# Homotopy Type Theory

Higher inductive types

## 6.1 Introduction

## 6.2 Induction principles and dependent paths

*NB* 6.2.1. Recall that for ordinary inductive types, we regard the computation rules for a recursively defined function as not merely judgmental equalities, but *definitional* ones, and thus we may use the notation $:\equiv$ for them. For instance, the truncated predecessor function $p : \mathbb{N} \to \mathbb{N}$ is defined by $p(0) :\equiv 0$ and $p(\mathsf{succ}(n)) :\equiv n$. In the case of higher inductive types, this sort of notation is reasonable for the point constructors (e.g. $f(\mathsf{base}) :\equiv b$), but for the path constructors it could be misleading, since equalities such as $f(\mathsf{loop}) = \ell$ are not judgmental. Thus, we hybridize the notations, writing instead $f(\mathsf{loop}) \coloneqq \ell$ for this sort of "propositional equality by definition".

*NB* 6.2.2. There are other possible ways to define dependent paths. For instance, instead of $p_*(u) = v$ we could consider $u = (p^{-1})_*(v)$. We could also obtain it as a special case of a more general "heterogeneous equality", or with a direct definition as an inductive type family. All these definitions result in equivalent types, so in that sense it doesn't much matter which we pick. However, choosing $p_*(u) = v$ as the definition makes it easiest to conclude other things about dependent paths, such as the fact that $\mathsf{apd}_f$ produces them, or that we can compute them in particular type families using the transport lemmas in **??**.

*NB* 6.2.3. When describing an application of this induction principle informally, we regard it as a splitting of the goal "$P(x)$ for all $x : \mathbb{S}^1$" into two cases, which we will sometimes introduce with phrases such as "when $x$ is $\mathsf{base}$" and "when $x$ varies along $\mathsf{loop}$", respectively. There is no specific mathematical meaning assigned to "varying along a path": it is just a convenient way to indicate the beginning of the corresponding section of a proof; see **??** for an example.

**L 6.2.4.** *If $A$ is a type together with $a : A$ and $p : a =_A a$, then there is a function $f : \mathbb{S}^1 \to A$ with*

$$f(\mathsf{base}) :\equiv a$$
$$\mathsf{ap}_f(\mathsf{loop}) \coloneqq p.$$

**L 6.2.5.** *If $A$ is a type and $f, g : \mathbb{S}^1 \to A$ are two maps together with two equalities $p, q$:*

$$p : f(\mathsf{base}) =_A g(\mathsf{base}),$$
$$q : f(\mathsf{loop}) =^{\lambda x.\, x =_A x}_{p} g(\mathsf{loop}).$$

*Then for all $x : \mathbb{S}^1$ we have $f(x) = g(x)$.*

**L 6.2.6.** *For any type $A$ we have a natural equivalence*

$$(\mathbb{S}^1 \to A) \;\simeq\; \sum_{x:A} (x = x).$$

## 6.3 The interval

**L 6.3.1.** *The type $I$ is contractible.*

**L 6.3.2.** *If $f, g : A \to B$ are two functions such that $f(x) = g(x)$ for every $x : A$, then $f = g$ in the type $A \to B$.*

## 6.4 Circles and spheres

**L 6.4.1.** $\mathsf{loop} \neq \mathsf{refl}_{\mathsf{base}}$.

**L 6.4.2.** *There exists an element of $\prod_{(x:\mathbb{S}^1)} (x = x)$ which is not equal to $x \mapsto \mathsf{refl}_x$.*

**C 6.4.3.** *If the type $\mathbb{S}^1$ belongs to some universe $\mathcal{U}$, then $\mathcal{U}$ is not a 1-type.*

**L 6.4.4.** *Given $f : A \to B$ and $x, y : A$ and $p, q : x = y$, and $r : p = q$, we have a path $\mathsf{ap}^2_f(r) : f(p) = f(q)$.*

**L 6.4.5.** *Given $P : A \to \mathcal{U}$ and $x, y : A$ and $p, q : x = y$ and $r : p = q$, for any $u : P(x)$ we have $\mathsf{transport}^2(r, u) : p_*(u) = q_*(u)$.*

**L 6.4.6.** *Given $P : A \to \mathcal{U}$ and $x, y : A$ and $p, q : x = y$ and $r : p = q$ and a function $f : \prod_{(x:A)} P(x)$, we have $\mathsf{apd}^2_f(r) : \mathsf{apd}_f(p) =^P_r \mathsf{apd}_f(q)$.*

## 6.5 Suspensions

**L 6.5.1.** $\Sigma \mathbf{2} \simeq \mathbb{S}^1$.

**L 6.5.2.** *For a type $A$ and a pointed type $(B, b_0)$, we have*

$$\mathsf{Map}_*(A_+, B) \simeq (A \to B)$$

**L 6.5.3.** *For pointed types $(A, a_0)$ and $(B, b_0)$ we have*

$$\mathsf{Map}_*(\Sigma A, B) \simeq \mathsf{Map}_*(A, \Omega B).$$

## 6.6 Cell complexes

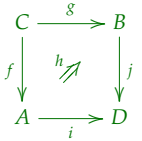
## 6.7 Hubs and spokes

*NB* 6.7.1. One might question the need for introducing the hub point $h$; why couldn't we instead simply add paths continuously relating the boundary of the disc to a point *on* that boundary, as shown in **???** However, this does not work without further modification. For if, given some $f : \mathbb{S}^1 \to X$, we give a path constructor connecting each $f(x)$ to $f(\mathsf{base})$, then what we end up with is more like the picture in **??** of a cone whose vertex is twisted around and glued to some point on its base. The problem is that the specified path from $f(\mathsf{base})$ to itself may not be reflexivity. We could remedy the problem by adding a 2-dimensional path constructor to ensure this, but using a separate hub avoids the need for any path constructors of dimension above 1.

*NB* 6.7.2. Note also that this "translation" of higher paths into 1-paths does not preserve judgmental computation rules for these paths, though it does preserve propositional ones.

## 6.8 Pushouts

**D 6.8.1.** Given a span $\mathscr{D} = (A \xleftarrow{f} C \xrightarrow{g} B)$ and a type $D$, a **cocone under $\mathscr{D}$ with vertex $D$** consists of functions $i : A \to D$ and $j : B \to D$ and a homotopy $h : \prod_{(c:C)} (i(f(c)) = j(g(c)))$:




We denote by $\mathsf{cocone}_{\mathscr{D}}(D)$ the type of all such cocones, i.e.

$$\mathsf{cocone}_{\mathscr{D}}(D) :\equiv \sum_{(i:A\to D)} \sum_{(j:B\to D)} \prod_{(c:C)} (i(f(c)) = j(g(c))).$$

**L 6.8.2.** *For any type $E$, there is an equivalence*

$$(A \sqcup^C B \to E) \;\simeq\; \mathsf{cocone}_{\mathscr{D}}(E).$$

*NB* 6.8.3. As remarked in **??**, the notations $\wedge$ and $\vee$ for the smash product and wedge of pointed spaces are also used in logic for "and" and "or", respectively. Since types in homotopy type theory can behave either like spaces or like propositions, there is technically a potential for conflict — but since they rarely do both at once, context generally disambiguates. Furthermore, the smash product and wedge only apply to *pointed* spaces, while the only pointed mere proposition is $\top \equiv \mathbf{1}$ — and we have $\mathbf{1} \wedge \mathbf{1} = \mathbf{1}$ and $\mathbf{1} \vee \mathbf{1} = \mathbf{1}$ for either meaning of $\wedge$ and $\vee$.

*NB* 6.8.4. Note that colimits do not in general preserve truncatedness. For instance, $\mathbb{S}^0$ and $\mathbf{1}$ are both sets, but the pushout of $\mathbf{1} \leftarrow \mathbb{S}^0 \to \mathbf{1}$ is $\mathbb{S}^1$, which is not a set. If we are interested in colimits in the category of $n$-types, therefore (and, in particular, in the category of sets), we need to "truncate" the colimit somehow. We will return to this point in **??????**.

## 6.9 Truncations

**L 6.9.1.** *Suppose given $B : \|A\|_0 \to \mathcal{U}$ together with $g : \prod_{(a:A)} B(|a|_0)$, and assume that each $B(x)$ is a set. Then there exists $f : \prod_{(x:\|A\|_0)} B(x)$ such that $f(|a|_0) \equiv g(a)$ for all $a : A$.*

**L 6.9.2.** *For any set $B$ and any type $A$, composition with $|-|_0 : A \to \|A\|_0$ determines an equivalence*

$$(\|A\|_0 \to B) \;\simeq\; (A \to B).$$

**L 6.9.3.** *Let $A \xleftarrow{f} C \xrightarrow{g} B$ be a span of sets. Then for any set $E$, there is a canonical equivalence*

$$\left(\|A \sqcup^C B\|_0 \to E\right) \;\simeq\; \mathsf{cocone}_{\mathscr{D}}(E).$$

## 6.10 Quotients

*NB* 6.10.1. It is not actually necessary for the definition of set-quotients, and most of their properties, that $A$ be a set. However, this is generally the case of most interest.

**L 6.10.2.** *The function $q : A \to A/R$ is surjective.*

**L 6.10.3.** *For any set $B$, precomposing with $q$ yields an equivalence*

$$(A/R \to B) \simeq \Big( \sum_{(f:A\to B)} \prod_{(a,b:A)} R(a,b) \to (f(a) = f(b)) \Big).$$

**D 6.10.4.** A predicate $P : A \to \mathsf{Prop}$ is an **equivalence class** of a relation $R : A \times A \to \mathsf{Prop}$ if there merely exists an $a : A$ such that for all $b : A$ we have $R(a,b) \simeq P(b)$.

**D 6.10.5.** We define

$$A /\!\!/ R :\equiv \{\, P : A \to \mathsf{Prop} \mid P \text{ is an equivalence class of } R \,\}.$$

The function $q' : A \to A /\!\!/ R$ is defined by $q'(a) :\equiv P_a$.

**T 6.10.6.** *For any equivalence relation $R$ on $A$, the type $A /\!\!/ R$ is equivalent to the set-quotient $A/R$.*

*NB* 6.10.7. The previous two constructions provide quotients in generality, but in particular cases there may be easier constructions. For instance, we may define the integers $\mathbb{Z}$ as a set-quotient

$$\mathbb{Z} :\equiv (\mathbb{N} \times \mathbb{N})/\sim$$

where $\sim$ is the equivalence relation defined by

$$(a,b) \sim (c,d) :\equiv (a + d = b + c).$$

In other words, a pair $(a,b)$ represents the integer $a - b$. In this case, however, there are *canonical representatives* of the equivalence classes: those of the form $(n,0)$ or $(0,n)$.

**L 6.10.8.** *Suppose $\sim$ is a relation on a set $A$, and there exists an idempotent $r : A \to A$ such that $(r(x) = r(y)) \simeq (x \sim y)$ for all $x, y : A$. (This implies $\sim$ is an equivalence relation.) Then the type*

$$(A/\sim) :\equiv \Big( \sum_{x:A} r(x) = x \Big)$$

*satisfies the universal property of the set-quotient of $A$ by $\sim$, and hence is equivalent to it. In other words, there is a map $q : A \to (A/\sim)$ such that for every set $B$, precomposition with $q$ induces an equivalence*

$$\Big( (A/\sim) \to B \Big) \simeq \Big( \sum_{(g:A\to B)} \prod_{(x,y:A)} (x \sim y) \to (g(x) = g(y)) \Big). \quad (6.10.9)$$

**C 6.10.10.** *Suppose $p : A \to B$ is a retraction between sets. Then $B$ is the quotient of $A$ by the equivalence relation $\sim$ defined by*

$$(a_1 \sim a_2) :\equiv (p(a_1) = p(a_2)).$$

*NB* 6.10.11. **??** applies to $\mathbb{Z}$ with the idempotent $r : \mathbb{N} \times \mathbb{N} \to \mathbb{N} \times \mathbb{N}$ defined by

$$r(a,b) = \begin{cases} (a-b, 0) & \text{if } a \geq b, \\ (0, b-a) & \text{otherwise.} \end{cases}$$

(This is a valid definition even constructively, since the relation $\geq$ on $\mathbb{N}$ is decidable.) Thus a non-negative integer is canonically represented as $(k,0)$ and a non-positive one by $(0,m)$, for $k, m : \mathbb{N}$. This division into cases implies the following "induction principle" for integers, which will be useful in **??**. (As usual, we identify a natural number $n$ with the corresponding non-negative integer, i.e. with the image of $(n,0) : \mathbb{N} \times \mathbb{N}$ in $\mathbb{Z}$.)

**L 6.10.12.** *Suppose $P : \mathbb{Z} \to \mathcal{U}$ is a type family and that we have*

- $d_0 : P(0)$,
- $d_+ : \prod_{(n:\mathbb{N})} P(n) \to P(\mathsf{succ}(n))$, *and*
- $d_- : \prod_{(n:\mathbb{N})} P(-n) \to P(-\mathsf{succ}(n))$.

*Then we have $f : \prod_{(z:\mathbb{Z})} P(z)$ such that*

- $f(0) = d_0$,
- $f(\mathsf{succ}(n)) = d_+(n, f(n))$ *for all* $n : \mathbb{N}$, *and*
- $f(-\mathsf{succ}(n)) = d_-(n, f(-n))$ *for all* $n : \mathbb{N}$.

**C 6.10.13.** *Let $A$ be a type with $a : A$ and $p : a = a$. There is a function $\prod_{(n:\mathbb{Z})}(a = a)$, denoted $n \mapsto p^n$, defined by*

$$\begin{aligned} p^0 &:\equiv \mathsf{refl}_a \\ p^{n+1} &:\equiv p^n \cdot p && \text{for } n \geq 0 \\ p^{n-1} &:\equiv p^n \cdot p^{-1} && \text{for } n \leq 0. \end{aligned}$$

## 6.11 Algebra

**D 6.11.1.** A **monoid** is a set $G$ together with

- a *multiplication* function $G \times G \to G$, written infix as $(x,y) \mapsto x \cdot y$; and
- a *unit* element $e : G$; such that
- for any $x : G$, we have $x \cdot e = x$ and $e \cdot x = x$; and
- for any $x, y, z : G$, we have $x \cdot (y \cdot z) = (x \cdot y) \cdot z$.

A **group** is a monoid $G$ together with

- an *inversion* function $i : G \to G$, written $x \mapsto x^{-1}$; such that
- for any $x : G$ we have $x \cdot x^{-1} = e$ and $x^{-1} \cdot x = e$.

*NB* 6.11.2. Note that we require a group to be a set. We could consider a more general notion of "$\infty$-group" which is not a set, but this would take us further afield than is appropriate at the moment. With our current definition, we may expect the resulting "group theory" to behave similarly to the way it does in set-theoretic mathematics (with the caveat that, unless we assume LEM, it will be "constructive" group theory).

*E* 6.11.3. The natural numbers $\mathbb{N}$ are a monoid under addition, with unit $0$, and also under multiplication, with unit $1$. If we define the arithmetical operations on the integers $\mathbb{Z}$ in the obvious way, then as usual they are a group under addition and a monoid under multiplication (and, of course, a ring). For instance, if $u, v \in \mathbb{Z}$ are represented by $(a,b)$ and $(c,d)$, respectively, then $u + v$ is represented by $(a + c, b + d)$, $-u$ is represented by $(b,a)$, and $uv$ is represented by $(ac + bd, ad + bc)$.

*E* 6.11.4. We essentially observed in **??** that if $(A,a)$ is a pointed type, then its loop space $\Omega(A,a) :\equiv (a =_A a)$ has all the structure of a group, except that it is not in general a set. It should be an "$\infty$-group" in the sense mentioned in **??**, but we can also make it a group by truncation. Specifically, we define the **fundamental group** of $A$ based at $a : A$ to be

$$\pi_1(A,a) :\equiv \|\Omega(A,a)\|_0.$$

This inherits a group structure; for instance, the multiplication $\pi_1(A,a) \times \pi_1(A,a) \to \pi_1(A,a)$ is defined by double induction on truncation from the concatenation of paths.

More generally, the $n^{\text{th}}$ **homotopy group** of $(A,a)$ is $\pi_n(A,a) :\equiv \|\Omega^n(A,a)\|_0$. Then $\pi_n(A,a) = \pi_1(\Omega^{n-1}(A,a))$ for $n \geq 1$, so it is also a group. (When $n = 0$, we have $\pi_0(A) \equiv \|A\|_0$, which is not a group.) Moreover, the Eckmann–Hilton argument (**??**) implies that if $n \geq 2$, then $\pi_n(A,a)$ is an *abelian* group, i.e. we have $x \cdot y = y \cdot x$ for all $x, y$. **??** will be largely the study of these groups.

**L 6.11.5.** *For any set $A$, the type $\mathsf{List}(A)$ is the free monoid on $A$. In other words, for any monoid $G$, composition with $\eta$ is an equivalence*

$$\hom_{\mathrm{Monoid}}(\mathsf{List}(A), G) \simeq (A \to G),$$

*where $\hom_{\mathrm{Monoid}}(-,-)$ denotes the set of monoid homomorphisms (functions which preserve the multiplication and unit).*

**T 6.11.6.** *$F(A)$ is the free group on $A$. In other words, for any (set) group $G$, composition with $\eta : A \to F(A)$ determines an equivalence*

$$\hom_{\mathrm{Group}}(F(A), G) \simeq (A \to G)$$

*where $\hom_{\mathrm{Group}}(-,-)$ denotes the set of group homomorphisms between two groups.*

**T 6.11.7.** *Let $A$ be a set, and let $F'(A)$ be the set-quotient of $\mathsf{List}(A + A)$ by the following relations.*

$$(\dots, a_1, a_2, \widehat{a_2}, a_3, \dots) = (\dots, a_1, a_3, \dots)$$
$$(\dots, a_1, \widehat{a_2}, a_2, a_3, \dots) = (\dots, a_1, a_3, \dots).$$

*Then $F'(A)$ is also the free group on the set $A$.*

*NB* 6.11.8. Nowhere in the construction of $F(A)$ and $F'(A)$, and the proof of their universal properties, did we use the assumption that $A$ is a set. Thus, we can actually construct the free group on an arbitrary type. Comparing universal properties, we conclude that $F(A) \simeq F(\|A\|_0)$.