (h,q) lie in this type, so $p : g\circ f \sim \ell$ and $q : h\circ f \sim \ell$; then we also have

$$\big((g,p)=(h,q)\big) \simeq \Big(\sum_{r:g\sim h} r\circ f = p \cdot q^{-1}\Big).$$

However, here the right-hand side is a fiber of the map

$$(-\circ f) : \Big(\prod_{b:B} Q(b)\Big) \to \Big(\prod_{a:A} Q(f(a))\Big)$$

where $Q(b) :\equiv (g(b)=h(b))$. Since P is a family of $(k+1)$-types, Q is a family of k-types, so the inductive hypothesis implies that this fiber is a $(k-n-2)$-type. Thus, all path spaces of $\mathsf{fib}_{(-\circ f)}(\ell)$ are $(k-n-2)$-types, so it is a $(k-n-1)$-type. □

Recall that if (A,a_0) and (B,b_0) are pointed types, then their **wedge** $A\vee B$ is defined to be the pushout of $A \xleftarrow{a_0} \mathbf{1} \xrightarrow{b_0} B$. There is a canonical map $i : A\vee B \to A\times B$ defined by the two maps $\lambda a.\,(a,b_0)$ and $\lambda b.\,(a_0,b)$; the following lemma essentially says that this map is highly connected if A and B are so. It is a bit more convenient both to prove and use, however, if we use the characterization of connectedness from Lemma 7.5.7 and substitute in the universal property of the wedge (generalized to type families).

Lemma 8.6.2 (Wedge connectivity lemma)**.** *Suppose that (A,a_0) and (B,b_0) are n- and m-connected pointed types, respectively, with $n,m \geq 0$, and let $P : A \to B \to (n+m)$-*Type*. Then for any $f : \prod_{(a:A)} P(a,b_0)$ and $g : \prod_{(b:B)} P(a_0,b)$ with $p : f(a_0) = g(b_0)$, there exists $h : \prod_{(a:A)}\prod_{(b:B)} P(a,b)$ with homotopies*

$$q : \prod_{a:A} h(a,b_0) = f(a) \qquad \text{and} \qquad r : \prod_{b:B} h(a_0,b) = g(b)$$

such that $p = q(a_0)^{-1} \cdot r(b_0)$.

Proof. Define $P : A \to \mathcal{U}$ by

$$P(a) :\equiv \sum_{k:\prod_{(b:B)} P(a,b)} (f(a) = k(b_0)).$$

Then we have $(g,p) : P(a_0)$. Since $a_0 : \mathbf{1} \to A$ is $(n-1)$-connected, if P is a family of $(n-1)$-types then we will have $\ell : \prod_{(a:A)} P(a)$ such that $\ell(a_0) = (g,p)$, in which case we can define $h(a,b) :\equiv \mathsf{pr}_1(\ell(a))(b)$. However, for fixed a, the type $P(a)$ is the fiber over $f(a)$ of the map

$$\Big(\prod_{b:B} P(a,b)\Big) \to P(a,b_0)$$

given by precomposition with $b_0 : \mathbf{1} \to B$. Since $b_0 : \mathbf{1} \to B$ is $(m-1)$-connected, for this fiber to be $(n-1)$-truncated, by Lemma 8.6.1 it suffices for each type $P(a,b)$ to be an $(n+m)$-type, which we have assumed. □

Let (X, x_0) be a pointed type, and recall the definition of the suspension ΣX from §6.5, with constructors $\mathsf{N}, \mathsf{S} : \Sigma X$ and $\mathsf{merid} : X \to (\mathsf{N} = \mathsf{S})$. We regard ΣX as a pointed space with basepoint N, so that we have $\Omega\Sigma X :\equiv (\mathsf{N} =_{\Sigma X} \mathsf{N})$. Then there is a canonical map

$$\sigma : X \to \Omega\Sigma X$$
$$\sigma(x) :\equiv \mathsf{merid}(x) \cdot \mathsf{merid}(x_0)^{-1}.$$

Remark 8.6.3. In classical algebraic topology, one considers the *reduced suspension*, in which the path $\mathsf{merid}(x_0)$ is collapsed down to a point, identifying N and S. The reduced and unreduced suspensions are homotopy equivalent, so the distinction is invisible to our purely homotopy-theoretic eyes — and higher inductive types only allow us to "identify" points up to a higher path anyway, there is no purpose to considering reduced suspensions in homotopy type theory. However, the "unreducedness" of our suspension is the reason for the (possibly unexpected) appearance of $\mathsf{merid}(x_0)^{-1}$ in the definition of σ.

Our goal is now to prove the following.

Theorem 8.6.4 (The Freudenthal suspension theorem)**.** *Suppose that X is n-connected and pointed, with $n \geq 0$. Then the map $\sigma : X \to \Omega\Sigma(X)$ is $2n$-connected.*

We will use the encode-decode method, but applied in a slightly different way. In most cases so far, we have used it to characterize the loop space $\Omega(A, a_0)$ of some type as equivalent to some other type B, by constructing a family $\mathsf{code} : A \to \mathcal{U}$ with $\mathsf{code}(a_0) :\equiv B$ and a family of equivalences $\mathsf{decode} : \prod_{(x:A)} \mathsf{code}(x) \simeq (a_0 = x)$.

In this case, however, we want to show that $\sigma : X \to \Omega\Sigma X$ is $2n$-connected. We could use a truncated version of the previous method, such as we will see in §8.7, to prove that $\|X\|_{2n} \to \|\Omega\Sigma X\|_{2n}$ is an equivalence—but this is a slightly weaker statement than the map being $2n$-connected (see Corollaries 8.4.8 and 8.8.5). However, note that in the general case, to prove that $\mathsf{decode}(x)$ is an equivalence, we could equivalently be proving that its fibers are contractible, and we would still be able to use induction over the base type. This we can generalize to prove connectedness of a map into a loop space, i.e. that the *truncations* of its fibers are contractible. Moreover, instead of constructing code and decode separately, we can construct directly a family of *codes for the truncations of the fibers*.

Definition 8.6.5. If X is n-connected and pointed with $n \geq 0$, then there is a family

$$\mathsf{code} : \prod_{y:\Sigma X} (\mathsf{N} = y) \to \mathcal{U} \tag{8.6.6}$$

such that

$$\mathsf{code}(\mathsf{N}, p) :\equiv \|\mathsf{fib}_\sigma(p)\|_{2n} \equiv \left\| \textstyle\sum_{(x:X)} (\mathsf{merid}(x) \cdot \mathsf{merid}(x_0)^{-1} = p) \right\|_{2n} \tag{8.6.7}$$
$$\mathsf{code}(\mathsf{S}, q) :\equiv \|\mathsf{fib}_{\mathsf{merid}}(q)\|_{2n} \equiv \left\| \textstyle\sum_{(x:X)} (\mathsf{merid}(x) = q) \right\|_{2n}. \tag{8.6.8}$$

Our eventual goal will be to prove that $\mathsf{code}(y, p)$ is contractible for all $y : \Sigma X$ and $p : \mathsf{N} = y$. Applying this with $y :\equiv \mathsf{N}$ will show that all fibers of σ are $2n$-connected, and thus σ is $2n$-connected.

Proof of Definition 8.6.5. We define $\mathsf{code}(y, p)$ by induction on $y : \Sigma X$, where the first two cases are (8.6.7) and (8.6.8). It remains to construct, for each $x_1 : X$, a dependent path

$$\mathsf{code}(\mathsf{N}) =^{\lambda y.\, (\mathsf{N}=y)\to\mathcal{U}}_{\mathsf{merid}(x_1)} \mathsf{code}(\mathsf{S}).$$

By Lemma 2.9.6, this is equivalent to giving a family of paths

$$\prod_{q:\mathsf{N}=\mathsf{S}} \mathsf{code}(\mathsf{N})(\mathsf{transport}^{\lambda y.\,(\mathsf{N}=y)}(\mathsf{merid}(x_1)^{-1}, q)) = \mathsf{code}(\mathsf{S})(q).$$

And by univalence and transport in path types, this is equivalent to a family of equivalences

$$\prod_{q:\mathsf{N}=\mathsf{S}} \mathsf{code}(\mathsf{N}, q \cdot \mathsf{merid}(x_1)^{-1}) \simeq \mathsf{code}(\mathsf{S}, q).$$

We will define a family of maps

$$\prod_{q:\mathsf{N}=\mathsf{S}} \mathsf{code}(\mathsf{N}, q \cdot \mathsf{merid}(x_1)^{-1}) \to \mathsf{code}(\mathsf{S}, q). \tag{8.6.9}$$

and then show that they are all equivalences. Thus, let $q : \mathsf{N} = \mathsf{S}$; by the universal property of truncation and the definitions of $\mathsf{code}(\mathsf{N}, -)$ and $\mathsf{code}(\mathsf{S}, -)$, it will suffice to define for each $x_2 : X$, a map

$$\left(\mathsf{merid}(x_2) \cdot \mathsf{merid}(x_0)^{-1} = q \cdot \mathsf{merid}(x_1)^{-1}\right) \to \left\| \textstyle\sum_{(x:X)} (\mathsf{merid}(x) = q) \right\|_{2n}.$$

Now for each $x_1, x_2 : X$, this type is $2n$-truncated, while X is n-connected. Thus, by Lemma 8.6.2, it suffices to define this map when x_1 is x_0, when x_2 is x_0, and check that they agree when both are x_0.

When x_1 is x_0, the hypothesis is $r : \mathsf{merid}(x_2) \cdot \mathsf{merid}(x_0)^{-1} = q \cdot \mathsf{merid}(x_0)^{-1}$. Thus, by canceling $\mathsf{merid}(x_0)^{-1}$ from r to get $r' : \mathsf{merid}(x_2) = q$, so we can define the image to be $|(x_2, r')|_{2n}$.

When x_2 is x_0, the hypothesis is $r : \mathsf{merid}(x_0) \cdot \mathsf{merid}(x_0)^{-1} = q \cdot \mathsf{merid}(x_1)^{-1}$. Rearranging this, we obtain $r'' : \mathsf{merid}(x_1) = q$, and we can define the image to be $|(x_1, r'')|_{2n}$.

Finally, when both x_1 and x_2 are x_0, it suffices to show the resulting r' and r'' agree; this is an easy lemma about path composition. This completes the definition of (8.6.9). To show that it is a family of equivalences, since being an equivalence is a mere proposition and $x_0 : \mathbf{1} \to X$ is (at least) (-1)-connected, it suffices to assume x_1 is x_0. In this case, inspecting the above construction we see that it is essentially the $2n$-truncation of the function that cancels $\mathsf{merid}(x_0)^{-1}$, which is an equivalence. □

In addition to (8.6.7) and (8.6.8), we will need to extract from the construction of code some information about how it acts on paths. For this we use the following lemma.

Lemma 8.6.10. *Let $A : \mathcal{U}$, $B : A \to \mathcal{U}$, and $C : \prod_{(a:A)} B(a) \to \mathcal{U}$, and also $a_1, a_2 : A$ with $m : a_1 = a_2$ and $b : B(a_2)$. Then the function*

$$\mathsf{transport}^{\widehat{C}}(\mathsf{pair}^{=}(m, t), -) : C(a_1, \mathsf{transport}^{B}(m^{-1}, b)) \to C(a_2, b),$$

where $t : \mathsf{transport}^{B}(m, \mathsf{transport}^{B}(m^{-1}, b)) = b$ is the obvious coherence path and $\widehat{C} : (\sum_{(a:A)} B(a)) \to \mathcal{U}$ is the uncurried form of C, is equal to the equivalence obtained by univalence from the composite

$$\begin{aligned} C(a_1, \mathsf{transport}^{B}(m^{-1}, b)) &= \mathsf{transport}^{\lambda a.\, B(a) \to \mathcal{U}}(m, C(a_1))(b) && \text{(by (2.9.4))} \\ &= C(a_2, b). && \text{(by } \mathsf{happly}(\mathsf{apd}_C(m), b)\text{)} \end{aligned}$$

Proof. By path induction, we may assume a_2 is a_1 and m is refl_{a_1}, in which case both functions are the identity. □

We apply this lemma with $A :\equiv \Sigma X$ and $B :\equiv \lambda y.\, (\mathsf{N} = y)$ and $C :\equiv \mathsf{code}$, while $a_1 :\equiv \mathsf{N}$ and $a_2 :\equiv \mathsf{S}$ and $m :\equiv \mathsf{merid}(x_1)$ for some $x_1 : X$, and finally $b :\equiv q$ is some path $\mathsf{N} = \mathsf{S}$. The computation rule for induction over ΣX identifies $\mathsf{apd}_C(m)$ with a path constructed in a certain way out of univalence and function extensionality. The second function described in Lemma 8.6.10 essentially consists of undoing these applications of univalence and function extensionality, reducing back to the particular functions (8.6.9) that we defined using Lemma 8.6.2. Therefore, Lemma 8.6.10 says that transporting along $\mathsf{pair}^{=}(q, t)$ essentially recovers these functions.

Finally, by construction, when x_1 or x_2 coincides with x_0 and the input is in the image of $|-|_{2n}$, we know more explicitly what these functions are. Thus, for any $x_2 : X$, we have

$$\mathsf{transport}^{\hat{\mathsf{code}}}(\mathsf{pair}^{=}(\mathsf{merid}(x_0), t), |(x_2, r)|_{2n}) = \left|(x_1, r')\right|_{2n} \tag{8.6.11}$$

where $r : \mathsf{merid}(x_2) \cdot \mathsf{merid}(x_0)^{-1} = \mathsf{transport}^{B}(\mathsf{merid}(x_0)^{-1}, q)$ is arbitrary as before, and $r' : \mathsf{merid}(x_2) = q$ is obtained from r by identifying its end point with $q \cdot \mathsf{merid}(x_0)^{-1}$ and canceling $\mathsf{merid}(x_0)^{-1}$. Similarly, for any $x_1 : X$, we have

$$\mathsf{transport}^{\hat{\mathsf{code}}}(\mathsf{pair}^{=}(\mathsf{merid}(x_1), t), |(x_0, r)|_{2n}) = \left|(x_1, r'')\right|_{2n} \tag{8.6.12}$$

where $r : \mathsf{merid}(x_0) \cdot \mathsf{merid}(x_0)^{-1} = \mathsf{transport}^{B}(\mathsf{merid}(x_1)^{-1}, q)$, and $r'' : \mathsf{merid}(x_1) = q$ is obtained by identifying its end point and rearranging paths.

Proof of Theorem 8.6.4. It remains to show that $\mathsf{code}(y, p)$ is contractible for each $y : \Sigma X$ and $p : \mathsf{N} = y$. First we must choose a center of contraction, say $c(y, p) : \mathsf{code}(y, p)$. This corresponds to the definition of the function encode in our previous proofs, so we define it by transport. Note that in the special case when y is N and p is $\mathsf{refl}_{\mathsf{N}}$, we have

$$\mathsf{code}(\mathsf{N}, \mathsf{refl}_{\mathsf{N}}) \equiv \left\| \textstyle\sum_{(x:X)} (\mathsf{merid}(x) \cdot \mathsf{merid}(x_0)^{-1} = \mathsf{refl}_{\mathsf{N}}) \right\|_{2n}.$$

Thus, we can choose $c(\mathsf{N}, \mathsf{refl}_{\mathsf{N}}) :\equiv \left|(x_0, \mathsf{rinv}_{\mathsf{merid}(x_0)})\right|_{2n}$, where rinv_q is the obvious path $q \cdot q^{-1} = \mathsf{refl}$ for any q. We can now obtain $c : \prod_{(y:\Sigma X)} \prod_{(p:\mathsf{N}=y)} \mathsf{code}(y, p)$ by path induction on p, but it will be important below that we can also give a concrete definition in terms of transport:

$$c(y, p) :\equiv \mathsf{transport}^{\hat{\mathsf{code}}}(\mathsf{pair}^{=}(p, \mathsf{tid}_p), c(\mathsf{N}, \mathsf{refl}_{\mathsf{N}}))$$

where $\hat{\mathsf{code}} : \left(\sum_{(y:\Sigma X)} (\mathsf{N} = y)\right) \to \mathcal{U}$ is the uncurried version of code, and $\mathsf{tid}_p : p_*(\mathsf{refl}) = p$ is a standard lemma.

Next, we must show that every element of $\mathsf{code}(y, p)$ is equal to $c(y, p)$. Again, by path induction, it suffices to assume y is N and p is $\mathsf{refl}_{\mathsf{N}}$. In fact, we will prove it more generally when y is N and p is arbitrary. That is, we will show that for any $p : \mathsf{N} = \mathsf{N}$ and $d : \mathsf{code}(\mathsf{N}, p)$ we have $d = c(\mathsf{N}, p)$. Since this equality is a $(2n-1)$-type, we may assume d is of the form $|(x_1, r)|_{2n}$ for some $x_1 : X$ and $r : \mathsf{merid}(x_1) \cdot \mathsf{merid}(x_0)^{-1} = p$.

Now by a further path induction, we may assume that r is reflexivity, and p is $\mathsf{merid}(x_1) \cdot \mathsf{merid}(x_0)^{-1}$. (This is why we generalized to arbitrary p above.) Thus, we have to prove that

$$\left|(x_1, \mathsf{refl}_{\mathsf{merid}(x_1) \cdot \mathsf{merid}(x_0)^{-1}})\right|_{2n} = c\left(\mathsf{N}, \mathsf{refl}_{\mathsf{merid}(x_1) \cdot \mathsf{merid}(x_0)^{-1}}\right). \tag{8.6.13}$$

By definition, the right-hand side of this equality is

$$\begin{aligned}
&\mathsf{transport}^{\hat{\mathsf{code}}}\Big(\mathsf{pair}^{=}(\mathsf{merid}(x_1)\cdot\mathsf{merid}(x_0)^{-1},_),\ |(x_0,_)|_{2n}\Big)\\
&\qquad=\mathsf{transport}^{\hat{\mathsf{code}}}\Big(\mathsf{pair}^{=}(\mathsf{merid}(x_0)^{-1},_),\\
&\qquad\qquad\mathsf{transport}^{\hat{\mathsf{code}}}\Big(\mathsf{pair}^{=}(\mathsf{merid}(x_1),_),\ |(x_0,_)|_{2n}\Big)\Big)\\
&\qquad\qquad=\mathsf{transport}^{\hat{\mathsf{code}}}\Big(\mathsf{pair}^{=}(\mathsf{merid}(x_0)^{-1},_),\ |(x_1,_)|_{2n}\Big)=|(x_1,_)|_{2n}
\end{aligned}$$

where the underscore _ ought to be filled in with suitable coherence paths. Here the first step is functoriality of transport, the second invokes (8.6.12), and the third invokes (8.6.11) (with transport moved to the other side). Thus we have the same first component as the left-hand side of (8.6.13). We leave it to the reader to verify that the coherence paths all cancel, giving reflexivity in the second component. □

Corollary 8.6.14 (Freudenthal Equivalence)**.** *Suppose that X is n-connected and pointed, with $n \geq 0$. Then $\|X\|_{2n} \simeq \|\Omega\Sigma(X)\|_{2n}$.*

Proof. By Theorem 8.6.4, σ is $2n$-connected. By Lemma 7.5.14, it is therefore an equivalence on $2n$-truncations. □

One important corollary of the Freudenthal suspension theorem is that the homotopy groups of spheres are stable in a certain range (these are the northeast-to-southwest diagonals in Table 8.1):

Corollary 8.6.15 (Stability for Spheres)**.** *If $k \leq 2n-2$, then $\pi_{k+1}(S^{n+1}) = \pi_k(S^n)$.*

Proof. Assume $k \leq 2n-2$. By Corollary 8.2.2, $\mathbb{S}^n$ is $(n-1)$-connected. Therefore, by Corollary 8.6.14,

$$\|\Omega(\Sigma(\mathbb{S}^n))\|_{2(n-1)} = \|\mathbb{S}^n\|_{2(n-1)}.$$

By Lemma 7.3.15, because $k \leq 2(n-1)$, applying $\|-\|_k$ to both sides shows that this equation holds for k:

$$\|\Omega(\Sigma(\mathbb{S}^n))\|_k = \|\mathbb{S}^n\|_k. \tag{8.6.16}$$

Then, the main idea of the proof is as follows; we omit checking that these equivalences act appropriately on the base points of these spaces, and that for $k > 0$ the equivalences respect multiplication:

$$\begin{aligned}
\pi_{k+1}(\mathbb{S}^{n+1}) &\equiv \left\|\Omega^{k+1}(\mathbb{S}^{n+1})\right\|_0\\
&\equiv \left\|\Omega^k(\Omega(\mathbb{S}^{n+1}))\right\|_0\\
&\equiv \left\|\Omega^k(\Omega(\Sigma(\mathbb{S}^n)))\right\|_0\\
&= \Omega^k(\|(\Omega(\Sigma(\mathbb{S}^n)))\|_k) && \text{(by Theorem 7.3.12)}\\
&= \Omega^k(\|\mathbb{S}^n\|_k) && \text{(by (8.6.16))}\\
&= \left\|\Omega^k(\mathbb{S}^n)\right\|_0 && \text{(by Theorem 7.3.12)}\\
&\equiv \pi_k(\mathbb{S}^n). && \square
\end{aligned}$$

This means that once we have calculated one entry in one of these stable diagonals, we know all of them. For example:

Theorem 8.6.17. *$\pi_n(\mathbb{S}^n) = \mathbb{Z}$ for every $n \geq 1$.*

Proof. The proof is by induction on n. We already have $\pi_1(\mathbb{S}^1) = \mathbb{Z}$ (Corollary 8.1.11) and $\pi_2(\mathbb{S}^2) = \mathbb{Z}$ (Corollary 8.5.2). When $n \geq 2$, $n \leq (2n-2)$. Therefore, by Corollary 8.6.15, $\pi_{n+1}(S^{n+1}) = \pi_n(S^n)$, and this equivalence, combined with the inductive hypothesis, gives the result. □

Corollary 8.6.18. *$\mathbb{S}^{n+1}$ is not an n-type for any $n \geq -1$.*

Corollary 8.6.19. *$\pi_3(\mathbb{S}^2) = \mathbb{Z}$.*

Proof. By Corollary 8.5.2, $\pi_3(\mathbb{S}^2) = \pi_3(\mathbb{S}^3)$. But by Theorem 8.6.17, $\pi_3(\mathbb{S}^3) = \mathbb{Z}$. □

8.7 The van Kampen theorem

The van Kampen theorem calculates the fundamental group π_1 of a (homotopy) pushout of spaces. It is traditionally stated for a topological space X which is the union of two open subspaces U and V, but in homotopy-theoretic terms this is just a convenient way of ensuring that X is the pushout of U and V over their intersection. Thus, we will prove a version of the van Kampen theorem for arbitrary pushouts.

In this section we will describe a proof of the van Kampen theorem which uses the same encode-decode method that we used for $\pi_1(\mathbb{S}^1)$ in §8.1. There is also a more homotopy-theoretic approach; see Exercise 9.11.

We need a more refined version of the encode-decode method. In §8.1 (as well as in §§2.12 and 2.13) we used it to characterize the path space of a (higher) inductive type W — deriving as a consequence a characterization of the loop space $\Omega(W)$, and thereby also of its 0-truncation $\pi_1(W)$. In the van Kampen theorem, our goal is only to characterize the fundamental group $\pi_1(W)$, and we do not have any explicit description of the loop spaces or the path spaces to use.

It turns out that we can use the same technique directly for a truncated version of the path fibration, thereby characterizing not only the fundamental *group* $\pi_1(W)$, but also the whole fundamental *groupoid*. Specifically, for a type X, write $\Pi_1 X : X \to X \to \mathcal{U}$ for the 0-truncation of its identity type, i.e. $\Pi_1 X(x,y) :\equiv \|x = y\|_0$. Note that we have induced groupoid operations

$$\begin{aligned}
(- \cdot -) &: \Pi_1 X(x,y) \to \Pi_1 X(y,z) \to \Pi_1 X(x,z) \\
(-)^{-1} &: \Pi_1 X(x,y) \to \Pi_1 X(y,x) \\
\mathsf{refl}_x &: \Pi_1 X(x,x) \\
\mathsf{ap}_f &: \Pi_1 X(x,y) \to \Pi_1 Y(fx,fy)
\end{aligned}$$

for which we use the same notation as the corresponding operations on paths.

8.7.1 Naive van Kampen

We begin with a "naive" version of the van Kampen theorem, which is useful but not quite as useful as the classical version. In §8.7.2 we will improve it to a more useful version.

Given types A, B, C and functions $f : A \to B$ and $g : A \to C$, let P be their pushout $B \sqcup^A C$. As we saw in §6.8, P is the higher inductive type generated by

- $i : B \to P$,
- $j : C \to P$, and
- for all $x : A$, a path $kx : ifx = jgx$.

Define $\mathsf{code} : P \to P \to \mathcal{U}$ by double induction on P as follows.

- $\mathsf{code}(ib, ib')$ is a set-quotient (see §6.10) of the type of sequences
$$(b, p_0, x_1, q_1, y_1, p_1, x_2, q_2, y_2, p_2, \dots, y_n, p_n, b')$$
where
  - $n : \mathbb{N}$
  - $x_k : A$ and $y_k : A$ for $0 < k \leq n$
  - $p_0 : \Pi_1 B(b, fx_1)$ and $p_n : \Pi_1 B(fy_n, b')$ for $n > 0$, and $p_0 : \Pi_1 B(b, b')$ for $n = 0$
  - $p_k : \Pi_1 B(fy_k, fx_{k+1})$ for $1 \leq k < n$
  - $q_k : \Pi_1 C(gx_k, gy_k)$ for $1 \leq k \leq n$

  The quotient is generated by the following equalities:
$$(\dots, q_k, y_k, \mathsf{refl}_{fy_k}, y_k, q_{k+1}, \dots) = (\dots, q_k \cdot q_{k+1}, \dots)$$
$$(\dots, p_k, x_k, \mathsf{refl}_{gx_k}, x_k, p_{k+1}, \dots) = (\dots, p_k \cdot p_{k+1}, \dots)$$
  (see Remark 8.7.3 below). We leave it to the reader to define this type of sequences precisely as an inductive type.
- $\mathsf{code}(jc, jc')$ is identical, with the roles of B and C reversed. We likewise notationally reverse the roles of x and y, and of p and q.
- $\mathsf{code}(ib, jc)$ and $\mathsf{code}(jc, ib)$ are similar, with the parity changed so that they start in one type and end in the other.
- For $a : A$ and $b : B$, we require an equivalence
$$\mathsf{code}(ib, ifa) \simeq \mathsf{code}(ib, jga). \tag{8.7.1}$$
  We define this to consist of the two functions defined on sequences by
$$(\dots, y_n, p_n, fa) \mapsto (\dots, y_n, p_n, a, \mathsf{refl}_{ga}, ga),$$
$$(\dots, x_n, p_n, a, \mathsf{refl}_{fa}, fa) \leftarrow\!\shortmid (\dots, x_n, p_n, ga).$$
  Both of these functions are easily seen to respect the equivalence relations, and hence to define functions on the types of codes. The left-to-right-to-left composite is
$$(\dots, y_n, p_n, fa) \mapsto (\dots, y_n, p_n, a, \mathsf{refl}_{ga}, a, \mathsf{refl}_{fa}, fa)$$
  which is equal to the identity by a generating equality of the quotient. The other composite is analogous. Thus we have defined an equivalence (8.7.1).
- Similarly, we require equivalences
$$\mathsf{code}(jc, ifa) \simeq \mathsf{code}(jc, jga)$$
$$\mathsf{code}(ifa, ib) \simeq (jga, ib)$$
$$\mathsf{code}(ifa, jc) \simeq (jga, jc)$$
  all of which are defined in exactly the same way (the second two by adding reflexivity terms on the beginning rather than the end).

- Finally, we need to know that for $a, a' : A$, the following diagram commutes:

$$\begin{array}{ccc} \mathsf{code}(ifa, ifa') & \longrightarrow & \mathsf{code}(ifa, jga') \\ \downarrow & & \downarrow \\ \mathsf{code}(jga, ifa') & \longrightarrow & \mathsf{code}(jga, jga') \end{array} \tag{8.7.2}$$

  This amounts to saying that if we add something to the beginning and then something to the end of a sequence, we might as well have done it in the other order.

Remark 8.7.3. One might expect to see in the definition of code some additional generating equations for the set-quotient, such as

$$\begin{aligned} (\dots, p_{k-1} \cdot fw, x'_k, q_k, \dots) &= (\dots, p_{k-1}, x_k, gw \cdot q_k, \dots) && (\text{for } w : \Pi_1 A(x_k, x'_k)) \\ (\dots, q_k \cdot gw, y'_k, p_k, \dots) &= (\dots, q_k, y_k, fw \cdot p_k, \dots). && (\text{for } w : \Pi_1 A(y_k, y'_k)) \end{aligned}$$

However, these are not necessary! In fact, they follow automatically by path induction on w. This is the main difference between the "naive" van Kampen theorem and the more refined one we will consider in the next subsection.

Continuing on, we can characterize transporting in the fibration code:

- For $p : b =_B b'$ and $u : P$, we have

$$\mathsf{transport}^{b \mapsto \mathsf{code}(u,ib)}(p, (\dots, y_n, p_n, b)) = (\dots, y_n, p_n \cdot p, b').$$

- For $q : c =_C c'$ and $u : P$, we have

$$\mathsf{transport}^{c \mapsto \mathsf{code}(u,jc)}(q, (\dots, x_n, q_n, c)) = (\dots, x_n, q_n \cdot q, c').$$

Here we are abusing notation by using the same name for a path in X and its image in $\Pi_1 X$. Note that transport in $\Pi_1 X$ is also given by concatenation with (the image of) a path. From this we can prove the above statements by induction on u. We also have:

- For $a : A$ and $u : P$,

$$\mathsf{transport}^{v \mapsto \mathsf{code}(u,v)}(ha, (\dots, y_n, p_n, fa)) = (\dots, y_n, p_n, a, \mathsf{refl}_{ga}, ga).$$

This follows essentially from the definition of code.

We also construct a function

$$r : \prod_{u:P} \mathsf{code}(u, u)$$

by induction on u as follows:

$$\begin{aligned} rib &:\equiv (b, \mathsf{refl}_b, b) \\ rjc &:\equiv (c, \mathsf{refl}_c, c) \end{aligned}$$

and for rka we take the composite equality

$$\begin{aligned} (ka, ka)_*(fa, \mathsf{refl}_{fa}, fa) &= (ga, \mathsf{refl}_{ga}, a, \mathsf{refl}_{fa}, a, \mathsf{refl}_{ga}, ga) \\ &= (ga, \mathsf{refl}_{ga}, ga) \end{aligned}$$

where the first equality is by the observation above about transporting in code, and the second is an instance of the set quotient relation used to define code.

We will now prove:

Theorem 8.7.4 (Naive van Kampen theorem)**.** *For all $u, v : P$ there is an equivalence*

$$\Pi_1 P(u,v) \simeq \mathsf{code}(u,v).$$

Proof. To define a function

$$\mathsf{encode} : \Pi_1 P(u,v) \to \mathsf{code}(u,v)$$

it suffices to define a function $(u =_P v) \to \mathsf{code}(u,v)$, since $\mathsf{code}(u,v)$ is a set. We do this by transport:

$$\mathsf{encode}(p) :\equiv \mathsf{transport}^{v \mapsto \mathsf{code}(u,v)}(p, r(u)).$$

Now to define

$$\mathsf{decode} : \mathsf{code}(u,v) \to \Pi_1 P(u,v)$$

we proceed as usual by induction on $u, v : P$. In each case for u and v, we apply i or j to all the equalities p_k and q_k as appropriate and concatenate the results in P, using h to identify the endpoints. For instance, when $u \equiv ib$ and $v \equiv ib'$, we define

$$\mathsf{decode}(b, p_0, x_1, q_1, y_1, p_1, \dots, y_n, p_n, b') :\equiv (p_0) \cdot h(x_1) \cdot j(q_1) \cdot h(y_1)^{-1} \cdot i(p_1) \cdot \dots \cdot h(y_n)^{-1} \cdot i(p_n). \tag{8.7.5}$$

This respects the set-quotient equivalence relation and the equivalences such as (8.7.1), since $h : fi \sim gj$ is natural and f and g are functorial.

As usual, to show that the composite

$$\Pi_1 P(u,v) \xrightarrow{\mathsf{encode}} \mathsf{code}(u,v) \xrightarrow{\mathsf{decode}} \Pi_1 P(u,v)$$

is the identity, we first peel off the 0-truncation (since the codomain is a set) and then apply path induction. The input refl_u goes to ru, which then goes back to refl_u (applying a further induction on u to decompose $\mathsf{decode}(ru)$).

Finally, consider the composite

$$\mathsf{code}(u,v) \xrightarrow{\mathsf{decode}} \Pi_1 P(u,v) \xrightarrow{\mathsf{encode}} \mathsf{code}(u,v).$$

We proceed by induction on $u, v : P$. When $u \equiv ib$ and $v \equiv ib'$, this composite is

$$\begin{aligned}
(b, p_0, x_1, q_1, y_1, p_1, \dots, y_n, p_n, b') &\mapsto \left(ip_0 \cdot hx_1 \cdot jq_1 \cdot hy_1{}^{-1} \cdot ip_1 \cdot \dots \cdot hy_n{}^{-1} \cdot ip_n\right)_*(rib) \\
&= (ip_n)_* \cdots (jq_1)_*(hx_1)_*(ip_0)_*(b, \mathsf{refl}_b, b) \\
&= (ip_n)_* \cdots (jq_1)_*(hx_1)_*(b, p_0, if x_1) \\
&= (ip_n)_* \cdots (jq_1)_*(b, p_0, x_1, \mathsf{refl}_{gx_1}, jgx_1) \\
&= (ip_n)_* \cdots (b, p_0, x_1, q_1, jgy_1) \\
&= \quad \vdots \\
&= (b, p_0, x_1, q_1, y_1, p_1, \dots, y_n, p_n, b').
\end{aligned}$$

i.e., the identity function. (To be precise, there is an implicit inductive argument needed here.) The other three point cases are analogous, and the path cases are trivial since all the types are sets. □

Theorem 8.7.4 allows us to calculate the fundamental groups of many types, provided A is a set, for in that case, each $\mathsf{code}(u,v)$ is, by definition, a set-quotient of a *set* by a relation.

Example 8.7.6. Let $A :\equiv \mathbf{2}$, $B :\equiv \mathbf{1}$, and $C :\equiv \mathbf{1}$. Then $P \simeq S^1$. Inspecting the definition of, say, $\mathsf{code}(i(\star), i(\star))$, we see that the paths all may as well be trivial, so the only information is in the sequence of elements $x_1, y_1, \dots, x_n, y_n : \mathbf{2}$. Moreover, if we have $x_k = y_k$ or $y_k = x_{k+1}$ for any k, then the set-quotient relations allow us to excise both of those elements. Thus, every such sequence is equal to a canonical *reduced* one in which no two adjacent elements are equal. Clearly such a reduced sequence is uniquely determined by its length (a natural number n) together with, if $n > 1$, the information of whether x_1 is $0_{\mathbf{2}}$ or $1_{\mathbf{2}}$, since that determines the rest of the sequence uniquely. And these data can, of course, be identified with an integer, where n is the absolute value and x_1 encodes the sign. Thus we recover $\pi_1(S^1) \cong \mathbb{Z}$.

Since Theorem 8.7.4 asserts only a bijection of families of sets, this isomorphism $\pi_1(S^1) \cong \mathbb{Z}$ is likewise only a bijection of sets. We could, however, define a concatenation operation on code (by concatenating sequences) and show that encode and decode form an isomorphism respecting this structure. (In the language of Chapter 9, these would be "pregroupoids".) We leave the details to the reader.

Example 8.7.7. More generally, let $B :\equiv \mathbf{1}$ and $C :\equiv \mathbf{1}$ but A be arbitrary, so that P is the suspension of A. Then once again the paths p_k and q_k are trivial, so that the only information in a path code is a sequence of elements $x_1, y_1, \dots, x_n, y_n : A$. The first two generating equalities say that adjacent equal elements can be canceled, so it makes sense to think of this sequence as a word of the form

$$x_1 y_1^{-1} x_2 y_2^{-1} \cdots x_n y_n^{-1}$$

in a group. Indeed, it looks similar to the free group on A (or equivalently on $\|A\|_0$; see Remark 6.11.8), but we are considering only words that start with a non-inverted element, alternate between inverted and non-inverted elements, and end with an inverted one. This effectively reduces the size of the generating set by one. For instance, if A has a point $a : A$, then we can identify $\pi_1(\Sigma A)$ with the group presented by $\|A\|_0$ as generators with the relation $|a|_0 = e$; see Exercises 8.10 and 8.11 for details.

Example 8.7.8. Let $A :\equiv \mathbf{1}$ and B and C be arbitrary, so that f and g simply equip B and C with basepoints b and c, say. Then P is the *wedge* $B \vee C$ of B and C (the coproduct in the category of based spaces). In this case, it is the elements x_k and y_k which are trivial, so that the only information is a sequence of loops $(p_0, q_1, p_1, \dots, p_n)$ with $p_k : \pi_1(B, b)$ and $q_k : \pi_1(C, c)$. Such sequences, modulo the equivalence relation we have imposed, are easily identified with the explicit description of the *free product* of the groups $\pi_1(B, b)$ and $\pi_1(C, c)$, as constructed in §6.11. Thus, we have $\pi_1(B \vee C) \cong \pi_1(B) * \pi_1(C)$.

However, Theorem 8.7.4 stops just short of being the full classical van Kampen theorem, which handles the case where A is not necessarily a set, and states that $\pi_1(B \sqcup^A C) \cong \pi_1(B) *_{\pi_1(A)} \pi_1(C)$ (with base point coming from A). Indeed, the conclusion of Theorem 8.7.4 says nothing at all about $\pi_1(A)$; the paths in A are "built into the quotienting" in a type-theoretic way that makes it hard to extract explicit information, in that $\mathsf{code}(u, v)$ is a set-quotient of a non-set by a relation. For this reason, in the next subsection we consider a better version of the van Kampen theorem.

8.7.2 The van Kampen theorem with a set of basepoints

The improvement of van Kampen we present now is closely analogous to a similar improvement in classical algebraic topology, where A is equipped with a *set S of base points*. In fact, it turns out to be unnecessary for our proof to assume that the "set of basepoints" is a *set* — it might just as

well be an arbitrary type; the utility of assuming S is a set arises later, when applying the theorem to obtain computations. What is important is that S contains at least one point in each connected component of A. We state this in type theory by saying that we have a type S and a function $k : S \to A$ which is surjective, i.e. (-1)-connected. If $S \equiv A$ and k is the identity function, then we will recover the naive van Kampen theorem. Another example to keep in mind is when A is pointed and (0-)connected, with $k : \mathbf{1} \to A$ the point: by Lemmas 7.5.2 and 7.5.11 this map is surjective just when A is 0-connected.

Let $A, B, C, f, g, P, i, j, h$ be as in the previous section. We now define, given our surjective map $k : S \to A$, an auxiliary type which improves the connectedness of k. Let T be the higher inductive type generated by

- A function $\ell : S \to T$, and
- For each $s, s' : S$, a function $m : (ks =_A ks') \to (\ell s =_T \ell s')$.

There is an obvious induced function $\bar{k} : T \to A$ such that $\bar{k}\ell = k$, and any $p : ks = ks'$ is equal to the composite $ks = \bar{k}\ell s \overset{\bar{k}mp}{=} \bar{k}\ell s' = ks'$.

Lemma 8.7.9. *$\bar{k}$ is 0-connected.*

Proof. We must show that for all $a : A$, the 0-truncation of the type $\sum_{(t:T)}(\bar{k}t = a)$ is contractible. Since contractibility is a mere proposition and k is (-1)-connected, we may assume that $a = ks$ for some $s : S$. Now we can take the center of contraction to be $|(\ell s, q)|_0$ where q is the equality $\bar{k}\ell s = ks$.

It remains to show that for any $\phi : \left\|\sum_{(t:T)}(\bar{k}t = ks)\right\|_0$ we have $\phi = |(\ell s, q)|_0$. Since the latter is a mere proposition, and in particular a set, we may assume that $\phi = |(t, p)|_0$ for $t : T$ and $p : \bar{k}t = ks$.

Now we can do induction on $t : T$. If $t \equiv \ell s'$, then $ks' = \bar{k}\ell s' \overset{p}{=} ks$ yields via m an equality $\ell s = \ell s'$. Hence by definition of $\bar{k}$ and of equality in homotopy fibers, we obtain an equality $(ks', p) = (ks, q)$, and thus $|(ks', p)|_0 = |(ks, q)|_0$. Next we must show that as t varies along m these equalities agree. But they are equalities in a set (namely $\left\|\sum_{(t:T)}(\bar{k}t = ks)\right\|_0$), and hence this is automatic. □

Remark 8.7.10. T can be regarded as the (homotopy) coequalizer of the "kernel pair" of k. If S and A were sets, then the (-1)-connectivity of k would imply that A is the 0-truncation of this coequalizer (see Chapter 10). For general types, higher topos theory suggests that (-1)-connectivity of k will imply instead that A is the colimit (a.k.a. "geometric realization") of the "simplicial kernel" of k. The type T is the colimit of the "1-skeleton" of this simplicial kernel, so it makes sense that it improves the connectivity of k by 1. More generally, we might expect the colimit of the n-skeleton to improve connectivity by n.

Now we define $\mathsf{code} : P \to P \to \mathcal{U}$ by double induction as follows

- $\mathsf{code}(ib, ib')$ is now a set-quotient of the type of sequences

$$(b, p_0, x_1, q_1, y_1, p_1, x_2, q_2, y_2, p_2, \dots, y_n, p_n, b')$$

  where

  - $n : \mathbb{N}$,

- $x_k : S$ and $y_k : S$ for $0 < k \leq n$,
- $p_0 : \Pi_1 B(b, fkx_1)$ and $p_n : \Pi_1 B(fky_n, b')$ for $n > 0$, and $p_0 : \Pi_1 B(b, b')$ for $n = 0$,
- $p_k : \Pi_1 B(fky_k, fkx_{k+1})$ for $1 \leq k < n$,
- $q_k : \Pi_1 C(gkx_k, gky_k)$ for $1 \leq k \leq n$.

The quotient is generated by the following equalities (see Remark 8.7.3):

$$\begin{aligned}
(\dots, q_k, y_k, \mathsf{refl}_{fy_k}, y_k, q_{k+1}, \dots) &= (\dots, q_k \cdot q_{k+1}, \dots) \\
(\dots, p_k, x_k, \mathsf{refl}_{gx_k}, x_k, p_{k+1}, \dots) &= (\dots, p_k \cdot p_{k+1}, \dots) \\
(\dots, p_{k-1} \cdot fw, x'_k, q_k, \dots) &= (\dots, p_{k-1}, x_k, gw \cdot q_k, \dots) && \text{(for } w : \Pi_1 A(kx_k, kx'_k)) \\
(\dots, q_k \cdot gw, y'_k, p_k, \dots) &= (\dots, q_k, y_k, fw \cdot p_k, \dots). && \text{(for } w : \Pi_1 A(ky_k, ky'_k))
\end{aligned}$$

We will need below the definition of the case of decode on such a sequence, which as before concatenates all the paths p_k and q_k together with instances of h to give an element of $\Pi_1 P(ifb, ifb')$, cf. (8.7.5). As before, the other three point cases are nearly identical.

- For $a : A$ and $b : B$, we require an equivalence

$$\mathsf{code}(ib, ifa) \simeq \mathsf{code}(ib, jga). \tag{8.7.11}$$

Since code is set-valued, by Lemma 8.7.9 we may assume that $a = \overline{k}t$ for some $t : T$. Next, we can do induction on t. If $t \equiv \ell s$ for $s : S$, then we define (8.7.11) as in §8.7.1:

$$\begin{aligned}
(\dots, y_n, p_n, fks) &\mapsto (\dots, y_n, p_n, s, \mathsf{refl}_{gks}, gks), \\
(\dots, x_n, p_n, s, \mathsf{refl}_{fks}, fks) &\mapsfrom (\dots, x_n, p_n, gks).
\end{aligned}$$

These respect the equivalence relations, and define quasi-inverses just as before. Now suppose t varies along $m_{s,s'}(w)$ for some $w : ks = ks'$; we must show that (8.7.11) respects transporting along $\overline{k}mw$. By definition of $\overline{k}$, this essentially boils down to transporting along w itself. By the characterization of transport in path types, what we need to show is that

$$w_*(\dots, y_n, p_n, fks) = (\dots, y_n, p_n \cdot fw, fks')$$

is mapped by (8.7.11) to

$$w_*(\dots, y_n, p_n, s, \mathsf{refl}_{gks}, gks) = (\dots, y_n, p_n, s, \mathsf{refl}_{gks} \cdot gw, gks')$$

But this follows directly from the new generators we have imposed on the set-quotient relation defining code.

- The other three requisite equivalences are defined similarly.
- Finally, since the commutativity (8.7.2) is a mere proposition, by (-1)-connectedness of k we may assume that $a = ks$ and $a' = ks'$, in which case it follows exactly as before.

Theorem 8.7.12 (van Kampen with a set of basepoints)**.** *For all $u, v : P$ there is an equivalence*

$$\Pi_1 P(u, v) \simeq \mathsf{code}(u, v).$$

with code *defined as in this section.*

Proof. Basically just like before. To show that decode respects the new generators of the quotient relation, we use the naturality of h. And to show that decode respects the equivalences such as (8.7.11), we need to induct on $\bar{k}$ and on T in order to decompose those equivalences into their definitions, but then it becomes again simply functoriality of f and g. The rest is easy. In particular, no additional argument is required for $\mathsf{encode} \circ \mathsf{decode}$, since the goal is to prove an equality in a set, and so the case of h is trivial. □

Theorem 8.7.12 allows us to calculate the fundamental group of a space A, even when A is not a set, provided S is a set, for in that case, each $\mathsf{code}(u, v)$ is, by definition, a set-quotient of a *set* by a relation. In that respect, it is an improvement over Theorem 8.7.4.

Example 8.7.13. Suppose $S :\equiv \mathbf{1}$, so that A has a basepoint $a :\equiv k(\star)$ and is connected. Then code for loops in the pushout can be identified with alternating sequences of loops in $\pi_1(B, f(a))$ and $\pi_1(C, g(a))$, modulo an equivalence relation which allows us to slide elements of $\pi_1(A, a)$ between them (after applying f and g respectively). Thus, $\pi_1(P)$ can be identified with the *amalgamated free product* $\pi_1(B) *_{\pi_1(A)} \pi_1(C)$ (the pushout in the category of groups), as constructed in §6.11. This (in the case when B and C are open subspaces of P and A their intersection) is probably the most classical version of the van Kampen theorem.

Example 8.7.14. As a special case of Example 8.7.13, suppose additionally that $C :\equiv \mathbf{1}$, so that P is the cofiber B/A. Then every loop in C is equal to reflexivity, so the relations on path codes allow us to collapse all sequences to a single loop in B. The additional relations require that multiplying on the left, right, or in the middle by an element in the image of $\pi_1(A)$ is the identity. We can thus identify $\pi_1(B/A)$ with the quotient of the group $\pi_1(B)$ by the normal subgroup generated by the image of $\pi_1(A)$.

Example 8.7.15. As a further special case of Example 8.7.14, let $B :\equiv S^1 \vee S^1$, let $A :\equiv S^1$, and let $f : A \to B$ pick out the composite loop $p \cdot q \cdot p^{-1} \cdot q^{-1}$, where p and q are the generating loops in the two copies of S^1 comprising B. Then P is a presentation of the torus T^2. Indeed, it is not hard to identify P with the presentation of T^2 as described in §6.7, using the cone on a particular loop. Thus, $\pi_1(T^2)$ is the quotient of the free group on two generators (i.e., $\pi_1(B)$) by the relation $p \cdot q \cdot p^{-1} \cdot q^{-1} = 1$. This clearly yields the free *abelian* group on two generators, which is $\mathbb{Z} \times \mathbb{Z}$.

Example 8.7.16. More generally, any CW complex can be obtained by repeatedly "coning off" spheres, as described in §6.7. That is, we start with a set X_0 of points ("0-cells"), which is the "0-skeleton" of the CW complex. We take the pushout

$$\begin{array}{ccc} S_1 \times \mathbb{S}^0 & \xrightarrow{f_1} & X_0 \\ \downarrow & & \downarrow \\ \mathbf{1} & \longrightarrow & X_1 \end{array}$$

for some set S_1 of 1-cells and some family f_1 of "attaching maps", obtaining the "1-skeleton" X_1. Then we take the pushout

$$\begin{array}{ccc} S_2 \times \mathbb{S}^1 & \xrightarrow{f_2} & X_1 \\ \downarrow & & \downarrow \\ \mathbf{1} & \longrightarrow & X_2 \end{array}$$

for some set S_2 of 2-cells and some family f_2 of attaching maps, obtaining the 2-skeleton X_2, and so on. The fundamental group of each pushout can be calculated from the van Kampen theorem:

we obtain the group presented by generators derived from the 1-skeleton, and relations derived from S_2 and f_2. The pushouts after this stage do not alter the fundamental group, since $\pi_1(\mathbb{S}^n)$ is trivial for $n > 1$ (see §8.3).

Example 8.7.17. In particular, suppose given any presentation of a (set-)group $G = \langle X \mid R \rangle$, with X a set of generators and R a set of words in these generators. Let $B :\equiv \bigvee_X S^1$ and $A :\equiv \bigvee_R S^1$, with $f : A \to B$ sending each copy of S^1 to the corresponding word in the generating loops of B. It follows that $\pi_1(P) \cong G$; thus we have constructed a connected type whose fundamental group is G. Since any group has a presentation, any group is the fundamental group of some type. If we 1-truncate such a type, we obtain a type whose only nontrivial homotopy group is G; this is called an **Eilenberg–Mac Lane space** $K(G, 1)$.

8.8 Whitehead's theorem and Whitehead's principle

In classical homotopy theory, a map $f : A \to B$ which induces an isomorphism $\pi_n(A, a) \cong \pi_n(B, f(a))$ for all points a in A (and also an isomorphism $\pi_0(A) \cong \pi_0(B)$) is necessarily a homotopy equivalence, as long as the spaces A and B are well-behaved (e.g. have the homotopy types of CW-complexes). This is known as *Whitehead's theorem*. In fact, the "ill-behaved" spaces for which Whitehead's theorem fails are invisible to type theory. Roughly, the well-behaved topological spaces suffice to present ∞-groupoids, and homotopy type theory deals with ∞-groupoids directly rather than actual topological spaces. Thus, one might expect that Whitehead's theorem would be true in univalent foundations.

However, this is *not* the case: Whitehead's theorem is not provable. In fact, there are known models of type theory in which it fails to be true, although for entirely different reasons than its failure for ill-behaved topological spaces. These models are "non-hypercomplete ∞-toposes" (see [Lur09]); roughly speaking, they consist of sheaves of ∞-groupoids over ∞-dimensional base spaces.

From a foundational point of view, therefore, we may speak of *Whitehead's principle* as a "classicality axiom", akin to LEM and AC. It may consistently be assumed, but it is not part of the computationally motivated type theory, nor does it hold in all natural models. But when working from set-theoretic foundations, this principle is invisible: it cannot fail to be true in a world where ∞-groupoids are built up out of sets (using topological spaces, simplicial sets, or any other such model).

This may seem odd, but actually it should not be surprising. Homotopy type theory is the *abstract* theory of homotopy types, whereas the homotopy theory of topological spaces or simplicial sets in set theory is a *concrete* model of this theory, in the same way that the integers are a concrete model of the abstract theory of rings. It is to be expected that any concrete model will have special properties which are not intrinsic to the corresponding abstract theory, but which we might sometimes want to assume as additional axioms (e.g. the integers are a Principal Ideal Domain, but not all rings are).

It is beyond the scope of this book to describe any models of type theory, so we will not explain how Whitehead's principle might fail in some of them. However, we can prove that it holds whenever the types involved are n-truncated for some finite n, by "downward" induction on n. In addition to being of interest in its own right (for instance, it implies the essential uniqueness of Eilenberg–Mac Lane spaces), the proof of this result will hopefully provide some intuitive explanation for why we cannot hope to prove an analogous theorem without truncation hypotheses.

We begin with the following modification of Theorem 4.6.3, which will eventually supply the induction step in the proof of the truncated Whitehead's principle. It may be regarded as a type-theoretic, ∞-groupoidal version of the classical statement that a fully faithful and essentially surjective functor is an equivalence of categories.

Theorem 8.8.1. *Suppose $f : A \to B$ is a function such that*

(i) $\|f\|_0 : \|A\|_0 \to \|B\|_0$ is surjective, and
(ii) for any $x, y : A$, the function $\mathsf{ap}_f : (x =_A y) \to (f(x) =_B f(y))$ is an equivalence.

Then f is an equivalence.

Proof. Note that (ii) is precisely the statement that f is an embedding, c.f. §4.6. Thus, by Theorem 4.6.3, it suffices to show that f is surjective, i.e. that for any $b : B$ we have $\left\|\mathsf{fib}_f(b)\right\|_{-1}$. Suppose given b; then since $\|f\|_0$ is surjective, there merely exists an $a : A$ such that $\|f\|_0(|a|_0) = |b|_0$. And since our goal is a mere proposition, we may assume given such an a. Then we have $|f(a)|_0 = \|f\|_0(|a|_0) = |b|_0$, hence $\|f(a) = b\|_{-1}$. Again, since our goal is still a mere proposition, we may assume $f(a) = b$. Hence $\mathsf{fib}_f(b)$ is inhabited, and thus merely inhabited. □

Since homotopy groups are truncations of loop spaces, rather than path spaces, we need to modify this theorem to speak about these instead. Recall the map Ωf from Definition 8.4.2.

Corollary 8.8.2. *Suppose $f : A \to B$ is a function such that*

(i) $\|f\|_0 : \|A\|_0 \to \|B\|_0$ is a bijection, and
(ii) for any $x : A$, the function $\Omega f : \Omega(A, x) \to \Omega(B, f(x))$ is an equivalence.

Then f is an equivalence.

Proof. By Theorem 8.8.1, it suffices to show that $\mathsf{ap}_f : (x =_A y) \to (f(x) =_B f(y))$ is an equivalence for any $x, y : A$. And by Corollary 4.4.6, we may assume $f(x) =_B f(y)$. In particular, $|f(x)|_0 = |f(y)|_0$, so since $\|f\|_0$ is an equivalence, we have $|x|_0 = |y|_0$, hence $|x = y|_{-1}$. Since we are trying to prove a mere proposition (f being an equivalence), we may assume given $p : x = y$. But now the following square commutes up to homotopy:

$$\begin{array}{ccc} \Omega(A, x) & \xrightarrow{- \cdot p} & (x =_A y) \\ {\scriptstyle \Omega f}\downarrow & & \downarrow{\scriptstyle \mathsf{ap}_f} \\ \Omega(B, f(x)) & \xrightarrow[- \cdot f(p)]{} & (f(x) =_B f(y)). \end{array}$$

The top and bottom maps are equivalences, and the left-hand map is so by assumption. Hence, by the 2-out-of-3 property, so is the right-hand map. □

Now we can prove the truncated Whitehead's principle.

Theorem 8.8.3. *Suppose A and B are n-types and $f : A \to B$ is such that*

(i) $\|f\|_0 : \|A\|_0 \to \|B\|_0$ is a bijection, and
(ii) $\pi_k(f) : \pi_k(A, x) \to \pi_k(B, f(x))$ is a bijection for all $k \geq 1$ and all $x : A$.

Then f is an equivalence.

Condition (i) is almost the case of (ii) when $k = 0$, except that it makes no reference to any basepoint $x : A$.

Proof. We proceed by induction on n. When $n = -2$, the statement is trivial. Thus, suppose it to be true for all functions between n-types, and let A and B be $(n+1)$-types and $f : A \to B$ as above. The first condition in Corollary 8.8.2 holds by assumption, so it will suffice to show that for any $x : A$, the function $\Omega f : \Omega(A, x) \to \Omega(B, f(x))$ is an equivalence.

Since $\Omega(A, x)$ and $\Omega(B, f(x))$ are n-types we can apply the induction hypothesis. We need to check that $\|\Omega f\|_0$ is a bijection, and that for all $k \geq 1$ and $p : x = x$ the map $\pi_k(\Omega f) : \pi_k(x = x, p) \to \pi_k(f(x) = f(x), \Omega f(p))$ is a bijection. The first statement holds by assumption, since $\|\Omega f\|_0 \equiv \pi_1(f)$. To prove the second statement, we generalize it first: we show that for all $y : A$ and $q : x = y$ we have $\pi_k(\mathsf{ap}_f) : \pi_k(x = y, q) \to \pi_k(f(x) = f(y), \mathsf{ap}_f(q))$. This implies the desired statement, since when $y :\equiv x$, we have $\pi_k(\Omega f) = \pi_k(\mathsf{ap}_f)$ modulo identifying their base points $\Omega f(p) = \mathsf{ap}_f(p)$. To prove the generalization, it suffices by path induction to prove it when q is refl_a. In this case, we have $\pi_k(\mathsf{ap}_f) = \pi_k(\Omega f) = \pi_{k+1}(f)$, and $\pi_{k+1}(f)$ is an bijection by the original assumptions. □

Note that if A and B are not n-types for any finite n, then there is no way for the induction to get started.

Corollary 8.8.4. *If A is a 0-connected n-type and $\pi_k(A, a) = 0$ for all k and $a : A$, then A is contractible.*

Proof. Apply Theorem 8.8.3 to the map $A \to \mathbf{1}$. □

As an application, we can deduce the converse of Corollary 8.4.8.

Corollary 8.8.5. *For $n \geq 0$, a map $f : A \to B$ is n-connected if and only if the following all hold:*

(i) $\|f\|_0 : \|A\|_0 \to \|B\|_0$ *is an isomorphism.*
(ii) For any $a : A$ and $k \leq n$, the map $\pi_k(f) : \pi_k(A, a) \to \pi_k(B, f(a))$ is an isomorphism.
(iii) For any $a : A$, the map $\pi_{n+1}(f) : \pi_{n+1}(A, a) \to \pi_{n+1}(B, f(a))$ is surjective.

Proof. The "only if" direction is Corollary 8.4.8. Conversely, by the long exact sequence of a fibration (Theorem 8.4.6), the hypotheses imply that $\pi_k(\mathsf{fib}_f(f(a))) = 0$ for all $k \leq n$ and $a : A$, and that $\left\|\mathsf{fib}_f(f(a))\right\|_0$ is contractible. Since $\pi_k(\mathsf{fib}_f(f(a))) = \pi_k(\left\|\mathsf{fib}_f(f(a))\right\|_n)$ for $k \leq n$, and $\left\|\mathsf{fib}_f(f(a))\right\|_n$ is n-connected, by Corollary 8.8.4 it is contractible for any a.

It remains to show that $\left\|\mathsf{fib}_f(b)\right\|_n$ is contractible for $b : B$ not necessarily of the form $f(a)$. However, by assumption, there is $x : \|A\|_0$ with $|b|_0 = \|f\|_0(x)$. Since contractibility is a mere proposition, we may assume x is of the form $|a|_0$ for $a : A$, in which case $|b|_0 = \|f\|_0(|a|_0) = |f(a)|_0$, and therefore $\|b = f(a)\|_{-1}$. Again since contractibility is a mere proposition, we may assume $b = f(a)$, and the result follows. □

A map f such that $\|f\|_0$ is a bijection and $\pi_k(f)$ is a bijection for all k is called **∞-connected** or a **weak equivalence**. This is equivalent to asking that f be n-connected for all n. A type Z is called **∞-truncated** or **hypercomplete** if the induced map

$$(- \circ f) : (B \to Z) \to (A \to Z)$$

is an equivalence whenever f is ∞-connected — that is, if Z thinks every ∞-connected map is an equivalence. Then if we want to assume Whitehead's principle as an axiom, we may use either of the following equivalent forms.

- Every ∞-connected function is an equivalence.
- Every type is ∞-truncated.

In higher topos models, the ∞-truncated types form a reflective subuniverse in the sense of §7.7 (the "hypercompletion" of an $(\infty,1)$-topos), but we do not know whether this is true in general.

It may not be obvious that there *are* any types which are not n-types for any n, but in fact there are. Indeed, in classical homotopy theory, $\mathbb{S}^n$ has this property for any $n \geq 2$. We have not proven this fact in homotopy type theory yet, but there are other types which we can prove to have "infinite truncation level".

Example 8.8.6. Suppose we have $B : \mathbb{N} \to \mathcal{U}$ such that for each n, the type $B(n)$ contains an n-loop which is not equal to n-fold reflexivity, say $p_n : \Omega^n(B(n), b_n)$ with $p_n \neq \mathsf{refl}^n_{b_n}$. (For instance, we could define $B(n) :\equiv \mathbb{S}^n$, with p_n the image of $1 : \mathbb{Z}$ under the isomorphism $\pi_n(\mathbb{S}^n) \cong \mathbb{Z}$.) Consider $C :\equiv \prod_{(n:\mathbb{N})} B(n)$, with the point $c : C$ defined by $c(n) :\equiv b_n$. Since loop spaces commute with products, for any m we have

$$\Omega^m(C,c) \;\simeq\; \prod_{n:\mathbb{N}} \Omega^m(B(n), b_n).$$

Under this equivalence, refl^m_c corresponds to the function $(n \mapsto \mathsf{refl}^m_{b_n})$. Now define q_m in the right-hand type by

$$q_m(n) :\equiv \begin{cases} p_n & m = n \\ \mathsf{refl}^m_{b_n} & m \neq n. \end{cases}$$

If we had $q_m = (n \mapsto \mathsf{refl}^m_{b_n})$, then we would have $p_n = \mathsf{refl}^n_{b_n}$, which is not the case. Thus, $q_m \neq (n \mapsto \mathsf{refl}^m_{b_n})$, and so there is a point of $\Omega^m(C,c)$ which is unequal to refl^m_c. Hence C is not an m-type, for any $m : \mathbb{N}$.

We expect it should also be possible to show that a universe $\mathcal{U}$ itself is not an n-type for any n, using the fact that it contains higher inductive types such as $\mathbb{S}^n$ for all n. However, this has not yet been done.

8.9 A general statement of the encode-decode method

We have used the encode-decode method to characterize the path spaces of various types, including coproducts (Theorem 2.12.5), natural numbers (Theorem 2.13.1), truncations (Theorem 7.3.12), the circle (Corollary 8.1.10), suspensions (Theorem 8.6.4), and pushouts (Theorem 8.7.12). Variants of this technique are used in the proofs of many of the other theorems mentioned in the introduction to this chapter, such as a direct proof of $\pi_n(\mathbb{S}^n)$, the Blakers–Massey theorem, and the construction of Eilenberg–Mac Lane spaces. While it is tempting to try to abstract the method into a lemma, this is difficult because slightly different variants are needed for different problems. For example, different variations on the same method can be used to characterize a loop space (as in Theorem 2.12.5 and Corollary 8.1.10) or a whole path space (as in Theorem 2.13.1), to give a complete characterization of a loop space (e.g. $\Omega^1(\mathbb{S}^1)$) or only to characterize some truncation of it (e.g. van Kampen), and to calculate homotopy groups or to prove that a map is n-connected (e.g. Freudenthal and Blakers–Massey).

However, we can state lemmas for specific variants of the method. The proofs of these lemmas are almost trivial; the main point is to clarify the method by stating them in generality. The simplest case is using an encode-decode method to characterize the loop space of a type, as in Theorem 2.12.5 and Corollary 8.1.10.

Lemma 8.9.1 (Encode-decode for Loop Spaces)**.** *Given a pointed type* (A, a_0) *and a fibration* $\mathsf{code} : A \to \mathcal{U}$*, if*

(i) $c_0 : \mathsf{code}(a_0)$,
(ii) $\mathsf{decode} : \prod_{(x:A)} \mathsf{code}(x) \to (a_0 = x)$,
(iii) for all $c : \mathsf{code}(a_0)$, $\mathsf{transport}^{\mathsf{code}}(\mathsf{decode}(c), c_0) = c$*, and*
(iv) $\mathsf{decode}(c_0) = \mathsf{refl}$,

then $(a_0 = a_0)$ *is equivalent to* $\mathsf{code}(a_0)$.

Proof. Define $\mathsf{encode} : \prod_{(x:A)}(a_0 = x) \to \mathsf{code}(x)$ by

$$\mathsf{encode}_x(\alpha) = \mathsf{transport}^{\mathsf{code}}(\alpha, c_0).$$

We show that encode_{a_0} and decode_{a_0} are quasi-inverses. The composition $\mathsf{encode}_{a_0} \circ \mathsf{decode}_{a_0}$ is immediate by assumption (iii). For the other composition, we show

$$\prod_{(x:A)} \prod_{(p:a_0=x)} \mathsf{decode}_x(\mathsf{encode}_x p) = p.$$

By path induction, it suffices to show $\mathsf{decode}_{a_0}(\mathsf{encode}_{a_o}\mathsf{refl}) = \mathsf{refl}$. After reducing the transport, it suffices to show $\mathsf{decode}_{a_0}(c_0) = \mathsf{refl}$, which is assumption (iv). □

If a fiberwise equivalence between $(a_0 = -)$ and code is desired, it suffices to strengthen condition (iii) to

$$\prod_{(x:A)} \prod_{(c:\mathsf{code}(x))} \mathsf{encode}_x(\mathsf{decode}_x(c)) = c.$$

However, to calculate a loop space (e.g. $\Omega(\mathbb{S}^1)$), this stronger assumption is not necessary.

Another variation, which comes up often when calculating homotopy groups, characterizes the truncation of a loop space:

Lemma 8.9.2 (Encode-decode for Truncations of Loop Spaces)**.** *Assume a pointed type* (A, a_0) *and a fibration* $\mathsf{code} : A \to \mathcal{U}$*, where for every* x*,* $\mathsf{code}(x)$ *is a k-type. Define*

$$\mathsf{encode} : \prod_{x:A} \|a_0 = x\|_k \to \mathsf{code}(x)$$

by truncation recursion (using the fact that $\mathsf{code}(x)$ *is a k-type), mapping* $\alpha : a_0 = x$ *to* $\mathsf{transport}^{\mathsf{code}}(\alpha, c_0)$*. Suppose:*

(i) $c_0 : \mathsf{code}(a_0)$,
(ii) $\mathsf{decode} : \prod_{(x:A)} \mathsf{code}(x) \to \|a_0 = x\|_k$,
(iii) $\mathsf{encode}_{a_0}(\mathsf{decode}_{a0}(c)) = c$ *for all* $c : \mathsf{code}(a_0)$*, and*
(iv) $\mathsf{decode}(c_0) = |\mathsf{refl}|$.

Then $\|a_0 = a_0\|_k$ *is equivalent to* $\mathsf{code}(a_0)$.

Proof. That decode ∘ encode is identity is immediate by (iii). To prove encode ∘ decode, we first do a truncation induction, by which it suffices to show

$$\prod_{(x:A)} \prod_{(p:a_0=x)} \mathsf{decode}_x(\mathsf{encode}_x(|p|_k)) = |p|_k.$$

The truncation induction is allowed because paths in a k-type are a k-type. To show this type, we do a path induction, and after reducing the encode, use assumption (iv). □

8.10 Additional Results

Though we do not present the proofs in this chapter, following results have also been established in homotopy type theory.

Theorem 8.10.1. *There exists a k such that for all $n \geq 3$, $\pi_{n+1}(\mathbb{S}^n) = \mathbb{Z}_k$.*

Notes on the proof. The proof consists of a calculation of $\pi_4(\mathbb{S}^3)$, together with an appeal to stability (Corollary 8.6.15). In the classical statement of this result, k is 2. While we have not yet checked that k is in fact 2, our calculation of $\pi_4(\mathbb{S}^3)$ is constructive, like all the rest of the proofs in this chapter. (More precisely, it doesn't use any additional axioms such as LEM or AC, making it as constructive as univalence and higher inductive types are.) Thus, given a computational interpretation of homotopy type theory, we could run the proof on a computer to verify that k is 2. This example is quite intriguing, because it is the first calculation of a homotopy group for which we have not needed to know the answer in advance. □

Theorem 8.10.2 (Blakers–Massey theorem)**.** *Suppose we are given maps $f : C \to X$, and $g : C \to Y$. Taking first the pushout $X \sqcup^C Y$ of f and g and then the pullback of its inclusions $\mathsf{inl} : X \to X \sqcup^C Y \leftarrow Y : \mathsf{inr}$, we have an induced map $C \to X \times_{(X \sqcup^C Y)} Y$.*

If f is i-connected and g is j-connected, then this induced map is $(i + j)$-connected. In other words, for any points $x : X$, $y : Y$, the corresponding fiber $C_{x,y}$ of $(f, g) : C \to X \times Y$ gives an approximation to the path space $\mathsf{inl}(x) =_{X \sqcup^C Y} \mathsf{inr}(y)$ in the pushout.

It should be noted that in classical algebraic topology, the Blakers–Massey theorem is often stated in a somewhat different form, where the maps f and g are replaced by inclusions of subcomplexes of CW complexes, and the homotopy pushout and homotopy pullback by a union and intersection, respectively. In order to express the theorem in homotopy type theory, we have to replace notions of this sort with ones that are homotopy-invariant. We have seen another example of this in the van Kampen theorem (§8.7), where we had to replace a union of open subsets by a homotopy pushout.

Theorem 8.10.3 (Eilenberg–Mac Lane Spaces)**.** *For any abelian group G and positive integer n, there is an n-type $K(G, n)$ such that $\pi_n(K(G, n)) = G$, and $\pi_k(K(G, n)) = 0$ for $k \neq n$.*

Theorem 8.10.4 (Covering spaces)**.** *For a connected space A, there is an equivalence between covering spaces over A and sets with an action of $\pi_1(A)$.*

Notes

The theorems described in this chapter are standard results in classical homotopy theory; many are described by [Hat02]. In these notes, we review the development of the new synthetic proofs of them in homotopy type theory. Table 8.2 lists the homotopy-theoretic theorems that have been proven in homotopy type theory, and whether they have been computer-checked. Almost all of these results were developed during the spring term at IAS in 2013, as part of a significant collaborative effort. Many people contributed to these results, for example by being the principal author of a proof, by suggesting problems to work on, by participating in many discussions and seminars about these problems, or by giving feedback on results. The following people were the principal authors of the first homotopy type theory proofs of the above theorems. Unless indicated otherwise, for the theorems that have been computer-checked, the principal authors were also the first ones to formalize the proof using a computer proof assistant.

Theorem	Status
$\pi_1(\mathbb{S}^1)$	✔✔
$\pi_{k<n}(\mathbb{S}^n)$	✔✔
long-exact-sequence of homotopy groups	✔✔
total space of Hopf fibration is $\mathbb{S}^3$	✔
$\pi_2(\mathbb{S}^2)$	✔✔
$\pi_3(\mathbb{S}^2)$	✔
$\pi_n(\mathbb{S}^n)$	✔✔
$\pi_4(\mathbb{S}^3)$	✔
Freudenthal suspension theorem	✔✔
Blakers–Massey theorem	✔✔
Eilenberg–Mac Lane spaces $K(G, n)$	✔✔
van Kampen theorem	✔✔
covering spaces	✔✔
Whitehead's principle for n-types	✔✔

Table 8.2: Theorems from homotopy theory proved by hand (✔) and by computer (✔✔).

- Shulman gave the homotopy-theoretic calculation of $\pi_1(\mathbb{S}^1)$. Licata later discovered the encode-decode proof and the encode-decode method.
- Brunerie calculated $\pi_{k<n}(\mathbb{S}^n)$. Licata later gave an encode-decode version.
- Voevodsky constructed the long exact sequence of homotopy groups.
- Lumsdaine constructed the Hopf fibration. Brunerie proved that its total space is $\mathbb{S}^3$, thereby calculating $\pi_2(\mathbb{S}^2)$ and $\pi_3(\mathbb{S}^3)$.
- Licata and Brunerie gave a direct calculation of $\pi_n(\mathbb{S}^n)$.
- Lumsdaine proved the Freudenthal suspension theorem; Licata and Lumsdaine formalized this proof.
- Lumsdaine, Finster, and Licata proved the Blakers–Massey theorem; Lumsdaine, Brunerie, Licata, and Hou formalized it.
- Licata gave an encode-decode calculation of $\pi_2(\mathbb{S}^2)$, and a calculation of $\pi_n(\mathbb{S}^n)$ using the Freudenthal suspension theorem; using similar techniques, he constructed $K(G, n)$.
- Shulman proved the van Kampen theorem; Hou formalized this proof.
- Licata proved Whitehead's theorem for n-types.
- Brunerie calculated $\pi_4(\mathbb{S}^3)$.
- Hou established the theory of covering spaces and formalized it.

The interplay between homotopy theory and type theory was crucial to the development of these results. For example, the first proof that $\pi_1(\mathbb{S}^1) = \mathbb{Z}$ was the one given in §8.1.5, which follows a classical homotopy theoretic one. A type-theoretic analysis of this proof resulted in the development of the encode-decode method. The first calculation of $\pi_2(\mathbb{S}^2)$ also followed classical methods, but this led quickly to an encode-decode proof of the result. The encode-decode calculation generalized to $\pi_n(\mathbb{S}^n)$, which in turn led to the proof of the Freudenthal suspension theorem, by combining an encode-decode argument with classical homotopy-theoretic reasoning about connectedness, which in turn led to the Blakers–Massey theorem and Eilenberg–Mac

Lane spaces. The rapid development of this series of results illustrates the promise of our new understanding of the connections between these two subjects.

Exercises

Exercise 8.1. Prove that homotopy groups respect products: $\pi_n(A \times B) \simeq \pi_n(A) \times \pi_n(B)$.

Exercise 8.2. Prove that if A is a set with decidable equality (see Definition 3.4.3), then its suspension ΣA is a 1-type. (It is an open question whether this is provable without the assumption of decidable equality.)

Exercise 8.3. Define $\mathbb{S}^\infty$ to be the colimit of the sequence $\mathbb{S}^0 \to \mathbb{S}^1 \to \mathbb{S}^2 \to \cdots$. Prove that $\mathbb{S}^\infty$ is contractible.

Exercise 8.4. Define $\mathbb{S}^\infty$ to be the higher inductive type generated by

- Two points $\mathsf{N} : \mathbb{S}^\infty$ and $\mathsf{S} : \mathbb{S}^\infty$, and
- For each $x : \mathbb{S}^\infty$, a path $\mathsf{merid}(x) : \mathsf{N} = \mathsf{S}$.

In other words, $\mathbb{S}^\infty$ is its own suspension. Prove that $\mathbb{S}^\infty$ is contractible.

Exercise 8.5. Suppose $f : X \to Y$ is a function and Y is connected. Show that for any $y_1, y_2 : Y$ we have $\left\|\mathsf{fib}_f(y_1) \simeq \mathsf{fib}_f(y_2)\right\|$.

Exercise 8.6. For any pointed type A, let $i_A : \Omega A \to \Omega A$ denote inversion of loops, $i_A :\equiv \lambda p.\, p^{-1}$. Show that $i_{\Omega A} : \Omega^2 A \to \Omega^2 A$ is equal to $\Omega(i_A)$.

Exercise 8.7. Define a **pointed equivalence** to be a pointed map whose underlying function is an equivalence.

(i) Show that the type of pointed equivalences between pointed types (X, x_0) and (Y, y_0) is equivalent to $(X, x_0) =_{\mathcal{U}_\bullet} (Y, y_0)$.
(ii) Reformulate the notion of pointed equivalence in terms of a pointed quasi-inverse and pointed homotopies, in one of the coherent styles from Chapter 4.

Exercise 8.8. Following the example of the Hopf fibration in §8.5, define the **junior Hopf fibration** as a fibration (that is, a type family) over $\mathbb{S}^1$ whose fiber over the basepoint is $\mathbb{S}^0$ and whose total space is $\mathbb{S}^1$. This is also called the "twisted double cover" of the circle $\mathbb{S}^1$.

Exercise 8.9. Again following the example of the Hopf fibration in §8.5, define an analogous fibration over $\mathbb{S}^4$ whose fiber over the basepoint is $\mathbb{S}^3$ and whose total space is $\mathbb{S}^7$. This is an open problem in homotopy type theory (such a fibration is known to exist in classical homotopy theory).

Exercise 8.10. Continuing from Example 8.7.7, prove that if A has a point $a : A$, then we can identify $\pi_1(\Sigma A)$ with the group presented by $\|A\|_0$ as generators with the relation $|a|_0 = e$. Then show that if we assume excluded middle, this is also the free group on $\|A\|_0 \setminus \{|a|_0\}$.

Exercise 8.11. Again continuing from Example 8.7.7, but this time without assuming A to be pointed, show that we can identify $\pi_1(\Sigma A)$ with the group presented by generators $\|A\|_0 \times \|A\|_0$ and relations

$$(a, b) = (b, a)^{-1}, \qquad (a, c) = (a, b) \cdot (b, c), \qquad \text{and} \qquad (a, a) = e.$$

Chapter 9

Category theory

Of the branches of mathematics, category theory is one which perhaps fits the least comfortably in set theoretic foundations. One problem is that most of category theory is invariant under weaker notions of "sameness" than equality, such as isomorphism in a category or equivalence of categories, in a way which set theory fails to capture. But this is the same sort of problem that the univalence axiom solves for types, by identifying equality with equivalence. Thus, in univalent foundations it makes sense to consider a notion of "category" in which equality of objects is identified with isomorphism in a similar way.

Ignoring size issues, in set-based mathematics a category consists of a *set* A_0 of objects and, for each $x, y \in A_0$, a *set* $\hom_A(x, y)$ of morphisms. Under univalent foundations, a "naive" definition of category would simply mimic this with a *type* of objects and *types* of morphisms. If we allowed these types to contain arbitrary higher homotopy, then we ought to impose higher coherence conditions, leading to some notion of $(\infty, 1)$-category, but at present our goal is more modest. We consider only 1-categories, and therefore we restrict the types $\hom_A(x, y)$ to be sets, i.e. 0-types. If we impose no further conditions, we will call this notion a *precategory*.

If we add the requirement that the type A_0 of objects is a set, then we end up with a definition that behaves much like the traditional set-theoretic one. Following Toby Bartels, we call this notion a *strict category*. Alternatively, we can require a generalized version of the univalence axiom, identifying $(x =_{A_0} y)$ with the type $\mathsf{iso}(x, y)$ of isomorphisms from x to y. Since we regard the latter choice as usually the "correct" definition, we will call it simply a *category*.

A good example of the difference between the three notions of category is provided by the statement "every fully faithful and essentially surjective functor is an equivalence of categories", which in classical set-based category theory is equivalent to the axiom of choice.

(i) For strict categories, this is still equivalent to the axiom of choice.

(ii) For precategories, there is no consistent axiom of choice which can make it true.

(iii) For categories, it is provable *without* any axiom of choice.

We will prove the latter statement in this chapter, as well as other pleasant properties of categories, e.g. that equivalent categories are equal (as elements of the type of categories). We will also describe a universal way of "saturating" a precategory A into a category $\widehat{A}$, which we call its *Rezk completion*, although it could also reasonably be called the *stack completion* (see the Notes).

The Rezk completion also sheds further light on the notion of equivalence of categories. For instance, the functor $A \to \widehat{A}$ is always fully faithful and essentially surjective, hence a "weak

equivalence". It follows that a precategory is a category exactly when it "sees" all fully faithful and essentially surjective functors as equivalences; thus our notion of "category" is already inherent in the notion of "fully faithful and essentially surjective functor".

We assume the reader has some basic familiarity with classical category theory. Recall that whenever we write $\mathcal{U}$ it denotes some universe of types, but perhaps a different one at different times; everything we say remains true for any consistent choice of universe levels. We will use the basic notions of homotopy type theory from Chapters 1 and 2 and the propositional truncation from Chapter 3, but not much else from Part I, except that our second construction of the Rezk completion will use a higher inductive type.

9.1 Categories and precategories

In classical mathematics, there are many equivalent definitions of a category. In our case, since we have dependent types, it is natural to choose the arrows to be a type family indexed by the objects. This matches the way hom-types are always used in category theory: we never even consider comparing two arrows unless we know their domains and codomains agree. Furthermore, it seems clear that for a theory of 1-categories, the hom-types should all be sets. This leads us to the following definition.

Definition 9.1.1. A **precategory** A consists of the following.

(i) A type A_0, whose elements are called **objects**. We write $a : A$ for $a : A_0$.
(ii) For each $a, b : A$, a set $\hom_A(a, b)$, whose elements are called **arrows** or **morphisms**.
(iii) For each $a : A$, a morphism $1_a : \hom_A(a, a)$, called the **identity morphism**.
(iv) For each $a, b, c : A$, a function

$$\hom_A(b, c) \to \hom_A(a, b) \to \hom_A(a, c)$$

called **composition**, and denoted infix by $g \mapsto f \mapsto g \circ f$, or sometimes simply by gf.
(v) For each $a, b : A$ and $f : \hom_A(a, b)$, we have $f = 1_b \circ f$ and $f = f \circ 1_a$.
(vi) For each $a, b, c, d : A$ and

$$f : \hom_A(a, b), \qquad g : \hom_A(b, c), \qquad h : \hom_A(c, d),$$

we have $h \circ (g \circ f) = (h \circ g) \circ f$.

The problem with the notion of precategory is that for objects $a, b : A$, we have two possibly-different notions of "sameness". On the one hand, we have the type $(a =_{A_0} b)$. But on the other hand, there is the standard categorical notion of *isomorphism*.

Definition 9.1.2. A morphism $f : \hom_A(a, b)$ is an **isomorphism** if there is a morphism $g : \hom_A(b, a)$ such that $g \circ f = 1_a$ and $f \circ g = 1_b$. We write $a \cong b$ for the type of such isomorphisms.

Lemma 9.1.3. *For any $f : \hom_A(a, b)$, the type "f is an isomorphism" is a mere proposition. Therefore, for any $a, b : A$ the type $a \cong b$ is a set.*

Proof. Suppose given $g : \hom_A(b, a)$ and $\eta : (1_a = g \circ f)$ and $\epsilon : (f \circ g = 1_b)$, and similarly g', η', and ϵ'. We must show $(g, \eta, \epsilon) = (g', \eta', \epsilon')$. But since all hom-sets are sets, their identity types are mere propositions, so it suffices to show $g = g'$. For this we have

$$g' = 1_a \circ g' = (g \circ f) \circ g' = g \circ (f \circ g') = g \circ 1_b = g$$

using η and ϵ'. □

If $f : a \cong b$, then we write f^{-1} for its inverse, which by Lemma 9.1.3 is uniquely determined.

The only relationship between these two notions of sameness that we have in a precategory is the following.

Lemma 9.1.4 (idtoiso). *If A is a precategory and $a, b : A$, then*

$$(a = b) \to (a \cong b).$$

Proof. By induction on identity, we may assume a and b are the same. But then we have $1_a : \hom_A(a, a)$, which is clearly an isomorphism. □

Evidently, this situation is analogous to the issue that motivated us to introduce the univalence axiom. In fact, we have the following:

Example 9.1.5. There is a precategory $\mathcal{S}et$, whose type of objects is Set, and with $\hom_{\mathcal{S}et}(A, B) :\equiv (A \to B)$. The identity morphisms are identity functions and the composition is function composition. For this precategory, Lemma 9.1.4 is equal to (the restriction to sets of) the map idtoeqv from §2.10.

Of course, to be more precise we should call this category $\mathcal{S}et_{\mathcal{U}}$, since its objects are only the *small sets* relative to a universe $\mathcal{U}$.

Thus, it is natural to make the following definition.

Definition 9.1.6. A **category** is a precategory such that for all $a, b : A$, the function $\mathsf{idtoiso}_{a,b}$ from Lemma 9.1.4 is an equivalence.

In particular, in a category, if $a \cong b$, then $a = b$.

Example 9.1.7. The univalence axiom implies immediately that $\mathcal{S}et$ is a category. One can also show, using univalence, that any precategory of set-level structures such as groups, rings, topological spaces, etc. is a category; see §9.8.

We also note the following.

Lemma 9.1.8. *In a category, the type of objects is a 1-type.*

Proof. It suffices to show that for any $a, b : A$, the type $a = b$ is a set. But $a = b$ is equivalent to $a \cong b$, which is a set. □

We write isotoid for the inverse $(a \cong b) \to (a = b)$ of the map idtoiso from Lemma 9.1.4. The following relationship between the two is important.

Lemma 9.1.9. *For $p : a = a'$ and $q : b = b'$ and $f : \hom_A(a, b)$, we have*

$$(p, q)_*(f) = \mathsf{idtoiso}(q) \circ f \circ \mathsf{idtoiso}(p)^{-1}. \tag{9.1.10}$$

Proof. By induction, we may assume p and q are refl_a and refl_b respectively. Then the left-hand side of (9.1.10) is simply f. But by definition, $\mathsf{idtoiso}(\mathsf{refl}_a)$ is 1_a, and $\mathsf{idtoiso}(\mathsf{refl}_b)$ is 1_b, so the right-hand side of (9.1.10) is $1_b \circ f \circ 1_a$, which is equal to f. □

Similarly, we can show

$$\mathsf{idtoiso}(p^{-1}) = (\mathsf{idtoiso}(p))^{-1} \tag{9.1.11}$$

$$\mathsf{idtoiso}(p \cdot q) = \mathsf{idtoiso}(q) \circ \mathsf{idtoiso}(p) \tag{9.1.12}$$

$$\mathsf{isotoid}(f \circ e) = \mathsf{isotoid}(e) \cdot \mathsf{isotoid}(f) \tag{9.1.13}$$

and so on.

Example 9.1.14. A precategory in which each set $\hom_A(a,b)$ is a mere proposition is equivalently a type A_0 equipped with a mere relation "$\leq$" that is reflexive ($a \leq a$) and transitive (if $a \leq b$ and $b \leq c$, then $a \leq c$). We call this a **preorder**.

In a preorder, a witness $f : a \leq b$ is an isomorphism just when there exists some witness $g : b \leq a$. Thus, $a \cong b$ is the mere proposition that $a \leq b$ and $b \leq a$. Therefore, a preorder A is a category just when (1) each type $a = b$ is a mere proposition, and (2) for any $a, b : A_0$ there exists a function $(a \cong b) \to (a = b)$. In other words, A_0 must be a set, and $\leq$ must be antisymmetric (if $a \leq b$ and $b \leq a$, then $a = b$). We call this a **(partial) order** or a **poset**.

Example 9.1.15. If A is a category, then A_0 is a set if and only if for any $a, b : A_0$, the type $a \cong b$ is a mere proposition. This is equivalent to saying that every isomorphism in A is an identity; thus it is rather stronger than the classical notion of "skeletal" category. Categories of this sort are sometimes called **gaunt** [BSP11]. There is not really any notion of "skeletality" for our categories, unless one considers Definition 9.1.6 itself to be such.

Example 9.1.16. For any 1-type X, there is a category with X as its type of objects and with $\hom(x,y) :\equiv (x = y)$. If X is a set, we call this the **discrete** category on X. In general, we call it a **groupoid** (see Exercise 9.6).

Example 9.1.17. For *any* type X, there is a precategory with X as its type of objects and with $\hom(x,y) :\equiv \|x = y\|_0$. The composition operation

$$\|y = z\|_0 \to \|x = y\|_0 \to \|x = z\|_0$$

is defined by induction on truncation from concatenation $(y = z) \to (x = y) \to (x = z)$. We call this the **fundamental pregroupoid** of X. (In fact, we have met it already in §8.7; see also Exercise 9.11.)

Example 9.1.18. There is a precategory whose type of objects is $\mathcal{U}$ and with $\hom(X,Y) :\equiv \|X \to Y\|_0$, and composition defined by induction on truncation from ordinary composition $(Y \to Z) \to (X \to Y) \to (X \to Z)$. We call this the **homotopy precategory of types**.

Example 9.1.19. Let $\mathcal{R}el$ be the following precategory:

- Its objects are sets.
- $\hom_{\mathcal{R}el}(X,Y) = X \to Y \to \mathsf{Prop}$.
- For a set X, we have $1_X(x,x') :\equiv (x = x')$.
- For $R : \hom_{\mathcal{R}el}(X,Y)$ and $S : \hom_{\mathcal{R}el}(Y,Z)$, their composite is defined by

$$(S \circ R)(x,z) :\equiv \left\| \sum_{y:Y} R(x,y) \times S(y,z) \right\|.$$

Suppose $R : \hom_{\mathcal{R}el}(X,Y)$ is an isomorphism, with inverse S. We observe the following.

(i) If $R(x,y)$ and $S(y',x)$, then $(R \circ S)(y',y)$, and hence $y' = y$. Similarly, if $R(x,y)$ and $S(y,x')$, then $x = x'$.

(ii) For any x, we have $x = x$, hence $(S \circ R)(x,x)$. Thus, there merely exists a $y : Y$ such that $R(x,y)$ and $S(y,x)$.

(iii) Suppose $R(x,y)$. By (ii), there merely exists a y' with $R(x,y')$ and $S(y',x)$. But then by (i), merely $y' = y$, and hence $y' = y$ since Y is a set. Therefore, by transporting $S(y',x)$ along this equality, we have $S(y,x)$. In conclusion, $R(x,y) \to S(y,x)$. Similarly, $S(y,x) \to R(x,y)$.

(iv) If $R(x,y)$ and $R(x,y')$, then by (iii), $S(y',x)$, so that by (i), $y = y'$. Thus, for any x there is at most one y such that $R(x,y)$. And by (ii), there merely exists such a y, hence there exists such a y.

In conclusion, if $R : \hom_{\mathcal{R}el}(X,Y)$ is an isomorphism, then for each $x : X$ there is exactly one $y : Y$ such that $R(x,y)$, and dually. Thus, there is a function $f : X \to Y$ sending each x to this y, which is an equivalence; hence $X = Y$. With a little more work, we conclude that $\mathcal{R}el$ is a category.

We might now restrict ourselves to considering categories rather than precategories. Instead, we will develop many concepts for precategories as well as categories, in order to emphasize how much better-behaved categories are, as compared both to precategories and to ordinary categories in classical mathematics.

We will also see in §§9.6–9.7 that in slightly more exotic contexts, there are uses for certain kinds of precategories other than categories, each of which "fixes" the equality of objects in different ways. This emphasizes the "pre"-ness of precategories: they are the raw material out of which multiple important categorical structures can be defined.

9.2 Functors and transformations

The following definitions are fairly obvious, and need no modification.

Definition 9.2.1. Let A and B be precategories. A **functor** $F : A \to B$ consists of

(i) A function $F_0 : A_0 \to B_0$, generally also denoted F.
(ii) For each $a, b : A$, a function $F_{a,b} : \hom_A(a,b) \to \hom_B(Fa, Fb)$, generally also denoted F.
(iii) For each $a : A$, we have $F(1_a) = 1_{Fa}$.
(iv) For each $a, b, c : A$ and $f : \hom_A(a,b)$ and $g : \hom_A(b,c)$, we have

$$F(g \circ f) = Fg \circ Ff.$$

Note that by induction on identity, a functor also preserves idtoiso.

Definition 9.2.2. For functors $F, G : A \to B$, a **natural transformation** $\gamma : F \to G$ consists of

(i) For each $a : A$, a morphism $\gamma_a : \hom_B(Fa, Ga)$ (the "components").
(ii) For each $a, b : A$ and $f : \hom_A(a,b)$, we have $Gf \circ \gamma_a = \gamma_b \circ Ff$ (the "naturality axiom").

Since each type $\hom_B(Fa, Gb)$ is a set, its identity type is a mere proposition. Thus, the naturality axiom is a mere proposition, so identity of natural transformations is determined by identity of their components. In particular, for any F and G, the type of natural transformations from F to G is again a set.

Similarly, identity of functors is determined by identity of the functions $A_0 \to B_0$ and (transported along this) of the corresponding functions on hom-sets.

Definition 9.2.3. For precategories A, B, there is a precategory B^A, called the **functor precategory**, defined by

- $(B^A)_0$ is the type of functors from A to B.

- $\hom_{B^A}(F, G)$ is the type of natural transformations from F to G.

Proof. We define $(1_F)_a :\equiv 1_{Fa}$. Naturality follows by the unit axioms of a precategory. For $\gamma : F \to G$ and $\delta : G \to H$, we define $(\delta \circ \gamma)_a :\equiv \delta_a \circ \gamma_a$. Naturality follows by associativity. Similarly, the unit and associativity laws for B^A follow from those for B. □

Lemma 9.2.4. *A natural transformation $\gamma : F \to G$ is an isomorphism in B^A if and only if each γ_a is an isomorphism in B.*

Proof. If γ is an isomorphism, then we have $\delta : G \to F$ that is its inverse. By definition of composition in B^A, $(\delta\gamma)_a \equiv \delta_a\gamma_a$ and similarly. Thus, $\delta\gamma = 1_F$ and $\gamma\delta = 1_G$ imply $\delta_a\gamma_a = 1_{Fa}$ and $\gamma_a\delta_a = 1_{Ga}$, so γ_a is an isomorphism.

Conversely, suppose each γ_a is an isomorphism, with inverse called δ_a, say. We define a natural transformation $\delta : G \to F$ with components δ_a; for the naturality axiom we have

$$Ff \circ \delta_a = \delta_b \circ \gamma_b \circ Ff \circ \delta_a = \delta_b \circ Gf \circ \gamma_a \circ \delta_a = \delta_b \circ Gf.$$

Now since composition and identity of natural transformations is determined on their components, we have $\gamma\delta = 1_G$ and $\delta\gamma = 1_F$. □

The following result is fundamental.

Theorem 9.2.5. *If A is a precategory and B is a category, then B^A is a category.*

Proof. Let $F, G : A \to B$; we must show that $\mathsf{idtoiso} : (F = G) \to (F \cong G)$ is an equivalence.

To give an inverse to it, suppose $\gamma : F \cong G$ is a natural isomorphism. Then for any $a : A$, we have an isomorphism $\gamma_a : Fa \cong Ga$, hence an identity $\mathsf{isotoid}(\gamma_a) : Fa = Ga$. By function extensionality, we have an identity $\bar{\gamma} : F_0 =_{(A_0 \to B_0)} G_0$.

Now since the last two axioms of a functor are mere propositions, to show that $F = G$ it will suffice to show that for any $a, b : A$, the functions

$$F_{a,b} : \hom_A(a, b) \to \hom_B(Fa, Fb) \qquad \text{and}$$
$$G_{a,b} : \hom_A(a, b) \to \hom_B(Ga, Gb)$$

become equal when transported along $\bar{\gamma}$. By computation for function extensionality, when applied to a, $\bar{\gamma}$ becomes equal to $\mathsf{isotoid}(\gamma_a)$. But by Lemma 9.1.9, transporting $Ff : \hom_B(Fa, Fb)$ along $\mathsf{isotoid}(\gamma_a)$ and $\mathsf{isotoid}(\gamma_b)$ is equal to the composite $\gamma_b \circ Ff \circ (\gamma_a)^{-1}$, which by naturality of γ is equal to Gf.

This completes the definition of a function $(F \cong G) \to (F = G)$. Now consider the composite

$$(F = G) \to (F \cong G) \to (F = G).$$

Since hom-sets are sets, their identity types are mere propositions, so to show that two identities $p, q : F = G$ are equal, it suffices to show that $p =_{F_0 = G_0} q$. But in the definition of $\bar{\gamma}$, if γ were of the form $\mathsf{idtoiso}(p)$, then γ_a would be equal to $\mathsf{idtoiso}(p_a)$ (this can easily be proved by induction on p). Thus, $\mathsf{isotoid}(\gamma_a)$ would be equal to p_a, and so by function extensionality we would have $\bar{\gamma} = p$, which is what we need.

Finally, consider the composite

$$(F \cong G) \to (F = G) \to (F \cong G).$$

Since identity of natural transformations can be tested componentwise, it suffices to show that for each a we have $\mathsf{idtoiso}(\bar{\gamma})_a = \gamma_a$. But as observed above, we have $\mathsf{idtoiso}(\bar{\gamma})_a = \mathsf{idtoiso}((\bar{\gamma})_a)$, while $(\bar{\gamma})_a = \mathsf{isotoid}(\gamma_a)$ by computation for function extensionality. Since isotoid and idtoiso are inverses, we have $\mathsf{idtoiso}(\bar{\gamma})_a = \gamma_a$ as desired. □

In particular, naturally isomorphic functors between categories (as opposed to precategories) are equal.

We now define all the usual ways to compose functors and natural transformations.

Definition 9.2.6. For functors $F : A \to B$ and $G : B \to C$, their composite $G \circ F : A \to C$ is given by

- The composite $(G_0 \circ F_0) : A_0 \to C_0$
- For each $a, b : A$, the composite

$$(G_{Fa,Fb} \circ F_{a,b}) : \hom_A(a, b) \to \hom_C(GFa, GFb).$$

It is easy to check the axioms.

Definition 9.2.7. For functors $F : A \to B$ and $G, H : B \to C$ and a natural transformation $\gamma : G \to H$, the composite $(\gamma F) : GF \to HF$ is given by

- For each $a : A$, the component γ_{Fa}.

Naturality is easy to check. Similarly, for γ as above and $K : C \to D$, the composite $(K\gamma) : KG \to KH$ is given by

- For each $b : B$, the component $K(\gamma_b)$.

Lemma 9.2.8. *For functors $F, G : A \to B$ and $H, K : B \to C$ and natural transformations $\gamma : F \to G$ and $\delta : H \to K$, we have*

$$(\delta G)(H\gamma) = (K\gamma)(\delta F).$$

Proof. It suffices to check componentwise: at $a : A$ we have

$$\begin{aligned}
((\delta G)(H\gamma))_a &\equiv (\delta G)_a (H\gamma)_a \\
&\equiv \delta_{Ga} \circ H(\gamma_a) \\
&= K(\gamma_a) \circ \delta_{Fa} && \text{(by naturality of } \delta) \\
&\equiv (K\gamma)_a \circ (\delta F)_a \\
&\equiv ((K\gamma)(\delta F))_a. && \square
\end{aligned}$$

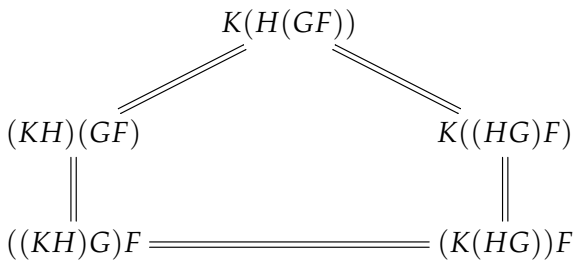
Classically, one defines the "horizontal composite" of $\gamma : F \to G$ and $\delta : H \to K$ to be the common value of $(\delta G)(H\gamma)$ and $(K\gamma)(\delta F)$. We will refrain from doing this, because while equal, these two transformations are not *definitionally* equal. This also has the consequence that we can use the symbol $\circ$ (or juxtaposition) for all kinds of composition unambiguously: there is only one way to compose two natural transformations (as opposed to composing a natural transformation with a functor on either side).

Lemma 9.2.9. *Composition of functors is associative: $H(GF) = (HG)F$.*

Proof. Since composition of functions is associative, this follows immediately for the actions on objects and on homs. And since hom-sets are sets, the rest of the data is automatic. □

The equality in Lemma 9.2.9 is likewise not definitional. (Composition of functions is definitionally associative, but the axioms that go into a functor must also be composed, and this breaks definitional associativity.) For this reason, we need also to know about *coherence* for associativity.

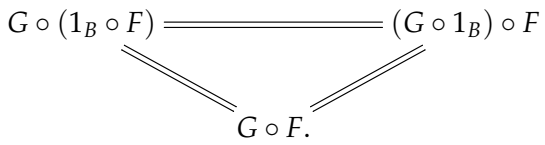
Lemma 9.2.10. *Lemma 9.2.9 is coherent, i.e. the following pentagon of equalities commutes:*



Proof. As in Lemma 9.2.9, this is evident for the actions on objects, and the rest is automatic. □

We will henceforth abuse notation by writing $H \circ G \circ F$ or HGF for either $H(GF)$ or $(HG)F$, transporting along Lemma 9.2.9 whenever necessary. We have a similar coherence result for units.

Lemma 9.2.11. *For a functor $F : A \to B$, we have equalities $(1_B \circ F) = F$ and $(F \circ 1_A) = F$, such that given also $G : B \to C$, the following triangle of equalities commutes.*



See Exercises 9.4 and 9.5 for further development of these ideas.

9.3 Adjunctions

The definition of adjoint functors is straightforward; the main interesting aspect arises from proof-relevance.

Definition 9.3.1. A functor $F : A \to B$ is a **left adjoint** if there exists

- A functor $G : B \to A$.
- A natural transformation $\eta : 1_A \to GF$ (the **unit**).
- A natural transformation $\epsilon : FG \to 1_B$ (the **counit**).
- $(\epsilon F)(F\eta) = 1_F$.
- $(G\epsilon)(\eta G) = 1_G$.

The last two equations are called the **triangle identities** or **zigzag identities**. We leave it to the reader to define right adjoints analogously.

Lemma 9.3.2. *If A is a category (but B may be only a precategory), then the type "F is a left adjoint" is a mere proposition.*

Proof. Suppose we are given (G, η, ϵ) with the triangle identities and also (G', η', ϵ'). Define $\gamma : G \to G'$ to be $(G'\epsilon)(\eta' G)$, and $\delta : G' \to G$ to be $(G\epsilon')(\eta G')$. Then

$$\begin{aligned}\delta\gamma &= (G\epsilon')(\eta G')(G'\epsilon)(\eta' G) \\ &= (G\epsilon')(GFG'\epsilon)(\eta G'FG)(\eta' G) \\ &= (G\epsilon)(G\epsilon' FG)(GF\eta' G)(\eta G) \\ &= (G\epsilon)(\eta G) \\ &= 1_G\end{aligned}$$

using Lemma 9.2.8 and the triangle identities. Similarly, we show $\gamma\delta = 1_{G'}$, so γ is a natural isomorphism $G \cong G'$. By Theorem 9.2.5, we have an identity $G = G'$.

Now we need to know that when η and ϵ are transported along this identity, they become equal to η' and ϵ'. By Lemma 9.1.9, this transport is given by composing with γ or δ as appropriate. For η, this yields

$$(G'\epsilon F)(\eta' GF)\eta = (G'\epsilon F)(G'F\eta)\eta' = \eta'$$

using Lemma 9.2.8 and the triangle identity. The case of ϵ is similar. Finally, the triangle identities transport correctly automatically, since hom-sets are sets. □

In §9.5 we will give another proof of Lemma 9.3.2.

9.4 Equivalences

It is usual in category theory to define an *equivalence of categories* to be a functor $F : A \to B$ such that there exists a functor $G : B \to A$ and natural isomorphisms $FG \cong 1_B$ and $GF \cong 1_A$. Unlike the property of being an adjunction, however, this would not be a mere proposition without truncating it, for the same reasons that the type of quasi-inverses is ill-behaved (see §4.1). And as in §4.2, we can avoid this by using the usual notion of *adjoint* equivalence.

Definition 9.4.1. A functor $F : A \to B$ is an **equivalence of (pre)categories** if it is a left adjoint for which η and ϵ are isomorphisms. We write $A \simeq B$ for the type of equivalences of categories from A to B.

By Lemmas 9.1.3 and 9.3.2, if A is a category, then the type "F is an equivalence of precategories" is a mere proposition.

Lemma 9.4.2. *If for $F : A \to B$ there exists $G : B \to A$ and isomorphisms $GF \cong 1_A$ and $FG \cong 1_B$, then F is an equivalence of precategories.*

Proof. Just like the proof of Theorem 4.2.3 for equivalences of types. □

Definition 9.4.3. We say a functor $F : A \to B$ is **faithful** if for all $a, b : A$, the function

$$F_{a,b} : \hom_A(a, b) \to \hom_B(Fa, Fb)$$

is injective, and **full** if for all $a, b : A$ this function is surjective. If it is both (hence each $F_{a,b}$ is an equivalence) we say F is **fully faithful**.

Definition 9.4.4. We say a functor $F : A \to B$ is **split essentially surjective** if for all $b : B$ there exists an $a : A$ such that $Fa \cong b$.

Lemma 9.4.5. *For any precategories A and B and functor $F : A \to B$, the following types are equivalent.*

(i) F is an equivalence of precategories.

(ii) F is fully faithful and split essentially surjective.

Proof. Suppose F is an equivalence of precategories, with G, η, ϵ specified. Then we have the function

$$\begin{aligned} \hom_B(Fa, Fb) &\to \hom_A(a, b), \\ g &\mapsto {\eta_b}^{-1} \circ G(g) \circ \eta_a. \end{aligned}$$

For $f : \hom_A(a, b)$, we have

$${\eta_b}^{-1} \circ G(F(f)) \circ \eta_a = {\eta_b}^{-1} \circ \eta_b \circ f = f$$

while for $g : \hom_B(Fa, Fb)$ we have

$$\begin{aligned} F({\eta_b}^{-1} \circ G(g) \circ \eta_a) &= F({\eta_b}^{-1}) \circ F(G(g)) \circ F(\eta_a) \\ &= \epsilon_{Fb} \circ F(G(g)) \circ F(\eta_a) \\ &= g \circ \epsilon_{Fa} \circ F(\eta_a) \\ &= g \end{aligned}$$

using naturality of ϵ, and the triangle identities twice. Thus, $F_{a,b}$ is an equivalence, so F is fully faithful. Finally, for any $b : B$, we have $Gb : A$ and $\epsilon_b : FGb \cong b$.

On the other hand, suppose F is fully faithful and split essentially surjective. Define $G_0 : B_0 \to A_0$ by sending $b : B$ to the $a : A$ given by the specified essential splitting, and write ϵ_b for the likewise specified isomorphism $FGb \cong b$.

Now for any $g : \hom_B(b, b')$, define $G(g) : \hom_A(Gb, Gb')$ to be the unique morphism such that $F(G(g)) = {(\epsilon_{b'})}^{-1} \circ g \circ \epsilon_b$ (which exists since F is fully faithful). Finally, for $a : A$ define $\eta_a : \hom_A(a, GFa)$ to be the unique morphism such that $F\eta_a = {\epsilon_{Fa}}^{-1}$. It is easy to verify that G is a functor and that (G, η, ϵ) exhibit F as an equivalence of precategories.

Now consider the composite (i)→(ii)→(i). We clearly recover the same function $G_0 : B_0 \to A_0$. For the action of G on hom-sets, we must show that for $g : \hom_B(b, b')$, $G(g)$ is the (necessarily unique) morphism such that $F(G(g)) = {(\epsilon_{b'})}^{-1} \circ g \circ \epsilon_b$. But this equation holds by the assumed naturality of ϵ. We also clearly recover ϵ, while η is uniquely characterized by $F\eta_a = {\epsilon_{Fa}}^{-1}$ (which is one of the triangle identities assumed to hold in the structure of an equivalence of precategories). Thus, this composite is equal to the identity.

Finally, consider the other composite (ii)→(i)→(ii). Since being fully faithful is a mere proposition, it suffices to observe that we recover, for each $b : B$, the same $a : A$ and isomorphism $Fa \cong b$. But this is clear, since we used this function and isomorphism to define G_0 and ϵ in (i), which in turn are precisely what we used to recover (ii) again. Thus, the composites in both directions are equal to identities, hence we have an equivalence (i) $\simeq$ (ii). □

However, if B is not a category, then neither type in Lemma 9.4.5 may necessarily be a mere proposition. This suggests considering as well the following notions.

Definition 9.4.6. A functor $F : A \to B$ is **essentially surjective** if for all $b : B$, there *merely* exists an $a : A$ such that $Fa \cong b$. We say F is a **weak equivalence** if it is fully faithful and essentially surjective.

Being a weak equivalence is *always* a mere proposition. For categories, however, there is no difference between equivalences and weak ones.

Lemma 9.4.7. *If $F : A \to B$ is fully faithful and A is a category, then for any $b : B$ the type $\sum_{(a:A)}(Fa \cong b)$ is a mere proposition. Hence a functor between categories is an equivalence if and only if it is a weak equivalence.*

Proof. Suppose given (a, f) and (a', f') in $\sum_{(a:A)}(Fa \cong b)$. Then $f'^{-1} \circ f$ is an isomorphism $Fa \cong Fa'$. Since F is fully faithful, we have $g : a \cong a'$ with $Fg = f'^{-1} \circ f$. And since A is a category, we have $p : a = a'$ with $\mathsf{idtoiso}(p) = g$. Now $Fg = f'^{-1} \circ f$ implies $((F_0)(p))_*(f) = f'$, hence (by the characterization of equalities in dependent pair types) $(a, f) = (a', f')$.

Thus, for fully faithful functors whose domain is a category, essential surjectivity is equivalent to split essential surjectivity, and so being a weak equivalence is equivalent to being an equivalence. □

This is an important advantage of our category theory over set-based approaches. With a purely set-based definition of category, the statement "every fully faithful and essentially surjective functor is an equivalence of categories" is equivalent to the axiom of choice AC. Here we have it for free, as a category-theoretic version of the principle of unique choice (§3.9). (In fact, this property characterizes categories among precategories; see §9.9.)

On the other hand, the following characterization of equivalences of categories is perhaps even more useful.

Definition 9.4.8. A functor $F : A \to B$ is an **isomorphism of (pre)categories** if F is fully faithful and $F_0 : A_0 \to B_0$ is an equivalence of types.

This definition is an exception to our general rule (see §2.4) of only using the word "isomorphism" for sets and set-like objects. However, it does carry an appropriate connotation here, because for general precategories, isomorphism is stronger than equivalence.

Note that being an isomorphism of precategories is always a mere property. Let $A \cong B$ denote the type of isomorphisms of (pre)categories from A to B.

Lemma 9.4.9. *For precategories A and B and $F : A \to B$, the following are equivalent.*

(i) F is an isomorphism of precategories.

(ii) There exist $G : B \to A$ and $\eta : 1_A = GF$ and $\epsilon : FG = 1_B$ such that

$$\mathsf{ap}_{(\lambda H.\, FH)}(\eta) = \mathsf{ap}_{(\lambda K.\, KF)}(\epsilon^{-1}). \tag{9.4.10}$$

(iii) There merely exist $G : B \to A$ and $\eta : 1_A = GF$ and $\epsilon : FG = 1_B$.

Note that if B_0 is not a 1-type, then (9.4.10) may not be a mere proposition.

Proof. First note that since hom-sets are sets, equalities between equalities of functors are uniquely determined by their object-parts. Thus, by function extensionality, (9.4.10) is equivalent to

$$(F_0)(\eta_0)_a = (\epsilon_0)^{-1}_{F_0 a}. \tag{9.4.11}$$

for all $a : A_0$. Note that this is precisely the triangle identity for G_0, η_0, and ϵ_0 to be a proof that F_0 is a half adjoint equivalence of types.

Now suppose (i). Let $G_0 : B_0 \to A_0$ be the inverse of F_0, with $\eta_0 : \mathsf{id}_{A_0} = G_0F_0$ and $\epsilon_0 : F_0G_0 = \mathsf{id}_{B_0}$ satisfying the triangle identity, which is precisely (9.4.11). Now define $G_{b,b'} : \hom_B(b,b') \to \hom_A(G_0b, G_0b')$ by

$$G_{b,b'}(g) :\equiv (F_{G_0b,G_0b'})^{-1}\Big(\mathsf{idtoiso}((\epsilon_0)^{-1}{}_{b'}) \circ g \circ \mathsf{idtoiso}((\epsilon_0)_b)\Big)$$

(using the assumption that F is fully faithful). Since idtoiso takes inverses to inverses and concatenation to composition, and F is a functor, it follows that G is a functor.

By definition, we have $(GF)_0 \equiv G_0F_0$, which is equal to id_{A_0} by η_0. To obtain $1_A = GF$, we need to show that when transported along η_0, the identity function of $\hom_A(a,a')$ becomes equal to the composite $G_{Fa,Fa'} \circ F_{a,a'}$. In other words, for any $f : \hom_A(a,a')$ we must have

$$\mathsf{idtoiso}((\eta_0)_{a'}) \circ f \circ \mathsf{idtoiso}((\eta_0)^{-1}{}_a) = (F_{GFa,GFa'})^{-1}\Big(\mathsf{idtoiso}((\epsilon_0)^{-1}{}_{Fa'}) \circ F_{a,a'}(f) \circ \mathsf{idtoiso}((\epsilon_0)_{Fa})\Big).$$

But this is equivalent to

$$(F_{GFa,GFa'})\Big(\mathsf{idtoiso}((\eta_0)_{a'}) \circ f \circ \mathsf{idtoiso}((\eta_0)^{-1}{}_a)\Big) = \mathsf{idtoiso}((\epsilon_0)^{-1}{}_{Fa'}) \circ F_{a,a'}(f) \circ \mathsf{idtoiso}((\epsilon_0)_{Fa}).$$

which follows from functoriality of F, the fact that F preserves idtoiso, and (9.4.11). Thus we have $\eta : 1_A = GF$.

On the other side, we have $(FG)_0 \equiv F_0G_0$, which is equal to id_{B_0} by ϵ_0. To obtain $FG = 1_B$, we need to show that when transported along ϵ_0, the identity function of $\hom_B(b,b')$ becomes equal to the composite $F_{Gb,Gb'} \circ G_{b,b'}$. That is, for any $g : \hom_B(b,b')$ we must have

$$F_{Gb,Gb'}\Big((F_{Gb,Gb'})^{-1}\Big(\mathsf{idtoiso}((\epsilon_0)^{-1}{}_{b'}) \circ g \circ \mathsf{idtoiso}((\epsilon_0)_b)\Big)\Big) = \mathsf{idtoiso}((\epsilon_0{}^{-1})_{b'}) \circ g \circ \mathsf{idtoiso}((\epsilon_0)_b).$$

But this is just the fact that $(F_{Gb,Gb'})^{-1}$ is the inverse of $F_{Gb,Gb'}$. And we have remarked that (9.4.10) is equivalent to (9.4.11), so (ii) holds.

Conversely, suppose given (ii); then the object-parts of G, η, and ϵ together with (9.4.11) show that F_0 is an equivalence of types. And for $a, a' : A_0$, we define $\overline{G}_{a,a'} : \hom_B(Fa,Fa') \to \hom_A(a,a')$ by

$$\overline{G}_{a,a'}(g) :\equiv \mathsf{idtoiso}(\eta^{-1})_{a'} \circ G(g) \circ \mathsf{idtoiso}(\eta)_a. \tag{9.4.12}$$

By naturality of $\mathsf{idtoiso}(\eta)$, for any $f : \hom_A(a,a')$ we have

$$\begin{aligned}\overline{G}_{a,a'}(F_{a,a'}(f)) &= \mathsf{idtoiso}(\eta^{-1})_{a'} \circ G(F(f)) \circ \mathsf{idtoiso}(\eta)_a \\ &= \mathsf{idtoiso}(\eta^{-1})_{a'} \circ \mathsf{idtoiso}(\eta)_{a'} \circ f \\ &= f.\end{aligned}$$

On the other hand, for $g : \hom_B(Fa,Fa')$ we have

$$\begin{aligned}F_{a,a'}(\overline{G}_{a,a'}(g)) &= F(\mathsf{idtoiso}(\eta^{-1})_{a'}) \circ F(G(g)) \circ F(\mathsf{idtoiso}(\eta)_a) \\ &= \mathsf{idtoiso}(\epsilon)_{Fa'} \circ F(G(g)) \circ \mathsf{idtoiso}(\epsilon^{-1})_{Fa} \\ &= \mathsf{idtoiso}(\epsilon)_{Fa'} \circ \mathsf{idtoiso}(\epsilon^{-1})_{Fa'} \circ g \\ &= g.\end{aligned}$$

(There are lemmas needed here regarding the compatibility of idtoiso and whiskering, which we leave it to the reader to state and prove.) Thus, $F_{a,a'}$ is an equivalence, so F is fully faithful; i.e. (i) holds.

Now the composite (i)→(ii)→(i) is equal to the identity since (i) is a mere proposition. On the other side, tracing through the above constructions we see that the composite (ii)→(i)→(ii) essentially preserves the object-parts G_0, η_0, ϵ_0, and the object-part of (9.4.10). And in the latter three cases, the object-part is all there is, since hom-sets are sets.

Thus, it suffices to show that we recover the action of G on hom-sets. In other words, we must show that if $g : \hom_B(b, b')$, then

$$G_{b,b'}(g) = \overline{G}_{G_0 b, G_0 b'}\left(\mathsf{idtoiso}((\epsilon_0)^{-1}{}_{b'}) \circ g \circ \mathsf{idtoiso}((\epsilon_0)_b)\right)$$

where $\overline{G}$ is defined by (9.4.12). However, this follows from functoriality of G and the *other* triangle identity, which we have seen in Chapter 4 is equivalent to (9.4.11).

Now since (i) is a mere proposition, so is (ii), so it suffices to show they are logically equivalent to (iii). Of course, (ii)→(iii), so let us assume (iii). Since (i) is a mere proposition, we may assume given G, η, and ϵ. Then G_0 along with η and ϵ imply that F_0 is an equivalence. Moreover, we also have natural isomorphisms $\mathsf{idtoiso}(\eta) : 1_A \cong GF$ and $\mathsf{idtoiso}(\epsilon) : FG \cong 1_B$, so by Lemma 9.4.2, F is an equivalence of precategories, and in particular fully faithful. □

From Lemma 9.4.9(ii) and idtoiso in functor categories, we conclude immediately that any isomorphism of precategories is an equivalence. For precategories, the converse can fail.

Example 9.4.13. Let X be a type and $x_0 : X$ an element, and let X_{ch} denote the *chaotic* or *indiscrete* precategory on X. By definition, we have $(X_{\mathrm{ch}})_0 :\equiv X$, and $\hom_{X_{\mathrm{ch}}}(x, x') :\equiv \mathbf{1}$ for all x, x'. Then the unique functor $X_{\mathrm{ch}} \to \mathbf{1}$ is an equivalence of precategories, but not an isomorphism unless X is contractible.

This example also shows that a precategory can be equivalent to a category without itself being a category. Of course, if a precategory is *isomorphic* to a category, then it must itself be a category.

However, for categories, the two notions coincide.

Lemma 9.4.14. *For categories A and B, a functor $F : A \to B$ is an equivalence of categories if and only if it is an isomorphism of categories.*

Proof. Since both are mere properties, it suffices to show they are logically equivalent. So first suppose F is an equivalence of categories, with (G, η, ϵ) given. We have already seen that F is fully faithful. By Theorem 9.2.5, the natural isomorphisms η and ϵ yield identities $1_A = GF$ and $FG = 1_B$, hence in particular identities $\mathsf{id}_A = G_0 \circ F_0$ and $F_0 \circ G_0 = \mathsf{id}_B$. Thus, F_0 is an equivalence of types.

Conversely, suppose F is fully faithful and F_0 is an equivalence of types, with inverse G_0, say. Then for each $b : B$ we have $G_0 b : A$ and an identity $FGb = b$, hence an isomorphism $FGb \cong b$. Thus, by Lemma 9.4.5, F is an equivalence of categories. □

Of course, there is yet a third notion of sameness for (pre)categories: equality. However, the univalence axiom implies that it coincides with isomorphism.

Lemma 9.4.15. *If A and B are precategories, then the function*

$$(A = B) \to (A \cong B)$$

(defined by induction from the identity functor) is an equivalence of types.

Proof. As usual for dependent sum types, to give an element of $A = B$ is equivalent to giving

- an identity $P_0 : A_0 = B_0$,
- for each $a, b : A_0$, an identity

$$P_{a,b} : \hom_A(a,b) = \hom_B(P_{0*}(a), P_{0*}(b)),$$

- identities $(P_{a,a})_*(1_a) = 1_{P_{0*}(a)}$ and $(P_{a,c})_*(gf) = (P_{b,c})_*(g) \circ (P_{a,b})_*(f)$.

(Again, we use the fact that the identity types of hom-sets are mere propositions.) However, by univalence, this is equivalent to giving

- an equivalence of types $F_0 : A_0 \simeq B_0$,
- for each $a, b : A_0$, an equivalence of types

$$F_{a,b} : \hom_A(a,b) \simeq \hom_B(F_0(a), F_0(b)),$$

- and identities $F_{a,a}(1_a) = 1_{F_0(a)}$ and $F_{a,c}(gf) = F_{b,c}(g) \circ F_{a,b}(f)$.

But this consists exactly of a functor $F : A \to B$ that is an isomorphism of categories. And by induction on identity, this equivalence $(A = B) \simeq (A \cong B)$ is equal to the one obtained by induction. □

Thus, for categories, equality also coincides with equivalence. We can interpret this as saying that categories, functors, and natural transformations form, not just a pre-2-category, but a 2-category (see Exercise 9.4).

Theorem 9.4.16. *If A and B are categories, then the function*

$$(A = B) \to (A \simeq B)$$

(defined by induction from the identity functor) is an equivalence of types.

Proof. By Lemmas 9.4.14 and 9.4.15. □

As a consequence, the type of categories is a 2-type. For since $A \simeq B$ is a subtype of the type of functors from A to B, which are the objects of a category, it is a 1-type; hence the identity types $A = B$ are also 1-types.

9.5 The Yoneda lemma

Recall that we have a category $\mathcal{S}et$ whose objects are sets and whose morphisms are functions. We now show that every precategory has a $\mathcal{S}et$-valued hom-functor. First we need to define opposites and products of (pre)categories.

Definition 9.5.1. For a precategory A, its **opposite** A^{op} is a precategory with the same type of objects, with $\hom_{A^{\mathrm{op}}}(a,b) :\equiv \hom_A(b,a)$, and with identities and composition inherited from A.

Definition 9.5.2. For precategories A and B, their **product** $A \times B$ is a precategory with $(A \times B)_0 :\equiv A_0 \times B_0$ and

$$\hom_{A\times B}((a,b),(a',b')) :\equiv \hom_A(a,a') \times \hom_B(b,b').$$

Identities are defined by $1_{(a,b)} :\equiv (1_a, 1_b)$ and composition by $(g,g')(f,f') :\equiv ((gf),(g'f'))$.

Lemma 9.5.3. *For precategories A, B, C, the following types are equivalent.*

(i) Functors $A \times B \to C$.

(ii) Functors $A \to C^B$.

Proof. Given $F : A \times B \to C$, for any $a : A$ we obviously have a functor $F_a : B \to C$. This gives a function $A_0 \to (C^B)_0$. Next, for any $f : \hom_A(a, a')$, we have for any $b : B$ the morphism $F_{(a,b),(a',b)}(f, 1_b) : F_a(b) \to F_{a'}(b)$. These are the components of a natural transformation $F_a \to F_{a'}$. Functoriality in a is easy to check, so we have a functor $\hat{F} : A \to C^B$.

Conversely, suppose given $G : A \to C^B$. Then for any $a : A$ and $b : B$ we have the object $G(a)(b) : C$, giving a function $A_0 \times B_0 \to C_0$. And for $f : \hom_A(a, a')$ and $g : \hom_B(b, b')$, we have the morphism

$$G(a')_{b,b'}(g) \circ G_{a,a'}(f)_b = G_{a,a'}(f)_{b'} \circ G(a)_{b,b'}(g)$$

in $\hom_C(G(a)(b), G(a')(b'))$. Functoriality is again easy to check, so we have a functor $\check{G} : A \times B \to C$.

Finally, it is also clear that these operations are inverses. □

Now for any precategory A, we have a hom-functor

$$\hom_A : A^{\mathrm{op}} \times A \to \mathcal{S}et.$$

It takes a pair $(a, b) : (A^{\mathrm{op}})_0 \times A_0 \equiv A_0 \times A_0$ to the set $\hom_A(a, b)$. For a morphism $(f, f') : \hom_{A^{\mathrm{op}} \times A}((a, b), (a', b'))$, by definition we have $f : \hom_A(a', a)$ and $f' : \hom_A(b, b')$, so we can define

$$\begin{aligned}(\hom_A)_{(a,b),(a',b')}(f, f') :\equiv (g \mapsto (f' g f)) \\ : \hom_A(a, b) \to \hom_A(a', b').\end{aligned}$$

Functoriality is easy to check.

By Lemma 9.5.3, therefore, we have an induced functor $\mathbf{y} : A \to \mathcal{S}et^{A^{\mathrm{op}}}$, which we call the **Yoneda embedding**.

Theorem 9.5.4 (The Yoneda lemma)**.** *For any precategory A, any $a : A$, and any functor $F : \mathcal{S}et^{A^{\mathrm{op}}}$, we have an isomorphism*

$$\hom_{\mathcal{S}et^{A^{\mathrm{op}}}}(\mathbf{y}a, F) \cong Fa. \tag{9.5.5}$$

Moreover, this is natural in both a and F.

Proof. Given a natural transformation $\alpha : \mathbf{y}a \to F$, we can consider the component $\alpha_a : \mathbf{y}a(a) \to Fa$. Since $\mathbf{y}a(a) \equiv \hom_A(a, a)$, we have $1_a : \mathbf{y}a(a)$, so that $\alpha_a(1_a) : Fa$. This gives a function $(\alpha \mapsto \alpha_a(1_a))$ from left to right in (9.5.5).

In the other direction, given $x : Fa$, we define $\alpha : \mathbf{y}a \to F$ by

$$\alpha_{a'}(f) :\equiv F_{a,a'}(f)(x).$$

Naturality is easy to check, so this gives a function from right to left in (9.5.5).

To show that these are inverses, first suppose given $x : Fa$. Then with α defined as above, we have $\alpha_a(1_a) = F_{a,a}(1_a)(x) = 1_{Fa}(x) = x$. On the other hand, if we suppose given $\alpha : \mathbf{y}a \to F$ and

define x as above, then for any $f : \hom_A(a', a)$ we have

$$\begin{aligned}\alpha_{a'}(f) &= \alpha_{a'}(\mathbf{y}a_{a,a'}(f)(1_a)) \\ &= (\alpha_{a'} \circ \mathbf{y}a_{a,a'}(f))(1_a) \\ &= (F_{a,a'}(f) \circ \alpha_a)(1_a) \\ &= F_{a,a'}(f)(\alpha_a(1_a)) \\ &= F_{a,a'}(f)(x).\end{aligned}$$

Thus, both composites are equal to identities. We leave the proof of naturality to the reader. □

Corollary 9.5.6. *The Yoneda embedding $\mathbf{y} : A \to \mathcal{S}et^{A^{\mathrm{op}}}$ is fully faithful.*

Proof. By Theorem 9.5.4, we have

$$\hom_{\mathcal{S}et^{A^{\mathrm{op}}}}(\mathbf{y}a, \mathbf{y}b) \cong \mathbf{y}b(a) \equiv \hom_A(a, b).$$

It is easy to check that this isomorphism is in fact the action of $\mathbf{y}$ on hom-sets. □

Corollary 9.5.7. *If A is a category, then $\mathbf{y}_0 : A_0 \to (\mathcal{S}et^{A^{\mathrm{op}}})_0$ is an embedding. In particular, if $\mathbf{y}a = \mathbf{y}b$, then $a = b$.*

Proof. By Corollary 9.5.6, $\mathbf{y}$ induces an isomorphism on sets of isomorphisms. But as A and $\mathcal{S}et^{A^{\mathrm{op}}}$ are categories and $\mathbf{y}$ is a functor, this is equivalently an isomorphism on identity types, which is the definition of being an embedding. □

Definition 9.5.8. A functor $F : \mathcal{S}et^{A^{\mathrm{op}}}$ is said to be **representable** if there exists $a : A$ and an isomorphism $\mathbf{y}a \cong F$.

Theorem 9.5.9. *If A is a category, then the type "F is representable" is a mere proposition.*

Proof. By definition "F is representable" is just the fiber of $\mathbf{y}_0$ over F. Since $\mathbf{y}_0$ is an embedding by Corollary 9.5.7, this fiber is a mere proposition. □

In particular, in a category, any two representations of the same functor are equal. We can use this to give a different proof of Lemma 9.3.2. First we give a characterization of adjunctions in terms of representability.

Lemma 9.5.10. *For any precategories A and B and a functor $F : A \to B$, the following types are equivalent.*

(i) F is a left adjoint.
(ii) For each $b : B$, the functor $(a \mapsto \hom_B(Fa, b))$ from A^{op} to $\mathcal{S}et$ is representable.

Proof. An element of the type (ii) consists of a function $G_0 : B_0 \to A_0$ together with, for every $a : A$ and $b : B$ an isomorphism

$$\gamma_{a,b} : \hom_B(Fa, b) \cong \hom_A(a, G_0 b)$$

such that $\gamma_{a,b}(g \circ Ff) = \gamma_{a',b}(g) \circ f$ for $f : \hom_A(a, a')$.

Given this, for $a : A$ we define $\eta_a :\equiv \gamma_{a,Fa}(1_{Fa})$, and for $b : B$ we define $\epsilon_b :\equiv (\gamma_{Gb,b})^{-1}(1_{Gb})$. Now for $g : \hom_B(b, b')$ we define

$$G_{b,b'}(g) :\equiv \gamma_{Gb,b'}(g \circ \epsilon_b)$$

The verifications that G is a functor and η and ϵ are natural transformations satisfying the triangle identities are exactly as in the classical case, and as they are all mere propositions we will not care about their values. Thus, we have a function (ii)→(i).

In the other direction, if F is a left adjoint, we of course have G_0 specified, and we can take $\gamma_{a,b}$ to be the composite

$$\hom_B(Fa, b) \xrightarrow{G_{Fa,b}} \hom_A(GFa, Gb) \xrightarrow{(-\circ\eta_a)} \hom_A(a, Gb).$$

This is clearly natural since η is, and it has an inverse given by

$$\hom_A(a, Gb) \xrightarrow{F_{a,Gb}} \hom_B(Fa, FGb) \xrightarrow{(\epsilon_b\circ-)} \hom_A(Fa, b)$$

(by the triangle identities). Thus we also have (i)→ (ii).

For the composite (ii)→(i)→ (ii), clearly the function G_0 is preserved, so it suffices to check that we get back γ. But the new γ is defined to take $f : \hom_B(Fa, b)$ to

$$\begin{aligned} G(f) \circ \eta_a &\equiv \gamma_{GFa,b}(f \circ \epsilon_{Fa}) \circ \eta_a \\ &= \gamma_{GFa,b}(f \circ \epsilon_{Fa} \circ F\eta_a) \\ &= \gamma_{GFa,b}(f) \end{aligned}$$

so it agrees with the old one.

Finally, for (i)→(ii)→ (i), we certainly get back the functor G on objects. The new $G_{b,b'} : \hom_B(b, b') \to \hom_A(Gb, Gb')$ is defined to take g to

$$\begin{aligned} \gamma_{Gb,b'}(g \circ \epsilon_b) &\equiv G(g \circ \epsilon_b) \circ \eta_{Gb} \\ &= G(g) \circ G\epsilon_b \circ \eta_{Gb} \\ &= G(g) \end{aligned}$$

so it agrees with the old one. The new η_a is defined to be $\gamma_{a,Fa}(1_{Fa}) \equiv G(1_{Fa}) \circ \eta_a$, so it equals the old η_a. And finally, the new ϵ_b is defined to be $(\gamma_{Gb,b})^{-1}(1_{Gb}) \equiv \epsilon_b \circ F(1_{Gb})$, which also equals the old ϵ_b. □

Corollary 9.5.11. *[Lemma 9.3.2] If A is a category and $F : A \to B$, then the type "F is a left adjoint" is a mere proposition.*

Proof. By Theorem 9.5.9, if A is a category then the type in Lemma 9.5.10(ii) is a mere proposition. □

9.6 Strict categories

Definition 9.6.1. A **strict category** is a precategory whose type of objects is a set.

In accordance with the mathematical red herring principle, a strict category is not necessarily a category. In fact, a category is a strict category precisely when it is gaunt (Example 9.1.15). Most of the time, category theory is about categories, not strict ones, but sometimes one wants to consider strict categories. The main advantage of this is that strict categories have a stricter notion of "sameness" than equivalence, namely isomorphism (or equivalently, by Lemma 9.4.15, equality).

Here is one origin of strict categories.

Example 9.6.2. Let A be a precategory and $x : A$ an object. Then there is a precategory $\mathsf{mono}(A, x)$ as follows:

- Its objects consist of an object $y : A$ and a monomorphism $m : \hom_A(y, x)$. (As usual, $m : \hom_A(y, x)$ is a **monomorphism** (or is **monic**) if $(m \circ f = m \circ g) \Rightarrow (f = g)$.)
- Its morphisms from (y, m) to (z, n) are arbitrary morphisms from y to z in A (not necessarily respecting m and n).

An equality $(y, m) = (z, n)$ of objects in $\mathsf{mono}(A, x)$ consists of an equality $p : y = z$ and an equality $p_*(m) = n$, which by Lemma 9.1.9 is equivalently an equality $m = n \circ \mathsf{idtoiso}(p)$. Since hom-sets are sets, the type of such equalities is a mere proposition. But since m and n are monomorphisms, the type of morphisms f such that $m = n \circ f$ is also a mere proposition. Thus, if A is a category, then $(y, m) = (z, n)$ is a mere proposition, and hence $\mathsf{mono}(A, x)$ is a strict category.

This example can be dualized, and generalized in various ways. Here is an interesting application of strict categories.

Example 9.6.3. Let E/F be a finite Galois extension of fields, and G its Galois group. Then there is a strict category whose objects are intermediate fields $F \subseteq K \subseteq E$, and whose morphisms are field homomorphisms which fix F pointwise (but need not commute with the inclusions into E). There is another strict category whose objects are subgroups $H \subseteq G$, and whose morphisms are morphisms of G-sets $G/H \to G/K$. The fundamental theorem of Galois theory says that these two precategories are isomorphic (not merely equivalent).

9.7 †-categories

It is also worth mentioning a useful kind of precategory whose type of objects is not a set, but which is not a category either.

Definition 9.7.1. A **†-precategory** is a precategory A together with the following.

(i) For each $x, y : A$, a function $(-)^\dagger : \hom_A(x, y) \to \hom_A(y, x)$.
(ii) For all $x : A$, we have $(1_x)^\dagger = 1_x$.
(iii) For all f, g we have $(g \circ f)^\dagger = f^\dagger \circ g^\dagger$.
(iv) For all f we have $(f^\dagger)^\dagger = f$.

Definition 9.7.2. A morphism $f : \hom_A(x, y)$ in a †-precategory is **unitary** if $f^\dagger \circ f = 1_x$ and $f \circ f^\dagger = 1_y$.

Of course, every unitary morphism is an isomorphism, and being unitary is a mere proposition. Thus for each $x, y : A$ we have a set of unitary isomorphisms from x to y, which we denote $(x \cong^\dagger y)$.

Lemma 9.7.3. *If $p : (x = y)$, then $\mathsf{idtoiso}(p)$ is unitary.*

Proof. By induction, we may assume p is refl_x. But then $(1_x)^\dagger \circ 1_x = 1_x \circ 1_x = 1_x$ and similarly. □

Definition 9.7.4. A **†-category** is a †-precategory such that for all $x, y : A$, the function

$$(x = y) \to (x \cong^{\dagger} y)$$

from Lemma 9.7.3 is an equivalence.

Example 9.7.5. The category $\mathcal{R}el$ from Example 9.1.19 becomes a †-precategory if we define $(R^{\dagger})(y, x) :\equiv R(x, y)$. The proof that $\mathcal{R}el$ is a category actually shows that every isomorphism is unitary; hence $\mathcal{R}el$ is also a †-category.

Example 9.7.6. Any groupoid becomes a †-category if we define $f^{\dagger} :\equiv f^{-1}$.

Example 9.7.7. Let $\mathcal{H}ilb$ be the following precategory.

- Its objects are finite-dimensional vector spaces equipped with an inner product $\langle -, - \rangle$.
- Its morphisms are arbitrary linear maps.

By standard linear algebra, any linear map $f : V \to W$ between finite dimensional inner product spaces has a uniquely defined adjoint $f^{\dagger} : W \to V$, characterized by $\langle fv, w \rangle = \langle v, f^{\dagger} w \rangle$. In this way, $\mathcal{H}ilb$ becomes a †-precategory. Moreover, a linear isomorphism is unitary precisely when it is an **isometry**, i.e. $\langle fv, fw \rangle = \langle v, w \rangle$. It follows from this that $\mathcal{H}ilb$ is a †-category, though it is not a category (not every linear isomorphism is unitary).

There has been a good deal of general theory developed for †-categories under classical foundations. It was observed early on that the unitary isomorphisms, not arbitrary isomorphisms, are the correct notion of "sameness" for objects of a †-category, which has caused some consternation among category theorists. Homotopy type theory resolves this issue by identifying †-categories, like strict categories, as simply a different kind of precategory.

9.8 The structure identity principle

The *structure identity principle* is an informal principle that expresses that isomorphic structures are identical. We aim to prove a general abstract result which can be applied to a wide family of notions of structure, where structures may be many-sorted or even dependently-sorted, infinitary, or even higher order.

The simplest kind of single-sorted structure consists of a type with no additional structure. The univalence axiom expresses the structure identity principle for that notion of structure in a strong form: for types A, B, the canonical function $(A = B) \to (A \simeq B)$ is an equivalence.

We start with a precategory X. In our application to single-sorted first order structures, X will be the category of $\mathcal{U}$-small sets, where $\mathcal{U}$ is a univalent type universe.

Definition 9.8.1. A **notion of structure** (P, H) over X consists of the following.

(i) A type family $P : X_0 \to \mathcal{U}$. For each $x : X_0$ the elements of Px are called (P, H)**-structures** on x.

(ii) For $x, y : X_0$ and $\alpha : Px$, $\beta : Py$, to each $f : \hom_X(x, y)$ a mere proposition

$$H_{\alpha\beta}(f).$$

If $H_{\alpha\beta}(f)$ is true, we say that f is a (P, H)**-homomorphism** from α to β.

(iii) For $x : X_0$ and $\alpha : Px$, we have $H_{\alpha\alpha}(1_x)$.

(iv) For $x, y, z : X_0$ and $\alpha : Px$, $\beta : Py$, $\gamma : Pz$, if $f : \hom_X(x, y)$ and $g : \hom_X(y, z)$, we have

$$H_{\alpha\beta}(f) \to H_{\beta\gamma}(g) \to H_{\alpha\gamma}(g \circ f).$$

When (P, H) is a notion of structure, for $\alpha, \beta : Px$ we define

$$(\alpha \leq_x \beta) :\equiv H_{\alpha\beta}(1_x).$$

By (iii) and (iv), this is a preorder (Example 9.1.14) with Px its type of objects. We say that (P, H) is a **standard notion of structure** if this preorder is in fact a partial order, for all $x : X$.

Note that for a standard notion of structure, each type Px must actually be a set. We now define, for any notion of structure (P, H), a **precategory of (P, H)-structures**, $A = \mathsf{Str}_{(P,H)}(X)$.

- The type of objects of A is the type $A_0 :\equiv \sum_{(x:X_0)} Px$. If $a \equiv (x, \alpha) : A_0$, we may write $|a| :\equiv x$.
- For $(x, \alpha) : A_0$ and $(y, \beta) : A_0$, we define

$$\hom_A((x, \alpha), (y, \beta)) :\equiv \left\{ f : x \to y \;\middle|\; H_{\alpha\beta}(f) \right\}.$$

The composition and identities are inherited from X; conditions (iii) and (iv) ensure that these lift to A.

Theorem 9.8.2 (Structure identity principle)**.** *If X is a category and (P, H) is a standard notion of structure over X, then the precategory $\mathsf{Str}_{(P,H)}(X)$ is a category.*

Proof. By the definition of equality in dependent pair types, to give an equality $(x, \alpha) = (y, \beta)$ consists of

- An equality $p : x = y$, and
- An equality $p_*(\alpha) = \beta$.

Since P is set-valued, the latter is a mere proposition. On the other hand, it is easy to see that an isomorphism $(x, \alpha) \cong (y, \beta)$ in $\mathsf{Str}_{(P,H)}(X)$ consists of

- An isomorphism $f : x \cong y$ in X, such that
- $H_{\alpha\beta}(f)$ and $H_{\beta\alpha}(f^{-1})$.

Of course, the second of these is also a mere proposition. And since X is a category, the function $(x = y) \to (x \cong y)$ is an equivalence. Thus, it will suffice to show that for any $p : x = y$ and for any $(\alpha : Px)$, $(\beta : Py)$, we have $p_*(\alpha) = \beta$ if and only if both $H_{\alpha\beta}(\mathsf{idtoiso}(p))$ and $H_{\beta\alpha}(\mathsf{idtoiso}(p)^{-1})$.

The "only if" direction is just the existence of the function idtoiso for the category $\mathsf{Str}_{(P,H)}(X)$. For the "if" direction, by induction on p we may assume that $y \equiv x$ and $p \equiv \mathsf{refl}_x$. However, in this case $\mathsf{idtoiso}(p) \equiv 1_x$ and therefore $\mathsf{idtoiso}(p)^{-1} = 1_x$. Thus, $\alpha \leq_x \beta$ and $\beta \leq_x \alpha$, which implies $\alpha = \beta$ since (P, H) is a standard notion of structure. □

As an example, this methodology gives an alternative way to express the proof of Theorem 9.2.5.

Example 9.8.3. Let A be a precategory and B a category. There is a precategory B^{A_0} whose objects are functions $A_0 \to B_0$, and whose set of morphisms from $F_0 : A_0 \to B_0$ to $G_0 : A_0 \to B_0$ is $\prod_{(a:A_0)} \hom_B(F_0 a, G_0 a)$. Composition and identities are inherited directly from those in B. It is easy to show that $\gamma : \hom_{B^{A_0}}(F_0, G_0)$ is an isomorphism exactly when each component γ_a is an isomorphism, so that we have $(F_0 \cong G_0) \simeq \prod_{(a:A_0)}(F_0 a \cong G_0 a)$. Moreover, the map $\mathsf{idtoiso} : (F_0 = G_0) \to (F_0 \cong G_0)$ of B^{A_0} is equal to the composite

$$(F_0 = G_0) \longrightarrow \prod_{a:A_0} (F_0 a = G_0 a) \longrightarrow \prod_{a:A_0} (F_0 a \cong G_0 a) \longrightarrow (F_0 \cong G_0)$$

in which the first map is an equivalence by function extensionality, the second because it is a dependent product of equivalences (since B is a category), and the third as remarked above. Thus, B^{A_0} is a category.

Now we define a notion of structure on B^{A_0} for which $P(F_0)$ is the type of operations $F : \prod_{(a,a':A_0)} \hom_A(a, a') \to \hom_B(F_0 a, F_0 a')$ which extend F_0 to a functor (i.e. preserve composition and identities). This is a set since each $\hom_B(-,-)$ is so. Given such F and G, we define $\gamma : \hom_{B^{A_0}}(F_0, G_0)$ to be a homomorphism if it forms a natural transformation. In Definition 9.2.3 we essentially verified that this is a notion of structure. Moreover, if F and F' are both structures on F_0 and the identity is a natural transformation from F to F', then for any $f : \hom_A(a, a')$ we have $F'f = F'f \circ 1_{F_0 a} = 1_{F_0 a} \circ Ff = Ff$. Applying function extensionality, we conclude $F = F'$. Thus, we have a *standard* notion of structure, and so by Theorem 9.8.2, the precategory B^A is a category.

As another example, we consider categories of structures for a first-order signature. We define a **first-order signature**, Ω, to consist of sets Ω_0 and Ω_1 of function symbols, $\omega : \Omega_0$, and relation symbols, $\omega : \Omega_1$, each having an arity $|\omega|$ that is a set. An **Ω-structure** a consists of a set $|a|$ together with an assignment of an $|\omega|$-ary function $\omega^a : |a|^{|\omega|} \to |a|$ on $|a|$ to each function symbol, ω, and an assignment of an $|\omega|$-ary relation ω^a on $|a|$, assigning a mere proposition $\omega^a x$ to each $x : |a|^{|\omega|}$, to each relation symbol. And given Ω-structures a, b, a function $f : |a| \to |b|$ is a **homomorphism** $a \to b$ if it preserves the structure; i.e. if for each symbol ω of the signature and each $x : |a|^{|\omega|}$,

(i) $f(\omega^a x) = \omega^b(f \circ x)$ if $\omega : \Omega_0$, and
(ii) $\omega^a x \to \omega^b(f \circ x)$ if $\omega : \Omega_1$.

Note that each $x : |a|^{|\omega|}$ is a function $x : |\omega| \to |a|$ so that $f \circ x : b^\omega$.

Now we assume given a (univalent) universe $\mathcal{U}$ and a $\mathcal{U}$-small signature Ω; i.e. $|\Omega|$ is a $\mathcal{U}$-small set and, for each $\omega : |\Omega|$, the set $|\omega|$ is $\mathcal{U}$-small. Then we have the category $\mathcal{S}et_\mathcal{U}$ of $\mathcal{U}$-small sets. We want to define the precategory of $\mathcal{U}$-small Ω-structures over $\mathcal{S}et_\mathcal{U}$ and use Theorem 9.8.2 to show that it is a category.

We use the first order signature Ω to give us a standard notion of structure (P, H) over $\mathcal{S}et_\mathcal{U}$.

Definition 9.8.4.

(i) For each $\mathcal{U}$-small set x define

$$Px :\equiv P_0 x \times P_1 x.$$

Here

$$P_0 x :\equiv \prod_{\omega:\Omega_0} x^{|\omega|} \to x, \text{ and}$$
$$P_1 x :\equiv \prod_{\omega:\Omega_1} x^{|\omega|} \to \mathsf{Prop}_\mathcal{U},$$

(ii) For $\mathcal{U}$-small sets x, y and $\alpha : P^\omega x,\ \beta : P^\omega y,\ f : x \to y$, define

$$H_{\alpha\beta}(f) :\equiv H_{0,\alpha\beta}(f) \wedge H_{1,\alpha\beta}(f).$$

Here

$$H_{0,\alpha\beta}(f) :\equiv \forall(\omega : \Omega_0).\forall(u : x^{|\omega|}).f(\alpha u) = \beta(f \circ u), \text{ and}$$
$$H_{1,\alpha\beta}(f) :\equiv \forall(\omega : \Omega_1).\forall(u : x^{|\omega|}).\alpha u \to \beta(f \circ u).$$

It is now routine to check that (P, H) is a standard notion of structure over $\mathcal{S}et_\mathcal{U}$ and hence we may use Theorem 9.8.2 to get that the precategory $Str_{(P,H)}(\mathcal{S}et_\mathcal{U})$ is a category. It only remains to observe that this is essentially the same as the precategory of $\mathcal{U}$-small Ω-structures over $\mathcal{S}et_\mathcal{U}$.

9.9 The Rezk completion

In this section we will give a universal way to replace a precategory by a category. In fact, we will give two. Both rely on the fact that "categories see weak equivalences as equivalences".

To prove this, we begin with a couple of lemmas which are completely standard category theory, phrased carefully so as to make sure we are using the eliminator for $\|-\|_{-1}$ correctly. One would have to be similarly careful in classical category theory if one wanted to avoid the axiom of choice: any time we want to define a function, we need to characterize its values uniquely somehow.

Lemma 9.9.1. *If A, B, C are precategories and $H : A \to B$ is an essentially surjective functor, then $(- \circ H) : C^B \to C^A$ is faithful.*

Proof. Let $F, G : B \to C$, and $\gamma, \delta : F \to G$ be such that $\gamma H = \delta H$; we must show $\gamma = \delta$. Thus let $b : B$; we want to show $\gamma_b = \delta_b$. This is a mere proposition, so since H is essentially surjective, we may assume given an $a : A$ and an isomorphism $f : Ha \cong b$. But now we have

$$\gamma_b = G(f) \circ \gamma_{Ha} \circ F(f^{-1}) = G(f) \circ \delta_{Ha} \circ F(f^{-1}) = \delta_b. \qquad \square$$

Lemma 9.9.2. *If A, B, C are precategories and $H : A \to B$ is essentially surjective and full, then $(- \circ H) : C^B \to C^A$ is fully faithful.*

Proof. It remains to show fullness. Thus, let $F, G : B \to C$ and $\gamma : FH \to GH$. We claim that for any $b : B$, the type

$$\sum_{(g:\hom_C(Fb,Gb))} \prod_{(a:A)} \prod_{(f:Ha\cong b)} (\gamma_a = Gf^{-1} \circ g \circ Ff) \tag{9.9.3}$$

is contractible. Since contractibility is a mere property, and H is essentially surjective, we may assume given $a_0 : A$ and $h : Ha_0 \cong b$.

Now take $g :\equiv Gh \circ \gamma_{a_0} \circ Fh^{-1}$. Then given any other $a : A$ and $f : Ha \cong b$, we must show $\gamma_a = Gf^{-1} \circ g \circ Ff$. Since H is full, there merely exists a morphism $k : \hom_A(a, a_0)$ such that $Hk = h^{-1} \circ f$. And since our goal is a mere proposition, we may assume given some such k. Then we have

$$\begin{aligned}
\gamma_a &= GHk^{-1} \circ \gamma_{a_0} \circ FHk \\
&= Gf^{-1} \circ Gh \circ \gamma_{a_0} \circ Fh^{-1} \circ Ff \\
&= Gf^{-1} \circ g \circ Ff.
\end{aligned}$$

Thus, (9.9.3) is inhabited. It remains to show it is a mere proposition. Let $g, g' : \hom_C(Fb, Gb)$ be such that for all $a : A$ and $f : Ha \cong b$, we have both $(\gamma_a = Gf^{-1} \circ g \circ Ff)$ and $(\gamma_a = Gf^{-1} \circ g' \circ Ff)$. The dependent product types are mere propositions, so all we have to prove is $g = g'$. But this is a mere proposition, so we may assume $a_0 : A$ and $h : Ha_0 \cong b$, in which case we have

$$g = Gh \circ \gamma_{a_0} \circ Fh^{-1} = g'.$$

This proves that (9.9.3) is contractible for all $b : B$. Now we define $\delta : F \to G$ by taking δ_b to be the unique g in (9.9.3) for that b. To see that this is natural, suppose given $f : \hom_B(b, b')$; we must show $Gf \circ \delta_b = \delta_{b'} \circ Ff$. As before, we may assume $a : A$ and $h : Ha \cong b$, and likewise $a' : A$ and $h' : Ha' \cong b'$. Since H is full as well as essentially surjective, we may also assume $k : \hom_A(a, a')$ with $Hk = {h'}^{-1} \circ f \circ h$.

Since γ is natural, $GHk \circ \gamma_a = \gamma_{a'} \circ FHk$. Using the definition of δ, we have

$$\begin{aligned} Gf \circ \delta_b &= Gf \circ Gh \circ \gamma_a \circ Fh^{-1} \\ &= Gh' \circ GHk \circ \gamma_a \circ Fh^{-1} \\ &= Gh' \circ \gamma_{a'} \circ FHk \circ Fh^{-1} \\ &= Gh' \circ \gamma_{a'} \circ {Fh'}^{-1} \circ Ff \\ &= \delta_{b'} \circ Ff. \end{aligned}$$

Thus, δ is natural. Finally, for any $a : A$, applying the definition of δ_{Ha} to a and 1_a, we obtain $\gamma_a = \delta_{Ha}$. Hence, $\delta \circ H = \gamma$. □

The rest of the theorem follows almost exactly the same lines, with the category-ness of C inserted in one crucial step, which we have italicized below for emphasis. This is the point at which we are trying to define a function into *objects* without using choice, and so we must be careful about what it means for an object to be "uniquely specified". In classical category theory, all one can say is that this object is specified up to unique isomorphism, but in set-theoretic foundations this is not a sufficient amount of uniqueness to give us a function without invoking AC. In univalent foundations, however, if C is a category, then isomorphism is equality, and we have the appropriate sort of uniqueness (namely, living in a contractible space).

Theorem 9.9.4. *If A, B are precategories, C is a category, and $H : A \to B$ is a weak equivalence, then $(- \circ H) : C^B \to C^A$ is an isomorphism.*

Proof. By Theorem 9.2.5, C^B and C^A are categories. Thus, by Lemma 9.4.14 it will suffice to show that $(- \circ H)$ is an equivalence. But since we know from the preceding two lemmas that it is fully faithful, by Lemma 9.4.7 it will suffice to show that it is essentially surjective. Thus, suppose $F : A \to C$; we want there to merely exist a $G : B \to C$ such that $GH \cong F$.

For each $b : B$, let X_b be the type whose elements consist of:

(i) An element $c : C$; and

(ii) For each $a : A$ and $h : Ha \cong b$, an isomorphism $k_{a,h} : Fa \cong c$; such that

(iii) For each (a, h) and (a', h') as in (ii) and each $f : \hom_A(a, a')$ such that $h' \circ Hf = h$, we have $k_{a',h'} \circ Ff = k_{a,h}$.

We claim that for any $b : B$, the type X_b is contractible. As this is a mere proposition, we may assume given $a_0 : A$ and $h_0 : Ha_0 \cong b$. Let $c^0 :\equiv Fa_0$. Next, given $a : A$ and $h : Ha \cong b$, since

H is fully faithful there is a unique isomorphism $g_{a,h} : a \to a_0$ with $Hg_{a,h} = {h_0}^{-1} \circ h$; define $k^0_{a,h} :\equiv Fg_{a,h}$. Finally, if $h' \circ Hf = h$, then ${h_0}^{-1} \circ h' \circ Hf = {h_0}^{-1} \circ h$, hence $g_{a',h'} \circ f = g_{a,h}$ and thus $k^0_{a',h'} \circ Ff = k^0_{a,h}$. Therefore, X_b is inhabited.

Now suppose given another $(c^1, k^1) : X_b$. Then $k^1_{a_0,h_0} : c^0 \equiv Fa_0 \cong c^1$. *Since C is a category, we have* $p : c^0 = c^1$ *with* $\mathsf{idtoiso}(p) = k^1_{a_0,h_0}$. And for any $a : A$ and $h : Ha \cong b$, by (iii) for (c^1, k^1) with $f :\equiv g_{a,h}$, we have

$$k^1_{a,h} = k^1_{a_0,h_0} \circ k^0_{a,h} = p_*(k^0_{a,h})$$

This gives the requisite data for an equality $(c^0, k^0) = (c^1, k^1)$, completing the proof that X_b is contractible.

Now since X_b is contractible for each b, the type $\prod_{(b:B)} X_b$ is also contractible. In particular, it is inhabited, so we have a function assigning to each $b : B$ a c and a k. Define $G_0(b)$ to be this c; this gives a function $G_0 : B_0 \to C_0$.

Next we need to define the action of G on morphisms. For each $b, b' : B$ and $f : \hom_B(b, b')$, let Y_f be the type whose elements consist of:

(iv) A morphism $g : \hom_C(Gb, Gb')$, such that

(v) For each $a : A$ and $h : Ha \cong b$, and each $a' : A$ and $h' : Ha' \cong b'$, and any $\ell : \hom_A(a, a')$, we have

$$(h' \circ H\ell = f \circ h) \to (k_{a',h'} \circ F\ell = g \circ k_{a,h}).$$

We claim that for any b, b' and f, the type Y_f is contractible. As this is a mere proposition, we may assume given $a_0 : A$ and $h_0 : Ha_0 \cong b$, and each $a'_0 : A$ and $h'_0 : Ha'_0 \cong b'$. Then since H is fully faithful, there is a unique $\ell_0 : \hom_A(a_0, a'_0)$ such that $h'_0 \circ H\ell_0 = f \circ h_0$. Define $g_0 :\equiv k_{a'_0,h'_0} \circ F\ell_0 \circ (k_{a_0,h_0})^{-1}$.

Now for any a, h, a', h', and ℓ such that $(h' \circ H\ell = f \circ h)$, we have $h^{-1} \circ h_0 : Ha_0 \cong Ha$, hence there is a unique $m : a_0 \cong a$ with $Hm = h^{-1} \circ h_0$ and hence $h \circ Hm = h_0$. Similarly, we have a unique $m' : a'_0 \cong a'$ with $h' \circ Hm' = h'_0$. Now by (iii), we have $k_{a,h} \circ Fm = k_{a_0,h_0}$ and $k_{a',h'} \circ Fm' = k_{a'_0,h'_0}$. We also have

$$\begin{aligned} Hm' \circ H\ell_0 &= (h')^{-1} \circ h'_0 \circ H\ell_0 \\ &= (h')^{-1} \circ f \circ h_0 \\ &= (h')^{-1} \circ f \circ h \circ h^{-1} \circ h_0 \\ &= H\ell \circ Hm \end{aligned}$$

and hence $m' \circ \ell_0 = \ell \circ m$ since H is fully faithful. Finally, we can compute

$$\begin{aligned} g_0 \circ k_{a,h} &= k_{a'_0,h'_0} \circ F\ell_0 \circ (k_{a_0,h_0})^{-1} \circ k_{a,h} \\ &= k_{a'_0,h'_0} \circ F\ell_0 \circ Fm^{-1} \\ &= k_{a'_0,h'_0} \circ (Fm')^{-1} \circ F\ell \\ &= k_{a',h'} \circ F\ell. \end{aligned}$$

This completes the proof that Y_f is inhabited. To show it is contractible, since hom-sets are sets, it suffices to take another $g_1 : \hom_C(Gb, Gb')$ satisfying (v) and show $g_0 = g_1$. However, we still have our specified $a_0, h_0, a'_0, h'_0, \ell_0$ around, and (v) implies both g_0 and g_1 must be equal to $k_{a'_0,h'_0} \circ F\ell_0 \circ (k_{a_0,h_0})^{-1}$.

This completes the proof that Y_f is contractible for each $b, b' : B$ and $f : \hom_B(b, b')$. Therefore, there is a function assigning to each such f its unique inhabitant; denote this function $G_{b,b'} : \hom_B(b, b') \to \hom_C(Gb, Gb')$. The proof that G is a functor is straightforward; in each case we can choose a, h and apply (v).

Finally, for any $a_0 : A$, defining $c :\equiv Fa_0$ and $k_{a,h} :\equiv Fg$, where $g : \hom_A(a, a_0)$ is the unique isomorphism with $Hg = h$, gives an element of X_{Ha_0}. Thus, it is equal to the specified one; hence $GHa = Fa$. Similarly, for $f : \hom_A(a_0, a_0')$ we can define an element of Y_{Hf} by transporting along these equalities, which must therefore be equal to the specified one. Hence, we have $GH = F$, and thus $GH \cong F$ as desired. □

Therefore, if a precategory A admits a weak equivalence functor $A \to \widehat{A}$ into a category, then that is its "reflection" into categories: any functor from A into a category will factor essentially uniquely through $\widehat{A}$. We now give two constructions of such a weak equivalence.

Theorem 9.9.5. *For any precategory A, there is a category $\widehat{A}$ and a weak equivalence $A \to \widehat{A}$.*

First proof. Let $\widehat{A}_0 :\equiv \left\{ F : \mathcal{S}et^{A^{\mathrm{op}}} \;\middle|\; \exists (a : A).\, (\mathbf{y}a \cong F) \right\}$, with hom-sets inherited from $\mathcal{S}et^{A^{\mathrm{op}}}$. Then the inclusion $\widehat{A} \to \mathcal{S}et^{A^{\mathrm{op}}}$ is fully faithful and an embedding on objects. Since $\mathcal{S}et^{A^{\mathrm{op}}}$ is a category (by Theorem 9.2.5, since $\mathcal{S}et$ is so by univalence), $\widehat{A}$ is also a category.

Let $A \to \widehat{A}$ be the Yoneda embedding. This is fully faithful by Corollary 9.5.6, and essentially surjective by definition of $\widehat{A}_0$. Thus it is a weak equivalence. □

This proof is very slick, but it has the drawback that it increases universe level. If A is a category in a universe $\mathcal{U}$, then in this proof $\mathcal{S}et$ must be at least as large as $\mathcal{S}et_{\mathcal{U}}$. Then $\mathcal{S}et_{\mathcal{U}}$ and $(\mathcal{S}et_{\mathcal{U}})^{A^{\mathrm{op}}}$ are not themselves categories in $\mathcal{U}$, but only in a higher universe, and *a priori* the same is true of $\widehat{A}$. One could imagine a resizing axiom that could deal with this, but it is also possible to give a direct construction using higher inductive types.

Second proof. We define a higher inductive type $\widehat{A}_0$ with the following constructors:

- A function $i : A_0 \to \widehat{A}_0$.
- For each $a, b : A$ and $e : a \cong b$, an equality $je : ia = ib$.
- For each $a : A$, an equality $j(1_a) = \mathsf{refl}_{ia}$.
- For each $(a, b, c : A)$, $(f : a \cong b)$, and $(g : b \cong c)$, an equality $j(g \circ f) = j(f) \cdot j(g)$.
- 1-truncation: for all $x, y : \widehat{A}_0$ and $p, q : x = y$ and $r, s : p = q$, an equality $r = s$.

Note that for any $a, b : A$ and $p : a = b$, we have $j(\mathsf{idtoiso}(p)) = i(p)$. This follows by path induction on p and the third constructor.

The type $\widehat{A}_0$ will be the type of objects of $\widehat{A}$; we now build all the rest of the structure. (The following proof is of the sort that can benefit a lot from the help of a computer proof assistant: it is wide and shallow with many short cases to consider, and a large part of the work consists of writing down what needs to be checked.)

Step 1: We define a family $\hom_{\widehat{A}} : \widehat{A}_0 \to \widehat{A}_0 \to \mathsf{Set}$ by double induction on $\widehat{A}_0$. Since Set is a 1-type, we can ignore the 1-truncation constructor. When x and y are of the form ia and ib, we take $\hom_{\widehat{A}}(ia, ib) :\equiv \hom_A(a, b)$. It remains to consider all the other possible pairs of constructors.

Let us keep $x = ia$ fixed at first. If y varies along the identity $je : ib = ib'$, for some $e : b \cong b'$, we require an identity $\hom_A(a, b) = \hom_A(a, b')$. By univalence, it suffices to give an

equivalence $\hom_A(a,b) \simeq \hom_A(a,b')$. We take this to be the function $(e \circ -) : \hom_A(a,b) \to \hom_A(a,b')$. To see that this is an equivalence, we give its inverse as $(e^{-1} \circ -)$, with witnesses to inversion coming from the fact that e^{-1} is the inverse of e in A.

As y varies along the identity $j(1_b) = \mathsf{refl}_{ib}$, we require an identity $(1_b \circ -) = \mathsf{refl}_{\hom_A(a,b)}$; this follows from the identity axiom $1_b \circ g = g$ of a precategory. Similarly, as y varies along the identity $j(g \circ f) = j(f) \cdot j(g)$, we require an identity $((g \circ f) \circ -) = (g \circ (f \circ -))$, which follows from associativity.

Now we consider the other constructors for x. Say that x varies along the identity $j(e) : ia = ia'$, for some $e : a \cong a'$; we again must deal with all the constructors for y. If y is ib, then we require an identity $\hom_A(a,b) = \hom_A(a',b)$. By univalence, this may come from an equivalence, and for this we can use $(- \circ e^{-1})$, with inverse $(- \circ e)$.

Still with x varying along $j(e)$, suppose now that y also varies along $j(f)$ for some $f : b \cong b'$. Then we need to know that the two concatenated identities

$$\hom_A(a,b) = \hom_A(a',b) = \hom_A(a',b') \qquad \text{and}$$
$$\hom_A(a,b) = \hom_A(a,b') = \hom_A(a',b')$$

are identical. This follows from associativity: $(f \circ -) \circ e^{-1} = f \circ (- \circ e^{-1})$. The other two constructors for y are trivial, since they are 2-fold equalities in sets.

For the next two constructors of x, all but the first constructor for y is likewise trivial. When x varies along $j(1_a) = \mathsf{refl}_{ia}$ and y is ib, we use the identity axiom again. Similarly, when x varies along $j(g \circ f) = j(f) \cdot j(g)$, we use associativity again. This completes the construction of $\hom_{\widehat{A}} : \widehat{A}_0 \to \widehat{A}_0 \to \mathsf{Set}$.

Step 2: We give the precategory structure on $\widehat{A}$, always by induction on $\widehat{A}_0$. We are now eliminating into sets (the hom-sets of $\widehat{A}$), so all but the first two constructors are trivial to deal with.

For identities, if x is ia then we have $\hom_{\widehat{A}}(x,x) \equiv \hom_A(a,a)$ and we define $1_x :\equiv 1_{ia}$. If x varies along je for $e : a \cong a'$, we must show that $\mathsf{transport}^{x \mapsto \hom_{\widehat{A}}(x,x)}(je, 1_{ia}) = 1_{ia'}$. But by definition of $\hom_{\widehat{A}}$, transporting along je is given by composing with e and e^{-1}, and we have $e \circ 1_{ia} \circ e^{-1} = 1_{ia'}$.

For composition, if x, y, z are ia, ib, ic respectively, then $\hom_{\widehat{A}}$ reduces to $\hom_A$ and we can define composition in $\widehat{A}$ to be composition in A. And when x, y, or z varies along je, then we verify the following equalities:

$$e \circ (g \circ f) = (e \circ g) \circ f,$$
$$g \circ f = (g \circ e^{-1}) \circ (e \circ f),$$
$$(g \circ f) \circ e^{-1} = g \circ (f \circ e^{-1}).$$

Finally, the associativity and unitality axioms are mere propositions, so all constructors except the first are trivial. But in that case, we have the corresponding axioms in A.

Step 3: We show that $\widehat{A}$ is a category. That is, we must show that for all $x, y : \widehat{A}$, the function $\mathsf{idtoiso} : (x = y) \to (x \cong y)$ is an equivalence. First we define, for all $x, y : \widehat{A}$, a function $k_{x,y} : (x \cong y) \to (x = y)$ by induction. As before, since our goal is a set, it suffices to deal with the first two constructors.

When x and y are ia and ib respectively, we have $\hom_{\widehat{A}}(ia, ib) \equiv \hom_A(a,b)$, with composition and identities inherited as well, so that $(ia \cong ib)$ is equivalent to $(a \cong b)$. But now we have the constructor $j : (a \cong b) \to (ia = ib)$.

Next, if y varies along $j(e)$ for some $e : b \cong b'$, we must show that for $f : a \cong b$ we have $j(j(e)_*(f)) = j(f) \cdot j(e)$. But by definition of $\hom_{\widehat{A}}$ on equalities, transporting along $j(e)$ is equivalent to post-composing with e, so this equality follows from the last constructor of $\widehat{A}_0$. The remaining case when x varies along $j(e)$ for $e : a \cong a'$ is similar. This completes the definition of $k : \prod_{(x,y:\widehat{A}_0)}(x \cong y) \to (x = y)$.

Now one thing we must show is that if $p : x = y$, then $k(\mathsf{idtoiso}(p)) = p$. By induction on p, we may assume it is refl_x, and hence $\mathsf{idtoiso}(p) \equiv 1_x$. Now we argue by induction on $x : \widehat{A}_0$, and since our goal is a mere proposition (since $\widehat{A}_0$ is a 1-type), all constructors except the first are trivial. But if x is ia, then $k(1_{ia}) \equiv j(1_a)$, which is equal to refl_{ia} by the third constructor of $\widehat{A}_0$.

To complete the proof that $\widehat{A}$ is a category, we must show that if $f : x \cong y$, then $\mathsf{idtoiso}(k(f)) = f$. By induction we may assume that x and y are ia and ib respectively, in which case f must arise from an isomorphism $g : a \cong b$ and we have $k(f) \equiv j(g)$. However, for any p we have $\mathsf{idtoiso}(p) = p_*(1)$, so in particular $\mathsf{idtoiso}(j(g)) = j(g)_*(1_{ia})$. And by definition of $\hom_{\widehat{A}}$ on equalities, this is given by composing 1_{ia} with the equivalence g, hence is equal to g.

Note the similarity of this step to the encode-decode method used in §§2.12 and 2.13 and Chapter 8. Once again we are characterizing the identity types of a higher inductive type (here, $\widehat{A}_0$) by defining recursively a family of codes (here, $(x, y) \mapsto (x \cong y)$) and encoding and decoding functions by induction on $\widehat{A}_0$ and on paths.

Step 4: We define a weak equivalence $I : A \to \widehat{A}$. We take $I_0 :\equiv i : A_0 \to \widehat{A}_0$, and by construction of $\hom_{\widehat{A}}$ we have functions $I_{a,b} : \hom_A(a, b) \to \hom_{\widehat{A}}(Ia, Ib)$ forming a functor $I : A \to \widehat{A}$. This functor is fully faithful by construction, so it remains to show it is essentially surjective. That is, for all $x : \widehat{A}$ we want there to merely exist an $a : A$ such that $Ia \cong x$. As always, we argue by induction on x, and since the goal is a mere proposition, all but the first constructor are trivial. But if x is ia, then of course we have $a : A$ and $Ia \equiv ia$, hence $Ia \cong ia$. (Note that if we were trying to prove I to be *split* essentially surjective, we would be stuck, because we know nothing about equalities in A_0 and thus have no way to deal with any further constructors.) □

We call the construction $A \mapsto \widehat{A}$ the **Rezk completion**, although there is also an argument (coming from higher topos semantics) for calling it the **stack completion**.

We have seen that most precategories arising in practice are categories, since they are constructed from $\mathcal{S}et$, which is a category by the univalence axiom. However, there are a few cases in which the Rezk completion is necessary to obtain a category.

Example 9.9.6. Recall from Example 9.1.17 that for any type X there is a pregroupoid with X as its type of objects and $\hom(x, y) :\equiv \|x = y\|_0$. Its Rezk completion is the *fundamental groupoid* of X. Recalling that groupoids are equivalent to 1-types, it is not hard to identify this groupoid with $\|X\|_1$.

Example 9.9.7. Recall from Example 9.1.18 that there is a precategory whose type of objects is $\mathcal{U}$ and with $\hom(X, Y) :\equiv \|X \to Y\|_0$. Its Rezk completion may be called the **homotopy category of types**. Its type of objects can be identified with $\|\mathcal{U}\|_1$ (see Exercise 9.9).

The Rezk completion also allows us to show that the notion of "category" is determined by the notion of "weak equivalence of precategories". Thus, insofar as the latter is inevitable, so is the former.

Theorem 9.9.8. *A precategory C is a category if and only if for every weak equivalence of precategories $H : A \to B$, the induced functor $(- \circ H) : C^B \to C^A$ is an isomorphism of precategories.*

Proof. "Only if" is Theorem 9.9.4. In the other direction, let H be $I : A \to \widehat{A}$. Then since $(- \circ I)_0$ is an equivalence, there exists $R : \widehat{A} \to A$ such that $RI = 1_A$. Hence $IRI = I$, but again since $(- \circ I)_0$ is an equivalence, this implies $IR = 1_{\widehat{A}}$. By Lemma 9.4.9(iii), I is an isomorphism of precategories. But then since $\widehat{A}$ is a category, so is A. □

Notes

The original definition of categories, of course, was in set-theoretic foundations, so that the collection of objects of a category formed a set (or, for large categories, a class). Over time, it became clear that all "category-theoretic" properties of objects were invariant under isomorphism, and that equality of objects in a category was not usually a very useful notion. Numerous authors [Bla79, Fre76, Mak95, Mak01] discovered that a dependently typed logic enabled formulating the definition of category without invoking any notion of equality for objects, and that the statements provable in this logic are precisely the "category-theoretic" ones that are invariant under isomorphism.

Although most of category theory appears to be invariant under isomorphism of objects and under equivalence of categories, there are some interesting exceptions, which have led to philosophical discussions about what it means to be "category-theoretic". For instance, Example 9.6.3 was brought up by Peter May on the categories mailing list in May 2010, as a case where it matters that two categories (defined as usual in set theory) are isomorphic rather than only equivalent. The case of †-categories was also somewhat confounding to those advocating an isomorphism-invariant version of category theory, since the "correct" notion of sameness between objects of a †-category is not ordinary isomorphism but *unitary* isomorphism.

Categories satisfying the "saturation" or "univalence" principle as in Definition 9.1.6 were first considered by Hofmann and Streicher [HS98]. The condition then occurred independently to Voevodsky, Shulman, and perhaps others around the same time several years later, and was formalized by Ahrens and Kapulkin [AKS13]. This framework puts all the above examples in a unified context: some precategories are categories, others are strict categories, and so on. A general theorem that "isomorphism implies equality" for a large class of algebraic structures (assuming the univalence axiom) was proven by Coquand and Danielsson; the formulation of the structure identity principle in §9.8 is due to Aczel.

Independently of philosophical considerations about category theory, Rezk [Rez01] discovered that when defining a notion of $(\infty,1)$-category, it was very convenient to use not merely a *set* of objects with spaces of morphisms between them, but a *space* of objects incorporating all the equivalences and homotopies between them. This yields a very well-behaved sort of model for $(\infty,1)$-categories as particular simplicial spaces, which Rezk called *complete Segal spaces*. One especially good aspect of this model is the analogue of Lemma 9.4.14: a map of complete Segal spaces is an equivalence just when it is a levelwise equivalence of simplicial spaces.

When interpreted in Voevodsky's simplicial set model of univalent foundations, our precategories are similar to a truncated analogue of Rezk's "Segal spaces", while our categories correspond to his "complete Segal spaces". Strict categories correspond instead to (a weakened and truncated version of) what are called "Segal categories". It is known that Segal categories and complete Segal spaces are equivalent models for $(\infty,1)$-categories (see e.g. [Ber09]), so that in the simplicial set model, categories and strict categories yield "equivalent" category theories—although as we have seen, the former still have many advantages. However, in the more general categorical semantics of a higher topos, a strict category corresponds to an internal category (in the traditional sense) in the corresponding 1-topos of sheaves, while a category corresponds to a *stack*. The latter are generally a more appropriate sort of "category" relative to a topos.

In Rezk's context, what we have called the "Rezk completion" corresponds to fibrant replacement in the model category for complete Segal spaces. Since this is built using a transfinite induction argument, it most closely matches our second construction as a higher inductive type. However, in higher topos models of homotopy type theory, the Rezk completion corresponds

to *stack completion*, which can be constructed either with a transfinite induction [JT91] or using a Yoneda embedding [Bun79].

Exercises

Exercise 9.1. For a precategory A and $a : A$, define the **slice precategory** A/a. Show that if A is a category, so is A/a.

Exercise 9.2. For any set X, prove that the slice category $\mathcal{S}et/X$ is equivalent to the functor category $\mathcal{S}et^X$, where in the latter case we regard X as a discrete category.

Exercise 9.3. Prove that a functor is an equivalence of categories if and only if it is a *right* adjoint whose unit and counit are isomorphisms.

Exercise 9.4. Define the notion of **pre-2-category**. Show that precategories, functors, and natural transformations as defined in §9.2 form a pre-2-category. Similarly, define a **pre-bicategory** by replacing the equalities (such as those in Lemmas 9.2.9 and 9.2.11) with natural isomorphisms satisfying analogous coherence conditions. Define a function from pre-2-categories to pre-bicategories, and show that it becomes an equivalence when restricted and corestricted to those whose hom-precategories are categories.

Exercise 9.5. Define a **2-category** to be a pre-2-category satisfying a condition analogous to that of Definition 9.1.6. Verify that the pre-2-category of categories $\mathcal{C}at$ is a 2-category. How much of this chapter can be done internally to an arbitrary 2-category?

Exercise 9.6. Define a 2-category whose objects are 1-types, whose morphisms are functions, and whose 2-morphisms are homotopies. Prove that it is equivalent, in an appropriate sense, to the full sub-2-category of $\mathcal{C}at$ spanned by the *groupoids* (categories in which every arrow is an isomorphism).

Exercise 9.7. Recall that a *strict category* is a precategory whose type of objects is a set. Prove that the pre-2-category of strict categories is equivalent to the following pre-2-category.

- Its objects are categories A equipped with a surjection $p_A : A'_0 \to A_0$, where A'_0 is a set.
- Its morphisms are functors $F : A \to B$ equipped with a function $F'_0 : A'_0 \to B'_0$ such that $p_B \circ F'_0 = F_0 \circ p_A$.
- Its 2-morphisms are simply natural transformations.

Exercise 9.8. Define the pre-2-category of †-categories, which has †-structures on its hom-precategories. Show that two †-categories are equal precisely when they are "unitarily equivalent" in a suitable sense.

Exercise 9.9. Prove that a function $X \to Y$ is an equivalence if and only if its image in the homotopy category of Example 9.9.7 is an isomorphism. Show that the type of objects of this category is $\|\mathcal{U}\|_1$.

Exercise 9.10. Construct the †-Rezk completion of a †-precategory into a †-category, and give it an appropriate universal property.

Exercise 9.11. Using fundamental (pre)groupoids from Examples 9.1.17 and 9.9.6 and the Rezk completion from §9.9, give a different proof of van Kampen's theorem (§8.7).

Exercise 9.12. Let X and Y be sets and $p : Y \to X$ a surjection.

(i) Define, for any precategory A, the category $\mathrm{Desc}(A, p)$ of **descent data** in A relative to p.

(ii) Show that any precategory A is a **prestack** for p, i.e. the canonical functor $A^X \to \mathrm{Desc}(A, p)$ is fully faithful.

(iii) Show that if A is a category, then it is a **stack** for p, i.e. $A^X \to \mathrm{Desc}(A, p)$ is an equivalence.

(iv) Show that the statement "every strict category is a stack for every surjection of sets" is equivalent to the axiom of choice.

Chapter 10

Set theory

Our conception of sets as types with particularly simple homotopical character, cf. §3.1, is quite different from the sets of Zermelo–Fraenkel set theory, which form a cumulative hierarchy with an intricate nested membership structure. For many mathematical purposes, the homotopy-theoretic sets are just as good as the Zermelo–Fraenkel ones, but there are important differences.

We begin this chapter in §10.1 by showing that the category $\mathcal{S}et$ has (most of) the usual properties of the category of sets. In constructive, predicative, univalent foundations, it is a "ΠW-pretopos"; whereas if we assume propositional resizing (§3.5) it is an elementary topos, and if we assume LEM and AC then it is a model of Lawvere's *Elementary Theory of the Category of Sets*. This is sufficient to ensure that the sets in homotopy type theory behave like sets as used by most mathematicians outside of set theory.

In the rest of the chapter, we investigate some subjects that traditionally belong to "set theory". In §§10.2–10.4 we study cardinal and ordinal numbers. These are traditionally defined in set theory using the global membership relation, but we will see that the univalence axiom enables an equally convenient, more "structural" approach.

Finally, in §10.5 we consider the possibility of constructing *inside* of homotopy type theory a cumulative hierarchy of sets, equipped with a binary membership relation akin to that of Zermelo–Fraenkel set theory. This combines higher inductive types with ideas from the field of algebraic set theory.

In this chapter we will often use the traditional logical notation described in §3.7. In addition to the basic theory of Chapters 2 and 3, we use higher inductive types for colimits and quotients as in §§6.8 and 6.10, as well as some of the theory of truncation from Chapter 7, particularly the factorization system of §7.6 in the case $n = -1$. In §10.3 we use an inductive family (§5.7) to describe well-foundedness, and in §10.5 we use a more complicated higher inductive type to present the cumulative hierarchy.

10.1 The category of sets

Recall that in Chapter 9 we defined the category $\mathcal{S}et$ to consist of all 0-types (in some universe $\mathcal{U}$) and maps between them, and observed that it is a category (not just a precategory). We consider successively the levels of structure which $\mathcal{S}et$ possesses.

10.1.1 Limits and colimits

Since sets are closed under products, the universal property of products in Theorem 2.15.2 shows immediately that $\mathcal{S}et$ has finite products. In fact, infinite products follow just as easily from the equivalence

$$\Big(X \to \prod_{a:A} B(a)\Big) \simeq \Big(\prod_{a:A} (X \to B(a))\Big).$$

And we saw in Exercise 2.11 that the pullback of $f : A \to C$ and $g : B \to C$ can be defined as $\sum_{(a:A)} \sum_{(b:B)} f(a) = g(b)$; this is a set if A, B, C are and inherits the correct universal property. Thus, $\mathcal{S}et$ is a *complete* category in the obvious sense.

Since sets are closed under $+$ and contain $\mathbf{0}$, $\mathcal{S}et$ has finite coproducts. Similarly, since $\sum_{(a:A)} B(a)$ is a set whenever A and each $B(a)$ are, it yields a coproduct of the family B in $\mathcal{S}et$. Finally, we showed in §7.4 that pushouts exist in n-types, which includes $\mathcal{S}et$ in particular. Thus, $\mathcal{S}et$ is also *cocomplete*.

10.1.2 Images

Next, we show that $\mathcal{S}et$ is a **regular category**, i.e.:

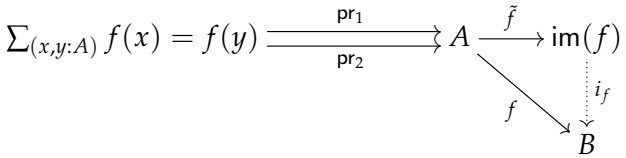
(i) $\mathcal{S}et$ is finitely complete.
(ii) The kernel pair $\mathsf{pr}_1, \mathsf{pr}_2 : (\sum_{(x,y:A)} f(x) = f(y)) \to A$ of any function $f : A \to B$ has a coequalizer.
(iii) Pullbacks of regular epimorphisms are again regular epimorphisms.

Recall that a **regular epimorphism** is a morphism that is the coequalizer of *some* pair of maps. Thus in (iii) the pullback of a coequalizer is required to again be a coequalizer, but not necessarily of the pulled-back pair.

The obvious candidate for the coequalizer of the kernel pair of $f : A \to B$ is the *image* of f, as defined in §7.6. Recall that we defined $\mathsf{im}(f) :\equiv \sum_{(b:B)} \big\|\mathsf{fib}_f(b)\big\|$, with functions $\tilde{f} : A \to \mathsf{im}(f)$ and $i_f : \mathsf{im}(f) \to B$ defined by

$$\tilde{f} :\equiv \lambda a.\, \Big(f(a),\, \Big|(a, \mathsf{refl}_{f(a)})\Big|\Big)$$
$$i_f :\equiv \mathsf{pr}_1$$

fitting into a diagram:



Recall that a function $f : A \to B$ is called *surjective* if $\forall(b : B).\, \big\|\mathsf{fib}_f(b)\big\|$, or equivalently $\forall(b : B).\, \exists(a : A).\, f(a) = b$. We have also said that a function $f : A \to B$ between sets is called *injective* if $\forall(a, a' : A).\, (f(a) = f(a')) \Rightarrow (a = a')$, or equivalently if each of its fibers is a mere proposition. Since these are the (-1)-connected and (-1)-truncated maps in the sense of Chapter 7, the general theory there implies that $\tilde{f}$ above is surjective and i_f is injective, and that this factorization is stable under pullback.

We now identify surjectivity and injectivity with the appropriate category-theoretic notions. First we observe that categorical monomorphisms and epimorphisms have a slightly stronger equivalent formulation.

Lemma 10.1.1. *For a morphism $f : \hom_A(a,b)$ in a category A, the following are equivalent.*

(i) *f is a **monomorphism**: for all $x : A$ and $g,h : \hom_A(x,a)$, if $f \circ g = f \circ h$ then $g = h$.*
(ii) *(If A has pullbacks) the diagonal map $a \to a \times_b a$ is an isomorphism.*
(iii) *For all $x : A$ and $k : \hom_A(x,b)$, the type $\sum_{(h:\hom_A(x,a))}(k = f \circ h)$ is a mere proposition.*
(iv) *For all $x : A$ and $g : \hom_A(x,a)$, the type $\sum_{(h:\hom_A(x,a))}(f \circ g = f \circ h)$ is contractible.*

Proof. The equivalence of conditions (i) and (ii) is standard category theory. Now consider the function $(f \circ -) : \hom_A(x,a) \to \hom_A(x,b)$ between sets. Condition (i) says that it is injective, while (iii) says that its fibers are mere propositions; hence they are equivalent. And (iii) implies (iv) by taking $k :\equiv f \circ g$ and recalling that an inhabited mere proposition is contractible. Finally, (iv) implies (i) since if $p : f \circ g = f \circ h$, then (g, refl) and (h,p) both inhabit the type in (iv), hence are equal and so $g = h$. □

Lemma 10.1.2. *A function $f : A \to B$ between sets is injective if and only if it is a monomorphism in $\mathcal{S}et$.*

Proof. Left to the reader. □

Of course, an **epimorphism** is a monomorphism in the opposite category. We now show that in $\mathcal{S}et$, the epimorphisms are precisely the surjections, and also precisely the coequalizers (regular epimorphisms).

The coequalizer of a pair of maps $f,g : A \to B$ in $\mathcal{S}et$ is defined as the 0-truncation of a general (homotopy) coequalizer. For clarity, we may call this the **set-coequalizer**. It is convenient to express its universal property as follows.

Lemma 10.1.3. *Let $f,g : A \to B$ be functions between sets A and B. The set-coequalizer $c_{f,g} : B \to Q$ has the property that, for any set C and any $h : B \to C$ with $h \circ f = h \circ g$, the type*

$$\sum_{k:Q\to C} (k \circ c_{f,g} = h)$$

is contractible.

Lemma 10.1.4. *For any function $f : A \to B$ between sets, the following are equivalent:*

(i) *f is an epimorphism.*
(ii) *Consider the pushout diagram*

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ \downarrow & & \downarrow{\scriptstyle \iota} \\ \mathbf{1} & \xrightarrow[t]{} & C_f \end{array}$$

in $\mathcal{S}et$ defining the mapping cone. Then the type C_f is contractible.
(iii) *f is surjective.*

Proof. Let $f : A \to B$ be a function between sets, and suppose it to be an epimorphism; we show C_f is contractible. The constructor $\mathbf{1} \to C_f$ of C_f gives us an element $t : C_f$. We have to show that

$$\prod_{x:C_f} x = t.$$

Note that $x = t$ is a mere proposition, hence we can use induction on C_f. Of course when x is t we have $\mathsf{refl}_t : t = t$, so it suffices to find

$$I_0 : \prod_{b:B} \iota(b) = t$$
$$I_1 : \prod_{a:A} \alpha_1(a)^{-1} \cdot I_0(f(a)) = \mathsf{refl}_t$$

where $\iota : B \to C_f$ and $\alpha_1 : \prod_{(a:A)} \iota(f(a)) = t$ are the other constructors of C_f. Note that α_1 is a homotopy from $\iota \circ f$ to $\mathsf{const}_t \circ f$, so we find the elements

$$(\iota, \mathsf{refl}_{\iota\circ f}), (\mathsf{const}_t, \alpha_1) : \sum_{h:B\to C_f} \iota \circ f \sim h \circ f.$$

By the dual of Lemma 10.1.1(iv) (and function extensionality), there is a path

$$\gamma : (\iota, \mathsf{refl}_{\iota\circ f}) = (\mathsf{const}_t, \alpha_1).$$

Hence, we may define $I_0(b) :\equiv \mathsf{happly}(\mathsf{ap}_{\mathsf{pr}_1}(\gamma), b) : \iota(b) = t$. We also have

$$\mathsf{ap}_{\mathsf{pr}_2}(\gamma) : \mathsf{ap}_{\mathsf{pr}_1}(\gamma)_*\big(\mathsf{refl}_{\iota\circ f}\big) = \alpha_1.$$

This transport involves precomposition with f, which commutes with happly. Thus, from transport in path types we obtain $I_0(f(a)) = \alpha_1(a)$ for any $a : A$, which gives us I_1.

Now suppose C_f is contractible; we show f is surjective. We first construct a type family $P : C_f \to \mathsf{Prop}$ by recursion on C_f, which is valid since Prop is a set. On the point constructors, we define

$$P(t) :\equiv \mathbf{1}$$
$$P(\iota(b)) :\equiv \big\|\mathsf{fib}_f(b)\big\|.$$

To complete the construction of P, it remains to give a path $\big\|\mathsf{fib}_f(f(a))\big\| =_{\mathsf{Prop}} \mathbf{1}$ for all $a : A$. However, $\big\|\mathsf{fib}_f(f(a))\big\|$ is inhabited by $(f(a), \mathsf{refl}_{f(a)})$. Since it is a mere proposition, this means it is contractible — and thus equivalent, hence equal, to $\mathbf{1}$. This completes the definition of P. Now, since C_f is assumed to be contractible, it follows that $P(x)$ is equivalent to $P(t)$ for any $x : C_f$. In particular, $P(\iota(b)) \equiv \big\|\mathsf{fib}_f(b)\big\|$ is equivalent to $P(t) \equiv \mathbf{1}$ for each $b : B$, and hence contractible. Thus, f is surjective.

Finally, suppose $f : A \to B$ to be surjective, and consider a set C and two functions $g, h : B \to C$ with the property that $g \circ f = h \circ f$. Since f is assumed to be surjective, for all $b : B$ the type $\big\|\mathsf{fib}_f(b)\big\|$ is contractible. Thus we have the following equivalences:

$$\begin{aligned}
\prod_{b:B} (g(b) = h(b)) &\simeq \prod_{b:B} \Big(\big\|\mathsf{fib}_f(b)\big\| \to (g(b) = h(b))\Big) \\
&\simeq \prod_{b:B} \Big(\mathsf{fib}_f(b) \to (g(b) = h(b))\Big) \\
&\simeq \prod_{(b:B)} \prod_{(a:A)} \prod_{(p:f(a)=b)} g(b) = h(b) \\
&\simeq \prod_{a:A} g(f(a)) = h(f(a))
\end{aligned}$$

using on the second line the fact that $g(b) = h(b)$ is a mere proposition, since C is a set. But by assumption, there is an element of the latter type. □

Theorem 10.1.5. *The category $\mathcal{S}et$ is regular. Moreover, surjective functions between sets are regular epimorphisms.*

Proof. It is a standard lemma in category theory that a category is regular as soon as it admits finite limits and a pullback-stable orthogonal factorization system $(\mathcal{E}, \mathcal{M})$ with $\mathcal{M}$ the monomorphisms, in which case $\mathcal{E}$ consists automatically of the regular epimorphisms. (See e.g. [Joh02, A1.3.4].) The existence of the factorization system was proved in Theorem 7.6.6. □

Lemma 10.1.6. *Pullbacks of regular epis in $\mathcal{S}et$ are regular epis.*

Proof. We showed in Theorem 7.6.9 that pullbacks of n-connected functions are n-connected. By Theorem 10.1.5, it suffices to apply this when $n = -1$. □

One of the consequences of $\mathcal{S}et$ being a regular category is that we have an "image" operation on subsets. That is, given $f : A \to B$, any subset $P : \mathcal{P}(A)$ (i.e. a predicate $P : A \to \mathsf{Prop}$) has an **image** which is a subset of B. This can be defined directly as $\{\, y : B \mid \exists (x : A).\, f(x) = y \wedge P(x) \,\}$, or indirectly as the image (in the previous sense) of the composite function

$$\{\, x : A \mid P(x) \,\} \to A \xrightarrow{f} B.$$

We will also sometimes use the common notation $\{\, f(x) \mid P(x) \,\}$ for the image of P.

10.1.3 Quotients

Now that we know that $\mathcal{S}et$ is regular, to show that $\mathcal{S}et$ is exact, we need to show that every equivalence relation is effective. In other words, given an equivalence relation $R : A \to A \to \mathsf{Prop}$, there is a coequalizer c_R of the pair $\mathsf{pr}_1, \mathsf{pr}_2 : \sum_{(x,y:A)} R(x,y) \to A$ and, moreover, the pr_1 and pr_2 form the kernel pair of c_R.

We have already seen, in §6.10, two general ways to construct the quotient of a set by an equivalence relation $R : A \to A \to \mathsf{Prop}$. The first can be described as the set-coequalizer of the two projections

$$\mathsf{pr}_1, \mathsf{pr}_2 : \Big(\sum_{x,y:A} R(x,y) \Big) \to A.$$

The important property of such a quotient is the following.

Definition 10.1.7. A relation $R : A \to A \to \mathsf{Prop}$ is said to be **effective** if the square

$$\begin{array}{ccc} \sum_{(x,y:A)} R(x,y) & \xrightarrow{\mathsf{pr}_1} & A \\ {\scriptstyle \mathsf{pr}_2}\downarrow & & \downarrow{\scriptstyle c_R} \\ A & \xrightarrow[c_R]{} & A/R \end{array}$$

is a pullback.

Since the standard pullback of c_R and itself is $\sum_{(x,y:A)} (c_R(x) = c_R(y))$, by Theorem 4.7.7 this is equivalent to asking that the canonical transformation $\prod_{(x,y:A)} R(x,y) \to (c_R(x) = c_R(y))$ be a fiberwise equivalence.

Lemma 10.1.8. *Suppose (A, R) is an equivalence relation. Then there is an equivalence*

$$(c_R(x) = c_R(y)) \simeq R(x,y)$$

for any $x, y : A$. In other words, equivalence relations are effective.

Proof. We begin by extending R to a relation $\widetilde{R} : A/R \to A/R \to \mathsf{Prop}$, which we will then show is equivalent to the identity type on A/R. We define $\widetilde{R}$ by double induction on A/R (note that Prop is a set by univalence for mere propositions). We define $\widetilde{R}(c_R(x), c_R(y)) :\equiv R(x,y)$. For $r : R(x,x')$ and $s : R(y,y')$, the transitivity and symmetry of R gives an equivalence from $R(x,y)$ to $R(x',y')$. This completes the definition of $\widetilde{R}$.

It remains to show that $\widetilde{R}(w,w') \simeq (w = w')$ for every $w, w' : A/R$. The direction $(w = w') \to \widetilde{R}(w,w')$ follows by transport once we show that $\widetilde{R}$ is reflexive, which is an easy induction. The other direction $\widetilde{R}(w,w') \to (w = w')$ is a mere proposition, so since $c_R : A \to A/R$ is surjective, it suffices to assume that w and w' are of the form $c_R(x)$ and $c_R(y)$. But in this case, we have the canonical map $\widetilde{R}(c_R(x), c_R(y)) :\equiv R(x,y) \to (c_R(x) = c_R(y))$. (Note again the appearance of the encode-decode method.) □

The second construction of quotients is as the set of equivalence classes of R (a subset of its power set):

$$A /\!\!/ R :\equiv \{\, P : A \to \mathsf{Prop} \mid P \text{ is an equivalence class of } R \,\}.$$

This requires propositional resizing in order to remain in the same universe as A and R.

Note that if we regard R as a function from A to $A \to \mathsf{Prop}$, then $A /\!\!/ R$ is equivalent to $\mathsf{im}(R)$, as constructed in §10.1.2. Now in Theorem 10.1.5 we have shown that images are coequalizers. In particular, we immediately get the coequalizer diagram

$$\textstyle\sum_{(x,y:A)} R(x) = R(y) \;\underset{\mathsf{pr}_2}{\overset{\mathsf{pr}_1}{\rightrightarrows}}\; A \longrightarrow A /\!\!/ R.$$

We can use this to give an alternative proof that any equivalence relation is effective and that the two definitions of quotients agree.

Theorem 10.1.9. *For any function $f : A \to B$ between any two sets, the relation $\ker(f) : A \to A \to \mathsf{Prop}$ given by $\ker(f,x,y) :\equiv (f(x) = f(y))$ is effective.*

Proof. We will use that $\mathsf{im}(f)$ is the coequalizer of $\mathsf{pr}_1, \mathsf{pr}_2 : (\sum_{(x,y:A)} f(x) = f(y)) \to A$. Note that the kernel pair of the function

$$c_f :\equiv \lambda a.\, \Big(f(a), \Big\|(a, \mathsf{refl}_{f(a)})\Big\|\Big) : A \to \mathsf{im}(f)$$

consists of the two projections

$$\mathsf{pr}_1, \mathsf{pr}_2 : \Big(\sum_{x,y:A} c_f(x) = c_f(y)\Big) \to A.$$

For any $x, y : A$, we have equivalences

$$\begin{aligned}(c_f(x) = c_f(y)) &\simeq \Big(\sum_{p : f(x) = f(y)} p_*\Big(\Big\|(x, \mathsf{refl}_{f(x)})\Big\|\Big) = \Big\|(y, \mathsf{refl}_{f(x)})\Big\|\Big)\\ &\simeq (f(x) = f(y)),\end{aligned}$$

where the last equivalence holds because $\big\|\mathsf{fib}_f(b)\big\|$ is a mere proposition for any $b : B$. Therefore, we get that

$$\Big(\sum_{x,y:A} c_f(x) = c_f(y)\Big) \simeq \Big(\sum_{x,y:A} f(x) = f(y)\Big)$$

and hence we may conclude that $\ker f$ is an effective relation for any function f. □

Theorem 10.1.10. *Equivalence relations are effective and there is an equivalence $A/R \simeq A /\!\!/ R$.*

Proof. We need to analyze the coequalizer diagram

$$\textstyle\sum_{(x,y:A)} R(x) = R(y) \rightrightarrows A \longrightarrow A /\!\!/ R$$

By the univalence axiom, the type $R(x) = R(y)$ is equivalent to the type of homotopies from $R(x)$ to $R(y)$, which is equivalent to $\prod_{(z:A)} R(x,z) \simeq R(y,z)$. Since R is an equivalence relation, the latter space is equivalent to $R(x,y)$. To summarize, we get that $(R(x) = R(y)) \simeq R(x,y)$, so R is effective since it is equivalent to an effective relation. Also, the diagram

$$\textstyle\sum_{(x,y:A)} R(x,y) \rightrightarrows A \longrightarrow A /\!\!/ R.$$

is a coequalizer diagram. Since coequalizers are unique up to equivalence, it follows that $A/R \simeq A /\!\!/ R$. □

We finish this section by mentioning a possible third construction of the quotient of a set A by an equivalence relation R. Consider the precategory with objects A and hom-sets R; the type of objects of the Rezk completion (see §9.9) of this precategory will then be the quotient. The reader is invited to check the details.

10.1.4 $\mathcal{S}et$ is a ΠW-pretopos

The notion of a ΠW-*pretopos* — that is, a locally cartesian closed category with disjoint finite coproducts, effective equivalence relations, and initial algebras for polynomial endofunctors — is intended as a "predicative" notion of topos, i.e. a category of "predicative sets", which can serve the purpose for constructive mathematics that the usual category of sets does for classical mathematics.

Typically, in constructive type theory, one resorts to an external construction of "setoids" — an exact completion — to obtain a category with such closure properties. In particular, the well-behaved quotients are required for many constructions in mathematics that usually involve (non-constructive) power sets. It is noteworthy that univalent foundations provides these constructions *internally* (via higher inductive types), without requiring such external constructions. This represents a powerful advantage of our approach, as we shall see in subsequent examples.

Theorem 10.1.11. *The category $\mathcal{S}et$ is a ΠW-pretopos.*

Proof. We have an initial object $\mathbf{0}$ and finite, disjoint sums $A + B$. These are stable under pullback, simply because pullback has a right adjoint. Indeed, $\mathcal{S}et$ is locally cartesian closed, since for any map $f : A \to B$ between sets, the "fibrant replacement" $\sum_{(a:A)} f(a) = b$ is equivalent to A (over B), and we have dependent function types for the replacement. We've just shown that $\mathcal{S}et$ is regular (Theorem 10.1.5) and that quotients are effective (Lemma 10.1.8). We thus have a locally cartesian closed pretopos. Finally, since the n-types are closed under the formation of W-types by Exercise 7.3, and by Theorem 5.4.7 W-types are initial algebras for polynomial endofunctors, we see that $\mathcal{S}et$ is a ΠW-pretopos. □

One naturally wonders what, if anything, prevents $\mathcal{S}et$ from being an (elementary) topos? In addition to the structure already mentioned, a topos has a *subobject classifier*: a pointed object

classifying (equivalence classes of) monomorphisms. (In fact, in the presence of a subobject classifier, things become somewhat simpler: one merely needs cartesian closure in order to get the colimits.) In homotopy type theory, univalence implies that the type $\mathsf{Prop} :\equiv \sum_{(X:\mathcal{U})} \mathsf{isProp}(X)$ does classify monomorphisms (by an argument similar to §4.8), but in general it is as large as the ambient universe $\mathcal{U}$. Thus, it is a "set" in the sense of being a 0-type, but it is not "small" in the sense of being an object of $\mathcal{U}$, hence not an object of the category $\mathcal{S}et$. However, if we assume an appropriate form of propositional resizing (see §3.5), then we can find a small version of Prop, so that $\mathcal{S}et$ becomes an elementary topos.

Theorem 10.1.12. *If there is a type $\Omega : \mathcal{U}$ of all mere propositions, then the category $\mathcal{S}et_{\mathcal{U}}$ is an elementary topos.*

A sufficient condition for this is the law of excluded middle, in the "mere-propositional" form that we have called LEM; for then we have $\mathsf{Prop} = \mathbf{2}$, which *is* small, and which then also classifies all mere propositions. Moreover, in topos theory a well-known sufficient condition for LEM is the axiom of choice, which is of course often assumed as an axiom in classical set theory. In the next section, we briefly investigate the relation between these conditions in our setting.

10.1.5 The axiom of choice implies excluded middle

We begin with the following lemma.

Lemma 10.1.13. *If A is a mere proposition then its suspension $\Sigma(A)$ is a set, and A is equivalent to $\mathsf{N} =_{\Sigma(A)} \mathsf{S}$.*

Proof. To show that $\Sigma(A)$ is a set, we define a family $P : \Sigma(A) \to \Sigma(A) \to \mathcal{U}$ with the property that $P(x, y)$ is a mere proposition for each $x, y : \Sigma(A)$, and which is equivalent to its identity type $\mathsf{Id}_{\Sigma(A)}$. We make the following definitions:

$$\begin{aligned} P(\mathsf{N},\mathsf{N}) &:\equiv \mathbf{1} & P(\mathsf{S},\mathsf{N}) &:\equiv A \\ P(\mathsf{N},\mathsf{S}) &:\equiv A & P(\mathsf{S},\mathsf{S}) &:\equiv \mathbf{1}. \end{aligned}$$

We have to check that the definition preserves paths. Given any $a : A$, there is a meridian $\mathsf{merid}(a) : \mathsf{N} = \mathsf{S}$, so we should also have

$$P(\mathsf{N},\mathsf{N}) = P(\mathsf{N},\mathsf{S}) = P(\mathsf{S},\mathsf{N}) = P(\mathsf{S},\mathsf{S}).$$

But since A is inhabited by a, it is equivalent to $\mathbf{1}$, so we have

$$P(\mathsf{N},\mathsf{N}) \simeq P(\mathsf{N},\mathsf{S}) \simeq P(\mathsf{S},\mathsf{N}) \simeq P(\mathsf{S},\mathsf{S}).$$

The univalence axiom turns these into the desired equalities. Also, $P(x, y)$ is a mere proposition for all $x, y : \Sigma(A)$, which is proved by induction on x and y, and using the fact that being a mere proposition is a mere proposition.

Note that P is a reflexive relation. Therefore we may apply Theorem 7.2.2, so it suffices to construct $\tau : \prod_{(x,y:\Sigma(A))} P(x, y) \to (x = y)$. We do this by a double induction. When x is N, we define $\tau(\mathsf{N})$ by

$$\tau(\mathsf{N},\mathsf{N},u) :\equiv \mathsf{refl}_{\mathsf{N}} \qquad \text{and} \qquad \tau(\mathsf{N},\mathsf{S},a) :\equiv \mathsf{merid}(a).$$

If A is inhabited by a then $\mathsf{merid}(a) : \mathsf{N} = \mathsf{S}$ so we also need $\mathsf{merid}(a)_*(\tau(\mathsf{N},\mathsf{N})) = \tau(\mathsf{N},\mathsf{S})$. This we get by function extensionality using the fact that, for all $x : A$,

$$\mathsf{merid}(a)_*(\tau(\mathsf{N},\mathsf{N},x)) = \tau(\mathsf{N},\mathsf{N},x) \cdot \mathsf{merid}(a)^{-1} \equiv \mathsf{refl}_{\mathsf{N}} \cdot \mathsf{merid}(a) = \mathsf{merid}(a) = \mathsf{merid}(x) \equiv \tau(\mathsf{N},\mathsf{S},x).$$

In a symmetric fashion we may define $\tau(\mathsf{S})$ by

$$\tau(\mathsf{S},\mathsf{N},a) :\equiv \mathsf{merid}(a)^{-1} \qquad \text{and} \qquad \tau(\mathsf{S},\mathsf{S},u) :\equiv \mathsf{refl}_{\mathsf{S}}.$$

To complete the construction of τ, we need to check $\mathsf{merid}(a)_*(\tau(\mathsf{N})) = \tau(\mathsf{S})$, given any $a : A$. The verification proceeds much along the same lines by induction on the second argument of τ.

Thus, by Theorem 7.2.2 we have that $\Sigma(A)$ is a set and that $P(x,y) \simeq (x = y)$ for all $x, y : \Sigma(A)$. Taking $x :\equiv \mathsf{N}$ and $y :\equiv \mathsf{S}$ yields $A \simeq (\mathsf{N} =_{\Sigma(A)} \mathsf{S})$ as desired. □

Theorem 10.1.14 (Diaconescu)**.** *The axiom of choice implies the law of excluded middle.*

Proof. We use the equivalent form of choice given in Lemma 3.8.2. Consider a mere proposition A. The function $f : \mathbf{2} \to \Sigma(A)$ defined by $f(0_{\mathbf{2}}) :\equiv \mathsf{N}$ and $f(1_{\mathbf{2}}) :\equiv \mathsf{S}$ is surjective. Indeed, we have $(0_{\mathbf{2}}, \mathsf{refl}_{\mathsf{N}}) : \mathsf{fib}_f(\mathsf{N})$ and $(1_{\mathbf{2}}, \mathsf{refl}_{\mathsf{S}}) : \mathsf{fib}_f(\mathsf{S})$. Since $\left\|\mathsf{fib}_f(x)\right\|$ is a mere proposition, by induction the claimed surjectivity follows.

By Lemma 10.1.13 the suspension $\Sigma(A)$ is a set, so by the axiom of choice there merely exists a section $g : \Sigma(A) \to \mathbf{2}$ of f. As equality on $\mathbf{2}$ is decidable we get

$$(g(f(0_{\mathbf{2}})) = g(f(1_{\mathbf{2}}))) + \neg(g(f(0_{\mathbf{2}})) = g(f(1_{\mathbf{2}}))),$$

and, since g is a section of f, hence injective,

$$(f(0_{\mathbf{2}}) = f(1_{\mathbf{2}})) + \neg(f(0_{\mathbf{2}}) = f(1_{\mathbf{2}})).$$

Finally, since $(f(0_{\mathbf{2}}) = f(1_{\mathbf{2}})) = (\mathsf{N} = \mathsf{S}) = A$ by Lemma 10.1.13, we have $A + \neg A$. □

Theorem 10.1.15. *If the axiom of choice holds then the category $\mathcal{S}et$ is a well-pointed boolean elementary topos with choice.*

Proof. Since AC implies LEM, we have a boolean elementary topos with choice by Theorem 10.1.12 and the remark following it. We leave the proof of well-pointedness as an exercise for the reader (Exercise 10.3). □

Remark 10.1.16. The conditions on a category mentioned in the theorem are known as Lawvere's axioms for the *Elementary Theory of the Category of Sets* [Law05].

10.2 Cardinal numbers

Definition 10.2.1. The **type of cardinal numbers** is the 0-truncation of the type Set of sets:

$$\mathsf{Card} :\equiv \|\mathsf{Set}\|_0$$

Thus, a **cardinal number**, or **cardinal**, is an inhabitant of $\mathsf{Card} \equiv \|\mathsf{Set}\|_0$. Technically, of course, there is a separate type $\mathsf{Card}_{\mathcal{U}}$ associated to each universe $\mathcal{U}$.

As usual for truncations, if A is a set, then $|A|_0$ denotes its image under the canonical projection $\mathsf{Set} \to \|\mathsf{Set}\|_0 \equiv \mathsf{Card}$; we call $|A|_0$ the **cardinality** of A. By definition, Card is a set. It also inherits the structure of a semiring from Set.

Definition 10.2.2. The operation of **cardinal addition**

$$(- + -) : \mathsf{Card} \to \mathsf{Card} \to \mathsf{Card}$$

is defined by induction on truncation:

$$|A|_0 + |B|_0 :\equiv |A + B|_0.$$

Proof. Since $\mathsf{Card} \to \mathsf{Card}$ is a set, to define $(\alpha + -) : \mathsf{Card} \to \mathsf{Card}$ for all $\alpha : \mathsf{Card}$, by induction it suffices to assume that α is $|A|_0$ for some $A : \mathsf{Set}$. Now we want to define $(|A|_0 + -) : \mathsf{Card} \to \mathsf{Card}$, i.e. we want to define $|A|_0 + \beta : \mathsf{Card}$ for all $\beta : \mathsf{Card}$. However, since Card is a set, by induction it suffices to assume that β is $|B|_0$ for some $B : \mathsf{Set}$. But now we can define $|A|_0 + |B|_0$ to be $|A + B|_0$. □

Definition 10.2.3. Similarly, the operation of **cardinal multiplication**

$$(- \cdot -) : \mathsf{Card} \to \mathsf{Card} \to \mathsf{Card}$$

is defined by induction on truncation:

$$|A|_0 \cdot |B|_0 :\equiv |A \times B|_0$$

Lemma 10.2.4. Card *is a commutative semiring, i.e. for* $\alpha, \beta, \gamma : \mathsf{Card}$ *we have the following.*

$$\begin{gathered}
(\alpha + \beta) + \gamma = \alpha + (\beta + \gamma) \\
\alpha + 0 = \alpha \\
\alpha + \beta = \beta + \alpha \\
(\alpha \cdot \beta) \cdot \gamma = \alpha \cdot (\beta \cdot \gamma) \\
\alpha \cdot 1 = \alpha \\
\alpha \cdot \beta = \beta \cdot \alpha \\
\alpha \cdot (\beta + \gamma) = \alpha \cdot \beta + \alpha \cdot \gamma
\end{gathered}$$

where $0 :\equiv |\mathbf{0}|_0$ *and* $1 :\equiv |\mathbf{1}|_0$.

Proof. We prove the commutativity of multiplication, $\alpha \cdot \beta = \beta \cdot \alpha$; the others are exactly analogous. Since Card is a set, the type $\alpha \cdot \beta = \beta \cdot \alpha$ is a mere proposition, and in particular a set. Thus, by induction it suffices to assume α and β are of the form $|A|_0$ and $|B|_0$ respectively, for some $A, B : \mathsf{Set}$. Now $|A|_0 \cdot |B|_0 \equiv |A \times B|_0$ and $|B|_0 \times |A|_0 \equiv |B \times A|_0$, so it suffices to show $A \times B = B \times A$. Finally, by univalence, it suffices to give an equivalence $A \times B \simeq B \times A$. But this is easy: take $(a, b) \mapsto (b, a)$ and its obvious inverse. □

Definition 10.2.5. The operation of **cardinal exponentiation** is also defined by induction on truncation:

$$|A|_0^{|B|_0} :\equiv |B \to A|_0.$$

Lemma 10.2.6. *For $\alpha, \beta, \gamma : \mathsf{Card}$ we have*

$$\begin{aligned}
\alpha^0 &= 1 \\
1^\alpha &= 1 \\
\alpha^1 &= \alpha \\
\alpha^{\beta+\gamma} &= \alpha^\beta \cdot \alpha^\gamma \\
\alpha^{\beta\cdot\gamma} &= (\alpha^\beta)^\gamma \\
(\alpha \cdot \beta)^\gamma &= \alpha^\gamma \cdot \beta^\gamma
\end{aligned}$$

Proof. Exactly like Lemma 10.2.4. □

Definition 10.2.7. The relation of **cardinal inequality**

$$(- \leq -) : \mathsf{Card} \to \mathsf{Card} \to \mathsf{Prop}$$

is defined by induction on truncation:

$$|A|_0 \leq |B|_0 :\equiv \|\mathsf{inj}(A, B)\|$$

where $\mathsf{inj}(A, B)$ is the type of injections from A to B. In other words, $|A|_0 \leq |B|_0$ means that there merely exists an injection from A to B.

Lemma 10.2.8. *Cardinal inequality is a preorder, i.e. for $\alpha, \beta : \mathsf{Card}$ we have*

$$\begin{aligned}
&\alpha \leq \alpha \\
&(\alpha \leq \beta) \to (\beta \leq \gamma) \to (\alpha \leq \gamma)
\end{aligned}$$

Proof. As before, by induction on truncation. For instance, since $(\alpha \leq \beta) \to (\beta \leq \gamma) \to (\alpha \leq \gamma)$ is a mere proposition, by induction on 0-truncation we may assume α, β, and γ are $|A|_0$, $|B|_0$, and $|C|_0$ respectively. Now since $|A|_0 \leq |C|_0$ is a mere proposition, by induction on (-1)-truncation we may assume given injections $f : A \to B$ and $g : B \to C$. But then $g \circ f$ is an injection from A to C, so $|A|_0 \leq |C|_0$ holds. Reflexivity is even easier. □

We may likewise show that cardinal inequality is compatible with the semiring operations.

Lemma 10.2.9. *Consider the following statements:*

(i) There is an injection $A \to B$.
(ii) There is a surjection $B \to A$.

Then, assuming excluded middle:

- *Given $a_0 : A$, we have (i)→(ii).*
- *Therefore, if A is merely inhabited, we have (i) → merely (ii).*
- *Assuming the axiom of choice, we have (ii) → merely (i).*

Proof. If $f : A \to B$ is an injection, define $g : B \to A$ at $b : B$ as follows. Since f is injective, the fiber of f at b is a mere proposition. Therefore, by excluded middle, either there is an $a : A$ with $f(a) = b$, or not. In the first case, define $g(b) :\equiv a$; otherwise set $g(b) :\equiv a_0$. Then for any $a : A$, we have $a = g(f(a))$, so g is surjective.

The second statement follows from this by induction on truncation. For the third, if $g : B \to A$ is surjective, then by the axiom of choice, there merely exists a function $f : A \to B$ with $g(f(a)) = a$ for all a. But then f must be injective. □

Theorem 10.2.10 (Schroeder–Bernstein)**.** *Assuming excluded middle, for sets A and B we have*

$$\mathsf{inj}(A, B) \to \mathsf{inj}(B, A) \to (A \cong B)$$

Proof. The usual "back-and-forth" argument applies without significant changes. Note that it actually constructs an isomorphism $A \cong B$ (assuming excluded middle so that we can decide whether a given element belongs to a cycle, an infinite chain, a chain beginning in A, or a chain beginning in B). □

Corollary 10.2.11. *Assuming excluded middle, cardinal inequality is a partial order, i.e. for $\alpha, \beta : \mathsf{Card}$ we have*

$$(\alpha \leq \beta) \to (\beta \leq \alpha) \to (\alpha = \beta).$$

Proof. Since $\alpha = \beta$ is a mere proposition, by induction on truncation we may assume α and β are $|A|_0$ and $|B|_0$, respectively, and that we have injections $f : A \to B$ and $g : B \to A$. But then the Schroeder–Bernstein theorem gives an isomorphism $A \cong B$, hence an equality $|A|_0 = |B|_0$. □

Finally, we can reproduce Cantor's theorem, showing that for every cardinal there is a greater one.

Theorem 10.2.12 (Cantor)**.** *For $A : \mathsf{Set}$, there is no surjection $A \to (A \to \mathbf{2})$.*

Proof. Suppose $f : A \to (A \to \mathbf{2})$ is any function, and define $g : A \to \mathbf{2}$ by $g(a) :\equiv \neg f(a)(a)$. If $g = f(a_0)$, then $g(a_0) = f(a_0)(a_0)$ but $g(a_0) = \neg f(a_0)(a_0)$, a contradiction. Thus, f is not surjective. □

Corollary 10.2.13. *Assuming excluded middle, for any $\alpha : \mathsf{Card}$, there is a cardinal β such that $\alpha \leq \beta$ and $\alpha \neq \beta$.*

Proof. Let $\beta = 2^\alpha$. Now we want to show a mere proposition, so by induction we may assume α is $|A|_0$, so that $\beta \equiv |A \to \mathbf{2}|_0$. Using excluded middle, we have a function $f : A \to (A \to \mathbf{2})$ defined by

$$f(a)(a') :\equiv \begin{cases} 1_\mathbf{2} & a = a' \\ 0_\mathbf{2} & a \neq a'. \end{cases}$$

And if $f(a) = f(a')$, then $f(a')(a) = f(a)(a) = 1_\mathbf{2}$, so $a = a'$; hence f is injective. Thus, $\alpha \equiv |A|_0 \leq |A \to \mathbf{2}|_0 \equiv 2^\alpha$.

On the other hand, if $2^\alpha \leq \alpha$, then we would have an injection $(A \to \mathbf{2}) \to A$. By Lemma 10.2.9, since we have $(\lambda x.\, 0_\mathbf{2}) : A \to \mathbf{2}$ and excluded middle, there would then be a surjection $A \to (A \to \mathbf{2})$, contradicting Cantor's theorem. □

10.3 Ordinal numbers

Definition 10.3.1. Let A be a set and

$$(- < -) : A \to A \to \mathsf{Prop}$$

a binary relation on A. We define by induction what it means for an element $a : A$ to be **accessible** by $<$:

- If b is accessible for every $b < a$, then a is accessible.

We write $\mathsf{acc}(a)$ to mean that a is accessible.

It may seem that such an inductive definition can never get off the ground, but of course if a has the property that there are *no* b such that $b < a$, then a is vacuously accessible.

Note that this is an inductive definition of a family of types, like the type of vectors considered in §5.7. More precisely, it has one constructor, say $\mathsf{acc}_<$, with type

$$\mathsf{acc}_< : \prod_{a:A} \Big(\prod_{b:A} (b < a) \to \mathsf{acc}(b) \Big) \to \mathsf{acc}(a).$$

The induction principle for acc says that for any $P : \prod_{(a:A)} \mathsf{acc}(a) \to \mathcal{U}$, if we have

$$f : \prod_{(a:A)} \prod_{(h:\prod_{(b:A)}(b<a)\to\mathsf{acc}(b))} \Big(\prod_{(b:A)} \prod_{(l:b<a)} P(b, h(b,l)) \Big) \to P(a, \mathsf{acc}_<(a,h)),$$

then we have $g : \prod_{(a:A)} \prod_{(c:\mathsf{acc}(a))} P(a,c)$ defined by induction, with

$$g(a, \mathsf{acc}_<(a,h)) \equiv f(a,\ h,\ \lambda b.\, \lambda l.\, g(b, h(b,l))).$$

This is a mouthful, but generally we apply it only in the simpler case where $P : A \to \mathcal{U}$ depends only on A. In this case the second and third arguments of f may be combined, so that what we have to prove is

$$f : \prod_{a:A} \Big(\prod_{b:A} (b < a) \to \mathsf{acc}(b) \times P(b) \Big) \to P(a).$$

That is, we assume every $b < a$ is accessible and $g(b) : P(b)$ is defined, and from these define $g(a) : P(a)$.

The omission of the second argument of P is justified by the following lemma, whose proof is the only place where we use the more general form of the induction principle.

Lemma 10.3.2. *Accessibility is a mere property.*

Proof. We must show that for any $a : A$ and $s_1, s_2 : \mathsf{acc}(a)$ we have $s_1 = s_2$. We prove this by induction on s_1, with

$$P_1(a, s_1) :\equiv \prod_{s_2:\mathsf{acc}(a)} (s_1 = s_2).$$

Thus, we must show that for any $a : A$ and $h_1 : \prod_{(b:A)} (b < a) \to \mathsf{acc}(b)$ and

$$k_1 : \prod_{(b:A)} \prod_{(l:b<a)} \prod_{(t:\mathsf{acc}(b))} h_1(b,l) = t,$$

we have $\mathsf{acc}_<(a,h) = s_2$ for any $s_2 : \mathsf{acc}(a)$. We regard this statement as $\prod_{(a:A)} \prod_{(s_2:\mathsf{acc}(a))} P_2(a, s_2)$, where

$$P_2(a, s_2) :\equiv \prod_{(h_1:\cdots)} \prod_{(k_1:\cdots)} (\mathsf{acc}_<(a,h_1) = s_2);$$

thus we may prove it by induction on s_2. Therefore, we assume $h_2 : \prod_{(b:A)} (b < a) \to \mathsf{acc}(b)$, and k_2 with a monstrous but irrelevant type, and must show that for any h_1 and k_1 with types as above, we have $\mathsf{acc}_<(a,h_1) = \mathsf{acc}_<(a,h_2)$. By function extensionality, it suffices to show $h_1(b,l) = h_2(b,l)$ for all $b : A$ and $l : b < a$. This follows from k_1. □

Definition 10.3.3. A binary relation $<$ on a set A is **well-founded** if every element of A is accessible.

The point of well-foundedness is that for $P : A \to \mathcal{U}$, we can use the induction principle of acc to conclude $\prod_{(a:A)} \mathsf{acc}(a) \to P(a)$, and then apply well-foundedness to conclude $\prod_{(a:A)} P(a)$. In other words, if from $\forall(b : A).\, (b < a) \to P(b)$ we can prove $P(a)$, then $\forall(a : A).\, P(a)$. This is called **well-founded induction**.

Lemma 10.3.4. *Well-foundedness is a mere property.*

Proof. Well-foundedness of $<$ is the type $\prod_{(a:A)} \mathsf{acc}(a)$, which is a mere proposition since each $\mathsf{acc}(a)$ is. □

Example 10.3.5. Perhaps the most familiar well-founded relation is the usual strict ordering on $\mathbb{N}$. To show that this is well-founded, we must show that n is accessible for each $n : \mathbb{N}$. This is just the usual proof of "strong induction" from ordinary induction on $\mathbb{N}$.

Specifically, we prove by induction on $n : \mathbb{N}$ that k is accessible for all $k \leq n$. The base case is just that 0 is accessible, which is vacuously true since nothing is strictly less than 0. For the inductive step, we assume that k is accessible for all $k \leq n$, which is to say for all $k < n + 1$; hence by definition $n + 1$ is also accessible.

A different relation on $\mathbb{N}$ which is also well-founded is obtained by setting only $n < \mathsf{succ}(n)$ for all $n : \mathbb{N}$. Well-foundedness of this relation is almost exactly the ordinary induction principle of $\mathbb{N}$.

Example 10.3.6. Let $A : \mathsf{Set}$ and $B : A \to \mathsf{Set}$ be a family of sets. Recall from §5.3 that the W-type $\mathsf{W}_{(a:A)} B(a)$ is inductively generated by the single constructor

- $\mathsf{sup} : \prod_{(a:A)} (B(a) \to \mathsf{W}_{(x:A)} B(x)) \to \mathsf{W}_{(x:A)} B(x)$

We define the relation $<$ on $\mathsf{W}_{(x:A)} B(x)$ by recursion on its second argument:

- For any $a : A$ and $f : B(a) \to \mathsf{W}_{(x:A)} B(x)$, we define $w < \mathsf{sup}(a, f)$ to mean that there merely exists a $b : B(a)$ such that $w = f(b)$.

Now we prove that every $w : \mathsf{W}_{(x:A)} B(x)$ is accessible for this relation, using the usual induction principle for $\mathsf{W}_{(x:A)} B(x)$. This means we assume given $a : A$ and $f : B(a) \to \mathsf{W}_{(x:A)} B(x)$, and also a lifting $f' : \prod_{(b:B(a))} \mathsf{acc}(f(b))$. But then by definition of $<$, we have $\mathsf{acc}(w)$ for all $w < \mathsf{sup}(a, f)$; hence $\mathsf{sup}(a, f)$ is accessible.

Well-foundedness allows us to define functions by recursion and prove statements by induction, such as for instance the following. Recall from §3.5 that $\mathcal{P}(B)$ denotes the *power set* $\mathcal{P}(B) :\equiv (B \to \mathsf{Prop})$.

Lemma 10.3.7. *Suppose B is a set and we have a function*

$$g : \mathcal{P}(B) \to B$$

Then if $<$ is a well-founded relation on A, there is a function $f : A \to B$ such that for all $a : A$ we have

$$f(a) = g\Big(\{\, f(a') \mid a' < a \,\} \Big).$$

(We are using the notation for images of subsets from §10.1.2.)

Proof. We first define, for every $a : A$ and $s : \mathsf{acc}(a)$, an element $\bar{f}(a,s) : B$. By induction, it suffices to assume that s is a function assigning to each $a' < a$ a witness $s(a') : \mathsf{acc}(a')$, and that moreover for each such a' we have an element $\bar{f}(a', s(a')) : B$. In this case, we define

$$\bar{f}(a,s) :\equiv g\Big(\{\, \bar{f}(a', s(a')) \mid a' < a \,\} \Big).$$

Now since $<$ is well-founded, we have a function $w : \prod_{(a:A)} \mathsf{acc}(a)$. Thus, we can define $f(a) :\equiv \bar{f}(a, w(a))$. □

In classical logic, well-foundedness has a more well-known reformulation. In the following, we say that a subset $B : \mathcal{P}(A)$ is **nonempty** if it is unequal to the empty subset $(\lambda x.\, \bot) : \mathcal{P}(X)$. We leave it to the reader to verify that assuming excluded middle, this is equivalent to mere inhabitation, i.e. to the condition $\exists(x : A).\, x \in B$.

Lemma 10.3.8. *Assuming excluded middle, $<$ is well-founded if and only if every nonempty subset $B : \mathcal{P}(A)$ merely has a minimal element.*

Proof. Suppose first $<$ is well-founded, and suppose $B \subseteq A$ is a subset with no minimal element. That is, for any $a : A$ with $a \in B$, there merely exists a $b : A$ with $b < a$ and $b \in B$.

We claim that for any $a : A$ and $s : \mathsf{acc}(a)$, we have $a \notin B$. By induction, we may assume s is a function assigning to each $a' < a$ a proof $s(a') : \mathsf{acc}(a)$, and that moreover for each such a' we have $a' \notin B$. If $a \in B$, then by assumption, there would merely exist a $b < a$ with $b \in B$, which contradicts this assumption. Thus, $a \notin B$; this completes the induction. Since $<$ is well-founded, we have $a \notin B$ for all $a : A$, i.e. B is empty.

Now suppose each nonempty subset merely has a minimal element. Let $B = \{\, a : A \mid \neg\mathsf{acc}(a) \,\}$. Then if B is nonempty, it merely has a minimal element. Thus there merely exists an $a : A$ with $a \in B$ such that for all $b < a$, we have $\mathsf{acc}(b)$. But then by definition (and induction on truncation), a is merely accessible, and hence accessible, contradicting $a \in B$. Thus, B is empty, so $<$ is well-founded. □

Definition 10.3.9. A well-founded relation $<$ on a set A is **extensional** if for any $a, b : A$, we have

$$\Big(\forall(c : A).\, (c < a) \Leftrightarrow (c < b)\Big) \to (a = b).$$

Note that since A is a set, extensionality is a mere proposition. This notion of "extensionality" is unrelated to function extensionality, and also unrelated to the extensionality of identity types. Rather, it is a "local" counterpart of the axiom of extensionality in classical set theory.

Theorem 10.3.10. *The type of extensional well-founded relations is a set.*

Proof. By the univalence axiom, it suffices to show that if $(A, <)$ is extensional and well-founded and $f : (A, <) \cong (A, <)$, then $f = \mathsf{id}_A$. We prove by induction on $<$ that $f(a) = a$ for all $a : A$. The inductive hypothesis is that for all $a' < a$, we have $f(a') = a'$.

Now since A is extensional, to conclude $f(a) = a$ it is sufficient to show

$$\forall(c : A).\, (c < f(a)) \Leftrightarrow (c < a).$$

However, since f is an automorphism, we have $(c < a) \Leftrightarrow (f(c) < f(a))$. But $c < a$ implies $f(c) = c$ by the inductive hypothesis, so $(c < a) \to (c < f(a))$. On the other hand, if $c < f(a)$, then $f^{-1}(c) < a$, and so $c = f(f^{-1}(c)) = f^{-1}(c)$ by the inductive hypothesis again; thus $c < a$. Therefore, we have $(c < a) \Leftrightarrow (c < f(a))$ for any $c : A$, so $f(a) = a$. □

Definition 10.3.11. If $(A, <)$ and $(B, <)$ are extensional and well-founded, a **simulation** is a function $f : A \to B$ such that

(i) if $a < a'$, then $f(a) < f(a')$, and
(ii) for all $a : A$ and $b : B$, if $b < f(a)$, then there merely exists an $a' < a$ with $f(a') = b$.

Lemma 10.3.12. *Any simulation is injective.*

Proof. We prove by double well-founded induction that for any $a, b : A$, if $f(a) = f(b)$ then $a = b$. The inductive hypothesis for $a : A$ says that for any $a' < a$, and any $b : B$, if $f(a') = f(b)$ then $a = b$. The inner inductive hypothesis for $b : A$ says that for any $b' < b$, if $f(a) = f(b')$ then $a = b'$.

Suppose $f(a) = f(b)$; we must show $a = b$. By extensionality, it suffices to show that for any $c : A$ we have $(c < a) \Leftrightarrow (c < b)$. If $c < a$, then $f(c) < f(a)$ by Definition 10.3.11(i). Hence $f(c) < f(b)$, so by Definition 10.3.11(ii) there merely exists $c' : A$ with $c' < b$ and $f(c) = f(c')$. By the inductive hypothesis for a, we have $c = c'$, hence $c < b$. The dual argument is symmetrical. □

In particular, this implies that in Definition 10.3.11(ii) the word "merely" could be omitted without change of sense.

Corollary 10.3.13. *If $f : A \to B$ is a simulation, then for all $a : A$ and $b : B$, if $b < f(a)$, there* purely *exists an $a' < a$ with $f(a') = b$.*

Proof. Since f is injective, $\sum_{(a:A)}(f(a) = b)$ is a mere proposition. □

We say that a subset $C : \mathcal{P}(B)$ is an **initial segment** if $c \in C$ and $b < c$ imply $b \in C$. The image of a simulation must be an initial segment, while the inclusion of any initial segment is a simulation. Thus, by univalence, every simulation $A \to B$ is *equal* to the inclusion of some initial segment of B.

Theorem 10.3.14. *For a set A, let $P(A)$ be the type of extensional well-founded relations on A. If $<_A : P(A)$ and $<_B : P(B)$ and $f : A \to B$, let $H_{<_A <_B}(f)$ be the mere proposition that f is a simulation. Then (P, H) is a standard notion of structure over $\mathcal{S}et$ in the sense of §9.8.*

Proof. We leave it to the reader to verify that identities are simulations, and that composites of simulations are simulations. Thus, we have a notion of structure. For standardness, we must show that if $<$ and $\prec$ are two extensional well-founded relations on A, and id_A is a simulation in both directions, then $<$ and $\prec$ are equal. Since extensionality and well-foundedness are mere propositions, for this it suffices to have $\forall(a, b : A).\, (a < b) \Leftrightarrow (a \prec b)$. But this follows from Definition 10.3.11(i) for id_A. □

Corollary 10.3.15. *There is a category whose objects are sets equipped with extensional well-founded relations, and whose morphisms are simulations.*

In fact, this category is a poset.

Lemma 10.3.16. *For extensional and well-founded $(A, <)$ and $(B, <)$, there is at most one simulation $f : A \to B$.*

Proof. Suppose $f, g : A \to B$ are simulations. Since being a simulation is a mere property, it suffices to show $\forall(a : A).\,(f(a) = g(a))$. By induction on $<$, we may suppose $f(a') = g(a')$ for all $a' < a$. And by extensionality of B, to have $f(a) = g(a)$ it suffices to have $\forall(b : B).\,(b < f(a)) \Leftrightarrow (b < g(a))$.

But since f is a simulation, if $b < f(a)$, then we have $a' < a$ with $f(a') = b$. By the inductive hypothesis, we have also $g(a') = b$, hence $b < g(a)$. The dual argument is symmetrical. □

Thus, if A and B are equipped with extensional and well-founded relations, we may write $A \leq B$ to mean there exists a simulation $f : A \to B$. Corollary 10.3.15 implies that if $A \leq B$ and $B \leq A$, then $A = B$.

Definition 10.3.17. An **ordinal** is a set A with an extensional well-founded relation which is *transitive*, i.e. satisfies $\forall(a, b, c : A).\,(a < b) \to (b < c) \to (a < c)$.

Example 10.3.18. Of course, the usual strict order on $\mathbb{N}$ is transitive. It is easily seen to be extensional as well; thus it is an ordinal. As usual, we denote this ordinal by ω.

Let Ord denote the type of ordinals. By the previous results, Ord is a set and has a natural partial order. We now show that Ord also admits a well-founded relation.

If A is an ordinal and $a : A$, let $A_{/a} :\equiv \{\, b : A \mid b < a \,\}$ denote the initial segment. Note that if $A_{/a} = A_{/b}$ as ordinals, then that isomorphism must respect their inclusions into A (since simulations form a poset), and hence they are equal as subsets of A. Therefore, since A is extensional, $a = b$. Thus the function $a \mapsto A_{/a}$ is an injection $A \to \mathsf{Ord}$.

Definition 10.3.19. For ordinals A and B, a simulation $f : A \to B$ is said to be **bounded** if there exists $b : B$ such that $A = B_{/b}$.

The remarks above imply that such a b is unique when it exists, so that boundedness is a mere property.

We write $A < B$ if there exists a bounded simulation from A to B. Since simulations are unique, $A < B$ is also a mere proposition.

Theorem 10.3.20. $(\mathsf{Ord}, <)$ *is an ordinal.*

More precisely, this theorem says that the type $\mathsf{Ord}_{\mathcal{U}_i}$ of ordinals in one universe is itself an ordinal in the next higher universe, i.e. $(\mathsf{Ord}_{\mathcal{U}_i}, <) : \mathsf{Ord}_{\mathcal{U}_{i+1}}$.

Proof. Let A be an ordinal; we first show that $A_{/a}$ is accessible (in Ord) for all $a : A$. By well-founded induction on A, suppose $A_{/b}$ is accessible for all $b < a$. By definition of accessibility, we must show that B is accessible in Ord for all $B < A_{/a}$. However, if $B < A_{/a}$ then there is some $b < a$ such that $B = (A_{/a})_{/b} = A_{/b}$, which is accessible by the inductive hypothesis. Thus, $A_{/a}$ is accessible for all $a : A$.

Now to show that A is accessible in Ord, by definition we must show B is accessible for all $B < A$. But as before, $B < A$ means $B = A_{/a}$ for some $a : A$, which is accessible as we just proved. Thus, Ord is well-founded.

For extensionality, suppose A and B are ordinals such that $\prod_{(C:\mathsf{Ord})}(C < A) \Leftrightarrow (C < B)$. Then for every $a : A$, since $A_{/a} < A$, we have $A_{/a} < B$, hence there is $b : B$ with $A_{/a} = B_{/b}$. Define $f : A \to B$ to take each a to the corresponding b; it is straightforward to verify that f is an isomorphism. Thus $A \cong B$, hence $A = B$ by univalence.

Finally, it is easy to see that $<$ is transitive. □

Treating Ord as an ordinal is often very convenient, but it has its pitfalls as well. For instance, consider the following lemma, where we pay attention to how universes are used.

Lemma 10.3.21. *Let $\mathcal{U}$ be a universe. For any $A : \mathsf{Ord}_{\mathcal{U}}$, there is a $B : \mathsf{Ord}_{\mathcal{U}}$ such that $A < B$.*

Proof. Let $B = A + \mathbf{1}$, with the element $\star : \mathbf{1}$ being greater than all elements of A. Then B is an ordinal and it is easy to see that $A \cong B_{/\star}$. □

The ordinal B constructed in the proof of Lemma 10.3.21 is called the **successor** of A.

This lemma illustrates a potential pitfall of the "typically ambiguous" style of using $\mathcal{U}$ to denote an arbitrary, unspecified universe. Consider the following alternative proof of it.

Another putative proof of Lemma 10.3.21. Note that $C < A$ if and only if $C = A_{/a}$ for some $a : A$. This gives an isomorphism $A \cong \mathsf{Ord}_{/A}$, so that $A < \mathsf{Ord}$. Thus we may take $B :\equiv \mathsf{Ord}$. □

The second proof would be valid if we had stated Lemma 10.3.21 in a typically ambiguous style. But the resulting lemma would be less useful, because the second proof would constrain the second "Ord" in the lemma statement to refer to a higher universe level than the first one. The first proof allows both universes to be the same.

Similar remarks apply to the next lemma, which could be proved in a less useful way by observing that $A \leq \mathsf{Ord}$ for any $A : \mathsf{Ord}$.

Lemma 10.3.22. *Let $\mathcal{U}$ be a universe. For any $X : \mathcal{U}$ and $F : X \to \mathsf{Ord}_{\mathcal{U}}$, there exists $B : \mathsf{Ord}_{\mathcal{U}}$ such that $Fx \leq B$ for all $x : X$.*

Proof. Let B be the quotient of the equivalence relation $\sim$ on $\sum_{(x:X)} Fx$ defined as follows:

$$(x, y) \sim (x', y') \;:\equiv\; \Big((Fx)_{/y} \cong (Fx')_{/y'}\Big).$$

Define $(x, y) < (x', y')$ if $(Fx)_{/y} < (Fx')_{/y'}$. This clearly descends to the quotient, and can be seen to make B into an ordinal. Moreover, for each $x : X$ the induced map $Fx \to B$ is a simulation. □

10.4 Classical well-orderings

We now show the equivalence of our ordinals with the more familiar classical well-orderings.

Lemma 10.4.1. *Assuming excluded middle, every ordinal is trichotomous:*

$$\forall (a, b : A).\, (a < b) \vee (a = b) \vee (b < a).$$

Proof. By induction on a, we may assume that for every $a' < a$ and every $b' : A$, we have $(a' < b') \vee (a' = b') \vee (b' < a')$. Now by induction on b, we may assume that for every $b' < b$, we have $(a < b') \vee (a = b') \vee (b' < a)$.

By excluded middle, either there merely exists a $b' < b$ such that $a < b'$, or there merely exists a $b' < b$ such that $a = b'$, or for every $b' < b$ we have $b' < a$. In the first case, merely $a < b$ by transitivity, hence $a < b$ as it is a mere proposition. Similarly, in the second case, $a < b$ by transport. Thus, suppose $\forall (b' : A).\, (b' < b) \to (b' < a)$.

Now analogously, either there merely exists $a' < a$ such that $b < a'$, or there merely exists $a' < a$ such that $a' = b$, or for every $a' < a$ we have $a' < b$. In the first and second cases, $b < a$, so we may suppose $\forall (a' : A).\, (a' < a) \to (a' < b)$. However, by extensionality, our two suppositions now imply $a = b$. □

Lemma 10.4.2. *A well-founded relation contains no cycles, i.e.*

$$\forall(n : \mathbb{N}).\, \forall(a : \mathbb{N}_n \to A).\, \neg\Big((a_0 < a_1) \wedge \cdots \wedge (a_{n-1} < a_n) \wedge (a_n < a_0)\Big).$$

Proof. We prove by induction on $a : A$ that there is no cycle containing a. Thus, suppose by induction that for all $a' < a$, there is no cycle containing a'. But in any cycle containing a, there is some element less than a and contained in the same cycle. □

In particular, a well-founded relation must be **irreflexive**, i.e. $\neg(a < a)$ for all a.

Theorem 10.4.3. *Assuming excluded middle, $(A, <)$ is an ordinal if and only if every nonempty subset $B \subseteq A$ has a least element.*

Proof. If A is an ordinal, then by Lemma 10.3.8 every nonempty subset merely has a minimal element. But trichotomy implies that any minimal element is a least element. Moreover, least elements are unique when they exist, so merely having one is as good as having one.

Conversely, if every nonempty subset has a least element, then by Lemma 10.3.8, A is well-founded. We also have trichotomy, since for any a, b the subset $\{\, a, b \,\} :\equiv \{\, x : A \mid x = a \vee x = b \,\}$ merely has a least element, which must be either a or b. This implies transitivity, since if $a < b$ and $b < c$, then either $a = c$ or $c < a$ would produce a cycle. Similarly, it implies extensionality, for if $\forall(c : A).\, (c < a) \Leftrightarrow (c < b)$, then $a < b$ implies (letting c be a) that $a < a$, which is a cycle, and similarly if $b < a$; hence $a = b$. □

In classical mathematics, the characterization of Theorem 10.4.3 is taken as the definition of a *well-ordering*, with the *ordinals* being a canonical set of representatives of isomorphism classes for well-orderings. In our context, the structure identity principle means that there is no need to look for such representatives: any well-ordering is as good as any other.

We now move on to consider consequences of the axiom of choice. For any set X, let $\mathcal{P}_+(X)$ denote the type of merely inhabited subsets of X:

$$\mathcal{P}_+(X) :\equiv \{\, Y : \mathcal{P}(X) \mid \exists(x : X).\, x \in Y \,\}.$$

Assuming excluded middle, this is equivalently the type of *nonempty* subsets of X, and we have $\mathcal{P}(X) \simeq (\mathcal{P}_+(X)) + \mathbf{1}$.

Theorem 10.4.4. *Assuming excluded middle, the following are equivalent.*

(i) For every set X, there merely exists a function $f : \mathcal{P}_+(X) \to X$ such that $f(Y) \in Y$ for all $Y : \mathcal{P}_+(X)$.

(ii) Every set merely admits the structure of an ordinal.

Of course, (i) is a standard classical version of the axiom of choice; see Exercise 10.10.

Proof. One direction is easy: suppose (ii). Since we aim to prove the mere proposition (i), we may assume A is an ordinal. But then we can define $f(B)$ to be the least element of B.

Now suppose (i). As before, since (ii) is a mere proposition, we may assume given such an f. We extend f to a function

$$\bar{f} : \mathcal{P}(X) \simeq (\mathcal{P}_+(X)) + \mathbf{1} \longrightarrow X + \mathbf{1}$$

in the obvious way. Now for any ordinal A, we can define $g_A : A \to X + \mathbf{1}$ by well-founded recursion:

$$g_A(a) :\equiv \bar{f}\Big(X \setminus \{\, g_A(b) \mid (b < a) \wedge (g_A(b) \in X) \,\}\Big)$$

(regarding X as a subset of $X + \mathbf{1}$ in the obvious way).

Let $A' :\equiv \{\, a : A \mid g_A(a) \in X \,\}$ be the preimage of $X \subseteq X + \mathbf{1}$; then we claim the restriction $g'_A : A' \to X$ is injective. For if $a, a' : A$ with $a \neq a'$, then by trichotomy and without loss of generality, we may assume $a' < a$. Thus $g_A(a') \in \{\, g_A(b) \mid b < a \,\}$, so since $f(Y) \in Y$ for all Y we have $g_A(a) \neq g_A(a')$.

Moreover, A' is an initial segment of A. For $g_A(a)$ lies in $\mathbf{1}$ if and only if $\{\, g_A(b) \mid b < a \,\} = X$, and if this holds then it also holds for any $a' > a$. Thus, A' is itself an ordinal.

Finally, since Ord is an ordinal, we can take $A :\equiv \mathsf{Ord}$. Let X' be the image of $g'_{\mathsf{Ord}} : \mathsf{Ord}' \to X$; then the inverse of g'_{Ord} yields an injection $H : X' \to \mathsf{Ord}$. By Lemma 10.3.22, there is an ordinal C such that $Hx \leq C$ for all $x : X'$. Then by Lemma 10.3.21, there is a further ordinal D such that $C < D$, hence $Hx < D$ for all $x : X'$. Now we have

$$\begin{aligned} g_{\mathsf{Ord}}(D) &= \bar{f}\Big(X \setminus \Big\{\, g_{\mathsf{Ord}}(B) \;\Big|\; B < D \wedge (g_{\mathsf{Ord}}(B) \in X) \,\Big\}\Big) \\ &= \bar{f}\Big(X \setminus \Big\{\, g_{\mathsf{Ord}}(B) \;\Big|\; g_{\mathsf{Ord}}(B) \in X \,\Big\}\Big) \end{aligned}$$

since if $B : \mathsf{Ord}$ and $(g_{\mathsf{Ord}}(B) \in X)$, then $B = Hx$ for some $x : X'$, hence $B < D$. Now if

$$\Big\{\, g_{\mathsf{Ord}}(B) \;\Big|\; g_{\mathsf{Ord}}(B) \in X \,\Big\}$$

is not all of X, then $g_{\mathsf{Ord}}(D)$ would lie in X but not in this subset, which would be a contradiction since D is itself a potential value for B. So this set must be all of X, and hence g'_{Ord} is surjective as well as injective. Thus, we can transport the ordinal structure on Ord' to X. □

Remark 10.4.5. If we had given the wrong proof of Lemma 10.3.21 or Lemma 10.3.22, then the resulting proof of Theorem 10.4.4 would be invalid: there would be no way to consistently assign universe levels. As it is, we require propositional resizing (which follows from LEM) to ensure that X' lives in the same universe as X (up to equivalence).

Corollary 10.4.6. *Assuming the axiom of choice, the function* $\mathsf{Ord} \to \mathsf{Set}$ *(which forgets the order structure) is a surjection.*

Note that Ord is a set, while Set is a 1-type. In general, there is no reason for a 1-type to admit any surjective function from a set. Even the axiom of choice does not appear to imply that *every* 1-type does so (although see Exercise 7.9), but it readily implies that this is so for 1-types constructed out of Set, such as the types of objects of categories of structures as in §9.8. The following corollary also applies to such categories.

Corollary 10.4.7. *Assuming* AC, *$\mathcal{S}et$ admits a weak equivalence functor from a strict category.*

Proof. Let $X_0 :\equiv \mathsf{Ord}$, and for $A, B : X_0$ let $\hom_X(A, B) :\equiv (A \to B)$. Then X is a strict category, since Ord is a set, and the above surjection $X_0 \to \mathsf{Set}$ extends to a weak equivalence functor $X \to \mathcal{S}et$. □

Now recall from §10.2 that we have a further surjection $|-|_0 : \mathsf{Set} \to \mathsf{Card}$, and hence a composite surjection $\mathsf{Ord} \to \mathsf{Card}$ which sends each ordinal to its cardinality.

Theorem 10.4.8. *Assuming* AC*, the surjection* Ord $\to$ Card *has a section.*

Proof. There is an easy and wrong proof of this: since Ord and Card are both sets, AC implies that any surjection between them *merely* has a section. However, we actually have a canonical *specified* section: because Ord is an ordinal, every nonempty subset of it has a uniquely specified least element. Thus, we can map each cardinal to the least element in the corresponding fiber. □

It is traditional in set theory to identify cardinals with their image in Ord: the least ordinal having that cardinality.

It follows that Card also canonically admits the structure of an ordinal: in fact, one isomorphic to Ord. Specifically, we define by well-founded recursion a function $\aleph : \mathsf{Ord} \to \mathsf{Ord}$, such that $\aleph(A)$ is the least ordinal having cardinality greater than $\aleph(A_{/a})$ for all $a : A$. Then (assuming AC) the image of $\aleph$ is exactly the image of Card.

10.5 The cumulative hierarchy

We can define a cumulative hierarchy V of all sets in a given universe $\mathcal{U}$ as a higher inductive type, in such a way that V is again a set (in a larger universe $\mathcal{U}'$), equipped with a binary "membership" relation $x \in y$ which satisfies the usual laws of set theory.

Definition 10.5.1. The **cumulative hierarchy** V relative to a type universe $\mathcal{U}$ is the higher inductive type generated by the following constructors.

(i) For every $A : \mathcal{U}$ and $f : A \to V$, there is an element $\mathsf{set}(A, f) : V$.

(ii) For all $A, B : \mathcal{U}$, $f : A \to V$ and $g : B \to V$ such that

$$\big(\forall(a : A).\, \exists(b : B).\, f(a) =_V g(b)\big) \wedge \big(\forall(b : B).\, \exists(a : A).\, f(a) =_V g(b)\big) \tag{10.5.2}$$

there is a path $\mathsf{set}(A, f) =_V \mathsf{set}(B, g)$.

(iii) The 0-truncation constructor: for all $x, y : V$ and $p, q : x = y$, we have $p = q$.

In set-theoretic language, $\mathsf{set}(A, f)$ can be understood as the set (in the sense of classical set theory) that is the image of A under f, i.e. $\{\, f(a) \mid a \in A \,\}$. However, we will avoid this notation, since it would clash with our notation for subtypes (but see (10.5.3) and Definition 10.5.7 below).

The hierarchy V is bootstrapped from the empty map $\mathsf{rec}_{\mathbf{0}}(V) : \mathbf{0} \to V$, which gives the empty set as $\emptyset = \mathsf{set}(\mathbf{0}, \mathsf{rec}_{\mathbf{0}}(V))$. Then the singleton $\{\emptyset\}$ enters V through $\mathbf{1} \to V$, defined as $\star \mapsto \emptyset$, and so on. (The definition can also be adapted to include an arbitrary set of "atoms" or "urelements", by adding an additional point constructor.) The type V lives in the same universe as the base universe $\mathcal{U}$.

The second constructor of V has a form unlike any we have seen before: it involves not only paths in V (which in §6.9 we claimed were slightly fishy) but truncations of sums of them. It certainly does not fit the general scheme described in §6.13, and thus it may not be obvious what its induction principle should be. Fortunately, like our first definition of the 0-truncation in §6.9, it can be re-expressed using auxiliary higher inductive types. We leave it to the reader to work out the details (see Exercise 10.11).

At the end of the day, the induction principle for V (written in pattern matching language) says that given $P : V \to \mathsf{Set}$, in order to construct $h : \prod_{(x:V)} P(x)$, it suffices to give the following.

(i) For any $f : A \to V$, construct $h(\mathsf{set}(A, f))$, assuming as given $h(f(a))$ for all $a : A$.

(ii) Verify that if $f : A \to V$ and $g : B \to V$ satisfy (10.5.2), then $h(\mathsf{set}(A, f)) =^P_q h(\mathsf{set}(B, g))$, where q is the path arising from the second constructor of V and (10.5.2), assuming inductively that $h(f(a))$ and $h(g(b))$ are defined for all $a : A$ and $b : B$, and that the following condition holds:

$$\begin{aligned} & \big(\forall(a : A).\, \exists(b : B).\, \exists(p : f(a) = g(b)).\, h(f(a)) =^P_p h(g(b))\big) \\ \wedge \quad & \big(\forall(b : B).\, \exists(a : A).\, \exists(p : f(a) = g(b)).\, h(f(a)) =^P_p h(g(b))\big) \end{aligned}$$

The second clause checks that the map being defined must respect the paths introduced in (10.5.2). As usual when we state higher induction principles using pattern matching, it may seem tautologous, but is not. The point is that "$h(f(a))$" is essentially a formal symbol which we cannot peek inside of, which $h(\mathsf{set}(A, f))$ must be defined in terms of. Thus, in the second clause, we assume equality of these formal symbols when appropriate, and verify that the elements resulting from the construction of the first clause are also equal. Of course, if P is a family of mere propositions, then the second clause is automatic.

Observe that, by induction, for each $v : V$ there merely exist $A : \mathcal{U}$ and $f : A \to V$ such that $v = \mathsf{set}(A, f)$. Thus, it is reasonable to try to define the **membership relation** $x \in v$ on V by setting:

$$(x \in \mathsf{set}(A, f)) :\equiv (\exists(a : A).\, x = f(a)).$$

To see that the definition is valid, we must use the recursion principle of V. Thus, suppose we have a path $\mathsf{set}(A, f) = \mathsf{set}(B, g)$ constructed through (10.5.2). If $x \in \mathsf{set}(A, f)$ then there merely is $a : A$ such that $x = f(a)$, but by (10.5.2) there merely is $b : B$ such that $f(a) = g(b)$, hence $x = g(b)$ and $x \in \mathsf{set}(B, g)$. The converse is symmetric.

The **subset relation** $x \subseteq y$ is defined on V as usual by

$$(x \subseteq y) :\equiv \forall(z : V).\, z \in x \Rightarrow z \in y.$$

A **class** may be taken to be a mere predicate on V. We can say that a class $C : V \to \mathsf{Prop}$ is a **V-set** if there merely exists $v : V$ such that

$$\forall(x : V).\, C(x) \Leftrightarrow x \in v.$$

We may also use the conventional notation for classes, which matches our standard notation for subtypes:

$$\{\, x \mid C(x) \,\} :\equiv \lambda x.\, C(x). \tag{10.5.3}$$

A class $C : V \to \mathsf{Prop}$ will be called **$\mathcal{U}$-small** if all of its values $C(x)$ lie in $\mathcal{U}$, specifically $C : V \to \mathsf{Prop}_{\mathcal{U}}$. Since V lives in the same universe $\mathcal{U}'$ as does the base universe $\mathcal{U}$ from which it is built, the same is true for the identity types $v =_V w$ for any $v, w : V$. To obtain a well-behaved theory in the absence of propositional resizing, therefore, it will be convenient to have a $\mathcal{U}$-small "resizing" of the identity relation, which we can define by induction as follows.

Definition 10.5.4. Define the **bisimulation** relation

$$\sim : V \times V \longrightarrow \mathsf{Prop}_{\mathcal{U}}$$

by double induction over V, where for $\mathsf{set}(A, f)$ and $\mathsf{set}(B, g)$ we let:

$$\mathsf{set}(A, f) \sim \mathsf{set}(B, g) :\equiv \big(\forall(a : A).\, \exists(b : B).\, f(a) \sim g(b)\big) \wedge \big(\forall(b : B).\, \exists(a : A).\, f(a) \sim g(b)\big).$$

To verify that the definition is correct, we just need to check that it respects paths $\mathsf{set}(A, f) = \mathsf{set}(B, g)$ constructed through (10.5.2), but this is obvious, and that $\mathsf{Prop}_{\mathcal{U}}$ is a set, which it is. Note that $u \sim v$ is in $\mathsf{Prop}_{\mathcal{U}}$ by construction.

Lemma 10.5.5. *For any $u, v : V$ we have $(u =_V v) = (u \sim v)$.*

Proof. An easy induction shows that $\sim$ is reflexive, so by transport we have $(u =_V v) \to (u \sim v)$. Thus, it remains to show that $(u \sim v) \to (u =_V v)$. By induction on u and v, we may assume they are $\mathsf{set}(A, f)$ and $\mathsf{set}(B, g)$ respectively. (We can ignore the path constructors of V, since $(u \sim v) \to (u =_V v)$ is a mere proposition.) Then by definition, $\mathsf{set}(A, f) \sim \mathsf{set}(B, g)$ implies $(\forall(a : A).\, \exists(b : B).\, f(a) \sim g(b))$ and conversely. But the inductive hypothesis then tells us that $(\forall(a : A).\, \exists(b : B).\, f(a) = g(b))$ and conversely. So by the path constructor for V we have $\mathsf{set}(A, f) =_V \mathsf{set}(B, g)$. □

One might think that we could omit the 0-truncation constructor of V and *prove* that V is 0-truncated by applying Theorem 7.2.2 to the bisimulation. However, in the proof of Lemma 10.5.5 we used the fact that V is 0-truncated, to conclude that $(u \sim v) \to (u =_V v)$ is a mere proposition so that in the induction it suffices to assume u and v are $\mathsf{set}(A, f)$ and $\mathsf{set}(B, g)$.

Now we can use the resized identity relation to get the following useful principle.

Lemma 10.5.6. *For every $u : V$ there is a given $A_u : \mathcal{U}$ and monic $m_u : A_u \rightarrowtail V$ such that $u = \mathsf{set}(A_u, m_u)$.*

Proof. Take any presentation $u = \mathsf{set}(A, f)$ and factor $f : A \to V$ as a surjection followed by an injection:

$$f = m_u \circ e_u : A \twoheadrightarrow A_u \rightarrowtail V.$$

Clearly $u = \mathsf{set}(A_u, m_u)$ if only A_u is still in $\mathcal{U}$, which holds if the kernel of $e_u : A \twoheadrightarrow A_u$ is in $\mathcal{U}$. But the kernel of $e_u : A \twoheadrightarrow A_u$ is the pullback along $f : A \to V$ of the identity on V, which we just showed to be $\mathcal{U}$-small, up to equivalence. Now, this construction of the pair (A_u, m_u) with $m_u : A_u \rightarrowtail V$ and $u = \mathsf{set}(A_u, m_u)$ from $u : V$ is unique up to equivalence over V, and hence up to identity by univalence. Thus by the principle of unique choice (3.9.2) there is a map $c : V \to \sum_{(A:\mathcal{U})}(A \to V)$ such that $c(u) = (A_u, m_u)$, with $m_u : A_u \rightarrowtail V$ and $u = \mathsf{set}(c(u))$, as claimed. □

Definition 10.5.7. For $u : V$, the just constructed monic presentation $m_u : A_u \rightarrowtail V$ such that $u = \mathsf{set}(A_u, m_u)$ may be called the **type of members** of u and denoted $m_u : [u] \rightarrowtail V$, or even $[u] \rightarrowtail V$. We can think of $[u]$ as the "subclass of V consisting of members of u".

Theorem 10.5.8. *The following hold for $(V, \in)$:*

(i) extensionality:
$$\forall(x, y : V).\, x \subseteq y \wedge y \subseteq x \Leftrightarrow x = y.$$

(ii) empty set: *for all $x : V$, we have $\neg(x \in \emptyset)$.*

(iii) pairing: *for all $u, v : V$, the class $\{u, v\} :\equiv \{\, x \mid x = u \vee x = v \,\}$ is a V-set.*

(iv) infinity: *there is a $v : V$ with $\emptyset \in v$ and $x \in v$ implies $x \cup \{x\} \in v$.*

(v) union: *for all $v : V$, the class $\cup v :\equiv \{\, x \mid \exists(u : V).\, x \in u \in v \,\}$ is a V-set.*

(vi) function set: *for all $u, v : V$, the class $v^u :\equiv \{\, x \mid x : u \to v \,\}$ is a V-set.*[1]

[1] Here $x : u \to v$ means that x is an appropriate set of ordered pairs, according to the usual way of encoding functions in set theory.

(vii) $\in$-induction: *if* $C : V \to \mathsf{Prop}$ *is a class such that* $C(a)$ *holds whenever* $C(x)$ *for all* $x \in a$*, then* $C(v)$ *for all* $v : V$*.*

(viii) replacement: *given any* $r : V \to V$ *and* $x : V$*, the class*

$$\{\, y \mid \exists(z : V).\, z \in x \wedge y = r(z) \,\}$$

is a V*-set.*

(ix) separation: *given any* $a : V$ *and* $\mathcal{U}$*-small* $C : V \to \mathsf{Prop}_{\mathcal{U}}$*, the class*

$$\{\, x \mid x \in a \wedge C(x) \,\}$$

is a V*-set.*

Sketch of proof.

(i) Extensionality: if $\mathsf{set}(A, f) \subseteq \mathsf{set}(B, g)$ then $f(a) \in \mathsf{set}(B, g)$ for every $a : A$, therefore for every $a : A$ there merely exists $b : B$ such that $f(a) = g(b)$. The assumption $\mathsf{set}(B, g) \subseteq \mathsf{set}(A, f)$ gives the other half of (10.5.2), therefore $\mathsf{set}(A, f) = \mathsf{set}(B, g)$.

(ii) Empty set: suppose $x \in \emptyset = \mathsf{set}(\mathbf{0}, \mathsf{rec}_{\mathbf{0}}(V))$. Then $\exists(a : \mathbf{0}).\, x = \mathsf{rec}_{\mathbf{0}}(V, a)$, which is absurd.

(iii) Pairing: given u and v, let $w = \mathsf{set}(\mathbf{2}, \mathsf{rec}_{\mathbf{2}}(V, u, v))$.

(iv) Infinity: take $w = \mathsf{set}(\mathbb{N}, I)$, where $I : \mathbb{N} \to V$ is given by the recursion $I(0) :\equiv \emptyset$ and $I(n+1) :\equiv I(n) \cup \{I(n)\}$.

(v) Union: Take any $v : V$ and any presentation $f : A \to V$ with $v = \mathsf{set}(A, f)$. Then let $\tilde{A} :\equiv \sum_{(a:A)} [fa]$, where $m_{fa} : [fa] \rightarrowtail V$ is the type of members from Definition 10.5.7. $\tilde{A}$ is plainly $\mathcal{U}$-small, and we have $\cup v :\equiv \mathsf{set}(\tilde{A}, \lambda x.\, m_{f(\mathsf{pr}_1(x))}(\mathsf{pr}_2(x)))$.

(vi) Function set: given $u, v : V$, take the types of members $[u] \rightarrowtail V$ and $[v] \rightarrowtail V$, and the function type $[u] \to [v]$. We want to define a map

$$r : ([u] \to [v]) \longrightarrow V$$

with "$r(f) = \{\, (x, f(x)) \mid x : [u] \,\}$", but in order for this to make sense we must first define the ordered pair (x, y), and then we take the map $r' : x \mapsto (x, f(x))$, and then we can put $r(f) :\equiv \mathsf{set}([u], r')$. But the ordered pair can be defined in terms of unordered pairing as usual.

(vii) $\in$-induction: let $C : V \to \mathsf{Prop}$ be a class such that $C(a)$ holds whenever $C(x)$ for all $x \in a$, and take any $v = \mathsf{set}(B, g)$. To show that $C(v)$ by induction, assume that $C(g(b))$ for all $b : B$. For every $x \in v$ there merely exists some $b : B$ with $x = g(b)$, and so $C(x)$. Thus $C(v)$.

(viii) Replacement: let C denote the class in question. The statement "C is a V-set" is a mere proposition, so we may proceed by induction as follows. Supposing x is $\mathsf{set}(A, f)$, we claim that $w :\equiv \mathsf{set}(A, r \circ f)$ is the set we are looking for. If $C(y)$ then there merely exists $z : V$ and $a : A$ such that $z = f(a)$ and $y = r(z)$, therefore $y \in w$. Conversely, if $y \in w$ then there merely exists $a : A$ such that $y = r(f(a))$, so if we take $z :\equiv f(a)$ we see that $C(y)$ holds.

(ix) Let us say that a class $C : V \to \mathsf{Prop}$ is **separable** if for any $a : V$ the class

$$a \cap C :\equiv \{\, x \mid x \in a \wedge C(x) \,\}$$

is a V-set. We need to show that any $\mathcal{U}$-small $C : V \to \mathsf{Prop}_{\mathcal{U}}$ is separable. Indeed, given $a = \mathsf{set}(A, f)$, let $A' = \sum_{(x:A)} C(fx)$, and take $f' = f \circ i$, where $i : A' \to A$ is the obvious inclusion. Then we can take $a' = \mathsf{set}(A', f')$ and we have $x \in a \wedge C(x) \Leftrightarrow x \in a'$ as claimed. We needed the assumption that C lands in $\mathcal{U}$ in order for $A' = \sum_{(x:A)} C(fx)$ to be in $\mathcal{U}$. □

It is also convenient to have a strictly syntactic criterion of separability, so that one can read off from the expression for a class that it produces a V-set. One such familiar condition is being "Δ_0", which means that the expression is built up from equality $x =_V y$ and membership $x \in y$, using only mere-propositional connectives $\neg$, $\wedge$, $\vee$, $\Rightarrow$ and quantifiers $\forall$, $\exists$ over particular sets, i.e. of the form $\exists(x \in a)$ and $\forall(y \in b)$ (these are called **bounded** quantifiers).

Corollary 10.5.9. *If the class $C : V \to \mathsf{Prop}$ is Δ_0 in the above sense, then it is separable.*

Proof. Recall that we have a $\mathcal{U}$-small resizing $x \sim y$ of identity $x = y$. Since $x \in y$ is defined in terms of $x = y$, we also have a $\mathcal{U}$-small resizing of membership

$$x \mathrel{\widetilde{\in}} \mathsf{set}(A, f) :\equiv \exists(a : A).\, x \sim f(a).$$

Now, let Φ be a Δ_0 expression for C, so that as classes $\Phi = C$ (strictly speaking, we should distinguish expressions from their meanings, but we will blur the difference). Let $\widetilde{\Phi}$ be the result of replacing all occurrences of $=$ and $\in$ by their resized equivalents $\sim$ and $\mathrel{\widetilde{\in}}$. Clearly then $\widetilde{\Phi}$ also expresses C, in the sense that for all $x : V$, $\widetilde{\Phi}(x) \Leftrightarrow C(x)$, and hence $\widetilde{\Phi} = C$ by univalence. It now suffices to show that $\widetilde{\Phi}$ is $\mathcal{U}$-small, for then it will be separable by the theorem.

We show that $\widetilde{\Phi}$ is $\mathcal{U}$-small by induction on the construction of the expression. The base cases are $x \sim y$ and $x \mathrel{\widetilde{\in}} y$, which have already been resized into $\mathcal{U}$. It is also clear that $\mathcal{U}$ is closed under the mere-propositional operations (and (-1)-truncation), so it just remains to check the bounded quantifiers $\exists(x \in a)$ and $\forall(y \in b)$. By definition,

$$\exists(x \in a)P(x) :\equiv \left\| \sum_{x:V} (x \mathrel{\widetilde{\in}} a \wedge P(x)) \right\|,$$
$$\forall(y \in b)P(x) :\equiv \prod_{x:V} (x \mathrel{\widetilde{\in}} a \to P(x)).$$

Let us consider $\left\| \sum_{(x:V)} (x \mathrel{\widetilde{\in}} a \wedge P(x)) \right\|$. Although the body $(x \mathrel{\widetilde{\in}} a \wedge P(x))$ is $\mathcal{U}$-small since $P(x)$ is so by the inductive hypothesis, the quantification over V need not stay inside $\mathcal{U}$. However, in the present case we can replace this with a quantification over the type $[a] \rightarrowtail V$ of members of a, and easily show that

$$\sum_{x:V} (x \mathrel{\widetilde{\in}} a \wedge P(x)) = \sum_{x:[a]} P(x).$$

The right-hand side does remain in $\mathcal{U}$, since both $[a]$ and $P(x)$ are in $\mathcal{U}$. The case of $\prod_{(x:V)}(x \mathrel{\widetilde{\in}} a \to P(x))$ is analogous, using $\prod_{(x:V)}(x \mathrel{\widetilde{\in}} a \to P(x)) = \prod_{(x:[a])} P(x)$. □

We have shown that in type theory with a universe $\mathcal{U}$, the cumulative hierarchy V is a model of a "constructive set theory" with many of the standard axioms. However, as far as we know, it lacks the *strong collection* and *subset collection* axioms which are included in Constructive Zermelo–Fraenkel Set Theory [Acz78]. In the usual interpretation of this set theory into type theory, these two axioms are consequences of the setoid-like definition of equality; while in other constructed models of set theory, strong collection may hold for other reasons. We do not know whether either of these axioms holds in our model $(V, \in)$, but it seems unlikely. Since V

is a higher inductive type *inside* the system, rather than being an *external* construction, it is not surprising that it differs in some ways from prior interpretations.

Finally, consider the result of adding the axiom of choice for sets to our type theory, in the form AC from §10.1.5 above. This has the consequence that LEM then also holds, by Theorem 10.1.14, and so Set is a topos with subobject classifier **2**, by Theorem 10.1.12. In this case, we have $\mathsf{Prop} = \mathbf{2} : \mathcal{U}$, and so *all classes are separable*. Thus we have shown:

Lemma 10.5.10. *In type theory with* AC*, the law of* ***(full) separation*** *holds for V: given* any *class $C : V \to \mathsf{Prop}$ and $a : V$, the class $a \cap C$ is a V-set.*

Theorem 10.5.11. *In type theory with* AC *and a universe $\mathcal{U}$, the cumulative hierarchy V is a model of Zermelo–Fraenkel set theory with choice, ZFC.*

Proof. We have all the axioms listed in Theorem 10.5.8, plus full separation, so we just need to show that there are power sets $\mathcal{P}(a) : V$ for all $a : V$. But since we have LEM these are simply function types $\mathcal{P}(a) = (a \to \mathbf{2})$. Thus V is a model of Zermelo–Fraenkel set theory ZF. We leave the verification of the set-theoretic axiom of choice from AC as an easy exercise. □

Notes

The basic properties one expects of the category of sets date back to the early days of elementary topos theory. The *Elementary theory of the category of sets* referred to in §10.1.5 was introduced by Lawvere in [Law05], as a category-theoretic axiomatization of set theory. The notion of ΠW-pretopos, regarded as a predicative version of an elementary topos, was introduced in [MP02]; see also [Pal09].

The treatment of the category of sets in §10.1 roughly follows that in [RS13]. The fact that epimorphisms are surjective (Lemma 10.1.4) is well known in classical mathematics, but is not as trivial as it may seem to prove *predicatively*. The proof in [MRR88] uses the power set operation (which is impredicative), although it can also be seen as a predicative proof of the weaker statement that a map in a universe $\mathcal{U}_i$ is surjective if it is an epimorphism in the next universe $\mathcal{U}_{i+1}$. A predicative proof for setoids was given by Wilander [Wil10]. Our proof is similar to Wilander's, but avoids setoids by using pushouts and univalence.

The implication in Theorem 10.1.14 from AC to LEM is an adaptation to homotopy type theory of a theorem from topos theory due to Diaconescu [Dia75]; it was posed as a problem already by Bishop [Bis67, Problem 2].

For the intuitionistic theory of ordinal numbers, see [Tay96, Tay99] and also [JM95]. Definitions of well-foundedness in type theory by an induction principle, including the inductive predicate of accessibility, were studied in [Hue80, Pau86, Nor88], although the idea dates back to Gentzen's proof of the consistency of arithmetic [Gen36].

The idea of algebraic set theory, which informs our development in §10.5 of the cumulative hierarchy, is due to [JM95], but it derives from earlier work by [Acz78].

Exercises

Exercise 10.1. Following the pattern of $\mathcal{S}et$, we would like to make a category $\mathcal{T}ype$ of all types and maps between them (in a given universe $\mathcal{U}$). In order for this to be a category in the sense of §9.1, however, we must first declare $\hom(X, Y) :\equiv \|X \to Y\|_0$, with composition defined by

induction on truncation from ordinary composition $(Y \to Z) \to (X \to Y) \to (X \to Z)$. This was defined as the *homotopy precategory of types* in Example 9.1.18. It is still not a category, however, but only a precategory (its type of objects $\mathcal{U}$ is not even a 0-type). It becomes a category by Rezk completion (see Example 9.9.7), and its type of objects can be identified with $\|\mathcal{U}\|_1$ by Exercise 9.9. Show that the resulting category $\mathcal{T}ype$, unlike $\mathcal{S}et$, is not a pretopos.

Exercise 10.2. Show that if every surjection has a section in the category $\mathcal{S}et$, then the axiom of choice holds.

Exercise 10.3. Show that with LEM, the category $\mathcal{S}et$ is well-pointed, in the sense that the following statement holds: for any $f, g : A \to B$, if $f \neq g$ then there is a function $a : 1 \to A$ such that $f(a) \neq g(a)$. Show that the slice category $\mathcal{S}et/\mathbf{2}$ consisting of functions $A \to \mathbf{2}$ and commutative triangles does not have this property. (Hint: the terminal object in $\mathcal{S}et/\mathbf{2}$ is the identity function $\mathbf{2} \to \mathbf{2}$, so in this category, there are objects X that have no elements $1 \to X$.)

Exercise 10.4. Prove that if $(A, <_A)$ and $(B, <_B)$ are well-founded, extensional, or ordinals, then so is $A + B$, with $<$ defined by

$$\begin{aligned}
(a < a') &:\equiv (a <_A a') && \text{for } a, a' : A \\
(b < b') &:\equiv (b <_B b') && \text{for } b, b' : B \\
(a < b) &:\equiv \mathbf{1} && \text{for } (a : A), (b : B) \\
(b < a) &:\equiv \mathbf{0} && \text{for } (a : A), (b : B).
\end{aligned}$$

Exercise 10.5. Prove that if $(A, <_A)$ and $(B, <_B)$ are well-founded, extensional, or ordinals, then so is $A \times B$, with $<$ defined by

$$((a, b) < (a', b')) :\equiv (a <_A a') \vee ((a = a') \wedge (b <_B b')).$$

Exercise 10.6. Define the usual algebraic operations on ordinals, and prove that they satisfy the usual properties.

Exercise 10.7. Note that $\mathbf{2}$ is an ordinal, under the obvious relation $<$ such that $0_{\mathbf{2}} < 1_{\mathbf{2}}$ only.

(i) Define a relation $<$ on Prop which makes it into an ordinal.
(ii) Show that $\mathbf{2} =_{\mathsf{Ord}} \mathsf{Prop}$ if and only if LEM holds.

Exercise 10.8. Recall that we denote $\mathbb{N}$ by ω when regarding it as an ordinal; thus we have also the ordinal $\omega + 1$. On the other hand, let us define

$$\mathbb{N}_\infty :\equiv \left\{ a : \mathbb{N} \to \mathbf{2} \;\middle|\; \forall (n : \mathbb{N}).\, (a_n \leq a_{\mathsf{succ}(n)}) \right\}$$

where $\leq$ denotes the obvious partial order on $\mathbf{2}$, with $0_{\mathbf{2}} \leq 1_{\mathbf{2}}$.

(i) Define a relation $<$ on $\mathbb{N}_\infty$ which makes it into an ordinal.
(ii) Show that $\omega + 1 =_{\mathsf{Ord}} \mathbb{N}_\infty$ if and only if the limited principle of omniscience (11.5.8) holds.

Exercise 10.9. Show that if $(A, <)$ is well-founded and extensional and $A : \mathcal{U}$, then there is a simulation $A \to V$, where $(V, \in)$ is the cumulative hierarchy from §10.5 built from the universe $\mathcal{U}$.

Exercise 10.10. Show that Theorem 10.4.4(i) is equivalent to the axiom of choice (3.8.1).

Exercise 10.11. Given types A and B, define a **bitotal relation** to be $R : A \to B \to \mathsf{Prop}$ such that

$$\Big(\forall (a : A).\, \exists (b : B).\, R(a, b)\Big) \wedge \Big(\forall (b : B).\, \exists (a : A).\, R(a, b)\Big).$$

For such A, B, R, let $A \sqcup^R B$ be the higher inductive type generated by

- $i : A \to A \sqcup^R B$
- $j : B \to A \sqcup^R B$
- For each $a : A$ and $b : B$ such that $R(a, b)$, a path $i(a) = j(b)$.

Show that the cumulative hierarchy V can be defined by the following more straightforward list of constructors, and that the resulting induction principle is the one given in §10.5.

- For every $A : \mathcal{U}$ and $f : A \to V$, there is an element $\mathsf{set}(A, f) : V$.
- For any $A, B : \mathcal{U}$ and bitotal relation $R : A \to B \to \mathsf{Prop}$, and any map $h : A \sqcup^R B \to V$, there is a path $\mathsf{set}(A, h \circ i) = \mathsf{set}(B, h \circ j)$.
- The 0-truncation constructor.

Exercise 10.12. In Constructive Zermelo–Fraenkel Set Theory, the **axiom of strong collection** has the form:

$$\Big(\forall(x \in v).\, \exists(y).\, R(x,y)\Big) \Rightarrow \\ \exists(w).\, \big[\big(\forall(x \in v).\, \exists(y \in w).\, R(x,y)\big) \wedge \big(\forall(y \in w).\, \exists(x \in v).\, R(x,y)\big)\big]$$

Does it hold in the cumulative hierarchy V? (We do not know the answer to this.)

Exercise 10.13. Verify that, if we assume AC, then the cumulative hierarchy V satisfies the usual set-theoretic axiom of choice, which may be stated in the form:

$$\forall(x : V).\, \Big((\forall(y \in x).\, \exists(z : V).\, z \in y) \Rightarrow \exists(c \in (\cup x)^x).\, \forall(y \in x).\, c(y) \in y\Big)$$

Exercise 10.14. Assuming propositional resizing, show that there is a mere predicate $\mathsf{isPlump} : \mathsf{Ord} \to \mathsf{Prop}$ such that for any $A : \mathsf{Ord}$ we have

$$\mathsf{isPlump}(A) = \Big(\forall(B < A).\, \mathsf{isPlump}(B)\Big) \wedge \Big(\forall(C, B : \mathsf{Ord}).\, C \leq B < A \wedge \mathsf{isPlump}(C) \Rightarrow C < A\Big).$$

Note that $\mathsf{isPlump}$ cannot be defined by a simple well-founded induction over Ord; you must use a different well-founded relation. We say that an ordinal A is **plump** [Tay96, Tay99] if $\mathsf{isPlump}(A)$.

Exercise 10.15. Show that LEM is equivalent to the statement "all ordinals are plump".

Exercise 10.16. Define the **plump successor** of an ordinal A to be

$$t(A) :\equiv \{\, B : \mathsf{Ord} \mid (B \leq A) \wedge \mathsf{isPlump}(B) \,\}$$

(i) By definition, $t(A)$ belongs to the next higher universe. Show that assuming propositional resizing, it is equal to an ordinal in the same universe as A.

(ii) Again assuming propositional resizing, show that if A is plump (Exercise 10.14) then so is $t(A)$.

Exercise 10.17. A **ZF-algebra** [JM95] relative to a universe $\mathcal{U}_i$ is a poset (see Example 9.1.14) $V : \mathcal{U}_{i+1}$, which has all suprema indexed by types in $\mathcal{U}_i$, and is equipped with a "successor" function $s : V \to V$ (not necessarily respecting $\leq$ in any way).

(i) Show that the cumulative hierarchy $(V_{\mathcal{U}_i}, \subseteq, s)$ is the initial ZF-algebra, where $s(x)$ is the singleton $\{\, x \,\}$.

(ii) Show that $(\mathsf{Ord}_{\mathcal{U}_i}, \leq, s)$ is the initial ZF-algebra with the property that $x \leq s(x)$ for all x, where $s(A) = A + \mathbf{1}$ is the successor from Lemma 10.3.21.

(iii) Assuming propositional resizing, show that $\left(\{\, A : \mathsf{Ord}_{\mathcal{U}_i} \mid \mathsf{isPlump}(A) \,\}, \leq, t\right)$ is the initial ZF-algebra with the property that $(x \leq y) \Rightarrow (t(x) \leq t(y))$ for all x, y, where t is the plump successor from Exercise 10.16.

Chapter 11

Real numbers

Any foundation of mathematics worthy of its name must eventually address the construction of real numbers as understood by mathematical analysis, namely as a complete archimedean ordered field. There are two notions of completeness. The one by Cauchy requires that the reals be closed under limits of Cauchy sequences, while the stronger one by Dedekind requires closure under Dedekind cuts. These lead to two ways of constructing reals, which we study in §11.2 and §11.3, respectively. In Theorems 11.2.14 and 11.3.50 we characterize the two constructions in terms of universal properties: the Dedekind reals are the final archimedean ordered field, and the Cauchy reals the initial Cauchy complete archimedean ordered field.

In traditional constructive mathematics, real numbers always seem to require certain compromises. For example, the Dedekind reals work better with power sets or some other form of impredicativity, while Cauchy reals work well in the presence of countable choice. However, we give a new construction of the Cauchy reals as a higher inductive-inductive type that seems to be a third possibility, which requires neither power sets nor countable choice.

In §11.4 we compare the two constructions of reals. The Cauchy reals are included in the Dedekind reals. They coincide if excluded middle or countable choice holds, but in general the inclusion might be proper.

In §11.5 we consider three notions of compactness of the closed interval $[0, 1]$. We first show that $[0, 1]$ is metrically compact in the sense that it is complete and totally bounded, and that uniformly continuous maps on metrically compact spaces behave as expected. In contrast, the Bolzano–Weierstraß property that every sequence has a convergent subsequence implies the limited principle of omniscience, which is an instance of excluded middle. Finally, we discuss Heine–Borel compactness. A naive formulation of the finite subcover property does not work, but a proof relevant notion of inductive covers does. This section is basically standard constructive analysis.

The development of real numbers and analysis in homotopy type theory can be easily made compatible with classical mathematics. By assuming excluded middle and the axiom of choice we get standard classical analysis: the Dedekind and Cauchy reals coincide, foundational questions about the impredicative nature of the Dedekind reals disappear, and the interval is as compact as it could be.

We close the chapter by constructing Conway's surreals as a higher inductive-inductive type in §11.6; the construction is more natural in univalent type theory than in classical set theory.

In addition to the basic theory of Chapters 2 and 3, as noted above we use "higher inductive-inductive types" for the Cauchy reals and the surreals: these combine the ideas of Chapter 6

with the notion of inductive-inductive type mentioned in §5.7. We will also frequently use the traditional logical notation described in §3.7, and the fact (proven in §10.1) that our "sets" behave the way we would expect.

Note that the total space of the universal cover of the circle, which in §8.1.5 played a role similar to "the real numbers" in classical algebraic topology, is *not* the type of reals we are looking for. That type is contractible, and thus equivalent to the singleton type, so it cannot be equipped with a non-trivial algebraic structure.

11.1 The field of rational numbers

We first construct the rational numbers $\mathbb{Q}$, as the reals can then be seen as a completion of $\mathbb{Q}$. An expert will point out that $\mathbb{Q}$ could be replaced by any approximate field, i.e., a subring of $\mathbb{Q}$ in which arbitrarily precise approximate inverses exist. An example is the ring of dyadic rationals, which are those of the form $n/2^k$. If we were implementing constructive mathematics on a computer, an approximate field would be more suitable, but we leave such finesse for those who care about the digits of π.

We constructed the integers $\mathbb{Z}$ in §6.10 as a quotient of $\mathbb{N} \times \mathbb{N}$, and observed that this quotient is generated by an idempotent. In §6.11 we saw that $\mathbb{Z}$ is the free group on $\mathbf{1}$; we could similarly show that it is the free commutative ring on $\mathbf{0}$. The field of rationals $\mathbb{Q}$ is constructed along the same lines as well, namely as the quotient

$$\mathbb{Q} :\equiv (\mathbb{Z} \times \mathbb{N})/\!\approx$$

where

$$(u, a) \approx (v, b) :\equiv (u(b+1) = v(a+1)).$$

In other words, a pair (u, a) represents the rational number $u/(1+a)$. There can be no division by zero because we cunningly added one to the denominator a. Here too we have a canonical choice of representatives, namely fractions in lowest terms. Thus we may apply Lemma 6.10.8 to obtain a set $\mathbb{Q}$, which again has a decidable equality.

We do not bother to write down the arithmetical operations on $\mathbb{Q}$ as we trust that our readers know how to compute with fractions even in the case when one is added to the denominator. Let us just record the conclusion that there is an entirely unproblematic construction of the ordered field of rational numbers $\mathbb{Q}$, with a decidable equality and decidable order. It can also be characterized as the initial ordered field.

Let $\mathbb{Q}_+ = \{\, q : \mathbb{Q} \mid q > 0 \,\}$ be the type of positive rational numbers.

11.2 Dedekind reals

Let us first recall the basic idea of Dedekind's construction. We use two-sided Dedekind cuts, as opposed to an often used one-sided version, because the symmetry makes constructions more elegant, and it works constructively as well as classically. A *Dedekind cut* consists of a pair (L, U) of subsets $L, U \subseteq \mathbb{Q}$, called the *lower* and *upper cut* respectively, which are:

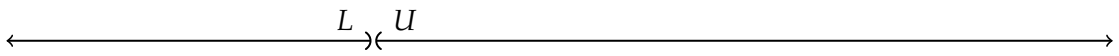
(i) *inhabited:* there are $q \in L$ and $r \in U$,

(ii) *rounded:* $q \in L \Leftrightarrow \exists(r \in \mathbb{Q}).\, q < r \wedge r \in L$ and $r \in U \Leftrightarrow \exists(q \in \mathbb{Q}).\, q \in U \wedge q < r$,

(iii) *disjoint:* $\neg(q \in L \wedge q \in U)$, and

(iv) *located:* $q < r \Rightarrow q \in L \vee r \in U$.

Reading the roundedness condition from left to right tells us that cuts are *open*, and from right to left that they are *lower*, respectively *upper*, sets. The locatedness condition states that there is no large gap between L and U. Because cuts are always open, they never include the "point in between", even when it is rational. A typical Dedekind cut looks like this:



We might naively translate the informal definition into type theory by saying that a cut is a pair of maps $L, U : \mathbb{Q} \to \mathsf{Prop}$. But we saw in §3.5 that Prop is an ambiguous notation for $\mathsf{Prop}_{\mathcal{U}_i}$ where $\mathcal{U}_i$ is a universe. Once we use a particular $\mathcal{U}_i$ to define cuts, the type of reals will reside in the next universe $\mathcal{U}_{i+1}$, a property of reals two levels higher in $\mathcal{U}_{i+2}$, a property of subsets of reals in $\mathcal{U}_{i+3}$, etc. In principle we should be able to keep track of the universe levels, especially with the help of a proof assistant, but doing so here would just burden us with bureaucracy that we prefer to avoid. We shall therefore make a simplifying assumption that a single type of propositions Ω is sufficient for all our purposes.

In fact, the construction of the Dedekind reals is quite resilient to logical manipulations. There are several ways in which we can make sense of using a single type Ω:

(i) We could identify Ω with the ambiguous Prop and track all the universes that appear in definitions and constructions.

(ii) We could assume the propositional resizing axiom, as in §3.5, which essentially collapses the $\mathsf{Prop}_{\mathcal{U}_i}$'s to the lowest level, which we call Ω.

(iii) A classical mathematician who is not interested in the intricacies of type-theoretic universes or computation may simply assume the law of excluded middle (3.4.1) for mere propositions so that $\Omega \equiv \mathbf{2}$. This not only eradicates questions about levels of Prop, but also turns everything we do into the standard classical construction of real numbers.

(iv) On the other end of the spectrum one might ask for a minimal requirement that makes the constructions work. The condition that a mere predicate be a Dedekind cut is expressible using only conjunctions, disjunctions, and existential quantifiers over $\mathbb{Q}$, which is a countable set. Thus we could take Ω to be the initial *σ-frame*, i.e., a lattice with countable joins in which binary meets distribute over countable joins. (The initial σ-frame cannot be the two-point lattice $\mathbf{2}$ because $\mathbf{2}$ is not closed under countable joins, unless we assume excluded middle.) This would lead to a construction of Ω as a higher inductive-inductive type, but one experiment of this kind in §11.3 is enough.

In all of the above cases Ω is a set. Without further ado, we translate the informal definition into type theory. Throughout this chapter, we use the logical notation from Definition 3.7.1.

Definition 11.2.1. A **Dedekind cut** is a pair (L, U) of mere predicates $L : \mathbb{Q} \to \Omega$ and $U : \mathbb{Q} \to \Omega$ which is:

(i) *inhabited:* $\exists(q : \mathbb{Q}).\, L(q)$ and $\exists(r : \mathbb{Q}).\, U(r)$,

(ii) *rounded:* for all $q, r : \mathbb{Q}$,

$$L(q) \Leftrightarrow \exists(r : \mathbb{Q}).\, (q < r) \wedge L(r) \qquad \text{and}$$
$$U(r) \Leftrightarrow \exists(q : \mathbb{Q}).\, (q < r) \wedge U(q),$$

(iii) *disjoint:* $\neg(L(q) \wedge U(q))$ for all $q : \mathbb{Q}$,
(iv) *located:* $(q < r) \Rightarrow L(q) \vee U(r)$ for all $q, r : \mathbb{Q}$.

We let $\mathsf{isCut}(L, U)$ denote the conjunction of these conditions. The type of **Dedekind reals** is

$$\mathbb{R}_{\mathsf{d}} :\equiv \{\, (L, U) : (\mathbb{Q} \to \Omega) \times (\mathbb{Q} \to \Omega) \mid \mathsf{isCut}(L, U) \,\}.$$

It is apparent that $\mathsf{isCut}(L, U)$ is a mere proposition, and since $\mathbb{Q} \to \Omega$ is a set the Dedekind reals form a set too. See Exercises 11.2 to 11.4 for variants of Dedekind cuts which lead to extended reals, lower and upper reals, and the interval domain.

There is an embedding $\mathbb{Q} \to \mathbb{R}_{\mathsf{d}}$ which associates with each rational $q : \mathbb{Q}$ the cut (L_q, U_q) where

$$L_q(r) :\equiv (r < q) \qquad \text{and} \qquad U_q(r) :\equiv (q < r).$$

We shall simply write q for the cut (L_q, U_q) associated with a rational number.

11.2.1 The algebraic structure of Dedekind reals

The construction of the algebraic and order-theoretic structure of Dedekind reals proceeds as usual in intuitionistic logic. Rather than dwelling on details we point out the differences between the classical and intuitionistic setup. Writing L_x and U_x for the lower and upper cut of a real number $x : \mathbb{R}_{\mathsf{d}}$, we define addition as

$$\begin{aligned} L_{x+y}(q) &:\equiv \exists(r, s : \mathbb{Q}).\, L_x(r) \wedge L_y(s) \wedge q = r + s, \\ U_{x+y}(q) &:\equiv \exists(r, s : \mathbb{Q}).\, U_x(r) \wedge U_y(s) \wedge q = r + s, \end{aligned}$$

and the additive inverse by

$$\begin{aligned} L_{-x}(q) &:\equiv \exists(r : \mathbb{Q}).\, U_x(r) \wedge q = -r, \\ U_{-x}(q) &:\equiv \exists(r : \mathbb{Q}).\, L_x(r) \wedge q = -r. \end{aligned}$$

With these operations $(\mathbb{R}_{\mathsf{d}}, 0, +, -)$ is an abelian group. Multiplication is a bit more cumbersome:

$$\begin{aligned} L_{x\cdot y}(q) :\equiv\ & \exists(a, b, c, d : \mathbb{Q}).\, L_x(a) \wedge U_x(b) \wedge L_y(c) \wedge U_y(d) \wedge \\ & \qquad q < \min(a \cdot c, a \cdot d, b \cdot c, b \cdot d), \\ U_{x\cdot y}(q) :\equiv\ & \exists(a, b, c, d : \mathbb{Q}).\, L_x(a) \wedge U_x(b) \wedge L_y(c) \wedge U_y(d) \wedge \\ & \qquad \max(a \cdot c, a \cdot d, b \cdot c, b \cdot d) < q. \end{aligned}$$

These formulas are related to multiplication of intervals in interval arithmetic, where intervals $[a, b]$ and $[c, d]$ with rational endpoints multiply to the interval

$$[a, b] \cdot [c, d] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)].$$

For instance, the formula for the lower cut can be read as saying that $q < x \cdot y$ when there are intervals $[a, b]$ and $[c, d]$ containing x and y, respectively, such that q is to the left of $[a, b] \cdot [c, d]$. It is generally useful to think of an interval $[a, b]$ such that $L_x(a)$ and $U_x(b)$ as an approximation of x, see Exercise 11.4.

We now have a commutative ring with unit $(\mathbb{R}_{\mathsf{d}}, 0, 1, +, -, \cdot)$. To treat multiplicative inverses, we must first introduce order. Define $\leq$ and $<$ as

$$\begin{aligned} (x \leq y) &:\equiv \forall(q : \mathbb{Q}).\, L_x(q) \Rightarrow L_y(q), \\ (x < y) &:\equiv \exists(q : \mathbb{Q}).\, U_x(q) \wedge L_y(q). \end{aligned}$$

Lemma 11.2.2. *For all $x : \mathbb{R}_{\mathsf{d}}$ and $q : \mathbb{Q}$, $L_x(q) \Leftrightarrow (q < x)$ and $U_x(q) \Leftrightarrow (x < q)$.*

Proof. If $L_x(q)$ then by roundedness there merely is $r > q$ such that $L_x(r)$, and since $U_q(r)$ it follows that $q < x$. Conversely, if $q < x$ then there is $r : \mathbb{Q}$ such that $U_q(r)$ and $L_x(r)$, hence $L_x(q)$ because L_x is a lower set. The other half of the proof is symmetric. □

The relation $\leq$ is a partial order, and $<$ is transitive and irreflexive. Linearity

$$(x < y) \vee (y \leq x)$$

is valid if we assume excluded middle, but without it we get weak linearity

$$(x < y) \Rightarrow (x < z) \vee (z < y). \tag{11.2.3}$$

At first sight it might not be clear what (11.2.3) has to do with linear order. But if we take $x \equiv u - \epsilon$ and $y \equiv u + \epsilon$ for $\epsilon > 0$, then we get

$$(u - \epsilon < z) \vee (z < u + \epsilon).$$

This is linearity "up to a small numerical error", i.e., since it is unreasonable to expect that we can actually compute with infinite precision, we should not be surprised that we can decide $<$ only up to whatever finite precision we have computed.

To see that (11.2.3) holds, suppose $x < y$. Then there merely exists $q : \mathbb{Q}$ such that $U_x(q)$ and $L_y(q)$. By roundedness there merely exist $r, s : \mathbb{Q}$ such that $r < q < s$, $U_x(r)$ and $L_y(s)$. Then, by locatedness $L_z(r)$ or $U_z(s)$. In the first case we get $x < z$ and in the second $z < y$.

Classically, multiplicative inverses exist for all numbers which are different from zero. However, without excluded middle, a stronger condition is required. Say that $x, y : \mathbb{R}_{\mathsf{d}}$ are **apart** from each other, written $x \# y$, when $(x < y) \vee (y < x)$:

$$(x \# y) :\equiv (x < y) \vee (y < x).$$

If $x \# y$, then $\neg(x = y)$. The converse is true if we assume excluded middle, but is not provable constructively. Indeed, if $\neg(x = y)$ implies $x \# y$, then a little bit of excluded middle follows; see Exercise 11.10.

Theorem 11.2.4. *A real is invertible if, and only if, it is apart from* 0.

Remark 11.2.5. We observe that a real is invertible if, and only if, it is merely invertible. Indeed, the same is true in any ring, since a ring is a set, and multiplicative inverses are unique if they exist. See the discussion following Corollary 3.9.2.

Proof. Suppose $x \cdot y = 1$. Then there merely exist $a, b, c, d : \mathbb{Q}$ such that $a < x < b$, $c < y < d$ and $0 < \min(ac, ad, bc, bd)$. From $0 < ac$ and $0 < bc$ it follows that a, b, and c are either all positive or all negative. Hence either $0 < a < x$ or $x < b < 0$, so that $x \# 0$.

Conversely, if $x \# 0$ then

$$\begin{aligned} L_{x^{-1}}(q) &:\equiv \exists(r : \mathbb{Q}).\, U_x(r) \wedge ((0 < r \wedge qr < 1) \vee (r < 0 \wedge 1 < qr)) \\ U_{x^{-1}}(q) &:\equiv \exists(r : \mathbb{Q}).\, L_x(r) \wedge ((0 < r \wedge qr > 1) \vee (r < 0 \wedge 1 > qr)) \end{aligned}$$

defines the desired inverse. Indeed, $L_{x^{-1}}$ and $U_{x^{-1}}$ are inhabited because $x \# 0$. □

The archimedean principle can be stated in several ways. We find it most illuminating in the form which says that $\mathbb{Q}$ is dense in $\mathbb{R}_\mathsf{d}$.

Theorem 11.2.6 (Archimedean principle for $\mathbb{R}_\mathsf{d}$)**.** *For all $x, y : \mathbb{R}_\mathsf{d}$ if $x < y$ then there merely exists $q : \mathbb{Q}$ such that $x < q < y$.*

Proof. By definition of $<$. □

Before tackling completeness of Dedekind reals, let us state precisely what algebraic structure they possess. In the following definition we are not aiming at a minimal axiomatization, but rather at a useful amount of structure and properties.

Definition 11.2.7. An **ordered field** is a set F together with constants 0, 1, operations $+$, $-$, $\cdot$, min, max, and mere relations $\leq$, $<$, $\#$ such that:

(i) $(F, 0, 1, +, -, \cdot)$ is a commutative ring with unit;
(ii) $x : F$ is invertible if, and only if, $x \# 0$;
(iii) $(F, \leq, \min, \max)$ is a lattice;
(iv) the strict order $<$ is transitive, irreflexive, and weakly linear ($x < y \Rightarrow x < z \vee z < y$);
(v) apartness $\#$ is irreflexive, symmetric and cotransitive ($x \# y \Rightarrow x \# z \vee y \# z$);
(vi) for all $x, y, z : F$:

$$\begin{aligned}
x \leq y &\Leftrightarrow \neg(y < x), & x < y \leq z &\Rightarrow x < z,\\
x \# y &\Leftrightarrow (x < y) \vee (y < x), & x \leq y < z &\Rightarrow x < z,\\
x \leq y &\Leftrightarrow x + z \leq y + z, & x \leq y \wedge 0 \leq z &\Rightarrow xz \leq yz,\\
x < y &\Leftrightarrow x + z < y + z, & 0 < z &\Rightarrow (x < y \Leftrightarrow xz < yz),\\
0 < x + y &\Rightarrow 0 < x \vee 0 < y, & 0 &< 1.
\end{aligned}$$

Every such field has a canonical embedding $\mathbb{Q} \to F$. An ordered field is **archimedean** when for all $x, y : F$, if $x < y$ then there merely exists $q : \mathbb{Q}$ such that $x < q < y$.

Theorem 11.2.8. *The Dedekind reals form an ordered archimedean field.*

Proof. We omit the proof in the hope that what we have demonstrated so far makes the theorem plausible. □

11.2.2 Dedekind reals are Cauchy complete

Recall that $x : \mathbb{N} \to \mathbb{Q}$ is a *Cauchy sequence* when it satisfies

$$\prod_{(\epsilon:\mathbb{Q}_+)} \sum_{(n:\mathbb{N})} \prod_{(m,k \geq n)} |x_m - x_k| < \epsilon. \tag{11.2.9}$$

Note that we did *not* truncate the inner existential because we actually want to compute rates of convergence—an approximation without an error estimate carries little useful information. By Theorem 2.15.7, (11.2.9) yields a function $M : \mathbb{Q}_+ \to \mathbb{N}$, called the *modulus of convergence*, such that $m, k \geq M(\epsilon)$ implies $|x_m - x_k| < \epsilon$. From this we get $|x_{M(\delta/2)} - x_{M(\epsilon/2)}| < \delta + \epsilon$ for all $\delta, \epsilon : \mathbb{Q}_+$. In fact, the map $(\epsilon \mapsto x_{M(\epsilon/2)}) : \mathbb{Q}_+ \to \mathbb{Q}$ carries the same information about the limit as the original Cauchy condition (11.2.9). We shall work with these approximation functions rather than with Cauchy sequences.

Definition 11.2.10. A **Cauchy approximation** is a map $x : \mathbb{Q}_+ \to \mathbb{R}_\mathsf{d}$ which satisfies

$$\forall(\delta, \epsilon : \mathbb{Q}_+).\, |x_\delta - x_\epsilon| < \delta + \epsilon. \tag{11.2.11}$$

The **limit** of a Cauchy approximation $x : \mathbb{Q}_+ \to \mathbb{R}_\mathsf{d}$ is a number $\ell : \mathbb{R}_\mathsf{d}$ such that

$$\forall(\epsilon, \theta : \mathbb{Q}_+).\, |x_\epsilon - \ell| < \epsilon + \theta.$$

Theorem 11.2.12. *Every Cauchy approximation in $\mathbb{R}_\mathsf{d}$ has a limit.*

Proof. Note that we are showing existence, not mere existence, of the limit. Given a Cauchy approximation $x : \mathbb{Q}_+ \to \mathbb{R}_\mathsf{d}$, define

$$L_y(q) :\equiv \exists(\epsilon, \theta : \mathbb{Q}_+).\, L_{x_\epsilon}(q + \epsilon + \theta),$$
$$U_y(q) :\equiv \exists(\epsilon, \theta : \mathbb{Q}_+).\, U_{x_\epsilon}(q - \epsilon - \theta).$$

It is clear that L_y and U_y are inhabited, rounded, and disjoint. To establish locatedness, consider any $q, r : \mathbb{Q}$ such that $q < r$. There is $\epsilon : \mathbb{Q}_+$ such that $5\epsilon < r - q$. Since $q + 2\epsilon < r - 2\epsilon$ merely $L_{x_\epsilon}(q + 2\epsilon)$ or $U_{x_\epsilon}(r - 2\epsilon)$. In the first case we have $L_y(q)$ and in the second $U_y(r)$.

To show that y is the limit of x, consider any $\epsilon, \theta : \mathbb{Q}_+$. Because $\mathbb{Q}$ is dense in $\mathbb{R}_\mathsf{d}$ there merely exist $q, r : \mathbb{Q}$ such that

$$x_\epsilon - \epsilon - \theta/2 < q < x_\epsilon - \epsilon - \theta/4 < x_\epsilon < x_\epsilon + \epsilon + \theta/4 < r < x_\epsilon + \epsilon + \theta/2,$$

and thus $q < y < r$. Now either $y < x_\epsilon + \theta/2$ or $x_\epsilon - \theta/2 < y$. In the first case we have

$$x_\epsilon - \epsilon - \theta/2 < q < y < x_\epsilon + \theta/2,$$

and in the second

$$x_\epsilon - \theta/2 < y < r < x_\epsilon + \epsilon + \theta/2.$$

In either case it follows that $|y - x_\epsilon| < \epsilon + \theta$. □

For sake of completeness we record the classic formulation as well.

Corollary 11.2.13. *Suppose $x : \mathbb{N} \to \mathbb{R}_\mathsf{d}$ satisfies the Cauchy condition* (11.2.9)*. Then there exists $y : \mathbb{R}_\mathsf{d}$ such that*

$$\prod_{(\epsilon:\mathbb{Q}_+)} \sum_{(n:\mathbb{N})} \prod_{(m \geq n)} |x_m - y| < \epsilon.$$

Proof. By Theorem 2.15.7 there is $M : \mathbb{Q}_+ \to \mathbb{N}$ such that $\bar{x}(\epsilon) :\equiv x_{M(\epsilon/2)}$ is a Cauchy approximation. Let y be its limit, which exists by Theorem 11.2.12. Given any $\epsilon : \mathbb{Q}_+$, let $n :\equiv M(\epsilon/4)$ and observe that, for any $m \geq n$,

$$|x_m - y| \leq |x_m - x_n| + |x_n - y| = |x_m - x_n| + |\bar{x}(\epsilon/2) - y| < \epsilon/4 + \epsilon/2 + \epsilon/4 = \epsilon. \qquad \square$$

11.2.3 Dedekind reals are Dedekind complete

We obtained $\mathbb{R}_\mathsf{d}$ as the type of Dedekind cuts on $\mathbb{Q}$. But we could have instead started with any archimedean ordered field F and constructed Dedekind cuts on F. These would again form an archimedean ordered field $\bar{F}$, the **Dedekind completion of** F, with F contained as a subfield. What happens if we apply this construction to $\mathbb{R}_\mathsf{d}$, do we get even more real numbers? The answer is negative. In fact, we shall prove a stronger result: $\mathbb{R}_\mathsf{d}$ is final.

Say that an ordered field F is **admissible for** Ω when the strict order $<$ on F is a map $< : F \to F \to \Omega$.

Theorem 11.2.14. *Every archimedean ordered field which is admissible for Ω is a subfield of $\mathbb{R}_\mathsf{d}$.*

Proof. Let F be an archimedean ordered field. For every $x : F$ define $L_x, U_x : \mathbb{Q} \to \Omega$ by

$$L_x(q) :\equiv (q < x) \qquad \text{and} \qquad U_x(q) :\equiv (x < q).$$

(We have just used the assumption that F is admissible for Ω.) Then (L_x, U_x) is a Dedekind cut. Indeed, the cuts are inhabited and rounded because F is archimedean and $<$ is transitive, disjoint because $<$ is irreflexive, and located because $<$ is a weak linear order. Let $e : F \to \mathbb{R}_\mathsf{d}$ be the map $e(x) :\equiv (L_x, U_x)$.

We claim that e is a field embedding which preserves and reflects the order. First of all, notice that $e(q) = q$ for a rational number q. Next we have the equivalences, for all $x, y : F$,

$$x < y \Leftrightarrow (\exists(q : \mathbb{Q}).\, x < q < y) \Leftrightarrow (\exists(q : \mathbb{Q}).\, U_x(q) \wedge L_y(q)) \Leftrightarrow e(x) < e(y),$$

so e indeed preserves and reflects the order. That $e(x + y) = e(x) + e(y)$ holds because, for all $q : \mathbb{Q}$,

$$q < x + y \Leftrightarrow \exists(r, s : \mathbb{Q}).\, r < x \wedge s < y \wedge q = r + s.$$

The implication from right to left is obvious. For the other direction, if $q < x + y$ then there merely exists $r : \mathbb{Q}$ such that $q - y < r < x$, and by taking $s :\equiv q - r$ we get the desired r and s. We leave preservation of multiplication by e as an exercise. □

To establish that the Dedekind cuts on $\mathbb{R}_\mathsf{d}$ do not give us anything new, we need just one more lemma.

Lemma 11.2.15. *If F is admissible for Ω then so is its Dedekind completion.*

Proof. Let $\bar{F}$ be the Dedekind completion of F. The strict order on $\bar{F}$ is defined by

$$((L, U) < (L', U')) :\equiv \exists(q : \mathbb{Q}).\, U(q) \wedge L'(q).$$

Since $U(q)$ and $L'(q)$ are elements of Ω, the lemma holds as long as Ω is closed under conjunctions and countable existentials, which we assumed from the outset. □

Corollary 11.2.16. *The Dedekind reals are Dedekind complete: for every real-valued Dedekind cut (L, U) there is a unique $x : \mathbb{R}_\mathsf{d}$ such that $L(y) = (y < x)$ and $U(y) = (x < y)$ for all $y : \mathbb{R}_\mathsf{d}$.*

Proof. By Lemma 11.2.15 the Dedekind completion $\bar{\mathbb{R}}_\mathsf{d}$ of $\mathbb{R}_\mathsf{d}$ is admissible for Ω, so by Theorem 11.2.14 we have an embedding $\bar{\mathbb{R}}_\mathsf{d} \to \mathbb{R}_\mathsf{d}$, as well as an embedding $\mathbb{R}_\mathsf{d} \to \bar{\mathbb{R}}_\mathsf{d}$. But these embeddings must be isomorphisms, because their compositions are order-preserving field homomorphisms which fix the dense subfield $\mathbb{Q}$, which means that they are the identity. The corollary now follows immediately from the fact that $\bar{\mathbb{R}}_\mathsf{d} \to \mathbb{R}_\mathsf{d}$ is an isomorphism. □

11.3 Cauchy reals

The Cauchy reals are, by intent, the completion of $\mathbb{Q}$ under limits of Cauchy sequences. In the classical construction of the Cauchy reals, we consider the set $\mathcal{C}$ of all Cauchy sequences in $\mathbb{Q}$ and then form a suitable quotient $\mathcal{C}/\!\approx$. Then, to show that $\mathcal{C}/\!\approx$ is Cauchy complete, we consider a Cauchy sequence $x : \mathbb{N} \to \mathcal{C}/\!\approx$, lift it to a sequence of sequences $\bar{x} : \mathbb{N} \to \mathcal{C}$, and construct the limit of x using $\bar{x}$. However, the lifting of x to $\bar{x}$ uses the axiom of countable choice (the instance

of (3.8.1) where $X = \mathbb{N}$) or the law of excluded middle, which we may wish to avoid. Every construction of reals whose last step is a quotient suffers from this deficiency. There are three common ways out of the conundrum in constructive mathematics:

(i) Pretend that the reals are a setoid $(\mathcal{C}, \approx)$, i.e., the type of Cauchy sequences $\mathcal{C}$ with a coincidence relation attached to it by administrative decree. A sequence of reals then simply *is* a sequence of Cauchy sequences representing them.

(ii) Give in to temptation and accept the axiom of countable choice. After all, the axiom is valid in most models of constructive mathematics based on a computational viewpoint, such as realizability models.

(iii) Declare the Cauchy reals unworthy and construct the Dedekind reals instead. Such a verdict is perfectly valid in certain contexts, such as in sheaf-theoretic models of constructive mathematics. However, as we saw in §11.2, the constructive Dedekind reals have their own problems.

Using higher inductive types, however, there is a fourth solution, which we believe to be preferable to any of the above, and interesting even to a classical mathematician. The idea is that the Cauchy real numbers should be the *free complete metric space* generated by $\mathbb{Q}$. In general, the construction of a free gadget of any sort requires applying the gadget operations repeatedly many times to the generators. For instance, the elements of the free group on a set X are not just binary products and inverses of elements of X, but words obtained by iterating the product and inverse constructions. Thus, we might naturally expect the same to be true for Cauchy completion, with the relevant "operation" being "take the limit of a Cauchy sequence". (In this case, the iteration would have to take place transfinitely, since even after infinitely many steps there will be new Cauchy sequences to take the limit of.)

The argument referred to above shows that if excluded middle or countable choice hold, then Cauchy completion is very special: when building the completion of a space, it suffices to stop applying the operation after *one step*. This may be regarded as analogous to the fact that free monoids and free groups can be given explicit descriptions in terms of (reduced) words. However, we saw in §6.11 that higher inductive types allow us to construct free gadgets *directly*, whether or not there is also an explicit description available. In this section we show that the same is true for the Cauchy reals (a similar technique would construct the Cauchy completion of any metric space; see Exercise 11.9). Specifically, higher inductive types allow us to *simultaneously* add limits of Cauchy sequences and quotient by the coincidence relation, so that we can avoid the problem of lifting a sequence of reals to a sequence of representatives.

11.3.1 Construction of Cauchy reals

The construction of the Cauchy reals $\mathbb{R}_c$ as a higher inductive type is a bit more subtle than that of the free algebraic structures considered in §6.11. We intend to include a "take the limit" constructor whose input is a Cauchy sequence of reals, but the notion of "Cauchy sequence of reals" depends on having some way to measure the "distance" between real numbers. In general, of course, the distance between two real numbers will be another real number, leading to a potentially problematic circularity.

However, what we actually need for the notion of Cauchy sequence of reals is not the general notion of "distance", but a way to say that "the distance between two real numbers is less than ϵ" for any $\epsilon : \mathbb{Q}_+$. This can be represented by a family of binary relations, which we will denote $\sim_\epsilon : \mathbb{R}_c \to \mathbb{R}_c \to \mathsf{Prop}$. The intended meaning of $x \sim_\epsilon y$ is $|x - y| < \epsilon$, but since we do not

have notions of subtraction, absolute value, or inequality available yet (we are only just defining $\mathbb{R}_c$, after all), we will have to define these relations $\sim_\epsilon$ at the same time as we define $\mathbb{R}_c$ itself. And since $\sim_\epsilon$ is a type family indexed by two copies of $\mathbb{R}_c$, we cannot do this with an ordinary mutual (higher) inductive definition; instead we have to use a *higher inductive-inductive definition*.

Recall from §5.7 that the ordinary notion of inductive-inductive definition allows us to define a type and a type family indexed by it by simultaneous induction. Of course, the "higher" version of this allows both the type and the family to have path constructors as well as point constructors. We will not attempt to formulate any general theory of higher inductive-inductive definitions, but hopefully the description we will give of $\mathbb{R}_c$ and $\sim_\epsilon$ will make the idea transparent.

Remark 11.3.1. We might also consider a *higher inductive-recursive definition*, in which $\sim_\epsilon$ is defined using the *recursion* principle of $\mathbb{R}_c$, simultaneously with the *inductive* definition of $\mathbb{R}_c$. We choose the inductive-inductive route instead for two reasons. Firstly, higher inductive-recursive definitions seem to be more difficult to justify in homotopical semantics. Secondly, and more importantly, the inductive-inductive definition yields a more powerful induction principle, which we will need in order to develop even the basic theory of Cauchy reals.

Finally, as we did for the discussion of Cauchy completeness of the Dedekind reals in §11.2.2, we will work with *Cauchy approximations* (Definition 11.2.10) instead of Cauchy sequences. Of course, our Cauchy approximations will now consist of Cauchy reals, rather than Dedekind reals or rational numbers.

Definition 11.3.2. Let $\mathbb{R}_c$ and the relation $\sim : \mathbb{Q}_+ \times \mathbb{R}_c \times \mathbb{R}_c \to \mathcal{U}$ be the following higher inductive-inductive type family. The type $\mathbb{R}_c$ of **Cauchy reals** is generated by the following constructors:

- *rational points:* for any $q : \mathbb{Q}$ there is a real $\mathsf{rat}(q)$.
- *limit points*: for any $x : \mathbb{Q}_+ \to \mathbb{R}_c$ such that

$$\forall(\delta, \epsilon : \mathbb{Q}_+).\, x_\delta \sim_{\delta+\epsilon} x_\epsilon \tag{11.3.3}$$

  there is a point $\mathsf{lim}(x) : \mathbb{R}_c$. We call x a **Cauchy approximation**.
- *paths:* for $u, v : \mathbb{R}_c$ such that

$$\forall(\epsilon : \mathbb{Q}_+).\, u \sim_\epsilon v \tag{11.3.4}$$

  then there is a path $\mathsf{eq}_{\mathbb{R}_c}(u, v) : u =_{\mathbb{R}_c} v$.

Simultaneously, the type family $\sim : \mathbb{R}_c \to \mathbb{R}_c \to \mathbb{Q}_+ \to \mathcal{U}$ is generated by the following constructors. Here q and r denote rational numbers; δ, ϵ, and η denote positive rationals; u and v denote Cauchy reals; and x and y denote Cauchy approximations:

- for any q, r, ϵ, if $-\epsilon < q - r < \epsilon$, then $\mathsf{rat}(q) \sim_\epsilon \mathsf{rat}(r)$,
- for any q, y, ϵ, δ, if $\mathsf{rat}(q) \sim_{\epsilon-\delta} y_\delta$, then $\mathsf{rat}(q) \sim_\epsilon \mathsf{lim}(y)$,
- for any x, r, ϵ, δ, if $x_\delta \sim_{\epsilon-\delta} \mathsf{rat}(r)$, then $\mathsf{lim}(x) \sim_\epsilon \mathsf{rat}(r)$,
- for any $x, y, \epsilon, \delta, \eta$, if $x_\delta \sim_{\epsilon-\delta-\eta} y_\eta$, then $\mathsf{lim}(x) \sim_\epsilon \mathsf{lim}(y)$,
- for any u, v, ϵ, if $\xi, \zeta : u \sim_\epsilon v$, then $\xi = \zeta$ (propositional truncation).

The first constructor of $\mathbb{R}_{\mathsf{c}}$ says that any rational number can be regarded as a real number. The second says that from any Cauchy approximation to a real number, we can obtain a new real number called its "limit". And the third expresses the idea that if two Cauchy approximations coincide, then their limits are equal.

The first four constructors of $\sim$ specify when two rational numbers are close, when a rational is close to a limit, and when two limits are close. In the case of two rational numbers, this is just the usual notion of ϵ-closeness for rational numbers, whereas the other cases can be derived by noting that each approximant x_δ is supposed to be within δ of the limit $\mathsf{lim}(x)$.

We remind ourselves of proof-relevance: a real number obtained from lim is represented not just by a Cauchy approximation x, but also a proof p of (11.3.3), so we should technically have written $\mathsf{lim}(x, p)$ instead of just $\mathsf{lim}(x)$. A similar observation also applies to $\mathsf{eq}_{\mathbb{R}_{\mathsf{c}}}$ and (11.3.4), but we shall write just $\mathsf{eq}_{\mathbb{R}_{\mathsf{c}}} : u = v$ instead of $\mathsf{eq}_{\mathbb{R}_{\mathsf{c}}}(u, v, p) : u = v$. These abuses of notation are mitigated by the fact that we are omitting mere propositions and information that is readily guessed. Likewise, the last constructor of $\sim_\epsilon$ justifies our leaving the other four nameless.

We are immediately able to populate $\mathbb{R}_{\mathsf{c}}$ with many real numbers. For suppose $x : \mathbb{N} \to \mathbb{Q}$ is a traditional Cauchy sequence of rational numbers, and let $M : \mathbb{Q}_+ \to \mathbb{N}$ be its modulus of convergence. Then $\mathsf{rat} \circ x \circ M : \mathbb{Q}_+ \to \mathbb{R}_{\mathsf{c}}$ is a Cauchy approximation, using the first constructor of $\sim$ to produce the necessary witness. Thus, $\mathsf{lim}(\mathsf{rat} \circ x \circ m)$ is a real number. Various famous real numbers such as $\sqrt{2}$, π, e, $\dots$ are all limits of such Cauchy sequences of rationals.

11.3.2 Induction and recursion on Cauchy reals

In order to do anything useful with $\mathbb{R}_{\mathsf{c}}$, of course, we need to give its induction principle. As is the case whenever we inductively define two or more objects at once, the basic induction principle for $\mathbb{R}_{\mathsf{c}}$ and $\sim$ requires a simultaneous induction over both at once. Thus, we should expect it to say that assuming two type families over $\mathbb{R}_{\mathsf{c}}$ and $\sim$, respectively, together with data corresponding to each constructor, there exist sections of both of these families. However, since $\sim$ is indexed on two copies of $\mathbb{R}_{\mathsf{c}}$, the precise dependencies of these families is a bit subtle. The induction principle will apply to any pair of type families:

$$\begin{aligned} &A : \mathbb{R}_{\mathsf{c}} \to \mathcal{U} \\ &B : \prod_{x,y:\mathbb{R}_{\mathsf{c}}} A(x) \to A(y) \to \prod_{\epsilon:\mathbb{Q}_+} (x \sim_\epsilon y) \to \mathcal{U}. \end{aligned}$$

The type of A is obvious, but the type of B requires a little thought. Since B must depend on $\sim$, but $\sim$ in turn depends on two copies of $\mathbb{R}_{\mathsf{c}}$ and one copy of $\mathbb{Q}_+$, it is fairly obvious that B must also depend on the variables $x, y : \mathbb{R}_{\mathsf{c}}$ and $\epsilon : \mathbb{Q}_+$ as well as an element of $(x \sim_\epsilon y)$. What is slightly less obvious is that B must also depend on $A(x)$ and $A(y)$.

This may be more evident if we consider the non-dependent case (the recursion principle), where A is a simple type (rather than a type family). In this case we would expect B not to depend on $x, y : \mathbb{R}_{\mathsf{c}}$ or $x \sim_\epsilon y$. But the recursion principle (along with its associated uniqueness principle) is supposed to say that $\mathbb{R}_{\mathsf{c}}$ with $\sim_\epsilon$ is an "initial object" in some category, so in this case the dependency structure of A and B should mirror that of $\mathbb{R}_{\mathsf{c}}$ and $\sim_\epsilon$: that is, we should have $B : A \to A \to \mathbb{Q}_+ \to \mathcal{U}$. Combining this observation with the fact that, in the dependent case, B must also depend on $x, y : \mathbb{R}_{\mathsf{c}}$ and $x \sim_\epsilon y$, leads inevitably to the type given above for B.

It is helpful to think of B as an ϵ-indexed family of relations between the types $A(x)$ and $A(y)$. With this in mind, we may write $B(x, y, a, b, \epsilon, \xi)$ as $(x, a) \frown_\epsilon^\xi (y, b)$. Since $\xi : x \sim_\epsilon y$ is unique when it exists, we generally omit it from the notation and write $(x, a) \frown_\epsilon (y, b)$; this is

harmless as long as we keep in mind that this relation is only defined when $x \sim_\epsilon y$. We may also sometimes simplify further and write $a \frown_\epsilon b$, with x and y inferred from the types of a and b, but sometimes it will be necessary to include them for clarity.

Now, given a type family $A : \mathbb{R}_{\mathsf{c}} \to \mathcal{U}$ and a family of relations $\frown$ as above, the hypotheses of the induction principle consist of the following data, one for each constructor of $\mathbb{R}_{\mathsf{c}}$ or $\sim$:

- For any $q : \mathbb{Q}$, an element $f_q : A(\mathsf{rat}(q))$.
- For any Cauchy approximation x, and any $a : \prod_{(\epsilon:\mathbb{Q}_+)} A(x_\epsilon)$ such that

$$\forall(\delta, \epsilon : \mathbb{Q}_+).\, (x_\delta, a_\delta) \frown_{\delta+\epsilon} (x_\epsilon, a_\epsilon), \tag{11.3.5}$$

  an element $f_{x,a} : A(\mathsf{lim}(x))$. We call such a a **dependent Cauchy approximation** over x.
- For $u, v : \mathbb{R}_{\mathsf{c}}$ such that $h : \forall(\epsilon : \mathbb{Q}_+).\, u \sim_\epsilon v$, and all $a : A(u)$ and $b : A(v)$ such that $\forall(\epsilon : \mathbb{Q}_+).\, (u, a) \frown_\epsilon (v, b)$, a dependent path $a =^A_{\mathsf{eq}_{\mathbb{R}_{\mathsf{c}}}(u,v)} b$.
- For $q, r : \mathbb{Q}$ and $\epsilon : \mathbb{Q}_+$, if $-\epsilon < q - r < \epsilon$, we have $(\mathsf{rat}(q), f_q) \frown_\epsilon (\mathsf{rat}(r), f_r)$.
- For $q : \mathbb{Q}$ and $\delta, \epsilon : \mathbb{Q}_+$ and y a Cauchy approximation, and b a dependent Cauchy approximation over y, if $\mathsf{rat}(q) \sim_{\epsilon-\delta} y_\delta$, then

$$(\mathsf{rat}(q), f_q) \frown_{\epsilon-\delta} (y_\delta, b_\delta) \;\Rightarrow\; (\mathsf{rat}(q), f_q) \frown_\epsilon (\mathsf{lim}(y), f_{y,b}).$$

- Similarly, for $r : \mathbb{Q}$ and $\delta, \epsilon : \mathbb{Q}_+$ and x a Cauchy approximation, and a a dependent Cauchy approximation over x, if $x_\delta \sim_{\epsilon-\delta} \mathsf{rat}(r)$, then

$$(x_\delta, a_\delta) \frown_{\epsilon-\delta} (\mathsf{rat}(r), f_r) \;\Rightarrow\; (\mathsf{lim}(x), f_{x,a}) \frown_\epsilon (\mathsf{rat}(q), f_r).$$

- For $\epsilon, \delta, \eta : \mathbb{Q}_+$ and x, y Cauchy approximations, and a and b dependent Cauchy approximations over x and y respectively, if we have $x_\delta \sim_{\epsilon-\delta-\eta} y_\eta$, then

$$(x_\delta, a_\delta) \frown_{\epsilon-\delta-\eta} (y_\eta, b_\eta) \;\Rightarrow\; (\mathsf{lim}(x), f_{x,a}) \frown_\epsilon (\mathsf{lim}(y), f_{y,b}).$$

- For $\epsilon : \mathbb{Q}_+$ and $x, y : \mathbb{R}_{\mathsf{c}}$ and $\xi, \zeta : x \sim_\epsilon y$, and $a : A(x)$ and $b : A(y)$, any two elements of $(x, a) \frown^\xi_\epsilon (y, b)$ and $(x, a) \frown^\zeta_\epsilon (y, b)$ are dependently equal over $\xi = \zeta$. Note that as usual, this is equivalent to asking that $\frown$ takes values in mere propositions.

Under these hypotheses, we deduce functions

$$f : \prod_{x:\mathbb{R}_{\mathsf{c}}} A(x)$$
$$g : \prod_{(x,y:\mathbb{R}_{\mathsf{c}})} \prod_{(\epsilon:\mathbb{Q}_+)} \prod_{(\xi:x\sim_\epsilon y)} (x, f(x)) \frown^\xi_\epsilon (y, f(y))$$

which compute as expected:

$$f(\mathsf{rat}(q)) :\equiv f_q, \tag{11.3.6}$$
$$f(\mathsf{lim}(x)) :\equiv f_{x,(f,g)[x]}. \tag{11.3.7}$$

Here $(f, g)[x]$ denotes the result of applying f and g to a Cauchy approximation x to obtain a dependent Cauchy approximation over x. That is, we define $(f, g)[x]_\epsilon :\equiv f(x_\epsilon) : A(x_\epsilon)$, and then for any $\epsilon, \delta : \mathbb{Q}_+$ we have $g(x_\epsilon, x_\delta, \epsilon + \delta, \xi)$ to witness the fact that $(f, g)[x]$ is a dependent

Cauchy approximation, where $\xi : x_\epsilon \sim_{\epsilon+\delta} x_\delta$ arises from the assumption that x is a Cauchy approximation.

We will never use this notation again, so don't worry about remembering it. Generally we use the pattern-matching convention, where f is defined by equations such as (11.3.6) and (11.3.7) in which the right-hand side of (11.3.7) may involve the symbols $f(x_\epsilon)$ and an assumption that they form a dependent Cauchy approximation.

However, this induction principle is admittedly still quite a mouthful. To help make sense of it, we observe that it contains as special cases two separate induction principles for $\mathbb{R}_{\mathsf{c}}$ and for $\sim$. Firstly, suppose given only a type family $A : \mathbb{R}_{\mathsf{c}} \to \mathcal{U}$, and define $\frown$ to be constant at $\mathbf{1}$. Then much of the required data becomes trivial, and we are left with:

- for any $q : \mathbb{Q}$, an element $f_q : A(\mathsf{rat}(q))$,
- for any Cauchy approximation x, and any $a : \prod_{(\epsilon:\mathbb{Q}_+)} A(x_\epsilon)$, an element $f_{x,a} : A(\mathsf{lim}(x))$,
- for $u, v : \mathbb{R}_{\mathsf{c}}$ and $h : \forall(\epsilon : \mathbb{Q}_+).\, u \sim_\epsilon v$, and $a : A(u)$ and $b : A(v)$, we have $a =^A_{\mathsf{eq}_{\mathbb{R}_{\mathsf{c}}}(u,v)} b$.

Given these data, the induction principle yields a function $f : \prod_{(x:\mathbb{R}_{\mathsf{c}})} A(x)$ such that

$$f(\mathsf{rat}(q)) :\equiv f_q,$$
$$f(\mathsf{lim}(x)) :\equiv f_{x,f(x)}.$$

We call this principle **$\mathbb{R}_{\mathsf{c}}$-induction**; it says essentially that if we take $\sim_\epsilon$ as given, then $\mathbb{R}_{\mathsf{c}}$ is inductively generated by its constructors.

Note that, if A is a mere property, then the third hypothesis in $\mathbb{R}_{\mathsf{c}}$-induction is automatic (we will see in a moment that these are in fact equivalent statements). Thus, we may prove mere properties of real numbers by simply proving them for rationals and for limits of Cauchy approximations. Here is an example.

Lemma 11.3.8. *For any $u : \mathbb{R}_{\mathsf{c}}$ and $\epsilon : \mathbb{Q}_+$, we have $u \sim_\epsilon u$.*

Proof. Define $A(u) :\equiv \forall(\epsilon : \mathbb{Q}_+).\, (u \sim_\epsilon u)$. Since this is a mere proposition (by the last constructor of $\sim$), by $\mathbb{R}_{\mathsf{c}}$-induction, it suffices to prove it when u is $\mathsf{rat}(q)$ and when u is $\mathsf{lim}(x)$. In the first case, we obviously have $|q - q| < \epsilon$ for any ϵ, hence $\mathsf{rat}(q) \sim_\epsilon \mathsf{rat}(q)$ by the first constructor of $\sim$. And in the second case, we may assume inductively that $x_\delta \sim_\epsilon x_\delta$ for all $\delta, \epsilon : \mathbb{Q}_+$. Then in particular, we have $x_{\epsilon/3} \sim_{\epsilon/3} x_{\epsilon/3}$, whence $\mathsf{lim}(x) \sim_\epsilon \mathsf{lim}(x)$ by the fourth constructor of $\sim$. □

From Lemma 11.3.8, we infer that a direct application of $\mathbb{R}_{\mathsf{c}}$-induction only has a chance to succeed if the family $A : \mathbb{R}_{\mathsf{c}} \to \mathcal{U}$ is a mere property. To see this, fix $u : \mathbb{R}_{\mathsf{c}}$. Taking v to be u, the third hypothesis of $\mathbb{R}_{\mathsf{c}}$-induction tells us that, for any $a : A(u)$, we have $a =^A_{\mathsf{eq}_{\mathbb{R}_{\mathsf{c}}}(u,u)} a$. Given a point $b : A(u)$ in addition, we also get $a =^A_{\mathsf{eq}_{\mathbb{R}_{\mathsf{c}}}(u,u)} b$. From the definition of the dependent path type, we conclude that the combination of these two paths implies $a = b$, i.e. all points in $A(u)$ are equal.

Theorem 11.3.9. *$\mathbb{R}_{\mathsf{c}}$ is a set.*

Proof. We have just shown that the mere relation $P(u, v) :\equiv \forall(\epsilon : \mathbb{Q}_+).\, (u \sim_\epsilon v)$ is reflexive. Since it implies identity, by the path constructor of $\mathbb{R}_{\mathsf{c}}$, the result follows from Theorem 7.2.2. □

We can also show that although $\mathbb{R}_{\mathsf{c}}$ may not be a quotient of the set of Cauchy sequences of *rationals*, it is nevertheless a quotient of the set of Cauchy sequences of *reals*. (Of course, this is not

a valid *definition* of $\mathbb{R}_c$, but it is a useful property.) We define the type of Cauchy approximations to be

$$\mathcal{C} :\equiv \{\, x : \mathbb{Q}_+ \to \mathbb{R}_c \mid \forall(\epsilon, \delta : \mathbb{Q}_+).\, x_\delta \sim_{\delta+\epsilon} x_\epsilon \,\}.$$

The second constructor of $\mathbb{R}_c$ gives a function $\lim : \mathcal{C} \to \mathbb{R}_c$.

Lemma 11.3.10. *Every real merely is a limit point:* $\forall(u : \mathbb{R}_c).\, \exists(x : \mathcal{C}).\, u = \lim(x)$. *In other words,* $\lim : \mathcal{C} \to \mathbb{R}_c$ *is surjective.*

Proof. By $\mathbb{R}_c$-induction, we may divide into cases on u. Of course, if u is a limit $\lim(x)$, the statement is trivial. So suppose u is a rational point $\mathsf{rat}(q)$; we claim u is equal to $\lim(\lambda\epsilon.\, \mathsf{rat}(q))$. By the path constructor of $\mathbb{R}_c$, it suffices to show $\mathsf{rat}(q) \sim_\epsilon \lim(\lambda\epsilon.\, \mathsf{rat}(q))$ for all $\epsilon : \mathbb{Q}_+$. And by the second constructor of $\sim$, for this it suffices to find $\delta : \mathbb{Q}_+$ such that $\mathsf{rat}(q) \sim_{\epsilon-\delta} \mathsf{rat}(q)$. But by the first constructor of $\sim$, we may take any $\delta : \mathbb{Q}_+$ with $\delta < \epsilon$. □

Lemma 11.3.11. *If A is a set and $f : \mathcal{C} \to A$ respects coincidence of Cauchy approximations, in the sense that*

$$\forall(x, y : \mathcal{C}).\, \lim(x) = \lim(y) \Rightarrow f(x) = f(y),$$

then f factors uniquely through $\lim : \mathcal{C} \to \mathbb{R}_c$.

Proof. Since $\lim$ is surjective, by Theorem 10.1.5, $\mathbb{R}_c$ is the quotient of $\mathcal{C}$ by the kernel pair of $\lim$. But this is exactly the statement of the lemma. □

For the second special case of the induction principle, suppose instead that we take A to be constant at $\mathbf{1}$. In this case, $\frown$ is simply an ϵ-indexed family of relations on ϵ-close pairs of real numbers, so we may write $u \frown_\epsilon v$ instead of $(u, \star) \frown_\epsilon (v, \star)$. Then the required data reduces to the following, where q, r denote rational numbers, ϵ, δ, η positive rational numbers, and x, y Cauchy approximations:

- if $-\epsilon < q - r < \epsilon$, then $\mathsf{rat}(q) \frown_\epsilon \mathsf{rat}(r)$,
- if $\mathsf{rat}(q) \sim_{\epsilon-\delta} y_\delta$ and $\mathsf{rat}(q) \frown_{\epsilon-\delta} y_\delta$, then $\mathsf{rat}(q) \frown_\epsilon \lim(y)$,
- if $x_\delta \sim_{\epsilon-\delta} \mathsf{rat}(r)$ and $x_\delta \frown_{\epsilon-\delta} \mathsf{rat}(r)$, then $\lim(y) \frown_\epsilon \mathsf{rat}(q)$,
- if $x_\delta \sim_{\epsilon-\delta-\eta} y_\eta$ and $x_\delta \frown_{\epsilon-\delta-\eta} y_\eta$, then $\lim(x) \frown_\epsilon \lim(y)$.

The resulting conclusion is $\forall(u, v : \mathbb{R}_c).\, \forall(\epsilon : \mathbb{Q}_+).\, (u \sim_\epsilon v) \to (u \frown_\epsilon v)$. We call this principle **$\sim$-induction**; it says essentially that if we take $\mathbb{R}_c$ as given, then $\sim_\epsilon$ is inductively generated (as a family of types) by *its* constructors. For example, we can use this to show that $\sim$ is symmetric.

Lemma 11.3.12. *For any $u, v : \mathbb{R}_c$ and $\epsilon : \mathbb{Q}_+$, we have $(u \sim_\epsilon v) = (v \sim_\epsilon u)$.*

Proof. Since both are mere propositions, by symmetry it suffices to show one implication. Thus, let $(u \frown_\epsilon v) :\equiv (v \sim_\epsilon u)$. By $\sim$-induction, we may reduce to the case that $u \sim_\epsilon v$ is derived from one of the four interesting constructors of $\sim$. In the first case when u and v are both rational, the result is trivial (we can apply the first constructor again). In the other three cases, the inductive hypothesis (together with commutativity of addition in $\mathbb{Q}$) yields exactly the input to another of the constructors of $\sim$ (the second and third constructors switch, while the fourth stays put). □

The general induction principle, which we may call **$(\mathbb{R}_c, \sim)$-induction**, is therefore a sort of joint $\mathbb{R}_c$-induction and $\sim$-induction. Consider, for instance, its non-dependent version, which we call **$(\mathbb{R}_c, \sim)$-recursion**, which is the one that we will have the most use for. Ordinary $\mathbb{R}_c$-recursion tells us that to define a function $f : \mathbb{R}_c \to A$ it suffices to:

(i) for every $q : \mathbb{Q}$ construct $f(\mathsf{rat}(q)) : A$,

(ii) for every Cauchy approximation $x : \mathbb{Q}_+ \to \mathbb{R}_\mathsf{c}$, construct $f(x) : A$, assuming that $f(x_\epsilon)$ has already been defined for all $\epsilon : \mathbb{Q}_+$,

(iii) prove $f(u) = f(v)$ for all $u, v : \mathbb{R}_\mathsf{c}$ satisfying $\forall(\epsilon : \mathbb{Q}_+).\, u \sim_\epsilon v$.

However, it is generally quite difficult to show (iii) without knowing something about how f acts on ϵ-close Cauchy reals. The enhanced principle of $(\mathbb{R}_\mathsf{c}, \sim)$-recursion remedies this deficiency, allowing us to specify an *arbitrary* "way in which f acts on ϵ-close Cauchy reals", which we can then prove to be the case by a simultaneous induction with the definition of f. This is the family of relations $\frown$. Since A is independent of $\mathbb{R}_\mathsf{c}$, we may assume for simplicity that $\frown$ depends only on A and $\mathbb{Q}_+$, and thus there is no ambiguity in writing $a \frown_\epsilon b$ instead of $(u, a) \frown_\epsilon (v, b)$. In this case, defining a function $f : \mathbb{R}_\mathsf{c} \to A$ by $(\mathbb{R}_\mathsf{c}, \sim)$-recursion requires the following cases (which we now write using the pattern-matching convention).

- For every $q : \mathbb{Q}$, construct $f(\mathsf{rat}(q)) : A$.
- For every Cauchy approximation $x : \mathbb{Q}_+ \to \mathbb{R}_\mathsf{c}$, construct $f(x) : A$, assuming inductively that $f(x_\epsilon)$ has already been defined for all $\epsilon : \mathbb{Q}_+$ and form a "Cauchy approximation with respect to $\frown$", i.e. that $\forall(\epsilon, \delta : \mathbb{Q}_+).\, (f(x_\epsilon) \frown_{\epsilon+\delta} f(x_\delta))$.
- Prove that the relations $\frown$ are *separated*, i.e. that, for any $a, b : A$, $(\forall(\epsilon : \mathbb{Q}_+).\, a \frown_\epsilon b) \Rightarrow (a = b)$.
- Prove that if $-\epsilon < q - r < \epsilon$ for $q, r : \mathbb{Q}$, then $f(\mathsf{rat}(q)) \frown_\epsilon f(\mathsf{rat}(r))$.
- For any $q : \mathbb{Q}$ and any Cauchy approximation y, prove that $f(\mathsf{rat}(q)) \frown_\epsilon f(\mathsf{lim}(y))$, assuming inductively that $\mathsf{rat}(q) \sim_{\epsilon-\delta} y_\delta$ and $f(\mathsf{rat}(q)) \frown_{\epsilon-\delta} f(y_\delta)$ for some $\delta : \mathbb{Q}_+$, and that $\eta \mapsto f(x_\eta)$ is a Cauchy approximation with respect to $\frown$.
- For any Cauchy approximation x and any $r : \mathbb{Q}$, prove that $f(\mathsf{lim}(x)) \frown_\epsilon f(\mathsf{rat}(r))$, assuming inductively that $x_\delta \sim_{\epsilon-\delta} \mathsf{rat}(r)$ and $f(x_\delta) \frown_{\epsilon-\delta} f(\mathsf{rat}(r))$ for some $\delta : \mathbb{Q}_+$, and that $\eta \mapsto f(x_\eta)$ is a Cauchy approximation with respect to $\frown$.
- For any Cauchy approximations x, y, prove that $f(\mathsf{lim}(x)) \frown_\epsilon f(\mathsf{lim}(y))$, assuming inductively that $x_\delta \sim_{\epsilon-\delta-\eta} y_\eta$ and $f(x_\delta) \frown_{\epsilon-\delta-\eta} f(y_\eta)$ for some $\delta, \eta : \mathbb{Q}_+$, and that $\theta \mapsto f(x_\theta)$ and $\theta \mapsto f(y_\theta)$ are Cauchy approximations with respect to $\frown$.

Note that in the last four proofs, we are free to use the specific definitions of $f(\mathsf{rat}(q))$ and $f(\mathsf{lim}(x))$ given in the first two data. However, the proof of separatedness must apply to *any* two elements of A, without any relation to f: it is a sort of "admissibility" condition on the family of relations $\frown$. Thus, we often verify it first, immediately after defining $\frown$, before going on to define $f(\mathsf{rat}(q))$ and $f(\mathsf{lim}(x))$.

Under the above hypotheses, $(\mathbb{R}_\mathsf{c}, \sim)$-recursion yields a function $f : \mathbb{R}_\mathsf{c} \to A$ such that $f(\mathsf{rat}(q))$ and $f(\mathsf{lim}(x))$ are judgmentally equal to the definitions given for them in the first two clauses. Moreover, we may also conclude

$$\forall(u, v : \mathbb{R}_\mathsf{c}).\, \forall(\epsilon : \mathbb{Q}_+).\, (u \sim_\epsilon v) \to (f(u) \frown_\epsilon f(v)). \tag{11.3.13}$$

As a paradigmatic example, $(\mathbb{R}_\mathsf{c}, \sim)$-recursion allows us to extend functions defined on $\mathbb{Q}$ to all of $\mathbb{R}_\mathsf{c}$, as long as they are sufficiently continuous.

Definition 11.3.14. A function $f : \mathbb{Q} \to \mathbb{R}_\mathsf{c}$ is **Lipschitz** if there exists $L : \mathbb{Q}_+$ (the **Lipschitz constant**) such that

$$|q - r| < \epsilon \Rightarrow (f(q) \sim_{L\epsilon} f(r))$$

for all $\epsilon : \mathbb{Q}_+$ and $q, r : \mathbb{Q}$. Similarly, $g : \mathbb{R}_c \to \mathbb{R}_c$ is **Lipschitz** if there exists $L : \mathbb{Q}_+$ such that

$$(u \sim_\epsilon v) \Rightarrow (g(u) \sim_{L\epsilon} g(v))$$

for all $\epsilon : \mathbb{Q}_+$ and $u, v : \mathbb{R}_c$..

In particular, note that by the first constructor of $\sim$, if $f : \mathbb{Q} \to \mathbb{Q}$ is Lipschitz in the obvious sense, then so is the composite $\mathbb{Q} \xrightarrow{f} \mathbb{Q} \to \mathbb{R}_c$.

Lemma 11.3.15. *Suppose $f : \mathbb{Q} \to \mathbb{R}_c$ is Lipschitz with constant $L : \mathbb{Q}_+$. Then there exists a Lipschitz map $\bar{f} : \mathbb{R}_c \to \mathbb{R}_c$, also with constant L, such that $\bar{f}(\mathsf{rat}(q)) \equiv f(q)$ for all $q : \mathbb{Q}$.*

Proof. We define $\bar{f}$ by $(\mathbb{R}_c, \sim)$-recursion, with codomain $A :\equiv \mathbb{R}_c$. We define the relation $\frown : \mathbb{R}_c \to \mathbb{R}_c \to \mathbb{Q}_+ \to \mathsf{Prop}$ to be

$$(u \frown_\epsilon v) :\equiv (u \sim_{L\epsilon} v).$$

For $q : \mathbb{Q}$, we define

$$\bar{f}(\mathsf{rat}(q)) :\equiv \mathsf{rat}(f(q)).$$

For a Cauchy approximation $x : \mathbb{Q}_+ \to \mathbb{R}_c$, we define

$$\bar{f}(\mathsf{lim}(x)) :\equiv \mathsf{lim}(\lambda\epsilon.\, \bar{f}(x_{\epsilon/L})).$$

For this to make sense, we must verify that $y :\equiv \lambda\epsilon.\, \bar{f}(x_{\epsilon/L})$ is a Cauchy approximation. However, the inductive hypothesis for this step is that for any $\delta, \epsilon : \mathbb{Q}_+$ we have $\bar{f}(x_\delta) \frown_{\delta+\epsilon} \bar{f}(x_\epsilon)$, i.e. $\bar{f}(x_\delta) \sim_{L\delta+L\epsilon} \bar{f}(x_\epsilon)$. Thus we have

$$y_\delta \equiv f(x_{\delta/L}) \sim_{\delta+\epsilon} f(x_{\epsilon/L}) \equiv y_\epsilon.$$

For proving separatedness, we simply observe that $\forall(\epsilon : \mathbb{Q}_+).\, a \frown_\epsilon b$ means $\forall(\epsilon : \mathbb{Q}_+).\, a \sim_{L\epsilon} b$, which implies $\forall(\epsilon : \mathbb{Q}_+).\, a \sim_\epsilon b$ and thus $a = b$.

To complete the $(\mathbb{R}_c, \sim)$-recursion, it remains to verify the four conditions on $\frown$. This basically amounts to proving that $\bar{f}$ is Lipschitz for all the four constructors of $\sim$.

(i) When u is $\mathsf{rat}(q)$ and v is $\mathsf{rat}(r)$ with $-\epsilon < |q - r| < \epsilon$, the assumption that f is Lipschitz yields $f(q) \sim_{L\epsilon} f(r)$, hence $\bar{f}(\mathsf{rat}(q)) \frown_\epsilon \bar{f}(\mathsf{rat}(r))$ by definition.

(ii) When u is $\mathsf{lim}(x)$ and v is $\mathsf{rat}(q)$ with $x_\eta \sim_{\epsilon-\eta} \mathsf{rat}(q)$, then the inductive hypothesis is $\bar{f}(x_\eta) \sim_{L\epsilon-L\eta} \mathsf{rat}(f(q))$, which proves $\bar{f}(\mathsf{lim}(x)) \sim_{L\epsilon} \bar{f}(\mathsf{rat}(q))$ by the third constructor of $\sim$.

(iii) The symmetric case when u is rational and v is a limit is essentially identical.

(iv) When u is $\mathsf{lim}(x)$ and v is $\mathsf{lim}(y)$, with $\delta, \eta : \mathbb{Q}_+$ such that $x_\delta \sim_{\epsilon-\delta-\eta} y_\eta$, the inductive hypothesis is $\bar{f}(x_\delta) \sim_{L\epsilon-L\delta-L\eta} \bar{f}(y_\eta)$, which proves $\bar{f}(\mathsf{lim}(x)) \sim_{L\epsilon} \bar{f}(\mathsf{lim}(y))$ by the fourth constructor of $\sim$.

This completes the $(\mathbb{R}_c, \sim)$-recursion, and hence the construction of $\bar{f}$. The desired equality $\bar{f}(\mathsf{rat}(q)) \equiv f(q)$ is exactly the first computation rule for $(\mathbb{R}_c, \sim)$-recursion, and the additional condition (11.3.13) says exactly that $\bar{f}$ is Lipschitz with constant L. □

At this point we have gone about as far as we can without a better characterization of $\sim$. We have specified, in the constructors of $\sim$, the conditions under which we want Cauchy reals of the two different forms to be ϵ-close. However, how do we know that in the resulting inductive-inductive type family, these are the *only* witnesses to this fact? We have seen that inductive type families (such as identity types, see §5.8) and higher inductive types have a tendency to contain "more than was put into them", so this is not an idle question.

In order to characterize $\sim$ more precisely, we will define a family of relations $\approx_\epsilon$ on $\mathbb{R}_\mathsf{c}$ *recursively*, so that they will compute on constructors, and prove that this family is equivalent to $\sim_\epsilon$.

Theorem 11.3.16. *There is a family of mere relations* $\approx : \mathbb{R}_\mathsf{c} \to \mathbb{R}_\mathsf{c} \to \mathbb{Q}_+ \to \mathsf{Prop}$ *such that*

$$(\mathsf{rat}(q) \approx_\epsilon \mathsf{rat}(r)) :\equiv (-\epsilon < q - r < \epsilon) \tag{11.3.17}$$
$$(\mathsf{rat}(q) \approx_\epsilon \mathsf{lim}(y)) :\equiv \exists(\delta : \mathbb{Q}_+).\, \mathsf{rat}(q) \approx_{\epsilon-\delta} y_\delta \tag{11.3.18}$$
$$(\mathsf{lim}(x) \approx_\epsilon \mathsf{rat}(r)) :\equiv \exists(\delta : \mathbb{Q}_+).\, x_\delta \approx_{\epsilon-\delta} \mathsf{rat}(r) \tag{11.3.19}$$
$$(\mathsf{lim}(x) \approx_\epsilon \mathsf{lim}(y)) :\equiv \exists(\delta, \eta : \mathbb{Q}_+).\, x_\delta \approx_{\epsilon-\delta-\eta} y_\eta. \tag{11.3.20}$$

Moreover, we have

$$(u \approx_\epsilon v) \Leftrightarrow \exists(\theta : \mathbb{Q}_+).\, (u \approx_{\epsilon-\theta} v) \tag{11.3.21}$$
$$(u \approx_\epsilon v) \to (v \sim_\delta w) \to (u \approx_{\epsilon+\delta} w) \tag{11.3.22}$$
$$(u \sim_\epsilon v) \to (v \approx_\delta w) \to (u \approx_{\epsilon+\delta} w). \tag{11.3.23}$$

The additional conditions (11.3.21)–(11.3.23) turn out to be required in order to make the inductive definition go through. Condition (11.3.21) is called being **rounded**. Reading it from right to left gives **monotonicity** of $\approx$,

$$(\delta < \epsilon) \wedge (u \approx_\delta v) \Rightarrow (u \approx_\epsilon v)$$

while reading it left to right to **openness** of $\approx$,

$$(u \approx_\epsilon v) \Rightarrow \exists(\delta : \mathbb{Q}_+).\, (\delta < \epsilon) \wedge (u \approx_\delta v).$$

Conditions (11.3.22) and (11.3.23) are forms of the triangle inequality, which say that $\approx$ is a "module" over $\sim$ on both sides.

Proof. We will define $\approx : \mathbb{R}_\mathsf{c} \to \mathbb{R}_\mathsf{c} \to \mathbb{Q}_+ \to \mathsf{Prop}$ by double $(\mathbb{R}_\mathsf{c}, \sim)$-recursion. First we will apply $(\mathbb{R}_\mathsf{c}, \sim)$-recursion with codomain the subset of $\mathbb{R}_\mathsf{c} \to \mathbb{Q}_+ \to \mathsf{Prop}$ consisting of those families of predicates which are rounded and satisfy the one appropriate form of the triangle inequality. Thinking of these predicates as half of a binary relation, we will write them as $(u, \epsilon) \mapsto (\diamondsuit \approx_\epsilon u)$, with the symbol $\diamondsuit$ referring to the whole relation. Now we can write A precisely as

$$\begin{aligned} A :\equiv \Bigg\{ \diamondsuit : \mathbb{R}_\mathsf{c} \to \mathbb{Q}_+ \to \mathsf{Prop} \;\Bigg|\; & \Big(\forall(u : \mathbb{R}_\mathsf{c}).\, \forall(\epsilon : \mathbb{Q}_+).\, \big((\diamondsuit \approx_\epsilon u) \Leftrightarrow \exists(\theta : \mathbb{Q}_+).\, (\diamondsuit \approx_{\epsilon-\theta} u)\big)\Big) \\ & \wedge \Big(\forall(u, v : \mathbb{R}_\mathsf{c}).\, \forall(\eta, \epsilon : \mathbb{Q}_+).\, (u \sim_\epsilon v) \to \\ & \qquad \big((\diamondsuit \approx_\eta u) \to (\diamondsuit \approx_{\eta+\epsilon} v)\big) \wedge \big((\diamondsuit \approx_\eta v) \to (\diamondsuit \approx_{\eta+\epsilon} u)\big)\Big) \Bigg\} \end{aligned}$$

As usual with subsets, we will use the same notation for an inhabitant of A and its first component $\diamondsuit$. As the family of relations required for $(\mathbb{R}_{\mathsf{c}}, \sim)$-recursion, we consider the following, which will ensure the other form of the triangle inequality:

$$(\diamondsuit \frown_\epsilon \heartsuit) :\equiv \forall(u : \mathbb{R}_{\mathsf{c}}).\, \forall(\eta : \mathbb{Q}_+).\, ((\diamondsuit \approx_\eta u) \to (\heartsuit \approx_{\epsilon+\eta} u)) \wedge ((\heartsuit \approx_\eta u) \to (\diamondsuit \approx_{\epsilon+\eta} u)).$$

We observe that these relations are separated. For assuming $\forall(\epsilon : \mathbb{Q}_+).\, (\diamondsuit \frown_\epsilon \heartsuit)$, to show $\diamondsuit = \heartsuit$ it suffices to show $(\diamondsuit \approx_\epsilon u) \Leftrightarrow (\heartsuit \approx_\epsilon u)$ for all $u : \mathbb{R}_{\mathsf{c}}$. But $\diamondsuit \approx_\epsilon u$ implies $\diamondsuit \approx_{\epsilon-\theta} u$ for some θ, by roundedness, which together with $\diamondsuit \frown_\epsilon \heartsuit$ implies $\heartsuit \approx_\epsilon u$; and the converse is identical.

Now the first two data the recursion principle requires are the following.

- For any $q : \mathbb{Q}$, we must give an element of A, which we denote $(\mathsf{rat}(q) \approx_{(-)} -)$.
- For any Cauchy approximation x, if we assume defined a function $\mathbb{Q}_+ \to A$, which we will denote by $\epsilon \mapsto (x_\epsilon \approx_{(-)} -)$, with the property that

$$\forall(u : \mathbb{R}_{\mathsf{c}}).\, \forall(\delta, \epsilon, \eta : \mathbb{Q}_+).\, (x_\delta \approx_\eta u) \to (x_\epsilon \approx_{\eta+\delta+\epsilon} u), \tag{11.3.24}$$

  we must give an element of A, which we write as $(\mathsf{lim}(x) \approx_{(-)} -)$.

In both cases, we give the required definition by using a nested $(\mathbb{R}_{\mathsf{c}}, \sim)$-recursion, with codomain the subset of $\mathbb{Q}_+ \to \mathsf{Prop}$ consisting of rounded families of mere propositions. Thinking of these propositions as zero halves of a binary relation, we will write them as $\epsilon \mapsto (\bullet \approx_\epsilon \triangle)$, with the symbol $\triangle$ referring to the whole family. Now we can write the codomain of these inner recursions precisely as

$$C :\equiv \left\{ \triangle : \mathbb{Q}_+ \to \mathsf{Prop} \;\middle|\; \forall(\epsilon : \mathbb{Q}_+).\, \Big((\bullet \approx_\epsilon \triangle) \Leftrightarrow \exists(\theta : \mathbb{Q}_+).\, (\bullet \approx_{\epsilon-\theta} \triangle)\Big) \right\}$$

We take the required family of relations to be the remnant of the triangle inequality:

$$(\triangle \smile_\epsilon \square) :\equiv \forall(\eta : \mathbb{Q}_+).\, ((\bullet \approx_\eta \triangle) \to (\bullet \approx_{\epsilon+\eta} \square)) \wedge ((\bullet \approx_\eta \square) \to (\bullet \approx_{\epsilon+\eta} \triangle)).$$

These relations are separated by the same argument as for $\frown$, using roundedness of all elements of C.

Note that if such an inner recursion succeeds, it will yield a family of predicates $\diamondsuit : \mathbb{R}_{\mathsf{c}} \to \mathbb{Q}_+ \to \mathsf{Prop}$ which are rounded (since their image in $\mathbb{Q}_+ \to \mathsf{Prop}$ lies in C) and satisfy

$$\forall(u, v : \mathbb{R}_{\mathsf{c}}).\, \forall(\epsilon : \mathbb{Q}_+).\, (u \sim_\epsilon v) \to \big((\diamondsuit \approx_{(-)} u) \smile_\epsilon (\diamondsuit \approx_{(-)} u)\big).$$

Expanding out the definition of $\smile$, this yields precisely the third condition for $\diamondsuit$ to belong to A; thus it is exactly what we need.

It is at this point that we can give the definitions (11.3.17)–(11.3.20), as the first two clauses of each of the two inner recursions, corresponding to rational points and limits. In each case, we must verify that the relation is rounded and hence lies in C. In the rational-rational case (11.3.17) this is clear, while in the other cases it follows from an inductive hypothesis. (In (11.3.18) the relevant inductive hypothesis is that $(\mathsf{rat}(q) \approx_{(-)} y_\delta) : C$, while in (11.3.19) and (11.3.20) it is that $(x_\delta \approx_{(-)} -) : A$.)

The remaining data of the sub-recursions consist of showing that (11.3.17)–(11.3.20) satisfy the triangle inequality on the right with respect to the constructors of $\sim$. There are eight cases — four in each sub-recursion — corresponding to the eight possible ways that u, v, and w in (11.3.22) can be chosen to be rational points or limits. First we consider the cases when u is $\mathsf{rat}(q)$.

(i) Assuming $\mathsf{rat}(q) \approx_\phi \mathsf{rat}(r)$ and $-\epsilon < |r-s| < \epsilon$, we must show $\mathsf{rat}(q) \approx_{\phi+\epsilon} \mathsf{rat}(s)$. But by definition of $\approx$, this reduces to the triangle inequality for rational numbers.

(ii) We assume $\phi, \epsilon, \delta : \mathbb{Q}_+$ such that $\mathsf{rat}(q) \approx_\phi \mathsf{rat}(r)$ and $\mathsf{rat}(r) \sim_{\epsilon-\delta} y_\delta$, and inductively that

$$\forall(\psi : \mathbb{Q}_+).\, (\mathsf{rat}(q) \approx_\psi \mathsf{rat}(r)) \to (\mathsf{rat}(q) \approx_{\psi+\epsilon-\delta} y_\delta). \tag{11.3.25}$$

We assume also that $\psi, \delta \mapsto (\mathsf{rat}(q) \approx_\psi y_\delta)$ is a Cauchy approximation with respect to $\smile$, i.e.

$$\forall(\psi, \xi, \zeta : \mathbb{Q}_+).\, (\mathsf{rat}(q) \approx_\psi y_\xi) \to (\mathsf{rat}(q) \approx_{\psi+\xi+\zeta} y_\zeta), \tag{11.3.26}$$

although we do not need this assumption in this case. Indeed, (11.3.25) with $\psi :\equiv \phi$ yields immediately $\mathsf{rat}(q) \approx_{\phi+\epsilon-\delta} y_\delta$, and hence $\mathsf{rat}(q) \approx_{\phi+\epsilon} \lim(y)$ by definition of $\approx$.

(iii) We assume $\phi, \epsilon, \delta : \mathbb{Q}_+$ such that $\mathsf{rat}(q) \approx_\phi \lim(y)$ and $y_\delta \sim_{\epsilon-\delta} \mathsf{rat}(r)$, and inductively that

$$\forall(\psi : \mathbb{Q}_+).\, (\mathsf{rat}(q) \approx_\psi y_\delta) \to (\mathsf{rat}(q) \approx_{\psi+\epsilon-\delta} \mathsf{rat}(r)). \tag{11.3.27}$$
$$\forall(\psi, \xi, \zeta : \mathbb{Q}_+).\, (\mathsf{rat}(q) \approx_\psi y_\xi) \to (\mathsf{rat}(q) \approx_{\psi+\xi+\zeta} y_\zeta). \tag{11.3.28}$$

By definition, $\mathsf{rat}(q) \approx_\phi \lim(y)$ means that we have $\theta : \mathbb{Q}_+$ with $\mathsf{rat}(q) \approx_{\phi-\theta} y_\theta$. By assumption (11.3.28), therefore, we have also $\mathsf{rat}(q) \approx_{\phi+\delta} y_\delta$, and then by (11.3.27) it follows that $\mathsf{rat}(q) \approx_{\phi+\epsilon} \mathsf{rat}(r)$, as desired.

(iv) We assume $\phi, \epsilon, \delta, \eta : \mathbb{Q}_+$ such that $\mathsf{rat}(q) \approx_\phi \lim(y)$ and $y_\delta \sim_{\epsilon-\delta-\eta} z_\eta$, and inductively that

$$\forall(\psi : \mathbb{Q}_+).\, (\mathsf{rat}(q) \approx_\psi y_\delta) \to (\mathsf{rat}(q) \approx_{\psi+\epsilon-\delta-\eta} z_\eta), \tag{11.3.29}$$
$$\forall(\psi, \xi, \zeta : \mathbb{Q}_+).\, (\mathsf{rat}(q) \approx_\psi y_\xi) \to (\mathsf{rat}(q) \approx_{\psi+\xi+\zeta} y_\zeta), \tag{11.3.30}$$
$$\forall(\psi, \xi, \zeta : \mathbb{Q}_+).\, (\mathsf{rat}(q) \approx_\psi z_\xi) \to (\mathsf{rat}(q) \approx_{\psi+\xi+\zeta} z_\zeta). \tag{11.3.31}$$

Again, $\mathsf{rat}(q) \approx_\phi \lim(y)$ means we have $\xi : \mathbb{Q}_+$ with $\mathsf{rat}(q) \approx_{\phi-\xi} y_\xi$, while (11.3.30) then implies $\mathsf{rat}(q) \approx_{\phi+\delta} y_\delta$ and (11.3.29) implies $\mathsf{rat}(q) \approx_{\phi+\epsilon-\eta} z_\eta$. But by definition of $\approx$, this implies $\mathsf{rat}(q) \approx_{\phi+\epsilon} \lim(z)$ as desired.

Now we move on to the cases when u is $\lim(x)$, with x a Cauchy approximation. In this case, the ambient inductive hypothesis of the definition of $(\lim(x) \approx_{(-)} -) : A$ is that we have $(x_\delta \approx_{(-)} -) : A$, so that in addition to being rounded they satisfy the triangle inequality on the right.

(v) Assuming $\lim(x) \approx_\phi \mathsf{rat}(r)$ and $-\epsilon < |r-s| < \epsilon$, we must show $\lim(x) \approx_{\phi+\epsilon} \mathsf{rat}(s)$. By definition of $\approx$, the former means $x_\delta \approx_{\phi-\delta} \mathsf{rat}(r)$, so that above triangle inequality implies $x_\delta \approx_{\epsilon+\phi-\delta} \mathsf{rat}(s)$, hence $\lim(x) \approx_{\phi+\epsilon} \mathsf{rat}(s)$ as desired.

(vi) We assume $\phi, \epsilon, \delta : \mathbb{Q}_+$ such that $\lim(x) \approx_\phi \mathsf{rat}(r)$ and $\mathsf{rat}(r) \sim_{\epsilon-\delta} y_\delta$, and two unneeded inductive hypotheses. By definition, we have $\eta : \mathbb{Q}_+$ such that $x_\eta \approx_{\phi-\eta} \mathsf{rat}(r)$, so the inductive triangle inequality gives $x_\eta \approx_{\phi+\epsilon-\eta-\delta} y_\delta$. The definition of $\approx$ then immediately yields $\lim(x) \approx_{\phi+\epsilon} \lim(y)$.

(vii) We assume $\phi, \epsilon, \delta : \mathbb{Q}_+$ such that $\lim(x) \approx_\phi \lim(y)$ and $y_\delta \sim_{\epsilon-\delta} \mathsf{rat}(r)$, and two unneeded inductive hypotheses. By definition we have $\xi, \theta : \mathbb{Q}_+$ such that $x_\xi \approx_{\phi-\xi-\theta} y_\theta$. Since y is a Cauchy approximation, we have $y_\theta \sim_{\theta+\delta} y_\delta$, so the inductive triangle inequality gives $x_\xi \approx_{\phi+\delta-\xi} y_\delta$ and then $x_\xi \sim_{\phi+\epsilon-\xi} \mathsf{rat}(r)$. The definition of $\approx$ then gives $\lim(x) \approx_{\phi+\epsilon} \mathsf{rat}(r)$, as desired.

(viii) Finally, we assume $\phi, \epsilon, \delta, \eta : \mathbb{Q}_+$ such that $\mathsf{lim}(x) \approx_\phi \mathsf{lim}(y)$ and $y_\delta \sim_{\epsilon-\delta-\eta} z_\eta$. Then as before we have $\xi, \theta : \mathbb{Q}_+$ with $x_\xi \approx_{\phi-\xi-\theta} y_\theta$, and two applications of the triangle inequality suffices as before.

This completes the two inner recursions, and thus the definitions of the families of relations $(\mathsf{rat}(q) \approx_{(-)} -)$ and $(\mathsf{lim}(x) \approx_{(-)} -)$. Since all are elements of A, they are rounded and satisfy the triangle inequality on the right with respect to $\sim$. What remains is to verify the conditions relating to $\frown$, which is to say that these relations satisfy the triangle inequality on the *left* with respect to the constructors of $\sim$. The four cases correspond to the four choices of rational or limit points for u and v in (11.3.23), and since they are all mere propositions, we may apply $\mathbb{R}_\mathsf{c}$-induction and assume that w is also either rational or a limit. This yields another eight cases, whose proofs are essentially identical to those just given; so we will not subject the reader to them. □

We can now prove:

Theorem 11.3.32. *For any $u, v : \mathbb{R}_\mathsf{c}$ and $\epsilon : \mathbb{Q}_+$ we have $(u \sim_\epsilon v) = (u \approx_\epsilon v)$.*

Proof. Since both are mere propositions, it suffices to prove bidirectional implication. For the left-to-right direction, we use $\sim$-induction applied to $C(u, v, \epsilon) :\equiv (u \approx_\epsilon v)$. Thus, it suffices to consider the four constructors of $\sim$. In each case, u and v are specialized to either rational points or limits, so that the definition of $\approx$ evaluates, and the inductive hypothesis always applies.

For the right-to-left direction, we use $\mathbb{R}_\mathsf{c}$-induction to assume that u and v are rational points or limits, allowing $\approx$ to evaluate. But now the definitions of $\approx$, and the inductive hypotheses, supply exactly the data required for the relevant constructors of $\sim$. □

Stretching a point, one might call $\approx$ a fibration of "codes" for $\sim$, with the two directions of the above proof being encode and decode respectively. By the definition of $\approx$, from Theorem 11.3.32 we get equivalences

$$\begin{aligned}
(\mathsf{rat}(q) \sim_\epsilon \mathsf{rat}(r)) &= (-\epsilon < q - r < \epsilon) \\
(\mathsf{rat}(q) \sim_\epsilon \mathsf{lim}(y)) &= \exists(\delta : \mathbb{Q}_+).\, \mathsf{rat}(q) \sim_{\epsilon-\delta} y_\delta \\
(\mathsf{lim}(x) \sim_\epsilon \mathsf{rat}(r)) &= \exists(\delta : \mathbb{Q}_+).\, x_\delta \sim_{\epsilon-\delta} \mathsf{rat}(r) \\
(\mathsf{lim}(x) \sim_\epsilon \mathsf{lim}(y)) &= \exists(\delta, \eta : \mathbb{Q}_+).\, x_\delta \sim_{\epsilon-\delta-\eta} y_\eta.
\end{aligned}$$

Our proof also provides the following additional information.

Corollary 11.3.33. *$\sim$ is rounded and satisfies the triangle inequality:*

$$(u \sim_\epsilon v) \;\simeq\; \exists(\theta : \mathbb{Q}_+).\, u \sim_{\epsilon-\theta} v \tag{11.3.34}$$

$$(u \sim_\epsilon v) \to (v \sim_\delta w) \to (u \sim_{\epsilon+\delta} w). \tag{11.3.35}$$

With the triangle inequality in hand, we can show that "limits" of Cauchy approximations actually behave like limits.

Lemma 11.3.36. *For any $u : \mathbb{R}_\mathsf{c}$, Cauchy approximation y, and $\epsilon, \delta : \mathbb{Q}_+$, if $u \sim_\epsilon y_\delta$ then $u \sim_{\epsilon+\delta} \mathsf{lim}(y)$.*

Proof. We use $\mathbb{R}_\mathsf{c}$-induction on u. If u is $\mathsf{rat}(q)$, then this is exactly the second constructor of $\sim$. Now suppose u is $\mathsf{lim}(x)$, and that each x_η has the property that for any y, ϵ, δ, if $x_\eta \sim_\epsilon y_\delta$ then $x_\eta \sim_{\epsilon+\delta} \mathsf{lim}(y)$. In particular, taking $y :\equiv x$ and $\delta :\equiv \eta$ in this assumption, we conclude that $x_\eta \sim_{\eta+\theta} \mathsf{lim}(x)$ for any $\eta, \theta : \mathbb{Q}_+$.

Now let y, ϵ, δ be arbitrary and assume $\mathsf{lim}(x) \sim_\epsilon y_\delta$. By roundedness, there is a θ such that $\mathsf{lim}(x) \sim_{\epsilon-\theta} y_\delta$. Then by the above observation, for any η we have $x_\eta \sim_{\eta+\theta/2} \mathsf{lim}(x)$, and hence $x_\eta \sim_{\epsilon+\eta-\theta/2} y_\delta$ by the triangle inequality. Hence, the fourth constructor of $\sim$ yields $\mathsf{lim}(x) \sim_{\epsilon+2\eta+\delta-\theta/2} \mathsf{lim}(y)$. Thus, if we choose $\eta :\equiv \theta/4$, the result follows. □

Lemma 11.3.37. *For any Cauchy approximation y and any $\delta, \eta : \mathbb{Q}_+$ we have $y_\delta \sim_{\delta+\eta} \mathsf{lim}(y)$.*

Proof. Take $u :\equiv y_\delta$ and $\epsilon :\equiv \eta$ in the previous lemma. □

Remark 11.3.38. We might have expected to have $y_\delta \sim_\delta \mathsf{lim}(y)$, but this fails in examples. For instance, consider x defined by $x_\epsilon :\equiv \epsilon$. Its limit is clearly 0, but we do not have $|\epsilon - 0| < \epsilon$, only $\leq$.

As an application, Lemma 11.3.37 enables us to show that the extensions of Lipschitz functions from Lemma 11.3.15 are unique.

Lemma 11.3.39. *Let $f, g : \mathbb{R}_\mathsf{c} \to \mathbb{R}_\mathsf{c}$ be continuous, in the sense that*

$$\forall(u : \mathbb{R}_\mathsf{c}).\, \forall(\epsilon : \mathbb{Q}_+).\, \exists(\delta : \mathbb{Q}_+).\, \forall(v : \mathbb{R}_\mathsf{c}).\, (u \sim_\delta v) \to (f(u) \sim_\epsilon f(v))$$

and analogously for g. If $f(\mathsf{rat}(q)) = g(\mathsf{rat}(q))$ for all $q : \mathbb{Q}$, then $f = g$.

Proof. We prove $f(u) = g(u)$ for all u by $\mathbb{R}_\mathsf{c}$-induction. The rational case is just the hypothesis. Thus, suppose $f(x_\delta) = g(x_\delta)$ for all δ. We will show that $f(\mathsf{lim}(x)) \sim_\epsilon g(\mathsf{lim}(x))$ for all ϵ, so that the path constructor of $\mathbb{R}_\mathsf{c}$ applies.

Since f and g are continuous, there exist θ, η such that for all v, we have

$$(\mathsf{lim}(x) \sim_\theta v) \to (f(\mathsf{lim}(x)) \sim_{\epsilon/2} f(v))$$
$$(\mathsf{lim}(x) \sim_\eta v) \to (g(\mathsf{lim}(x)) \sim_{\epsilon/2} g(v)).$$

Choosing $\delta < \min(\theta, \eta)$, by Lemma 11.3.37 we have both $\mathsf{lim}(x) \sim_\theta y_\delta$ and $\mathsf{lim}(x) \sim_\eta y_\delta$. Hence

$$f(\mathsf{lim}(x)) \sim_{\epsilon/2} f(y_\delta) = g(y_\delta) \sim_{\epsilon/2} g(\mathsf{lim}(x))$$

and thus $f(\mathsf{lim}(x)) \sim_\epsilon g(\mathsf{lim}(x))$ by the triangle inequality. □

11.3.3 The algebraic structure of Cauchy reals

We first define the additive structure $(\mathbb{R}_\mathsf{c}, 0, +, -)$. Clearly, the additive unit element 0 is just $\mathsf{rat}(0)$, while the additive inverse $- : \mathbb{R}_\mathsf{c} \to \mathbb{R}_\mathsf{c}$ is obtained as the extension of the additive inverse $- : \mathbb{Q} \to \mathbb{Q}$, using Lemma 11.3.15 with Lipschitz constant 1. We have to work a bit harder for addition.

Lemma 11.3.40. *Suppose $f : \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ satisfies, for all $q, r, s : \mathbb{Q}$,*

$$|f(q,s) - f(r,s)| \leq |q - r| \qquad \text{and} \qquad |f(q,r) - f(q,s)| \leq |r - s|.$$

Then there is a function $\bar{f} : \mathbb{R}_\mathsf{c} \times \mathbb{R}_\mathsf{c} \to \mathbb{R}_\mathsf{c}$ such that $\bar{f}(\mathsf{rat}(q), \mathsf{rat}(r)) = f(q,r)$ for all $q, r : \mathbb{Q}$. Furthermore, for all $u, v, w : \mathbb{R}_\mathsf{c}$ and $q : \mathbb{Q}_+$,

$$u \sim_\epsilon v \Rightarrow \bar{f}(u,w) \sim_\epsilon \bar{f}(v,w) \quad \text{and} \quad v \sim_\epsilon w \Rightarrow \bar{f}(u,v) \sim_\epsilon \bar{f}(u,w).$$

Proof. We use $(\mathbb{R}_\mathsf{c}, \sim)$-recursion to construct the curried form of $\bar{f}$ as a map $\mathbb{R}_\mathsf{c} \to A$ where A is the space of non-expanding real-valued functions:

$$A :\equiv \{\, h : \mathbb{R}_\mathsf{c} \to \mathbb{R}_\mathsf{c} \mid \forall(\epsilon : \mathbb{Q}_+).\, \forall(u, v : \mathbb{R}_\mathsf{c}).\, u \sim_\epsilon v \Rightarrow h(u) \sim_\epsilon h(v) \,\}.$$

We shall also need a suitable $\frown_\epsilon$ on A, which we define as

$$(h \frown_\epsilon k) :\equiv \forall(u : \mathbb{R}_\mathsf{c}).\, h(u) \sim_\epsilon k(u).$$

Clearly, if $\forall(\epsilon : \mathbb{Q}_+).\, h \frown_\epsilon k$ then $h(u) = k(u)$ for all $u : \mathbb{R}_\mathsf{c}$, so $\frown$ is separated.

For the base case we define $\bar{f}(\mathsf{rat}(q)) : A$, where $q : \mathbb{Q}$, as the extension of the Lipschitz map $\lambda r.\, f(q, r)$ from $\mathbb{Q} \to \mathbb{Q}$ to $\mathbb{R}_\mathsf{c} \to \mathbb{R}_\mathsf{c}$, as constructed in Lemma 11.3.15 with Lipschitz constant 1. Next, for a Cauchy approximation x, we define $\bar{f}(\mathsf{lim}(x)) : \mathbb{R}_\mathsf{c} \to \mathbb{R}_\mathsf{c}$ as

$$\bar{f}(\mathsf{lim}(x))(v) :\equiv \mathsf{lim}(\lambda\epsilon.\, \bar{f}(x_\epsilon)(v)).$$

For this to be a valid definition, $\lambda\epsilon.\, \bar{f}(x_\epsilon)(v)$ should be a Cauchy approximation, so consider any $\delta, \epsilon : \mathbb{Q}$. Then by assumption $\bar{f}(x_\delta) \frown_{\delta+\epsilon} \bar{f}(x_\epsilon)$, hence $\bar{f}(x_\delta)(v) \sim_{\delta+\epsilon} \bar{f}(x_\epsilon)(v)$. Furthermore, $\bar{f}(\mathsf{lim}(x))$ is non-expanding because $\bar{f}(x_\epsilon)$ is such by induction hypothesis. Indeed, if $u \sim_\epsilon v$ then, for all $\epsilon : \mathbb{Q}$,

$$\bar{f}(x_{\epsilon/3})(u) \sim_{\epsilon/3} \bar{f}(x_{\epsilon/3})(v),$$

therefore $\bar{f}(\mathsf{lim}(x))(u) \sim_\epsilon \bar{f}(\mathsf{lim}(x))(v)$ by the fourth constructor of $\sim$.

We still have to check four more conditions, let us illustrate just one. Suppose $\epsilon : \mathbb{Q}_+$ and for some $\delta : \mathbb{Q}_+$ we have $\mathsf{rat}(q) \sim_{\epsilon-\delta} y_\delta$ and $\bar{f}(\mathsf{rat}(q)) \frown_{\epsilon-\delta} \bar{f}(y_\delta)$. To show $\bar{f}(\mathsf{rat}(q)) \frown_\epsilon \bar{f}(\mathsf{lim}(y))$, consider any $v : \mathbb{R}_\mathsf{c}$ and observe that

$$\bar{f}(\mathsf{rat}(q))(v) \sim_{\epsilon-\delta} \bar{f}(y_\delta)(v).$$

Therefore, by the second constructor of $\sim$, we have $\bar{f}(\mathsf{rat}(q))(v) \sim_\epsilon \bar{f}(\mathsf{lim}(y))(v)$ as required. □

We may apply Lemma 11.3.40 to any bivariate rational function which is non-expanding separately in each variable. Addition is such a function, therefore we get $+ : \mathbb{R}_\mathsf{c} \times \mathbb{R}_\mathsf{c} \to \mathbb{R}_\mathsf{c}$. Furthermore, the extension is unique as long as we require it to be non-expanding in each variable, and just as in the univariate case, identities on rationals extend to identities on reals. Since composition of non-expanding maps is again non-expanding, we may conclude that addition satisfies the usual properties, such as commutativity and associativity. Therefore, $(\mathbb{R}_\mathsf{c}, 0, +, -)$ is a commutative group.

We may also apply Lemma 11.3.40 to the functions $\min : \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ and $\max : \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$, which turns $\mathbb{R}_\mathsf{c}$ into a lattice. The partial order $\leq$ on $\mathbb{R}_\mathsf{c}$ is defined in terms of max as

$$(u \leq v) :\equiv (\max(u, v) = v).$$

The relation $\leq$ is a partial order because it is such on $\mathbb{Q}$, and the axioms of a partial order are expressible as equations in terms of min and max, so they transfer to $\mathbb{R}_\mathsf{c}$.

Another function which extends to $\mathbb{R}_\mathsf{c}$ by the same method is the absolute value $|-|$. Again, it has the expected properties because they transfer from $\mathbb{Q}$ to $\mathbb{R}_\mathsf{c}$.

From $\leq$ we get the strict order $<$ by

$$(u < v) :\equiv \exists(q, r : \mathbb{Q}).\, (u \leq \mathsf{rat}(q)) \wedge (q < r) \wedge (\mathsf{rat}(r) \leq v).$$

That is, $u < v$ holds when there merely exists a pair of rational numbers $q < r$ such that $x \leq \mathsf{rat}(q)$ and $\mathsf{rat}(r) \leq v$. It is not hard to check that $<$ is irreflexive and transitive, and has other properties that are expected for an ordered field. The archimedean principle follows directly from the definition of $<$.

Theorem 11.3.41 (Archimedean principle for $\mathbb{R}_c$)**.** *For every $u, v : \mathbb{R}_c$ such that $u < v$ there merely exists $q : \mathbb{Q}$ such that $u < q < v$.*

Proof. From $u < v$ we merely get $r, s : \mathbb{Q}$ such that $u \leq r < s \leq v$, and we may take $q :\equiv (r+s)/2$. □

We now have enough structure on $\mathbb{R}_c$ to express $u \sim_\epsilon v$ with standard concepts.

Lemma 11.3.42. *If $q : \mathbb{Q}$ and $u : \mathbb{R}_c$ satisfy $u \leq \mathsf{rat}(q)$, then for any $v : \mathbb{R}_c$ and $\epsilon : \mathbb{Q}_+$, if $u \sim_\epsilon v$ then $v \leq \mathsf{rat}(q+\epsilon)$.*

Proof. Note that the function $\max(\mathsf{rat}(q), -) : \mathbb{R}_c \to \mathbb{R}_c$ is Lipschitz with constant 1. First consider the case when $u = \mathsf{rat}(r)$ is rational. For this we use induction on v. If v is rational, then the statement is obvious. If v is $\mathsf{lim}(y)$, we assume inductively that for any ϵ, δ, if $\mathsf{rat}(r) \sim_\epsilon y_\delta$ then $y_\delta \leq \mathsf{rat}(q+\epsilon)$, i.e. $\max(\mathsf{rat}(q+\epsilon), y_\delta) = \mathsf{rat}(q+\epsilon)$.

Now assuming ϵ and $\mathsf{rat}(r) \sim_\epsilon \mathsf{lim}(y)$, we have θ such that $\mathsf{rat}(r) \sim_{\epsilon-\theta} \mathsf{lim}(y)$, hence $\mathsf{rat}(r) \sim_\epsilon y_\delta$ whenever $\delta < \theta$. Thus, the inductive hypothesis gives $\max(\mathsf{rat}(q+\epsilon), y_\delta) = \mathsf{rat}(q+\epsilon)$ for such δ. But by definition,

$$\max(\mathsf{rat}(q+\epsilon), \mathsf{lim}(y)) \equiv \mathsf{lim}(\lambda\delta.\ \max(\mathsf{rat}(q+\epsilon), y_\delta)).$$

Since the limit of an eventually constant Cauchy approximation is that constant, we have

$$\max(\mathsf{rat}(q+\epsilon), \mathsf{lim}(y)) = \mathsf{rat}(q+\epsilon),$$

hence $\mathsf{lim}(y) \leq \mathsf{rat}(q+\epsilon)$.

Now consider a general $u : \mathbb{R}_c$. Since $u \leq \mathsf{rat}(q)$ means $\max(\mathsf{rat}(q), u) = \mathsf{rat}(q)$, the assumption $u \sim_\epsilon v$ and the Lipschitz property of $\max(\mathsf{rat}(q), -)$ imply $\max(\mathsf{rat}(q), v) \sim_\epsilon \mathsf{rat}(q)$. Thus, since $\mathsf{rat}(q) \leq \mathsf{rat}(q)$, the first case implies $\max(\mathsf{rat}(q), v) \leq \mathsf{rat}(q+\epsilon)$, and hence $v \leq \mathsf{rat}(q+\epsilon)$ by transitivity of $\leq$. □

Lemma 11.3.43. *Suppose $q : \mathbb{Q}$ and $u : \mathbb{R}_c$ satisfy $u < \mathsf{rat}(q)$. Then:*

(i) For any $v : \mathbb{R}_c$ and $\epsilon : \mathbb{Q}_+$, if $u \sim_\epsilon v$ then $v < \mathsf{rat}(q+\epsilon)$.

(ii) There exists $\epsilon : \mathbb{Q}_+$ such that for any $v : \mathbb{R}_c$, if $u \sim_\epsilon v$ we have $v < \mathsf{rat}(q)$.

Proof. By definition, $u < \mathsf{rat}(q)$ means there is $r : \mathbb{Q}$ with $r < q$ and $u \leq \mathsf{rat}(r)$. Then by Lemma 11.3.42, for any ϵ, if $u \sim_\epsilon v$ then $v \leq \mathsf{rat}(r+\epsilon)$. Conclusion (i) follows immediately since $r + \epsilon < q + \epsilon$, while for (ii) we can take any $\epsilon < q - r$. □

We are now able to show that the auxiliary relation $\sim$ is what we think it is.

Theorem 11.3.44. $(u \sim_\epsilon v) \simeq (|u - v| < \mathsf{rat}(\epsilon))$ *for all $u, v : \mathbb{R}_c$ and $\epsilon : \mathbb{Q}_+$.*

Proof. The Lipschitz properties of subtraction and absolute value imply that if $u \sim_\epsilon v$, then $|u - v| \sim_\epsilon |u - u| = 0$. Thus, for the left-to-right direction, it will suffice to show that if $u \sim_\epsilon 0$, then $|u| < \mathsf{rat}(\epsilon)$. We proceed by $\mathbb{R}_c$-induction on u.

If u is rational, the statement follows immediately since absolute value and order extend the standard ones on $\mathbb{Q}_+$. If u is $\mathsf{lim}(x)$, then by roundedness we have $\theta : \mathbb{Q}_+$ with $\mathsf{lim}(x) \sim_{\epsilon-\theta} 0$. By the triangle inequality, therefore, we have $x_{\theta/3} \sim_{\epsilon-2\theta/3} 0$, so the inductive hypothesis yields $|x_{\theta/3}| < \mathsf{rat}(\epsilon - 2\theta/3)$. But $x_{\theta/3} \sim_{2\theta/3} \mathsf{lim}(x)$, hence $|x_{\theta/3}| \sim_{2\theta/3} |\mathsf{lim}(x)|$ by the Lipschitz property, so Lemma 11.3.43(i) implies $|\mathsf{lim}(x)| < \mathsf{rat}(\epsilon)$.

In the other direction, we use $\mathbb{R}_c$-induction on u and v. If both are rational, this is the first constructor of $\sim$.

If u is $\mathsf{rat}(q)$ and v is $\lim(y)$, we assume inductively that for any ϵ, δ, if $|\mathsf{rat}(q) - y_\delta| < \mathsf{rat}(\epsilon)$ then $\mathsf{rat}(q) \sim_\epsilon y_\delta$. Fix an ϵ such that $|\mathsf{rat}(q) - \lim(y)| < \mathsf{rat}(\epsilon)$. Since $\mathbb{Q}$ is order-dense in $\mathbb{R}_c$, there exists $\theta < \epsilon$ with $|\mathsf{rat}(q) - \lim(y)| < \mathsf{rat}(\theta)$. Now for any δ, η we have $\lim(y) \sim_{2\delta} y_\delta$, hence by the Lipschitz property

$$|\mathsf{rat}(q) - \lim(y)| \sim_{\delta+\eta} |\mathsf{rat}(q) - y_\delta|.$$

Thus, by Lemma 11.3.43(i), we have $|\mathsf{rat}(q) - y_\delta| < \mathsf{rat}(\theta + 2\delta)$. So by the inductive hypothesis, $\mathsf{rat}(q) \sim_{\theta+2\delta} y_\delta$, and thus $\mathsf{rat}(q) \sim_{\theta+4\delta} \lim(y)$ by the triangle inequality. Thus, it suffices to choose $\delta :\equiv (\epsilon - \theta)/4$.

The remaining two cases are entirely analogous. □

Next, we would like to equip $\mathbb{R}_c$ with multiplicative structure. For each $q : \mathbb{Q}$ the map $r \mapsto q \cdot r$ is Lipschitz with constant[1] $|q| + 1$, and so we can extend it to multiplication by q on the real numbers. Therefore $\mathbb{R}_c$ is a vector space over $\mathbb{Q}$. In general, we can define multiplication of real numbers as

$$u \cdot v :\equiv \tfrac{1}{2} \cdot ((u + v)^2 - u^2 - v^2), \tag{11.3.45}$$

so we just need squaring $u \mapsto u^2$ as a map $\mathbb{R}_c \to \mathbb{R}_c$. Squaring is not a Lipschitz map, but it is Lipschitz on every bounded domain, which allows us to patch it together. Define the open and closed intervals

$$[u, v] :\equiv \{\, x : \mathbb{R}_c \mid u \le x \le v \,\} \qquad \text{and} \qquad (u, v) :\equiv \{\, x : \mathbb{R}_c \mid u < x < v \,\}.$$

Although technically an element of $[u, v]$ or (u, v) is a Cauchy real number together with a proof, since the latter inhabits a mere proposition it is uninteresting. Thus, as is common with subset types, we generally write simply $x : [u, v]$ whenever $x : \mathbb{R}_c$ is such that $u \le x \le v$, and similarly.

Theorem 11.3.46. *There exists a unique function $(-)^2 : \mathbb{R}_c \to \mathbb{R}_c$ which extends squaring $q \mapsto q^2$ of rational numbers and satisfies*

$$\forall (n : \mathbb{N}).\, \forall (u, v : [-n, n]).\, |u^2 - v^2| \le 2 \cdot n \cdot |u - v|.$$

Proof. We first observe that for every $u : \mathbb{R}_c$ there merely exists $n : \mathbb{N}$ such that $-n \le u \le n$, see Exercise 11.7, so the map

$$e : \Big(\sum_{n:\mathbb{N}} [-n, n]\Big) \to \mathbb{R}_c \qquad \text{defined by} \qquad e(n, x) :\equiv x$$

is surjective. Next, for each $n : \mathbb{N}$, the squaring map

$$s_n : \{\, q : \mathbb{Q} \mid -n \le q \le n \,\} \to \mathbb{Q} \qquad \text{defined by} \qquad s_n(q) :\equiv q^2$$

is Lipschitz with constant $2n$, so we can use Lemma 11.3.15 to extend it to a map $\bar{s}_n : [-n, n] \to \mathbb{R}_c$ with Lipschitz constant $2n$, see Exercise 11.8 for details. The maps $\bar{s}_n$ are compatible: if $m < n$ for some $m, n : \mathbb{N}$ then s_n restricted to $[-m, m]$ must agree with s_m because both are Lipschitz, and therefore continuous in the sense of Lemma 11.3.39. Therefore, by Theorem 10.1.5 the map

$$\Big(\sum_{n:\mathbb{N}} [-n, n]\Big) \to \mathbb{R}_c, \qquad \text{given by} \qquad (n, x) \mapsto s_n(x)$$

factors uniquely through $\mathbb{R}_c$ to give us the desired function. □

[1] We defined Lipschitz constants as *positive* rational numbers.

At this point we have the ring structure of the reals and the archimedean order. To establish $\mathbb{R}_c$ as an archimedean ordered field, we still need inverses.

Theorem 11.3.47. *A Cauchy real is invertible if, and only if, it is apart from zero.*

Proof. First, suppose $u : \mathbb{R}_c$ has an inverse $v : \mathbb{R}_c$ By the archimedean principle there is $q : \mathbb{Q}$ such that $|v| < q$. Then $1 = |uv| < |u| \cdot v < |u| \cdot q$ and hence $|u| > 1/q$, which is to say that $u \# 0$.

For the converse we construct the inverse map

$$(-)^{-1} : \{\, u : \mathbb{R}_c \mid u \# 0 \,\} \to \mathbb{R}_c$$

by patching together functions, similarly to the construction of squaring in Theorem 11.3.46. We only outline the main steps. For every $q : \mathbb{Q}$ let

$$[q, \infty) :\equiv \{\, u : \mathbb{R}_c \mid q \leq u \,\} \qquad \text{and} \qquad (-\infty, q] :\equiv \{\, u : \mathbb{R}_c \mid u \leq -q \,\}.$$

Then, as q ranges over $\mathbb{Q}_+$, the types $(-\infty, q]$ and $[q, \infty)$ jointly cover $\{\, u : \mathbb{R}_c \mid u \# 0 \,\}$. On each such $[q, \infty)$ and $(-\infty, q]$ the inverse function is obtained by an application of Lemma 11.3.15 with Lipschitz constant $1/q^2$. Finally, Theorem 10.1.5 guarantees that the inverse function factors uniquely through $\{\, u : \mathbb{R}_c \mid u \# 0 \,\}$. □

We summarize the algebraic structure of $\mathbb{R}_c$ with a theorem.

Theorem 11.3.48. *The Cauchy reals form an archimedean ordered field.*

11.3.4 Cauchy reals are Cauchy complete

We constructed $\mathbb{R}_c$ by closing $\mathbb{Q}$ under limits of Cauchy approximations, so it better be the case that $\mathbb{R}_c$ is Cauchy complete. Thanks to Theorem 11.3.44 there is no difference between a Cauchy approximation $x : \mathbb{Q}_+ \to \mathbb{R}_c$ as defined in the construction of $\mathbb{R}_c$, and a Cauchy approximation in the sense of Definition 11.2.10 (adapted to $\mathbb{R}_c$).

Thus, given a Cauchy approximation $x : \mathbb{Q}_+ \to \mathbb{R}_c$ it is quite natural to expect that $\mathsf{lim}(x)$ is its limit, where the notion of limit is defined as in Definition 11.2.10. But this is so by Theorem 11.3.44 and Lemma 11.3.37. We have proved:

Theorem 11.3.49. *Every Cauchy approximation in $\mathbb{R}_c$ has a limit.*

An archimedean ordered field in which every Cauchy approximation has a limit is called **Cauchy complete**. The Cauchy reals are the least such field.

Theorem 11.3.50. *The Cauchy reals embed into every Cauchy complete archimedean ordered field.*

Proof. Suppose F is a Cauchy complete archimedean ordered field. Because limits are unique, there is an operator lim which takes Cauchy approximations in F to their limits. We define the embedding $e : \mathbb{R}_c \to F$ by $(\mathbb{R}_c, \sim)$-recursion as

$$e(\mathsf{rat}(q)) :\equiv q \qquad \text{and} \qquad e(\mathsf{lim}(x)) :\equiv \lim(e \circ x).$$

A suitable $\frown$ on F is

$$(a \frown_\epsilon b) :\equiv |a - b| < \epsilon.$$

This is a separated relation because F is archimedean. The rest of the clauses for $(\mathbb{R}_c, \sim)$-recursion are easily checked. One would also have to check that e is an embedding of ordered fields which fixes the rationals. □

11.4 Comparison of Cauchy and Dedekind reals

Let us also say something about the relationship between the Cauchy and Dedekind reals. By Theorem 11.3.48, $\mathbb{R}_{\mathsf{c}}$ is an archimedean ordered field. It is also admissible for Ω, as can be easily checked. (In case Ω is the initial σ-frame it takes a simple induction, while in other cases it is immediate.) Therefore, by Theorem 11.2.14 there is an embedding of ordered fields

$$\mathbb{R}_{\mathsf{c}} \to \mathbb{R}_{\mathsf{d}}$$

which fixes the rational numbers. (We could also obtain this from Theorems 11.2.12 and 11.3.50.) In general we do not expect $\mathbb{R}_{\mathsf{c}}$ and $\mathbb{R}_{\mathsf{d}}$ to coincide without further assumptions.

Lemma 11.4.1. *If for every $x : \mathbb{R}_{\mathsf{d}}$ there merely exists*

$$c : \prod_{q,r:\mathbb{Q}} (q < r) \to (q < x) + (x < r) \tag{11.4.2}$$

then the Cauchy and Dedekind reals coincide.

Proof. Note that the type in (11.4.2) is an untruncated variant of (11.2.3), which states that $<$ is a weak linear order. We already know that $\mathbb{R}_{\mathsf{c}}$ embeds into $\mathbb{R}_{\mathsf{d}}$, so it suffices to show that every Dedekind real merely is the limit of a Cauchy sequence of rational numbers.

Consider any $x : \mathbb{R}_{\mathsf{d}}$. By assumption there merely exists c as in the statement of the lemma, and by inhabitation of cuts there merely exist $a, b : \mathbb{Q}$ such that $a < x < b$. We construct a sequence $f : \mathbb{N} \to \{ (q, r) \in \mathbb{Q} \times \mathbb{Q} \mid q < r \}$ by recursion:

(i) Set $f(0) :\equiv (a, b)$.

(ii) Suppose $f(n)$ is already defined as (q_n, r_n) such that $q_n < r_n$. Define $s :\equiv (2q_n + r_n)/3$ and $t :\equiv (q_n + 2r_n)/3$. Then $c(s, t)$ decides between $s < x$ and $x < t$. If it decides $s < x$ then we set $f(n+1) :\equiv (s, r_n)$, otherwise $f(n+1) :\equiv (q_n, t)$.

Let us write (q_n, r_n) for the n-th term of the sequence f. Then it is easy to see that $q_n < x < r_n$ and $|q_n - r_n| \leq (2/3)^n \cdot |q_0 - r_0|$ for all $n : \mathbb{N}$. Therefore $q_0, q_1, \ldots$ and $r_0, r_1, \ldots$ are both Cauchy sequences converging to the Dedekind cut x. We have shown that for every $x : \mathbb{R}_{\mathsf{d}}$ there merely exists a Cauchy sequence converging to x. □

The lemma implies that either countable choice or excluded middle suffice for coincidence of $\mathbb{R}_{\mathsf{c}}$ and $\mathbb{R}_{\mathsf{d}}$.

Corollary 11.4.3. *If excluded middle or countable choice holds then $\mathbb{R}_{\mathsf{c}}$ and $\mathbb{R}_{\mathsf{d}}$ are equivalent.*

Proof. If excluded middle holds then $(x < y) \to (x < z) + (z < y)$ can be proved: either $x < z$ or $\neg(x < z)$. In the former case we are done, while in the latter we get $z < y$ because $z \leq x < y$. Therefore, we get (11.4.2) so that we can apply Lemma 11.4.1.

Suppose countable choice holds. The set $S = \{ (q, r) \in \mathbb{Q} \times \mathbb{Q} \mid q < r \}$ is equivalent to $\mathbb{N}$, so we may apply countable choice to the statement that x is located,

$$\forall((q, r) : S).\, (q < x) \vee (x < r).$$

Note that $(q < x) \vee (x < r)$ is expressible as an existential statement $\exists(b : \mathbf{2}).\, (b = 0_{\mathbf{2}} \to q < x) \wedge (b = 1_{\mathbf{2}} \to x < r)$. The (curried form) of the choice function is then precisely (11.4.2) so that Lemma 11.4.1 is applicable again. □

11.5 Compactness of the interval

We already pointed out that our constructions of reals are entirely compatible with classical logic. Thus, by assuming the law of excluded middle (3.4.1) and the axiom of choice (3.8.1) we could develop classical analysis, which would essentially amount to copying any standard book on analysis.

Nevertheless, anyone interested in computation, for example a numerical analyst, ought to be curious about developing analysis in a computationally meaningful setting. That analysis in a constructive setting is even possible was demonstrated by [Bis67]. As a sample of the differences and similarities between classical and constructive analysis we shall briefly discuss just one topic—compactness of the closed interval $[0,1]$ and a couple of theorems surrounding the concept.

Compactness is no exception to the common phenomenon in constructive mathematics that classically equivalent notions bifurcate. The three most frequently used notions of compactness are:

(i) **metrically compact:** "Cauchy complete and totally bounded",
(ii) **Bolzano–Weierstraß compact:** "every sequence has a convergent subsequence",
(iii) **Heine–Borel compact:** "every open cover has a finite subcover".

These are all equivalent in classical mathematics. Let us see how they fare in homotopy type theory. We can use either the Dedekind or the Cauchy reals, so we shall denote the reals just as $\mathbb{R}$. We first recall several basic definitions.

Definition 11.5.1. A **metric space** (M,d) is a set M with a map $d : M \times M \to \mathbb{R}$ satisfying, for all $x,y,z : M$,

$$d(x,y) \geq 0, \qquad d(x,y) = d(y,x),$$
$$d(x,y) = 0 \Leftrightarrow x = y, \qquad d(x,z) \leq d(x,y) + d(y,z).$$

Definition 11.5.2. A **Cauchy approximation** in M is a sequence $x : \mathbb{Q}_+ \to M$ satisfying

$$\forall(\delta,\epsilon).\, d(x_\delta, x_\epsilon) < \delta + \epsilon.$$

The **limit** of a Cauchy approximation $x : \mathbb{Q}_+ \to M$ is a point $\ell : M$ satisfying

$$\forall(\epsilon,\theta : \mathbb{Q}_+).\, d(x_\epsilon, \ell) < \epsilon + \theta.$$

A **complete metric space** is one in which every Cauchy approximation has a limit.

Definition 11.5.3. For a positive rational ϵ, an **ϵ-net** in a metric space (M,d) is an element of

$$\sum_{(n:\mathbb{N})} \sum_{(x_1,\ldots,x_n:M)} \forall(y : M).\, \exists(k \leq n).\, d(x_k, y) < \epsilon.$$

In words, this is a finite sequence of points $x_1,\ldots,x_n$ such that every point in M merely is within ϵ of some x_k.

A metric space (M,d) is **totally bounded** when it has ϵ-nets of all sizes:

$$\prod_{(\epsilon:\mathbb{Q}_+)} \sum_{(n:\mathbb{N})} \sum_{(x_1,\ldots,x_n:M)} \forall(y : M).\, \exists(k \leq n).\, d(x_k, y) < \epsilon.$$

Remark 11.5.4. In the definition of total boundedness we used sloppy notation $\sum_{(n:\mathbb{N})} \sum_{(x_1,\ldots,x_n:M)}$. Formally, we should have written $\sum_{(x:\mathsf{List}(M))}$ instead, where $\mathsf{List}(M)$ is the inductive type of finite lists from §5.1. However, that would make the rest of the statement a bit more cumbersome to express.

Note that in the definition of total boundedness we require pure existence of an ϵ-net, not mere existence. This way we obtain a function which assigns to each $\epsilon : \mathbb{Q}_+$ a specific ϵ-net. Such a function might be called a "modulus of total boundedness". In general, when porting classical metric notions to homotopy type theory, we should use propositional truncation sparingly, typically so that we avoid asking for a non-constant map from $\mathbb{R}$ to $\mathbb{Q}$ or $\mathbb{N}$. For instance, here is the "correct" definition of uniform continuity.

Definition 11.5.5. A map $f : M \to \mathbb{R}$ on a metric space is **uniformly continuous** when

$$\prod_{(\epsilon:\mathbb{Q}_+)} \sum_{(\delta:\mathbb{Q}_+)} \forall (x, y : M).\, d(x, y) < \delta \Rightarrow |f(x) - f(y)| < \epsilon.$$

In particular, a uniformly continuous map has a modulus of uniform continuity, which is a function that assigns to each ϵ a corresponding δ.

Let us show that $[0, 1]$ is compact in the first sense.

Theorem 11.5.6. *The closed interval $[0, 1]$ is complete and totally bounded.*

Proof. Given $\epsilon : \mathbb{Q}_+$, there is $n : \mathbb{N}$ such that $2/k < \epsilon$, so we may take the ϵ-net $x_i = i/k$ for $i = 0, \ldots, k-1$. This is an ϵ-net because, for every $y : [0, 1]$ there merely exists i such that $0 \leq i < k$ and $(i-1)/k < y < (i+1)/k$, and so $|y - x_i| < 2/k < \epsilon$.

For completeness of $[0, 1]$, consider a Cauchy approximation $x : \mathbb{Q}_+ \to [0, 1]$ and let ℓ be its limit in $\mathbb{R}$. Since max and min are Lipschitz maps, the retraction $r : \mathbb{R} \to [0, 1]$ defined by $r(x) :\equiv \max(0, \min(1, x))$ commutes with limits of Cauchy approximations, therefore

$$r(\ell) = r(\lim x) = \lim(r \circ x) = r(\lim x) = \ell,$$

which means that $0 \leq \ell \leq 1$, as required. □

We thus have at least one good notion of compactness in homotopy type theory. Unfortunately, it is limited to metric spaces because total boundedness is a metric notion. We shall consider the other two notions shortly, but first we prove that a uniformly continuous map on a totally bounded space has a **supremum**, i.e. an upper bound which is less than or equal to all other upper bounds.

Theorem 11.5.7. *A uniformly continuous map $f : M \to \mathbb{R}$ on a totally bounded metric space (M, d) has a supremum $m : \mathbb{R}$. For every $\epsilon : \mathbb{Q}_+$ there exists $u : M$ such that $|m - f(u)| < \epsilon$.*

Proof. Let $h : \mathbb{Q}_+ \to \mathbb{Q}_+$ be the modulus of uniform continuity of f. We define an approximation $x : \mathbb{Q}_+ \to \mathbb{R}$ as follows: for any $\epsilon : \mathbb{Q}$ total boundedness of M gives a $h(\epsilon)$-net $y_0, \ldots, y_n$. Define

$$x_\epsilon :\equiv \max(f(y_0), \ldots, f(y_n)).$$

We claim that x is a Cauchy approximation. Consider any $\epsilon, \eta : \mathbb{Q}$, so that

$$x_\epsilon \equiv \max(f(y_0), \ldots, f(y_n)) \quad \text{and} \quad x_\eta \equiv \max(f(z_0), \ldots, f(z_m))$$

for some $h(\epsilon)$-net $y_0, \ldots, y_n$ and $h(\eta)$-net $z_0, \ldots, z_m$. Every z_i is merely $h(\epsilon)$-close to some y_j, therefore $|f(z_i) - f(y_j)| < \epsilon$, from which we may conclude that

$$f(z_i) < \epsilon + f(y_j) \leq \epsilon + x_\epsilon,$$

therefore $x_\eta < \epsilon + x_\epsilon$. Symmetrically we obtain $x_\eta < \eta + x_\eta$, therefore $|x_\eta - x_\epsilon| < \eta + \epsilon$.

We claim that $m :\equiv \lim x$ is the supremum of f. To prove that $f(x) \leq m$ for all $x : M$ it suffices to show $\neg(m < f(x))$. So suppose to the contrary that $m < f(x)$. There is $\epsilon : \mathbb{Q}_+$ such that $m + \epsilon < f(x)$. But now merely for some y_i participating in the definition of x_ϵ we get $|f(x) - f(y_i) < \epsilon$, therefore $m < f(x) - \epsilon < f(y_i) \leq m$, a contradiction.

We finish the proof by showing that m satisfies the second part of the theorem, because it is then automatically a least upper bound. Given any $\epsilon : \mathbb{Q}_+$, on one hand $|m - f(x_{\epsilon/2})| < 3\epsilon/4$, and on the other $|f(x_{\epsilon/2}) - f(y_i)| < \epsilon/4$ merely for some y_i participating in the definition of $x_{\epsilon/2}$, therefore by taking $u :\equiv y_i$ we obtain $|m - f(u)| < \epsilon$ by triangle inequality. □

Now, if in Theorem 11.5.7 we also knew that M were complete, we could hope to weaken the assumption of uniform continuity to continuity, and strengthen the conclusion to existence of a point at which the supremum is attained. The usual proofs of these improvements rely on the facts that in a complete totally bounded space

(i) continuity implies uniform continuity, and

(ii) every sequence has a convergent subsequence.

The first statement follows easily from Heine–Borel compactness, and the second is just Bolzano–Weierstraß compactness. Unfortunately, these are both somewhat problematic. Let us first show that Bolzano–Weierstraß compactness implies an instance of excluded middle known as the **limited principle of omniscience**: for every $\alpha : \mathbb{N} \to \mathbf{2}$,

$$\Big(\sum_{n:\mathbb{N}} \alpha(n) = 1_\mathbf{2}\Big) + \Big(\prod_{n:\mathbb{N}} \alpha(n) = 0_\mathbf{2}\Big). \tag{11.5.8}$$

Computationally speaking, we would not expect this principle to hold, because it asks us to decide whether infinitely many values of a function are $0_\mathbf{2}$.

Theorem 11.5.9. *Bolzano–Weierstraß compactness of* $[0,1]$ *implies the limited principle of omniscience.*

Proof. Given any $\alpha : \mathbb{N} \to \mathbf{2}$, define the sequence $x : \mathbb{N} \to [0,1]$ by

$$x_n :\equiv \begin{cases} 0 & \text{if } \alpha(k) = 0_\mathbf{2} \text{ for all } k < n, \\ 1 & \text{if } \alpha(k) = 1_\mathbf{2} \text{ for some } k < n. \end{cases}$$

If the Bolzano–Weierstraß property holds, there exists a strictly increasing $f : \mathbb{N} \to \mathbb{N}$ such that $x \circ f$ is a Cauchy sequence. For a sufficiently large $n : \mathbb{N}$ the n-th term $x_{f(n)}$ is within $1/6$ of its limit. Either $x_{f(n)} < 2/3$ or $x_{f(n)} > 1/3$. If $x_{f(n)} < 2/3$ then x_n converges to 0 and so $\prod_{(n:\mathbb{N})} \alpha(n) = 0_\mathbf{2}$. If $x_{f(n)} > 1/3$ then $x_{f(n)} = 1$, therefore $\sum_{(n:\mathbb{N})} \alpha(n) = 1_\mathbf{2}$. □

While we might not mourn Bolzano–Weierstraß compactness too much, it seems harder to live without Heine–Borel compactness, as attested by the fact that both classical mathematics and Brouwer's Intuitionism accepted it. As we do not want to wade too deeply into general

topology, we shall work with basic open sets. In the case of $\mathbb{R}$ these are the open intervals with rational endpoints. A family of such intervals, indexed by a type I, would be a map

$$\mathcal{F} : I \to \{\, (q, r) : \mathbb{Q} \times \mathbb{Q} \mid q < r \,\},$$

with the idea that a pair of rationals (q, r) with $q < r$ determines the type $\{\, x : \mathbb{R} \mid q < x < r \,\}$. It is slightly more convenient to allow degenerate intervals as well, so we take a **family of basic intervals** to be a map

$$\mathcal{F} : I \to \mathbb{Q} \times \mathbb{Q}.$$

To be quite precise, a family is a dependent pair $(I, \mathcal{F})$, not just $\mathcal{F}$. A **finite family of basic intervals** is one indexed by $\{\, m : \mathbb{N} \mid m < n \,\}$ for some $n : \mathbb{N}$. We usually present it by a finite list $[(q_0, r_0), \ldots, (q_{n-1}, r_{n-1})]$. Finally, a **finite subfamily** of $(I, \mathcal{F})$ is given by a list of indices $[i_1, \ldots, i_n]$ which then determine the finite family $[\mathcal{F}(i_1), \ldots, \mathcal{F}(i_n)]$.

As long as we are aware of the distinction between a pair (q, r) and the corresponding interval $\{\, x : \mathbb{R} \mid q < x < r \,\}$, we may safely use the same notation (q, r) for both. Intersections and inclusions of intervals are expressible in terms of their endpoints:

$$\begin{aligned}(q, r) \cap (s, t) &:\equiv (\max(q, s), \min(r, t)),\\ (q, r) \subseteq (s, t) &:\equiv (q < r \Rightarrow s \leq q < r \leq t).\end{aligned}$$

We say that $(I, \lambda i.\, (q_i, r_i))$ **(pointwise) covers** $[a, b]$ when

$$\forall (x : [a, b]).\, \exists (i : I).\, q_i < x < r_i. \tag{11.5.10}$$

The **Heine–Borel compactness for** $[0, 1]$ states that every covering family of $[0, 1]$ merely has a finite subfamily which still covers $[0, 1]$.

Theorem 11.5.11. *If excluded middle holds then* $[0, 1]$ *is Heine–Borel compact.*

Proof. Assume for the purpose of reaching a contradiction that a family $(I, \lambda i.\, (a_i, b_i))$ covers $[0, 1]$ but no finite subfamily does. We construct a sequence of closed intervals $[q_n, r_n]$ which are nested, their sizes shrink to 0, and none of them is covered by a finite subfamily of $(I, \lambda i.\, (a_i, b_i))$.

We set $[q_0, r_0] :\equiv [0, 1]$. Assuming $[q_n, r_n]$ has been constructed, let $s :\equiv (2q_n + r_n)/3$ and $t :\equiv (q_n + 2r_n)/3$. Both $[q_n, t]$ and $[s, r_n]$ are covered by $(I, \lambda i.\, (a_i, b_i))$, but they cannot both have a finite subcover, or else so would $[q_n, r_n]$. Either $[q_n, t]$ has a finite subcover or it does not. If it does we set $[q_{n+1}, r_{n+1}] :\equiv [s, r_n]$, otherwise we set $[q_{n+1}, r_{n+1}] :\equiv [q_n, t]$.

The sequences $q_0, q_1, \ldots$ and $r_0, r_1, \ldots$ are both Cauchy and they converge to a point $x : [0, 1]$ which is contained in every $[q_n, r_n]$. There merely exists $i : I$ such that $a_i < x < b_i$. Because the sizes of the intervals $[q_n, r_n]$ shrink to zero, there is $n : \mathbb{N}$ such that $a_i < q_n \leq x \leq r_n < b_i$, but this means that $[q_n, r_n]$ is covered by a single interval (a_i, b_i), while at the same time it has no finite subcover. A contradiction. □

Without excluded middle, or a pinch of Brouwerian Intuitionism, we seem to be stuck. Nevertheless, Heine–Borel compactness of $[0, 1]$ *can* be recovered in a constructive setting, in a fashion that is still compatible with classical mathematics! For this to be done, we need to revisit the notion of cover. The trouble with (11.5.10) is that the truncated existential allows a space to be covered in any haphazard way, and so computationally speaking, we stand no chance of merely extracting a finite subcover. By removing the truncation we get

$$\prod_{(x:[0,1])} \sum_{(i:I)} q_i < x < r_i, \tag{11.5.12}$$

which might help, were it not too demanding of covers. With this definition we could not even show that $(0,3)$ and $(2,5)$ cover $[1,4]$ because that would amount to exhibiting a non-constant map $[1,4] \to \mathbf{2}$, see Exercise 11.6. Here we can take a lesson from "pointfree topology" (i.e. locale theory): the notion of cover ought to be expressed in terms of open sets, without reference to points. Such a "holistic" view of space will then allow us to analyze the notion of cover, and we shall be able to recover Heine–Borel compactness. Locale theory uses power sets, which we could obtain by assuming propositional resizing; but instead we can steal ideas from the predicative cousin of locale theory, which is called "formal topology".

Suppose that we have a family $(I, \mathcal{F})$ and an interval (a, b). How might we express the fact that (a, b) is covered by the family, without referring to points? Here is one: if (a, b) equals some $\mathcal{F}(i)$ then it is covered by the family. And another one: if (a, b) is covered by some other family $(J, \mathcal{G})$, and in turn each $\mathcal{G}(j)$ is covered by $(I, \mathcal{F})$, then (a, b) is covered $(I, \mathcal{F})$. Notice that we are listing *rules* which can be used to *deduce* that $(I, \mathcal{F})$ covers (a, b). We should find sufficiently good rules and turn them into an inductive definition.

Definition 11.5.13. The **inductive cover** $\lhd$ is a mere relation

$$\lhd : (\mathbb{Q} \times \mathbb{Q}) \to \Big(\sum_{I:\mathcal{U}} (I \to \mathbb{Q} \times \mathbb{Q})\Big) \to \mathsf{Prop}$$

defined inductively by the following rules, where q, r, s, t are rational numbers and $(I, \mathcal{F})$, $(J, \mathcal{G})$ are families of basic intervals:

(i) *reflexivity:* $\mathcal{F}(i) \lhd (I, \mathcal{F})$ for all $i : I$,
(ii) *transitivity:* if $(q, r) \lhd (J, \mathcal{G})$ and $\forall (j : J).\, \mathcal{G}(j) \lhd (I, \mathcal{F})$ then $(q, r) \lhd (I, \mathcal{F})$,
(iii) *monotonicity:* if $(q, r) \subseteq (s, t)$ and $(s, t) \lhd (I, \mathcal{F})$ then $(q, r) \lhd (I, \mathcal{F})$,
(iv) *localization:* if $(q, r) \lhd (I, \mathcal{F})$ then $(q, r) \cap (s, t) \lhd (I, \lambda i.\, (\mathcal{F}(i) \cap (s, t)))$.
(v) if $q < s < t < r$ then $(q, r) \lhd [(q, t), (r, s)]$,
(vi) $(q, r) \lhd (\{\, (s, t) : \mathbb{Q} \times \mathbb{Q} \mid q < s < t < r \,\}, \lambda u.\, u)$.

The definition should be read as a higher-inductive type in which the listed rules are point constructors, and the type is (-1)-truncated. The first four clauses are of a general nature and should be intuitively clear. The last two clauses are specific to the real line: one says that an interval may be covered by two intervals if they overlap, while the other one says that an interval may be covered from within. Incidentally, if $r \leq q$ then (q, r) is covered by the empty family by the last clause.

Inductive covers enjoy the Heine–Borel property, the proof of which requires a lemma.

Lemma 11.5.14. *Suppose $q < s < t < r$ and $(q, r) \lhd (I, \mathcal{F})$. Then there merely exists a finite subfamily of $(I, \mathcal{F})$ which inductively covers (s, t).*

Proof. We prove the statement by induction on $(q, r) \lhd (I, \mathcal{F})$. There are six cases:

(i) Reflexivity: if $(q, r) = \mathcal{F}(i)$ then by monotonicity (s, t) is covered by the finite subfamily $[\mathcal{F}(i)]$.
(ii) Transitivity: suppose $(q, r) \lhd (J, \mathcal{G})$ and $\forall (j : J).\, \mathcal{G}(j) \lhd (I, \mathcal{F})$. By the inductive hypothesis there merely exists $[\mathcal{G}(j_1), \ldots, \mathcal{G}(j_n)]$ which covers (s, t). Again by the inductive hypothesis, each of $\mathcal{G}(j_k)$ is covered by a finite subfamily of $(I, \mathcal{F})$, and we can collect these into a finite subfamily which covers (s, t).

(iii) Monotonicity: if $(q,r) \subseteq (u,v)$ and $(u,v) \lhd (I,\mathcal{F})$ then we may apply the inductive hypothesis to $(u,v) \lhd (I,\mathcal{F})$ because $u < s < t < v$.

(iv) Localization: suppose $(q',r') \lhd (I,\mathcal{F})$ and $(q,r) = (q',r') \cap (a,b)$. Because $q' < s < t < r'$, by the inductive hypothesis there is a finite subcover $[\mathcal{F}(i_1),\ldots,\mathcal{F}(i_n)]$ of (s,t). We also know that $a < s < t < b$, therefore $(s,t) = (s,t) \cap (a,b)$ is covered by $[\mathcal{F}(i_1) \cap (a,b),\ldots,\mathcal{F}(i_n) \cap (a,b)]$, which is a finite subfamily of $(I, \lambda i.\, (\mathcal{F}(i) \cap (a,b)))$.

(v) If $(q,r) \lhd [(q,v),(u,r)]$ for some $q < u < v < r$ then by monotonicity $(s,t) \lhd [(q,v),(u,r)]$.

(vi) Finally, $(s,t) \lhd (\{\, (u,v) : \mathbb{Q} \times \mathbb{Q} \mid q < u < v < r \,\}, \lambda z.\, z)$ by reflexivity. □

Say that $(I,\mathcal{F})$ **inductively covers** $[a,b]$ when there merely exists $\epsilon : \mathbb{Q}_+$ such that $(a-\epsilon, b+\epsilon) \lhd (I,\mathcal{F})$.

Corollary 11.5.15. *A closed interval is Heine–Borel compact for inductive covers.*

Proof. Suppose $[a,b]$ is inductively covered by $(I,\mathcal{F})$, so there merely is $\epsilon : \mathbb{Q}_+$ such that $(a-\epsilon, b+\epsilon) \lhd (I,\mathcal{F})$. By Lemma 11.5.14 there is a finite subcover of $(a-\epsilon/2, b+\epsilon/2)$, which is therefore a finite subcover of $[a,b]$. □

Experience from formal topology shows that the rules for inductive covers are sufficient for a constructive development of pointfree topology. But we can also provide our own evidence that they are a reasonable notion.

Theorem 11.5.16.

(i) An inductive cover is also a pointwise cover.

(ii) Assuming excluded middle, a pointwise cover is also an inductive cover.

Proof.

(i) Consider a family of basic intervals $(I,\mathcal{F})$, where we write $(q_i,r_i) :\equiv \mathcal{F}(i)$, an interval (a,b) inductively covered by $(I,\mathcal{F})$, and x such that $a < x < b$. We prove by induction on $(a,b) \lhd (I,\mathcal{F})$ that there merely exists $i : I$ such that $q_i < x < r_i$. Most cases are pretty obvious, so we show just two. If $(a,b) \lhd (I,\mathcal{F})$ by reflexivity, then there merely is some $i : I$ such that $(a,b) = (q_i,r_i)$ and so $q_i < x < r_i$. If $(a,b) \lhd (I,\mathcal{F})$ by transitivity via $(J, \lambda j.\, (s_j,t_j))$ then by the inductive hypothesis there merely is $j : J$ such that $s_j < x < t_j$, and then since $(s_j,t_j) \lhd (I,\mathcal{F})$ again by the inductive hypothesis there merely exists $i : I$ such that $q_i < x < r_i$. Other cases are just as exciting.

(ii) Suppose $(I, \lambda i.\, (q_i,r_i))$ pointwise covers (a,b). By Item (vi) of Definition 11.5.13 it suffices to show that $(I, \lambda i.\, (q_i,r_i))$ inductively covers (c,d) whenever $a < c < d < b$, so consider such c and d. By Theorem 11.5.11 there is a finite subfamily $[i_1,\ldots,i_n]$ which already pointwise covers $[c,d]$, and hence (c,d). Let $\epsilon : \mathbb{Q}_+$ be a Lebesgue number for $(q_{i_1},r_{i_1}),\ldots,(q_{i_n},r_{i_n})$ as in Exercise 11.12. There is a positive $k : \mathbb{N}$ such that $2(d-c)/k < \min(1,\epsilon)$. For $0 \leq i \leq k$ let

$$c_k :\equiv ((k-i)c + id)/k.$$

The intervals (c_0,c_2), (c_1,c_3), …, (c_{k-2},c_k) inductively cover (c,d) by repeated use of transitivity and Item (v) in Definition 11.5.13. Because their widths are below ϵ each of them is contained in some (q_i,r_i), and we may use transitivity and monotonicity to conclude that $(I, \lambda i.\, (q_i,r_i))$ inductively cover (c,d). □

The upshot of the previous theorem is that, as far as classical mathematics is concerned, there is no difference between a pointwise and an inductive cover. In particular, since it is consistent to assume excluded middle in homotopy type theory, we cannot exhibit an inductive cover which fails to be a pointwise cover. Or to put it in a different way, the difference between pointwise and inductive covers is not what they cover but in the *proofs* that they cover.

We could write another book by going on like this, but let us stop here and hope that we have provided ample justification for the claim that analysis can be developed in homotopy type theory. The curious reader should consult Exercise 11.13 for constructive versions of the mean value theorem.

11.6 The surreal numbers

In this section we consider another example of a higher inductive-inductive type, which draws together many of our threads: Conway's field No of *surreal numbers* [Con76]. The surreal numbers are the natural common generalization of the (Dedekind) real numbers (§11.2) and the ordinal numbers (§10.3). Conway, working in classical mathematics with excluded middle and Choice, defines a surreal number to be a pair of *sets* of surreal numbers, written $\{ L \mid R \}$, such that every element of L is strictly less than every element of R. This obviously looks like an inductive definition, but there are three issues with regarding it as such.

Firstly, the definition requires the relation of (strict) inequality between surreals, so that relation must be defined simultaneously with the type No of surreals. (Conway avoids this issue by first defining *games*, which are like surreals but omit the compatibility condition on L and R.) As with the relation $\sim$ for the Cauchy reals, this simultaneous definition could *a priori* be either inductive-inductive or inductive-recursive. We will choose to make it inductive-inductive, for the same reasons we made that choice for $\sim$.

Moreover, we will define strict inequality $<$ and non-strict inequality $\leq$ for surreals separately (and mutually inductively). Conway defines $<$ in terms of $\leq$, in a way which is sensible classically but not constructively. Furthermore, a negative definition of $<$ would make it unacceptable as a hypothesis of the constructor of a higher inductive type (see §5.6).

Secondly, Conway says that L and R in $\{ L \mid R \}$ should be "sets of surreal numbers", but the naive meaning of this as a predicate $\mathsf{No} \to \mathsf{Prop}$ is not positive, hence cannot be used as input to an inductive constructor. However, this would not be a good type-theoretic translation of what Conway means anyway, because in set theory the surreal numbers form a proper class, whereas the sets L and R are true (small) sets, not arbitrary subclasses of No. In type theory, this means that No will be defined relative to a universe $\mathcal{U}$, but will itself belong to the next higher universe $\mathcal{U}'$, like the sets Ord and Card of ordinals and cardinals, the cumulative hierarchy V, or even the Dedekind reals in the absence of propositional resizing. We will then require the "sets" L and R of surreals to be $\mathcal{U}$-small, and so it is natural to represent them by *families* of surreals indexed by some $\mathcal{U}$-small type. (This is all exactly the same as what we did with the cumulative hierarchy in §10.5.) That is, the constructor of surreals will have type

$$\prod_{\mathcal{L},\mathcal{R}:\mathcal{U}} (\mathcal{L} \to \mathsf{No}) \to (\mathcal{R} \to \mathsf{No}) \to (\text{some condition}) \to \mathsf{No}$$

which is indeed strictly positive.

Finally, after giving the mutual definitions of No and its ordering, Conway declares two surreal numbers x and y to be *equal* if $x \leq y$ and $y \leq x$. This is naturally read as passing to a quotient of the set of "pre-surreals" by an equivalence relation. However, in the absence of the

axiom of choice, such a quotient presents the same problem as the quotient in the usual construction of Cauchy reals: it will no longer be the case that a pair of families *of surreals* yield a new surreal $\{ L \mid R \}$, since we cannot necessarily "lift" L and R to families of pre-surreals. Of course, we can solve this problem in the same way we did for Cauchy reals, by using a *higher* inductive-inductive definition.

Definition 11.6.1. The type No of **surreal numbers**, along with the relations $< : \mathsf{No} \to \mathsf{No} \to \mathcal{U}$ and $\leq : \mathsf{No} \to \mathsf{No} \to \mathcal{U}$, are defined higher inductive-inductively as follows. The type No has the following constructors.

- For any $\mathcal{L}, \mathcal{R} : \mathcal{U}$ and functions $\mathcal{L} \to \mathsf{No}$ and $\mathcal{R} \to \mathsf{No}$, whose values we write as x^L and x^R for $L : \mathcal{L}$ and $R : \mathcal{R}$ respectively, if $\forall(L : \mathcal{L}).\, \forall(R : \mathcal{R}).\, x^L < x^R$, then there is a surreal number x.
- For any $x, y : \mathsf{No}$ such that $x \leq y$ and $y \leq x$, we have $\mathsf{eq}_{\mathsf{No}}(x, y) : x = y$.

We will refer to the inputs of the first constructor as a **cut**. If x is the surreal number constructed from a cut, then the notation x^L will implicitly assume $L : \mathcal{L}$, and similarly x^R will assume $R : \mathcal{R}$. In this way we can usually avoid naming the indexing types $\mathcal{L}$ and $\mathcal{R}$, which is convenient when there are many different cuts under discussion. Following Conway, we call x^L a *left option* of x and x^R a *right option*.

The path constructor implies that different cuts can define the same surreal number. Thus, it does not make sense to speak of the left or right options of an arbitrary surreal number x, unless we also know that x is defined by a particular cut. Thus in what follows we will say, for instance, "given a cut defining a surreal number x" in contrast to "given a surreal number x".

The relation $\leq$ has the following constructors.

- Given cuts defining two surreal numbers x and y, if $x^L < y$ for all L, and $x < y^R$ for all R, then $x \leq y$.
- Propositional truncation: for any $x, y : \mathsf{No}$, if $p, q : x \leq y$, then $p = q$.

And the relation $<$ has the following constructors.

- Given cuts defining two surreal numbers x and y, if there is an L such that $x \leq y^L$, then $x < y$.
- Given cuts defining two surreal numbers x and y, if there is an R such that $x^R \leq y$, then $x < y$.
- Propositional truncation: for any $x, y : \mathsf{No}$, if $p, q : x < y$, then $p = q$.

We compare this with Conway's definitions:

- If L, R are any two sets of numbers, and no member of L is $\geq$ any member of R, then there is a number $\{ L \mid R \}$. All numbers are constructed in this way.
- $x \geq y$ iff (no $x^R \leq y$ and $x \leq$ no y^L).
- $x = y$ iff ($x \geq y$ and $y \geq x$).
- $x > y$ iff ($x \geq y$ and $y \not\geq x$).

The inclusion of $x \geq y$ in the definition of $x > y$ is unnecessary if all objects are [surreal] numbers rather than "games". Thus, Conway's $<$ is just the negation of his $\geq$, so that his condition for $\{ L \mid R \}$ to be a surreal is the same as ours. Negating Conway's $\leq$ and canceling double negations, we arrive at our definition of $<$, and we can then reformulate his $\leq$ in terms of $<$ without negations.

We can immediately populate No with many surreal numbers. Like Conway, we write

$$\{\, x, y, z, \dots \mid u, v, w, \dots \,\}$$

for the surreal number defined by a cut where $\mathcal{L} \to \mathsf{No}$ and $\mathcal{R} \to \mathsf{No}$ are families described by $x, y, z, \dots$ and $u, v, w, \dots$. Of course, if $\mathcal{L}$ or $\mathcal{R}$ are $\mathbf{0}$, we leave the corresponding part of the notation empty. There is an unfortunate clash with the standard notation $\{\, x : A \mid P(x) \,\}$ for subsets, but we will not use the latter in this section.

- We define $\iota_{\mathbb{N}} : \mathbb{N} \to \mathsf{No}$ recursively by

$$\begin{aligned} \iota_{\mathbb{N}}(0) &:\equiv \{\, \mid \,\}, \\ \iota_{\mathbb{N}}(\mathsf{succ}(n)) &:\equiv \{\, \iota_{\mathbb{N}}(n) \mid \,\}. \end{aligned}$$

  That is, $\iota_{\mathbb{N}}(0)$ is defined by the cut consisting of $\mathbf{0} \to \mathsf{No}$ and $\mathbf{0} \to \mathsf{No}$. Similarly, $\iota_{\mathbb{N}}(\mathsf{succ}(n))$ is defined by $\mathbf{1} \to \mathsf{No}$ (picking out $\iota_{\mathbb{N}}(n)$) and $\mathbf{0} \to \mathsf{No}$.
- Similarly, we define $\iota_{\mathbb{Z}} : \mathbb{Z} \to \mathsf{No}$ using the sign-case recursion principle (Lemma 6.10.12):

$$\begin{aligned} \iota_{\mathbb{Z}}(0) &:\equiv \{\, \mid \,\}, \\ \iota_{\mathbb{Z}}(n+1) &:\equiv \{\, \iota_{\mathbb{Z}}(n) \mid \,\} && n \geq 0, \\ \iota_{\mathbb{Z}}(n-1) &:\equiv \{\, \mid \iota_{\mathbb{Z}}(n) \,\} && n \leq 0. \end{aligned}$$

- By a **dyadic rational** we mean a pair (a, n) where $a : \mathbb{Z}$ and $n : \mathbb{N}$, and such that if $n > 0$ then a is odd. We will write it as $a/2^n$, and identify it with the corresponding rational number. If $\mathbb{Q}_D$ denotes the set of dyadic rationals, we define $\iota_{\mathbb{Q}_D} : \mathbb{Q}_D \to \mathsf{No}$ by induction on n:

$$\begin{aligned} \iota_{\mathbb{Q}_D}(a/2^0) &:\equiv \iota_{\mathbb{Z}}(a), \\ \iota_{\mathbb{Q}_D}(a/2^n) &:\equiv \{\, a/2^n - 1/2^n \mid a/2^n + 1/2^n \,\}, \quad \text{for } n > 0. \end{aligned}$$

  Here we use the fact that if $n > 0$ and a is odd, then $a/2^n \pm 1/2^n$ is a dyadic rational with a smaller denominator than $a/2^n$.
- We define $\iota_{\mathbb{R}_{\mathsf{d}}} : \mathbb{R}_{\mathsf{d}} \to \mathsf{No}$, where $\mathbb{R}_{\mathsf{d}}$ is (any version of) the Dedekind reals from §11.2, by

$$\iota_{\mathbb{R}_{\mathsf{d}}}(x) :\equiv \{\, q \in \mathbb{Q}_D \text{ such that } q < x \mid q \in \mathbb{Q}_D \text{ such that } x < q \,\}.$$

  Unlike in the previous cases, it is not obvious that this extends $\iota_{\mathbb{Q}_D}$ when we regard dyadic rationals as Dedekind reals. This follows from the simplicity theorem (Theorem 11.6.2).
- Recall the type Ord of *ordinals* from §10.3, which is well-ordered by the relation $<$, where $A < B$ means that $A = B_{/b}$ for some $b : B$. We define $\iota_{\mathsf{Ord}} : \mathsf{Ord} \to \mathsf{No}$ by well-founded recursion (Lemma 10.3.7) on Ord:

$$\iota_{\mathsf{Ord}}(A) :\equiv \{\, \iota_{\mathsf{Ord}}(A_{/a}) \text{ for all } a : A \mid \,\}.$$

  It will also follow from the simplicity theorem that ι_{Ord} restricted to finite ordinals agrees with $\iota_{\mathbb{N}}$. (We caution the reader, however, that unlike the above examples, ι_{Ord} is not constructively injective unless we restrict it to a smaller class of ordinals; see Exercises 11.16 and 11.17.)

- A few more interesting examples taken from Conway:

$$\begin{aligned}
\omega &:\equiv \{\, 0,1,2,3,\dots \mid \,\} \qquad \text{(also an ordinal)}\\
-\omega &:\equiv \{\, \mid \dots,-3,-2,-1,0\,\}\\
1/\omega &:\equiv \{\, 0 \mid 1, \tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{8}, \dots\,\}\\
\omega - 1 &:\equiv \{\, 0,1,2,3,\dots \mid \omega\,\}\\
\omega/2 &:\equiv \{\, 0,1,2,3,\dots \mid \dots,\omega-2,\omega-1,\omega\,\}.
\end{aligned}$$

In identifying surreal numbers presented by different cuts, the following simple observation is useful.

Theorem 11.6.2 (Conway's simplicity theorem)**.** *Suppose x and z are surreal numbers defined by cuts, and that the following hold.*

- *$x^L < z < x^R$ for all L and R.*
- *For every left option z^L of z, there exists a left option $x^{L'}$ with $z^L \le x^{L'}$.*
- *For every right option z^R of z, there exists a right option $x^{R'}$ with $x^{R'} \le z^R$.*

Then $x = z$.

Proof. Applying the path constructor of No, we must show $x \le z$ and $z \le x$. The first entails showing $x^L < z$ for all L, which we assumed, and $x < z^R$ for all R. But by assumption, for any z^R there is an $x^{R'}$ with $x^{R'} \le z^R$ hence $x < z^R$ as desired. Thus $x \le z$; the proof of $z \le x$ is symmetric. □

In order to say much more about surreal numbers, however, we need their induction principle. The mutual induction principle for $(\mathsf{No}, \le, <)$ applies to three families of types:

$$\begin{aligned}
A &: \mathsf{No} \to \mathcal{U}\\
B &: \prod_{(x,y:\mathsf{No})} \prod_{(a:A(x))} \prod_{(b:A(y))} (x \le y) \to \mathcal{U}\\
C &: \prod_{(x,y:\mathsf{No})} \prod_{(a:A(x))} \prod_{(b:A(y))} (x < y) \to \mathcal{U}.
\end{aligned}$$

As with the induction principle for Cauchy reals, it is helpful to think of B and C as families of relations between the types $A(x)$ and $A(y)$. Thus we write $B(x,y,a,b,\xi)$ as $(x,a) \preccurlyeq^{\xi} (y,b)$ and $C(x,y,a,b,\xi)$ as $(x,a) \prec^{\xi} (y,b)$. Similarly, we usually omit the ξ since it inhabits a mere proposition and so is uninteresting, and we may often omit x and y as well, writing simply $a \preccurlyeq b$ or $a \prec b$. With these notations, the hypotheses of the induction principle are the following.

- For any cut defining a surreal number x, together with

  (i) for each L, an element $a^L : A(x^L)$, and
  (ii) for each R, an element $a^R : A(x^R)$, such that
  (iii) for all L and R we have $(x^L, a^L) \prec (x^R, a^R)$

  there is a specified element $f_a : A(x)$. We call such data a **dependent cut** over the cut defining x.
- For any $x, y : \mathsf{No}$ with $a : A(x)$ and $b : A(y)$, if $x \le y$ and $y \le x$ and also $(x,a) \preccurlyeq (y,b)$ and $(y,b) \preccurlyeq (x,a)$, then $a =^{A}_{\mathsf{eq}_{\mathsf{No}}} b$.

- Given cuts defining two surreal numbers x and y, and dependent cuts a over x and b over y, such that for all L we have $x^L < y$ and $(x^L, a^L) \lhd (y, f_b)$, and for all R we have $x < y^R$ and $(x, f_a) \lhd (y^R, b^R)$, then $(x, f_a) \trianglelefteq (y, f_b)$.
- $\trianglelefteq$ takes values in mere propositions.
- Given cuts defining two surreal numbers x and y, dependent cuts a over x and b over y, and an L_0 such that $x \leq y^{L_0}$ and $(x, f_a) \trianglelefteq (y^{L_0}, b^{L_0})$, we have $(x, f_a) \lhd (y, f_b)$.
- Given cuts defining two surreal numbers x and y, dependent cuts a over x and b over y, and an R_0 such that $x^{R_0} \leq y$ together with $(x^{R_0}, a^{R_0}), \trianglelefteq (y, f_b)$, we have $(x, f_a) \lhd (y, f_b)$.
- $\lhd$ takes values in mere propositions.

Under these hypotheses we deduce a function $f : \prod_{(x:\mathsf{No})} A(x)$ such that

$$\begin{aligned} f(x) &\equiv f_{f[x]} \\ (x \leq y) &\Rightarrow (x, f(x)) \trianglelefteq (y, f(y)) \\ (x < y) &\Rightarrow (x, f(x)) \lhd (y, f(y)). \end{aligned} \tag{11.6.3}$$

In the computation rule (11.6.3) for the point constructor, x is a surreal number defined by a cut, and $f[x]$ denotes the dependent cut over x defined by applying f (and using the fact that f takes $<$ to $\lhd$). As usual, we will generally use pattern-matching notation, where the definition of f on a cut $\{\, x^L \mid x^R \,\}$ may use the symbols $f(x^L)$ and $f(x^R)$ and the assumption that they form a dependent cut.

As with the Cauchy reals, we have special cases resulting from trivializing some of A, $\trianglelefteq$, and $\lhd$. Taking $\trianglelefteq$ and $\lhd$ to be constant at $\mathbf{1}$, we have No**-induction**, which for simplicity we state only for mere properties:

- Given $P : \mathsf{No} \to \mathsf{Prop}$, if $P(x)$ holds whenever x is a surreal number defined by a cut such that $P(x^L)$ and $P(x^R)$ hold for all L and R, then $P(x)$ holds for all $x : \mathsf{No}$.

This should be compared with Conway's remark:

> In general when we wish to establish a proposition $P(x)$ for all numbers x, we will prove it inductively by deducing $P(x)$ from the truth of all the propositions $P(x^L)$ and $P(x^R)$. We regard the phrase "all numbers are constructed in this way" as justifying the legitimacy of this procedure.

With No-induction, we can prove

Theorem 11.6.4 (Conway's Theorem 0)**.**

(i) For any $x : \mathsf{No}$, we have $x \leq x$.

(ii) For any $x : \mathsf{No}$ defined by a cut, we have $x^L < x$ and $x < x^R$ for all L and R.

Proof. Note first that if $x \leq x$, then whenever x occurs as a left option of some cut y, we have $x < y$ by the first constructor of $<$, and similarly whenever x occurs as a right option of a cut y, we have $y < x$ by the second constructor of $<$. In particular, (i)⇒(ii).

We prove (i) by No-induction on x. Thus, assume x is defined by a cut such that $x^L \leq x^L$ and $x^R \leq x^R$ for all L and R. But by our observation above, these assumptions imply $x^L < x$ and $x < x^R$ for all L and R, yielding $x \leq x$ by the constructor of $\leq$. □

Corollary 11.6.5. No *is a 0-type.*

Proof. The mere relation $R(x,y) :\equiv (x \le y) \wedge (y \le x)$ implies identity by the path constructor of No, and contains the diagonal by Theorem 11.6.4(i). Thus, Theorem 7.2.2 applies. □

By contrast, Conway's Theorem 1 (transitivity of $\le$) is somewhat harder to establish with our definition; see Corollary 11.6.17.

We will also need the joint recursion principle, $(\mathsf{No}, \le, <)$**-recursion**. It is convenient to state this as follows. Suppose A is a type equipped with relations $\preccurlyeq : A \to A \to \mathsf{Prop}$ and $\lhd : A \to A \to \mathsf{Prop}$. Then we can define $f : \mathsf{No} \to A$ by doing the following.

(i) For any x defined by a cut, assuming $f(x^L)$ and $f(x^R)$ to be defined such that $f(x^L) \lhd f(x^R)$ for all L and R, we must define $f(x)$. (We call this the *primary clause* of the recursion.)
(ii) Prove that $\preccurlyeq$ is *antisymmetric*: if $a \preccurlyeq b$ and $b \preccurlyeq a$, then $a = b$.
(iii) For x, y defined by cuts such that $x^L < y$ for all L and $x < y^R$ for all R, and assuming inductively that $f(x^L) \lhd f(y)$ for all L, $f(x) \lhd f(y^R)$ for all R, and also that $f(x^L) \lhd f(x^R)$ and $f(y^L) \lhd f(y^R)$ for all L and R, we must prove $f(x) \preccurlyeq f(y)$.
(iv) For x, y defined by cuts and an L_0 such that $x \le y^{L_0}$, and assuming inductively that $f(x) \preccurlyeq f(y^{L_0})$, and also that $f(x^L) \lhd f(x^R)$ and $f(y^L) \lhd f(y^R)$ for all L and R, we must prove $f(x) \lhd f(y)$.
(v) For x, y defined by cuts and an R_0 such that $x^{R_0} \le y$, and assuming inductively that $f(x^{R_0}) \preccurlyeq f(y)$, and also that $f(x^L) \lhd f(x^R)$ and $f(y^L) \lhd f(y^R)$ for all L and R, we must prove $f(x) \lhd f(y)$.

The last three clauses can be more concisely described by saying we must prove that f (as defined in the first clause) takes $\le$ to $\preccurlyeq$ and $<$ to $\lhd$. We will refer to these properties by saying that f *preserves inequalities*. Moreover, in proving that f preserves inequalities, we may assume the particular instance of $\le$ or $<$ to be obtained from one of its constructors, and we may also use inductive hypotheses that f preserves all inequalities appearing in the input to that constructor.

If we succeed at (i)–(v) above, then we obtain $f : \mathsf{No} \to A$, which computes on cuts as specified by (i), and which preserves all inequalities:

$$\forall (x, y : \mathsf{No}).\, \Big((x \le y) \to (f(x) \preccurlyeq f(y))\Big) \wedge \Big((x < y) \to (f(x) \lhd f(y))\Big).$$

Like $(\mathbb{R}_\mathsf{c}, \sim)$-recursion for the Cauchy reals, this recursion principle is essential for defining functions on No, since we cannot first define a function on "pre-surreals" and only later prove that it respects the notion of equality.

Example 11.6.6. Let us define the *negation* function $\mathsf{No} \to \mathsf{No}$. We apply the joint recursion principle with $A :\equiv \mathsf{No}$, with $(x \preccurlyeq y) :\equiv (y \le x)$, and $(x \lhd y) :\equiv (y < x)$. Clearly this $\preccurlyeq$ is antisymmetric.

For the main clause in the definition, we assume x defined by a cut, with $-x^L$ and $-x^R$ defined such that $-x^L \lhd -x^R$ for all L and R. By definition, this means $-x^R < -x^L$ for all L and R, so we can define $-x$ by the cut $\{\, -x^R \mid -x^L \,\}$. This notation, which follows Conway, refers to the cut whose left options are indexed by the type $\mathcal{R}$ indexing the right options of x, and whose right options are indexed by the type $\mathcal{L}$ indexing the left options of x, with the corresponding families $\mathcal{R} \to \mathsf{No}$ and $\mathcal{L} \to \mathsf{No}$ defined by composing those for x with negation.

We now have to verify that f preserves inequalities.

- For $x \le y$, we may assume $x^L < y$ for all L and $x < y^R$ for all R, and show $-y \le -x$. But inductively, we may assume $-y < -x^L$ and $-y^R < -x$, which gives the desired result, by definition of $-y$, $-x$, and the constructor of $\le$.

- For $x < y$, in the first case when it arises from some $x \leq y^{L_0}$, we may inductively assume $-y^{L_0} \leq -x$, in which case $-y < -x$ follows by the constructor of $<$.
- Similarly, if $x < y$ arises from $x^{R_0} \leq y$, the inductive hypothesis is $-y \leq -x^R$, yielding $-y < -x$ again.

To do much more than this, however, we will need to characterize the relations $\leq$ and $<$ more explicitly, as we did for the Cauchy reals in Theorem 11.3.32. Also as there, we will have to simultaneously prove a couple of essential properties of these relations, in order for the induction to go through.

Theorem 11.6.7. *There are relations* $\preceq : \mathsf{No} \to \mathsf{No} \to \mathsf{Prop}$ *and* $\prec : \mathsf{No} \to \mathsf{No} \to \mathsf{Prop}$ *such that if* x *and* y *are surreals defined by cuts, then*

$$(x \preceq y) :\equiv \big(\forall (L).\, x^L \prec y\big) \wedge \big(\forall (R).\, x \prec y^R\big)$$
$$(x \prec y) :\equiv \big(\exists (L).\, x \preceq y^L\big) \vee \big(\exists (R).\, x^R \preceq y\big).$$

Moreover, we have

$$(x \prec y) \to (x \preceq y) \tag{11.6.8}$$

and all the reasonable transitivity properties making $\prec$ *and* $\preceq$ *into a "bimodule" over* $\leq$ *and* $<$*:*

$$\begin{array}{ll} (x \leq y) \to (y \preceq z) \to (x \preceq z) & (x \preceq y) \to (y \leq z) \to (x \preceq z) \\ (x \leq y) \to (y \prec z) \to (x \prec z) & (x \preceq y) \to (y < z) \to (x \prec z) \\ (x < y) \to (y \preceq z) \to (x \prec z) & (x \prec y) \to (y \leq z) \to (x \prec z). \end{array} \tag{11.6.9}$$

Proof. We define $\preceq$ and $\prec$ by double $(\mathsf{No}, \leq, <)$-induction on x, y. The first induction is a simple recursion, whose codomain is the subset A of $(\mathsf{No} \to \mathsf{Prop}) \times (\mathsf{No} \to \mathsf{Prop})$ consisting of pairs of predicates of which one implies the other and which satisfy "transitivity on the right", i.e. (11.6.8) and the right column of (11.6.9) with $(x \preceq -)$ and $(x \prec -)$ replaced by the two given predicates. As in the proof of Theorem 11.3.16, we regard these predicates as half of binary relations, writing them as $y \mapsto (\diamondsuit \preceq y)$ and $y \mapsto (\diamondsuit \prec y)$, with $\diamondsuit$ denoting the pair of relations. We equip A with the following two relations:

$$(\diamondsuit \trianglelefteq \heartsuit) :\equiv \forall (y : \mathsf{No}).\, \Big((\heartsuit \preceq y) \to (\diamondsuit \preceq y)\Big) \wedge \Big((\heartsuit \prec y) \to (\diamondsuit \prec y)\Big),$$
$$(\diamondsuit \triangleleft \heartsuit) :\equiv \forall (y : \mathsf{No}).\, \Big((\heartsuit \preceq y) \to (\diamondsuit \prec y)\Big).$$

Note that $\trianglelefteq$ is antisymmetric, since if $\diamondsuit \trianglelefteq \heartsuit$ and $\heartsuit \trianglelefteq \diamondsuit$, then $(\heartsuit \preceq y) \Leftrightarrow (\diamondsuit \preceq y)$ and $(\heartsuit \prec y) \Leftrightarrow (\diamondsuit \prec y)$ for all y, hence $\diamondsuit = \heartsuit$ by univalence for mere propositions and function extensionality. Moreover, to say that a function $\mathsf{No} \to A$ preserves inequalities is exactly to say that, when regarded as a pair of binary relations on No, it satisfies "transitivity on the left" (the left column of (11.6.9)).

Now for the primary clause of the recursion, we assume given x defined by a cut, and relations $(x^L \prec -)$, $(x^R \prec -)$, $(x^L \preceq -)$, and $(x^R \preceq -)$ for all L and R, of which the strict ones imply the non-strict ones, which satisfy transitivity on the right, and such that

$$\forall (L, R).\, \forall (y : \mathsf{No}).\, \Big((x^R \preceq y) \to (x^L \prec y)\Big). \tag{11.6.10}$$

We now have to define $(x \prec y)$ and $(x \preceq y)$ for all y. Here in contrast to Theorem 11.3.16, rather than a nested recursion, we use a nested induction, in order to be able to inductively use

transitivity on the left with respect to the inequalities $x^L < x$ and $x < x^R$. Define $A' : \mathsf{No} \to \mathcal{U}$ by taking $A'(y)$ to be the subset A' of $\mathsf{Prop} \times \mathsf{Prop}$ consisting of two mere propositions, denoted $\triangle \preceq y$ and $\triangle \prec y$ (with $\triangle : A'(y)$), such that

$$(\triangle \prec y) \to (\triangle \preceq y) \tag{11.6.11}$$
$$\forall(L).\, (\triangle \preceq y) \to (x^L \prec y) \tag{11.6.12}$$
$$\forall(R).\, (x^R \preceq y) \to (\triangle \prec y). \tag{11.6.13}$$

Using notation analogous to $\leqslant$ and $\lhd$, we equip A' with the two relations defined for $\triangle : A'(y)$ and $\square : A'(z)$ by

$$(\triangle \sqsubseteq \square) :\equiv \Big((\triangle \preceq y) \to (\square \preceq z)\Big) \wedge \Big((\triangle \prec y) \to (\square \prec z)\Big)$$
$$(\triangle \sqsubset \square) :\equiv \Big((\triangle \preceq y) \to (\square \prec z)\Big).$$

Again, $\sqsubseteq$ is evidently antisymmetric in the appropriate sense. Moreover, a function $\prod_{(y:\mathsf{No})} A'(y)$ which preserves inequalities is precisely a pair of predicates of which one implies the other, which satisfy transitivity on the right, and transitivity on the left with respect to the inequalities $x^L < x$ and $x < x^R$. Thus, this inner induction will provide what we need to complete the primary clause of the outer recursion.

For the primary clause of the inner induction, we assume also given y defined by a cut, and properties $(x \prec y^L)$, $(x \prec y^R)$, $(x \preceq y^L)$, and $(x \preceq y^R)$ for all L and R, with the strict ones implying the non-strict ones, transitivity on the left with respect to $x^L < x$ and $x < x^R$, and on the right with respect to $y^L < y^R$. We can now give the definitions specified in the theorem statement:

$$(x \preceq y) :\equiv (\forall(L).\, x^L \prec y) \wedge (\forall(R).\, x \prec y^R), \tag{11.6.14}$$
$$(x \prec y) :\equiv (\exists(L).\, x \preceq y^L) \vee (\exists(R).\, x^R \preceq y). \tag{11.6.15}$$

For this to define an element of $A'(y)$, we must show first that $(x \prec y) \to (x \preceq y)$. The assumption $x \prec y$ has two cases. On one hand, if there is L_0 with $x \preceq y^{L_0}$, then by transitivity on the right with respect to $y^{L_0} < y^R$, we have $x \prec y^R$ for all R. Moreover, by transitivity on the left with respect to $x^L < x$, we have $x^L \prec y^{L_0}$ for any L, hence $x^L \prec y$ by transitivity on the right. Thus, $x \preceq y$.

On the other hand, if there is R_0 with $x^{R_0} \preceq y$, then by (11.6.10), we have $x^L \prec y$ for all L. And by transitivity on the left and right with respect to $x < x^{R_0}$ and $y < y^R$, we have $x \prec y^R$ for any R. Thus, $x \preceq y$.

We also need to show that these definitions are transitive on the left with respect to $x^L < x$ and $x < x^R$. But if $x \preceq y$, then $x^L \prec y$ for all L by definition; while if $x^R \preceq y$, then $x \prec y$ also by definition.

Thus, (11.6.14) and (11.6.15) do define an element of $A'(y)$. We now have to verify that this definition preserves inequalities, as a dependent function into A', i.e. that these relations are transitive on the right. Remember that in each case, we may assume inductively that they are transitive on the right with respect to all inequalities arising in the inequality constructor.

- Suppose $x \preceq y$ and $y \leq z$, the latter arising from $y^L < z$ and $y < z^R$ for all L and R. Then the inductive hypothesis (of the inner recursion) applied to $y < z^R$ yields $x \prec z^R$ for any R. Moreover, by definition $x \preceq y$ implies that $x^L \prec y$ for any L, so by the inductive hypothesis of the outer recursion we have $x^L \prec z$. Thus, $x \preceq z$.

- Suppose $x \preceq y$ and $y < z$. First, suppose $y < z$ arises from $y \leq z^{L_0}$. Then the inner inductive hypothesis applied to $y \leq z^{L_0}$ yields $x \preceq z^{L_0}$, hence $x \prec z$.

  Second, suppose $y < z$ arises from $y^{R_0} \leq z$. Then by definition, $x \preceq y$ implies $x \prec y^{R_0}$, and then the inner inductive hypothesis for $y^{R_0} \leq z$ yields $x \prec z$.
- Suppose $x \prec y$ and $y \leq z$, the latter arising from $y^L < z$ and $y < z^R$ for all L and R. By definition, $x \prec y$ implies there merely exists R_0 with $x^{R_0} \preceq y$ or L_0 with $x \preceq y^{L_0}$. If $x^{R_0} \preceq y$, then the outer inductive hypothesis yields $x^{R_0} \preceq z$, hence $x \prec z$. If $x \preceq y^{L_0}$, then the inner inductive hypothesis for $y^{L_0} < z$ (which holds by the constructor of $y \leq z$) yields $x \prec z$.

This completes the inner induction. Thus, for any x defined by a cut, we have $(x \prec -)$ and $(x \preceq -)$ defined by (11.6.14) and (11.6.15), and transitive on the right.

To complete the outer recursion, we need to verify these definitions are transitive on the left. After a No-induction on z, we end up with three cases that are essentially identical to those just described above for transitivity on the right. Hence, we omit them. □

Theorem 11.6.16. *For any $x, y : \mathsf{No}$ we have $(x < y) = (x \prec y)$ and $(x \leq y) = (x \preceq y)$.*

Proof. From left to right, we use $(\mathsf{No}, \leq, <)$-induction where $A(x) :\equiv \mathbf{1}$, with $\preceq$ and $\prec$ supplying the relations $\trianglelefteq$ and $\triangleleft$. In all the constructor cases, x and y are defined by cuts, so the definitions of $\preceq$ and $\prec$ evaluate, and the inductive hypotheses apply.

From right to left, we use No-induction to assume that x and y are defined by cuts. But now the definitions of $\preceq$ and $\prec$, and the inductive hypotheses, supply exactly the data required for the relevant constructors of $\leq$ and $<$. □

Corollary 11.6.17. *The relations $\leq$ and $<$ on No satisfy*

$$\forall (x, y : \mathsf{No}).\, (x < y) \to (x \leq y)$$

and are transitive:

$$\begin{aligned}
&(x \leq y) \to (y \leq z) \to (x \leq z)\\
&(x \leq y) \to (y < z) \to (x < z)\\
&(x < y) \to (y \leq z) \to (x < z).
\end{aligned}$$

As with the Cauchy reals, the joint $(\mathsf{No}, \leq, <)$-recursion principle remains essential when defining all operations on No.

Example 11.6.18. We define $+ : \mathsf{No} \to \mathsf{No} \to \mathsf{No}$ by recursion on the first argument, followed by induction on the second argument. For the outer recursion, we take the codomain to be the subset of $\mathsf{No} \to \mathsf{No}$ consisting of functions g such that $(x < y) \to (g(x) < g(y))$ and $(x \leq y) \to (g(x) \leq g(y))$ for all x, y. For such g, h we define $(g \trianglelefteq h) :\equiv \forall (x : \mathsf{No}).\, g(x) \leq h(x)$ and $(g \triangleleft h) :\equiv \forall (x : \mathsf{No}).\, g(x) < h(x)$. Clearly $\trianglelefteq$ is antisymmetric.

For the primary clause of the recursion, we suppose x defined by a cut, that the functions $(x^L + -)$ and $(x^R + -)$ are defined, preserve inequalities, and satisfy $x^L + y < x^R + y$, and we define $(x + -)$. As in Theorem 11.6.7, rather than an inner recursion, we use an inner induction into the family $A : \mathsf{No} \to \mathcal{U}$, where $A(y)$ is the subset of those $z : \mathsf{No}$ such that each $x^L + y < z$ and each $x^R + y > z$. We equip A with the relations $\leq$ and $<$ induced from No, so that antisymmetry is obvious. For the primary clause of the inner recursion, we suppose also y defined by a cut, with each $x + y^L$ and $x + y^R$ defined and satisfying $x^L + y^L < x + y^L$, $x^L + y^R < x + y^R$, $x +$

$y^L < x^R + y^L$, and $x + y^R < x^R + y^R$ (these come from the additional conditions imposed on elements of $A(y)$), and also $x + y^L < x + y^R$ (since the elements $x + y^L$ and $x + y^R$ of $A(y)$ form a dependent cut). Now we give Conway's definition:

$$x + y :\equiv \{\, x^L + y, x + y^L \mid x^R + y, x + y^R \,\}.$$

In other words, the left options of $x + y$ are all numbers of the form $x^L + y$ for some left option x^L, or $x + y^L$ for some left option y^L. We must show that each of these left options is less than each of these right options:

- $x^L + y < x^R + y$ by the outer inductive hypothesis.
- $x^L + y < x^L + y^R < x + y^R$, the first since $(x^L + -)$ preserves inequalities, and the second since $x + y^R : A(y^R)$.
- $x + y^L < x^R + y^L < x^R + y$, the first since $x + y^L : A(y^L)$ and the second since $(x^R + -)$ preserves inequalities.
- $x + y^L < x + y^R$ by the inner inductive hypothesis (specifically, the fact that we have a dependent cut).

We also have to show that $x + y$ thusly defined lies in $A(y)$, i.e. that $x^L + y < x + y$ and $x + y < x^R + y$; but this is true by Theorem 11.6.4(ii).

Next we have to verify that the definition of $(x + -)$ preserves inequality:

- If $y \leq z$ arises from knowing that $y^L < z$ and $y < z^R$ for all L and R, then the inner inductive hypothesis gives $x + y^L < x + z$ and $x + y < x + z^R$, while the outer inductive hypotheses give $x^L + y \leq x^L + z$ and $x^R + y \leq x^R + z$. Moreover, since $x^R + y$ is by definition a right option of $x + y$, we have $x + y < x^R + y$. Similarly, we find that $x^L + z$ is a left option of $x + z$, so that $x^L + z < x + z$. Thus, using transitivity, we have $x^L + y < x + z$ and $x + y < x^R + z$; so we may conclude $x + y \leq x + z$ by the constructor of $\leq$.
- If $y < z$ arises from an L_0 with $y \leq z^{L_0}$, then inductively $x + y \leq x + z^{L_0}$, hence $x + y < x + z$ since $x + z^{L_0}$ is a right option of $x + z$.
- Similarly, if $y < z$ arises from $y^{R_0} \leq z$, then $x + y < x + z$ since $x + y^{R_0} \leq x + z$.

This completes the inner induction. For the outer recursion, we have to verify that $+$ preserves inequality on the left as well. After an No-induction, this proceeds in exactly the same way.

In the Appendix to Part Zero of [Con76], Conway discusses how the surreal numbers may be formalized in ZFC set theory: by iterating along the ordinals and passing to sets of representatives of lowest rank for each equivalence class, or by representing numbers with "sign-expansions". He then remarks that

> The curiously complicated nature of these constructions tells us more about the nature of formalizations within ZF than about our system of numbers...

and goes on to advocate for a general theory of "permissible kinds of construction" which should include

(i) Objects may be created from earlier objects in any reasonably constructive fashion.

(ii) Equality among the created objects can be any desired equivalence relation.

Condition (i) can be naturally read as justifying general principles of *inductive definition*, such as those presented in §§5.6 and 5.7. In particular, the condition of strict positivity for constructors can be regarded as a formalization of what it means to be "reasonably constructive". Condition (ii) then suggests we should extend this to *higher* inductive definitions of all sorts, in which we can impose path constructors making objects equal in any reasonable way. For instance, in the next paragraph Conway says:

> ...we could also, for instance, freely create a new object (x, y) and call it the ordered pair of x and y. We could also create an ordered pair $[x, y]$ different from (x, y) but co-existing with it...If instead we wanted to make (x, y) into an unordered pair, we could define equality by means of the equivalence relation $(x, y) = (z, t)$ if and only if $x = z, y = t$ *or* $x = t, y = z$.

The freedom to introduce new objects with new names, generated by certain forms of constructors, is precisely what we have in the theory of inductive definitions. Just as with our two copies of the natural numbers $\mathbb{N}$ and $\mathbb{N}'$ in §5.2, if we wrote down an identical definition to the cartesian product type $A \times B$, we would obtain a distinct product type $A \times' B$ whose canonical elements we could freely write as $[x, y]$. And we could make one of these a type of unordered pairs by adding a suitable path constructor.

To be sure, Conway's point was not to complain about ZF in particular, but to argue against all foundational theories at once:

> ...this proposal is not of any particular theory as an alternative to ZF... What is proposed is instead that we give ourselves the freedom to create arbitrary mathematical theories of these kinds, but prove a metatheorem which ensures once and for all that any such theory could be formalized in terms of any of the standard foundational theories.

One might respond that, in fact, univalent foundations is not one of the "standard foundational theories" which Conway had in mind, but rather the *metatheory* in which we may express our ability to create new theories, and about which we may prove Conway's metatheorem. For instance, the surreal numbers are one of the "mathematical theories" Conway has in mind, and we have seen that they can be constructed and justified inside univalent foundations. Similarly, Conway remarked earlier that

> ...set theory would be such a theory, sets being constructed from earlier ones by processes corresponding to the usual axioms, and the equality relation being that of having the same members.

This description closely matches the higher-inductive construction of the cumulative hierarchy of set theory in §10.5. Conway's metatheorem would then correspond to the fact we have referred to several times that we can construct a model of univalent foundations inside ZFC (which is outside the scope of this book).

However, univalent foundations is so rich and powerful in its own right that it would be foolish to relegate it to only a metatheory in which to construct set-like theories. We have seen that even at the level of sets (0-types), the higher inductive types in univalent foundations yield direct constructions of objects by their universal properties (§6.11), such as a constructive theory of Cauchy completion (§11.3). But most importantly, the potential to model homotopy theory and category theory directly in the foundational system (Chapters 8 and 9) gives univalent foundations an advantage which no set-theoretic foundation can match.

Notes

Defining algebraic operations on Dedekind reals, especially multiplication, is both somewhat tricky and tedious. There are several ways to get arithmetic going: each has its own advantages,

but they all seem to require some technical work. For instance, Richman [Ric08] defines multiplication on the Dedekind reals first on the positive cuts and then extends it algebraically to all Dedekind cuts, while Conway [Con76] has observed that the definition of multiplication for surreal numbers works well for Dedekind reals.

Our treatment of the Dedekind reals borrows many ideas from [BT09] where the Dedekind reals are constructed in the context of Abstract Stone Duality. This is a (restricted) form of simply typed λ-calculus with a distinguished object Σ which classifies open sets, and by duality also the closed ones. In [BT09] you can also find detailed proofs of the basic properties of arithmetical operations.

The fact that $\mathbb{R}_c$ is the least Cauchy complete archimedean ordered field, as was proved in Theorem 11.3.50, indicates that our Cauchy reals probably coincide with the Escardó-Simpson reals [ES01]. It would be interesting to check whether this is really the case. The notion of Escardó-Simpson reals, or more precisely the corresponding closed interval, is interesting because it can be stated in any category with finite products.

In constructive set theory augmented by the "regular extension axiom", one may also try to define Cauchy completion by closing under limits of Cauchy sequences with a transfinite iteration. It would also be interesting to check whether this construction agrees with ours.

It is constructive folklore that coincidence of Cauchy and Dedekind reals requires dependent choice but it is less well known that countable choice suffices. Recall that **dependent choice** states that for a total relation R on A, by which we mean $\forall(x : A).\, \exists(y : A).\, R(x, y)$, and for any $a : A$ there merely exists $f : \mathbb{N} \to A$ such that $f(0) = a$ and $R(f(n), f(n+1))$ for all $n : \mathbb{N}$. Our Corollary 11.4.3 uses the typical trick for converting an application of dependent choice to one using countable choice. Namely, we use countable choice once to make in advance all the choices that could come up, and then use the choice function to avoid the dependent choices.

The intricate relationship between various notions of compactness in a constructive setting is discussed in [BIS02]. Palmgren [Pal07] has a good comparison between pointwise analysis and pointfree topology.

The surreal numbers were defined by [Con76], using a sort of inductive definition but without justifying it explicitly in terms of any foundational system. For this reason, some later authors have tended to use sign-expansions or other more explicit presentations which can be coded more obviously into set theory. The idea of representing them in type theory was first considered by Hancock, while Setzer and Forsberg [FS12] noted that the surreals and their inequality relations $<$ and $\leq$ naturally form an inductive-inductive definition. The *higher* inductive-inductive version presented here, which builds in the correct notion of equality for surreals, is new.

Exercises

Exercise 11.1. Give an alternative definition of the Dedekind reals by first defining the square and then use Eq. (11.3.45). Check that one obtains a commutative ring.

Exercise 11.2. Suppose we remove the boundedness condition in Definition 11.2.1. Then we obtain the **extended reals** which contain $-\infty :\equiv (\mathbf{0}, \mathbb{Q})$ and $\infty :\equiv (\mathbb{Q}, \mathbf{0})$. Which definitions of arithmetical operations on cuts still make sense for extended reals? What algebraic structure do we get?

Exercise 11.3. By considering one-sided cuts we obtain **lower** and **upper** Dedekind reals, respectively. For example, a lower real is given by a predicate $L : \mathbb{Q} \to \Omega$ which is

(i) *inhabited:* $\exists(q : \mathbb{Q}).\, L(q)$ and
(ii) *rounded:* $L(q) = \exists(r : \mathbb{Q}).\, q < r \wedge L(r)$.

(We could also require $\exists(r : \mathbb{Q}).\, \neg L(r)$ to exclude the cut $\infty :\equiv \mathbb{Q}$.) Which arithmetical operations can you define on the lower reals? In particular, what happens with the additive inverse?

Exercise 11.4. Suppose we remove the locatedness condition in Definition 11.2.1. Then we obtain the **interval domain** $\mathbb{I}$ because cuts are allowed to have "gaps", which are just intervals. Define the partial order $\sqsubseteq$ on $\mathbb{I}$ by

$$((L, U) \sqsubseteq (L', U')) :\equiv (\forall(q : \mathbb{Q}).\, L(q) \Rightarrow L'(q)) \wedge (\forall(q : \mathbb{Q}).\, U(q) \Rightarrow U'(q)).$$

What are the maximal elements of $\mathbb{I}$ with respect to $\mathbb{I}$? Define the "endpoint" operations which assign to an element of the interval domain its lower and upper endpoints. Are the endpoints reals, lower reals, or upper reals (see Exercise 11.3)? Which definitions of arithmetical operations on cuts still make sense for the interval domain?

Exercise 11.5. Show that, for all $x, y : \mathbb{R}_{\mathsf{d}}$,

$$\neg(x < y) \Rightarrow y \leq x$$

and

$$(x \leq y) \simeq \Big(\prod_{\epsilon:\mathbb{Q}_+} x < y + \epsilon\Big).$$

Does $\neg(x \leq y)$ imply $y < x$?

Exercise 11.6.

(i) Assuming excluded middle, construct a non-constant map $\mathbb{R}_{\mathsf{d}} \to \mathbb{Z}$.
(ii) Suppose $f : \mathbb{R}_{\mathsf{d}} \to \mathbb{Z}$ is a map such that $f(0) = 0$ and $f(x) \neq 0$ for all $x > 0$. Derive from this the limited principle of omniscience (11.5.8).

Exercise 11.7. Show that in an ordered field F, density of $\mathbb{Q}$ and the traditional archimedean axiom are equivalent:

$$(\forall(x, y : F).\, x < y \Rightarrow \exists(q : \mathbb{Q}).\, x < q < y) \Leftrightarrow (\forall(x : F).\, \exists(k : \mathbb{Z}).\, x < k).$$

Exercise 11.8. Suppose $a, b : \mathbb{Q}$ and $f : \{\, q : \mathbb{Q} \mid a \leq q \leq b \,\} \to \mathbb{R}_{\mathsf{c}}$ is Lipschitz with constant L. Show that there exists a unique extension $\bar{f} : [a, b] \to \mathbb{R}_{\mathsf{c}}$ of f which is Lipschitz with constant L. Hint: rather than redoing Lemma 11.3.15 for closed intervals, observe that there is a retraction $r : \mathbb{R}_{\mathsf{c}} \to [-n, n]$ and apply Lemma 11.3.15 to $f \circ r$.

Exercise 11.9. Generalize the construction of $\mathbb{R}_{\mathsf{c}}$ to construct the Cauchy completion of any metric space. First, think about which notion of real numbers is most natural as the codomain for the distance function of a metric space. Does it matter? Next, work out the details of two constructions:

(i) Follow the construction of Cauchy reals to define the completion of a metric space as an inductive-inductive type closed under limits of Cauchy sequences.
(ii) Use the following construction due to Lawvere [Law74] and Richman [Ric00], where the completion of a metric space (M, d) is given as the type of **locations**. A location is a function $f : M \to \mathbb{R}$ such that

(a) $f(x) \geq |f(y) - d(x,y)|$ for all $x, y : M$, and

(b) $\inf_{x \in M} f(x) = 0$, by which we mean $\forall(\epsilon : \mathbb{Q}_+).\, \exists(x : M).\, |f(x)| < \epsilon$ and $\forall(x : M).\, f(x) \geq 0$.

The idea is that f looks like it is measuring the distance from a point.

Finally, prove the following universal property of metric completions: a locally uniformly continuous map from a metric space to a Cauchy complete metric space extends uniquely to a locally uniformly continuous map on the completion. (We say that a map is **locally uniformly continuous** if it is uniformly continuous on open balls.)

Exercise 11.10. **Markov's principle** says that for all $f : \mathbb{N} \to \mathbf{2}$,

$$(\neg\neg\exists(n : \mathbb{N}).\, f(n) = 1_{\mathbf{2}}) \Rightarrow \exists(n : \mathbb{N}).\, f(n) = 1_{\mathbf{2}}.$$

This is a particular instance of the law of double negation (3.4.2). Show that $\forall(x, y : \mathbb{R}_{\mathsf{d}}).\, x \neq y \Rightarrow x \# y$ implies Markov's principle. Does the converse hold as well?

Exercise 11.11. Verify that the following "no zero divisors" property holds for the real numbers: $xy \# 0 \Leftrightarrow x \# 0 \wedge y \# 0$.

Exercise 11.12. Suppose $(q_1, r_1), \ldots, (q_n, r_n)$ pointwise cover (a, b). Then there is $\epsilon : \mathbb{Q}_+$ such that whenever $a < x < y < b$ and $|x - y| < \epsilon$ then there merely exists i such that $q_i < x < r_i$ and $q_i < y < r_i$. Such an ϵ is called a **Lebesgue number** for the given cover.

Exercise 11.13. Prove the following approximate version of the intermediate value theorem:

> *If $f : [0,1] \to \mathbb{R}$ is uniformly continuous and $f(0) < 0 < f(1)$ then for every $\epsilon : \mathbb{Q}_+$ there merely exists $x : [0,1]$ such that $|f(x)| < \epsilon$.*

Hint: do not try to use the bisection method because it leads to the axiom of choice. Instead, approximate f with a piecewise linear map. How do you construct a piecewise linear map?

Exercise 11.14. Check whether everything in [Knu74] can be done using the higher inductive-inductive surreals of §11.6.

Exercise 11.15. Recall the function $\iota_{\mathbb{R}_{\mathsf{d}}} : \mathbb{R}_{\mathsf{d}} \to \mathsf{No}$ defined on page 399.

(i) Show that $\iota_{\mathbb{R}_{\mathsf{d}}}$ is injective.

(ii) There are obvious extensions of $\iota_{\mathbb{R}_{\mathsf{d}}}$ to the extended reals (Exercise 11.2) and the interval domain (Exercise 11.4). Are they injective?

Exercise 11.16. Show that the function $\iota_{\mathsf{Ord}} : \mathsf{Ord} \to \mathsf{No}$ defined on page 399 is injective if and only if LEM holds.

Exercise 11.17. Define a type POrd equipped with binary relations $\leq$ and $<$ by mimicking the definition of No but using only left options.

(i) Construct a map $j : \mathsf{POrd} \to \mathsf{No}$ and show that it is an embedding.

(ii) Show that POrd is an ordinal (in the next higher universe, like Ord) under the relation $<$.

(iii) Assuming propositional resizing, show that POrd is equivalent to the subset

$$\{\, A : \mathsf{Ord} \mid \mathsf{isPlump}(A) \,\}$$

of Ord from Exercise 10.14. Conclude that $\iota_{\mathsf{Ord}} : \mathsf{Ord} \to \mathsf{No}$ is injective when restricted to plump ordinals.

In the absence of propositional resizing, we may still refer to elements of POrd (or their images in No) as **plump ordinals**.

Exercise 11.18. Define a surreal number to be a **pseudo-ordinal** if it is equal to a cut $\{ x^L \mid \}$ with no right options (but its left options may themselves have right options). Show that the statement "every pseudo-ordinal is a plump ordinal" is equivalent to LEM.

Exercise 11.19. Note that Theorem 11.6.7 and Example 11.6.18 both use a similar pattern to define a function $\mathsf{No} \to \mathsf{No} \to B$: an outer No-recursion whose codomain is the set of order-preserving functions $\mathsf{No} \to B$, followed by an inner No-induction into a family $A : \mathsf{No} \to \mathcal{U}$ where $A(y)$ is a subset of B ensuring that the inequalities $x^L < x$ and $x < x^R$ are also preserved. Formulate and prove a general principle of "double No-recursion" that generalizes these proofs.

Appendix

Appendix

Formal type theory

Just as one can develop mathematics in set theory without explicitly using the axioms of Zermelo–Fraenkel set theory, in this book we have developed mathematics in univalent foundations without explicitly referring to a formal system of homotopy type theory. Nevertheless, it is important to *have* a precise description of homotopy type theory as a formal system in order to, for example,

- state and prove its metatheoretic properties, including logical consistency,
- construct models, e.g. in simplicial sets, model categories, higher toposes, etc., and
- implement it in proof assistants like COQ or AGDA.

Even the logical consistency of homotopy type theory, namely that in the empty context there is no term $a : \mathbf{0}$, is not obvious: if we had erroneously chosen a definition of equivalence for which $\mathbf{0} \simeq \mathbf{1}$, then univalence would imply that $\mathbf{0}$ has an element, since $\mathbf{1}$ does. Nor is it obvious that, for example, our definition of $\mathbb{S}^1$ as a higher inductive type yields a type which behaves like the ordinary circle.

There are two aspects of type theory which we must pin down before addressing such questions. Recall from the Introduction that type theory comprises a set of rules specifying when the judgments $a : A$ and $a \equiv a' : A$ hold—for example, products are characterized by the rule that whenever $a : A$ and $b : B$, $(a, b) : A \times B$. To make this precise, we must first define precisely the syntax of terms—the objects $a, a', A, \ldots$ which these judgments relate; then, we must define precisely the judgments and their rules of inference—the manner in which judgments can be derived from other judgments.

In this appendix, we present two formulations of Martin-Löf type theory, and of the extensions that constitute homotopy type theory. The first presentation (Appendix A.1) describes the syntax of terms and the forms of judgments as an extension of the untyped λ-calculus, while leaving the rules of inference informal. The second (Appendix A.2) defines the terms, judgments, and rules of inference inductively in the style of natural deduction, as is customary in much type-theoretic literature.

Preliminaries

In Chapter 1, we presented the two basic **judgments** of type theory. The first, $a : A$, asserts that a term a has type A. The second, $a \equiv b : A$, states that the two terms a and b are **judgmentally equal** at type A. These judgments are inductively defined by a set of inference rules described in Appendix A.2.

To construct an element a of a type A is to derive $a : A$; in the book, we give informal arguments which describe the construction of a, but formally, one must specify a precise term a and a full derivation that $a : A$.

However, the main difference between the presentation of type theory in the book and in this appendix is that here judgments are explicitly formulated in an ambient **context**, or list of assumptions, of the form

$$x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n.$$

An element $x_i : A_i$ of the context expresses the assumption that the variable x_i has type A_i. The variables $x_1, \ldots, x_n$ appearing in the context must be distinct. We abbreviate contexts with the letters Γ and Δ.

The judgment $a : A$ in context Γ is written

$$\Gamma \vdash a : A$$

and means that $a : A$ under the assumptions listed in Γ. When the list of assumptions is empty, we write simply

$$\vdash a : A$$

or

$$\cdot \vdash a : A$$

where $\cdot$ denotes the empty context. The same applies to the equality judgment

$$\Gamma \vdash a \equiv b : A$$

However, such judgments are sensible only for **well-formed** contexts, a notion captured by our third and final judgment

$$(x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n)\ \mathsf{ctx}$$

expressing that each A_i is a type in the context $x_1 : A_1, x_2 : A_2, \ldots, x_{i-1} : A_{i-1}$. In particular, therefore, if $\Gamma \vdash a : A$ and $\Gamma\ \mathsf{ctx}$, then we know that each A_i contains only the variables $x_1, \ldots, x_{i-1}$, and that a and A contain only the variables $x_1, \ldots, x_n$.

In informal mathematical presentations, the context is implicit. At each point in a proof, the mathematician knows which variables are available and what types they have, either by historical convention (n is usually a number, f is a function, etc.) or because variables are explicitly introduced with sentences such as "let x be a real number". We discuss some benefits of using explicit contexts in Appendices A.2.4 and A.2.5.

We write $B[a/x]$ for the **substitution** of a term a for free occurrences of the variable x in the term B, with possible capture-avoiding renaming of bound variables, as discussed in §1.2. The general form of substitution

$$B[a_1, \ldots, a_n/x_1, \ldots, x_n]$$

substitutes expressions $a_1, \ldots, a_n$ for the variables $x_1, \ldots, x_n$ simultaneously.

To **bind a variable** x **in an expression** B means to incorporate both of them into a larger expression, called an **abstraction**, whose purpose is to express the fact that x is "local" to B, i.e., it is not to be confused with other occurrences of x appearing elsewhere. Bound variables are familiar to programmers, but less so to mathematicians. Various notations are used for binding, such as $x \mapsto B$, $\lambda x.\, B$, and $x\,.\,B$, depending on the situation. We may write $C[a]$ for the substitution of a term a for the variable in the abstracted expression, i.e., we may define $(x.B)[a]$ to be $B[a/x]$.

As discussed in §1.2, changing the name of a bound variable everywhere within an expression ("α-conversion") does not change the expression. Thus, to be very precise, an expression is an equivalence class of syntactic forms which differ in names of bound variables.

One may also regard each variable x_i of a judgment

$$x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n \vdash a : A$$

to be bound in its **scope**, consisting of the expressions $A_{i+1}, \ldots, A_n$, a, and A.

A.1 The first presentation

The objects and types of our type theory may be written as terms using the following syntax, which is an extension of λ-calculus with *variables* $x, x', \ldots$, *primitive constants* $c, c', \ldots$, *defined constants* $f, f', \ldots$, and term forming operations

$$t ::= x \mid \lambda x.\, t \mid t(t') \mid c \mid f$$

The notation used here means that a term t is either a variable x, or it has the form $\lambda x.\, t$ where x is a variable and t is a term, or it has the form $t(t')$ where t and t' are terms, or it is a primitive constant c, or it is a defined constant f. The syntactic markers 'λ', '(', ')', and '.' are punctuation for guiding the human eye.

We use $t(t_1, \ldots, t_n)$ as an abbreviation for the repeated application $t(t_1)(t_2)\ldots(t_n)$. We may also use *infix* notation, writing $t_1 \star t_2$ for $\star(t_1, t_2)$ when $\star$ is a primitive or defined constant.

Each defined constant has zero, one or more **defining equations**. There are two kinds of defined constant. An *explicit* defined constant f has a single defining equation

$$f(x_1, \ldots, x_n) :\equiv t,$$

where t does not involve f. For example, we might introduce the explicit defined constant $\circ$ with defining equation

$$\circ(x, y)(z) :\equiv x(y(z)),$$

and use infix notation $x \circ y$ for $\circ(x, y)$. This of course is just composition of functions.

The second kind of defined constant is used to specify a (parameterized) mapping $f(x_1, \ldots, x_n, x)$, where x ranges over a type whose elements are generated by zero or more primitive constants. For each such primitive constant c there is a defining equation of the form

$$f(x_1, \ldots, x_n, c(y_1, \ldots, y_m)) :\equiv t,$$

where f may occur in t, but only in such a way that it is clear that the equations determine a totally defined function. The paradigm examples of such defined functions are the functions defined by primitive recursion on the natural numbers. We may call this kind of definition of a function a *total recursive definition*. In computer science and logic this kind of definition of a function on a recursive data type has been called a **definition by structural recursion**.

Convertibility $t \downarrow t'$ between terms t and t' is the equivalence relation generated by the defining equations for constants, the computation rule

$$(\lambda x.\, t)(u) :\equiv t[u/x],$$

and the rules which make it a *congruence* with respect to application and λ-abstraction:

- if $t \downarrow t'$ and $s \downarrow s'$ then $t(s) \downarrow t'(s')$, and
- if $t \downarrow t'$ then $(\lambda x.\, t) \downarrow (\lambda x.\, t')$.

The equality judgment $t \equiv u : A$ is then derived by the following single rule:

- if $t : A$, $u : A$, and $t \downarrow u$, then $t \equiv u : A$.

Judgmental equality is an equivalence relation.

A.1.1 Type universes

We postulate a hierarchy of **universes** denoted by primitive constants

$$\mathcal{U}_0, \quad \mathcal{U}_1, \quad \mathcal{U}_2, \quad \ldots$$

The first two rules for universes say that they form a cumulative hierarchy of types:

- $\mathcal{U}_m : \mathcal{U}_n$ for $m < n$,
- if $A : \mathcal{U}_m$ and $m \leq n$, then $A : \mathcal{U}_n$,

and the third expresses the idea that an object of a universe can serve as a type and stand to the right of a colon in judgments:

- if $\Gamma \vdash A : \mathcal{U}_n$, and x is a new variable,[1] then $\vdash (\Gamma, x : A)$ ctx.

In the body of the book, an equality judgment $A \equiv B : \mathcal{U}_n$ between types A and B is usually abbreviated to $A \equiv B$. This is an instance of typical ambiguity, as we can always switch to a larger universe, which however does not affect the validity of the judgment.

The following conversion rule allows us to replace a type by one equal to it in a typing judgment:

- if $a : A$ and $A \equiv B$ then $a : B$.

A.1.2 Dependent function types (Π-types)

We introduce a primitive constant c_Π, but write $c_\Pi(A, \lambda x.\, B)$ as $\prod_{(x:A)} B$. Judgments concerning such expressions and expressions of the form $\lambda x.\, b$ are introduced by the following rules:

- if $\Gamma \vdash A : \mathcal{U}_n$ and $\Gamma, x : A \vdash B : \mathcal{U}_n$, then $\Gamma \vdash \prod_{(x:A)} B : \mathcal{U}_n$
- if $\Gamma, x : A \vdash b : B$ then $\Gamma \vdash (\lambda x.\, b) : (\prod_{(x:A)} B)$
- if $\Gamma \vdash g : \prod_{(x:A)} B$ and $\Gamma \vdash t : A$ then $\Gamma \vdash g(t) : B[t/x]$

If x does not occur freely in B, we abbreviate $\prod_{(x:A)} B$ as the non-dependent function type $A \to B$ and derive the following rule:

- if $\Gamma \vdash g : A \to B$ and $\Gamma \vdash t : A$ then $\Gamma \vdash g(t) : B$

Using non-dependent function types and leaving implicit the context Γ, the rules above can be written in the following alternative style that we use in the rest of this section of the appendix:

- if $A : \mathcal{U}_n$ and $B : A \to \mathcal{U}_n$, then $\prod_{(x:A)} B(x) : \mathcal{U}_n$
- if $x : A \vdash b : B$ then $\lambda x.\, b : \prod_{(x:A)} B(x)$
- if $g : \prod_{(x:A)} B(x)$ and $t : A$ then $g(t) : B(t)$

[1] By "new" we mean that it does not appear in Γ or A.

A.1.3 Dependent pair types (Σ-types)

We introduce primitive constants c_Σ and c_{pair}. An expression of the form $c_\Sigma(A, \lambda a.\, B)$ is written as $\sum_{(a:A)} B$, and an expression of the form $c_{\mathsf{pair}}(a, b)$ is written as (a, b). We write $A \times B$ instead of $\sum_{(x:A)} B$ if x is not free in B.

Judgments concerning such expressions are introduced by the following rules:

- if $A : \mathcal{U}_n$ and $B : A \to \mathcal{U}_n$, then $\sum_{(x:A)} B(x) : \mathcal{U}_n$
- if, in addition, $a : A$ and $b : B(a)$, then $(a, b) : \sum_{(x:A)} B(x)$

If we have A and B as above, $C : (\sum_{(x:A)} B(x)) \to \mathcal{U}_m$, and

$$d : \prod_{(x:A)} \prod_{(y:B(x))} C((x, y))$$

we can introduce a defined constant

$$f : \prod_{(p:\sum_{(x:A)} B(x))} C(p)$$

with the defining equation

$$f((x, y)) :\equiv d(x, y).$$

Note that C, d, x, and y may contain extra implicit parameters $x_1, \ldots, x_n$ if they were obtained in some non-empty context; therefore, the fully explicit recursion schema is

$$f(x_1, \ldots, x_n, (x(x_1, \ldots, x_n), y(x_1, \ldots, x_n))) :\equiv d(x_1, \ldots, x_n, (x(x_1, \ldots, x_n), y(x_1, \ldots, x_n))).$$

A.1.4 Coproduct types

We introduce primitive constants c_+, c_{inl}, and c_{inr}. We write $A + B$ instead of $c_+(A, B)$, $\mathsf{inl}(a)$ instead of $c_{\mathsf{inl}}(a)$, and $\mathsf{inr}(a)$ instead of $c_{\mathsf{inr}}(a)$:

- if $A, B : \mathcal{U}_n$ then $A + B : \mathcal{U}_n$
- moreover, $\mathsf{inl} : A \to A + B$ and $\mathsf{inr} : B \to A + B$

If we have A and B as above, $C : A + B \to \mathcal{U}_m$, $d : \prod_{(x:A)} C(\mathsf{inl}(x))$, and $e : \prod_{(y:B)} C(\mathsf{inr}(y))$, then we can introduce a defined constant $f : \prod_{(z:A+B)} C(z)$ with the defining equations

$$f(\mathsf{inl}(x)) :\equiv d(x) \qquad \text{and} \qquad f(\mathsf{inr}(y)) :\equiv e(y).$$

A.1.5 The finite types

We introduce primitive constants $\star$, $\mathbf{0}$, $\mathbf{1}$, satisfying the following rules:

- $\mathbf{0} : \mathcal{U}_0$, $\mathbf{1} : \mathcal{U}_0$
- $\star : \mathbf{1}$

Given $C : \mathbf{0} \to \mathcal{U}_n$ we can introduce a defined constant $f : \prod_{(x:\mathbf{0})} C(x)$, with no defining equations.

Given $C : \mathbf{1} \to \mathcal{U}_n$ and $d : C(\star)$ we can introduce a defined constant $f : \prod_{(x:\mathbf{1})} C(x)$, with defining equation $f(\star) :\equiv d$.

A.1.6 Natural numbers

The type of natural numbers is obtained by introducing primitive constants $\mathbb{N}$, 0, and succ with the following rules:

- $\mathbb{N} : \mathcal{U}_0$,
- $0 : \mathbb{N}$,
- $\mathsf{succ} : \mathbb{N} \to \mathbb{N}$.

Furthermore, we can define functions by primitive recursion. If we have $C : \mathbb{N} \to \mathcal{U}_k$ we can introduce a defined constant $f : \prod_{(x:\mathbb{N})} C(x)$ whenever we have

$$\begin{aligned} &d : C(0) \\ &e : \textstyle\prod_{(x:\mathbb{N})}(C(x) \to C(\mathsf{succ}(x))) \end{aligned}$$

with the defining equations

$$f(0) :\equiv d \qquad \text{and} \qquad f(\mathsf{succ}(x)) :\equiv e(x, f(x)).$$

A.1.7 W-types

For W-types we introduce primitive constants c_{W} and c_{sup}. An expression of the form $c_{\mathsf{W}}(A, \lambda x.\, B)$ is written as $\mathsf{W}_{(x:A)}B$, and an expression of the form $c_{\mathsf{sup}}(x, u)$ is written as $\mathsf{sup}(x, u)$:

- if $A : \mathcal{U}_n$ and $B : A \to \mathcal{U}_n$, then $\mathsf{W}_{(x:A)}B(x) : \mathcal{U}_n$
- if moreover, $a : A$ and $u : B(a) \to \mathsf{W}_{(x:A)}B(x)$ then $\mathsf{sup}(a, u) : \mathsf{W}_{(x:A)}B(x)$.

Here also we can define functions by total recursion. If we have A and B as above and $C : (\mathsf{W}_{(x:A)}B(x)) \to \mathcal{U}_m$, then we can introduce a defined constant $f : \prod_{(z:\mathsf{W}_{(x:A)}B(x))} C(z)$ whenever we have

$$d : \textstyle\prod_{(a:A)}\prod_{(u:B(a)\to \mathsf{W}_{(x:A)}B(x))}((\prod_{(y:B(a))} C(u(y))) \to C(\mathsf{sup}(a, u)))$$

with the defining equation

$$f(\mathsf{sup}(a, u)) :\equiv d(a, u, f \circ u).$$

A.1.8 Identity types

We introduce primitive constants $c_=$ and c_{refl}. We write $a =_A b$ for $c_=(A, a, b)$ and refl_a for $c_{\mathsf{refl}}(A, a)$, when $a : A$ is understood:

- If $A : \mathcal{U}_n$, $a : A$, and $b : A$ then $a =_A b : \mathcal{U}_n$.
- If $a : A$ then $\mathsf{refl}_a : a =_A a$.

Given $a : A$, if $y : A, z : a =_A y \vdash C : \mathcal{U}_m$ and $\vdash d : C[a, \mathsf{refl}_a / y, z]$ then we can introduce a defined constant

$$f : \textstyle\prod_{(y:A)}\prod_{(z:a=_A y)} C$$

with defining equation

$$f(a, \mathsf{refl}_a) :\equiv d.$$

A.2 The second presentation

In this section, there are three kinds of judgments

$$\Gamma \ \mathsf{ctx} \qquad\qquad \Gamma \vdash a : A \qquad\qquad \Gamma \vdash a \equiv a' : A$$

which we specify by providing inference rules for deriving them. A typical **inference rule** has the form

$$\frac{\mathcal{J}_1 \qquad \cdots \qquad \mathcal{J}_k}{\mathcal{J}}\ \textsc{Name}$$

It says that we may derive the **conclusion** $\mathcal{J}$, provided that we have already derived the **hypotheses** $\mathcal{J}_1, \ldots, \mathcal{J}_k$. (Note that, being judgments rather than types, these are not hypotheses *internal* to the type theory in the sense of §1.1; they are instead hypotheses in the deductive system, i.e. the metatheory.) On the right we write the NAME of the rule, and there may be extra side conditions that need to be checked before the rule is applicable.

A **derivation** of a judgment is a tree constructed from such inference rules, with the judgment at the root of the tree. For example, with the rules given below, the following is a derivation of $\cdot \vdash \lambda x.\, x : \mathbf{1} \to \mathbf{1}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\cdot\ \mathsf{ctx}}\ \textsc{ctx-emp}}{\vdash \mathbf{1} : \mathcal{U}_0}\ \mathbf{1}\textsc{-form}}{x{:}\mathbf{1}\ \mathsf{ctx}}\ \textsc{ctx-ext}}{x{:}\mathbf{1} \vdash x : \mathbf{1}}\ \textsc{Vble}}{\cdot \vdash \lambda x.\, x : \mathbf{1} \to \mathbf{1}}\ \Pi\textsc{-intro}$$

A.2.1 Contexts

A context is a list

$$x_1{:}A_1, x_2{:}A_2, \ldots, x_n{:}A_n$$

which indicates that the distinct variables $x_1, \ldots, x_n$ are assumed to have types $A_1, \ldots, A_n$, respectively. The list may be empty. We abbreviate contexts with the letters Γ and Δ, and we may juxtapose them to form larger contexts.

The judgment $\Gamma\ \mathsf{ctx}$ formally expresses the fact that Γ is a well-formed context, and is governed by the rules of inference

$$\frac{}{\cdot\ \mathsf{ctx}}\ \textsc{ctx-emp} \qquad\qquad \frac{x_1{:}A_1, \ldots, x_{n-1}{:}A_{n-1} \vdash A_n : \mathcal{U}_i}{(x_1{:}A_1, \ldots, x_n{:}A_n)\ \mathsf{ctx}}\ \textsc{ctx-ext}$$

with a side condition for the second rule: the variable x_n must be distinct from the variables $x_1, \ldots, x_{n-1}$. Note that the hypothesis and conclusion of ctx-EXT are judgments of different forms: the hypothesis says that in the context of variables $x_1, \ldots, x_{n-1}$, the expression A_n has type $\mathcal{U}_i$; while the conclusion says that the extended context $(x_1{:}A_1, \ldots, x_n{:}A_n)$ is well-formed. (It is a meta-theoretic property of the system that if $x_1{:}A_1, \ldots, x_n{:}A_n \vdash b : B$ is derivable, then the context $(x_1{:}A_1, \ldots, x_n{:}A_n)$ must be well-formed; thus ctx-EXT does not need to hypothesize well-formedness of the context to the left of x_n.)

A.2.2 Structural rules

The fact that the context holds assumptions is expressed by the rule which says that we may derive those typing judgments which are listed in the context:

$$\frac{(x_1{:}A_1, \ldots, x_n{:}A_n)\ \mathsf{ctx}}{x_1{:}A_1, \ldots, x_n{:}A_n \vdash x_i : A_i}\ \mathsf{Vble}$$

As with ctx-EXT, the hypothesis and conclusion of the rule Vble are judgments of different forms, only now they are reversed: we start with a well-formed context and derive a typing judgment.

The following important principles, called **substitution** and **weakening**, need not be explicitly assumed. Rather, it is possible to show, by induction on the structure of all possible derivations, that whenever the hypotheses of these rules are derivable, their conclusion is also derivable.[2] For the typing judgments these principles are manifested as

$$\frac{\Gamma \vdash a : A \qquad \Gamma, x{:}A, \Delta \vdash b : B}{\Gamma, \Delta[a/x] \vdash b[a/x] : B[a/x]}\ \mathsf{Subst}_1 \qquad \frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma, \Delta \vdash b : B}{\Gamma, x{:}A, \Delta \vdash b : B}\ \mathsf{Wkg}_1$$

and for judgmental equalities they become

$$\frac{\Gamma \vdash a : A \qquad \Gamma, x{:}A, \Delta \vdash b \equiv c : B}{\Gamma, \Delta[a/x] \vdash b[a/x] \equiv c[a/x] : B[a/x]}\ \mathsf{Subst}_2 \qquad \frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma, \Delta \vdash b \equiv c : B}{\Gamma, x{:}A, \Delta \vdash b \equiv c : B}\ \mathsf{Wkg}_2$$

In addition to the judgmental equality rules given for each type former, we also assume that judgmental equality is an equivalence relation respected by typing.

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash a \equiv a : A} \qquad \frac{\Gamma \vdash a \equiv b : A}{\Gamma \vdash b \equiv a : A} \qquad \frac{\Gamma \vdash a \equiv b : A \qquad \Gamma \vdash b \equiv c : A}{\Gamma \vdash a \equiv c : A}$$

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash A \equiv B : \mathcal{U}_i}{\Gamma \vdash a : B} \qquad \frac{\Gamma \vdash a \equiv b : A \qquad \Gamma \vdash A \equiv B : \mathcal{U}_i}{\Gamma \vdash a \equiv b : B}$$

Additionally, for all the type formers below, we assume rules stating that each constructor preserves definitional equality in each of its arguments; for instance, along with the Π-INTRO rule, we assume the rule

$$\frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma, x{:}A \vdash B : \mathcal{U}_i \qquad \Gamma, x{:}A \vdash b \equiv b' : B}{\Gamma \vdash \lambda x.\, b \equiv \lambda x.\, b' : \prod_{(x:A)} B}\ \Pi\text{-INTRO-EQ}$$

However, we omit these rules for brevity.

A.2.3 Type universes

We postulate an infinite hierarchy of type universes

$$\mathcal{U}_0, \quad \mathcal{U}_1, \quad \mathcal{U}_2, \quad \ldots$$

Each universe is contained in the next, and any type in $\mathcal{U}_i$ is also in $\mathcal{U}_{i+1}$:

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathcal{U}_i : \mathcal{U}_{i+1}}\ \mathcal{U}\text{-INTRO} \qquad \frac{\Gamma \vdash A : \mathcal{U}_i}{\Gamma \vdash A : \mathcal{U}_{i+1}}\ \mathcal{U}\text{-CUMUL}$$

[2] Such rules are called **admissible**.

We shall set up the rules of type theory in such a way that $\Gamma \vdash a : A$ implies $\Gamma \vdash A : \mathcal{U}_i$ for some i. In other words, if A plays the role of a type then it is in some universe. Another property of our type system is that $\Gamma \vdash a \equiv b : A$ implies $\Gamma \vdash a : A$ and $\Gamma \vdash b : A$.

A.2.4 Dependent function types (Π-types)

In §1.2, we introduced non-dependent functions $A \to B$ in order to define a family of types as a function $\lambda(x:A).\, B : A \to \mathcal{U}_i$, which then gives rise to a type of dependent functions $\prod_{(x:A)} B$. But with explicit contexts we may replace $\lambda(x:A).\, B : A \to \mathcal{U}_i$ with the judgment

$$x{:}A \vdash B : \mathcal{U}_i.$$

Consequently, we may define dependent functions directly, without reference to non-dependent ones. This way we follow the general principle that each type former, with its constants and rules, should be introduced independently of all other type formers. In fact, henceforth each type former is introduced systematically by:

- a **formation rule**, stating when the type former can be applied;
- some **introduction rules**, stating how to inhabit the type;
- **elimination rules**, or an induction principle, stating how to use an element of the type;
- **computation rules**, which are judgmental equalities explaining what happens when elimination rules are applied to results of introduction rules;
- optional **uniqueness principles**, which are judgmental equalities explaining how every element of the type is uniquely determined by the results of elimination rules applied to it.

(See also Remark 1.5.1.)

For the dependent function type these rules are:

$$\frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma, x{:}A \vdash B : \mathcal{U}_i}{\Gamma \vdash \prod_{(x:A)} B : \mathcal{U}_i}\ \Pi\text{-FORM} \qquad \frac{\Gamma, x{:}A \vdash b : B}{\Gamma \vdash \lambda(x:A).\, b : \prod_{(x:A)} B}\ \Pi\text{-INTRO}$$

$$\frac{\Gamma \vdash f : \prod_{(x:A)} B \qquad \Gamma \vdash a : A}{\Gamma \vdash f(a) : B[a/x]}\ \Pi\text{-ELIM} \qquad \frac{\Gamma, x{:}A \vdash b : B \qquad \Gamma \vdash a : A}{\Gamma \vdash (\lambda(x:A).\, b)(a) \equiv b[a/x] : B[a/x]}\ \Pi\text{-COMP}$$

$$\frac{\Gamma \vdash f : \prod_{(x:A)} B}{\Gamma \vdash f \equiv (\lambda x.\, f(x)) : \prod_{(x:A)} B}\ \Pi\text{-UNIQ}$$

The expression $\lambda(x:A).\, b$ binds free occurrences of x in b, as does $\prod_{(x:A)} B$ for B.

When x does not occur freely in B so that B does not depend on A, we obtain as a special case the ordinary function type $A \to B :\equiv \prod_{(x:A)} B$. We take this as the *definition* of $\to$.

We may abbreviate an expression $\lambda(x:A).\, b$ as $\lambda x.\, b$, with the understanding that the omitted type A should be filled in appropriately before type-checking.

A.2.5 Dependent pair types (Σ-types)

In §1.6, we needed $\to$ and $\prod$ types in order to define the introduction and elimination rules for $\sum$; as with $\prod$, contexts allow us to state the rules for $\sum$ independently. Recall that the elimination rule for a positive type such as Σ is called *induction* and denoted by ind.

$$\frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma, x{:}A \vdash B : \mathcal{U}_i}{\Gamma \vdash \sum_{(x:A)} B : \mathcal{U}_i}\ \Sigma\text{-FORM}$$

$$\frac{\Gamma, x{:}A \vdash B : \mathcal{U}_i \qquad \Gamma \vdash a : A \qquad \Gamma \vdash b : B[a/x]}{\Gamma \vdash (a, b) : \sum_{(x:A)} B}\ \Sigma\text{-INTRO}$$

$$\frac{\Gamma, z{:}\sum_{(x:A)} B \vdash C : \mathcal{U}_i \qquad \Gamma, x{:}A, y{:}B \vdash g : C[(x,y)/z] \qquad \Gamma \vdash p : \sum_{(x:A)} B}{\Gamma \vdash \mathsf{ind}_{\sum_{(x:A)} B}(z.C, x.y.g, p) : C[p/z]}\ \Sigma\text{-ELIM}$$

$$\frac{\begin{array}{c}\Gamma, z{:}\sum_{(x:A)} B \vdash C : \mathcal{U}_i \qquad \Gamma, x{:}A, y{:}B \vdash g : C[(x,y)/z] \\ \Gamma \vdash a : A \qquad \Gamma \vdash b : B[a/x]\end{array}}{\Gamma \vdash \mathsf{ind}_{\sum_{(x:A)} B}(z.C, x.y.g, (a,b)) \equiv g[a, b/x, y] : C[(a,b)/z]}\ \Sigma\text{-COMP}$$

The expression $\sum_{(x:A)} B$ binds free occurrences of x in B. Furthermore, because $\mathsf{ind}_{\sum_{(x:A)} B}$ has some arguments with free variables beyond those in Γ, we bind (following the variable names above) z in C, and x and y in g. These bindings are written as $z.C$ and $x.y.g$, to indicate the names of the bound variables. In particular, we treat $\mathsf{ind}_{\sum_{(x:A)} B}$ as a primitive, two of whose arguments contain binders; this is superficially similar to, but different from, $\mathsf{ind}_{\sum_{(x:A)} B}$ being a function that takes functions as arguments.

When B does not contain free occurrences of x, we obtain as a special case the cartesian product $A \times B :\equiv \sum_{(x:A)} B$. We take this as the *definition* of the cartesian product.

Notice that we don't postulate a judgmental uniqueness principle for Σ-types, even though we could have; see Corollary 2.7.3 for a proof of the corresponding propositional uniqueness principle.

A.2.6 Coproduct types

$$\frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma \vdash B : \mathcal{U}_i}{\Gamma \vdash A + B : \mathcal{U}_i}\ +\text{-FORM}$$

$$\frac{\begin{array}{c}\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma \vdash B : \mathcal{U}_i \\ \Gamma \vdash a : A\end{array}}{\Gamma \vdash \mathsf{inl}(a) : A + B}\ +\text{-INTRO}_1 \qquad \frac{\begin{array}{c}\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma \vdash B : \mathcal{U}_i \\ \Gamma \vdash b : B\end{array}}{\Gamma \vdash \mathsf{inr}(b) : A + B}\ +\text{-INTRO}_2$$

$$\frac{\begin{array}{c}\Gamma, z{:}(A+B) \vdash C : \mathcal{U}_i \\ \Gamma, x{:}A \vdash c : C[\mathsf{inl}(x)/z] \qquad \Gamma, y{:}B \vdash d : C[\mathsf{inr}(y)/z] \\ \Gamma \vdash e : A + B\end{array}}{\Gamma \vdash \mathsf{ind}_{A+B}(z.C, x.c, y.d, e) : C[e/z]}\ +\text{-ELIM}$$

$$\frac{\begin{array}{c}\Gamma, z{:}(A+B) \vdash C : \mathcal{U}_i \qquad \Gamma, x{:}A \vdash c : C[\mathsf{inl}(x)/z] \qquad \Gamma, y{:}B \vdash d : C[\mathsf{inr}(y)/z] \\ \Gamma \vdash a : A\end{array}}{\Gamma \vdash \mathsf{ind}_{A+B}(z.C, x.c, y.d, \mathsf{inl}(a)) \equiv c[a/x] : C[\mathsf{inl}(a)/z]}\ +\text{-COMP}_1$$

$$\frac{\begin{array}{c}\Gamma, z{:}(A+B) \vdash C : \mathcal{U}_i \qquad \Gamma, x{:}A \vdash c : C[\mathsf{inl}(x)/z] \qquad \Gamma, y{:}B \vdash d : C[\mathsf{inr}(y)/z] \\ \Gamma \vdash b : B\end{array}}{\Gamma \vdash \mathsf{ind}_{A+B}(z.C, x.c, y.d, \mathsf{inr}(b)) \equiv d[b/y] : C[\mathsf{inr}(b)/z]}\ +\text{-COMP}_2$$

In ind_{A+B}, z is bound in C, x is bound in c, and y is bound in d.

A.2.7 The empty type 0

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathbf{0} : \mathcal{U}_i}\ \mathbf{0}\text{-FORM} \qquad \frac{\Gamma, x{:}\mathbf{0} \vdash C : \mathcal{U}_i \qquad \Gamma \vdash a : \mathbf{0}}{\Gamma \vdash \mathsf{ind}_{\mathbf{0}}(x.C, a) : C[a/x]}\ \mathbf{0}\text{-ELIM}$$

In $\mathsf{ind}_{\mathbf{0}}$, x is bound in C. The empty type has no introduction rule and no computation rule.

A.2.8 The unit type 1

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathbf{1} : \mathcal{U}_i}\ \mathbf{1}\text{-FORM} \qquad \frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \star : \mathbf{1}}\ \mathbf{1}\text{-INTRO} \qquad \frac{\Gamma, x{:}\mathbf{1} \vdash C : \mathcal{U}_i \qquad \Gamma \vdash c : C[\star/x] \qquad \Gamma \vdash a : \mathbf{1}}{\Gamma \vdash \mathsf{ind}_{\mathbf{1}}(x.C, c, a) : C[a/x]}\ \mathbf{1}\text{-ELIM}$$

$$\frac{\Gamma, x{:}\mathbf{1} \vdash C : \mathcal{U}_i \qquad \Gamma \vdash c : C[\star/x]}{\Gamma \vdash \mathsf{ind}_{\mathbf{1}}(x.C, c, \star) \equiv c : C[\star/x]}\ \mathbf{1}\text{-COMP}$$

In $\mathsf{ind}_{\mathbf{1}}$ the variable x is bound in C.

Notice that we do not postulate a judgmental uniqueness principle for the unit type; see §1.5 for a proof of the corresponding propositional uniqueness statement.

A.2.9 The natural number type

We give the rules for natural numbers, following §1.9.

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathbb{N} : \mathcal{U}_i}\ \mathbb{N}\text{-FORM} \qquad \frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash 0 : \mathbb{N}}\ \mathbb{N}\text{-INTRO}_1 \qquad \frac{\Gamma \vdash n : \mathbb{N}}{\Gamma \vdash \mathsf{succ}(n) : \mathbb{N}}\ \mathbb{N}\text{-INTRO}_2$$

$$\frac{\Gamma, x{:}\mathbb{N} \vdash C : \mathcal{U}_i \qquad \Gamma \vdash c_0 : C[0/x] \qquad \Gamma, x{:}\mathbb{N}, y{:}C \vdash c_s : C[\mathsf{succ}(x)/x] \qquad \Gamma \vdash n : \mathbb{N}}{\Gamma \vdash \mathsf{ind}_{\mathbb{N}}(x.C, c_0, x.y.c_s, n) : C[n/x]}\ \mathbb{N}\text{-ELIM}$$

$$\frac{\Gamma, x{:}\mathbb{N} \vdash C : \mathcal{U}_i \qquad \Gamma \vdash c_0 : C[0/x] \qquad \Gamma, x{:}\mathbb{N}, y{:}C \vdash c_s : C[\mathsf{succ}(x)/x]}{\Gamma \vdash \mathsf{ind}_{\mathbb{N}}(x.C, c_0, x.y.c_s, 0) \equiv c_0 : C[0/x]}\ \mathbb{N}\text{-COMP}_1$$

$$\frac{\Gamma, x{:}\mathbb{N} \vdash C : \mathcal{U}_i \qquad \Gamma \vdash c_0 : C[0/x] \qquad \Gamma, x{:}\mathbb{N}, y{:}C \vdash c_s : C[\mathsf{succ}(x)/x] \qquad \Gamma \vdash n : \mathbb{N}}{\begin{array}{l}\Gamma \vdash\ \mathsf{ind}_{\mathbb{N}}(x.C, c_0, x.y.c_s, \mathsf{succ}(n)) \\ \qquad\qquad \equiv c_s[n, \mathsf{ind}_{\mathbb{N}}(x.C, c_0, x.y.c_s, n)/x, y] : C[\mathsf{succ}(n)/x]\end{array}}\ \mathbb{N}\text{-COMP}_2$$

In $\mathsf{ind}_{\mathbb{N}}$, x is bound in C, and x and y are bound in c_s.

Other inductively defined types follow the same general scheme.

A.2.10 Identity types

The presentation here corresponds to the (unbased) path induction principle for identity types in §1.12.

$$\frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma \vdash a : A \qquad \Gamma \vdash b : A}{\Gamma \vdash a =_A b : \mathcal{U}_i} =\text{-FORM} \qquad \frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma \vdash a : A}{\Gamma \vdash \mathsf{refl}_a : a =_A a} =\text{-INTRO}$$

$$\frac{\begin{array}{c}\Gamma, x{:}A, y{:}A, p{:}x =_A y \vdash C : \mathcal{U}_i \\ \Gamma, z{:}A \vdash c : C[z, z, \mathsf{refl}_z / x, y, p] \qquad \Gamma \vdash a : A \qquad \Gamma \vdash b : A \qquad \Gamma \vdash p' : a =_A b\end{array}}{\Gamma \vdash \mathsf{ind}_{=_A}(x.y.p.C, z.c, a, b, p') : C[a, b, p'/x, y, p]} =\text{-ELIM}$$

$$\frac{\Gamma, x{:}A, y{:}A, p{:}x =_A y \vdash C : \mathcal{U}_i \qquad \Gamma, z{:}A \vdash c : C[z, z, \mathsf{refl}_z / x, y, p] \qquad \Gamma \vdash a : A}{\Gamma \vdash \mathsf{ind}_{=_A}(x.y.p.C, z.c, a, a, \mathsf{refl}_a) \equiv c[a/z] : C[a, a, \mathsf{refl}_a / x, y, p]} =\text{-COMP}$$

In $\mathsf{ind}_{=_A}$, x, y, and p are bound in C, and z is bound in c.

A.2.11 Definitions

Although the rules we listed so far allows us to construct everything we need directly, we would still like to be able to use named constants, such as isequiv, as a matter of convenience. Informally, we can think of these constants simply as abbreviations, but the situation is a bit subtler in the formalization.

For example, consider function composition, which takes $f : A \to B$ and $g : B \to C$ to $g \circ f : A \to C$. Somewhat unexpectedly, to make this work formally, $\circ$ must take as arguments not only f and g, but also their types A, B, C:

$$\circ :\equiv \lambda(A : \mathcal{U}_i).\, \lambda(B : \mathcal{U}_i).\, \lambda(C : \mathcal{U}_i).\, \lambda(g : B \to C).\, \lambda(f : A \to B).\, \lambda(x : A).\, g(f(x)).$$

From a practical perspective, we do not want to annotate each application of $\circ$ with A, B and C, as they are usually quite easily guessed from surrounding information. We would like to simply write $g \circ f$. Then, strictly speaking, $g \circ f$ is not an abbreviation for $\lambda(x : A).\, g(f(x))$, because it involves additional **implicit arguments** which we want to suppress.

Inference of implicit arguments, typical ambiguity (§1.3), ensuring that symbols are only defined once, etc., are collectively called **elaboration**. Elaboration must take place prior to checking a derivation, and is thus not usually presented as part of the core type theory. However, it is essentially impossible to use any implementation of type theory which does not perform elaboration; see [Coq12, Nor07] for further discussion.

A.3 Homotopy type theory

In this section we state the additional axioms of homotopy type theory which distinguish it from standard Martin-Löf type theory: function extensionality, the univalence axiom, and higher inductive types. We state them in the style of the second presentation Appendix A.2, although the first presentation Appendix A.1 could be used just as well.

A.3.1 Function extensionality and univalence

There are two basic ways of introducing axioms which do not introduce new syntax or judgmental equalities (function extensionality and univalence are of this form): either add a primitive constant to inhabit the axiom, or prove all theorems which depend on the axiom by hypothesizing a variable that inhabits the axiom, cf. §1.1. While these are essentially equivalent, we opt for

the former approach because we feel that the axioms of homotopy type theory are an essential part of the core theory.

Axiom 2.9.3 is formalized by introduction of a constant funext which asserts that happly is an equivalence:

$$\frac{\Gamma \vdash f : \prod_{(x:A)} B \qquad \Gamma \vdash g : \prod_{(x:A)} B}{\Gamma \vdash \mathsf{funext}(f,g) : \mathsf{isequiv}(\mathsf{happly}_{f,g})}\ \Pi\text{-EXT}$$

The definitions of happly and isequiv can be found in (2.9.2) and §4.5, respectively.

Axiom 2.10.3 is formalized in a similar fashion, too:

$$\frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma \vdash B : \mathcal{U}_i}{\Gamma \vdash \mathsf{univalence}(A,B) : \mathsf{isequiv}(\mathsf{idtoeqv}_{A,B})}\ \mathcal{U}_i\text{-UNIV}$$

The definition of idtoeqv can be found in (2.10.2).

A.3.2 The circle

Here we give an example of a basic higher inductive type; others follow the same general scheme, albeit with elaborations.

Note that the rules below do not precisely follow the pattern of the ordinary inductive types in Appendix A.2: the rules refer to the notions of transport and functoriality of maps (§2.2), and the second computation rule is a propositional, not judgmental, equality. These differences are discussed in §6.2.

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathbb{S}^1 : \mathcal{U}_i}\ \mathbb{S}^1\text{-FORM} \qquad \frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathsf{base} : \mathbb{S}^1}\ \mathbb{S}^1\text{-INTRO}_1 \qquad \frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathsf{loop} : \mathsf{base} =_{\mathbb{S}^1} \mathsf{base}}\ \mathbb{S}^1\text{-INTRO}_2$$

$$\frac{\Gamma, x{:}\mathbb{S}^1 \vdash C : \mathcal{U}_i \qquad \Gamma \vdash b : C[\mathsf{base}/x] \qquad \Gamma \vdash \ell : b =^C_{\mathsf{loop}} b \qquad \Gamma \vdash p : \mathbb{S}^1}{\Gamma \vdash \mathsf{ind}_{\mathbb{S}^1}(x.C, b, \ell, p) : C[p/x]}\ \mathbb{S}^1\text{-ELIM}$$

$$\frac{\Gamma, x{:}\mathbb{S}^1 \vdash C : \mathcal{U}_i \qquad \Gamma \vdash b : C[\mathsf{base}/x] \qquad \Gamma \vdash \ell : b =^C_{\mathsf{loop}} b}{\Gamma \vdash \mathsf{ind}_{\mathbb{S}^1}(x.C, b, \ell, \mathsf{base}) \equiv b : C[\mathsf{base}/x]}\ \mathbb{S}^1\text{-COMP}_1$$

$$\frac{\Gamma, x{:}\mathbb{S}^1 \vdash C : \mathcal{U}_i \qquad \Gamma \vdash b : C[\mathsf{base}/x] \qquad \Gamma \vdash \ell : b =^C_{\mathsf{loop}} b}{\Gamma \vdash \mathbb{S}^1\text{-}\mathsf{loopcomp} : \mathsf{apd}_{(\lambda y.\, \mathsf{ind}_{\mathbb{S}^1}(x.C,b,\ell,y))}(\mathsf{loop}) = \ell}\ \mathbb{S}^1\text{-COMP}_2$$

In $\mathsf{ind}_{\mathbb{S}^1}$, x is bound in C. The notation $b =^C_{\mathsf{loop}} b$ for dependent paths was introduced in §6.2.

A.4 Basic metatheory

This section discusses the meta-theoretic properties of the type theory presented in Appendix A.1, and similar results hold for Appendix A.2. Figuring out which of these still hold when we add the features from Appendix A.3 quickly leads to open questions, as discussed at the end of this section.

Recall that Appendix A.1 defines the terms of type theory as an extension of the untyped λ-calculus. The λ-calculus has its own notion of computation, namely the computation rule:

$$(\lambda x.\, t)(u) :\equiv t[u/x].$$

This rule, together with the defining equations for the defined constants form *rewriting rules* that determine reduction steps for a rewriting system. These steps yield a notion of computation in the sense that each rule has a natural direction: one simplifies $(\lambda x.\, t)(u)$ by evaluating the function at its argument.

Moreover, this system is *confluent*, that is, if a simplifies in some number of steps to both a' and a'', there is some b to which both a' and a'' eventually simplify. Thus we can define $t \downarrow u$ to mean that t and u simplify to the same term.

(The situation is similar in Appendix A.2: Although there we presented the computation rules as undirected equalities $\equiv$, we can give an operational semantics by saying that the application of an eliminator to an introductory form simplifies to its equal, not the other way around.)

Using standard techniques from type theory, it is possible to show that the system in Appendix A.1 has the following properties:

Theorem A.4.1. *If $A : \mathcal{U}$ and $A \downarrow A'$ then $A' : \mathcal{U}$. If $t : A$ and $t \downarrow t'$ then $t' : A$.*

We say that a term is **normalizable** (respectively, **strongly normalizable**) if some (respectively, every), sequence of rewriting steps from the term terminates.

Theorem A.4.2. *If $A : \mathcal{U}$ then A is strongly normalizable. If $t : A$ then A and t are strongly normalizable.*

We say that a term is in **normal form** if it cannot be further simplified, and that a term is **closed** if no variable occurs freely in it. A closed normal type has to be a primitive type, i.e., of the form $c(\vec{v})$ for some primitive constant c (where the list $\vec{v}$ of closed normal terms may be omitted if empty, for instance, as with $\mathbb{N}$). In fact, we can explicitly describe all normal forms:

Lemma A.4.3. *The terms in normal form can be described by the following syntax:*

$$v ::= k \mid \lambda x.\, v \mid c(\vec{v}) \mid f(\vec{v}),$$
$$k ::= x \mid k(v) \mid f(\vec{v})(k),$$

where $f(\vec{v})$ represents a partial application of the defined function f. In particular, a type in normal form is of the form k or $c(\vec{v})$.

Theorem A.4.4. *If A is in normal form then the judgment $A : \mathcal{U}$ is decidable. If $A : \mathcal{U}$ and t is in normal form then the judgment $t : A$ is decidable.*

Logical consistency (of the system in Appendix A.1) follows immediately: if we had $a : \mathbf{0}$ in the empty context, then by Theorems A.4.1 and A.4.2, a simplifies to a normal term $a' : \mathbf{0}$. But by Lemma A.4.3 no such term exists.

Corollary A.4.5. *The system in Appendix A.1 is logically consistent.*

Similarly, we have the *canonicity* property that if $a : \mathbb{N}$ in the empty context, then a simplifies to a normal term $\mathrm{succ}^k(0)$ for some numeral k.

Corollary A.4.6. *The system in Appendix A.1 has the canonicity property.*

Finally, if a, A are in normal form, it is *decidable* whether $a : A$; in other words, because typechecking amounts to verifying the correctness of a proof, this means we can always "recognize a correct proof when we see one".

Corollary A.4.7. *The property of being a proof in the system in Appendix A.1 is decidable.*

The above results do not apply to the extended system of homotopy type theory (i.e., the above system extended by Appendix A.3), since occurrences of the univalence axiom and constructors of higher inductive types never simplify, breaking Lemma A.4.3. It is an open question whether one can simplify applications of these constants in order to restore canonicity. We also do not have a schema describing all permissible higher inductive types, nor are we certain how to correctly formulate their rules (e.g., whether the computation rules on higher constructors should be judgmental equalities).

The consistency of Martin-Löf type theory extended with univalence and higher inductive types could be shown by inventing an appropriate normalization procedure, but currently the only proofs that these systems are consistent are via semantic models—for univalence, a model in Kan complexes due to Voevodsky [KLV12], and for higher inductive types, a model due to Lumsdaine and Shulman [LS17].

Other metatheoretic issues, and a summary of our current results, are discussed in greater length in the "Constructivity" and "Open problems" sections of the introduction to this book.

Notes

The system of rules with introduction (primitive constants) and elimination and computation rules (defined constant) is inspired by Gentzen natural deduction. The possibility of strengthening the elimination rule for existential quantification was indicated in [How80]. The strengthening of the axioms for disjunction appears in [ML98], and for absurdity elimination and identity type in [ML75]. The W-types were introduced in [ML82]. They generalize a notion of trees introduced by [Tai68].

The generalized form of primitive recursion for natural numbers and ordinals appear in [Hil26]. This motivated Gödel's system T, [Göd58], which was analyzed by [Tai67], who used, following [Göd58], the terminology "definitional equality" for conversion: two terms are *judgmentally equal* if they reduce to a common term by means of a sequence of applications of the reduction rules. This terminology was also used by de Bruijn [dB73] in his presentation of *AUTOMATH*.

Our second presentation comprises fairly standard presentation of intensional Martin-Löf type theory, with some additional features needed in homotopy type theory. Compared to a reference presentation of [Hof97], the type theory of this book has a few non-critical differences:

- universes à la Russell, in the sense of [ML84]; and
- judgmental η and function extensionality for Π types;

and a few features essential for homotopy type theory:

- the univalence axiom; and
- higher inductive types.

As a matter of convenience, the book primarily defines functions by induction using definition by *pattern matching*. It is possible to formalize the notion of pattern matching, as done in Appendix A.1. However, the standard type-theoretic presentation, adopted in Appendix A.2, is to introduce a single *dependent eliminator* for each type former, from which functions out of that type must be defined. This approach is easier to formalize both syntactically and semantically,

as it amounts to the universal property of the type former. The two approaches are equivalent; see §1.10 for a longer discussion.

Bibliography

[AB04] Steven Awodey and Andrej Bauer. Propositions as [types]. *Journal of Logic and Computation*, 14(4):447–471, 2004. (Cited on pages 117, 203, and 234.)

[Acz78] Peter Aczel. The type theoretic interpretation of constructive set theory. In A. MacIntyre, L. Pacholski, and J. Paris, editors, *Logic Colloquium '77*, volume 96 of *Studies in Logic and the Foundations of Mathematics*, pages 55–66. North-Holland, Amsterdam, 1978. (Cited on pages 359 and 360.)

[AG02] Peter Aczel and Nicola Gambino. Collection principles in dependent type theory. In Paul Callaghan, Zhaohui Luo, James McKinna, and Robert Pollack, editors, *Types for Proofs and Programs, International Workshop, TYPES 2000, Durham, UK, December 8-12, 2000, Selected Papers*, volume 2277 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2002. (Cited on page 117.)

[AGS12] Steve Awodey, Nicola Gambino, and Kristina Sojakova. Inductive types in homotopy type theory. In *Proceedings of the 2012 27th Annual IEEE/ACM Symposium on Logic in Computer Science*, pages 95–104. IEEE Computer Society, 2012, arXiv:1201.3898. (Cited on page 163.)

[AKL13] Jeremy Avigad, Krzysztof Kapulkin, and Peter LeFanu Lumsdaine. Homotopy limits in Coq, 2013. arXiv:1304.0680. (Cited on page 97.)

[AKS13] Benedikt Ahrens, Krzysztof Kapulkin, and Michael Shulman. Univalent categories and the Rezk completion, 2013. arXiv:1303.0584. (Cited on page 331.)

[Alt99] Thorsten Altenkirch. Extensional equality in intensional type theory. In *14th Annual IEEE Symposium on Logic in Computer Science, Trento, Italy, July 2–5, 1999*, pages 412–420, 1999. (Cited on page 204.)

[AMS07] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra. Observational equality, now! In Aaron Stump and Hongwei Xi, editors, *Proceedings of the ACM Workshop Programming Languages meets Program Verification, PLPV 2007, Freiburg, Germany, October 5, 2007*, 2007. (Cited on page 204.)

[Ang13] Carlo Angiuli. The $(\infty, 1)$-accidentopos model of unintentional type theory. *Sigbovik '13*, April 1 2013. (No citations.)

[AW09] Steve Awodey and Michael A. Warren. Homotopy theoretic models of identity types. *Mathematical Proceedings of the Cambridge Philosophical Society*, 146:45–55, 2009. (Cited on pages 3 and 96.)

[Bau13] Andrej Bauer. Five stages of accepting constructive mathematics, 2013. `http://video.ias.edu/members/1213/0318-AndrejBauer`. (No citations.)

[BCH13] Bruno Barras, Thierry Coquand, and Simon Huber. A generalization of Takeuti-Gandy interpretation. `http://uf-ias-2012.wikispaces.com/file/view/semi.pdf`, 2013. (Cited on page 10.)

[Bee85] Michael Beeson. *Foundations of Constructive Mathematics*. Springer, 1985. (Cited on page 51.)

[Ber09] Julia E. Bergner. A survey of $(\infty,1)$-categories. In John C. Baez and J. Peter May, editors, *Towards Higher Categories*, volume 152 of *The IMA Volumes in Mathematics and its Applications*, pages 69–83. Springer, 2009, arXiv:math.CT/0610239. (Cited on page 331.)

[Bis67] Erret Bishop. *Foundations of constructive analysis*. McGraw-Hill Book Co., New York, 1967. (Cited on pages 360 and 391.)

[BIS02] Douglas Bridges, Hajime Ishihara, and Peter Schuster. Compactness and continuity, constructively revisited. In Julian C. Bradfield, editor, *Computer Science Logic, 16th International Workshop, CSL 2002, 11th Annual Conference of the EACSL, Edinburgh, Scotland, UK, September 22-25, 2002, Proceedings*, volume 2471 of *Lecture Notes in Computer Science*, pages 89–102. Springer, 2002. (Cited on page 408.)

[Bla79] Georges Blanc. Équivalence naturelle et formules logiques en théorie des catégories. *Archiv für Mathematische Logik und Grundlagenforschung*, 19(3-4):131–137, 1978/79. (Cited on page 331.)

[Bou68] Nicolas Bourbaki. *Theory of Sets*. Hermann, Paris, 1968. (Cited on page 96.)

[BSP11] Clark Barwick and Christopher Schommer-Pries. On the unicity of the homotopy theory of higher categories, 2011. arXiv:1112.0040. (Cited on page 306.)

[BT09] Andrej Bauer and Paul Taylor. The Dedekind reals in abstract Stone duality. *Mathematical structures in computer science*, 19(4):757–838, 2009. (Cited on page 408.)

[Bun79] Marta Bunge. Stack completions and Morita equivalence for categories in a topos. *Cahiers de Topologie et Géométrie Différentielle*, 20(4):401–436, 1979. (Cited on page 332.)

[CAB+86] Robert L. Constable, Stuart F. Allen, H. M. Bromley, W. R. Cleaveland, J. F. Cremer, Robert W. Harper, Douglas J. Howe, T. B. Knoblock, N. P. Mendler, P. Panangaden, James T. Sasaki, and Scott F. Smith. *Implementing Mathematics with the NuPRL Proof Development System*. Prentice Hall, 1986. (Cited on pages 51, 53, 117, and 203.)

[Car95] Aurelio Carboni. Some free constructions in realizability and proof theory. *Journal of Pure and Applied Algebra*, 103:117–148, 1995. (Cited on page 204.)

[CDP14] Jesper Cockx, Dominique Devriese, and Frank Piessens. Pattern matching without K. In *Proceedings of the 19th ACM SIGPLAN International Conference on Functional Programming, ICFP 2014, Gothenburg, Sweden, September 1-3, 2014*, 2014. (Cited on page 52.)

[Chu40] Alonzo Church. A formulation of of the simple theory of types. *Journal of Symbolic Logic*, 5:56–68, 1940. (Cited on page 2.)

[Chu41] Alonzo Church. *The Calculi of Lambda Conversation*. Princeton University Press, 1941. (Cited on page 2.)

[CM85] Robert L. Constable and N. P. Mendler. Recursive definitions in type theory. In Rohit Parikh, editor, *Logics of Programs, Conference, Brooklyn College, June 17–19, 1985, Proceedings*, volume 193 of *Lecture Notes in Computer Science*, pages 61–78, 1985. (Cited on page 163.)

[Con76] John H. Conway. *On numbers and games*. A K Peters Ltd., 1976. (Cited on pages 397, 406, and 408.)

[Con85] Robert L. Constable. Constructive mathematics as a programming logic I: Some principles of theory. In *Annals of Mathematics*, volume 24, pages 21–37. Elsevier Science Publishers, B.V. (North-Holland), 1985. Reprinted from *Topics in the Theory of Computation*, Selected Papers of the International Conference on Foundations of Computation Theory, FCT '83. (Cited on page 117.)

[Coq92a] Thierry Coquand. The paradox of trees in type theory. *BIT Numerical Mathematics*, 32(1):10–14, 1992. (Cited on page 23.)

[Coq92b] Thierry Coquand. Pattern matching with dependent types. In *Proceedings of the Workshop on Types for Proofs and Programs*, pages 71–83, 1992. (Cited on page 52.)

[Coq12] Coq Development Team. *The Coq Proof Assistant Reference Manual*. INRIA-Rocquencourt, 2012. (Cited on pages 51 and 426.)

[CP90] Thierry Coquand and Christine Paulin. Inductively defined types. In *COLOG-88 (Tallinn, 1988)*, volume 416 of *Lecture Notes in Computer Science*, pages 50–66. Springer, 1990. (Cited on page 163.)

[dB73] Nicolaas Govert de Bruijn. *AUTOMATH, a language for mathematics*. Les Presses de l'Université de Montréal, Montreal, Quebec, 1973. Séminaire de Mathématiques Supérieures, No. 52 (Été 1971). (Cited on pages 51, 52, and 429.)

[Dia75] Radu Diaconescu. Axiom of choice and complementation. *Proceedings of the American Mathematical Society*, 51:176–178, 1975. (Cited on page 360.)

[dPGM04] Valeria de Paiva, Rajeev Goré, and Michael Mendler. Modalities in constructive logics and type theories. *Journal of Logic and Computation*, 14(4):439–446, 2004. (Cited on page 234.)

[Dyb91] Peter Dybjer. Inductive sets and families in Martin-Löf's type theory and their set-theoretic semantics. In Gerard Huet and Gordon Plotkin, editors, *Logical Frameworks*, pages 280–30. Cambridge University Press, 1991. (Cited on page 163.)

[Dyb00] Peter Dybjer. A general formulation of simultaneous inductive-recursive definitions in type theory. *Journal of Symbolic Logic*, 65(2):525–549, 2000. (Cited on page 163.)

[ES01] Martín Hötzel Escardó and Alex K. Simpson. A universal characterization of the closed Euclidean interval. In *16th Annual IEEE Symposium on Logic in Computer Science, Boston, Massachusetts, USA, June 16-19, 2001, Proceedings*, pages 115–125. IEEE Computer Society, 2001. (Cited on page 408.)

[EucBC] Euclid. *Elements, Vols. 1–13*. Elsevier, 300 BC. (Cited on page 57.)

[Fre76] Peter Freyd. Properties invariant within equivalence types of categories. In *Algebra, topology, and category theory (a collection of papers in honor of Samuel Eilenberg)*, pages 55–61. Academic Press, 1976. (Cited on page 331.)

[FS12] Fredrik Nordvall Forsberg and Anton Setzer. A finite axiomatisation of inductive-inductive definitions. `http://cs.swan.ac.uk/~csfnf/papers/indind_finite.pdf`, 2012. (Cited on page 408.)

[GAA+13] Georges Gonthier, Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, François Garillot, Stéphane Le Roux, Assia Mahboubi, Russell O'Connor, Sidi Ould Biha, Ioana Pasca, Laurence Rideau, Alexey Solovyev, Enrico Tassi, and Laurent Thery. A machine-checked proof of the odd order theorem. In *Interactive Theorem Proving*, 2013. (Cited on page 6.)

[Gar09] Richard Garner. On the strength of dependent products in the type theory of Martin-Löf. *Annals of Pure and Applied Logic*, 160(1):1–12, 2009. (Cited on page 96.)

[Gen36] Gerhard Gentzen. Die Widerspruchsfreiheit der reinen Zahlentheorie. *Mathematische Annalen*, 112(1):493–565, 1936. (Cited on page 360.)

[Göd58] Kurt Gödel. Über eine bisher noch nicht benützte Erweiterung des finiten Standpunktes. *Dialectica. International Journal of Philosophy*, 12:280–287, 1958. (Cited on page 429.)

[Hat02] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002. `http://www.math.cornell.edu/~hatcher/AT/ATpage.html`. (Cited on page 300.)

[Hed98] Michael Hedberg. A coherence theorem for Martin-Löf's type theory. *Journal of Functional Programming*, 8(4):413–436, 1998. (Cited on page 233.)

[Hey66] Arend Heyting. *Intuitionism: an introduction*. Studies in logic and the foundations of mathematics. North-Holland Pub. Co., 1966. (Cited on page 51.)

[Hil26] David Hilbert. Über das Unendliche. *Mathematische Annalen*, 95(1):161–190, 1926. (Cited on page 429.)

[Hof95] Martin Hofmann. *Extensional concepts in intensional type theory*. PhD thesis, University of Edinburgh, 1995. (Cited on page 204.)

[Hof97] Martin Hofmann. Syntax and semantics of dependent types. In *Semantics and logics of computation*, volume 14 of *Publictions of the Newton Institute*, pages 79–130. Cambridge University Press, Cambridge, 1997. (Cited on page 429.)

[How80] William A. Howard. The formulae-as-types notion of construction. In J. Roger Seldin, Jonathan P.; Hindley, editor, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490. Academic Press, 1980. original paper manuscript from 1969. (Cited on pages 51, 52, 96, and 429.)

[HS98] Martin Hofmann and Thomas Streicher. The groupoid interpretation of type theory. In Giovanni Sambin and Jan M. Smith, editors, *Twenty-five years of constructive type theory (Venice, 1995)*, volume 36 of *Oxford Logic Guides*, pages 83–111. Oxford University Press, New York, 1998. (Cited on pages 4, 96, and 331.)

[Hue80] Gérard Huet. Confluent reductions: Abstract properties and applications to term rewriting systems: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, 1980. (Cited on page 360.)

[Jac99] Bart Jacobs. *Categorical logic and type theory*, volume 141 of *Studies in Logic and the Foundations of Mathematics*. Elsevier, 1999. (Cited on page 117.)

[JM95] A. Joyal and I. Moerdijk. *Algebraic set theory*, volume 220 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 1995. (Cited on pages 360 and 362.)

[Joh02] Peter T. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium: Volumes 1 and 2*. Number 43 in Oxford Logic Guides. Oxford Science Publications, 2002. (Cited on pages 117 and 339.)

[JT91] André Joyal and Myles Tierney. Strong stacks and classifying spaces. In *Category Theory. Proceedings of the International Conference held in Como, Italy, July 22–28, 1990*, volume 1488 of *Lecture Notes in Mathematics*, pages 213–236. Springer, Berlin, 1991. (Cited on page 332.)

[KECA13] Nicolai Kraus, Martın Escardó, Thierry Coquand, and Thorsten Altenkirch. Generalizations of Hedberg's theorem. In Masahito Hasegawa, editor, *11th International Conference, Typed Lambda Calculus and Applications 2013, Eindhoven, The Netherlands, June 26–28, 2013. Proceedings*, volume 7941 of *Lecture Notes in Computer Science*, pages 173–188. Springer Berlin Heidelberg, 2013. (Cited on pages 118 and 233.)

[KLN04] Fairouz Kamareddine, Twan Laan, and Rob Nederpelt. *A Modern Perspective on Type Theory: From its Origins until Today*. Number 29 in Applied Logic. Kluwer, 2004. (Cited on page 2.)

[KLV12] Chris Kapulkin, Peter LeFanu Lumsdaine, and Vladimir Voevodsky. The simplicial model of univalent foundations, 2012. arXiv:1211.2851. (Cited on pages 10, 96, 163, and 429.)

[Knu74] Donald Ervin Knuth. *Surreal Numbers*. Addison-Wesley, 1974. (Cited on page 410.)

[Kol32] Andrey Kolmogorov. Zur Deutung der intuitionistischen Logik. *Mathematische Zeitschrift*, 35:58–65, 1932. (Cited on page 8.)

[Law74] F. William Lawvere. Metric spaces, generalized logic, and closed categories. *Rendiconti del Seminario Matematico e Fisico di Milano*, 43:135–166, 1974. Reprinted as Reprints in Theory and Applications of Categories 1:1–37, 2002. (Cited on page 409.)

[Law05] F. William Lawvere. An elementary theory of the category of sets (long version) with commentary. *Reprints in Theory and Applications of Categories*, 11:1–35, 2005. Reprinted and expanded from Proc. Nat. Acad. Sci. U.S.A. **52** (1964), With comments by the author and Colin McLarty. (Cited on pages 6, 343, and 360.)

[Law06] F. William Lawvere. Adjointness in foundations. *Reprints in Theory and Applications of Categories*, 16:1–16, 2006. Reprinted from Dialectica **23** (1969). (Cited on pages 51 and 163.)

[LH12] Daniel R. Licata and Robert Harper. Canonicity for 2-dimensional type theory. In *Proceedings of the 39th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 337–348, New York, NY, USA, 2012. ACM. (Cited on pages 9, 10, and 96.)

[LS13] Daniel R. Licata and Michael Shulman. Calculating the fundamental group of the circle in homotopy type theory. In *LICS 2013: Proceedings of the Twenty-Eighth Annual ACM/IEEE Symposium on Logic in Computer Science*, 2013. (Cited on page 96.)

[LS17] Peter LeFanu Lumsdaine and Michael Shulman. Semantics of higher inductive types. arXiv:1705.07088, 2017. (Cited on pages 10, 163, 203, and 429.)

[Lum10] Peter LeFanu Lumsdaine. Weak ω-categories from intensional type theory. *Typed lambda calculi and applications*, 6:1–19, 2010. arXiv:0812.0409. (Cited on page 96.)

[Lur09] Jacob Lurie. *Higher topos theory*. Number 170 in Annals of Mathematics Studies. Princeton University Press, 2009. arXiv:math.CT/0608040. (Cited on pages 9, 137, 234, and 295.)

[Mak95] Michael Makkai. First order logic with dependent sorts, with applications to category theory. `http://www.math.mcgill.ca/makkai/folds/`, 1995. (Cited on page 331.)

[Mak01] Michael Makkai. On comparing definitions of weak n-category. `http://www.math.mcgill.ca/makkai/`, August 2001. (Cited on page 331.)

[ML71] Per Martin-Löf. Hauptsatz for the intuitionistic theory of iterated inductive definitions. In *Proceedings of the Second Scandinavian Logic Symposium (University of Oslo 1970)*, volume 63 of *Studies in Logic and the Foundations of Mathematics*, pages 179–216. North-Holland, 1971. (Cited on page 163.)

[ML75] Per Martin-Löf. An intuitionistic theory of types: predicative part. In H.E. Rose and J.C. Shepherdson, editors, *Logic Colloquium '73, Proceedings of the Logic Colloquium*, volume 80 of *Studies in Logic and the Foundations of Mathematics*, pages 73–118. North-Holland, 1975. (Cited on pages 2, 51, 163, and 429.)

[ML82] Per Martin-Löf. Constructive mathematics and computer programming. In L. Jonathan Cohen, Jerzy Łoś, Helmut Pfeiffer, and Klaus-Peter Podewski, editors, *Logic, Methodology and Philosophy of Science VI, Proceedings of the Sixth International Congress of Logic, Methodology and Philosophy of Science, Hannover 1979*, volume 104 of *Studies in Logic and the Foundations of Mathematics*, pages 153–175. North-Holland, 1982. (Cited on pages 2, 51, 163, and 429.)

[ML84] Per Martin-Löf. *Intuitionistic type theory*, volume 1 of *Studies in Proof Theory*. Bibliopolis, 1984. Notes by Giovanni Sambin of a series of lectures given in Padua, June 1980. (Cited on pages 2, 51, 52, and 429.)

[ML98] Per Martin-Löf. An intuitionistic theory of types. In Giovanni Sambin and Jan M. Smith, editors, *Twenty-five years of constructive type theory (Venice, 1995)*, volume 36 of *Oxford Logic Guides*, pages 127–172. Oxford University Press, 1998. (Cited on pages 2, 51, 52, 53, 95, and 429.)

[ML06] Per Martin-Löf. 100 years of Zermelo's axiom of choice: what was the problem with it? *The Computer Journal*, 49(3):345–350, 2006. (Cited on page 117.)

[Mog89] Eugenio Moggi. Notions of computation and monads. *Information and Computation*, 93:55–92, 1989. (Cited on page 234.)

[MP00] Ieke Moerdijk and Erik Palmgren. Wellfounded trees in categories. In *Proceedings of the Workshop on Proof Theory and Complexity, PTAC'98 (Aarhus)*, volume 104, pages 189–218, 2000. (Cited on page 163.)

[MP02] Ieke Moerdijk and Erik Palmgren. Type theories, toposes and constructive set theory: predicative aspects of AST. *Annals of Pure and Applied Logic*, 114(1–3):155–201, 2002. (Cited on page 360.)

[MRR88] Ray Mines, Fred Richman, and Wim Ruitenburg. *A course in constructive algebra*. Springer-Verlag, 1988. (Cited on page 360.)

[MS05] Maria Emilia Maietti and Giovanni Sambin. Toward a minimalist foundation for constructive mathematics. In Laura Crosilla and Peter Schuster, editors, *From Sets and Types to Topology and Analysis: Practicable Foundations for Constructive Mathematics*, volume 48 of *Oxford Logic Guides*, pages 91–114. Clarendon Press, 2005. (Cited on page 117.)

[Nor88] Bengt Nordström. Terminating general recursion. *BIT Numerical Mathematics*, 28(3):605–619, 1988. (Cited on page 360.)

[Nor07] Ulf Norell. *Towards a practical programming language based on dependent type theory*. PhD thesis, Chalmers, Göteborg University, 2007. (Cited on pages 51 and 426.)

[Pal07] Erik Palmgren. A constructive and functorial embedding of locally compact metric spaces into locales. *Topology and its Applications*, 154(9):1854–1880, 2007. (Cited on page 408.)

[Pal09] Erik Palmgren. Constructivist and structuralist foundations: Bishop's and Lawvere's theories of sets. `http://www.math.uu.se/~palmgren/cetcs.pdf`, 2009. (Cited on page 360.)

[Pau86] Lawrence C. Paulson. Constructing recursion operators in intuitionistic type theory. *Journal of Symbolic Computation*, 2(4):325–355, 1986. (Cited on page 360.)

[Pie02] Benjamin C. Pierce. *Types and Programming Languages*. MIT Press, 2002. (Cited on page 2.)

[PM93] Christine Paulin-Mohring. Inductive Definitions in the System Coq - Rules and Properties. In Marc Bezem and Jan Friso Groote, editors, *Proceedings of the conference Typed Lambda Calculi and Applications*, number 664 in Lecture Notes in Computer Science, 1993. (Cited on page 53.)

[PPM90] Frank Pfenning and Christine Paulin-Mohring. Inductively defined types in the calculus of constructions. In Michael G. Main, Austin Melton, Michael W. Mislove, and David A. Schmidt, editors, *Mathematical Foundations of Programming Semantics, 5th International Conference, Tulane University, New Orleans, Louisiana, USA, March 29 – April 1, 1989, Proceedings*, number 442 in Lecture Notes in Computer Science, pages 209–228. Springer, 1990. (Cited on page 163.)

[PS89] Kent Petersson and Dan Synek. A set constructor for inductive sets in Martin-Löf's type theory. In David H. Pitt, David E. Rydeheard, Peter Dybjer, Andrew M. Pitts, and Axel Poigné, editors, *Category Theory and Computer Science, Manchester, UK, September 5–8, 1989, Proceedings*, volume 389 of *Lecture Notes in Computer Science*, pages 128–140. Springer, 1989. (Cited on page 163.)

[Rez01] Charles Rezk. A model for the homotopy theory of homotopy theory. *Transactions of the American Mathematical Society*, 353(3):973–1007, 2001. arXiv:math.AT/9811037. (Cited on page 331.)

[Rez05] Charles Rezk. Toposes and homotopy toposes. `http://www.math.uiuc.edu/~rezk/homotopy-topos-sketch.pdf`, 2005. (Cited on pages 9 and 137.)

[Ric00] Fred Richman. The fundamental theorem of algebra: a constructive development without choice. *Pacific Journal of Mathematics*, 196(1):213–230, 2000. (Cited on page 409.)

[Ric08] Fred Richman. Real numbers and other completions. *Mathematical Logic Quarterly*, 54(1):98–108, 2008. (Cited on page 408.)

[RS13] Egbert Rijke and Bas Spitters. Sets in homotopy type theory, 2013. arXiv:1305.3835. (Cited on page 360.)

[Rus08] Bertand Russell. Mathematical logic based on the theory of types. *American Journal of Mathematics*, 30:222–262, 1908. (Cited on page 2.)

[Sco70] Dana Scott. Constructive validity. In M. Laudet, D. Lacombe, L. Nolin, and M. Schützenberger, editors, *Symposium on Automatic Demonstration*, volume 125, pages 237–275. Springer-Verlag, 1970. (Cited on page 51.)

[Som10] Giovanni Sommaruga. *History and Philosophy of Constructive Type Theory*. Number 290 in Synthese Library. Kluwer, 2010. (Cited on page 2.)

[Spi11] Arnaud Spiwack. *A Journey Exploring the Power and Limits of Dependent Type Theory*. PhD thesis, École Polytechnique, Palaiseau, France, 2011. (Cited on page 117.)

[SS12] Urs Schreiber and Michael Shulman. Quantum gauge field theory in cohesive homotopy type theory. *Quantum Physics and Logic*, 2012. (Cited on page 234.)

[Str93] Thomas Streicher. Investigations into intensional type theory, 1993. Habilitationsschrift, Ludwig-Maximilians-Universität München. (Cited on pages 52, 53, and 208.)

[Tai67] William W. Tait. Intensional interpretations of functionals of finite type. I. *The Journal of Symbolic Logic*, 32:198–212, 1967. (Cited on pages 51 and 429.)

[Tai68] William W. Tait. Constructive reasoning. In *Logic, Methodology and Philos. Sci. III (Proc. Third Internat. Congr., Amsterdam, 1967)*, pages 185–199. North-Holland, Amsterdam, 1968. (Cited on pages 51, 52, and 429.)

[Tay96] Paul Taylor. Intuitionistic sets and ordinals. *The Journal of Symbolic Logic*, 61(3):705–744, 1996. (Cited on pages 360 and 362.)

[Tay99] Paul Taylor. *Practical Foundations of Mathematics*. Cambridge University Press, 1999. (Cited on pages 360 and 362.)

[TV02] Bertrand Toën and Gabriele Vezzosi. Homotopical algebraic geometry I: Topos theory, 2002. arXiv:math/0207028. (Cited on page 9.)

[TvD88a] Anne Sjerp Troelstra and Dirk van Dalen. *Constructivism in mathematics. Vol. I*, volume 121 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 1988. An introduction. (Cited on pages 8 and 117.)

[TvD88b] Anne Sjerp Troelstra and Dirk van Dalen. *Constructivism in mathematics. Vol. II*, volume 123 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 1988. An introduction. (Cited on pages 8 and 117.)

[vdBG11] Benno van den Berg and Richard Garner. Types are weak ω-groupoids. *Proceedings of the London Mathematical Society*, 102(2):370–394, 2011, http://plms.oxfordjournals.org/content/102/2/370.full.pdf+html. (Cited on page 96.)

[vdBM15] Benno van den Berg and Ieke Moerdijk. W-types in homotopy type theory. *Mathematical Structures in Computer Science*, 25:1100–1115, 6 2015. (Cited on page 163.)

[Voe06] Vladimir Voevodsky. A very short note on the homotopy λ-calculus. `http://www.math.ias.edu/~vladimir/Site3/Univalent_Foundations_files/Hlambda_short_current.pdf`, 2006. (Cited on page 3.)

[Voe12] Vladimir Voevodsky. A universe polymorphic type system. `http://uf-ias-2012.wikispaces.com/file/view/Universe+polymorphic+type+sytem.pdf`, 2012. (Cited on pages 10 and 117.)

[War08] Michael A. Warren. *Homotopy Theoretic Aspects of Constructive Type Theory*. PhD thesis, Carnegie Mellon University, 2008. (Cited on page 96.)

[Wik13] Wikipedia. Homotopy groups of spheres, April 2013. (Cited on page 261.)

[Wil10] Olov Wilander. Setoids and universes. *Mathematical Structures in Computer Science*, 20(4):563–576, 2010. (Cited on page 360.)

[WR27] Alfred North Whitehead and Bertrand Russell. *Principia mathematica, 3 vol.s*. Cambridge University Press, Cambridge, 1910–1913; Second edition, 1925–1927. (Cited on pages 96 and 117.)

Index of symbols

$\mathsf{transportconst}^X_Y(Z)$	transporting in a constant family, p. 69
$\|A\|_n$	n-truncation of A, p. 212
$\|a\|^A_n$, $\|a\|_n$	image of $a : A$ in $\|A\|_n$, p. 212
$\|A\|$	propositional truncation of A, p. 108 and p. 185
$\|a\|$	image of $a : A$ in $\|A\|$, p. 108 and p. 185
$\top$	logical truth, p. 109
$\_$	an unnamed object or variable
$A \cup B$	union of subsets, p. 109
$\mathsf{uniq}_{A\times B}$	uniqueness principle for the product $A \times B$, p. 28
$\mathsf{uniq}_{\mathbf{1}}$	uniqueness principle for $\mathbf{1}$, p. 29
$\mathcal{U}$	universe type, p. 23
$\mathcal{U}_{\bigcirc}$	universe of modal types, p. 232
$\mathcal{U}_{\bullet}$	universe of pointed types, p. 65
ua	inverse to $\mathsf{idtoeqv}$ from univalence, p. 83
V	cumulative hierarchy, p. 355
$\mathsf{WAlg}(A, B)$	type of w-algebras, p. 148
$\mathsf{WHom}_{A,B}(C, D)$	type of W-homomorphisms, p. 148
$\mathsf{W}_{(x:A)}B(x)$	W-type (inductive type), p. 144
$A \vee B$	wedge of A and B, p. 184
$\mathbf{y}$	Yoneda embedding, p. 317
$\mathbb{Z}$	type of integers, p. 189

Index