Higher Inductive Types as Homotopy-Initial Algebras

Kristina Sojakova
Carnegie Mellon University
kristinas@cmu.edu

Abstract

Homotopy Type Theory is a new field of mathematics based on the recently-discovered correspondence between Martin-Löf's constructive type theory and abstract homotopy theory. We have a powerful interplay between these disciplines - we can use geometric intuition to formulate new concepts in type theory and, conversely, use type-theoretic machinery to verify and often simplify existing mathematical proofs.

Higher inductive types form a crucial part of this new system since they allow us to represent mathematical objects, such as spheres, tori, pushouts, and quotients, in the type theory. We investigate a class of higher inductive types called W-suspensions which generalize Martin-Löf's well-founded trees. We show that a propositional variant of W-suspensions, whose computational behavior is determined up to a higher path, is characterized by the universal property of being a homotopy-initial algebra. As a corollary we get that W-suspensions in the strict form are homotopy-initial.

Categories and Subject Descriptors F.4.1 [*Mathematical Logic and Formal Languages*]: Mathematical Logic

Keywords homotopy-initial algebra; W-suspension; higher inductive type; homotopy type theory

1. Introduction

Homotopy Type Theory (HoTT) has recently generated significant interest among type theorists and mathematicians alike. It uncovers deep connections between Martin-Löf's dependent type theory ([17, 18]) and the fields of abstract homotopy theory, higher categories, and algebraic topology ([2, 6–9, 12, 14, 24–27]). Insights from homotopy theory are used to add new concepts to the type theory, such as the representation of various geometric objects as higher inductive types. Conversely, type theory is used to formalize and verify existing mathematical proofs using proof assistants such as Coq [5] and Agda [19]. Moreover, type-theoretic insights often help us discover novel proofs of known results which are simpler than their homotopy-theoretic versions: the calculation of $\pi_n(\mathbf{S}^n)$ ([11, 13]); the Freudenthal Suspension Theorem [23]; the Blakers-Massey Theorem [23], etc.

As a formal system, HoTT [23] is a generalization of intensional Martin-Löf Type Theory with two features motivated by abstract homotopy theory: Voevodsky's *univalence axiom* ([9, 26]) and *higher-inductive types* ([15, 20]). The slogan in HoTT is that *types are topological spaces, terms are points, and proofs of identity are paths between points*. The structure of an identity type in HoTT is thus far more complex than just consisting of reflexivity paths, despite the definition of $\mathsf{Id}_A(x, y)$ as an inductive type with a single constructor $\mathsf{refl}_A(x) : \mathsf{Id}_A(x, x)$. It is a beautiful, and perhaps surprising, fact that not only does this richer theory admit an interpretation into homotopy theory ([2], [9]) but that many fundamental concepts and results from mathematics arise naturally as constructions and theorems of HoTT.

For example, the unit circle $\mathbf{S}^1$ is defined as a higher inductive type with a point base and a loop loop based at base. It comes with a recursion principle which says that to construct a function $f : \mathbf{S}^1 \to X$, it suffices to supply a point $x : X$ and a loop based at x. The value $f(\mathsf{base})$ then computes to x. Such definitional computation rules are convenient to work with but also pose some conceptual difficulties. For instance, an alternative encoding of the circle as a higher inductive type $\mathbf{S}^1_a$ specifies two points south, north and two paths from north to south, called east and west. The recursion principle then says that in order to construct a function $f : \mathbf{S}^1_a \to X$, it suffices to supply two points $x, y : X$ and two paths between them. The values $f(\mathsf{north})$ and $f(\mathsf{south})$ then compute to x and y respectively.

We have a natural way of relating these two representations via an equivalence: in one direction, map base to north and loop to east; in the other direction, map both north and south to base and map east to loop and west to the identity path at base. Unfortunately, the types $\mathbf{S}^1$, $\mathbf{S}^1_a$ related this way, while equivalent, do *not* satisfy the same definitional laws, which poses a compatibility issue. Even more importantly, we do not have a way of *internalizing* these notions of a circle and working with them inside the type theory, since we can only talk about definitional equalities on the meta-level.

In this paper we thus study higher inductive types abstractly, as arbitrary types endowed with certain constructors and *propositional* computation behavior: in the case of $\mathbf{S}^1$, for example, we say that a type C with constructors $b : C$ and $l : c = c$ satisfies the recursion principle for a circle if for any other type X, point $x : X$ and loop based at x, there exists a function $f : C \to X$ for which there is a *path between* $f(b)$ *and* x (and which satisfies a higher coherence condition). We note that we require *no* change to the underlying type theory: the particular higher inductive type $\mathbf{S}^1$ just becomes a specific instance of the abstract definition of a circle, one whose computation rules happen to hold definitionally.

A major advantage of types with propositional computation rules is that we can internalize the definitions and reason about them within the type theory - and in particular, use proof assistants to verify the results. In this respect, our work is complementary to [16], which gives an external, category-theoretic semantics for a certain class of higher inductives. Another advantage of propositional computation behavior is portability: relaxing the computation laws satisfied by the types $\mathbf{S}^1$ and $\mathbf{S}^1_a$ to their propositional

POPL '15, January 15–17 2015, Mumbai, India.
Copyright © 2015 ACM 978-1-4503-3300-9/15/01...$15.00.
http://dx.doi.org/10.1145/2676726.2676983

counterparts results in two notions of a circle that are equivalent. This in particular means that any type C which is a circle in one sense is also a circle in the alternate sense. We can thus state and prove results about either of these specifications, knowing that the proofs carry over to any particular implementation - be it $\mathbf{S}^1$, $\mathbf{S}^1_a$, or a third one.

It further turns out that types with propositional rules tend to keep many of their desirable properties; for instance, it can be shown that the main result of [13], that the fundamental group of the circle is the group of integers, carries over to the case when *both* the circle and the integer types have propositional computational behavior. In addition, we can now show that higher inductive types are characterized by the universal property of being a *homotopy-initial algebra*. This notion was first introduced in [3], where an analogous result was established for the "ordinary" inductive type of well-founded trees (Martin-Löf's W-types). In the higher-dimensional setting, an *algebra* is a type X together with a number of finitary operations $f, g, h \ldots$, which are allowed to act not only on X but also on any higher identity type over X. An *algebra homomorphism* has to preserve all operations up to a higher homotopy. Finally, an algebra $\mathcal{X}$ is *homotopy-initial* if the type of homomorphisms from $\mathcal{X}$ to any other algebra $\mathcal{Y}$ is contractible.

Our main theorem is stated for a class of higher inductive types which we call W-suspensions; they generalize ordinary W-types as well as the higher inductive type $\mathbb{S}$ and others. We show that the induction principle for W-suspensions is equivalent (as a type) to homotopy-initiality. This extends the main result of [3] for "ordinary" inductive types to the important, and much more difficult, higher-dimensional case.

2. Basic Homotopy Type Theory

The core of HoTT is a dependent type theory with

- dependent pair types $\Sigma_{x:A}B(x)$ and dependent function types $\Pi_{x:A}B(x)$ (with the non-dependent versions $A \times B$ and $A \to B$). To stay consistent with the presentation in [23], we assume definitional η-conversion for functions but do not assume it for pairs.
- a cumulative hierarchy of universes $\mathcal{U}_0 : \mathcal{U}_1 : \mathcal{U}_2 : \ldots$ in the style of Russell.
- intensional identity types $\mathsf{Id}_A(x, y)$, also denoted $x =_A y$. We have the usual formation and introduction rules; the elimination and computation rules are recalled below:

$$\frac{E : \Pi_{x,y:A}\mathsf{Id}_A(x,y) \to \mathcal{U}_i \qquad d : \Pi_{x:A}E(x,x,\mathsf{refl}_A(x))}{\mathsf{J}(E,d) : \Pi_{x,y:A}\Pi_{p:\mathsf{Id}_A(x,y)}E(x,y,p)}$$

$$\frac{\begin{array}{c}E : \Pi_{x,y:A}\mathsf{Id}_A(x,y) \to \mathcal{U}_i \\ d : \Pi_{x:A}E(x,x,\mathsf{refl}_A(x)) \qquad a : A\end{array}}{\mathsf{J}(E,d)(a,a,\mathsf{refl}_A(a)) \equiv d(a) : E(a,a,\mathsf{refl}_A(a))}$$

These rules are, of course, applicable in any context Γ; we follow the standard convention of omitting it. If the type $\mathsf{Id}_A(x,y)$ is inhabited, we call x and y *equal*. If we do not care about the specific equality witness, we often simply say that $x =_A y$ or if the type A is clear, $x = y$. A term $p : x =_A y$ will be often called a *path* and the process of applying the identity elimination rule will be referred to as *path induction*. Definitional equality between $x, y : A$ will be denoted as $x \equiv y : A$.

We emphasize that apart from the aforementioned identity rules, univalence, and higher inductive types there are no other rules governing the behavior of identity types - in particular, we assert neither any form of Streicher's K-rule [22] nor the identity reflection rule.

The rest of this section describes the univalence axiom and some key properties of identity types; higher inductive types are discussed in Section. 3. For a thorough exposition of homotopy type theory we refer the reader to [23].

2.1 Groupoid laws

Proofs of identity behave much like paths in topological spaces: they can be reversed, concatenated, mapped along functions, etc. Below we summarize a few of these properties:

- For any path $p : x =_A y$ there is a path $p^{-1} : y =_A x$, and we have $\mathsf{refl}_A(x)^{-1} \equiv \mathsf{refl}_A(x)$.
- For any paths $p : x =_A y$ and $q : y =_A z$ there is a path $p \cdot q : x =_A z$, and we have $\mathsf{refl}_A(x) \cdot \mathsf{refl}_A(x) \equiv \mathsf{refl}_A(x)$.
- Associativity of composition: for any paths $p : x =_A y$, $q : y =_A z$, and $r : z =_A u$ we have $(p \cdot q) \cdot r = p \cdot (q \cdot r)$.
- We have $\mathsf{refl}_A(x) \cdot p = p$ and $p \cdot \mathsf{refl}_A(y) = p$ for any $p : x =_A y$.
- For any $p : x =_A y$, $q : y =_A z$ we have $p \cdot p^{-1} = \mathsf{refl}_A(x)$, $p^{-1} \cdot p = \mathsf{refl}_A(y)$, and $(p^{-1})^{-1} = p$, $(p \cdot q)^{-1} = q^{-1} \cdot p^{-1}$.
- For any type family $P : A \to \mathcal{U}_i$ and path $p : x =_A y$ there are functions $p^P_* : P(x) \to P(y)$ and $p^*_P : P(y) \to P(x)$, called the *covariant transport* and *contravariant transport*, respectively. We furthermore have $\mathsf{refl}_A(x)^P_* \equiv \mathsf{refl}_A(x)^*_P \equiv \mathsf{id}_{P(x)}$.
- We have $(p^{-1})^P_* = p^*_P$, $(p^{-1})^*_P = p^P_*$ and $(p \cdot q)^P_* = q^P_* \circ p^P_*$, $(p \cdot q)^*_P = p^*_P \circ q^*_P$ for any family $P : A \to \mathcal{U}_i$ and paths $p : x =_A y$, $q : y =_A z$.
- For any function $f : A \to B$ and path $p : x =_A y$, there is a path $\mathsf{ap}_f(p) : f(x) =_B f(y)$ and we have $\mathsf{ap}_f(\mathsf{refl}_A(x)) \equiv \mathsf{refl}_B(f(x))$.
- We have $\mathsf{ap}_f(p^{-1}) = \mathsf{ap}_f(p)^{-1}$ and $\mathsf{ap}_f(p \cdot q) = \mathsf{ap}_f(p) \cdot \mathsf{ap}_f(q)$ for any $f : A \to B$ and $p : x =_A y$, $q : y =_A z$.
- We have $\mathsf{ap}_{g \circ f}(p) = \mathsf{ap}_g(\mathsf{ap}_f(p))$ for any $f : A \to B$, $g : B \to C$ and $p : x =_A y$.
- For a dependent function $f : \Pi_{x:A}B(x)$ and path $p : x =_A y$, there are paths $\mathsf{dap}_f(p) : p^B_*(f(x)) =_{B(y)} f(y)$ and $\mathsf{dap}^f(p) : p^*_B(f(y)) =_{B(x)} f(x)$. We also have $\mathsf{dap}_f(\mathsf{refl}_A(x)) \equiv \mathsf{dap}^f(\mathsf{refl}_A(x)) \equiv \mathsf{refl}_{B(x)}(f(x))$.
- All constructs respect propositional equality.

2.2 Homotopies between functions

A homotopy between two functions is in a sense a "natural transformation":

Definition 1. *For $f, g : \Pi_{x:A}B(x)$, we define the type*

$$f \sim g :\equiv \Pi_{a:A}(f(a) =_{B(a)} g(a))$$

and call it the type of homotopies *between f and g.*

Definition 2. *For $f : A \to B$ and $g : A' \to B$, we define the type*

$$f \sim_{\mathcal{H}} g :\equiv \Pi_{a:A}\Pi_{a':A}(f(a) =_B g(a'))$$

and call it the type of heterogeneous homotopies *between f and g.*

We now introduce some notation that will be needed later.

- For any $f, g : X \to Y$, $p : x =_X y$, $\alpha : f \sim g$, there is a path
$$\mathsf{nat}(\alpha, p) : \alpha(x) \cdot \mathsf{ap}_g(p) = \mathsf{ap}_f(p) \cdot \alpha(y)$$
- For any $f : X \to Z$, $g : Y \to Z$, $p : x_1 =_X x_2$, $q : y_1 =_Y y_2$, $\alpha : f \sim_{\mathcal{H}} g$, there is a path
$$\mathsf{nat}_{\mathcal{H}}(\alpha, p, q) : \alpha(x_1, y_1) \cdot \mathsf{ap}_g(q) = \mathsf{ap}_f(p) \cdot \alpha(x_2, y_2)$$

2.3 Truncation levels

In general, the structure of paths on a type A can be highly nontrivial - we can have many distinct *0-cells* $x, y, \ldots : A$; there can be many distinct *1-cells* $p, q, \ldots : x =_A y$; there can be many distinct *2-cells* $\gamma, \delta, \ldots : p =_{x =_A y} q$; ad infinitum. The hierarchy of truncation levels describes those types which are, informally speaking, trivial beyond a certain dimension: a type A of truncation level n can be characterized by the property that all m-cells for $m > n$ with the same source and target are equal. From this intuitive description we can see that the hierarchy is cumulative.

It is customary to also speak of truncation levels -2 and -1, called *contractible types* and *mere propositions* respectively:

Definition 3. *A type $A : \mathcal{U}_i$ is called* contractible *if there exists a point $a : A$ such that any other point $x : A$ is equal to a:*

$$\mathsf{is\text{-}contr}(A) :\equiv \Sigma_{a:A}\Pi_{x:A}(a =_A x)$$

A type $A : \mathcal{U}_i$ is called a mere proposition *if all its inhabitants are equal:*

$$\mathsf{is\text{-}prop}(A) :\equiv \Pi_{x,y:A}(x =_A y)$$

Thus, a contractible type can be seen as having exactly one inhabitant, up to equality; a mere proposition can be seen as having at most one inhabitant, up to equality. Clearly:

Lemma 4. *If A is contractible then A is a mere proposition.*

The existence of a path between any two points implies more than just path-connectedness:

Lemma 5. *If A is a mere proposition, then $x =_A y$ is contractible for any $x, y : A$.*

Thus, contractible types are in a sense the "nicest" possible: any two points are equal up to a *1-cell*, which itself is unique up to a *2-cell*, which itself is unique up to a *3-cell*, and so on. Mere propositions are the "nicest" ones after contractible spaces. We can now easily show:

Corollary 6. *For any A, $\mathsf{is\text{-}contr}(A)$ and $\mathsf{is\text{-}prop}(A)$ are mere propositions.*

2.4 Equivalences

A crucial concept in HoTT is that of an equivalence between types.

Definition 7. *A map $f : A \to B$ is called an* equivalence *if it has both a left and a right inverse:*

$$\mathsf{iseq}(f) :\equiv \big(\Sigma_{g:B\to A}(g \circ f \sim \mathsf{id}_A)\big) \times \big(\Sigma_{h:B\to A}(f \circ h \sim \mathsf{id}_B)\big)$$

We define

$$(A \simeq B) :\equiv \Sigma_{f:A\to B}\mathsf{iseq}(f)$$

and call A and B equivalent *if the above type is inhabited.*

Unsurprisingly, we can prove that A and B are equivalent by constructing functions going back and forth, which compose to identity on both sides[1]; this is also a necessary condition.

Lemma 8. *Two types A and B are equivalent if and only if there exist functions $f : A \to B$ and $g : B \to A$ such that $g \circ f \sim \mathsf{id}_A$ and $f \circ g \sim \mathsf{id}_B$.*

We will refer to such functions f and g as forming a *quasi-equivalence* and say that f and g are *quasi-inverses* of each other. From this we can easily show:

Lemma 9. *Equivalence of types is an equivalence relation.*

[1] Although the type of such functions itself is not equivalent to $A \simeq B$, see Chpt. 4 of [23].

We call A and B *logically equivalent* if there are exist functions $f : A \to B$, $g : B \to A$. It is immediate that if both types are mere propositions then logical equivalence implies $A \simeq B$. For example:

Corollary 10. *For any A, $\mathsf{is\text{-}contr}(A) \simeq (A \times \mathsf{is\text{-}prop}(A))$.*

Many "diagram-like" operations on paths turn out to be equivalences. For instance:

- For any $u : a =_X b$, $v : b =_X d$, $w : a =_X c$, $z : c =_X d$ there is a map
$$\mathbf{I}^1_\square : (u = w \cdot z \cdot v^{-1}) \to (u \cdot v = w \cdot z)$$
defined in the obvious way by induction on v and w. This map is an equivalence.
- For any $u : a =_X b$, $v : b =_X d$, $w : a =_X c$, $z : c =_X d$ there is a map
$$\mathbf{I}^2_\square : (u = w \cdot z \cdot v^{-1}) \to (w^{-1} \cdot u = z \cdot v^{-1})$$
defined in the obvious way by induction on v and w. This map is an equivalence.

2.5 Structure of path types

Let us first consider the product type $A \times B$. We would like for two pairs $c, d : A \times B$ to be equal precisely when their first and second projections are equal. By path induction we can easily construct a function

$$^{=}\mathsf{E}^{\times}_{c,d} : (c = d) \to (\pi_1(c) = \pi_1(d)) \times (\pi_2(c) = \pi_2(d))$$

We can show:

Lemma 11. *The map $^{=}\mathsf{E}^{\times}_{c,d}$ is an equivalence for any $c, d : A \times B$.*

We will denote the quasi-inverse of $^{=}\mathsf{E}^{\times}_{c,d}$ by $^{\times}\mathsf{E}^{=}_{c,d}$. For brevity we will often omit the subscripts.

We have a similar correspondence for dependent pairs; however, the second projections of $c, d : \Sigma_{x:A}B(x)$ now lie in different fibers of B and we employ (covariant) transport. By path induction we can define a map

$$^{=}\mathsf{E}^{\Sigma}_{c,d} : (c = d) \to \Sigma_{(p:\pi_1(c)=\pi_1(d))}(p^B_*(\pi_2(c)) = \pi_2(d))$$

Lemma 12. *The map $^{=}\mathsf{E}^{\Sigma}_{c,d}$ is an equivalence for any $c, d : \Sigma_{x:A}B(x)$.*

We will denote the quasi-inverse of $^{=}\mathsf{E}^{\Sigma}_{c,d}$ by $^{\Sigma}\mathsf{E}^{=}_{c,d}$. We also have an analogous correspondence using a contravariant transport.

We would like for two types $A, B : \mathcal{U}_i$ to be equal precisely when they are equivalent. As before, we can easily obtain a function

$$^{=}\mathsf{E}^{\simeq}_{A,B} : (A = B) \to (A \simeq B)$$

The univalence axiom now states that this map is an equivalence:

Axiom 1 (Univalence)**.** *The map $^{=}\mathsf{E}^{\simeq}_{A,B}$ is an equivalence for any $A, B : \mathcal{U}_i$.*

We will denote the quasi-inverse of $^{=}\mathsf{E}^{\simeq}_{A,B}$ by $^{\simeq}\mathsf{E}^{=}_{A,B}$.

It follows from univalence that *equivalent types are equal* and hence they satisfy the same properties:

Lemma 13. *For any type family $P : \mathcal{U}_i \to \mathcal{U}_j$, and types $A, B : \mathcal{U}_i$ with $A \simeq B$, we have that $P(A) \simeq P(B)$. Thus in particular, $P(A)$ is inhabited precisely when $P(B)$ is.*

Finally, two functions $f, g : \Pi_{x:A}B(x)$ should be equal precisely when there exists a homotopy between them. Constructing a map

$$^{=}\mathsf{E}^{\Pi}_{f,g} : (f = g) \to (f \sim g)$$

is easy. Showing that this map is an equivalence (or even constructing a map in the opposite direction) is much harder, and is in fact among the chief consequences of univalence:

Lemma 14. *The map* ${}^{=}\mathsf{E}^{\Pi}_{f,g}$ *is an equivalence for any* $f, g : \Pi_{x:A} B(x)$.

Proof. See Chpt. 4.9 of [23]. □

We will denote the quasi-inverse of ${}^{=}\mathsf{E}^{\Pi}_{f,g}$ by ${}^{\Pi}\mathsf{E}^{=}_{f,g}$.

3. Higher Inductive Types

In this section we describe some higher inductive types of interest and use these specific examples as an introduction to the terminology and methodology that will follow in Sect. 4.

An inductive type X can be understood as being *freely generated* by a collection of constructors: in the familiar case of natural numbers, we have the two constructors for zero and successor. The property of being freely generated can be stated as an induction principle: in order to show that a property $P : \mathbb{N} \to \mathcal{U}_i$ holds for all $n : \mathbb{N}$, it suffices to show that it holds for zero and is preserved by the successor operation. As a special case, we get the recursion principle: in order to define a map $f : \mathbb{N} \to C$, is suffices to determine its value at zero and its behavior with respect to successor.

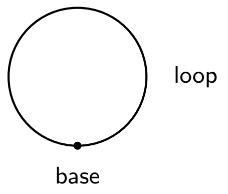
Higher inductive types generalize ordinary inductive types by allowing constructors involving *path spaces of* X rather than just X itself, as the next example shows.

3.1 The circle

The unit circle $\mathbf{S^1}$ is represented as an inductive type $\mathbb{S} : \mathcal{U}_0$ with two constructors [13]:

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

pictured as



This in particular means that we have further paths, such as $\mathsf{loop}^{-1} \cdot \mathsf{loop} \cdot \mathsf{loop} \cdot \mathsf{refl}_{\mathbb{S}}(\mathsf{base})$ (which is equal to loop).

We can reason about the circle using the principle of *circle recursion*, also called *simple elimination* for $\mathbb{S}$, which tells us that in order to construct a function out of $\mathbb{S}$ into a type C, it suffices to supply a point $c : C$ and a loop $s : c =_C c$.

$$\frac{C : \mathcal{U}_i \qquad c : C \qquad s : c =_C c}{\mathsf{rec}_{\mathbb{S}}(C, c, s) : \mathbb{S} \to C}$$

Furthermore, the recursor has the expected behavior on the 0-cell constructor base (we omit the premises):

$$\mathsf{rec}_{\mathbb{S}}(C, c, s)(\mathsf{base}) \equiv c : C$$

We also have a computation rule for the 1-cell constructor loop:

$$\mathsf{ap}_{\mathsf{rec}_{\mathbb{S}}(C,c,s)}(\mathsf{loop}) =_{\mathsf{Id}_C(c,c)} s$$

This rule type-checks by virtue of the previous one. We note that in order to record the effect of the recursor on the path loop, we use the "action-on-paths" construct ap. Since this is a derived notion rather than a primitive one, we state the rule as a propositional rather than definitional equality.

We also have the more general principle of *circle induction*, also called *dependent elimination* for $\mathbb{S}$, which subsumes recursion. Instead of a type $C : \mathcal{U}_i$ we now have a type family $E : \mathbb{S} \to \mathcal{U}_i$. Where previously we required a point $c : C$, we now need a point $e : E(\mathsf{base})$. Finally, an obvious generalization of needing a loop $s : c =_C c$ would be to ask for a loop $d : e =_{E(\mathsf{base})} e$. However, this would be incorrect: once we have our desired inductor of type $\Pi_{x:\mathbb{S}} E(x)$, its effect on loop is not a loop at e in the fiber $E(\mathsf{base})$ but a path from $\mathsf{loop}^E_*(e)$ to e in $E(\mathsf{base})$ (or its contravariant version). The induction principle thus takes the following form:

$$\frac{E : \mathbb{S} \to \mathcal{U}_i \qquad e : E(\mathsf{base}) \qquad d : \mathsf{loop}^E_*(e) =_{E(\mathsf{base})} e}{\mathsf{ind}_{\mathbb{S}}(E, e, d) : \Pi_{x:\mathbb{S}} E(x)}$$

We have the associated computation rules:

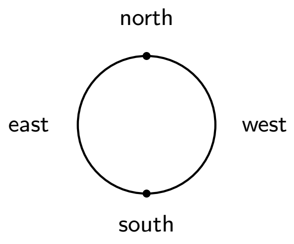
$$\mathsf{ind}_{\mathbb{S}}(E, e, d)(\mathsf{base}) \equiv e : E(\mathsf{base})$$
$$\mathsf{dap}_{\mathsf{ind}_{\mathbb{S}}(E,e,d)}(\mathsf{loop}) =_{\mathsf{Id}_{E(\mathsf{base})}(\mathsf{loop}^E_*(e),e)} d$$

3.2 The circle, round two

We could have alternatively represented the circle as an inductive type $\mathbb{S}_a : \mathcal{U}_0$ with four constructors:

$$\mathsf{north} : \mathbb{S}_a$$
$$\mathsf{south} : \mathbb{S}_a$$
$$\mathsf{east} : \mathsf{north} =_{\mathbb{S}_a} \mathsf{south}$$
$$\mathsf{west} : \mathsf{north} =_{\mathbb{S}_a} \mathsf{south}$$

pictured as



We now have the recursion principle

$$\frac{C : \mathcal{U}_i \qquad c : C \qquad d : C \qquad p : c =_C d \qquad q : c =_C d}{\mathsf{rec}_{\mathbb{S}_a}(C, c, d, p, q) : \mathbb{S}_a \to C}$$

with the computation rules

$$\mathsf{rec}_{\mathbb{S}_a}(C, c, d, p, q, \mathsf{north}) \equiv c : C$$
$$\mathsf{rec}_{\mathbb{S}_a}(C, c, d, p, q, \mathsf{south}) \equiv d : C$$

and

$$\mathsf{ap}_{\mathsf{rec}_{\mathbb{S}_a}(C,c,d,p,q)}(\mathsf{east}) = p$$
$$\mathsf{ap}_{\mathsf{rec}_{\mathbb{S}_a}(C,c,d,p,q)}(\mathsf{west}) = q$$

The corresponding induction principle is

$$\frac{\begin{array}{c} E : \mathbb{S}_a \to \mathcal{U}_i \qquad u : E(\mathsf{north}) \qquad v : E(\mathsf{south}) \\ \mu : \mathsf{east}^E_*(u) =_{E(\mathsf{south})} v \qquad \nu : \mathsf{west}^E_*(u) =_{E(\mathsf{south})} v \end{array}}{\mathsf{ind}_{\mathbb{S}_a}(E, u, v, \mu, \nu) : \Pi_{x:\mathbb{S}_a} E(x)}$$

with the associated computation rules

$$\mathsf{ind}_{\mathbb{S}_a}(E, u, v, \mu, \nu, \mathsf{north}) \equiv u : E(\mathsf{north})$$
$$\mathsf{ind}_{\mathbb{S}_a}(E, u, v, \mu, \nu, \mathsf{south}) \equiv v : E(\mathsf{south})$$

and

$$\mathsf{dap}_{\mathsf{ind}_{\mathbb{S}_a}(E,u,v,\mu,\nu)}(\mathsf{east}) = \mu$$
$$\mathsf{dap}_{\mathsf{ind}_{\mathbb{S}_a}(E,u,v,\mu,\nu)}(\mathsf{west}) = \nu$$

As expected, the two circle types are equivalent:

Lemma 15. *We have $\mathbb{S} \simeq \mathbb{S}_a$.*

Proof sketch. From left to right, map base to north and loop to $\mathsf{east} \cdot \mathsf{west}^{-1}$. From right to left, map both north and south to base, east to loop, and west to $\mathsf{refl}_{\mathbb{S}}(\mathsf{base})$. Using the respective induction principles, show that these two mappings compose to identity on both sides and apply Lem. 8. $\square$

3.3 Computation laws, revisited

By Lem. 15 the types $\mathbb{S}$ and $\mathbb{S}_a$ are equivalent and hence satisfy the same properties (see Lem. 13). We would thus expect the induction principle for $\mathbb{S}$ to carry over to $\mathbb{S}_a$, and vice versa. Indeed, with a little effort we can show the former:

Lemma 16. *The type $\mathbb{S}_a$ satisfies the induction and computation laws for $\mathbb{S}$, with* north *acting as the constructor* base *and* $\mathsf{east} \cdot \mathsf{west}^{-1}$ *acting as the constructor* loop.

In the other direction, though, we hit a snag - the only obvious choice we have is to define both points north and south to be base, one of the paths west and east to be loop, and the other one the identity path at base. This, however, does not give us the desired induction principle: unless the two given points $u : E(\mathsf{base})$ and $v : E(\mathsf{base})$ happen to be definitionally equal, we will not be able to map base to both of them, as required by the computation rules.

This poses more than just a conceptual problem - in mathematics, we often have several possible definitions of a given notion, all of which are interchangeable from the point of view of a "user". Having two definitions of a circle which are not (known to be) interchangeable, however, can be problematic: any theorem we establish about or by appealing to $\mathbb{S}_a$ might no longer hold - or even type-check! - when using $\mathbb{S}$ instead. To see this, take the second computation law for $\mathbb{S}_a$, $\mathsf{dap}_{\mathsf{ind}_{\mathbb{S}_a}(E,u,v,\mu,\nu)}(\mathsf{west}) = \nu$. If we attempt to "implement" $\mathbb{S}_a$ using the circle $\mathbb{S}$ instead - by taking $\mathsf{north}, \mathsf{south} :\equiv \mathsf{base}$, $\mathsf{east} :\equiv \mathsf{loop}$, $\mathsf{west} :\equiv \mathsf{refl}_{\mathbb{S}}(\mathsf{base})$ as in the proof of Lem. 15 - the computation law is no longer well-typed since the left-hand side reduces to a reflexivity path whereas the right hand side is a path from u to v.

This is one of the motivations for considering inductive types with *propositional* computation behavior: we now want to investigate types which "act like the circle" up to propositional equality. In the case of $\mathbb{S}$, such a type $C : \mathcal{U}_i$ should come with a point $b : C$ and loop $l : c =_C c$. In the case of $\mathbb{S}_a$, such a type should come with two points $n, s : C$ and two paths $e, w : n =_C s$. We can express this more concisely as follows:

Definition 17. *Define the type of* $\mathbb{S}$-algebras *on a universe $\mathcal{U}_i$ as*

$$\mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i} :\equiv \Sigma_{C:\mathcal{U}_i}\Sigma_{b:C}(b = b)$$

Definition 18. *Define the type of* $\mathbb{S}_a$-algebras *on a universe $\mathcal{U}_i$ as*

$$\mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i} :\equiv \Sigma_{C:\mathcal{U}_i}\Sigma_{n,s:C}(n = s) \times (n = s)$$

We are now interested in maps between algebras which in a suitable sense preserve the distinguished points and paths, i.e., *algebra homomorphisms*. A homomorphism between two $\mathbb{S}$-algebras (C, c, p) and (D, d, q) should be a function $f : C \to D$ for which we have a path $\beta : f(c) = d$. Furthermore, f should also appropriately relate p and q. To figure out what this means, we observe that if we map p along f, we obtain a path $\mathsf{ap}_f(p) : f(c) = f(c)$. Each of the (identical) endpoints is equal to d, via the path β. Thus, we now have another path $\beta^{-1} \cdot \mathsf{ap}_f(p) \cdot \beta : d = d$. It is reasonable to require that this path be equal to q, i.e., that the following diagram commutes:

$$\begin{tikzcd}
f(c) \arrow[r, dash, "\mathsf{ap}_f(p)"] \arrow[d, dash, "\beta"'] & f(c) \arrow[d, dash, "\beta"] \\
d \arrow[r, dash, "q"'] & d
\end{tikzcd}$$

Likewise, a homomorphism between two $\mathbb{S}_a$-algebras (C, a, b, p, q) and (D, c, d, r, s) should be a function $f : C \to D$ for which we have paths $\beta : f(a) = c, \gamma : f(b) = d$ and for which the following diagrams commute:

$$\begin{tikzcd}
f(a) \arrow[r, dash, "\mathsf{ap}_f(p)"] \arrow[d, dash, "\beta"'] & f(b) \arrow[d, dash, "\gamma"] \\
c \arrow[r, dash, "r"'] & d
\end{tikzcd}
\qquad
\begin{tikzcd}
f(a) \arrow[r, dash, "\mathsf{ap}_f(q)"] \arrow[d, dash, "\beta"'] & f(b) \arrow[d, dash, "\gamma"] \\
c \arrow[r, dash, "s"'] & d
\end{tikzcd}$$

In other words, an $\mathbb{S}$- or $\mathbb{S}_a$-homomorphism behaves just like a function constructed by the appropriate circle recursion, albeit with propositional computation laws for points and paths. We can express this as follows:

Definition 19. *For algebras $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$, $\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_j}$, define the type of* $\mathbb{S}$-homomorphisms *from $\mathcal{X}$ to $\mathcal{Y}$ by*

$$\begin{aligned}&\mathbb{S}\text{-}\mathsf{Hom}\ (C, c, p)\ (D, d, q) :\equiv \\ &\quad \Sigma_{f:C\to D}\Sigma_{\beta:f(c)=d}\big(\mathsf{ap}_f(p) \cdot \beta = \beta \cdot q\big)\end{aligned}$$

Definition 20. *For algebras $\mathcal{X} : \mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}$, $\mathcal{Y} : \mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_j}$, define the type of* $\mathbb{S}_a$-homomorphisms *from $\mathcal{X}$ to $\mathcal{Y}$ by*

$$\begin{aligned}&\mathbb{S}_a\text{-}\mathsf{Hom}\ (C, a, b, p, q)\ (D, c, d, r, s) :\equiv \Sigma_{f:C\to D}\Sigma_{\beta:f(a)=c} \\ &\quad \Sigma_{\gamma:f(b)=d}\big(\mathsf{ap}_f(p) \cdot \gamma = \beta \cdot r\big) \times \big(\mathsf{ap}_f(q) \cdot \gamma = \beta \cdot s\big)\end{aligned}$$

We note that to be able to form the type of homomorphisms as we just did, it is crucial to have the computation laws stated propositionally. The recursion principle now becomes a property internal to the type theory and can be expressed compactly as saying that there is a homomorphism into any other algebra $\mathcal{Y}$:

Definition 21. *An algebra $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$* satisfies the $\mathbb{S}$-recursion principle *on a universe $\mathcal{U}_j$ if for any algebra $\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_j}$ there exists a homomorphism from $\mathcal{X}$ to $\mathcal{Y}$:*

$$\mathsf{has}\text{-}\mathbb{S}\text{-}\mathsf{rec}_{\mathcal{U}_j}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_j}\big)\ \mathbb{S}\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y}$$

Definition 22. *An algebra $\mathcal{X} : \mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}$* satisfies the $\mathbb{S}_a$-recursion principle *on a universe $\mathcal{U}_j$ if for any algebra $\mathcal{Y} : \mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_j}$ there exists a homomorphism from $\mathcal{X}$ to $\mathcal{Y}$:*

$$\mathsf{has}\text{-}\mathbb{S}_a\text{-}\mathsf{rec}_{\mathcal{U}_j}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : \mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_j}\big)\ \mathbb{S}_a\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y}$$

To express the induction principle in a similar fashion, we first need to introduce dependent or *fibered* versions of algebras and algebra homomorphisms:

Definition 23. *Define the type of* fibered $\mathbb{S}$-algebras *on a universe $\mathcal{U}_j$ over an algebra $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$ by*

$$\mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_j}\ (C, c, p) :\equiv \Sigma_{E:C\to\mathcal{U}_j}\Sigma_{e:E(c)}\big(p_*^E(e) = e\big)$$

Definition 24. *Define the type of* fibered $\mathbb{S}_a$-algebras *on a universe $\mathcal{U}_j$ over an algebra $\mathcal{X} : \mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}$ by*

$$\begin{aligned}&\mathbb{S}_a\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_j}\ (C, c, d, p, q) :\equiv \\ &\quad \Sigma_{E:C\to\mathcal{U}_j}\Sigma_{u:E(c)}\Sigma_{v:E(d)}\big(p_*^E(u) = v\big) \times \big(q_*^E(u) = v\big)\end{aligned}$$

Definition 25. *For algebras* $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$, $\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_j}\ \mathcal{X}$, *define the type of* fibered $\mathbb{S}$-homomorphisms *from* $\mathcal{X}$ *to* $\mathcal{Y}$ *by*

$$\mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Hom}\ (C, c, p)\ (E, e, q) :\equiv$$
$$\Sigma_{f:(\Pi x:C)E(x)}\Sigma_{\beta:f(c)=e}\big(\mathsf{dap}_f(p) \cdot \beta = \mathsf{ap}_{p_*^E}(\beta) \cdot q\big)$$

Pictorially, the last component witnesses the commuting diagram

$$\begin{tikzcd}
p_*^E(f(c)) \arrow[r, dash, "\mathsf{dap}_f(p)"] \arrow[d, dash, "\mathsf{ap}_{p_*^E}(\beta)"'] & f(c) \arrow[d, dash, "\beta"] \\
p_*^E(e) \arrow[r, dash, "q"'] & e
\end{tikzcd}$$

Definition 26. *For algebras* $\mathcal{X} : \mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}$, $\mathcal{Y} : \mathbb{S}_a\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_j}\ \mathcal{X}$, *define the type of* fibered $\mathbb{S}_a$-homomorphisms *from* $\mathcal{X}$ *to* $\mathcal{Y}$ *by*

$$\mathbb{S}_a\text{-}\mathsf{Fib}\text{-}\mathsf{Hom}\ (C, a, b, p, q)\ (D, c, d, r, s) :\equiv \Sigma_{f:(\Pi x:C)E(x)}$$
$$\Sigma_{\beta:f(a)=c}\Sigma_{\gamma:f(b)=d}\big(\mathsf{dap}_f(p) \cdot \gamma = \mathsf{ap}_{p_*^E}(\beta) \cdot r\big) \times$$
$$\big(\mathsf{dap}_f(q) \cdot \gamma = \mathsf{ap}_{q_*^E}(\beta) \cdot s\big)$$

Pictorially, the last two components witness the commuting diagrams

$$\begin{tikzcd}
p_*^E(f(a)) \arrow[r, dash, "\mathsf{dap}_f(p)"] \arrow[d, dash, "\mathsf{ap}_{p_*^E}(\beta)"'] & f(b) \arrow[d, dash, "\gamma"] \\
p_*^E(c) \arrow[r, dash, "r"'] & d
\end{tikzcd}
\qquad
\begin{tikzcd}
p_*^E(f(a)) \arrow[r, dash, "\mathsf{dap}_f(q)"] \arrow[d, dash, "\mathsf{ap}_{p_*^E}(\beta)"'] & f(b) \arrow[d, dash, "\gamma"] \\
p_*^E(c) \arrow[r, dash, "s"'] & d
\end{tikzcd}$$

The induction principle can now be expressed as saying that there is a fibered homomorphism into any fibered algebra $\mathcal{Y}$:

Definition 27. *An algebra* $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$ satisfies the $\mathbb{S}$-induction principle *on a universe* $\mathcal{U}_j$ *if for any fibered algebra* $\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_j}\ \mathcal{X}$ *there exists a fibered homomorphism from* $\mathcal{X}$ *to* $\mathcal{Y}$*:*

$$\mathsf{has}\text{-}\mathbb{S}\text{-}\mathsf{ind}_{\mathcal{U}_j}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_j}\big)\ \mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y}$$

Definition 28. *An algebra* $\mathcal{X} : \mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}$ satisfies the $\mathbb{S}_a$-induction principle *on universe* $\mathcal{U}_j$ *if for any fibered algebra* $\mathcal{Y} : \mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_j}\ \mathcal{X}$ *there exists a fibered homomorphism from* $\mathcal{X}$ *to* $\mathcal{Y}$*:*

$$\mathsf{has}\text{-}\mathbb{S}_a\text{-}\mathsf{ind}_{\mathcal{U}_j}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : \mathbb{S}_a\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_j}\big)\ \mathbb{S}_a\text{-}\mathsf{Fib}\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y}$$

3.4 Relating the two circles

We first note that the notions of $\mathbb{S}$-algebras and $\mathbb{S}_a$-algebras are in fact the same:

Lemma 29. *We have a function*

$$\mathbb{S}\text{-}\mathsf{to}\text{-}\mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i} \to \mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}$$

which is an equivalence.

Proof. Define the equivalence between $\mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$ and $\mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}$ by the quasi-inverses

$$(C, c, p) \mapsto (C, c, c, p, \mathsf{refl}(c))$$
$$(C, a, b, p, q) \mapsto (C, a, p \cdot q^{-1})$$

□

Next, we note that the notions of fibered $\mathbb{S}$-algebras and fibered $\mathbb{S}_a$-algebras are the same, in the following sense:

Lemma 30. *For any algebra* $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$ *we have a function*

$$\mathbb{S}\text{-}\mathsf{to}\text{-}\mathbb{S}_a\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_i}(\mathcal{X}) :$$
$$\mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \mathcal{X} \to \mathbb{S}_a\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \big(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \mathcal{X}\big)$$

which is an equivalence.

Proof. Fix algebra $(C, c, p) : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$. Define the equivalence between $\mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ (C, c, p)$ and $\mathbb{S}_a\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ (C, c, c, p, \mathsf{refl}(c))$ by the quasi-inverses

$$(E, e, q) \mapsto (E, e, e, q, \mathsf{refl}(e))$$
$$(E, a, b, r, s) \mapsto (E, a, r \cdot s^{-1})$$

□

The notions of $\mathbb{S}$-homomorphisms and $\mathbb{S}_a$-homomorphisms also coincide:

Lemma 31. *For any algebras* $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$, $\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_j}$ *we have*

$$\mathbb{S}\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y} \simeq \mathbb{S}_a\text{-}\mathsf{Hom}\ \big(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \mathcal{X}\big)\ \big(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \mathcal{Y}\big)$$

Finally, the respective fibered versions of $\mathbb{S}$-homomorphisms and $\mathbb{S}_a$-homomorphisms coincide:

Lemma 32. *For any algebras* $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$, $\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_j}\ \mathcal{X}$ *we have*

$$\mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y} \simeq$$
$$\mathbb{S}_a\text{-}\mathsf{Fib}\text{-}\mathsf{Hom}\ \big(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \mathcal{X}\big)\ \big(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathbb{S}_a\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_i}(\mathcal{X})\ \mathcal{Y}\big)$$

We can now show that $\mathbb{S}$-recursion is the same as $\mathbb{S}_a$-recursion, and likewise for induction:

Lemma 33. *For any* $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$ *we have*

$$\mathsf{has}\text{-}\mathbb{S}\text{-}\mathsf{rec}_{\mathcal{U}_j}(\mathcal{X}) \simeq \mathsf{has}\text{-}\mathbb{S}_a\text{-}\mathsf{rec}_{\mathcal{U}_j}(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}(\mathcal{X}))$$
$$\mathsf{has}\text{-}\mathbb{S}\text{-}\mathsf{ind}_{\mathcal{U}_j}(\mathcal{X}) \simeq \mathsf{has}\text{-}\mathbb{S}_a\text{-}\mathsf{ind}_{\mathcal{U}_j}(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathbb{S}_a\text{-}\mathsf{Alg}_{\mathcal{U}_i}(\mathcal{X}))$$

Corollary 34. *The* $\mathbb{S}_a$*-algebra* $(\mathbb{S}, \mathsf{base}, \mathsf{base}, \mathsf{loop}, \mathsf{refl}_{\mathbb{S}}(\mathsf{base}))$ *satisfies the* $\mathbb{S}_a$*-induction principle on any universe* $\mathcal{U}_j$*.*

Corollary 35. *The* $\mathbb{S}$*-algebra* $(\mathbb{S}_a, \mathsf{north}, \mathsf{east} \cdot \mathsf{west}^{-1})$ *satisfies the* $\mathbb{S}$*-induction principle on any universe* $\mathcal{U}_j$*.*

3.5 Propositional truncation

Another example of a higher inductive type is the *propositional truncation* $||A|| : \mathcal{U}_i$ of a type $A : \mathcal{U}_i$, investigated in [1] in an extensional setting under the name *bracket types*. Intuitively, $||A||$ represents the "squashing" of A which makes all the elements in A equal. The need for such a type arises when we wish to hide information: having a term $a : A$ is very different from having a $b : ||A||$. In the latter case, we know that the *provable failure* of A to be inhabited, that is, a term of type $A \to \mathbf{0}$, would lead to a contradiction. However, we do not have a generic way of constructing an inhabitant of A.

Formally, we define $||A||$ as the higher inductive type generated by a constructor $|\cdot|$, which projects a given element of A down to $||A||$, and a truncation constructor, which states that $||A||$ is indeed a mere proposition[2]:

$$|\cdot| : A \to ||A||$$
$$\mathsf{sq} : \Pi_{x,y:||A||}(x =_{||A||} y)$$

As usual, the recursion principle states that given a structure of the same form, we have a function out of $||A||$ which preserves the constructors:

$$\frac{C : \mathcal{U}_j \qquad c : A \to C \qquad s : \Pi_{x,y:C}(x =_C y)}{\mathsf{rec}_{||A||}(C, c, s) : ||A|| \to C}$$

where for each $a : A$ we have

$$\mathsf{rec}_{||A||}(C, c, s, |a|) \equiv c(a) : C$$

[2] Hence the name *propositional* truncation; see Chpt. 6 of [23] for other kinds of truncation.

and for each $k, l : ||A||$ we have

$$\mathsf{ap}_{\mathsf{rec}_{||A||}(C,c,s)}(\mathsf{sq}(k,l)) = \\ s\big(\mathsf{rec}_{||A||}(C,c,s,k), \mathsf{rec}_{||A||}(C,c,s,l)\big)$$

We note that we are only able to eliminate into types which are themselves mere propositions. This together with Lem. 5 implies that the second computation law always holds. We have included it nonetheless to illustrate the general pattern.

To state the induction principle, we need to suitably generalize the last hypothesis. As before, we note that once the desired map $f : \Pi_{x:||A||}E(x)$ is constructed, it will give us a path from $\mathsf{sq}(k,l)^E_*(f(k))$ to $f(l)$ in $E(l)$ for any $k, l : ||A||$. Hence, E should already come equipped with such a family of paths - except, of course, we have no way of referring to $f(k)$ and $f(l)$ before f is constructed. Thus, we simply require that such a path exists for *all* points $u : E(k)$ and $v : E(l)$:

$$\frac{E : ||A|| \to \mathcal{U}_j \qquad e : \Pi_{a:A}E(|a|) \\ q : \Pi_{x,y:||A||}\Pi_{u:E(x)}\Pi_{v:E(y)}\big(\mathsf{sq}(x,y)^E_*(u) =_{E(y)} v\big)}{\mathsf{ind}_{||A||}(E,e,q) : \Pi_{x:||A||}E(x)}$$

where for each $a : A$ we have

$$\mathsf{ind}_{||A||}(E,e,q,|a|) \equiv e(a) : E(|a|)$$

and for each $k, l : ||A||$ we have

$$\mathsf{dap}_{\mathsf{ind}_{||A||}(E,e,q)}(\mathsf{sq}(k,l)) = \\ q\big(k, l, \mathsf{ind}_{||A||}(E,e,q,k), \mathsf{ind}_{||A||}(E,e,q,l)\big)$$

The second rule again turns out to always hold, as we will see shortly.

3.6 Revisiting the truncation computation laws

We point out that the truncation $||A||$ as defined has its share of unexpected behavior. For instance, as the type $\mathbb{N}$ of natural numbers is inhabited, it follows that $||\mathbb{N}|| = \mathbf{1}$. It is not obvious, however, how to turn $\mathbf{1}$ itself into a truncation of $\mathbb{N}$, since the first computation law ought to hold *definitionally*. More surprising yet is the observation by N. Kraus in [10] that there exists a map f such that $f \circ |\cdot| \equiv \mathsf{id}_{\mathbb{N}}$; this is another somewhat strange side effect of the definitional computation law for $|\cdot|$.

In light of these issues, we follow our methodology for circles and investigate types which "act like the type $||A||$" up to propositional equality. We have the following:

Definition 36. *Define the type of* $||A||$-algebras *on a universe* $\mathcal{U}_j$ *as*

$$||A||\text{-}\mathsf{Alg}_{\mathcal{U}_j} :\equiv \Sigma_{C:\mathcal{U}_j}(A \to C) \times \mathsf{is\text{-}prop}(C)$$

A natural definition of an algebra homomorphism between two $||A||$-algebras (C,c,p) and (D,d,q) is a map $f : C \to D$ together with path families

$$\beta : \Pi_{a:A}\big(f(c(a)) = d(a)\big) \\ \gamma : \Pi_{x,y:C}\big(\mathsf{ap}_f(p(x,y)) = q(f(x),f(y))\big)$$

However, the type D is a mere proposition. Thus by Lem. 5 and the fact that a family of contractible types is itself contractible, it follows that both of the above types are equivalent to $\mathbf{1}$. Hence, we have the simple definition:

Definition 37. *For algebras* $\mathcal{X} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_j}$, $\mathcal{Y} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_k}$, *define the type of* $||A||$-homomorphisms *from* $\mathcal{X}$ *to* $\mathcal{Y}$ *by*

$$||A||\text{-}\mathsf{Hom}\ (C,c,p)\ (D,d,q) :\equiv C \to D$$

As before, the recursion principle states that there is a homomorphism to any other algebra $\mathcal{Y}$:

Definition 38. *An algebra* $\mathcal{X} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_j}$ satisfies the recursion principle *on a universe* $\mathcal{U}_k$ *if for any algebra* $\mathcal{Y} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_k}$ *there exists a* $||A||$*-homomorphism from* $\mathcal{X}$ *to* $\mathcal{Y}$*:*

$$\mathsf{has}\text{-}||A||\text{-}\mathsf{rec}_{\mathcal{U}_k}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_k}\big)\ ||A||\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y}$$

Based on the $||A||$-elimination rule, a natural definition of a fibered algebra over (C,c,p) is a family of types $E : C \to \mathcal{U}_k$ endowed with a function $e : \Pi_{a:A}E(c(a))$ and path family

$$q : \Pi_{x,y:C}\Pi_{u:E(x)}\Pi_{v:E(y)}(p(x,y)^E_*(u) = v)$$

Using the fact that C is a mere proposition, we can show, however, that the above type is equivalent to the condition that E is a family of mere propositions, $\Pi_{x:C}\mathsf{is\text{-}prop}(E(x))$. We can thus define:

Definition 39. *Define the type of fibered* $||A||$*-algebras on a universe* $\mathcal{U}_k$ *over* $\mathcal{X} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_j}$ *by*

$$||A||\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_k}\ (C,c,p) :\equiv \\ \Sigma_{E:C\to\mathcal{U}_k}\big(\Pi_{a:A}E(c(a))\big) \times \big(\Pi_{x:C}\mathsf{is\text{-}prop}(E(x))\big)$$

Analogously to the non-fibered case, a natural definition of a fibered $||A||$-homomorphism from (C,c,p) to (E,e,q) is a function $f : \Pi_{x:C}E(x)$ together with path families

$$\beta : \Pi_{a:A}\big(f(c(a)) = e(a)\big) \\ \gamma : \Pi_{x,y:C}\big(\mathsf{dap}_f(p(x,y)) = q(x,y,f(x),f(y))\big)$$

Since E is a family of mere propositions, by exactly the same reasoning we get that both of the above types are equivalent to $\mathbf{1}$. Hence, we can define:

Definition 40. *For algebras* $\mathcal{X} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_j}$, $\mathcal{Y} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_k}\ \mathcal{X}$, *define the type of* fibered $||A||$-homomorphisms *from* $\mathcal{X}$ *to* $\mathcal{Y}$ *by*

$$||A||\text{-}\mathsf{Fib}\text{-}\mathsf{Hom}\ (C,c,p)\ (E,e,q) :\equiv \Pi_{x:C}E(x)$$

As before, the induction principle states that there is a fibered homomorphism to any fibered algebra $\mathcal{Y}$:

Definition 41. *An algebra* $\mathcal{X} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_j}$ satisfies the $||A||$-induction principle *on a universe* $\mathcal{U}_k$ *if for any fibered algebra* $\mathcal{Y} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_k}\ \mathcal{X}$ *there exists a fibered homomorphism from* $\mathcal{X}$ *to* $\mathcal{Y}$*:*

$$\mathsf{has}\text{-}||A||\text{-}\mathsf{ind}_{\mathcal{U}_k}(\mathcal{X}) :\equiv \\ \big(\Pi\mathcal{Y} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_k}\ \mathcal{X}\big)\ ||A||\text{-}\mathsf{Fib}\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y}$$

It is now easy to show that induction and recursion for $||A||$ are in fact equivalent. As we will see in the next section, this result is an analogue of our main theorem 53. The proof given below for truncations illustrates the basic idea behind the proof of Thm. 53.

We note that since universe levels are cumulative, the technical restriction that $k \geq j$ does not pose a problem.

Lemma 42. *For* $A : \mathcal{U}_i$*, the following conditions on an algebra* $\mathcal{X} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_j}$ *are equivalent:*

- *$\mathcal{X}$ satisfies the induction principle on the universe $\mathcal{U}_k$*
- *$\mathcal{X}$ satisfies the recursion principle on the universe $\mathcal{U}_k$*

for $k \geq j$*. In other words, we have*

$$\mathsf{has}\text{-}||A||\text{-}\mathsf{ind}_{\mathcal{U}_k}(\mathcal{X}) \simeq \mathsf{has}\text{-}||A||\text{-}\mathsf{rec}_{\mathcal{U}_k}(\mathcal{X})$$

provided $k \geq j$*. Moreover, the two types above are mere propositions.*

Proof. The fact that the types are mere propositions is clear. The direction from right to left is obvious. For the other direction, fix algebras $(C,c,p) : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_j}$, $(E,e,q) : ||A||\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_k}\ (C,c,p)$. The total space $\Sigma_{x:C}E(x) : \mathcal{U}_k$ is a mere proposition, we can thus apply recursion with the projection map $a \mapsto (c(a), e(a))$ to get a function $f : C \to \Sigma_{x:C} E(x)$. A homotopy $\alpha : \mathsf{fst} \circ f \sim \mathsf{id}_C$ exists as C is a mere proposition. Applying second projection and transporting gives us a map $x \mapsto \alpha(x)_*^E(\pi_2(f(x)))$ and we are done. $\square$

4. W-Suspensions as Homotopy-Initial Algebras

Here we consider a class of higher inductive types which we call W-*suspensions*; informally, they combine Martin-Löf's W-types [18], also known as well-founded trees, with (a generalized form of) suspensions ([23], Chpt. 6.5). Ordinary W-types allow proper induction on the level of points but have no higher-dimensional constructors. Suspensions, on the other hand, only provide vacuous induction on the point level, in the form of two nullary constructors; however, they allow us to specify an arbitrary number of path constructors between these two points. A suitable combination of these two classes of types keeps the phenomenons of induction and higher-dimensionality orthogonal, which gives us a well-behaved elimination principle.

4.1 W-suspensions

Formally, given types $A, C : \mathcal{U}_i$, a type family $B : A \to \mathcal{U}_i$, and functions $\mathtt{l}, \mathtt{r} : C \to A$, the W-suspension $\mathsf{W}(A, B, C, \mathtt{l}, \mathtt{r}) : \mathcal{U}_i$ is the higher inductive type generated by the constructors

$$\begin{aligned} &\mathsf{sup} : \Pi_{a:A}\big(B(a) \to \mathsf{W}(A,B,C,\mathtt{l},\mathtt{r})\big) \to \mathsf{W}(A,B,C,\mathtt{l},\mathtt{r}) \\ &\mathsf{cell} : \Pi_{c:C}\Pi_{t:B(\mathtt{l}\ c)\to\mathsf{W}(A,B,C,\mathtt{l},\mathtt{r})}\Pi_{s:B(\mathtt{r}\ c)\to\mathsf{W}(A,B,C,\mathtt{l},\mathtt{r})} \\ &\qquad \mathsf{sup}(\mathtt{l}\ c, t) = \mathsf{sup}(\mathtt{r}\ c, s) \end{aligned}$$

From now on we will write W instead of $\mathsf{W}(A, B, C, \mathtt{l}, \mathtt{r})$, unless indicated otherwise.

As in the case of ordinary W-types, the type A can be thought of as the type of labels for points and for any $a : A$, the type $B(a)$ represents the arity of the label a, i.e., it is the index type for the arguments of a. Similarly, the type C represents the type of labels for paths between points. For any $c : C$, the terms $\mathtt{l}(c)$ and $\mathtt{r}(c)$ determine the respective labels of the left and right endpoints of the paths labeled by c. As can be read off from the type of the constructor cell, each label $c : C$ determines a family of paths in W, one for each pair of terms $t : B(\mathtt{l}\ c) \to \mathsf{W}$ and $s : B(\mathtt{r}\ c) \to \mathsf{W}$.

An ordinary W-type $\mathsf{W}_{x:A}B(x)$ arises as a W-suspension in the obvious way by taking $A :\equiv A$, $B :\equiv B$, $C :\equiv \mathbf{0}$, and letting both $\mathtt{l}$ and $\mathtt{r}$ be the canonical function from $\mathbf{0}$ into A. We can encode the circle $\mathbb{S}$ by taking $A, C :\equiv \mathbf{1}$, $B(-) :\equiv \mathbf{0}$, $\mathtt{l}(-) :\equiv \star$, $\mathtt{r}(-) :\equiv \star$. The circle $\mathbb{S}_a$ arises when we take $A, C :\equiv \mathbf{2}$, $B(-) :\equiv \mathbf{0}$, $\mathtt{l}(-) :\equiv \top$, $\mathtt{r}(-) :\equiv \bot$. Other types which can be represented in this form include the interval type, suspensions - hence in particular all the higher spheres $\mathbf{S}^n$ - and of course all ordinary inductive types arising as W-types, e.g., natural numbers, lists, and so on.

As an additional example we consider positive integers modulo two. Let $\mathbf{4}$ be the inductive type with constructors $\mathtt{tt}, \mathtt{tf}, \mathtt{ft}, \mathtt{ff} : \mathbf{4}$. We put $A :\equiv \mathbf{4}$; $B(\mathtt{tt}) :\equiv \mathbf{0}$, $B(\mathtt{ff}) :\equiv \mathbf{0}$, $B(\mathtt{tf}) :\equiv \mathbf{1}$, $B(\mathtt{ft}) :\equiv \mathbf{1}$; $C :\equiv \mathbf{2}$; $\mathtt{l}(\top) :\equiv \mathtt{tt}$, $\mathtt{l}(\bot) :\equiv \mathtt{ff}$; $\mathtt{r}(\top) :\equiv \mathtt{tf}$, $\mathtt{r}(\bot) :\equiv \mathtt{ft}$. The nullary point labels $\mathtt{tt}$ and $\mathtt{ff}$ encode the positive integers one and two, respectively. The unary point label $\mathtt{tf}$ represents the function $n \mapsto 2n + 1$ and the unary point label $\mathtt{ft}$ represents the function $n \mapsto 2(n + 1)$. The path label $\top$ represents equations of the form $(\mathtt{tt}, -) = (\mathtt{tf}, -)$, to equate all odd positive integers. The path label $\bot$ represents equations of the form $(\mathtt{ff}, -) = (\mathtt{ft}, -)$, to equate all even positive integers.

W-suspensions come with the expected recursion principle: Given terms

- $E : \mathcal{U}_j$
- $e : \Pi_{a:A}(B(a) \to E) \to E$
- $q : \Pi_{c:C}\Pi_{u:B(\mathtt{l}\ c)\to E}\Pi_{v:B(\mathtt{r}\ c)\to E}\big(e(\mathtt{l}\ c, u) = e(\mathtt{r}\ c, v)\big)$

there is a recursor $\mathsf{rec}_{\mathsf{W}}(E, e, q) : \mathsf{W} \to E$. The recursor satisfies the computation laws

- $\mathsf{rec}_{\mathsf{W}}(E, e, q, \mathsf{sup}(a, t)) \equiv e\big(a, \mathsf{rec}_{\mathsf{W}}(E, e, q) \circ t\big)$

for any $a : A, t : B(a) \to \mathsf{W}$ and

- $\mathsf{ap}_{\mathsf{rec}_{\mathsf{W}}(E,e,q)}(\mathsf{cell}(c, t, s)) =$ $q\Big(c, \mathsf{rec}_{\mathsf{W}}(E, e, q) \circ t, \mathsf{rec}_{\mathsf{W}}(E, e, q) \circ s\Big)$

for any $c : C, t : B(\mathtt{l}\ c) \to \mathsf{W}, s : B(\mathtt{r}\ c) \to \mathsf{W}$. Similarly, we have an induction principle: Given terms

- $E : \mathsf{W} \to \mathcal{U}_j$
- $e : \Pi_{a:A}\Pi_{t:B(a)\to\mathsf{W}}\big(\Pi_{b:B(a)}E(t\ b)\big) \to E(\mathsf{sup}(a, t))$
- $q : \Pi_{c:C}\Pi_{t:B(\mathtt{l}\ c)\to\mathsf{W}}\Pi_{u:(\Pi b:B(\mathtt{l}\ c))E(t\ b)}\Pi_{s:B(\mathtt{r}\ c)\to\mathsf{W}}$ $\Pi_{v:(\Pi b:B(\mathtt{r}\ c))E(s\ b)}\big(\mathsf{cell}(c, t, s)_*^E\ e(\mathtt{l}\ c, t, u) = e(\mathtt{r}\ c, s, v)\big)$

there is an inductor $\mathsf{ind}_{\mathsf{W}}(E, e, q) : \Pi_{w:\mathsf{W}}E(w)$. The inductor satisfies the computation laws

- $\mathsf{ind}_{\mathsf{W}}(E, e, q, \mathsf{sup}(a, t)) \equiv e\big(a, t, \mathsf{ind}_{\mathsf{W}}(E, e, q) \circ t\big)$

for any $a : A, t : B(a) \to \mathsf{W}$ and

- $\mathsf{dap}_{\mathsf{ind}_{\mathsf{W}}(E,e,q)}(\mathsf{cell}(c, t, s)) =$ $q\Big(c, t, \mathsf{ind}_{\mathsf{W}}(E, e, q) \circ t, s, \mathsf{ind}_{\mathsf{W}}(E, e, q) \circ s\Big)$

for any $c : C, t : B(\mathtt{l}\ c) \to \mathsf{W}, s : B(\mathtt{r}\ c) \to \mathsf{W}$.

Following the now-familiar pattern, we define W-suspension algebras and homomorphisms:

Definition 43. *Define the type of* W-algebras *on a universe* $\mathcal{U}_j$ *by*

$$\mathsf{W\text{-}Alg}_{\mathcal{U}_j}(A, B, C, \mathtt{l}, \mathtt{r}) :\equiv \textit{type of triples } (D, d, p)$$

where

- $D : \mathcal{U}_j$
- $d : \Pi_{a:A}(B(a) \to D) \to D$
- $p : \Pi_{c:C}\Pi_{u:B(\mathtt{l}\ c)\to D}\Pi_{v:B(\mathtt{r}\ c)\to D}\big(d(\mathtt{l}\ c, u) = d(\mathtt{r}\ c, v)\big)$

As before, we will keep the parameters $A, B, C, \mathtt{l}, \mathtt{r}$ implicit if no confusion arises in doing so.

Definition 44. *Define the type of* fibered W-algebras *on a universe* $\mathcal{U}_k$ *over* $\mathcal{X} : \mathsf{W\text{-}Alg}_{\mathcal{U}_j}$ *by*

$$\mathsf{W\text{-}Fib\text{-}Alg}_{\mathcal{U}_k}\ (D, d, p) :\equiv \textit{type of triples } (E, e, q)$$

where

- $E : D \to \mathcal{U}_k$
- $e : \Pi_{a:A}\Pi_{t:B(a)\to D}\big(\Pi_{b:B(a)}E(t\ b)\big) \to E(d(a, t))$
- $q : \Pi_{c:C}\Pi_{t:B(\mathtt{l}\ c)\to D}\Pi_{s:B(\mathtt{r}\ c)\to D}\Pi_{u:(\Pi b:B(\mathtt{l}\ c))E(t\ b)}$ $\Pi_{v:(\Pi b:B(\mathtt{r}\ c))E(s\ b)}\Big(p(c, t, s)_*^E\ e(\mathtt{l}\ c, t, u) = e(\mathtt{r}\ c, s, v)\Big)$

In order to express the type of homomorphisms between two W-algebras, we again need to use propositional instead of definitional equality.

Definition 45. *For* $\mathcal{X} : \mathsf{W\text{-}Alg}_{\mathcal{U}_j}$ *and* $\mathcal{Y} : \mathsf{W\text{-}Alg}_{\mathcal{U}_k}$, *define the type of* W-homomorphisms *from* $\mathcal{X}$ *to* $\mathcal{Y}$ *by*

$$\mathsf{W\text{-}Hom}\ (D, d, p)\ (E, e, q) :\equiv \textit{type of triples } (f, \beta, \theta)$$

where

- $f : D \to E$
- $\beta : \Pi_{a:A}\Pi_{t:B(a)\to D}\big(f(d(a, t)) = e(a, f \circ t)\big)$

- $\theta : \Pi_{c:C}\Pi_{t:B(\mathtt{l}\ c)\to D}\Pi_{s:B(\mathtt{r}\ c)\to D}\Big(\mathsf{ap}_f(p(c,t,s)) = \beta(\mathtt{l}\ c,t) \cdot q(c, f\circ t, f\circ s) \cdot \beta(\mathtt{r}\ c,s)^{-1}\Big)$

Pictorially, $\theta(c,t,s)$ witnesses the commuting diagram:

$$\begin{tikzcd}
f(d(\mathtt{l}\ c,t)) \arrow[r, dash, "\mathsf{ap}_f(p_{c,t,s})"] \arrow[d, dash, "\beta_{\mathtt{l}(c),t}"] & f(d(\mathtt{r}\ c,s)) \arrow[d, dash, "\beta_{\mathtt{r}(c),s}"'] \\
e(\mathtt{l}\ c, f\circ t) \arrow[r, dash, "q_{c,f\circ t,f\circ s}"'] & e(\mathtt{r}\ c, f\circ s)
\end{tikzcd}$$

Definition 46. *For $\mathcal{X}$: W-$\mathsf{Alg}_{\mathcal{U}_j}$ and $\mathcal{Y}$: W-Fib-$\mathsf{Alg}_{\mathcal{U}_k}$ $\mathcal{X}$, define the type of fibered* W-homomorphisms *from $\mathcal{X}$ to $\mathcal{Y}$ by*

$$\mathsf{W\text{-}Fib\text{-}Hom}\ (D,d,p)\ (E,e,q) :\equiv \textit{type of triples } (f,\beta,\theta)$$

where

- $f : \Pi_{x:D}E(x)$
- $\beta : \Pi_{a:A}\Pi_{t:B(a)\to D}\big(f(d(a,t)) = e(a,t,f\circ t)\big)$
- $\theta : \Pi_{c:C}\Pi_{t:B(\mathtt{l}\ c)\to D}\Pi_{s:B(\mathtt{r}\ c)\to D}\Big(\mathsf{dap}_f(p(c,t,s)) = \mathsf{ap}_{p(c,t,s)^E_*}(\beta(\mathtt{l}\ c,t)) \cdot q(c,t,s,f\circ t, f\circ s) \cdot \beta(\mathtt{r}\ c,s)^{-1}\Big)$

Pictorially, $\theta(c,t,s)$ witnesses the commuting diagram:

$$\begin{tikzcd}
(p_{c,t,s})^E_*(f(d(\mathtt{l}\ c,t))) \arrow[r, dash, "\mathsf{dap}_f(p_{c,t,s})"] \arrow[d, dash, "\mathsf{ap}_{p(c,t,s)^E_*}(\beta_{\mathtt{l}(c),t})"] & f(d(\mathtt{r}\ c,s)) \arrow[d, dash, "\beta_{\mathtt{r}(c),s}"'] \\
(p_{c,t,s})^E_*(e(\mathtt{l}\ c,t,f\circ t)) \arrow[r, dash, "q_{c,t,s,f\circ t,f\circ s}"'] & e(\mathtt{r}\ c,s,f\circ s)
\end{tikzcd}$$

The recursion and induction principles are defined as usual:

Definition 47. *An algebra $\mathcal{X}$: W-$\mathsf{Alg}_{\mathcal{U}_j}$* satisfies the recursion principle *on a universe $\mathcal{U}_k$ if for any algebra $\mathcal{Y}$: W-$\mathsf{Alg}_{\mathcal{U}_k}$ there exists a* W*-homomorphism from $\mathcal{X}$ to $\mathcal{Y}$:*

$$\mathsf{has\text{-}W\text{-}rec}_{\mathcal{U}_k}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : \mathsf{W\text{-}Alg}_{\mathcal{U}_k}\big)\ \mathsf{W\text{-}Hom}\ \mathcal{X}\ \mathcal{Y}$$

Definition 48. *An algebra $\mathcal{X}$: W-$\mathsf{Alg}_{\mathcal{U}_j}$* satisfies the induction principle *on a universe $\mathcal{U}_k$ if for any fibered algebra $\mathcal{Y}$: W-$\mathsf{Alg}_{\mathcal{U}_k}$ $\mathcal{X}$ there exists a fibered* W*-homomorphism from $\mathcal{X}$ to $\mathcal{Y}$:*

$$\mathsf{has\text{-}W\text{-}ind}_{\mathcal{U}_k}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : \mathsf{W\text{-}Fib\text{-}Alg}_{\mathcal{U}_k}\ \mathcal{X}\big)\ \mathsf{W\text{-}Fib\text{-}Hom}\ \mathcal{X}\ \mathcal{Y}$$

We will also need the following uniqueness properties which state that any two (fibered) homomorphisms into any (fibered) algebra $\mathcal{Y}$ are equal:

Definition 49. *An algebra $\mathcal{X}$: W-$\mathsf{Alg}_{\mathcal{U}_j}$* satisfies the recursion uniqueness principle *on a universe $\mathcal{U}_k$ if for any other algebra $\mathcal{Y}$: W-$\mathsf{Alg}_{\mathcal{U}_k}$ the type of* W*-homomorphisms from $\mathcal{X}$ to $\mathcal{Y}$ is a mere proposition:*

$$\mathsf{has\text{-}W\text{-}rec\text{-}uniq}_{\mathcal{U}_k}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : \mathsf{W\text{-}Alg}_{\mathcal{U}_k}\big)\ \mathsf{is\text{-}prop}(\mathsf{W\text{-}Hom}\ \mathcal{X}\ \mathcal{Y})$$

Definition 50. *An algebra $\mathcal{X}$: W-$\mathsf{Alg}_{\mathcal{U}_j}$* satisfies the induction uniqueness principle *on a universe $\mathcal{U}_k$ if for any fibered algebra $\mathcal{Y}$: W-$\mathsf{Alg}_{\mathcal{U}_k}$ $\mathcal{X}$ the type of* W*-homomorphisms from $\mathcal{X}$ to $\mathcal{Y}$ is a mere proposition:*

$$\mathsf{has\text{-}W\text{-}ind\text{-}uniq}_{\mathcal{U}_k}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : \mathsf{W\text{-}Fib\text{-}Alg}_{\mathcal{U}_k}\ \mathcal{X}\big)\ \mathsf{is\text{-}prop}(\mathsf{W\text{-}Fib\text{-}Hom}\ \mathcal{X}\ \mathcal{Y})$$

We now define the key concept of homotopy-initiality [3], which translates the notion of existence plus uniqueness into the homotopy type-theoretic setting as contractibility:

Definition 51. *An algebra $\mathcal{X}$: W-$\mathsf{Alg}_{\mathcal{U}_j}$ is* homotopy-initial *on a universe $\mathcal{U}_k$ if for any other algebra $\mathcal{Y}$: W-$\mathsf{Alg}_{\mathcal{U}_k}$ the type of* W*-homomorphisms from $\mathcal{X}$ to $\mathcal{Y}$ is contractible:*

$$\mathsf{is\text{-}W\text{-}hinit}_{\mathcal{U}_k}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : \mathsf{W\text{-}Alg}_{\mathcal{U}_k}\big)\ \mathsf{is\text{-}contr}(\mathsf{W\text{-}Hom}\ \mathcal{X}\ \mathcal{Y})$$

Lemma 52. *For any $\mathcal{X}$: W-$\mathsf{Alg}_{\mathcal{U}_j}$ we have*

$$\mathsf{is\text{-}W\text{-}hinit}_{\mathcal{U}_k}(\mathcal{X}) \simeq \mathsf{has\text{-}W\text{-}rec}_{\mathcal{U}_k}(\mathcal{X}) \times \mathsf{has\text{-}W\text{-}rec\text{-}uniq}_{\mathcal{U}_k}(\mathcal{X})$$

4.2 Main result

Our main result establishes the equivalence between the universal property of being homotopy-initial and the satisfaction of the induction principle:

Theorem 53. *For $A, C : \mathcal{U}_i$, $B : A \to \mathcal{U}_i$, $\mathtt{l}, \mathtt{r} : C \to A$, the following conditions on an algebra $\mathcal{X}$: W-$\mathsf{Alg}_{\mathcal{U}_j}(A,B,C,\mathtt{l},\mathtt{r})$ are equivalent:*

- *$\mathcal{X}$ satisfies the induction principle on the universe $\mathcal{U}_k$*
- *$\mathcal{X}$ is homotopy-initial on the universe $\mathcal{U}_k$*

for $k \geq j$. In other words, we have

$$\mathsf{has\text{-}W\text{-}ind}_{\mathcal{U}_k}(\mathcal{X}) \simeq \mathsf{is\text{-}W\text{-}hinit}_{\mathcal{U}_k}(\mathcal{X})$$

provided $k \geq j$. Moreover, the two types above are mere propositions.

By Lem. 52, homotopy-intiality is equivalent to the principles of recursion plus recursion uniqueness. The uniqueness condition is necessary since in general, the recursion principle does not fully determine an inductive type: the recursion principle for the circle, for example, is also satisfied by the disjoint union of *two* circles.

Before we proceed to the proof of the general case, we look at the analogue of the main theorem for propositional truncations. We can define homotopy-initial $||A||$-algebras as expected:

Definition 54. *An algebra $\mathcal{X}$: $||A||$-$\mathsf{Alg}_{\mathcal{U}_j}$ is* homotopy-initial *on a universe $\mathcal{U}_k$ if for any other algebra $\mathcal{Y}$: $||A||$-$\mathsf{Alg}_{\mathcal{U}_k}$ the type of $||A||$-homomorphisms from $\mathcal{X}$ to $\mathcal{Y}$ is contractible:*

$$\mathsf{is}\text{-}||A||\text{-}\mathsf{hinit}_{\mathcal{U}_k}(\mathcal{X}) :\equiv \big(\Pi\mathcal{Y} : ||A||\text{-}\mathsf{Alg}_{\mathcal{U}_k}\big)\ \mathsf{is\text{-}contr}(||A||\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y})$$

Since we operate in the setting of mere propositions, we do not have to formulate an analogous uniqueness condition, which is uniquely satisfied and hence redundant. Instead, we have:

Lemma 55. *For any $\mathcal{X}$: $||A||$-$\mathsf{Alg}_{\mathcal{U}_j}$ we have*

$$\mathsf{is}\text{-}||A||\text{-}\mathsf{hinit}_{\mathcal{U}_k}(\mathcal{X}) \simeq \mathsf{has}\text{-}||A||\text{-}\mathsf{rec}_{\mathcal{U}_k}(\mathcal{X})$$

Thus, Lem. 42 is the analogue of our main result for truncations.

Proof outline A crucial step of the proof is the characterization of the path space $\mu = \nu$ between two (fibered) W-homomorphisms $\mu, \nu : \mathcal{X} \to \mathcal{Y}$ in a more explicit form. For simplicity we only consider the non-fibered case here. We recall that a homomorphism between two algebras (D,d,p), (E,e,q) is a triple (f,β,θ), where $f : C \to D$ is a function between the carrier types, β specifies the behavior of f on the 0-cells, i.e., the value of $f(d(a,t))$, and θ specifies the behavior of f on the 1-cells, i.e., the value of $\mathsf{ap}_f(p(c,t,s))$.

Using the characterization of paths between tuples together with function extensionality, the path space $(f,\beta,\theta) = (g,\gamma,\phi)$ between two homomorphisms should be equivalent to a type of triples (α,η,ψ), where $\alpha : f \sim g$ is a homotopy relating the two underlying mappings, and η, ψ relate β to γ resp. θ to ϕ in an appropriate way. We will call such a triple (α, η, ψ) a W-*cell*. The recursion uniqueness condition on an algebra $\mathcal{X}$ can then be equivalently expressed as saying that for any algebra $\mathcal{Y}$ and homomorphisms μ, ν from $\mathcal{X}$ to $\mathcal{Y}$, there exists a W-cell between μ and ν.

We point out that this uniqueness condition can itself be understood as a certain form of induction, albeit a very specific one. The existence of a W-cell between any two homomorphisms (f, β, θ), (g, γ, ϕ) in particular guarantees the existence of a dependent function $\alpha : \Pi_{x:X}(f(x) = g(x))$ - the "inductor". The behavior of α on the 0-cells, i.e., the value of $\alpha(d(a,t))$, is specified by the term η, which thus serves as a witness for the first "computation rule". Finally, the behavior of α on the 1-cells, i.e., the value of $\mathsf{dap}_\alpha(p(c,t,s))$, is specified by the term ψ, which hence serves as a witness for the second "computation rule." We observe the same pattern in the case of propositional truncations: homomorphisms between $||A||$-algebras are just maps between the carrier types, hence there are no "computation rules" to speak of. A cell between $||A||$-homomorphisms f and g would just be a homotopy $\alpha : \Pi_{x:X}(f(x) = g(x))$ - the "inductor". The existence of such α is of course a moot point since we work with mere propositions.

We can thus see why the full induction principle for W-suspensions gives us homotopy-initiality: the latter essentially amounts to the recursion principle plus a specific form of induction, both of which are implied by the general induction principle. The hardest part of the proof is showing the converse, i.e., that the general induction principle can be recovered from homotopy-initiality.

We are now ready to give the formal definition of a W-cell. There is an analogous definition of a *fibered* W-*cell*, which uses the fibered versions of W-algebras and homomorphisms, and for which we refer the reader to [21].

Definition 56. *For algebras* $\mathcal{X} : \mathsf{W\text{-}Alg}_{\mathcal{U}_j}$, $\mathcal{Y} : \mathsf{W\text{-}Alg}_{\mathcal{U}_k}$ *and homomorphisms* $\mu, \nu : \mathsf{W\text{-}Hom}\ \mathcal{X}\ \mathcal{Y}$, *we define the type of* W-cells *between μ and ν by*

$$\mathsf{W\text{-}Cell}\ (D,d,p)\ (E,e,q)\ (f,\beta,\theta)\ (g,\gamma,\phi) :=$$
$$\textit{type of triples } (\alpha, \eta, \psi)$$

where

- $\alpha : f \sim g$
- $\eta : \Pi_{a:A}\Pi_{t:B(a)\to D}\Big(\alpha(d(a,t)) =$ $\beta(a,t) \cdot \mathsf{ap}_{e(a)}({}^{\Pi}\mathsf{E}^{=}(\alpha \circ t)) \cdot \gamma(a,t)^{-1}\Big)$
- ψ *asserts the commutativity of the diagram in Fig. 1 for any* $c : C$, $t : B(\mathtt{l}\ c) \to D$, $s : B(\mathtt{r}\ c) \to D$.

Pictorially, $\eta(a,t)$ witnesses the commuting diagram:

$$\begin{tikzcd}
f(d(a,t)) \arrow[r, dash, "\alpha(d(a,t))"] \arrow[d, dash, "\beta(a,t)"'] & g(d(a,t)) \arrow[d, dash, "\gamma(a,t)"] \\
e(a, f\circ t) \arrow[r, dash, "\mathsf{ap}_{e(a)}({}^{\Pi}\mathsf{E}^{=}(\alpha\circ t))"'] & e(a, g\circ t)
\end{tikzcd}$$

We will often leave the algebra arguments to a W-cell implicit.

Remark: Although the diagram in Fig. 1 does not explicitly mention the term $\mathsf{dap}_\alpha(p(c,t,s))$, it nevertheless specifies its value uniquely since this term is expressible using $\mathsf{nat}(\alpha, p(c,t,s))$.

Lemma 57. *For algebras* $\mathcal{X} : \mathsf{W\text{-}Alg}_{\mathcal{U}_j}$, $\mathcal{Y} : \mathsf{W\text{-}Alg}_{\mathcal{U}_k}$ *and homomorphisms* $\mu, \nu : \mathsf{W\text{-}Hom}\ \mathcal{X}\ \mathcal{Y}$, *the path space $\mu = \nu$ is equivalent to the type of* W*-cells between μ and ν:*

$$\mu = \nu\ \simeq\ \mathsf{W\text{-}Cell}\ \mu\ \nu$$

The proof of the main result now consists of the following steps:

1) *Show that the induction principle implies the recursion principle, that is for any* $\mathcal{X} : \mathsf{W\text{-}Alg}_{\mathcal{U}_j}$ *we have*
$$\mathsf{has\text{-}W\text{-}ind}_{\mathcal{U}_k}(\mathcal{X}) \to \mathsf{has\text{-}W\text{-}rec}_{\mathcal{U}_k}(\mathcal{X})$$
2) *Show that the induction principle implies both uniqueness conditions, that is for any* $\mathcal{X} : \mathsf{W\text{-}Alg}_{\mathcal{U}_j}$ *we have*:
$$\mathsf{has\text{-}W\text{-}ind}_{\mathcal{U}_k}(\mathcal{X}) \to \mathsf{has\text{-}W\text{-}ind\text{-}uniq}_{\mathcal{U}_k}(\mathcal{X})$$
$$\mathsf{has\text{-}W\text{-}ind}_{\mathcal{U}_k}(\mathcal{X}) \to \mathsf{has\text{-}W\text{-}rec\text{-}uniq}_{\mathcal{U}_k}(\mathcal{X})$$
3) *Show that the recursion plus recursion uniqueness principles imply the induction principle, that is for any* $\mathcal{X} : \mathsf{W\text{-}Alg}_{\mathcal{U}_j}$ *we have*:
$$\mathsf{has\text{-}W\text{-}rec}_{\mathcal{U}_k}(\mathcal{X}) \times \mathsf{has\text{-}W\text{-}rec\text{-}uniq}_{\mathcal{U}_k}(\mathcal{X}) \to$$
$$\mathsf{has\text{-}W\text{-}ind}_{\mathcal{U}_k}(\mathcal{X})$$

Using Lem. 52 we thus obtain a logical equivalence between $\mathsf{has\text{-}W\text{-}ind}_{\mathcal{U}_k}(\mathcal{X})$ and $\mathsf{is\text{-}W\text{-}hinit}_{\mathcal{U}_k}(\mathcal{X})$. It remains to show that both of these types are mere propositions. The latter is a mere proposition by Lem. 6. To show that $\mathsf{has\text{-}W\text{-}ind}_{\mathcal{U}_k}(\mathcal{X})$ is a mere proposition, it is sufficient to do so under the assumption that it is inhabited. Since $\mathcal{X}$ satisfies the induction principle, by the second step it satisfies the induction uniqueness principle. This means that for any fibered algebra $\mathcal{Y}$, the type $\mathsf{W\text{-}Fib\text{-}Hom}\ \mathcal{X}\ \mathcal{Y}$ is a mere proposition. Since a family of mere propositions is itself a mere proposition, this finishes the proof.

We now sketch the proof of the crucial part, step three. Fix an algebra $(D,d,p) : \mathsf{W\text{-}Alg}_{\mathcal{U}_j}$ and assume that $\mathsf{has\text{-}W\text{-}rec}_{\mathcal{U}_k}(D,d,p)$ and $\mathsf{has\text{-}W\text{-}rec\text{-}uniq}_{\mathcal{U}_k}(D,d,p)$ hold. Fix any algebra (E,e,q) fibered over (D,d,p). In order to apply the recursion principle, we need to turn this into a non-fibered algebra (E', e', q'). The first component is easy: the only reasonable choice we have is to put $E' := \Sigma_{x:D}E(x)$; we note that since $D : \mathcal{U}_j$, $E : D \to \mathcal{U}_k$, and $j \le k$, we indeed have $\Sigma_{x:D}E(x) : \mathcal{U}_k$ as needed. For the second component, we put $e'(a,u) := \Big(d(a, \pi_1 \circ u), e\big(a, \pi_1\circ u, \pi_2 \circ u\big)\Big)$. The last component we omit for brevity. The recursion principle then gives us a homomorphism (f, β, θ), where in particular $f : D \to \Sigma_{x:D}E(x)$. We next show that the function $\pi_1 \circ f$ is in fact the identity on D (up to a homotopy). We can do this by endowing both of the functions $\pi_1 \circ f$ and id_D with a homomorphism structure on the algebra (D,d,p); by the recursion uniqueness principle it then follows that these homomorphisms are equal, and in particular they are equal as maps. Thus we have a homotopy $\alpha : \pi_1 \circ f \sim \mathsf{id}_D$ and the underlying map of our desired fibered homomorphism can be defined as $x \mapsto \alpha(x)^E_*(\pi_2(f(x)))$. Endowing this map with a homomorphism structure involves a significant amount of higher path manipulations.

To illustrate some of the issues that arise when dealing with higher paths, we show the proof of step one. Fix an algebra $(D,d,p) : \mathsf{W\text{-}Alg}_{\mathcal{U}_j}$ and assume that $\mathsf{has\text{-}W\text{-}ind}_{\mathcal{U}_k}(D,d,p)$ holds. Fix any other algebra (E,e,q). In order to apply the induction principle, we need to turn this into a fibered algebra (E', e', q'). The first two components are easy: put $E'(x) := E$ and $e'(a,t,u) := e(a,u)$. For the last component, we note that the transport between any two fibers of a constant type family is constant. We can thus define $q'(c,t,s,u,v)$ to be the path in Fig. 2 a). The induction principle then gives us a map $f : D \to E$, a path family $\beta : \Pi_{a:C}\Pi_{t:B(a)\to D}\big(f(d(a,t)) = e(a, f\circ t)\big)$, and for any $c : C$,

$$\begin{tikzcd}
\alpha(d(\mathtt{l}\ c,t)) \cdot \mathsf{ap}_g(p_{c,t,s}) \arrow[r, dash, "\mathsf{nat}(\alpha,\, p(c,t,s))"] \arrow[d, dash, "via\ \eta(\mathtt{l}\ c,t)"'] & \mathsf{ap}_f(p_{c,t,s}) \cdot \alpha(d(\mathtt{r}\ c,s)) \arrow[d, dash, "via\ \eta(\mathtt{r}\ c,s)"] \\
\Big(\beta_{\mathtt{l}(c),t} \cdot \mathsf{ap}_{e(\mathtt{l}\ c)}({}^{\Pi}\mathsf{E}^{=}(\alpha \circ t)) \cdot \gamma^{-1}_{\mathtt{l}(c),t}\Big) \cdot \mathsf{ap}_g(p_{c,t,s}) \arrow[d, dash] & \mathsf{ap}_f(p_{c,t,s}) \cdot \Big(\beta(\mathtt{r}\ c,s) \cdot \mathsf{ap}_{e(\mathtt{r}\ c)}({}^{\Pi}\mathsf{E}^{=}(\alpha \circ s)) \cdot \gamma^{-1}_{\mathtt{r}(c),s}\Big) \arrow[d, dash] \\
\Big(\beta_{\mathtt{l}(c),t} \cdot \mathsf{ap}_{e(\mathtt{l}\ c)}({}^{\Pi}\mathsf{E}^{=}(\alpha \circ t))\Big) \cdot \Big(\gamma^{-1}_{\mathtt{l}(c),t} \cdot \mathsf{ap}_g(p_{c,t,s})\Big) \arrow[d, dash, "via\ \mathbf{I}^2_{\square}(\phi(c,t,s))"'] & \Big(\mathsf{ap}_f(p_{c,t,s}) \cdot \beta(\mathtt{r}\ c,s)\Big) \cdot \Big(\mathsf{ap}_{e(\mathtt{r}\ c)}({}^{\Pi}\mathsf{E}^{=}(\alpha \circ s)) \cdot \gamma^{-1}_{\mathtt{r}(c),s}\Big) \arrow[d, dash, "via\ \mathbf{I}^1_{\square}(\theta(c,t,s))"] \\
\Big(\beta_{\mathtt{l}(c),t} \cdot \mathsf{ap}_{e(\mathtt{l}\ c)}({}^{\Pi}\mathsf{E}^{=}(\alpha \circ t))\Big) \cdot \Big(q_{c,g\circ t,g\circ s} \cdot \gamma^{-1}_{\mathtt{r}(c),s}\Big) \arrow[d, dash] & \Big(\beta_{\mathtt{l}(c),t} \cdot q_{c,f\circ t,f\circ s}\Big) \cdot \Big(\mathsf{ap}_{e(\mathtt{r}\ c)}({}^{\Pi}\mathsf{E}^{=}(\alpha \circ s)) \cdot \gamma^{-1}_{\mathtt{r}(c),s}\Big) \arrow[d, dash] \\
\beta_{\mathtt{l}(c),t} \cdot \Big(\mathsf{ap}_{e(\mathtt{l}\ c)}({}^{\Pi}\mathsf{E}^{=}(\alpha \circ t)) \cdot q_{c,g\circ t,g\circ s}\Big) \cdot \gamma^{-1}_{\mathtt{r}(c),s} \arrow[r, dash, bend right, "via\ \mathsf{nat}_{\mathcal{H}}\big(q(c),\, {}^{\Pi}\mathsf{E}^{=}(\alpha \circ t),\, {}^{\Pi}\mathsf{E}^{=}(\alpha \circ s)\big)^{-1}"'] & \beta_{\mathtt{l}(c),t} \cdot \Big(q_{c,f\circ t,f\circ s} \cdot \mathsf{ap}_{e(\mathtt{r}\ c)}({}^{\Pi}\mathsf{E}^{=}(\alpha \circ s))\Big) \cdot \gamma^{-1}_{\mathtt{r}(c),s}
\end{tikzcd}$$

Figure 1. Diagram for the definition of a W-cell.

$$\begin{tikzcd}
(p_{c,t,s})^{-\mapsto Y}_{*}(e(\mathtt{l}\ c,u)) \arrow[d, dash] & (p_{c,t,s})^{-\mapsto Y}_{*}(f(d(\mathtt{l}\ c,t))) \arrow[d, dash] \\
e(\mathtt{l}\ c,u) \arrow[d, dash, "q_{c,u,v}"] & f(d(\mathtt{l}\ c,t)) \arrow[d, dash, "\mathsf{ap}_f(p_{c,t,s})"] \\
e(\mathtt{r}\ c,v) & f(d(\mathtt{r}\ c,s)) \\
\textbf{a)} & \textbf{b)}
\end{tikzcd}$$

Figure 2.

$$\begin{tikzcd}
(p_{c,t,s})^{-\mapsto Y}_{*}(f(d(\mathtt{l}\ c,t))) \arrow[rr, dash, bend left, "\mathsf{ap}_{p(c,t,s)^{-\mapsto Y}_{*}}(\beta(\mathtt{l}\ c,t))"] \arrow[dd, dash, "\mathsf{dap}_f(p_{c,t,s})"'] & & (p_{c,t,s})^{-\mapsto Y}_{*}(e(\mathtt{l}\ c,f\circ t)) \arrow[d, dash] \\
& \theta(c,t,s) & e(\mathtt{l}\ c,f\circ t) \arrow[d, dash, "q_{c,f\circ t,f\circ s}"] \\
f(d(\mathtt{r}\ c,s)) \arrow[rr, dash, "\beta(\mathtt{r}\ c,s)"'] & & e(\mathtt{r}\ c,f\circ s)
\end{tikzcd}$$

Figure 3.

$$\begin{tikzcd}
(p_{c,t,s})^{-\mapsto Y}_{*}(f(d(\mathtt{l}\ c,t))) \arrow[r, dash, bend left, "\mathsf{ap}_{p(c,t,s)^{-\mapsto Y}_{*}}(\beta(\mathtt{l}\ c,t))"] \arrow[d, dash] \arrow[dr, phantom, "A"] & (p_{c,t,s})^{-\mapsto Y}_{*}(e(\mathtt{l}\ c,f\circ t)) \arrow[d, dash] \\
f(d(\mathtt{l}\ c,t)) \arrow[r, dash, "\beta(\mathtt{l}\ c,s)"'] \arrow[d, dash, "\mathsf{ap}_f(p_{c,t,s})"'] \arrow[dr, phantom, "B"] & e(\mathtt{l}\ c,f\circ t) \arrow[d, dash, "q_{c,f\circ t,f\circ s}"] \\
f(d(\mathtt{r}\ c,s)) \arrow[r, dash, "\beta(\mathtt{r}\ c,s)"'] & e(\mathtt{r}\ c,f\circ s)
\end{tikzcd}$$

Figure 4.

$t : B(\mathtt{l}\ c) \to D$, $s : B(\mathtt{r}\ c) \to D$ a witness $\theta(c,t,s)$ for the commutativity of the diagram in Fig. 3. Using path induction we can express $\mathsf{dap}_f(p_{c,t,s})$ equivalently as the path in Fig. 2 b). Thus the outer rectangle in the diagram in Fig. 4 commutes. Suitable path induction shows that rectangle *A* commutes; hence rectangle *B* commutes too and we are done.

4.3 Definability

We now show how to derive the analogue of our main result for the circle $\mathbb{S}$; the cases of $\mathbb{S}_a$ and other inductive types presentable as W-suspensions follow the same methodology. In the rest of this section we work with a specific W-suspension $\mathsf{W}(A,B,C,\mathtt{l},\mathtt{r}) : \mathcal{U}_0$ where $A, C :\equiv \mathbf{1}$, $B(-) :\equiv \mathbf{0}$, $\mathtt{l}(-) :\equiv \star$, $\mathtt{r}(-) :\equiv \star$.

Definition 58. *An algebra $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_j}$ is* homotopy-initial *on a universe $\mathcal{U}_k$ if for any other algebra $\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_k}$ the type of $\mathbb{S}$-homomorphisms from $\mathcal{X}$ to $\mathcal{Y}$ is contractible:*

$$\mathsf{is}\text{-}\mathbb{S}\text{-}\mathsf{hinit}_{\mathcal{U}_k}(\mathcal{X}) :\equiv \big(\Pi \mathcal{Y} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_k}\big)\ \mathsf{is}\text{-}\mathsf{contr}(\mathbb{S}\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y})$$

Lemma 59. *We have a function*

$$\mathbb{S}\text{-}\mathsf{to}\text{-}\mathsf{W}\text{-}\mathsf{Alg}_{\mathcal{U}_i} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i} \to \mathsf{W}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$$

which is an equivalence.

Proof. We define the function by the mapping

$$(D,d,p) \mapsto \big(D, \lambda_a \lambda_t d, \lambda_c \lambda_t \lambda_s p\big)$$

It is not hard to see that this is an equivalence. ☐

Lemma 60. *For any algebra $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$ we have a function*

$$\mathbb{S}\text{-}\mathsf{to}\text{-}\mathsf{W}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_i}(\mathcal{X}) : \\ \mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \mathcal{X} \to \mathsf{W}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \big(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathsf{W}\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \mathcal{X}\big)$$

which is an equivalence.

Proof. Fix an algebra $(D,d,p) : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$. We define the function by the mapping

$$(E,e,q) \mapsto \big(E, \lambda_a \lambda_t \lambda_u e, \lambda_c \lambda_t \lambda_u \lambda_s \lambda_v q\big)$$

It is not hard to see that this is an equivalence. ☐

Lemma 61. *For any algebras* $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$, $\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_j}$ *we have*

$$\mathbb{S}\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y} \simeq \mathsf{W}\text{-}\mathsf{Hom}\ \left(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathsf{W}\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \mathcal{X}\right)\ \left(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathsf{W}\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \mathcal{Y}\right)$$

Lemma 62. *For any algebras* $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$, $\mathcal{Y} : \mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_j}\ \mathcal{X}$ *we have*

$$\begin{aligned}&\mathbb{S}\text{-}\mathsf{Fib}\text{-}\mathsf{Hom}\ \mathcal{X}\ \mathcal{Y} \simeq \\ &\quad \mathsf{W}\text{-}\mathsf{Fib}\text{-}\mathsf{Hom}\ \left(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathsf{W}\text{-}\mathsf{Alg}_{\mathcal{U}_i}\ \mathcal{X}\right)\ \left(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathsf{W}\text{-}\mathsf{Fib}\text{-}\mathsf{Alg}_{\mathcal{U}_i}(\mathcal{X})\ \mathcal{Y}\right)\end{aligned}$$

Lemma 63. *For any* $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$ *we have*

$$\begin{aligned}\mathsf{has}\text{-}\mathbb{S}\text{-}\mathsf{rec}_{\mathcal{U}_j}(\mathcal{X}) &\simeq \mathsf{has}\text{-}\mathsf{W}\text{-}\mathsf{rec}_{\mathcal{U}_j}(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathsf{W}\text{-}\mathsf{Alg}_{\mathcal{U}_i}(\mathcal{X}))\\ \mathsf{has}\text{-}\mathbb{S}\text{-}\mathsf{ind}_{\mathcal{U}_j}(\mathcal{X}) &\simeq \mathsf{has}\text{-}\mathsf{W}\text{-}\mathsf{ind}_{\mathcal{U}_j}(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathsf{W}\text{-}\mathsf{Alg}_{\mathcal{U}_i}(\mathcal{X}))\\ \mathsf{is}\text{-}\mathbb{S}\text{-}\mathsf{hinit}_{\mathcal{U}_j}(\mathcal{X}) &\simeq \mathsf{is}\text{-}\mathsf{W}\text{-}\mathsf{hinit}_{\mathcal{U}_j}(\mathbb{S}\text{-}\mathsf{to}\text{-}\mathsf{W}\text{-}\mathsf{Alg}_{\mathcal{U}_i}(\mathcal{X}))\end{aligned}$$

Corollary 64. *For an algebra* $\mathcal{X} : \mathbb{S}\text{-}\mathsf{Alg}_{\mathcal{U}_i}$, *the following conditions are equivalent:*

- $\mathcal{X}$ *satisfies the induction principle on the universe* $\mathcal{U}_j$
- $\mathcal{X}$ *is homotopy-initial on the universe* $\mathcal{U}_j$

In other words, we have

$$\mathsf{has}\text{-}\mathbb{S}\text{-}\mathsf{ind}_{\mathcal{U}_j}(\mathcal{X}) \simeq \mathsf{is}\text{-}\mathbb{S}\text{-}\mathsf{hinit}_{\mathcal{U}_j}(\mathcal{X})$$

Moreover, the two types above are mere propositions.

Proof. We use Lem. 63 and 53. □

5. Conclusion

We have investigated higher inductive types with propositional computational behavior and shown that they can be equivalently characterized as homotopy-initial algebras. We have stated and proved this result for propositional truncations and for the so-called W-suspensions, which subsume a number of other interesting cases - ordinary W-types, the unit circle $\mathbf{S}^1$, the interval type $\mathbf{I}$, all suspensions, and thus all the higher spheres $\mathbf{S}^n$. The characterization of these individual types as homotopy-initial algebras can be easily obtained as a corollary to our main theorem. Furthermore, we can readily apply the method presented here to obtain an analogous result for set truncations and set quotients. We conjecture that similar results can be established for other higher inductive types - such as homotopy (co)limits, tori, group quotients, or real numbers - following the same methodology. We are planning to formalize the results presented here in the Coq proof assistant.

Finally, we remark that the entire field of Homotopy Type Theory is a subject of intense research and many questions pertaining to higher inductive types and univalence are yet to be satisfactorily answered. Two important open problems are finding a unifying schema for general higher inductive types (see [16] for progress towards this goal) and developing a computational interpretation of HoTT (partially answered by the cubical set model [4]).

Acknowledgments

Thanks to Profs. Steve Awodey and Frank Pfenning. Support for this research was provided by the Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) through the Carnegie Mellon Portugal Program, as well as by the National Science Foundation under award No. NSF-CCF-DMS 1351344. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

References

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

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

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

[4] M. Bezem, T. Coquand, and S. Huber. A model of type theory in cubical sets, 2014. Available at `http://www.cse.chalmers.se/~coquand/mod1.pdf`.

[5] Coq Development Team. *The Coq Proof Assistant Reference Manual, version 8.4pl3.* INRIA, 2012. Available at `coq.inria.fr`.

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

[7] R. Garner. Two-dimensional models of type theory. *Mathematical Structures in Computer Science*, 19(4):687–736, 2009.

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

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

[10] N. Kraus. The truncation map $|\_| : \mathbb{N} \to ||\mathbb{N}||$ is nearly invertible, 2013. Post on the Homotopy Type Theory blog.

[11] D. Licata and G. Brunerie. $\pi_n(\mathbf{S}^n)$ in Homotopy Type Theory. In *Certified Programs and Proofs*, volume 8307 of *LNCS*, pages 1–16. Springer, 2013.

[12] D. Licata and R. Harper. Canonicity for 2-dimensional type theory. In *Principles of Programming Languages (POPL 2012)*, pages 337–348. ACM, 2012.

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

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

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

[16] P. Lumsdaine and M. Shulman. Semantics of higher inductive types, 2012. Available at `http://uf-ias-2012.wikispaces.com/file/view/semantics.pdf`.

[17] P. Martin-Löf. An intuitionistic theory of types: Predicative part. In *Logic Colloquium 1973*, pages 73–118. North-Holland, 1975.

[18] P. Martin-Löf. Constructive mathematics and computer programming. In *Logic, Methodology, and Philosophy of Science*, pages 153–175. North-Holland, 1982.

[19] U. Norell. *Towards a practical programming language based on dependent type theory.* PhD thesis, Chalmers University of Technology, 2007.

[20] M. Shulman. Homotopy Type Theory, VI, 2011. Post on the n-category cafe blog.

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

[22] T. Streicher. Investigations into intensional type theory. Habilitation Thesis. Available from the authors web page, 1993.

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

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

[25] B. van den Berg and R. Garner. Topological and simplicial models of identity types. *ACM TOCL*, 13(1), 2012.

[26] V. Voevodsky. Univalent foundations of mathematics, 2011. Invited talk at the Workshop on Logic, Language, Information and Computation (WoLLIC 2011).

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