Eckmann-Hilton and the Hopf Fibration in Homotopy Type Theory

Raymond Baker

Department of Mathematics

University of Colorado Boulder

2023/04/06

Thesis Advisor: Professor Jonathan Wise, Department of Mathematics

Honors Council Representative: Professor Nathaniel Thiem, Department of Mathematics

Outside Reader: Professor Raul Saucedo, Department of Philosophy

Contents

0.1 Introduction

This thesis explores the connection between the Hopf fibration and the Eckmann-Hilton argument through the lens of Homotopy Type Theory (HoTT). Both the Hopf fibration and the Eckmann-Hilton argument are familiar constructions from classical homotopy theory. The Hopf fibration is a non-trivial map $\mathbb{S}^3 \to \mathbb{S}^2$ whose fiber is $\mathbb{S}^1$. The fiber sequence (and induced long exact sequence of homotopy groups) of the Hopf fibration gives an equivalence $\Omega^n(\mathbb{S}^2) \cong \Omega^n(\mathbb{S}^3)$, for $n \geq 3$. This result implies, in particular, that $\pi_3(\mathbb{S}^2) \cong \mathbb{Z}$. This is a surprising result. Given that $\mathbb{S}^2$ is generated by a single two dimensional loop, why should $\mathbb{S}^2$ have any non-trivial loops at dimension three, much less have the same loop space as $\mathbb{S}^3$ from there on out?

The Eckmann-Hilton argument provides an answer to this question. The Eckmann-Hilton argument can be phrased in different ways. A simple way to phrase it is simply that a monoid object in the category of monoids is necessarily a commutative monoid. However, a more apt phrasing is the following: for any space X and $n \geq 2$, the concatenation of loops in $\Omega^n(X)$ is commutative (up to homotopy of paths). Taking X to be $\mathbb{S}^2$ and n equal to 2, Eckmann-Hilton tells us that the concatenation of loops in $\Omega^2(\mathbb{S}^2)$ is commutative. But this commutativity only holds up to homotopy. The homotopy itself is a three dimensional path and this three dimensional path lends a generator of $\Omega^3(\mathbb{S}^3)$.

This is a result known to hold classically. Proofs of this claim, however, are hard to come by. And discussion of the claim usually involves talk of higher groups, braided groups, and the free braided group on one generator. The goal of this thesis is to write out an elementary proof of the above claim in the language of HoTT. The proof will be elementary in the sense that it makes no reference to braided groups or the like. In theory, this proof will be accesible to any undergraduate familiar with the basics of HoTT: Martin-Löf Type Theory (MLTT), Univalence, Higher Inductive Types (HITs), and some basic background theory. To make this thesis self contained, I will introduce each of these concepts and all prerequisite

background theory before turning to the Hopf fibration and Eckmann-Hilton.

The thesis is broken up into two parts. Part I serves as a limited introduction to the basics of HoTT. Part II of the thesis culminates in Part III which contains the desired proof.

Part I

Background Theory

In this part of the thesis we begin to develop some of the requisite theory for Part II. Included in this development are many results that serve to build intuition for how the types and the type formers encapsulate both higher categorical concepts and logical concepts. We will make explicit note of these results as they appear.

Chapter 1

The Types

We begin by looking at each of the type formers that we discussed and proving some lemmas about them. As mentioned, these lemmas will be necessary for Part III. But these lemmas also serve as a guide for our intuition about the types.

1.1 Crash Course on Types

In this section we give a very brief introduction to what a type is, how different types are defined, and how to work with them. Type theory, like set theory, is a formal system meant to encode the constructions of ordinary mathematics. The basic objects of type theory are called types. A type can be thought of as a collection. But a type differs from a set in a few ways. We will not go into a full discussion on the difference between types and sets. Rather we will just outline how to work with types in Homotopy Type Theory. For a proper introduction to type theory, the reader should consult [1], which we will from here on out refer to as "the HoTT Book".

A type can be thought of as collecting together a type of mathematical object (in fact this is the namesake). Its members are called terms or points. A simple example of a type

is $\mathbb{N}$, the type of natural numbers. A point in $\mathbb{N}$ is exactly a numeral. But we will also have occasion to work with abstract types whose points may not have a familiar description.

In contrast to set theory, a type is not presented by a formulae that defines its terms. Instead, a type is always presented by a universal property. For example, $\mathbb{N}$ is presented by a universal property that asserts that it is the type freely generated by a point $0 : \mathbb{N}$ and a function $\mathsf{succ} : \mathbb{N} \to \mathbb{N}$. Any "primitive type" like $\mathbb{N}$ will always be presented this way. This universal property tells us how we can map into and out of $\mathbb{N}$. Since $\mathbb{N}$ is freely generated by 0 and $\succ$, to construct a map $\mathbb{N} \to X$, we need to only specify where 0 goes and where $\succ(n)$ goes, given that we know where n goes. This is just famliar recursion on $\mathbb{N}$.

Of course we are also interested in bulding new types out of old types. These types too are defined by universal properties. For example, given two type A and B, we will the type $A \times B$ of pair of terms from A and B. This type is defined as the categorical product: maps into $A \times B$ are equivalent to pairs of maps into A and into B. In the following sections, we will review some of the fundamental types and constructions on types.

1.2 Path Types

Given an arbitrary type $A : U$, there is not much one can say about A. In fact, one of the only things we can say about A in absence of more details is that A comes equipped with a family of identity (which we are calling path) types. For any two points a and a' in A (written $a, a' : A$), we can form a new type denoted $a = a'$. We want to think of this new type as the type of equalties between a and a'. One of the central ideas of Homotopy Type Theory is that this type also behaves like the type of paths in a space from a to a'.

The universal property of the family of paths types is that it is freely generated by points $1_a : a = a$, which we call the reflexivity or trivial paths. This tells us that to define a map out of $a = a'$ for each a and $a' : A$, we only have to specify where each 1_a goes. This is going

to be called path induction. As it turns out, this is actually endows A with a hefty amount of structure. The identity types of A endow A with the structure of a globular ∞-groupoid. For $x, y : A$, the type $x = y$ behaves like the type of paths (or isomorphisms) from x to y. Unfortunately, the terminology seems to have gravitated towards topological: usually it is said the the type $x = y$ behaves like the type of paths from x to y, even thought call this the type of isomorphism is much closer to the truth. But, in sticking with convention, we too will refer to terms $p : x = y$ as paths from $x = y$ and may even refer to the type $x = y$ as the type of paths from x to y.

We cannot state internally in HoTT that the paths types behave like an ∞-groupoid (though it is an area of research). But we can flesh out as much structure as we like.

1.2.1 Groupoidal Operations on Paths

If the types $x = y$ behave like isomorpisms in an ∞-groupoind, then they should have composition and inverse operations. Our goal in this subsection is to define these operators. As a brief aside, we introduce some notation for function. For two types X and Y, we will denote the type of function from $X \to Y$ by $X \to Y$. Additionally, we have a notion of a family of types over X. This can be thought of as an assignment of a type $B(x)$ to each point $x : X$. We have already seen an example of this: for any type A and point $a : A$, we have the family of identity types, which assigns to each point $x : A$ the type $a = x$. We also have the family that assigns to each point $a : A$ the type $a = a$. We will want to talk about "maping into the family". For example, we want to be able to express in the type theory that, for each $a : A$, we have a term of type $a = a$ (namely 1_a). For a family B over some type X, we will denote the type of "maps into the type family" by $\prod_{x:X} B(x)$. For now, we can think of this as a universal quantifier. A term of this type gives an assignment of a point in $B(x)$ for all $x : X$.

Now we return to defining groupoidal operations on the path types.

Definition 1.2.1 (Composition of Paths, $(-)\cdot(-)$)**.** *We have a term of type*

$$(-)\cdot(-) : \prod_{x,yz:A} x = y \to y = z \to x = z$$

Proof. Let $x, y, z : A$ and $p : x = y$ and $q : y = z$. We need to construct a term $p \cdot q : x = z$. We can do this by path induction on $x, y, p : x = y$. That is, it suffices to specify where the reflexivity path 1_x gets mapped to. So our goal becomes $1_x \cdot q : x = z$. We again use path induction on q so that our goal further reduces to $1_x \cdot 1_x : x = x$. Thus we can define $1_x \cdot 1_x :\equiv q$. □

There are multiple ways to define $(-)\cdot(-)$. We could have used path induction on only p and defined $1_x \cdot q \equiv q$. Though these functions have different reduction behavior, they themselves will be equal. We can prove this fact once we discuss function extensionality. The reason for our choice of definition is simply to keep in line with the conventions of [1], which we will call the HoTT book from here on out.

If we think of terms $p : x = y$ as identifications or proofs of equality, then preceeding definition can, in this light, be thought of as a lemma establshing that equality is transitive.

Of course a groupoid is not a groupoid without inverses.

Definition 1 (Inverse of a path, $(-)^{-1}$)**.** *We want to define a function* $(-)^{-1} : \prod_{x,y:A} x = y \to y = x$.

Let $x, y : A$ *and* $p : x = y$*. Our goal is to construct a term* $p^{-1} : y = x$*. By path induction on* p*, it suffices to assume* $y \equiv x$ *and* $p \equiv 1_x$*. Then our goal becomes* $1_x^{-1} : x = x$*. Thus we may set* $1_x^{-1} \equiv 1_x$.$\diamond$

Under the proposition as types paradigm, this proves that equality is symmetric. Since we have reflexivity for free (witnessed by $1_{(-)} : \prod_{x:A} x = x$), these definitions serve to show that the paths types are an appropriate notion of equality.

1.2.2 Groupoidal Identities

But the path types, and the operators we have defined on them, do more than serve as mere witnesses to equality; they encode groupoidal structure. To demonstrate this, we show that the operators satisfy appropriate groupoidal identites. Before proving these identites, we should note that In the land of 1-groupoids, the identites take the form of *properties*. That is, in a one groupoid, the identity has the property that $1 \cdot p \equiv p$. But in HoTT, and higher groupoids more generally, the identites take the form of additional paths; the identities are additional stuff (see, Stuff, Structure, and Properties). Thus, we do not have that $1_x \cdot p \equiv p$. Instead we will have a path $\mathsf{l\text{-}unit}_p : 1_x \cdot p = p$.

Lemma 0.1 (Unit Laws, $\mathsf{l\text{-}unit}_p$, $\mathsf{r\text{-}unit}_p$)**.** *For the left unit law, we have a term* $\mathsf{l\text{-}unit}_{(-)} : \prod_{x,y:A} \prod_{p:x=y} 1_x \cdot p = p$. *For the right unit law, we have term* $\mathsf{r\text{-}unit}_{(-)} : \prod_{x,y:A} \prod_{p:x=y} p \cdot 1_y = p$.

Proof. Both identites can be proven by path induction. We only prove the $\mathsf{l\text{-}unit}$ identity, since $\mathsf{r\text{-}unit}$ can be defined similarly. By path inductionon p, our goal reduces to constructing a term $\mathsf{l\text{-}unit}_{1_x} : 1_x \cdot 1_x = 1_x$. But we have $1_x \cdot 1_x \equiv 1_x$, so we define $\mathsf{l\text{-}unit}_{1_x} \equiv 1_x^2$. □

Under the propositions as types paragdim, this theorem shows that 1_x behaves like the identity. Under the homotopical interpretation, we can think of this as a definition of a homotopy $\mathsf{l\text{-}unit} : (\lambda p.1_x \cdot p) \sim \mathsf{id}_{x=y}$. This perspective will be important later on.

Now we move on to the inverse laws.

Lemma 1.2.1 (Inverse Laws, $\mathsf{l\text{-}inv}_p$, $\mathsf{r\text{-}inv}_p$)**.** *For* $p : x = y$*, we have terms* $\mathsf{l\text{-}inv}_p : p^{-1} \cdot p = 1_y$ *and* $\mathsf{r\text{-}inv}_p : p \cdot p^{-1} = 1_x$.

We leave the proof to the reader.

1.3 Type Families

In this section we will take a brief look at type famlies, which we have only briefly mentioned. First we need to introduce the universe of types U. The universe is itself a type whose point are (small) types. Due to size issues, we can never have one universe to rule them all, since a universe cannot contain itself. However, we will not encounter size issues in this thesis, so we will not discuss them.

For a fixed type $A : U$, a type family over A is a function of the form $B : A \to U$. Thus, a type family is an assignment of a type $B(a)$ to each point $a : A$. But, as we shall see, B also carries with it a hefty amount of higher dimensional structure induced by the paths in A. As we will see, type families behave like fiber bundles over A. In more categorical language, a type family $B : A \to U$ behaves like a higher (co)presheaf on A: it assigns to each point $a : A$ an ∞-groupoid $B(a)$ and to each path $p : a = a'$ an equivalence $B(a) \simeq B(a')$. We will briefly outline some important properties of a type family.

First we discuss the structure in a type family induced by paths in A. This induced structure takes the form of an equivalence called "transport", denoted tr^B.

Theorem 1.3.1 (transport , tr^B)**.** *Given points a and a' in A and a path $p : a = a'$, we have a canonical function $\mathsf{tr}^B(p) : B(x) \to B(y)$.*

Proof. We can define $\mathsf{tr}^B(p)$ by path induction on p. So we may assume that $a' \equiv a$ and $p \equiv 1_a$. Since $1_a^{-1} \equiv 1_a$, we only need to define function $\mathsf{tr}^B(1_a) : B(a) \to B(a)$. The obvious (and only) candidate is $\mathsf{tr}^B(1_a) :\equiv \mathsf{id}_{B(a)}$. $\square$

When the type family is clear from context, we may denote $\mathsf{tr}^B(p)$ by $(p)_*$. To prove this is an equivalence, we require an additional definition.

Theorem 1.3.2 (2-D transport, $\mathsf{tr}^{(B)^2}$)**.** *Let $p, p' : x = y$ and $\alpha : p = p'$. Then, for each $u : B(x)$, we have a term $\mathsf{tr}^{(B)^2}(\alpha)(u) = \mathsf{tr}^B(p)(u) = \mathsf{tr}^B(p')(u)$.*

Proof. By path induction on α. □

Once we discuss homotopies, it will be clear that $\mathsf{tr}^{(B)^2}(\alpha)$ is homotopy between $\mathsf{tr}^B(p)$ and $\mathsf{tr}^B(p')$. We can think of $\mathsf{tr}^{(B)^2}$ as a higher dimensional transport: a path $p : x = y$ induces an equality $B(x) = B(y)$ and a 2-path induces a higher equality $\mathsf{tr}^B(p) = \mathsf{tr}^B(p')$.

For an arbitrary type family B and path p, there is not much one can say about the function $\mathsf{tr}^B(p)$. Some of the subsequent sections deal with characterizing $\mathsf{tr}^B(p)$ in specific instances. We can, however, prove some basic results about tr^B. We already know that $\mathsf{tr}^B(p^{-1}) = \mathsf{tr}^B(p)^{-1}$. The next theorem characterizes $\mathsf{tr}^B(p \cdot q)$.

Theorem 1.3.3 (transport on concatenation, $\mathsf{tr\text{-}concat}$)**.** *Let $p : a = a'$ and $q : a' = a''$. Then, for every $u : B(a)$, we have $\mathsf{tr}^B(p \cdot q)(u) = \mathsf{tr}^B(q) \circ \mathsf{tr}^B(p)(u)$.*

Proof. By path induction on p and q, it suffices to demonstrate this for the case $p \equiv q \equiv 1_a$. But in this case, we have $p \cdot q \equiv 1_a \cdot 1_a \equiv 1_a$. So

$$\mathsf{tr}^B(p \cdot q) \equiv \mathsf{tr}^B(1_a) \equiv \mathsf{id} \equiv \mathsf{id} \circ \mathsf{id} \equiv \mathsf{tr}^B(q) \circ \mathsf{tr}^B(p)$$

□

This theorem amounts to constructing a homotopy between the two functions $\mathsf{tr}^B(p \cdot q)$ and $\mathsf{tr}^B(q) \circ \mathsf{tr}^B(p)$.

Putting these previous results together, we get the following theorem.

Theorem 1.3.4. *For $p : x = y$, $\mathsf{tr}^B(p)$ is an equivalence with inverse $\mathsf{tr}^B(p^{-1})$.*

Proof. We have $\mathsf{tr}^B(p) \circ \mathsf{tr}^B(p^{-1}) \sim \mathsf{tr}^B(p^{-1} \cdot p) \sim \mathsf{tr}^B(1) \equiv \mathsf{id}$. The other direction is similar. □

The equivalence $\mathsf{tr}^B(p)$ lets us define an very important concept: dependent paths. Type

families $B : A \to U$ often aries as a consructions on A. That is, for each $a : A$, we think of deriving a structure $B(a)$ which may tell us something about the point a. The equivalence tr^B demonstrates that, given a path $p : a = a'$, the resulting constructions $B(a)$ and $B(a')$ must be equivalent (in a specified way). In the same context, we may want to consider two point $u : B(a)$ and $v : B(a')$ "the same". However, we cannot ask for a path between the two points since u and v need not be of the same type (which is a prerequisite to even form the type $u = v$). However, we can ask for a path between u and v under the equivalence $\mathsf{tr}^B(p)$. That is, we can ask for a term of type $\mathsf{tr}^B(p)(u) = v$. We will call a term of this type a dependent path over p. We will often denote this type by $u =^B_p v$.

Definition 1.3.1 (dependent paths)**.** *Given $B : A \to U$ and $p : a = a'$ in A, we can form the type $u =^B_p v :\equiv \mathsf{tr}^B(p)(u) = v$ of dependent paths from u to v.*

Dependent paths are a fundamental concept in homotopy type theory. We will flesh out more of their properties in section 1.6. One thing to note is that, if the path p is non-trivial, then the type of depedent paths can be non-trivial as well. Consider, for instance, a loop $l : a = a$. This induces an autoequivalence $\mathsf{tr}^B(l) : B(a) \simeq B(a)$. This autoequivalence need not be the identity function. In fact, part of what makes homotopy type theory tick is that the equivalence $\mathsf{tr}^B(l)$ can be non-trival. This means that, for a point $u : B(a)$, we need not have a term of type $u =^B_l u$. We will see our first example of this when we consider the incarnation of the circle in homotopy type theory.

So far, these results suggest thinking of a type family B as a copresheaf over A. We have yet to relate B to a fibration or fiber bundle. Section 1.5 does exactly that. But, before we can turn our attention make such ideas as path lifting properties precise, we need to discuss how function interact with paths.

1.4 Function Types

In this section we characterize the path types of function types and we discuss some of the further properties of functions in HoTT. A main result of this section is that functions behave like functors between ∞-groupoids.

1.4.1 Paths in Function Types

For types $A : U$ and $B : A \to U$ and functions $f, g : \prod_{x:A} Bx$, we can define the type of "homotopies" between f and g:

$$f \sim g :\equiv \prod_{x:A} fx = gx$$

This is just the type of point-wise identifications between two functions. One might object that this is an insufficent definition since a homotopy should be a *continous* point-wise identification. But, in fact, in HoTT, *all functions are continous*. Closer to the truth, all functions are functors. This is sufficent to endow any term of type $f \sim g$ with the structure of a *natural transformation*. We will show this in a later section.

One might hope that we can prove $(f = g) \simeq (f \sim g)$. Sadly, unlike $\sum$-types, the universal property of function types is insufficent to characterize the path type $f = g$. So we characterize it "by hand". For each $f : \prod_{x:A} Bx$, we have a canonical element $\mathsf{refl\text{-}htpy}_f : f \sim f$ given by $\mathsf{refl\text{-}htpy}_f :\equiv (\lambda x.1_{fx})$. We use this to define a define a cannonical map

$$\mathsf{happly} : \prod_{f,g:\prod_{x:A} Bx} (f = g) \to (f \sim g)$$

We can define this by path induction. We set $\mathsf{happly}(1_f) :\equiv \mathsf{refl\text{-}htpy}_f$. When working with happly, we leave the f and g implicit. The base type theory, MLTT, is not strong enough to prove that happly is an equivalence. But we can assume as an axiom that we have a term

$\mathsf{fe} : \mathsf{is\text{-}equiv}(\mathsf{happly})$. So now we have $(\mathsf{happly}, \mathsf{fe}) : (f = g) \simeq (f \sim g)$. We engage in the usual abuse of notation and write fe for the inverse function $f \sim g \to f = g$.

1.4.2 Homotopies

Since homotopies serve as paths in function types, it is important to develop some basic theory of homotopies. We will show that we can concatenate and invert homotopies analogous to how we can concatenate and invert paths.

Lemma 1.4.1 $(H \cdot_h H')$**.** *Given functions $f, g, h : \prod_{x:A} Bx$ and homotopies $H : f \sim g$ and $H' : g \sim h$, we can construct a homotopy $H \cdot_h H' : f \sim h$.*

Proof. Let $x : A$. We want to construct a term $(H \cdot_h H')x : fx = hx$. Since $Hx : fx = gx$ and $H'x : gx = hx$, we set $(H \cdot_h H')(x) :\equiv Hx \cdot H'x$. □

Lemma 1.4.2 (H^{-1})**.** *Given $H : f \sim g$, we can construct a term $H^{-1} : g \sim f$*

Proof. Let $x : A$. Since $Hx : fx = gx$, we can define $(H^{-1})(x) :\equiv (Hx)^{-1}$. □

1.4.3 Functions are Functors

Now we turn our attention to proving that every function is a functor. The proofs are considerably more straight forwards, and highlight the conceptual aspects better, when we restrict our attention to non-dependent functions $f : A \to B$. So we first prove the results for a regular functions, then generalize to dependent functions.

Non-Dependent Functions

In this section, unless otherwise specified, we use $A, B : U$ with $f, g : A \to B$. If f really behaves as a functor, it ought to send paths in A to paths in B.

Lemma 1.4.3 (Action on Paths, ap)**.** *For each $x, y : A$, we have a function $\mathsf{ap}(f) : (x = y) \to (fx = fy)$.*

Proof. We can define this function by path induction. So it suffices to assume $y \equiv x$ and specify the value of $\mathsf{ap}(f)(1_x) : fx = fx$. Thus we may set $\mathsf{ap}(f)(1_x) :\equiv 1_{fx}$ □

As is customary with functors, we will abuse notation and usually write $f(p)$ instead of $\mathsf{ap}(f)(p)$. There are a few cases where this may be confusing. For instance, if we have a function $f : \prod_{x,y:A} x = y \to B$. It is important not to confuse $f(p)$ for $p : x = y$ and $f(\alpha)$ for $\alpha : p = p'$, since there are coherence theorems that apply to the latter but not the former. In these cases, where it is important to disagmiguate, we may write the extra ap or simply state which we are using.

To show that ap really behaves like the action on paths of a functor, we need to show it preserves the operations on paths. First we show that *ap* of the identity function id is also the identity function.

Lemma 1.4.4 (ap of id_A, $\mathsf{ap\text{-}id}$)**.** *For any type A and $p : x = y$ in A, we have a term $\mathsf{ap\text{-}id} : \mathsf{id}_A(p) = p$.*

Proof. By path induction it suffices to specify $\mathsf{ap\text{-}id}(1_x) : \mathsf{id}_A(1_x) = 1_x$. But $\mathsf{id}_A(1_x) \equiv 1_{\mathsf{id}_A(x)} \equiv 1_x$. Thus we set $\mathsf{ap\text{-}id}(1_x) :\equiv 1_x^2$. □

Lemma 1.4.5 (Action on paths on concatination of paths, $\mathsf{ap\text{-}comp}$)**.** *For $f : A \to B$ with $p : x = y$ and $q : y = z$ in A, we have a term $\mathsf{ap\text{-}comp}_{f,p,q} : f(p \cdot q) = f(p) \cdot f(q)$.*

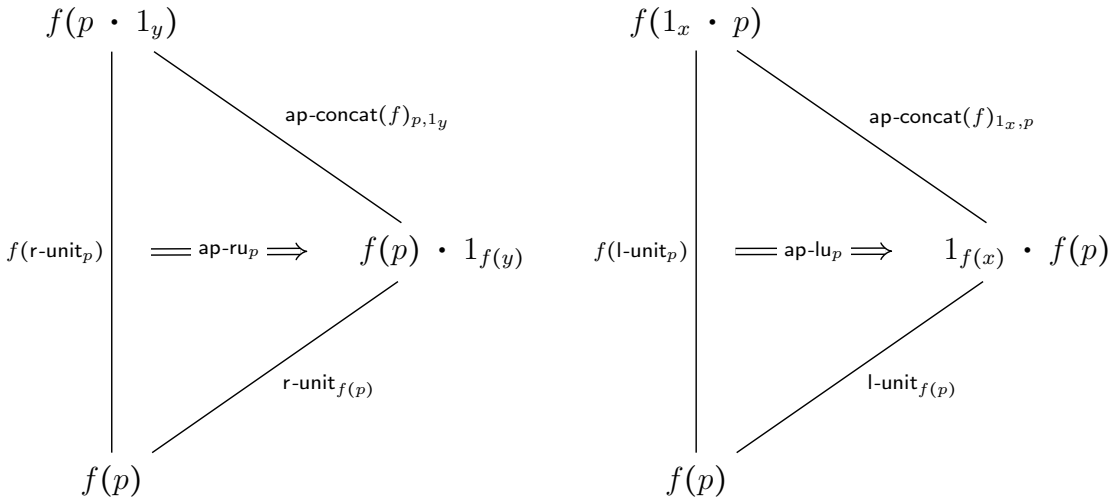
Proof. We can use path induction on both p and q (since they are not parallel). Since $1_x \cdot 1_x \equiv 1_x$, our goal reduces to $1_{f(x)} = 1_{f(x)}$. Thus, we set $\mathsf{ap\text{-}comp} :\equiv 1^2_{f(x)}$. □. □

Lemma 1.4.6 (Action on paths on inverses, $\mathsf{ap\text{-}inv}$)**.** *For $f : A \to B$ and $p : x = y$, we have a term $\mathsf{ap\text{-}inv}_{f,p} : f(p^{-1}) = f(p)^{-1}$.*

Proof. By path induction on p our goal reduces to $1_{f(x)} = 1_{f(x)}$. So we may set $\mathsf{ap\text{-}inv} :\equiv 1^2_{f(x)}$. □

We can also show that f preserves the coherence paths in A.

Lemma 1.4.7 (ap-ru and ap-lu)**.** *For $f : A \to B$ and $p : x = y$ in A, we have the following two commutative diagrams:*



Proof. By path induction on p, everything reduces to a trivial path. □

We can derive similar coherences fo l-inv, r-inv and all other coherences in 1-paths. But we will only need the two already given and the coherences for l-inv and r-inv.

Though these lemmas do not suffice to prove that a function is a functor, they are essentially all of the functorial structure we need for this thesis (some lemmas are saved for latter sections). For a genuine proof that functions are functors, one needs to look at the semantics of HoTT.

1.4.4 Dependent Functions

There is an anlouge of ap for dependent functions, though it takes a bit more care to make sense of it. Suppose we have a function $f : \prod_{x:A} Bx$ and a path $p : x = y$ in A. In lieu of this path, we want to relate the values of fx and fy. But, it does not make sense to ask for a path $fx = fy$ since $fx : Bx$ and $fy : By$ and Bx need not be the same type as By! However, we know that p induces an equivalence $\mathsf{tr}^B p : Bx \simeq By$ and we can ask that, under this equivalence, fx equals fy. That is, we can ask for a dependent path $f(x) =^B_p f(y)$.

Lemma 1.4.8 (Dependent action on paths, apd)**.** *Given $f : \prod_{x:A} Bx$ and $p : x = y$ in A, we have a term $\mathsf{apd}(f)(p) : f(x) =^B_p f(y)$. That is, for each f and $x, y : A$, we have a function $\mathsf{apd}(f) : \prod_{p:x=y} \mathsf{tr}^B(p)(fx) = fy$.*

Proof. By path induction on p, it suffices to assume $y \equiv x$ and $p \equiv 1_x$. Thus we need to specify specify $\mathsf{apd}(f)(1_x) : \mathsf{tr}^B(1_x)(fx) = fx$. But $\mathsf{tr}^B(1_x)(fx) \equiv fx$. Thus we may define $\mathsf{apd}(f)(1_x) :\equiv 1_{fx}$. □ □

Just like with the non-dependent action on paths, we will abuse notation and write $f(p)$ for $\mathsf{apd}(f)(p)$. Of course apd has coherences analgous to those of ap. However, more care is necessary in formulating these coherences since they require working with dependent paths. We will delay formulating these coherences until we have developed some machinery for working with dependent paths.

1.4.5 Homotopies are natural transformations (apd of homotopies)

Lets return our attention to the task of showing that non-dependent functions behave like functors. We have already shown that functions have an induced action on paths and coherences like that of a functor. But if functions really behave like functors, then homotopies ought to behave like natural transformations. Consider two functions $f, g : A \to B$ and a homotopy $H : f \sim g \equiv \prod_{x:A} fx = gx$. Since H is a dependent function, it has an induced

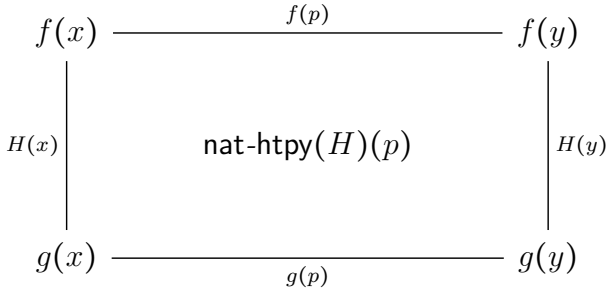
$\mathsf{apd}(H) : \prod_{p:x=y} H(x) =_p^{\lambda f x=gx} H(y)$. In its current, $\mathsf{apd}(H)$ is unenlightening. But, due to the specific structure of the type family $\lambda(x).f(x) = g(x)$, we can give a nice characterization of the type of dependent paths.

Theorem 1.4.1 (dependent paths in $\lambda(x).f(x) = g(x)$)**.** *Let $p : x = y$ with $q : f(x) = g(x)$ and $r : f(y) = g(y)$. Then we have $(q =_p^{\lambda f x=gx} r) \simeq (q \cdot g(p) = f(p) \cdot r)$*

Proof. By path induction on p, it suffices to establish an equivalence between $q = r$ and $(q \cdot g(1_x) = f(1_x) \cdot r)$. But $g(1_x) \equiv 1_{g(x)}$ and $f(1_x) \equiv 1_{f(x)}$. Thus we can use the equivalence that sends $t : q = r$ to $\mathsf{l\text{-}unit}_t \cdot t \cdot \mathsf{r\text{-}unit}_t$. □

Thus, the type of $\mathsf{apd}(H)(p)$ is equivalent to $g(p) \cdot H(y) = f(p) \cdot H(x)$. This can be readily recognized as a commutative square expressing the naturality of H. It will be helpful to have a function that takes us directly from paths to in A to naturality squares

Theorem 1.4.2 (naturality of homotopies, $\mathsf{nat\text{-}htpy}$)**.** *Given $f, g : A \to B$ and a homotopy $H : f \sim g$, we have a function $\mathsf{nat\text{-}htpy}(H) : \prod_{p:x=y} g(p) \cdot H(y) = f(p) \cdot H(x)$. That is, for each $p : x = y$, we have a commutative diagram:*

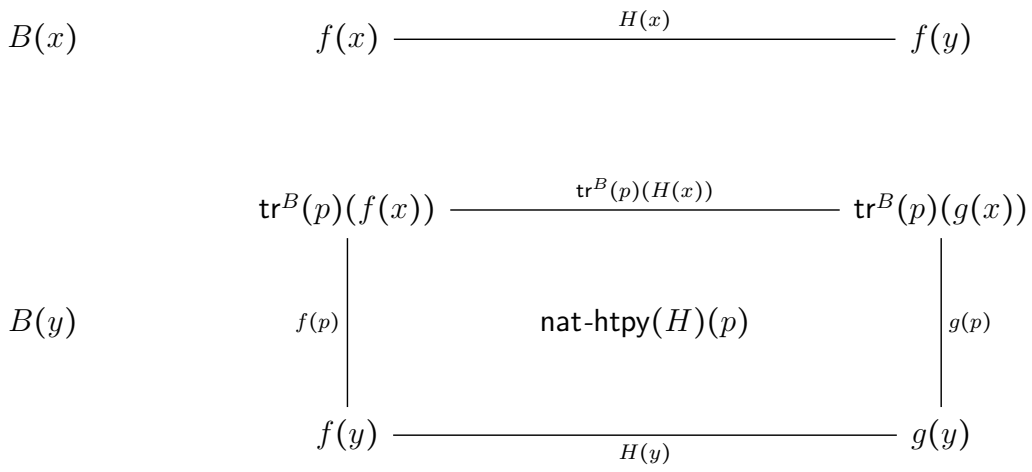


Proof. We define $\mathsf{nat\text{-}htpy}$ by using apd and then applying the equivalence between the two types mentioned above. □

Again, we will abuse notation and write $H(p)$ for $\mathsf{nat\text{-}htpy}(H)(p)$. Since we will never have a reason to use $\mathsf{apd}(H)(p)$, this will not cause us any problems in general. However, it is possible for that, for some previously defined homotopy H, the notation $H(p)$ is already

meaningful (an example of this will arise later with l-unit). In this case, we will either use the longer notation or introduce something specific to the homotopy H. Thus, we will normally write diagrams of the above form with just $H(p)$ as a filler. Before moving on, we give the analog of nat-htpy for dependent functions. Similar to the issues we ran into with apd for dependent functions, we cannot ask directly for a commutative square involving $H(x)$ and $H(y)$ since $H(x) : Bx$ and $H(y) : By$. But we can use the induced equivalence $\mathsf{tr}^B(p)$ and its action on paths.

Lemma 1.4.9. *For $f, g : \prod_{x:A} Bx$ and a homotopy $H : f \sim g$, we have a term $\mathsf{nat\text{-}htpyd}(H) : \prod_{p:x=y} f(p) \cdot H(y) = \mathsf{tr}^B(p)(H(x)) \cdot g(p)$. That is, for each $p : x = y$, we have a commutative diagram:*



Here, $f(p)$ and $g(p)$ are the dependent action on paths. But $\mathsf{tr}^B(p)(H(x))$ is the non-dependent action on paths of the equivalence $\mathsf{tr}^B(p) : Bx \simeq By$. Again, we will from here on out abuse notation and simply write $H(p)$ for $\mathsf{nat\text{-}htpyd}(H)(p)$. We can use these results for a great many things. Many of our definitions take the form of homotopies. For example, pretty much every idenity and coherence path we have defined can be viewed as a homotopy between two different ways of composing functions. We will use this perspective later when working with higher paths.

1.4.6 Functions of Two Variable Functions and ap

Here we have brief interlude on the action on paths of a function in two variables. In HoTT, two variable functions $A \times B \to C$ are quite frequently defined by currying a function $A \to B \to C$. Thus it is worthwhile to understand the behavior of the action on paths of a binary function in terms of its curried counterpart. Consider a curried two variable function function $f : A \to B \to C$ and two paths $p : a_0 = a_1$ and $q : b_0 = b_1$. These paths correspond to a path $(p, q) : (a_0, b_0) = (a_1, b_1)$ in $A \times B$. This should determine a path $f(a_0)(b_0) = f(a_1)(b_1)$. But we cannot directly apply ap to f and a path in $A \times B$. We can only apply ap to f and p to get $f(p) : f(a_0) = (a_1)$, a path in a function space. Thus, the following variant of action on paths will also be helpful.

Lemma 1.4.10 (Binary Action on Paths, $\mathsf{bin\text{-}ap}$)**.** *For $f : A \to B \to C$, and each $a_0, a_1 : A$ and $b_0, b_1 : B$, we have a function $\mathsf{bin\text{-}ap} : (a_0 = a_1) \times (b_0 = b_1) \to f(a_0, b_0) = f(a_1, b_1)$.*

Proof. By path induction in each argument, it suffices to set $\mathsf{bin\text{-}ap}(f)(1_{a_0}, 1_{b_0}) :\equiv 1_{f(a_0,b_0)}$. □

There are of course coherences for $\mathsf{bin\text{-}ap}$ anlogous to ap. First we prove that $\mathsf{bin\text{-}ap}$ is the action on paths of a function defined by currying.

Lemma 1.4.11. *Let $f : A \to B \to C$ and $\hat{f} : A \times B \to C$ denote its uncurried counterpart. Let $p : a_0 = a_1$ and $q : b_0 = b_1$. Then $\mathsf{bin\text{-}ap}(f)(p, q) = \mathsf{ap}(\hat{f})(p, q)$.*

Proof. By path induction on p and q, both sides reduce to the identity. □

In virtue of this lemma, we will use the notation $f(p, q)$ to denote either $\mathsf{bin\text{-}ap}(f)(p, q)$ or $\mathsf{ap}(\hat{f})(p, q)$, depending on what the domain of f is. We will also generally write $f(a, b)$ for what should be written as $f(a)(b)$.

Many of the coherences for $\mathsf{bin\text{-}ap}$ play an essential role in the Eckmann-Hilton argument, since $(- \cdot -)$ is a function $(x = y) \to (y = z) \to (x = z)$. Each proof is an easy path induction.

A theme in HoTT is that stament of a lemma is fairly intricate, while the proof the lemmas is quite simple.

Lemma 1.4.12 (Binary action on concatination of paths, bap-concat)**.** *For $f : A \to B \to C$ with $p : a_0 = a_1$ and $p' : a_1 = a_2$ in A and $q : b_0 = b_1$ and $q' : b_1 = b_2$ in B, we have a term* $\mathsf{bap\text{-}comp} : f(p \cdot p', q \cdot q') = f(p,q) \cdot f(p',q')$

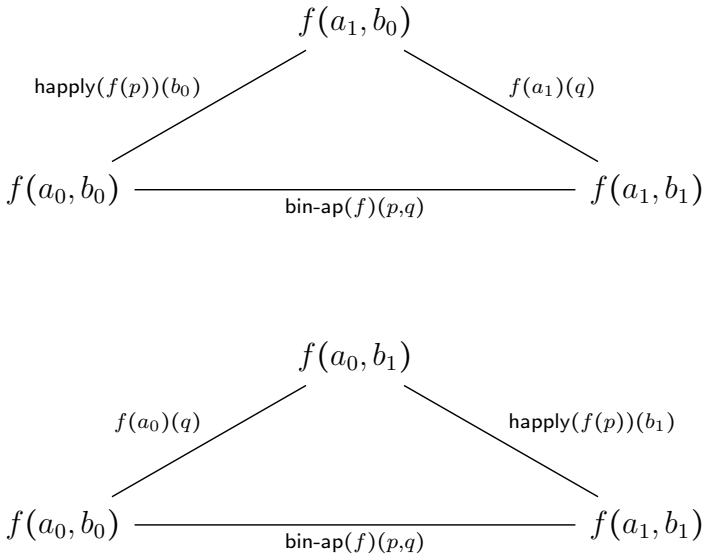
Proof. Since none of the paths are parallel, we may path induct on each path. Thus our goal reduces to providing a path $\mathsf{bap\text{-}comp} : f(1_{a_0} \cdot 1_{a_0}, 1_{b_0} \cdot 1_{b_0}) = f(1_{a_0}, 1_{b_0}) \cdot f(1_{a_0}, 1_{b_0})$. But each side reduces to $1_{f(a_0,b_0)}$. Thus we set $\mathsf{bap\text{-}comp} :\equiv 1^2_{f(a_0,b_0)}$. □

Lemma 1.4.13 (Binary action on paths on inverses, bap-inv)**.** *For $f : A \to B \to C$ with $p : a = a'$ and $q : b = b'$, we have term* $\mathsf{bap\text{-}inv} : f(p^{-1}, q^{-1}) = f(p,q)^{-1}$.

Proof. We use path induction on p and q. Since $1_a^{-1} \equiv 1_a$ (and likewise for b), our goal reduces to $1_{f(a,b)} = 1_{f(a,b)}$. □

bin-ap and ap

The rest of the coherences that feature into the Eckmann-Hilton proof establish certain relations between bin-ap and *ap*. Given f as in the definition of bin-ap and $p : a_0 = a_1$ and $q : b_0 = b_1$, we of course have $f(p,q) : f(a_0,b_0) = f(a_1,b_1)$. But, we also have $f(p) : f(a_0) = f(a_1)$, which is a path in the function type $B \to C$. Thus $\mathsf{happly}(f(p)) : f(a_0) \sim f(a_1)$. We can apply this homotopy to b_0 to obtain a path $f(a_0,b_0) = f(a_1,b_0)$. Now consider the function $f(a_1)$, of type $B \to C$. We have $f(a_1)(q) : f(a_1,b_0) = f(a_1,b_1)$. We can concatinate these two paths to get a path in $f(a_0,b_0) = f(a_1,b_1)$, parallel to $f(p,q)$. Additionally, we could have carried out our factorization in the opposite order, i.e., conisdering $f(a_0)(q)$ and $\mathsf{happly}(f(p))(b_1)$. We can put this all together to get the following two diagrams:



Lemma 1.4.14 (Binary Action on Paths Coherence , $\mathsf{coh}_{\mathsf{bin\text{-}ap}}$)**.** *The above diagrams commute. That is, we have terms* $\mathsf{r\text{-}coh}_{\mathsf{bin\text{-}ap,ap}} : \mathsf{happly}(f(p))(b_0) \cdot f(a_1)(q) = \mathsf{bin\text{-}ap}(f)(p,q)$ *and* $\mathsf{l\text{-}coh}_{\mathsf{bin\text{-}ap,ap}} : f(a_0)(q) \cdot \mathsf{happly}(f(p))(b_1) = \mathsf{bin\text{-}ap}(f)(p,q)$

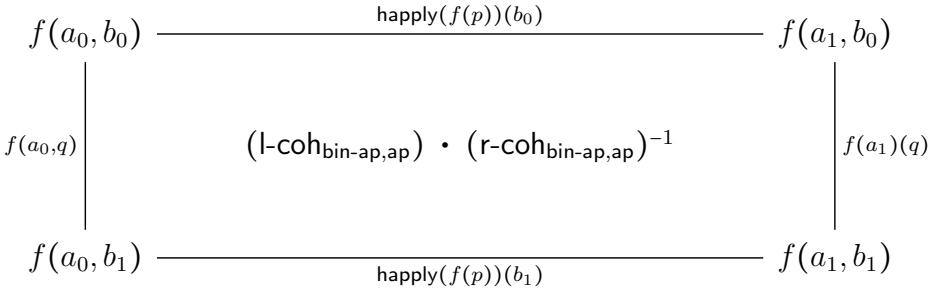
The choice of which coherence path gets the left versus right marker is a little bit arbitrary here. We choose right for the first since, in the composition of paths, the action on paths $f(a_1)(q)$ appears on the right. We choose left for the second for similar reasons.

Proof. Let us first construct $\mathsf{r\text{-}coh}_{\mathsf{bin\text{-}ap,ap}}$. We do so by path induction on p and q. Essentially both sides reduce to trivial paths. In particular, $f(p) \equiv f(1_{a_0}) \equiv 1_{f(a_0)}$. We also have that $\mathsf{happly}(1_{f(a_0)})(b_0) \equiv \mathsf{refl\text{-}htpy}_f(b_0) \equiv 1_{f(a_0,b_0)}$. Lastly $f(a_1)(q) \equiv f(a_0)(1_{b_0}) \equiv 1_{f(a_0,b_0)}$. Thus, the left hand side of the equation becomes $1_{f(a_0,b_0)} \cdot 1_{f(a_0,b_0)} \equiv 1_{f(a_0,b_0)}$. Since $\mathsf{bin\text{-}ap}(f)(1_{a_0}, 1_{b_0}) \equiv 1_{f(a_0,b_0)}$, we can set $\mathsf{r\text{-}coh}_{\mathsf{bin\text{-}ap,ap}} :\equiv 1^2_{f(a_0,b_0)}$ when p and q are trivial. This completes the definition of $\mathsf{r\text{-}coh}_{\mathsf{bin\text{-}ap,ap}}$

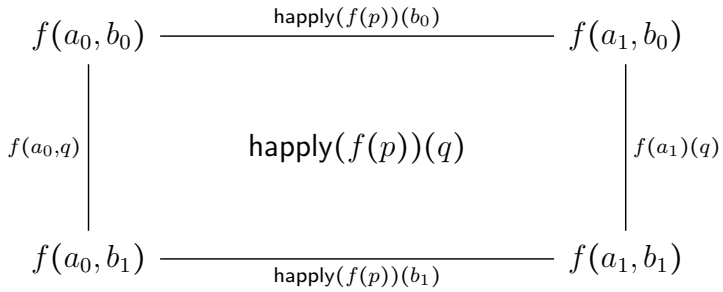
The construction of $\mathsf{l\text{-}coh}_{\mathsf{bin\text{-}ap,ap}}$ is similar. By path induction, our goal reduces to $1_{f(a_0,b_0)} = 1_{f(a_0,b_0)}$, which we inhabit with $1^2_{f(a_0,b_0)}$. □

We can paste these two triangle together along their common diagonal to obatin a com-

mutative square with filler $(\mathsf{l\text{-}coh}_{\mathsf{bin\text{-}ap,ap}}) \cdot (\mathsf{r\text{-}coh}_{\mathsf{bin\text{-}ap,ap}})^{-1} : f(a_0)(q) \cdot \mathsf{happly}(f(p))(b_1) = \mathsf{happly}(f(p))(b_0) \cdot f(a_1)(q)$:



This path will be important in the Eckmann-Hilton argument, so we will give it a name. We'll call this filler $\mathsf{bap\text{-}swap}_{p,q}$. Note that we can also apply $\mathsf{nat\text{-}htpy}$ to $\mathsf{happly}(f(p))$ to obtain the follow commutative diagram directly:



In fact, these two fillers are equal.

Lemma 1.4.15 (Coherence between $\mathsf{happly}(f(p))(q)$ and $(\mathsf{l\text{-}coh}_{\mathsf{bin\text{-}ap,ap}}) \cdot (\mathsf{r\text{-}coh}_{\mathsf{bin\text{-}ap,ap}})^{-1}$)**.** *We have a term* $\mathsf{coh}_{\mathsf{happly}(f(p))(q)} : \mathsf{nat\text{-}htpy}(\mathsf{happly}(f(p))) = (\mathsf{l\text{-}coh}_{\mathsf{bin\text{-}ap,ap}}) \cdot (\mathsf{r\text{-}coh}_{\mathsf{bin\text{-}ap,ap}})^{-1}$.

Proof. We use path induction on p and q. By definition, both $(\mathsf{l\text{-}coh}_{\mathsf{bin\text{-}ap,ap}})$ and $(\mathsf{r\text{-}coh}_{\mathsf{bin\text{-}ap,ap}})$ reduce to $1_{f(a_0,b_0)}$ on trivial paths. But $\mathsf{happly}(f(1_{a_0}))(1_{b_0}) :\equiv \mathsf{happly}(1_{f(a_0)})(1_{b_0}) \equiv \mathsf{refl\text{-}htpy}_{f(a_0)}(1_{b_0})$. But, by lemma **??**, $\mathsf{refl\text{-}htpy}_{f(a_0)}(1_{b_0}) = \mathsf{r\text{-}unit}_{1_{f(b_0)}} \cdot \mathsf{l\text{-}unit}^{-1}_{1_{f(b_0)}} \equiv 1^2_{f(a_0,b_0)} \cdot 1^2_{f(a_0,b_0)} \equiv 1^2_{f(a_0,b_0)}$. Thus, when p and q are the trivial paths, we can set $\mathsf{coh}_{\mathsf{happly}(f(p))(q)} :\equiv 1^3_{f(a_0,b_0)}$. □

So why define $\mathsf{bap\text{-}swap}$ as we did? Our definition has the virtue of definitionally factoring

through $f(p,q)$, which will serve to our benefit when proving Eckmann-Hilton.

This concludes our in depth look at functions. These results play a foundational role it the proofs to come.

1.5 Sigma Types

In this section we discuss $\sum$-types. The $\sum$ construction behaves like a functor $U \to U$, which takes as an input a base type $A : U$ and a type family $B : A \to U$ and spits out "the total space of $B : A \to U$", denoted $\sum_{x:A} B(x)$. The main result of this section is show that the construction $\sum$ behaves like the Grothendieck construction and the type $\sum_{x:A} B(x)$ behaves like a fibration over A.

The type $\sum_{x:A} B(x)$ is defined as the type of "dependent pairs" of point $a : A$ and points $u : B(a)$. We can construct a term $(a,u) : \sum_{x:A} B(x)$ from such an a and u. To construct a map $(\sum_{x:A} B(x)) \to X$, we can instead construct a map $B(x) \to X$ for each $x : A$ (i.e., a term of type $\prod_{x:A} B(x) \to X$). This lets us define a projection onto A:

Theorem 1.5.1 (pr_1 , pr_2)**.** *We have a canonical function $(\sum_{x:A} B(x)) \to A$ that sends a pair (x,u) to x. There is also a canonical dependent function $\mathsf{pr}_2 : \prod_{z:\sum_{x:A} B(x)} B \circ pr_1(z)$ that sends (x,u) to u.*

Proof. We may define pr_1 by constructing the curried version of type $\prod_{x:A} B(x) \to A$. Let $x : A$ and $u : B(x)$. We define $\mathsf{pr}_1(x,u) :\equiv x$. We similarly define pr_2 by sending (x,u) to u. □

Lemma 1 (lift)**.** *For $A : U$, $x, y : A$, $B : A \to U$, $u : Bx$ and a path $p : x = y$, we have path $\mathsf{lift}^B(u,p) : (x,u) = (x, \mathsf{tr}^B(p)(u))$ in $\sum_{x:A} Bx$ such that $\mathsf{pr}_1(\mathsf{lift}^B(u,p)) = p$.*

Proof. We define $\mathsf{lift}(u,p)$ by path induction on p. When $y \equiv x$ and $p \equiv 1_x$ our goal becomes constructing a path $(x,u) = (x, \mathsf{tr}^B(1_x)(u))$. But $\mathsf{tr}^B(1)(u) \equiv u$. Thus we can inhabit our

goal with $\mathsf{lift}(u, 1_x) :\equiv 1_{(x,u)}$. This completes the definition of lift.

To complete the proof, we need to show $\mathsf{pr}_1(\mathsf{lift}^B(u,p)) = p$. By path induction, it suffices to show that $\mathsf{pr}_1(\mathsf{lift}(u,1_x)) = 1_x$. But $\mathsf{lift}(u,1_x) \equiv 1_{(u,x)}$ and $\mathsf{pr}_1(1_{(u,x)}) \equiv 1_{\mathsf{pr}_1(x,u)} \equiv 1_x$. Thus we may inhabit our goal with 1^2_x. $\square$

Though we have not proven the full path lifting property required of fibrations, we can see why we should expect $\mathsf{pr}_1 : (\sum_{x:A} B(x)) \to A$ to behave like a fibration over A. This leads us to our next important result. The $\sum$-construction takes something that behaves like a copresheaf and spits out a fibration. This is reminiscent of the Grothendieck construction. The simpliest verison of the Grothendieck construction arieses in 1-categories and, in this context, is often called the category of elements. Denoted $\int$, the Grothedieck construction takes a set valued functor $F : C \to \mathrm{Set}$, and spits out a category with projection $(\int F) \to C$. The objects of $\int F$ are pairs (c, x), with $c \in C$ and $x \in Fc$. The arrows $(c, x) \to (c', x')$ in $\int F$ are arrows $f : c \to c'$ such that $Ff(x) = x'$. We already can see the parallel between objects of $\int F$ and the terms of $\sum_{x:A} Bx$, since a term of the latter is equal to one of the form (a, b) with $a : A$ and $b : Ba$. To see the analogy between arrows in $\int F$ and paths in $\sum_{x:A} Bx$, we require a lemma:

Theorem 1.5.2 (Paths in $\sum$-types)**.** *For $z, z' : \sum_{x:A} Bx$, we have*

$$(z = z') \simeq \Big(\sum_{p:\mathsf{pr}_1(a)=\mathsf{pr}_1(a')} \mathsf{pr}_2(z) =^B_p \mathsf{pr}_2(z')\Big)$$

Proof. We can construct the forwards map by sending $p : z = z'$ to $(\mathsf{pr}_1(p), \mathsf{pr}_2(p))$, where there latter function applications denote the action on paths and dependent action on paths, respectively. We can define the inverse function $(\sum_{p:\mathsf{pr}_1(z)=\mathsf{pr}_1(z')} \mathsf{tr}^B(p)(\mathsf{pr}_2(z)) = \mathsf{pr}_2(z')n) \to (z = z')$ by $\sum$-induction. Thus we need a function $\prod_{p:\mathsf{pr}_1(z)=\mathsf{pr}_1(z')}(\mathsf{tr}^B(p)(\mathsf{pr}_2(z)) = \mathsf{pr}_2(z')) \to z = z'$. By further inducting on z and z', we can assume $z \equiv (a, u)$ and $z' \equiv (a', u')$. This allows us to use path induction on p and the path in the second argument. So it suffices to

specify the image of $(1_a, 1_u)$. The obvious choice is $1_{(a,u)}$.

Verifying that these maps define an equivalence is a straight forwards application of induction principles. □

This is a practically important theorem, since it provieds a convnient way to prove that two terms in a $\sum$-type are equal. But it also makes the analougy between $\sum$-types and the category of elements more salient. We will have to return to the claim from section 1.3 (that type families behave like fiber bundles). Our brief discussion about $\sum$-types tells us that every type family $B : A \to U$ induces a fibration $\mathsf{pr}_1 : (\sum_{x:A} B(x)) \to A$. In fact, we will be able to prove that a type family over A is equivalent to a fibration over A (in a sense to be made precise). This, however, requires a few lemmas and some constructions not yet introduced. We will prove this claim in section 3.2.

1.6 Dependent Paths

Dependent paths have already shown up in a couple of different place. We now turn our attention to fleshing out their structure. Let $A : U$ and $B : A \to U$ and consider a path in the base space $p : x = y$ in A and points $u : B(x)$ and $v : B(y)$ over x and y. A dependent path $k : u =^B_p v$ over p can be thought of in a few different ways. We can think of k as establishing that, under the canonical equivalence $\mathsf{tr}^B(p)$, u is made equal to v. But, our discussion of $\sum$-types suggests another fruitful perspective: a dependent path $k : u =^B_p v$ is a path from u to v that "lies over p". We can make that precise with the following theorem (which is essentially a helpful rephrasing of theorem 1.5.2).

Theorem 1.6.1. *A dependent path $k : u =^B_p v$ determines a path $\bar{k} : (x, u) = (y, v)$ in $\sum_{x:A} B(x)$ such that $\mathsf{pr}_1(\bar{k}) = p$ (where this latter function applications is the action on paths).*

Proof. By theorem 1.5.2, we have an equivalence $e : \sum_{t:x=y} u =^B_t v) \simeq ((x, u) = (y, v))$. Thus, a

dependent path $u =^B_p v$ determines a path $e(p,k) : (x,u) = (y,v)$. Further, the inverse of the equivalence e is given by taking the action on paths of pr_1 and pr_2. Thus, $(\mathsf{pr}_1 \circ e(p,k), \mathsf{pr}_2 \circ e(p,k)) = (p,k)$. Applying theorem **??** to this type implies that $\mathsf{pr}_1 \circ e(p,k) = p$. □

In virtue of the preceeding theorem and theorem 1.5.2, we from here on out implicitly imply the equivalence of theorem 1.5.2. Thus, when we speak of paths in a sum type, we will be speaking about a dependent pair of paths. Now we want to define some operations on dependent paths that are suggested by their status as paths in a $\sum$-type. First we translate the concatination of paths in a sum type to an operation on dependent paths, which we call "dependent concatenation" of paths.

Theorem 1.6.2 (Dependent Composition)**.** *Consider points $x, y, z : A$ in the base space with paths $p : x = y$ and $q : y = z$. Now we take points $u : B(x)$, $v : B(y)$ and $w : B(z)$ over A. Then we have a function $(- \cdot_d -) : u =^B_p v \to v =^B_q w \to u =^B_{p \cdot q} w$*

Proof. Given $h : u =^B_p$ and $k : v =^B_q w$, we want to construct a term $h \cdot_d k : u =^B_{p \cdot q} w$. We have the following diagram:

$$(p \cdot q)_*(u) \xrightarrow{\mathsf{tr\text{-}concat}_{p,q}} q_* \circ p_*(u) \xrightarrow{\mathsf{ap}(q_*)(h)} q_*(v) \xrightarrow{k} w$$

So we define $h \cdot_d k$ to be the comoposite. □

We now wish to define the dependent unit path. There are two choices here: (i) we could define the dependent unit so that for each $x : A$ and $u : Bx$, we have $1_u : u =^B_{1_x} u$, or (ii) we could define the dependent unit so that for each $p : x = y$ and $u : Bx$, we have $1_{p_* u} : u =^B_p u$. It may be tempting to choose the second definition, since it encompases the first. However, the second definition leads to certain problems. For one, $1_{p_* u} \cdot_d h$ is not well typed!

Definition 0.1 (Dependent Unit)**.** *We have family $\prod_{x:A} \prod_{u:Bx} u =^B_{1_x} u$.*

Proof. We define the family of dependent unit paths $\prod_{x:A} \prod_{u:Bx} u =^{B}_{1_x} u$ by send x and u to 1_u. This is well typed since $1_* u \equiv u$. □

We now need to verify that these paths behave like inverses under dependent composition. Consider $1_u \cdot h : u =^{B}_{1_u \cdot p} w$. We cannot directly construct a path $1_u \cdot_d h = h$, since the left and right hand side are not in the same type. However, we construct a dependent path $1_u \cdot_d h =^{B^2}_{\mathsf{l\text{-}inv}_p} h$.

Lemma 0.1 (Dependent Unit Laws)**.** *We have a term* $\mathsf{l\text{-}unitd}_h : 1_u \cdot_d h =^{B^2}_{\mathsf{l\text{-}inv}_p} h$*. We also have* $\mathsf{r\text{-}unitd}_h : h \cdot_d 1_v =^{B^2}_{\mathsf{r\text{-}inv}_p} h$*.*

Proof. Each of these identities are easy to see by writing out the definition of $- \cdot_d -$ with 1_x or 1_y in the requiste spots. □

Now we want to define the depndent inverse.

Definition 0.1 (Dependent Inverse)**.** *We have a function* $(-)^{-1^{\mathsf{d}}} : u =^{B}_{p} v \to v =^{B}_{p^{-1}} u$*.*

Proof. Given $h : u =^{B}_{p} v$, we want to construct a term $h^{-1^{\mathsf{d}}} : v =^{B}_{p^{-1}} u$. We have the following diagram:

$$p_*^{-1}(v) \xrightarrow{\mathsf{ap}(p_*^{-1})(h^{-1})} p_*^{-1} \circ p_*(u) \xrightarrow{\mathsf{tr\text{-}concat}^{-1}_{p,p^{-1}}} (p \cdot p^{-1})_*(u) \xrightarrow{\mathsf{tr}^{(B)^2}(\mathsf{r\text{-}inv}_p)(u)} \mathsf{tr}^{B}(1_x)(u) \equiv u$$

We define $h^{-1^{\mathsf{d}}}$ to be the composite. □

Now we need to verify that $h^{-1^{\mathsf{d}}}$ behaves like a dependent inverse. If we consider the composition $h \cdot_d h^{-1^{\mathsf{d}}} : u =^{B}_{p \cdot p^{-1}} u$, it is not well typed to ask if this is equal to $1_u : u =^{B}_{1_x} u$. However, we can ask for a dependent path $\mathsf{r\text{-}invd}_h : h \cdot_d h^{-1^{\mathsf{d}}} =^{B^2}_{\mathsf{r\text{-}inv}_p} 1_u$.

Lemma 0.1. *We have a term* $\mathsf{r\text{-}invd}_h : h \cdot_d h^{-1^{\mathsf{d}}} =^{B^2}_{\mathsf{r\text{-}inv}_p} 1_u$*.*

Proof. By definition our goal is $h \cdot_d h^{-1^{\mathsf{d}}} = \mathsf{tr}^{(B)^2}(\mathsf{r\text{-}inv}_p)(u)$. By looking at the diagram defining $h \cdot_d h^{-1^{\mathsf{d}}}$ (i.e. replacing q with p^{-1} and k with $h^{-1^{\mathsf{d}}}$ the diagram in definition 1.6.2 , we can see that see that all paths but $\mathsf{tr}^{(B)^2}(\mathsf{r\text{-}inv}_p)(u)$ cancel. □

We can also prove the left dependent inverse law without path induction, but the proof is a bit more tedious since things do not cancel directly. But we will not need an explicit description of $\mathsf{l\text{-}invd}_h$, so we can convince ourselves that it exists with path induction.

Lemma 0.1. *We have a term* $\mathsf{l\text{-}invd}_h : h^{-1^{\mathsf{d}}} \cdot_d h :=^{B^2}_{\mathsf{l\text{-}inv}_p} 1_v$

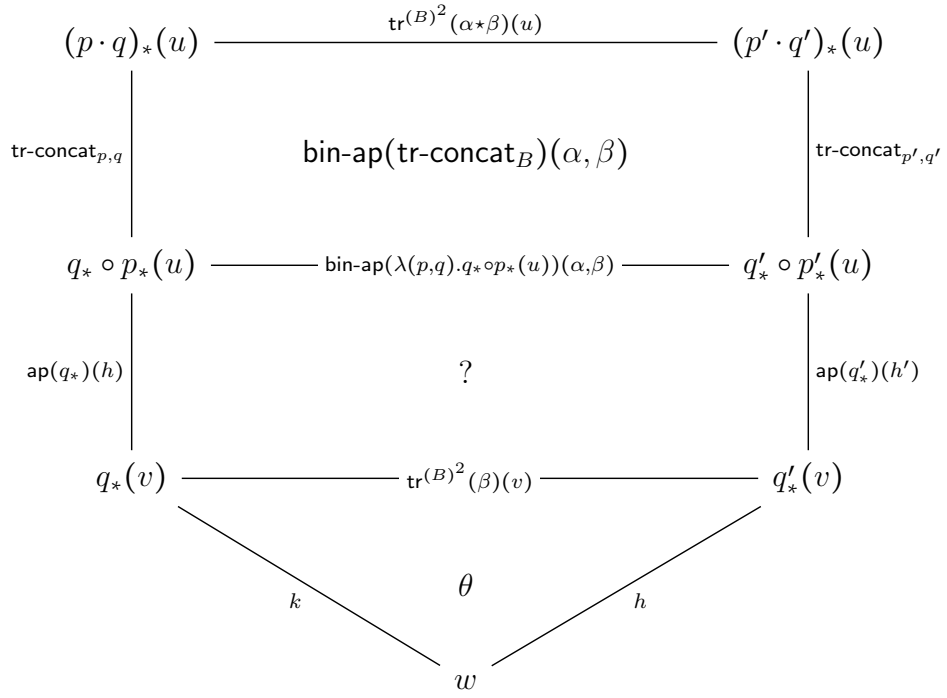
Proof. By path induction on p. □

There are of course dependent analougs of all other identities (e.g., associativity and inverse distributing over concatenation). Again, we will not need explicit descriptions of these paths, so I ommit the proofs. ???????Add THE LEMMA STATEMENTS?

Now we want to define the dependent horizontal composition of dependent 2-paths. This will take the form of a function $h =^{B^2}_p h' \to k =^{B^2}_p k' \to h \cdot_d k =^{B^2}_{p \cdot q} h' \cdot_d k'$. Though this is a bit more laborious than than defining the previous dependent operators, it can still be done without path induction. This is a worthwhile endevor because it gives us better reduction behavior. It is also a nice example of "a diagram chase" in HoTT.

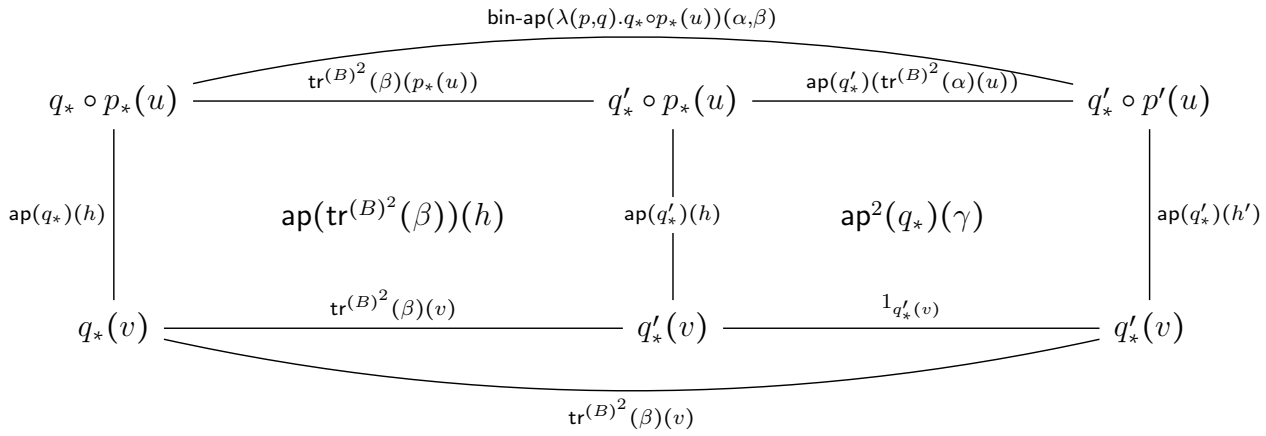
Definition 0.1 (Dependent Horizontal Composition, $\gamma \star_d \theta$)**.** *We have a function* $(- \star -) : h =^{B^2}_\alpha h' \to k =^{B^2}_\beta k' \to h \cdot_d k =^{B^2}_{\alpha \star \beta} h' \cdot_d k'$.

Proof. Given $\gamma : h =^{B^2}_\alpha h'$, and $\theta : k =^{B^2}_\beta k'$, we want to construct a term $\gamma \star_d \theta : h \cdot_d k =^{B^2}_{p \cdot q} h' \cdot_d k'$. We will construct this term by considering a sequence of diagrams (i.e., a diagram chase). First consider the following diagram:



Recall our convention for commutative squares is that the filling path has type $\mathsf{left} \cdot \mathsf{bottom} = \mathsf{top} \cdot \mathsf{right}$. We take the filling path of the commutative triangle to have type $\mathsf{left} = \mathsf{base} \cdot \mathsf{right}$ (though this could be infered from the type of θ). Now note that the left leg of the diagram is definitionally $h \cdot_d k$. Similarly, the right leg is definitionally $h' \cdot_d k'$. Thus, a path filling the whole diagram will have type $h \cdot_d k =^{B^2}_{\alpha \star \beta} h' \cdot_d k'$. So to complete the definition of $\gamma \star_d \theta$, it suffices to fill the middle square marked with "?" and paste the diagram.

In order to fill the middle square we turn to another diagram:



We fill the top hole of the diagram with $\mathsf{r\text{-}coh}^{-1}_{\mathsf{bin\text{-}ap},\mathsf{ap}}$ from lemma 1.4.14. We fill the bottom hole with $\mathsf{r\text{-}unit}^{-1}_{\mathsf{tr}^{(B)^2}(\alpha)(v)}$. This completes the definition of $\gamma \star_d \theta$. □

1.7 Higher Inductive Types

Here we discuss higher inductive types. Though higher inductive types are novel to HoTT, the idea they capture behind is famaliar. A higher inductive type is simply a type freely generated by constructors, exactly how $\mathbb{N}$ is freely generated by a point and an endomorphism. The only difference between HITs and ordinary inductive types is that the we allows for "higher constructors", which let us postulate a type freely genererated by paths. The most basic example as a type generated by a point b_1 and a loop $\mathsf{loop} : \mathsf{b}_1 = \mathsf{b}_1$.

The circle, $\mathbb{S}^1$ is thus defined by the universal property that maps out of the circle $\mathbb{S}^1 \to X$ are equivalent to a pair of a point $x : X$ and a loop $l : x = x$ at x. We can similarly define $\mathbb{S}^2$ and $\mathbb{S}^3$ and freely generated by a 2 and 3-loop, respectively. In general, classifying the behavior and path spaces of higher inductive (like we did for the previous type) is an extremely hard problem. Attempting such a classification is called synthetic homotopy theory, and it is the project that will occupy our attention for the rest of the thesis.

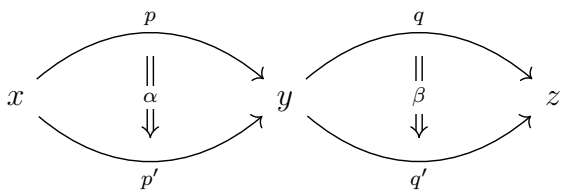
Chapter 2

Higher Paths

In this chapter we outline some of the essential mechanics for working with higher paths in HoTT. We will restrict our attention primarily to 2-paths and 3-paths. We begin by taking a closer look at the algebra of 2-paths and some more intricate coherence paths.

2.1 Horizontal Composition

We will prove some more coherences for the horizontal composition of 2-paths. We will also then show how many of these coherences specialize to 2-loops. These latter coherences play an essential role in the Eckmann-Hilton argument. First conisder arbitrary 2-cells. We want to consider two adjacent 2-cells like so:



In the type theory, this amounts to working in a context

$$(A : U)\ (x, y, z : A)\ (p, p' : x = y)\ (q, q' : y = z)\ (\alpha : p = p')\ (\beta : q = q')$$

We have previously defined $\alpha \star \beta$ as $\mathsf{bin\text{-}ap}(- \cdot -)(\alpha, \beta) : p \cdot q = p' \cdot q'$, where $(- \cdot -) : x = y \to y = z \to x = z$ is the concatenation of paths. Much of the Eckmann-Hilton argument comes from characterizing the behavior of $(- \star -)$, and then applying these characterizations to 2-loops. Since we have defined $(- \star -)$ using binary action on paths, it does not reduce unless both 2-paths are trivial. Thus our first task is characterizing the behavior of $(- \star -)$ when one path is trivial. A horizontal composite with one trivial term, such as $(\alpha \star 1_q)$ or $(1_p \star \beta)$, is called a right or left whiskering, repsectively (the side of the whiskering is the side of the lower dimensional path).

Lemma 1 $(\mathsf{coh}_{\mathsf{ap}(1_x \,\cdot\, -)}, \mathsf{coh}_{\mathsf{ap}(-\,\cdot\, 1_y)})$**.** *For any $p, p' : x = y$ with $\alpha : p = p'$ and $s : z = x$, we have a term $\mathsf{coh}_{\mathsf{ap}(s \,\cdot\, -)} : (s \cdot -)(\alpha) = (1_s \star \alpha)$. For $s : y = z$ we have a term $\mathsf{coh}_{\mathsf{ap}(-\,\cdot\, s)} : (- \cdot s)(\alpha) = (\alpha \star 1_s)$.*

Here we are using $(1_x \cdot -)(\alpha)$ to denote $\mathsf{ap}(1_x \cdot -)(\alpha)$.

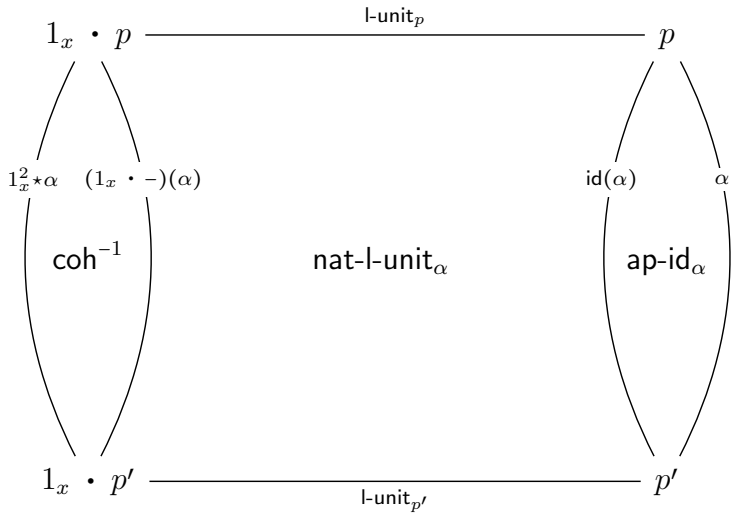
Proof. We prove this by path induction on α. For the first, both sides reduce to $1_{(q \,\cdot\, p)}$ when α is trival. So we set $\mathsf{coh}_{\mathsf{ap}(q \,\cdot\, -)} :\equiv 1^2_{(q \,\cdot\, p)}$ when $\alpha \equiv 1_p$.

Similarly we can set $\mathsf{coh}_{\mathsf{ap}(-\,\cdot\, q)} :\equiv 1^2_p \,\cdot\, _q$ when α is trivial. □

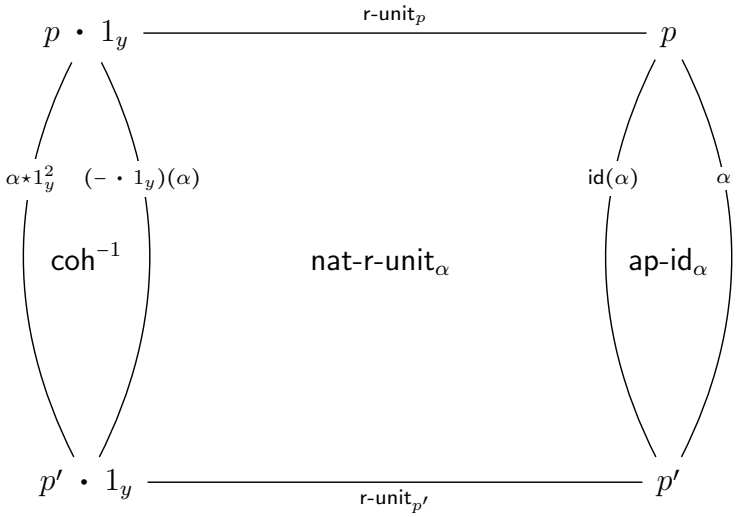
If we whisker by the trivial 2-path, i.e., $(\alpha \star 1^2_y)$, we can further describe the behavior of $(- \star -)$. The reader famliar with category theory may recall that $1^2_x \star \alpha \equiv \alpha$ in the 2-category Cat. But the same is not true in HoTT. The two terms are not even in the same type! The term $(1^2_x \star \alpha)$ is of type $(1_x \cdot p) = (1_x \cdot p')$. But $(1_x \cdot p)$ is not definitionally the same as p. However, we do have a canonical equivalence between these two types induced by $\mathsf{l\text{-}unit}$. We can think of $\mathsf{l\text{-}unit}$ as a homotopy $\mathsf{l\text{-}unit} : (\lambda(p).1_x \cdot p) \simeq id_{x=y}$. Thus our results on the functorial of homtopies applies to $\mathsf{l\text{-}unit}$. So

Lemma 1 ($\mathsf{nat\text{-}l\text{-}unit}_\alpha$, $\mathsf{nat\text{-}r\text{-}unit}_\alpha$)**.** *We have terms* $\mathsf{coh}_{1_x^2 \star -} : (1_x^2 \star \alpha) \cdot \mathsf{l\text{-}unit}_{p'} = \mathsf{l\text{-}unit}_p \cdot \alpha$ *and* $\mathsf{coh}_{- \star 1_y^2} : (\alpha \star 1_y^2) \cdot \mathsf{r\text{-}unit}_{p'} = \mathsf{r\text{-}unit}_p \cdot \alpha$.

Proof. Applying $\mathsf{nat\text{-}htpy}$ to $\mathsf{l\text{-}unit}$, and our previous coherence paths lends the following commutative diagrams:



Here coh denotes $\mathsf{coh}_{\mathsf{ap}(1_x \cdot -)}$ from the preceeding lemma. We paste this diagram to define $\mathsf{coh}_{1_x^2 \star -}$. From $\mathsf{r\text{-}unit}$ we obtain:



Here coh denotes $\mathsf{coh}_{\mathsf{ap}(- \cdot 1_y)}$ from the previous lemma. We paste this diagram to define

$\mathsf{coh}_{-\star 1_y^2}$. □

From here on out we will abuse notation a bit and identify $(- \cdot 1_y)(\alpha)$ with $\alpha \star 1_y^2$. Thus we will write $\mathsf{nat\text{-}r\text{-}unit}_\alpha$ for the path $(\alpha \star 1_y^2) \cdot \mathsf{r\text{-}unit}_{p'} = \mathsf{r\text{-}unit}_p \cdot \alpha$. Similarly for the left unit law. This is our first example of a "two dimensional coherence law". This law is a two dimensional since it only applies to 2-paths and makes essential use of the structure of 2-paths

We can see how this coherence law further simplifies on 2-loops. If α is a 2-loop, then $p \equiv p' \equiv 1_x$. But $\mathsf{l\text{-}unit}_{1_x} \equiv 1_x^2$. Thus, our path $\mathsf{l\text{-}unit}_\alpha$ has type $(1_x^2 \star \alpha) \cdot 1_x^2 = 1_x^2 \cdot \alpha$. But this type is equivalent to $(1_x^2 \star \alpha) = \alpha$. We can obtain a similar result for the right hand side. This leads us to the following lemma.

Lemma 1. *For a 2-loop $\alpha : \Omega^2 X$, we have $\mathsf{l\text{-}unit\text{-}}\Omega^2_\alpha : 1_x^2 \star \alpha = \alpha$ and $\mathsf{r\text{-}unit\text{-}}\Omega^2_\alpha : \alpha \star 1_x^2 = \alpha$.*

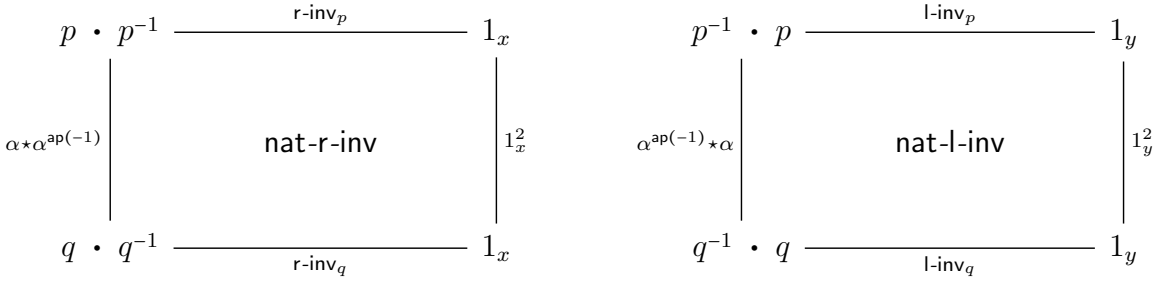
Proof. This follows from the above considerations. □

We can obtain a similar coherence path for inverses. First we need to introduce the "horizontal inverse" of a 2-path.

Definition 0.1 ($(-)^{\mathsf{ap}(-1)}$)**.** *We have function $(-)^{\mathsf{ap}(-1)} : p = q \to p^{-1} = q^{-1}$ for any paths p and q.*

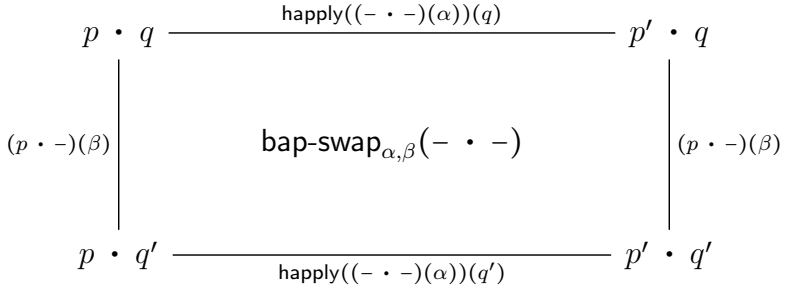
Proof. We use $\mathsf{ap}((-)^{-1})$. □

Lemma 1. *For $\alpha : p = q$, we have*



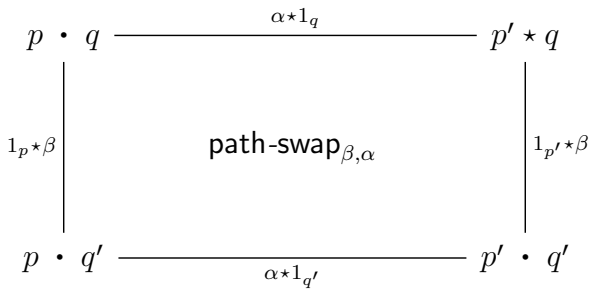
Proof. We think of r-inv as a homotopy $\mathsf{r\text{-}inv} : (-)^{-1} \sim \mathsf{const} 1_x$ and apply nat-htpy, noting that the action on paths of a constant function is always the trivial path. Similarly for l-inv. □

Now we turn our attention the coherence on 2-paths that is at the heart of Eckmnan-Hilton. This coherence follows from the definition of $(- \star -)$ in terms of bin-ap. Due to how it will be used later down the road, we repackage it. We can apply bap-swap to obtain the following diagram:



But we have previously proven that $\mathsf{happly}((- \cdot -)(\alpha))(q) = (\alpha \star 1_q)$ and $(p \cdot -)(\beta) = (1_p \star \beta)$. This leads us to the next lemma.

Lemma 1 (path-swap)**.** *For any 2-paths α and β, we have a commutative diagram*



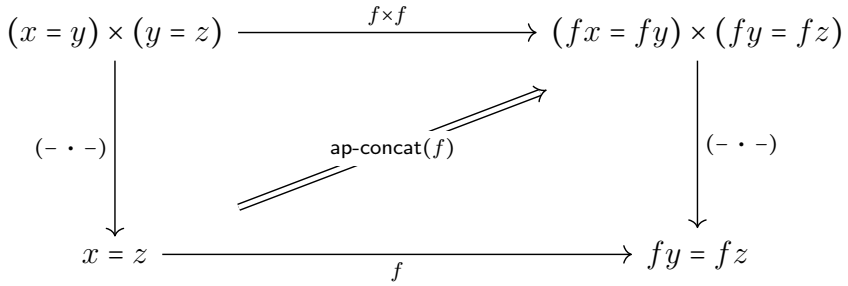
Note that we have reversed the order of variables α and β. This will be convenient later on.

Proof. This follows from piecing togther the above two coherence paths and diagram, using $\mathsf{bap\text{-}swap}_{\alpha,\beta}$ □

Since we defined bap-swap to factor through $(- \cdot -)(\alpha, \beta) \equiv (\alpha \star \beta)$, so does path-swap.

2.2 Higher ap

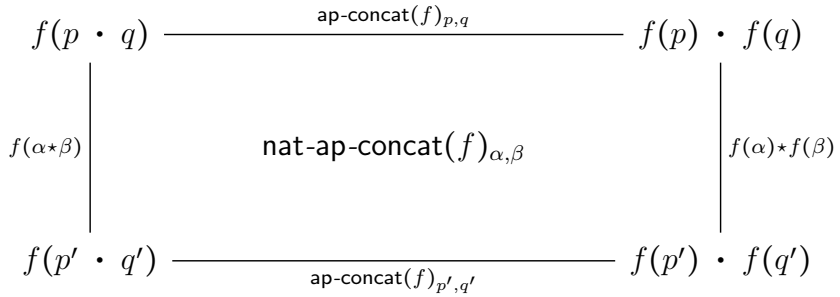
In this section we discuss some of the coherences between functions and higher paths. Each of the functorality properties we proved about a function can be formulated as homotopies. For example, ap-concat can be thought of as a homotopy



As such, it comes with a functorial action on paths, i.e., the function nat-htpy. This functoriality leads to the following theorem.

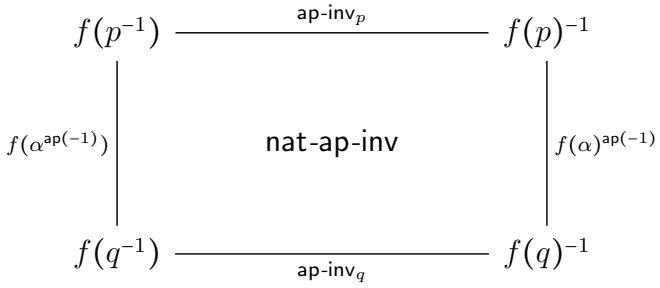
Lemma 0.1 (nat-ap-concat)**.** *Paths fom* $(p, q) = (p', q')$ *in* $(x = y) \times (y = z)$ *are equivalent*

to pairs of paths $(p = p') \times (q = q')$*. For a pair* (α, β) *we have the following commutative diagram*



Proof. This follows from considering ap-concat as a homotopy of $(\lambda(p,q).f(p \cdot q)) \sim (\lambda(p,q).(f(p) \cdot f(q)))$ and applying nat-htpy. The filler can also be defined directly by path induction on α and β. □

Lemma 1. *For* $\alpha : p = p'$ *in* A *we have*



Proof. This follows from thinking of ap-inv as a homotopy and applying nat-htpy. □

In the section on horizontal composition, we proved some higher dimensional coherences between two paths, such as nat-l-unit and nat-l-inv. We can show that these coherences too are preserved every function, though we will not write out full proofs of this.

Chapter 3

Certain Types, A Closer Look

In this chapter we take a closer look at certain types that we will have to work with in the main proof of the thesis. This includes specific constructions, such as taking the fiber of a map, as well as certain type families, like families of functions, and higher inductive types like the spheres. To keep things short, many of the theorems will be stated without proof. However, any omitted proof will be rather trivial, usually relying on just path induction.

3.1 Higher Homotopies

Here we give some lemmas describing how homotopies act on paths. We also give some lemmas describing higher homotopies, that is, homotopies of homotopies.

In our first foray into homotopies (subsection 1.4.5), we gave a characterization of the action on paths of homotopies in terms of a naturality condition. But, since we are working with infinite dimensional structures, homotopies will have an induced naturality condition for each dimension of path. We now turn our attention to unwinding some of the higher naturality conditions of homotopies. Let $H : f \sim g$. Now suppose we have a 2-path $\alpha : p = q$. Both p

and q induce their own naturality condition:

$$\begin{array}{ccc} f(x) & \xrightarrow{f(p)} & f(y) \\ {\scriptstyle H(x)}\downarrow & H(p) & \downarrow{\scriptstyle H(y)} \\ g(x) & \xrightarrow[g(p)]{} & g(y) \end{array} \qquad \begin{array}{ccc} f(x) & \xrightarrow{f(q)} & f(y) \\ {\scriptstyle H(x)}\downarrow & H(q) & \downarrow{\scriptstyle H(y)} \\ g(x) & \xrightarrow[g(q)]{} & g(y) \end{array}$$

The induced naturality condition of α can be seen as giving a cohence between the above two naturality conditions.

$$\begin{array}{ccc} H(x) \cdot g(p) & \xrightarrow{1_{H(x)} \star g(\alpha)} & H(x) \cdot g(q) \\ {\scriptstyle H(p)}\downarrow & H(\alpha) & \downarrow{\scriptstyle H(q)} \\ f(p) \cdot H(y) & \xrightarrow[f(\alpha) \star 1_{H(y)}]{} & f(q) \cdot H(y) \end{array}$$

Lemma 3.1.1. *The above diagram is equivalent to* $\mathsf{apd}^2(H)(\alpha)$.

We can repeat this process for a 3-paths $\gamma : \alpha = \beta$. We derive the following diagram.

$$\begin{array}{ccc} H(p) \cdot (f(\alpha) \star 1_{H(y)}) & \xrightarrow{1_{H(p)} \star (f(\gamma) \square 1^2_{H(y)})} & H(p) \cdot (f(\beta) \star 1_{H(y)}) \\ {\scriptstyle H(\alpha)}\downarrow & H(\gamma) & \downarrow{\scriptstyle H(\beta)} \\ (1_{H(x)} \star g(\alpha)) \cdot H(q) & \xrightarrow[(1_{H(y)} \square g(\gamma)) \star H(q)]{} & (1_{H(y)} \star g(\beta)) \cdot H(q) \end{array}$$

Here, $\square$ denotes the 3-dimensional composition of 3-paths.

Lemma 3.1.2. *The above diagram is equivalent to* $\mathsf{apd}^3(H)(\gamma)$.

3.2 Fibers of Maps

In this section we take a look at important construction called the homotopy fiber of a map. Consider types A and B and a map $f : A \to B$. Fix a point $b : B$. It is natural to wonder what in A gets mapped to b. The answer to this question is provided by the fiber of f over b.

Definition 3.2.1. *The fiber of f over b, denoted $\mathsf{fib}_f(b)$, is defined as*

$$\mathsf{fib}_f(b) :\equiv \sum_{a:A} f(a) = b$$

.

So, the fiber of f over b consists of pairs of a point $a : A$ and a path from $f(a)$ to b. The rest of this section investigates some properties of fibers.

3.2.1 Paths in the fiber

First we characterize paths in the fiber.

Lemma 3.2.1. *Let $(a, p), (a', p') : \mathsf{fib}_f(b)$. Then there are three equivalent descriptions of the paths space $(a, p) = (a', p')$:*

$$\sum_{q:a=a'} p =_q^{\lambda(x).f(x)=b} p'$$

$$\sum_{q:a=a'} p = \mathsf{hpf}(q) \cdot p'$$

$$\mathsf{fib}_{\mathsf{ap}(f)}(p \cdot (p')^{-1}) \equiv \sum_{q:a=a'} f(q) = p \cdot (p')^{-1}$$

We will refer to these characterizations as the standard characterization, the commuting triangle characterization, and the fiber of ap characterization.

Proof. The standard characterization follows directly from theorem 1.5.2 applied to $\mathsf{fib}_f(b)$. The commuting triangle characterization follows from the first by computing $\mathsf{tr}^{\lambda(x).f(x)=b}(q)(p)$ as $f(q)^{-1} \cdot p$ (using path induction) and rearranging inverses. The fiber of ap characterization follows from the second by moving p' to the other side. □

By using the third characterization, we can recursively characterize 2-paths, 3-paths, and so on. Unless otherwise specified, we will implicitly using the commuting triangle characterization of paths in the fiber. The above lemma gives a useful condition for when a loop in the fiber is trivial.

Lemma 3.2.2. *Let* $(a,p) : \mathsf{fib}_f(b)$ *and* $(l,s) : (a,p) = (a,p)$*. Then* $(l,s) = 1_{(a,p)}$ *if* s *is in the image of* f*, i.e., there is a path* $\alpha : l = 1_a$ *such that* $f(\alpha) = s$*.*

Proof. This follows from applying the fiber of ap characteriziation to the space of 2-paths. □

3.2.2 The family fib

Now we turn our attention to constructing fiber bundles and returning to some claims from sections 1.3 and 1.5. Any map $f : A \to B$ induces a type family over B such that the total space of this family is equivalent to A. This fact is classically known as "every map is equivalent to a fibration". This is the family of fibers of f, which sends a point $b : B$ to the type $\mathsf{fib}_f(b)$. Now we characterize some of the higher dimensional data in the family fib_f. It is easy to compute transport on a 1-path.

Lemma 3.2.3. *For* $q : x = y$ *in* B *and* $(z,p) : \mathsf{fib}(x)$*, we have* $\mathsf{tr}^{\mathsf{fib}}(q)(z,p) = (z, p \cdot q)$*. I.e., we have a homotopy* $\mathsf{tr}^{\mathsf{fib}}(q) \sim \lambda(z,p).(z, p \cdot q)$*.*

Proof. By path induction on q, we need to construct a term $(z,p) = (z, p \cdot 1_x)$. We can use $(1_z, \mathsf{r\text{-}unit}_p^{-1})$. □

Now we wish to characterize transport on a 2-path.

Lemma 3.2.4. *For a 2-path $\alpha : q = q'$ in B we have a commutative square of homotopies*

$$\begin{array}{ccc} \mathsf{tr}^{\mathsf{fib}}(q) & \longrightarrow & \lambda(z,p).(z, p \cdot q) \\ {\scriptstyle \mathsf{tr}^{(\mathsf{fib})^2}(\alpha)}\downarrow & & \downarrow{\scriptstyle \lambda(z,p).(1_z, 1_p \star \alpha)} \\ \mathsf{tr}^{\mathsf{fib}}(q') & \longrightarrow & \lambda(z,p).(z, p \cdot q') \end{array}$$

where the horizontal homotopies are from the above lemma.

Proof. We prove this by path induction on α. But $\mathsf{tr}^{(\mathsf{fib})^2}(1_q) \equiv \mathsf{refl\text{-}htpy}$. Additionally, since we have assumed $q \equiv q'$ in the path induction, the horizontal homotopies cancel. Thus our goal is equivalent to constructing a term $\mathsf{refl\text{-}htpy} \sim \lambda(z,p).(1_z, 1_p \star 1_q)$, which is clearly true. □

The next lemma characterizes the naturality condition of the homotopy $\lambda(z,p).(1_z, 1_p \star \alpha)$. This lemma plays a fundamental role in our construction of the Hopf fibration.

Lemma 3.2.5. *For a path $(1_z, \beta) : (z,p) = (z,p')$ in the fiber (using the stardard characterization, so that $\beta : p = p'$), we have that the naturality condition of above homotopy induced by β is*

$$\begin{array}{ccc} (z, p \cdot q) & \xrightarrow{(1_z, \beta \star 1_q)} & (z, p' \cdot q) \\ {\scriptstyle (1_z, 1_p \star \alpha)}\downarrow & \mathsf{path\text{-}swap}_{\alpha,\beta} & \downarrow{\scriptstyle (1_z, 1_p \star \alpha)} \\ (z, p \cdot q') & \xrightarrow[(1_z, \beta \star 1_{q'})]{} & (z, p' \cdot q') \end{array}$$

Proof. By path induction on β. □

Part II

Eckmann-Hilton and The Hopf Fibration

Now we turn our attention to the main result of the thesis: showing that the Eckmann-Hilton argument can be used to construct a generator of $\Omega^3(\mathbb{S}^2)$ (up to sign). We will show this by constructing a map $\mathsf{hpf} : \mathbb{S}^3 \to \mathbb{S}^2$ such that: (i) the fiber of hpf is $\mathbb{S}^1$ and (ii) the image $\Omega^3(\mathsf{hpf})(\mathsf{surf}_3)$ of the generating loop of $\mathbb{S}^3$ is a path induced by Eckmann-Hilton. Before diving into the proof, we first give a conceptual sketch for how the proof will proceed and how this part of the thesis is organized.

In our construction of the Hopf fibration, there are three essential parts of the proof: (i) constructing a fibration over $\mathbb{S}^2$, (ii) proving that the total space of this fibration is $\mathbb{S}^3$, and (iii) proving that the fiber over one of the poles of $\mathbb{S}^2$ is $\mathbb{S}^1$. All results related to homotopy groups and loop spaces follow from considering the fiber sequence of the Hopf fibration. Clearly one ought to optimize their construction of the fibration to make characterizing its total space and fibers as pain free as possible. Thus there are two main strategies for constructing the fibration: either construct it so that its obvious the fiber is $\mathbb{S}^1$ or construct it so that its obvious that the total space is $\mathbb{S}^3$.

The HoTT book's construction of the Hopf fibration, the authors take the first route. They construct the Hopf fibration by defining a type family $\mathcal{H} : \mathbb{S}^2 \to U$ with $\mathcal{H}(\mathsf{N}_2) :\equiv \mathbb{S}^1$. From this construction it is immediate that the fibration induced by $\mathcal{H}$ has fiber $\mathbb{S}^1$. The rest of the HoTT book's proof consists in showing that $\sum_{x:\mathbb{S}^2} \mathcal{H}(x) \simeq \mathbb{S}^3$. Once this equivalence is constructed, they define the map $\mathbb{S}^3 \to \mathbb{S}^2$ as the composite $\mathbb{S}^3 \simeq \sum_{x:\mathbb{S}^2} \mathcal{H}(x) \to \mathbb{S}^2$.

Our construction of the Hopf fibration goes the opposite route. We will construct the Hopf fibration by first constructing a term $\mathsf{eh} : \Omega^3(\mathbb{S}^2)$ using the Eckmann-Hilton argument. We will call this term the Eckmann-Hilton 3-loop. Then we can use the suspension loop space adjunction, specifically the equivalence $\Omega^3(\mathbb{S}^2) \simeq (\mathbb{S}^3 \to \mathbb{S}^2)$, to obtain a map $\mathsf{hpf} : \mathbb{S}^3 \to \mathbb{S}^2$ with $\mathsf{hpf}(\mathsf{N}_3) \equiv \mathsf{N}_2$ and $\mathsf{hpf}(\mathsf{surf}_3) = \mathsf{eh}$. Thus it is immediate from our construction of the Hopf fibration that its total space is $\mathbb{S}^3$. So the bulk of the proof lies in characterizing the fibers of the map hpf.

It is sufficent to prove that the fiber of hpf over either of the poles is $\mathbb{S}^1$. However, we instead prove something slightly stronger. We will characterize the entire type family $\mathsf{fib} :\equiv (\lambda(x).\mathsf{fib}_{\mathsf{hpf}}(x)) : \mathbb{S}^2 \to U$. That is, we will construct a fiberwise equivalence between the total spaces of fib and another family over $\mathbb{S}^2$. This is indeed a stronger result since it will also give an equivalence between the higher dimensional data in each family. The reason we choose to prove a stronger claim is not solely to get a stronger result. In fact the reason we prove a stronger claim is a familiar one: it is often easier to prove a more general claim in HoTT since this may allow us to use more induction principles.

The characterization of fib naturally breaks up into four sections. First we will flesh out some of the properties of the type family fib. The goal of this section is to gain enough understanding of the behavior of fib so that we can motivate the construction of the type family $\mathcal{H}$ in the second section. Then we will construct a fiberwise equivalence $\sum_{x:\mathbb{S}^2} \mathcal{H}(x) \simeq \sum_{x:\mathbb{S}^2} \mathsf{fib}(x)$. This will imply that the fiber of hpf over the pole of $\mathbb{S}^2$ is $\mathbb{S}^1$, providing everything necessary for the analysis of the fiber sequence of hpf.

The construction of the fiberwise equivalence will itself be broken up into three sections: (i) the construction of the forwards map $g : \prod_{x:\mathbb{S}^2} \mathcal{H}(x) \to \mathsf{fib}(x)$, (ii) construction of the backwards map $f : \prod_{x:\mathbb{S}^2} \mathsf{fib}(x) \to \mathcal{H}(x)$, and (iii) proving these maps are fiberwise inverses of each other.

We begin by reviewing the Eckmann-Hilton argument and constructing the Eckmann-Hilton 3-loop.

Chapter 4

Eckmann-Hilton

The Eckmann-Hilton argument is often first introduced in one of two ways. The first way states that any monoid object in the category of monoids is a commutative monoid. The second states that the composition of paths in any double loop space is commutative. The second version of the Eckmann-Hilton argument, the verison we will be concerned with, usually shows up when proving that all higher homotopy groups are abelian. The proof usually proceeds taking two 2-loops in a space X and constructing a homotopy of paths between $p \cdot q$ and $q \cdot p$ in the space. Then we conlude that $[p] * [q] = [q] * [p]$ in the homotopy group. The second version of Eckmann-Hilton is also often a theorem in a first course on category theory: given a category C, the vertical composition of natural transformations in $\mathsf{Cat}(C, C)(1_C, 1_C)$ is commutative.

Of course the two versions of Eckmann-Hilton are essentially the same. We can see the second version of the Eckmann-Hilton argument as a special case of the first. Sticking with the category theory example, the idea behind the proof is that we naturally have two different types of composition operations on natural transformations, veritical and horizontal composition. In general, these two types of composition take different forms of pasting diagrams as their input. But, when we fix a category C and consider the natural trans-

formations $1_C \Rightarrow 1_C$, both types of composition define operations from pairs $1_C \times 1_C \to 1_C$. Further, the composition laws on natural transformations imply that vertical composition is a monoid homomorphism of horizontal composition. We can also see the first version of the Eckmann-Hilton argument as a special case of the second since any commutative monoid lends a 2-category with one object and one 1-morphism. Since the two versions of the argmuent can rightly be considered the same, we will from here on out only consider the second version and refer to it simply as "the Eckmann-Hilton" argument.

Certain aspects of the Eckmann-Hilton argument become more salient in Homotopy Type Theory. In the context of HoTT, the Eckmann-Hilton argument states that, for any type $X : U$, given 2-loops $\alpha, \beta : \Omega^2(X)$ we have that $\alpha \cdot \beta = \beta \cdot \alpha$. But we cannot simply assert equality in HoTT; really we need to specify a 3-path $\mathsf{EH}(\alpha, \beta) : \alpha \cdot \beta = \beta \cdot \alpha$. We do not have to worry about this in the world of 2-categories since all 3-cells are trivial.

In this chapter we will review the Eckmann-Hilton argument and prove some properties of EH. Then we will be able to use this to construct a 3-loop eh in $\mathbb{S}^3$ by taking $\mathsf{EH}(\mathsf{surf}_2, \mathsf{surf}_2^{-1}) : \mathsf{surf}_2 \cdot \mathsf{surf}_2^{-1} = \mathsf{surf}_2^{-1} \cdot \mathsf{surf}_2$ and applying inverse laws. We will then prove some properties of eh.

4.1 The Eckmann-Hilton Argument

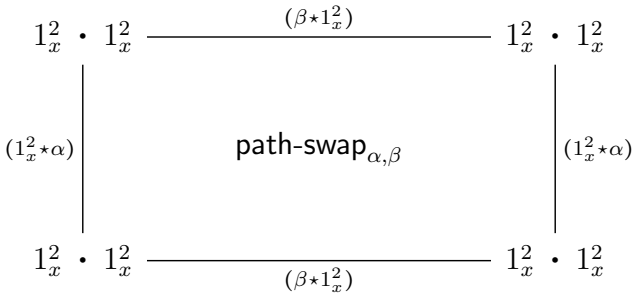
Now we give the Eckmann-Hilton argument. Our goal is to construct a term $\mathsf{EH}(\alpha, \beta) : \alpha \cdot \beta = \beta \cdot \alpha$, for any 2-loops α and β. We cannot do so by path induction on α and β since we are not proving something about all paths, but about solely loops. In other words, α and β have both of their endpoints stuck so we can cannot contract them away. So the way we construct EH is by considering lemmas about arbitary 2-paths and showing how these lemmas specialize to 2-loops.

We have already seen all the lemmas necessary for EH, we need only apply them to 2-loops

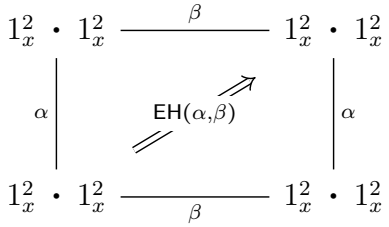
and piece them together.

Lemma 4.1.1 (Eckmann-Hilton, EH)**.** *Let $A : U$ and $a : A$. Consider two 2-loops $\alpha, \beta : \Omega^2(A, a)$. Then we have a path $\mathsf{EH}(\alpha, \beta) : \alpha \cdot \beta = \beta \cdot \alpha$. In other words, we have a homotopy $\mathsf{EH} : (\lambda(\alpha, \beta).\alpha \cdot \beta) \sim (\lambda(\alpha, \beta).\beta \cdot \alpha)$.*

Proof. Let $\alpha, \beta : \Omega^2(A, a)$. We can apply path-swap to α and β to obtain



Of course $1_x^2 \cdot 1_x^2 \equiv 1_x^2$. Now we can use our paths $\mathsf{l\text{-}unit\text{-}}\Omega^2_\beta : 1_x^2 \star \beta = \beta$ and $\mathsf{r\text{-}unit\text{-}}\Omega^2_\alpha : 1_x^2 \star \alpha = \alpha$ to obtain



We can write out the explicit formula for $\mathsf{EH}(\alpha, \beta)$ as:

$$
\begin{array}{c}
\alpha \cdot \beta \\
\Big| \\
{\scriptstyle (\mathsf{l\text{-}unit\text{-}}\Omega^2_\alpha)^{-1} \star (\mathsf{r\text{-}unit\text{-}}\Omega^2_\beta)^{-1}} \\
\Big| \\
(1^2_x \star \alpha) \cdot (\beta \star 1^2_x) \\
\Big| \\
{\scriptstyle \mathsf{path\text{-}swap}_{\alpha,\beta}} \\
\Big| \\
(\beta \star 1^2_x) \cdot (1^2_x \star \alpha) \\
\Big| \\
{\scriptstyle (\mathsf{r\text{-}unit\text{-}}\Omega^2_\beta) \star (\mathsf{l\text{-}unit\text{-}}\Omega^2_\alpha)} \\
\Big| \\
\beta \cdot \alpha
\end{array}
$$

This completes the Eckmann-Hilton argument. □

Since $\mathsf{path\text{-}swap}_{\alpha,\beta}$ factors through $\alpha \star \beta$, we can easily extract from our above proof a proof that $\alpha \cdot \beta = \alpha \star \beta$.

4.1.1 Properties of EH

Before moving on, we will prove a few lemmas about the construction EH.

Suppose we have a $\alpha, \beta : \Omega^2(A, a)$ and a function $f : A \to B$. It is natural to wonder how $f(\mathsf{EH}(\alpha, \beta))$ relates to $\mathsf{EH}(f(\alpha), f(\beta))$. The two terms are not of the same type, since the first has type $f(\alpha \cdot \beta) = f(\beta \cdot \alpha)$ and the second has type $f(\alpha) \cdot f(\beta) = f(\beta) \cdot f(\alpha)$, so we cannot ask for the two terms to be equal. But we have canonical equivalence between these two types via $\mathsf{ap\text{-}concat}(f)_{\alpha,\beta}$. Thus we can sensibly ask for the following diagram commute:

$$\begin{array}{ccc} f(\alpha \cdot \beta) & \xrightarrow{\mathsf{ap\text{-}concat}(f)_{\alpha,\beta}} & f(\alpha) \cdot f(\beta) \\ \Big\downarrow{\scriptstyle f(\mathsf{EH}(\alpha,\beta))} & ? & \Big\downarrow{\scriptstyle \mathsf{EH}(f(\alpha),f(\beta))} \\ f(\beta \cdot \alpha) & \xrightarrow[\mathsf{ap\text{-}concat}(f)_{\beta,\alpha}]{} & f(\beta) \cdot f(\alpha) \end{array}$$

In order to prove this, we first need one lemma.

Lemma 4.1.2. *For arbitary 2-paths α and β, the following diagram commutes:*

$$\begin{array}{ccc} f((\alpha \star 1_q) \cdot (1_{p'} \star \beta)) & \xrightarrow{\mathsf{ap\text{-}concat}_{(\alpha\star 1_q),(1_{p'}\star\beta)}} & f(\alpha \star 1_x^2) \cdot f(1_x^2 \star \beta)) \\ \Big\downarrow{\scriptstyle f(\mathsf{bap\text{-}swap}_{\alpha,\beta})} & & \Big\downarrow{\scriptstyle \mathsf{bap\text{-}swap}_{f(\alpha),f(\beta)}} \\ f((1_p \star \beta) \cdot (\alpha \star 1_{q'})) & \xrightarrow[\mathsf{ap\text{-}concat}_{(1_q\star\beta),(\alpha\star 1_{p'})}]{} & f(1_x^2 \star \beta) \cdot f(\alpha \star 1_x^2)) \end{array}$$

Proof. We may induct on α and β. Then everything reduces to the trivial path. □

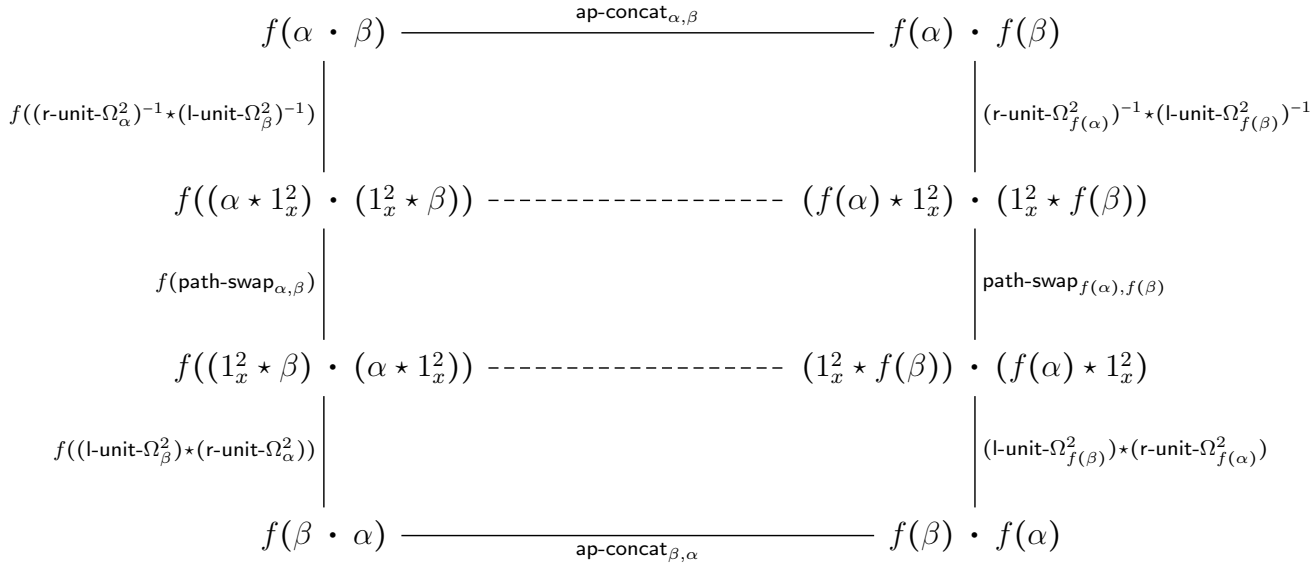
Now we prove that functions preserve EH.

Lemma 4.1.3. *We have a term* $\mathsf{ap\text{-}EH} : f(\mathsf{EH}(\alpha,\beta)) \cdot \mathsf{ap\text{-}concat}(f)_{\beta,\alpha} = \mathsf{ap\text{-}concat}(f)_{\alpha,\beta} \cdot \mathsf{EH}(f(\alpha), f(\beta))$

Since $\mathsf{EH}(\alpha,\beta)$ is composite of paths, $\mathsf{ap\text{-}concat}$ twice lend a path:

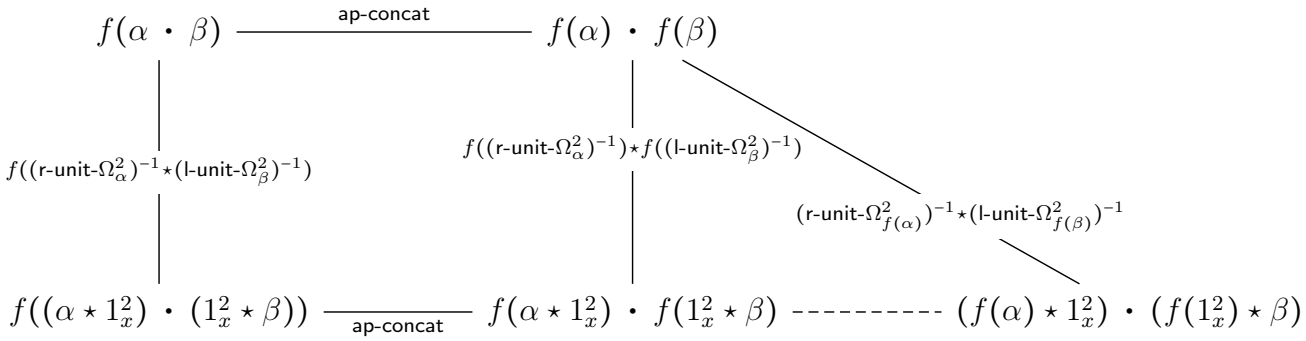
$$\begin{array}{ccc} & & f(\alpha \cdot \beta) \\ & {\scriptstyle f((\mathsf{r\text{-}unit}\text{-}\Omega^2_\alpha)^{-1}\star(\mathsf{l\text{-}unit}\text{-}\Omega^2_\beta)^{-1})} \nearrow & \\ f((\alpha \star 1_x^2) \cdot (1_x^2 \star \beta)) & & \Big\downarrow{\scriptstyle f(\mathsf{EH}(\alpha,\beta))} \\ \Big\downarrow{\scriptstyle f(\mathsf{path\text{-}swap})} & & \\ f((1_x^2 \star \beta) \cdot (\alpha \star 1_x^2)) & \searrow {\scriptstyle f((\mathsf{l\text{-}unit}\text{-}\Omega^2_\beta)\star(\mathsf{r\text{-}unit}\text{-}\Omega^2_\alpha))} & \\ & & f(\beta \cdot \alpha) \end{array}$$

Thus our goal is equivalent to making the following diagram commute:

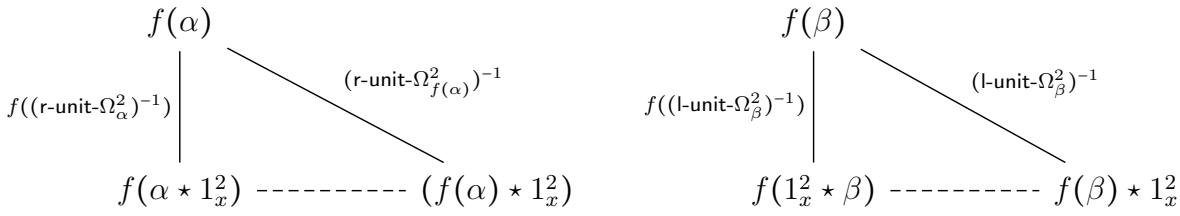


The right side of the diagram is definitionally $\mathsf{EH}(f(\alpha), f(\beta))$. This diagram naturally splits up into three subdiagrams (along the dotted lines, though we have yet to construct these dotted paths). We will make the whole diagram commute by constructing a filler for each subdiagram and pasting.

The middle subdiagram commutes by our previous lemma. So we only have to contruct fillers for the top and bottom subdiagrams. We construct the filler of the top subdiagram and ommit the proof for the bottom, since it is very similar. We can split up the top diagram up as



The left square commutes via nat-ap-concat. Now we will construct the dotted path and filler for the triangle. To make this trange commute we will construct two commutative triangles:



The commutativity of the original triangle will follow from the commutativity of these two triangles the functoriality of $(- \star -)$. The commutativity of each triangle follows from some simple lemmas about the preservation of nat-r-unit and nat-l-unit under the image of a function. Thus the top subdiagram commutes. Constructing the filler of the bottom subdiagram is exactly analgous.

4.2 The Eckmann-Hilton Loop

In this section we use Eckmann-Hilton to contruct a 3-loop and prove some properties of this 3-loop. Let $\alpha : \Omega^2 X$ be a 2-loop. Then we can apply EH to α and its inverse. This lends

a path $\alpha \cdot \alpha^{-1} = \alpha^{-1} \cdot \alpha$. We can use inverse laws to tie off the ends an obtain a loop one dimension above α. Thus, we define

$$\mathsf{eh}_\alpha :\equiv \mathsf{r\text{-}inv}_\alpha^{-1} \cdot \mathsf{EH}(\alpha, \alpha^{-1}) \cdot \mathsf{l\text{-}inv}_\alpha$$

Explicitly, we have:

$$\begin{array}{c}
1_x^2 \\
\Big| \; \mathsf{r\text{-}inv}_\alpha^{-1} \\
\alpha \cdot \alpha^{-1} \\
\Big| \; (\mathsf{l\text{-}unit\text{-}}\Omega^2_\alpha)^{-1} \star (\mathsf{r\text{-}unit\text{-}}\Omega^2_{\alpha^{-1}})^{-1} \\
(1_x^2 \star \alpha) \cdot (\alpha^{-1} \star 1_x^2) \\
\Big| \; \mathsf{path\text{-}swap}_{\alpha,\alpha^{-1}} \\
(\alpha^{-1} \star 1_x^2) \cdot (1_x^2 \star \alpha) \\
\Big| \; (\mathsf{r\text{-}unit\text{-}}\Omega^2_{\alpha^{-1}}) \star (\mathsf{l\text{-}unit\text{-}}\Omega^2_\alpha) \\
\alpha^{-1} \cdot \alpha \\
\Big| \; \mathsf{l\text{-}inv}_\alpha \\
1_x^2
\end{array}$$

We will call a 3-loop of this from an Eckmann-Hilton 3-loop and may denote it by eh_α.

4.2.1 Properties

Now we show that the eh construction is preserved by functions.

Lemma 4.2.1. *Let $f : X \to Y$ and $\alpha : \Omega^2(X)$. Then $f(\mathsf{eh}_\alpha) = \mathsf{eh}_{f(\alpha)}$.*

Proof. This follows from our previous lemma constructing ap-EH and some simple lemmas demonstrating the preservation of r-inv and l-inv under functions. □

4.2.2 Eckmann-Hilton in $\mathbb{S}^2$

Now we introduce the Eckmann-Hilton 3-loop that will occupy our attention for the remainder of the thesis. Since we have surf_2 in $\Omega^2(\mathbb{S}^2)$, we have an induced $\mathsf{eh}_{\mathsf{surf}_2} : \Omega^3(\mathbb{S}^2)$. Since we are now concerned with only one Eckmann-Hilton loop, we will drop the subscript and denote this by just eh. We can write out eh as

$$
\begin{array}{c}
1^2_{\mathsf{N}_2} \\
\Big| \; \mathsf{r\text{-}inv}^{-1}_{\mathsf{surf}_2} \\
\mathsf{surf}_2 \cdot \mathsf{surf}_2^{-1} \\
\Big| \; (\mathsf{l\text{-}unit\text{-}}\Omega^2_{\mathsf{surf}_2})^{-1} \star (\mathsf{r\text{-}unit\text{-}}\Omega^2_{\mathsf{surf}_2^{-1}})^{-1} \\
(1^2_x \star \mathsf{surf}_2) \cdot (\mathsf{surf}_2^{-1} \star 1^2_x) \\
\Big| \; \mathsf{path\text{-}swap}_{\mathsf{surf}_2, \mathsf{surf}_2^{-1}} \\
(\mathsf{surf}_2^{-1} \star 1^2_x) \cdot (1^2_x \star \mathsf{surf}_2) \\
\Big| \; (\mathsf{r\text{-}unit\text{-}}\Omega^2_{\mathsf{surf}_2^{-1}}) \star (\mathsf{l\text{-}unit\text{-}}\Omega^2_{\mathsf{surf}_2}) \\
\mathsf{surf}_2^{-1} \cdot \mathsf{surf}_2 \\
\Big| \; \mathsf{l\text{-}inv}_{\mathsf{surf}_2} \\
1^2_{\mathsf{N}_2}
\end{array}
$$

Chapter 5

The Hopf Fibration

In this chapter we use $\mathsf{eh} : \Omega^3(\mathbb{S}^2)$ to construct a fibration $\mathsf{hpf} : \mathbb{S}^3 \to \mathbb{S}^2$. The construction is immediate from the suspension loop space adjunction. Thus most of the chapter is spent investigating and characterizing the fiber of hpf. The proof that the fiber of hpf is $\mathbb{S}^1$ is in the next chapter.

5.1 The map $\mathsf{hpf} : \mathbb{S}^3 \to \mathbb{S}^2$

First we define the map $\mathsf{hpf} : \mathbb{S}^3 \to \mathbb{S}^2$ and engage in a preliminary investigation of its fiber. The definition of hpf is short and sweet.

Definition 5.1.1 (The Hopf Fibration, hpf)**.** *We define a map* $\mathsf{hpf} : \mathbb{S}^3 \to \mathbb{S}^2$ *such that* $\mathsf{hpf}(\mathsf{surf}_3) = \mathsf{eh}$

Proof. By the universal property of $\mathbb{S}^2$, we have that $(\mathbb{S}^3 \to \mathbb{S}^2) \simeq \Omega^3(\mathbb{S}^2)$. We choose $\mathsf{eh} : \Omega^3(\mathbb{S}^2)$ to define hpf. Thus it is immediate from the definition that $\mathsf{hpf}(\mathsf{surf}_3) = \mathsf{eh}$. □

5.1.1 The Fiber of hpf

Now we turn our attention to investigating the fiber of hpf and setting the stage for the rest of the proof. In this section we will draw heavily on section 3.2 on the fibers of maps. The reader not familiar with the homotopy fiber of a map, or the incarnation of this construction in HoTT, is advised to at least skim that section over before reading on. But we will make an effort to reiterate enough of the information of 3.2 so that a reader already familiar with the fiber of a map can get by just reading this section. First we review some results from that section and make some remarks specific to hpf. The fiber of hpf over a point $x : \mathbb{S}^2$ consists of pair of a point $z : \mathbb{S}^3$ and a path $\mathsf{hpf}(z) = x$. This is defined as

$$\mathsf{fib}_{\mathsf{hpf}}(x) :\equiv \sum_{z:\mathbb{S}^3} \mathsf{hpf} z = x$$

By our characterization of paths in the fiber in 3.2.1, we have three equivalent forms for the path space in the fiber $(z, p) = (z', p')$, namely

$$\sum_{q:z=z'} p =^{\mathsf{fib}}_{q} p'$$

$$\sum_{q:z=z'} p = \mathsf{hpf}(q) \cdot p'$$

$$\mathsf{fib}_{\mathsf{hpf}}(p \cdot (p')^{-1})$$

We will refer to these as the transport characterization, the commuting triangle characterization, and the fiber of ap characterization, respectively. Usually one ought to use one of the last two characterizations of paths in a fiber. But, in the course of characterizing fib, we will find that we only ever have occasion have to work in the fiber $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$ and with

paths of the form $(1_{\mathsf{N}_3}, \alpha)$, i.e., whose first compent is 1_{N_3}. In this case, it is best to work with the first characterization since the the second component definitionally simplifies to $p = p'$. Thus, unless otherwise specified, we will be implicitly using the first characterization throughout the proof. We will also have to work with 2-paths in the fiber, to which similar remarks apply. To avoid confusion, it is worth pointing out that a 1-path in the fiber involves a 2-path in $\mathbb{S}^2$ and a 2-paths in the fiber involves a 3-path in $\mathbb{S}^2$. This is because points in the fiber involve a 1-path in $\mathbb{S}^2$.

Lets take a brief look at some points and paths in the fiber of hpf to see why one might expect that the fiber is $\mathbb{S}^1$. By definition, $\mathsf{hpf}(\mathsf{N}_3) \equiv \mathsf{N}_2$. Thus we have a canonical point $(\mathsf{N}_3, 1_{\mathsf{N}_2})$ in $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$. In fact, we can't really scrounge up another point; this is the only point we can construct in an empty context (i.e., without other assumptions). We can picture this point as

$$\mathsf{N}_3$$

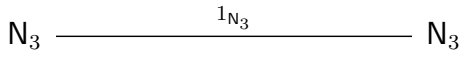
$$\begin{array}{c} \mathsf{hpf}(\mathsf{N}_3) :\equiv \mathsf{N}_2 \\ \Big| \\ {\scriptstyle 1_{\mathsf{N}_2}} \\ \Big| \\ \mathsf{N}_2 \end{array}$$

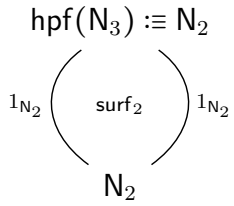
Now consider a loop at this point. A loop has the form

$$\sum_{q:\mathsf{N}_3 = \mathsf{N}_3} 1_{\mathsf{N}_2} =^{\mathsf{fib}}_{q} 1_{\mathsf{N}_2}$$

But $\mathbb{S}^3$ doesn't have any interesting paths of type $\mathsf{N}_3 = \mathsf{N}_3$. Thus we are almost forced to fix $q \equiv 1_{\mathsf{N}_3}$. Now we only need a path $1_{\mathsf{N}_2} = 1_{\mathsf{N}_2}$. The canonical choices are of the form surf_2^n, for each $n : \mathbb{Z}$. Of course, we want to pick a generator, thus we should limit out attention to surf_2

and surf_2^{-1}. Choosing the first for now, we can picture this loop in the fiber as follows:





Of course, just because we can define a loop does not mean its non-trivial. We can ask if this loop is trivial by asking for a 2-path in the fiber $(1_{\mathsf{N}_3}, \mathsf{surf}_2) = (1_{\mathsf{N}_3}, 1^2_{\mathsf{N}_2})$. But, using the fiber of ap characterization of paths in the fiber, such a path must be a term of $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{surf}_2)$. Thus, asking for the loop $(1_{\mathsf{N}_3}, \mathsf{surf}_2)$ to be trivial is equivalent to asking for surf_2 to be in the image of hpf. Since surf_2 is not in the image of hpf (or, at least, so we expect to be the case), indeed the loop we have selected should be non-trivial. Similar reasoning will lead us to believe that $(1_{\mathsf{N}_3}, \mathsf{surf}_2^2)$, and $(1_{\mathsf{N}_3}, \mathsf{surf}_2^n)$ for any $n : \mathbb{Z}$, are all non-trivial. This is just a special case of some reasoning from lemma 3.2.2 on the conditions for when a loop in the fiber is non-trivial. This gives us a reason to expect (not just hope) that the fiber of hpf is equivalent to $\mathbb{S}^1$.

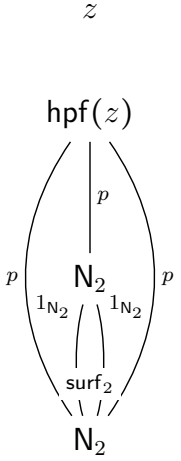
But this does not suffice to characterize the fiber, of course. We will actually have to exhibit an equivalence between the fiber and $\mathbb{S}^1$. As we mentioned, we will not be constructing an equivalence between $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$ and $\mathbb{S}^1$ directly, but will instead give a fiberwise equivalence between fib and a yet to be constructed $\mathcal{H}$. Thus, it is in our best interest to have a good understanding of the family fib, and not just $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$. This will help us in our selection of $\mathcal{H}$, as well as in the construction of the fiberwise equivalence.

Since fib is a family over $\mathbb{S}^2$, it is uniquely determined by its descent data. This data comprises where fib sends N_2 and the structure surf_2 induces, i.e., $\mathsf{tr}^{(\mathsf{fib})^2}(\mathsf{surf}_2) : \mathsf{id} \sim \mathsf{id}$. Thus, to

get good enough understanding of fib, we need only focus on understanding $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$ and $\mathsf{tr}^{(\mathsf{fib})^2}(\mathsf{surf}_2)$. We have just given a preliminary characterization of $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$. So now we need to characterize $\mathsf{tr}^{(\mathsf{fib})^2}(\mathsf{surf}_2)$. We already have a general characterization of $\mathsf{tr}^{(\mathsf{fib})^2}$ from section 3.2 on fibers of maps. In particular, lemma 3.2.4 gives a commutative square relating $\mathsf{tr}^{(\mathsf{fib})^2}(\mathsf{surf}_2)$ and $\lambda(z,p).(1_z, 1_p \star \mathsf{surf}_2)$. Since $\mathsf{surf}_2 : 1_{\mathsf{N}_2} = 1_{\mathsf{N}_2}$, there are some further simplifications we can make. We can package them into the following commutative square:

$$\begin{array}{ccc}
\mathsf{id} \equiv \mathsf{tr}^{\mathsf{fib}}(1_{\mathsf{N}_2}) & \xrightarrow{\lambda(z,p).(1_z, \mathsf{r\text{-}unit}_p)} & \lambda(z,p).(z, p \cdot 1_{\mathsf{N}_2}) \\
\Big\downarrow{\scriptstyle \mathsf{tr}^{(\mathsf{fib})^2}(\mathsf{surf}_2)} & & \Big\downarrow{\scriptstyle \lambda(z,p).(1_z, 1_p \star \mathsf{surf}_2)} \\
\mathsf{id} \equiv \mathsf{tr}^{\mathsf{fib}}(1_{\mathsf{N}_2}) & \xrightarrow[\lambda(z,p).(1_z, \mathsf{r\text{-}unit}_p)]{} & \lambda(z,p).(z, p \cdot 1_{\mathsf{N}_2})
\end{array}$$

We can write the explicit values for the horizontal homotopies since they were defined by path induction and reduce on 1_{N_2}. On a point (z,p) in the fiber, we can picture the action of $\mathsf{tr}^{(\mathsf{fib})^2}(\mathsf{surf}_2)$ as follows:



Thus we arrive at the following characterization of the desencent data of fib.

Lemma 5.1.1 (descent data in fib)**.** *We have that* $\mathsf{fib}(\mathsf{N}_2) :\equiv \mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$ *and* $\mathsf{tr}^{(\mathsf{fib})^2}(\mathsf{surf}_2) \sim \lambda(z,p).(1_z, \mathsf{r\text{-}unit}_p^{-1} \cdot (1_p \star \mathsf{surf}_2) \cdot \mathsf{r\text{-}unit}_p)$.

Proof. This follows from the above considerations. □

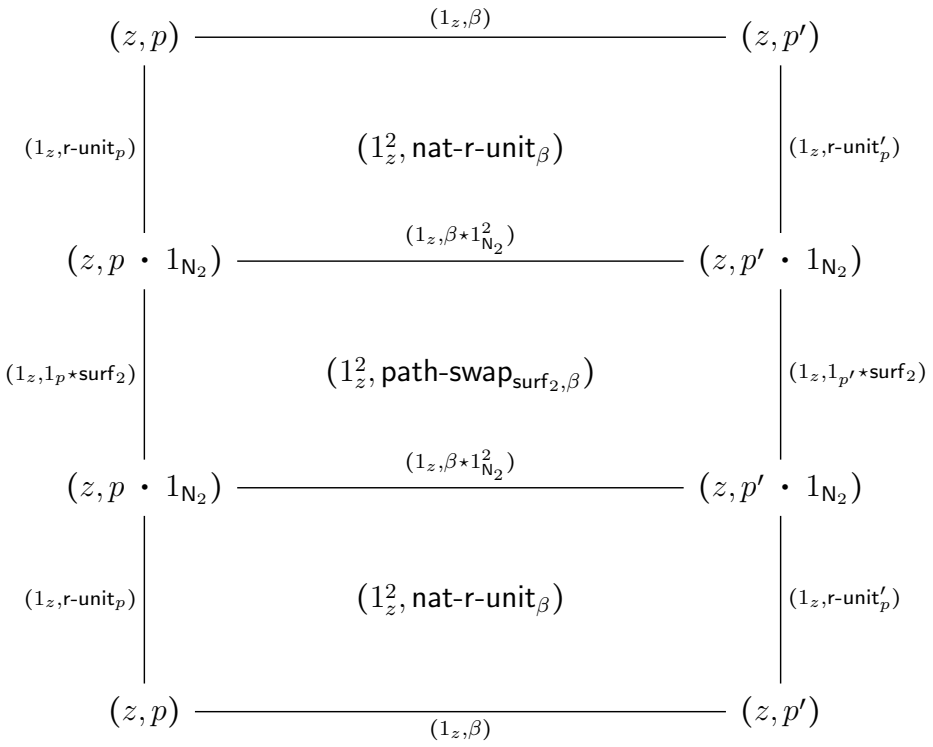
In virtue of this lemma, we will work directly with $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$ and $\lambda(z,p).(1_z, \mathsf{r\text{-}unit}_p^{-1} \cdot (1_p \star \mathsf{surf}_2) \cdot \mathsf{r\text{-}unit}_p)$. We will also use the shorthand $(\mathsf{fib})^2$ to refer to the latter homotopy. It is important to note that we should expect $(\mathsf{fib})^2$ to be non-trivial since surf_2 lends a non-trivial loop in the fiber.

Since $(\mathsf{fib})^2$ is a homotopy, it has a naturality condition induced by paths in the fiber of hpf. We want to characterize this naturality condition. We will use the characterization of the naturality condition of $\lambda(z,p).(1_z, 1_p \star \mathsf{surf}_2)$ and the homotopy in lemma 5.1.1 to derive a characterization for $(\mathsf{fib})^2$.

For a path $(1_z, \beta) : (z,p) = (z,p')$, lemma 3.2.5 tells us that the naturality condition of $\lambda(z,p).(1_z, 1_p \star \mathsf{surf}_2)$ is given by

$$\begin{array}{ccc}
(z, p \cdot 1_{\mathsf{N}_2}) & \xrightarrow{(1_z, \beta \star 1^2_{\mathsf{N}_2})} & (z, p' \cdot 1_{\mathsf{N}_2}) \\
{\scriptstyle (1_z, 1_p \star \mathsf{surf}_2)}\downarrow & (1^2_z, \mathsf{path\text{-}swap}_{\mathsf{surf}_2, \beta}) & \downarrow{\scriptstyle (1_z, 1_p \star \mathsf{surf}_2)} \\
(z, p \cdot 1_{\mathsf{N}_2}) & \xrightarrow[(1_z, \beta \star 1^2_{\mathsf{N}_2})]{} & (z, p' \cdot 1_{\mathsf{N}_2})
\end{array}$$

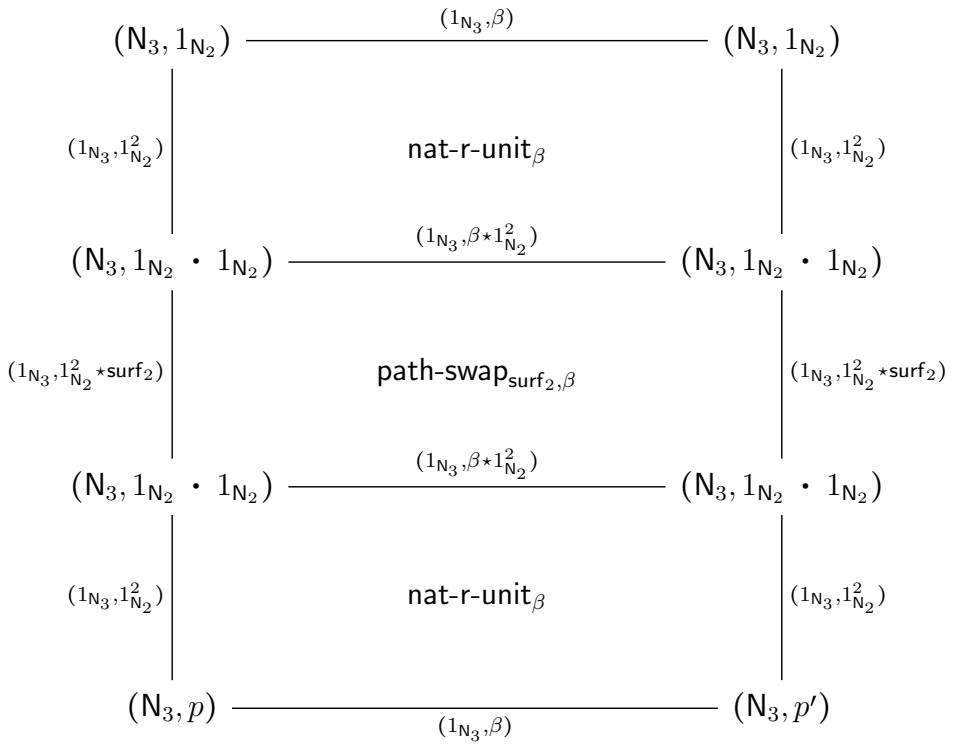
Now, the naturality squares of composites of homotopies can be computed as a pasted composite of each naturality condition. This lets us compute the naturality of $(\mathsf{fib})^2$ on $(1_z, \beta)$ with the following diagram.



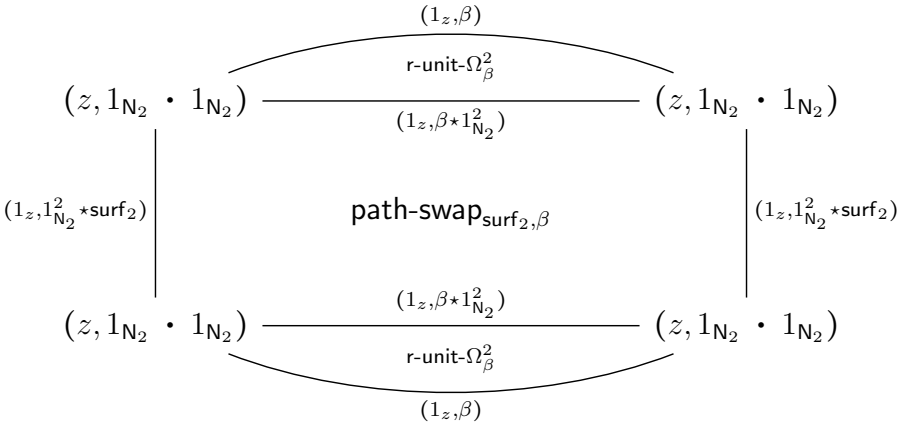
Lemma 5.1.2. *For $\beta : p = p'$, the above diagram computes* $\mathsf{nat\text{-}htpy}((\mathsf{fib})^2)(1_z,\beta)$.

Proof. By the above considerations. □

We can give an even better characterization of $(\mathsf{fib})^2$ when β is a 2-loop. We will essentially characterize $(\mathsf{fib})^2(\beta)$ as the Eckmann-Hilton path. When β is a 2-loop (i.e. $z \equiv \mathsf{N}_3$ and $p \equiv p' \equiv 1_{\mathsf{N}_2}$), our digram becomes



The obvious equivalence ("compressing" the extra 1_{N_2}'s) sends the previous diagram to



It is easy to see that this path, when concatenated on the left by $(\mathsf{l\text{-}unit}\text{-}\Omega^2_{\mathsf{surf}_2} \star 1_{(1_z,\beta)})^{-1}$ and on the right by $(1_{(1_z,\beta)} \star \mathsf{l\text{-}unit}\text{-}\Omega^2_{\mathsf{surf}_2})$, lends the path $\mathsf{EH}(\mathsf{surf}_2, \beta)$.

Lemma 5.1.3. *For a 2-loop $\beta : 1_{\mathsf{N}_2} = 1_{\mathsf{N}_2}$, we have*

$$(\mathsf{l\text{-}unit}\text{-}\Omega^2_{\mathsf{surf}_2} \star 1_{(1_{\mathsf{N}_3},\beta)})^{-1} \cdot (\mathsf{fib})^2(1_{\mathsf{N}_3}, \beta) \cdot (1_{(1_{\mathsf{N}_3},\beta)} \star \mathsf{l\text{-}unit}\text{-}\Omega^2_{\mathsf{surf}_2}) = \mathsf{EH}(\mathsf{surf}_2, \beta)$$

Proof. By the above remarks. □

As might be expected, this observation is at the heart of our upcoming characterization of the fiber. We now have all the data we need to make an educated selection of the type family $\mathcal{H}$.

5.2 The family $\mathcal{H} : \mathbb{S}^2 \to U$

In this section we will construct the family $\mathcal{H} : \mathbb{S}^2 \to U$. There are only two conditions we know $\mathcal{H}$ must satisfy: (i) $\mathcal{H}$ is familiar enough that proving it is fiberwise equivalent to fib is actually helpful and (ii) we can actually prove a fiberwise equivalence.

Satisfying (i) is easy. We only need $\mathcal{H}(\mathsf{N}_2) :\equiv \mathbb{S}^1$, since this plus a fiberwise equivalence implies $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2) \simeq \mathbb{S}^1$. Satisfying (ii) is of course the tricky part, but the results of the previous section are here to help. In particular, lemma 5.1.1 characterizing the descent data of fib will be our guide. We know that, in order for $\mathcal{H}$ to actually be equivalent to fib, the descent data of the two families must "match". We have seen that

$$(\mathsf{fib})^2 :\equiv \lambda(z,p).(1_z, \mathsf{r\text{-}unit}_p^{-1} \cdot (1_p \star \mathsf{surf}_2) \cdot \mathsf{r\text{-}unit}_p)$$

is the 2-dimensional descent data of fib. We expect this to be non-trivial, since surf_2 is not in the image of hpf. From our preliminary look at $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$, we saw that it should have base point $(\mathsf{N}_3, 1_{\mathsf{N}_2})$. If we apply $(\mathsf{fib})^2$ to this point, we get

$$\begin{aligned}(1_{\mathsf{N}_3}, \mathsf{r\text{-}unit}_{1_{\mathsf{N}_2}}^{-1} \cdot (1^2_{\mathsf{N}_2} \star \mathsf{surf}_2) \cdot \mathsf{r\text{-}unit}_{1_{\mathsf{N}_2}}) &\equiv (1_{\mathsf{N}_3}, 1^2_{\mathsf{N}_2} \cdot (1^2_{\mathsf{N}_2} \star \mathsf{surf}_2) \cdot 1^2_{\mathsf{N}_2}) \\ &= (1_{\mathsf{N}_3}, (1^2_{\mathsf{N}_2} \star \mathsf{surf}_2)) \\ &= (1_{\mathsf{N}_3}, \mathsf{surf}_2)\end{aligned}$$

But, we expect that $(1_{\mathsf{N}_3}, \mathsf{surf}_2)$ is the generating loop of $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$. Thus, we should choose for $(\mathcal{H})^2$ a non-trivial homotopy that is induced by a generator (or sends the base point of $\mathbb{S}^1$ to a generator).

There are two such homotopies $\mathsf{id}_{\mathbb{S}^1} \sim \mathsf{id}_{\mathbb{S}^1}$, namely L and L^{-1}. The homotopy L is defined by sending b_1 to loop and choosing the naturality condition $1_{\mathsf{loop} \,\cdot\, \mathsf{loop}}$. The homotopy L^{-1} is the inverse homotopy of L. It can also be defined directly by sending b_1 to loop^{-1} and choosing the naturality condition $\mathsf{l\text{-}inv}_{\mathsf{loop}} \,\cdot\, \mathsf{r\text{-}inv}^{-1}_{\mathsf{loop}}$. Which choice of homotopy is best depends in part on how the 3-loop eh is defined. Since we have defined eh in a way that uses surf_2^{-1}, we will want to choose L^{-1}.

We are now ready to define $\mathcal{H}$.

Definition 5.2.1. *We define a family* $\mathcal{H} : \mathbb{S}^1 \to U$.

Proof. We choose $\mathbb{S}^1$ over N_2 and L^{-1} over surf_2. □

This completes the definition of $\mathcal{H}$. As usual, we will work just with $(\mathcal{H})^2 \equiv \mathsf{L}^{-1}$ instead of $\mathsf{tr}^{(H)^2}(\mathsf{surf}_2)$. Before moving on, we will prove a few lemmas about $\mathcal{H}$ that will be useful later on.

Chapter 6

Eckmann-Hilton and The Hopf Fibration

This chapter contains the actual result of the thesis. We will prove that the Eckmann-Hilton 3-loop eh generates $\Omega^3(\mathbb{S}^2)$. We will prove this as a corollary of constructing a fiberwise equivalence:

$$\prod_{x:\mathbb{S}^2} \mathcal{H}(x) \simeq \mathsf{fib}(x)$$

We will break up the construction of the fiberwise equivalence into constructing a forwards map

$$g : \prod_{x:\mathbb{S}^2} \mathcal{H}(x) \to \mathsf{fib}(x)$$

constructing a backwards map

$$f : \prod_{x:\mathbb{S}^2} \mathsf{fib}(x) \to \mathcal{H}(x)$$

and fiberwise homotopies

$$\prod_{x:\mathbb{S}^2} g(x) \circ f(x) \sim \mathsf{id}$$

$$\prod_{x:\mathbb{S}^2} f(x) \circ g(x) \sim \mathsf{id}$$

Then, all claims other claims will follow from considering the fiber sequence of hpf and applying these results.

6.1 The forwards map, $g : \prod_{x:\mathbb{S}^2} \mathcal{H}(x) \to \mathsf{fib}(x)$

We begin with the construction of g. This is the more intresting of the two maps, since its construction hinges upon our choice of $(\mathcal{H})^2$ and the fact that $\mathsf{hpf}(\mathsf{surf}_3) = \mathsf{eh}$. By the universal property of $\mathbb{S}^2$, in order to construct g, it suffices to specify $g(\mathsf{N}_2) : \mathbb{S}^1 \to \mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$ and then specify a dependent 2-loop $g(\mathsf{surf}_2)$ at $g(\mathsf{N}_2)$. This construction (especially of the 2-loop) relies on characterization of dependent paths in a family of the form $A(x) \to B(x)$, as well as a characterzation of homotopies between functions with domain $\mathbb{S}^1$. Each characterization is straight forwards and left to the reader.

First we define $g(\mathsf{N}_2)$. A map out of $\mathbb{S}^1$ is equivalent to a loop in the codomain. Since we want this map to be an equivalence, we need to pick a generating loop of $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$. As we saw in 5.1, there are two candidates for a generating loop $(1_{\mathsf{N}_3}, \mathsf{surf}_2)$ and $(1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1})$. It doesn't really matter which we choose, but it will make calculations easier if we choose the inverse. Thus

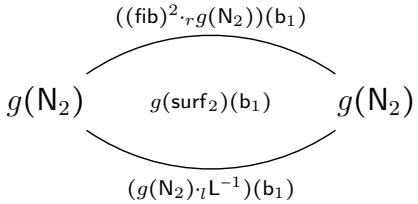
Definition 6.1.1. *We have a map* $g(\mathsf{N}_2) : \mathbb{S}^1 \to \mathsf{fib}_{\mathsf{hpf}}(\mathsf{N}_2)$

Proof. We use $\mathbb{S}^1$-induction and set $g(\mathsf{N}_2)(\mathsf{b}_1) :\equiv (\mathsf{N}_3, 1_{\mathsf{N}_2})$ and $g(\mathsf{N}_2)(\mathsf{loop}) := (1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1})$. □

Now we need a dependent 2-loop $g(\mathsf{surf}_2)$ at $g(\mathsf{N}_2)$. But a dependent 2-loop in the family $\lambda(x).\mathcal{H}(x) \to \mathsf{fib}(x)$ is equivalent to a homotopy

$$g(\mathsf{surf}_2) : (\mathsf{fib})^2 \cdot_r g(\mathsf{N}_2) \sim g(\mathsf{N}_2) \cdot_l \mathsf{L}^{-1}$$

Thus we have two homotopies $(\mathsf{fib})^2 \cdot_r g(\mathsf{N}_2)$ and $g(\mathsf{N}_2) \cdot_l \mathsf{L}^{-1}$ of type $g(\mathsf{N}_2) \sim g(\mathsf{N}_2)$ and we need to show that these homotopies are themselves homotopic to each other. Since the function $g(\mathsf{N}_2)$ has domain $\mathbb{S}^1$, we can use the characterization of homotopies and higher homotopies out of the circle. In particular, a homotopy out of $\mathbb{S}^1$ is uniquely determined by its image on the base point and the naturality condition induced by the loop. Thus, defining the homotopy $g(\mathsf{surf}_2)$ is equivalent to providing a path



and a commutative square

$$\begin{array}{ccc}
((\mathsf{fib})^2 \cdot_r g(\mathsf{N}_2))(\mathsf{b}_1) \cdot g(\mathsf{N}_2)(\mathsf{loop}) & \xrightarrow{((\mathsf{fib})^2 \cdot_r g(\mathsf{N}_2))(\mathsf{loop})} & g(\mathsf{N}_2)(\mathsf{loop}) \cdot ((\mathsf{fib})^2 \cdot_r g(\mathsf{N}_2))(\mathsf{b}_1) \\
\Big\downarrow{\scriptstyle g(\mathsf{surf}_2)(\mathsf{b}_1) \star 1_{g(\mathsf{N}_2)(\mathsf{loop})}} & & \Big\downarrow{\scriptstyle 1_{g(\mathsf{N}_2)(\mathsf{loop})} \star g(\mathsf{surf}_2)(\mathsf{b}_1)} \\
(g(\mathsf{N}_2) \cdot_l \mathsf{L}^{-1})(\mathsf{b}_1) \cdot g(\mathsf{N}_2)(\mathsf{loop}) & \xrightarrow[(g(\mathsf{N}_2) \cdot_l \mathsf{L}^{-1})(\mathsf{loop})]{} & g(\mathsf{N}_2)(\mathsf{loop}) \cdot (g(\mathsf{N}_2) \cdot_l \mathsf{L}^{-1})(\mathsf{b}_1)
\end{array}$$

Here the horizontal paths are the naturality conditions of the respective homotopies. We can compute

$$\begin{aligned}
(\mathsf{fib})^2 \cdot_r g(\mathsf{N}_2)(\mathsf{b}_1) &\equiv (\mathsf{fib})^2(\mathsf{N}_3, 1_{\mathsf{N}_2}) \\
&\equiv (1_{\mathsf{N}_3}, \mathsf{r\text{-}unit}^{-1}_{1_{\mathsf{N}_2}} \cdot (1^2_{\mathsf{N}_2} \star \mathsf{surf}_2) \cdot \mathsf{r\text{-}unit}_{1_{\mathsf{N}_2}}) \\
&\equiv (1_{\mathsf{N}_3}, 1^2_{\mathsf{N}_2} \cdot (1^2_{\mathsf{N}_2} \star \mathsf{surf}_2) \cdot 1^2_{\mathsf{N}_2}) \\
&= (1_{\mathsf{N}_3}, (1^2_{\mathsf{N}_2} \star \mathsf{surf}_2)) \text{ , by using unit laws}
\end{aligned}$$

We also have

$$\begin{aligned}
(g(\mathsf{N}_2) \cdot_l \mathsf{L}^{-1})(\mathsf{b}_1) &\equiv g(\mathsf{N}_2)(\mathsf{loop}^{-1}) \\
&= g(\mathsf{N}_2)(\mathsf{loop})^{-1} \text{ , by } \mathsf{ap\text{-}inv} \\
&\equiv (1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1})^{-1} \\
&= (1_{\mathsf{N}_3}, \mathsf{surf}_2) \text{ , by } \mathsf{inv\text{-}inv}
\end{aligned}$$

Thus, we can simplify and rewrite the boundary of the desired commutative square as

$$\begin{array}{ccc}
(1_{\mathsf{N}_3}, (1^2_{\mathsf{N}_2} \star \mathsf{surf}_2) \cdot \mathsf{surf}_2^{-1}) & \xrightarrow{(\mathsf{fib})^2(1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1})} & (1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1} \cdot (1^2_{\mathsf{N}_2} \star \mathsf{surf}_2)) \\
\Big\downarrow{\scriptstyle g(\mathsf{surf}_2)(\mathsf{b}_1) \star 1_{g(\mathsf{N}_2)(\mathsf{loop})}} & & \Big\downarrow{\scriptstyle 1_{g(\mathsf{N}_2)(\mathsf{loop})} \star g(\mathsf{surf}_2)(\mathsf{b}_1)} \\
(1_{\mathsf{N}_3}, \mathsf{surf}_2 \cdot \mathsf{surf}_2^{-1}) & \xrightarrow[g(\mathsf{N}_2)(\mathsf{l\text{-}inv}) \cdot g(\mathsf{N}_2)(\mathsf{r\text{-}inv})^{-1}]{} & (1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1} \cdot \mathsf{surf}_2)
\end{array}$$

Here we have suppresed many coherence paths to increase the readability of the diagram. Later it will become important that these coherence paths cancel.

Constructing the path $g(\mathsf{surf}_2)(\mathsf{b}_1)$ is easy. Being sloppy and supressing some coherence

paths, we can use the computations above and set $g(\mathsf{surf}_2)(\mathsf{b}_1) :\equiv (1^2_{\mathsf{N}_3}, \mathsf{l\text{-}unit\text{-}}\Omega^2)$. Now we need to provide a commutative square. First we rewrite it boundary as

$$\begin{array}{ccccc}
(1_{\mathsf{N}_3}, (1^2_{\mathsf{N}_2} \star \mathsf{surf}_2) \cdot \mathsf{surf}_2^{-1}) & & \xrightarrow{(\mathsf{fib})^2(1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1})} & & (1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1} \cdot (1^2_{\mathsf{N}_2} \star \mathsf{surf}_2)) \\
\Big\downarrow{\scriptstyle (1^2_{\mathsf{N}_3}, \mathsf{l\text{-}unit\text{-}}\Omega^2 \star 1_{\mathsf{surf}_2^{-1}})} & & & & \Big\downarrow{\scriptstyle (1_{\mathsf{N}_3}, 1_{\mathsf{surf}_2^{-1}} \star \mathsf{l\text{-}unit\text{-}}\Omega^2)} \\
(1_{\mathsf{N}_3}, \mathsf{surf}_2 \cdot \mathsf{surf}_2^{-1}) & \xrightarrow[g(\mathsf{N}_2)(\mathsf{l\text{-}inv})]{} & (1_{\mathsf{N}_3}, 1^2_{\mathsf{N}_2}) & \xrightarrow[g(\mathsf{N}_2)(\mathsf{r\text{-}inv})^{-1}]{} & (1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1} \cdot \mathsf{surf}_2)
\end{array}$$

Note that showing the square commutes is equivalent to showing that the loop starting and ending at $(1_{\mathsf{N}_3}, 1^2_{\mathsf{N}_2})$ is trivial. We will show this loop is trivial by showing that it is equal to $(1^2_{\mathsf{N}_3}, \mathsf{eh})$. Then we will see that this latter path is trivial since $\mathsf{hpf}(\mathsf{surf}_3) = \mathsf{eh}$.

Lemma 5.1.3 tells us that tracing the diagram along the path left-top-right is equal to $\mathsf{EH}(\mathsf{surf}_2, \mathsf{surf}_2^{-1})$. Now we just need to characterize the bottom two paths. As we should expect, these will indeed simplify to $(1^2_{\mathsf{N}_3}, \mathsf{r\text{-}inv}_{\mathsf{surf}_2})$ and $(1_{\mathsf{N}_3}, \mathsf{l\text{-}inv}^{-1}_{\mathsf{surf}_2})$, though it is a bit tedious (and conceptually uninteresting) to give all the details. So I only give a sketch as to how to show this. Since functions prelerve inverse laws, we can obtain

$$g(\mathsf{N}_2)(\mathsf{l\text{-}inv}_{\mathsf{loop}}) = \mathsf{ap\text{-}concat}(g(\mathsf{N}_2))_{\mathsf{loop}^{-1}, \mathsf{loop}} \cdot (\mathsf{ap\text{-}inv}(g(\mathsf{N}_2))_{\mathsf{loop}} \star 1) \cdot \mathsf{l\text{-}inv}_{g(\mathsf{N}_2)(\mathsf{loop})}$$

We know that $g(\mathsf{N}_2)(\mathsf{loop}) = (1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1})$. Now, $\mathsf{l\text{-}inv}_{g(\mathsf{N}_2)(\mathsf{loop})} = (\mathsf{l\text{-}inv}_{1_{\mathsf{N}_3}}, \mathsf{l\text{-}inv}_{\mathsf{surf}_2^{-1}}) \equiv (1^2_{\mathsf{N}_3}, \mathsf{l\text{-}inv}_{\mathsf{surf}_2^{-1}})$. From here we use a coherence between on an inverse path to obtain $\mathsf{l\text{-}inv}_{\mathsf{surf}_2^{-1}} = (\mathsf{inv\text{-}inv} \star 1) \cdot \mathsf{r\text{-}inv}_{\mathsf{surf}_2}$.

We can use this to replace $g(\mathsf{N}_2)(\mathsf{l\text{-}inv})$ in the above diagram. Of course we will end up with a hefty amount of extra coherence paths. But we have been implicitly applying many coherence paths when simplifying the boundary of the square. As it turns out (and I will leave to the

reader to verify this if they so choose), if we write out all the implicitly applied coherence paths, they we cancel with almost everything, leaving only $\mathsf{r\text{-}inv}_{\mathsf{surf}_2}$. Similar remarks apply to the other side of the bottom.

Putting this all together, we see that the loop tracing the boundary of the diagram in question is equal to

$$(1^2_{\mathsf{N}_3}, \mathsf{r\text{-}inv}^{-1} \cdot \mathsf{EH}(\mathsf{surf}_2, \mathsf{surf}_2^{-1}) \cdot \mathsf{l\text{-}inv}) \equiv (1^2_{\mathsf{N}_3}, \mathsf{eh})$$

Now we can see why the loop is trivial. We need a path $(1^2_{\mathsf{N}_3}, \mathsf{eh}) = (1^2_{\mathsf{N}_3}, 1^3_{\mathsf{N}_2})$. By our fiber of ap characterization of paths in the fiber, this type is equivalent to $\mathsf{fib}_{\mathsf{hpf}}(\mathsf{eh})$. We inhabit this latter type with $(\mathsf{surf}_3, 1_{\mathsf{eh}})$, since we defined $\mathsf{hpf}(\mathsf{surf}_3) = \mathsf{eh}$. This defines $g(\mathsf{surf}_2)(\mathsf{loop})$. We can now collect all the above to define g:

Definition 6.1.2. *We have a function* $g : \prod_{x:\mathbb{S}^2} \mathcal{H}(x) \to \mathsf{fib}(x)$

Proof. We defined g by $\mathbb{S}^2$-induction. We defined $g(\mathsf{N}_2)$ in definition 6.1.1 by choosing the loop $(1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1})$ in the fiber. Then we constructed the homotopy $g(\mathsf{surf}_2)$. We set $g(\mathsf{surf}_2)(\mathsf{b}_1) :\equiv (1^2_{\mathsf{N}_3}, \mathsf{l\text{-}unit\text{-}}\Omega^2)$. For the naturality square $g(\mathsf{surf}_2)(\mathsf{loop})$, we used the fact that $\mathsf{hpf}(\mathsf{surf}_3) = \mathsf{eh}$. □

This concludes the construction of the forwards map.

6.2 The backwards map, $f : \prod_{x:\mathbb{S}^2} \mathsf{fib}(x) \to \mathcal{H}(x)$

Now we define the backwards map $f : \prod_{x:\mathbb{S}^2} \mathsf{fib}(x) \to \mathcal{H}(x)$. This is a much more trivial affair. To define f, we first define a helpful function.

Lemma 6.2.1. *We have a function* $\mathsf{diag} : \prod_{z:\mathbb{S}^3} \mathcal{H} \circ \mathsf{hpf}(z)$.

Proof. We define this by $\mathbb{S}^3$-induction. We set $\mathsf{diag}(\mathsf{N}_3) :\equiv \mathsf{b}_1$, since $\mathcal{H} \circ \mathsf{hpf}(\mathsf{N}_3) \equiv \mathbb{S}^1$. Now we need to specify a dependent 3-loop at b_1. But a dependent 3-loop is ultimately a 3-path. Since $\mathbb{S}^1$ is a 1-type, every type of 3-paths in $\mathbb{S}^1$ is contractible. □

In some ways, this lemma partially states that hpf behaves like the projection from $\sum_{x:\mathbb{S}^2} \mathcal{H}(x)$, the the projection of a fibration followed by the original family necessarily has a section. Now we can define f.

Definition 6.2.1. *We have a map* $f : \prod_{x:\mathbb{S}^2} \mathsf{fib}(x) \to \mathcal{H}(x)$.

Proof. Let $x : \mathbb{S}^2$. We need a function $f(x) : \mathsf{fib}_{\mathsf{hpf}}(x) \to \mathcal{H}(x)$. But $\mathsf{fib}_{\mathsf{hpf}}(x) \equiv \sum_{z:\mathbb{S}^3} \mathsf{hpf}(z) = x$. Thus, we can define $f(x)$ by $\sum$-induction. We can set $f(x, z, p) :\equiv \mathsf{tr}^{\mathcal{H}}(p)(\mathsf{diag}(z))$. This definition is well typed since $\mathsf{diag}(z) : \mathcal{H} \circ \mathsf{hpf}(z)$ and $p : \mathsf{hpf}(z) = x$. □

This completes the definition of f. But, before we move on, we need to characterize the action on paths of f. For a path (or 2-path) in $\mathbb{S}^2$, the action of f on such a path is uninteresting. We are interested in characterizing the action on paths of $f(x)$, for some fixed $x : \mathbb{S}^2$. As before, we are only interested in the paths in the fiber of the form $(1_z, q)$.

Lemma 6.2.2. *For a path* $(1_z, q) : (z, p) = (z, p')$ *in the fiber, we have that* $f(x)(1_z, q) = \mathsf{tr}^{(H)^2}(q)(\mathsf{diag}(z))$.

Proof. By path induction on q, both sides reduce to $1_{\mathsf{tr}^H(p)(\mathsf{diag}(z))}$. □

6.3 The homotopies

Now we are ready to construct the homotopies. We will first construct the fiberwise homotopy

$$K : \prod_{x:\mathbb{S}^2} f(x) \circ g(x) \sim \mathsf{id}_{\mathcal{H}(x)}$$

We do this by $\mathbb{S}^2$ induction. Thus, we need to specify a homotopy $K(\mathsf{N}_2) : f(\mathsf{N}_2) \circ g(\mathsf{N}_2) \sim \mathsf{id}_{\mathbb{S}^1}$ and then specify a dependent 2-loop at this homotopy. We can construct the homotopy by $\mathbb{S}^1$ induction. As before, this is equivalent to giving a path

$$K(\mathsf{N}_2)(\mathsf{b}_1) : (f(\mathsf{N}_2) \circ g(\mathsf{N}_2))(\mathsf{b}_1) = \mathsf{b}_1$$

and then a naturality square

$$\begin{array}{ccc} (f(\mathsf{N}_2) \circ g(\mathsf{N}_2))(\mathsf{b}_1) & \xrightarrow{(f(\mathsf{N}_2) \circ g(\mathsf{N}_2))(\mathsf{loop})} & (f(\mathsf{N}_2) \circ g(\mathsf{N}_2))(\mathsf{b}_1) \\ {\scriptstyle K(\mathsf{N}_2)(\mathsf{b}_1)}\big\downarrow & K(\mathsf{N}_2)(\mathsf{b}_1) & \big\downarrow{\scriptstyle K(\mathsf{N}_2)(\mathsf{b}_1)} \\ \mathsf{b}_1 & \underset{\mathsf{loop}}{\longrightarrow} & \mathsf{b}_1 \end{array}$$

We can compute

$$\begin{aligned} (f(\mathsf{N}_2) \circ g(\mathsf{N}_2))(\mathsf{b}_1) &\equiv f(\mathsf{N}_2)(\mathsf{N}_3, 1_{\mathsf{N}_2}) \\ &\equiv \mathsf{tr}^{\mathcal{H}}(1_{\mathsf{N}_2})(\mathsf{diag}(\mathsf{N}_3)) \\ &\equiv \mathsf{diag}(\mathsf{N}_3) \\ &\equiv \mathsf{b}_1 \end{aligned}$$

Thus we can simply set $K(\mathsf{N}_2)(\mathsf{b}_1) :\equiv 1_{\mathsf{b}_1}$. The boundary of our diagram thus simplifies to showing that $(f(\mathsf{N}_2) \circ g(\mathsf{N}_2))(\mathsf{loop}) = \mathsf{loop}$. We can compute

$$
\begin{aligned}
(f(\mathsf{N}_2) \circ g(\mathsf{N}_2))(\mathsf{loop}) &= f(\mathsf{N}_2)(1_{\mathsf{N}_3}, \mathsf{surf}_2^{-1}) \\
&\equiv \mathsf{tr}^{(\mathcal{H})^2}(\mathsf{surf}_2^{-1})(\mathsf{diag}(\mathsf{N}_3)) \text{ , by lemma 6.2.2} \\
&= \mathsf{tr}^{(\mathcal{H})^2}(\mathsf{surf}_2)^{-1}(\mathsf{b}_1) \\
&= (\mathsf{L}^{-1})^{-1}(\mathsf{b}_1) \\
&= \mathsf{L}(\mathsf{b}_1) \\
&= \mathsf{loop}
\end{aligned}
$$

This completes the definition of $K(\mathsf{N}_2)$. Now we need to specify a dependent 2-loop at $K(\mathsf{N}_2)$. A dependent 2-loop will be equivalent to a homotopy between two homotopies. But each of the two homotopies has domain (and so its component paths) in $\mathbb{S}^1$. Since $\mathbb{S}^1$ is a 1-type, we are gaurenteed to have this homotopy of homotopies. This completes the definition of K.

Now we need to construct a fiberwise homotopy $M : \prod_{x:\mathbb{S}^2} g(x) \circ f(x) \sim \mathsf{id}_{\mathsf{fib}(x)}$. We can rewrite this as

$$\prod_{x:\mathbb{S}^2} \prod_{w:\mathsf{fib}(x)} (g(x) \circ f(x))(w) = w$$

But, by currying $\mathsf{fib}(x) \equiv \sum_{z:\mathbb{S}^3} \mathsf{hpf}(z) = x$, this is equivalent to

$$\prod_{x:\mathbb{S}^2} \prod_{z:\mathbb{S}^3} \prod_{p:\mathsf{hpf}(z)=x} (g(x) \circ f(x))(z,p) = (z,p)$$

But now we can use path induction on p and the free end point x. Thus, it suffices to assume $x \equiv \mathsf{hpf}(z)$ and $p \equiv 1_{\mathsf{hpf}(z)}$ and inhabit the type

$$\prod_{z:\mathbb{S}^3}(g(\mathsf{hpf}(z)) \circ f(\mathsf{hpf}(z)))(z, 1_{\mathsf{hpf}(z)}) = (z, 1_{\mathsf{hpf}(z)})$$

But this definitionally simplify this to

$$\prod_{z:\mathbb{S}^3} g(\mathsf{hpf}(z))(\mathsf{diag}(z)) = (z, 1_{\mathsf{hpf}(z)})$$

We view can this of this as the type of homotopies between two functions. The functions in question are

$$\lambda(z).g(\mathsf{hpf}(z))(\mathsf{diag}(z)) : \prod_{z:\mathbb{S}^3} \mathsf{fib}_{\mathsf{hpf}}(\mathsf{hpf}(z))$$

and

$$\lambda(z).(z, 1_{\mathsf{hpf}(z)}) : \prod_{z:\mathbb{S}^3} \mathsf{fib}_{\mathsf{hpf}}(\mathsf{hpf}(z))$$

We will inhabit this type by $\mathbb{S}^3$-induction. On N_3, both sides definitionally simplify to $(\mathsf{N}_3, 1_{\mathsf{N}_2})$. Thus we can choose $1_{(\mathsf{N}_3, 1_{\mathsf{N}_2})}$ at N_3. Now we just need to show that the image of surf_3 under the two of these functions is equal. It is rather straight forwards to compute that the latter functions action on surf_3 is equivalent to $(\mathsf{surf}_3, 1_{\mathsf{eh}})$.

It is slightly trickier to compute the action on paths of the first function. Since the computation is long and technical, and has to do mostly with some general facts about functions in HoTT, I will only include a conceptual outline of the computation. First recall that $\mathsf{diag}(\mathsf{surf}_3)$ is a trival path. This will allow us to compute that $(\lambda(z).g(\mathsf{hpf}(z))(\mathsf{diag}(z)))(\mathsf{surf}_3) = g(\mathsf{hpf}(\mathsf{surf}_3))(\mathsf{b}_1)$. Next recall that $\mathsf{surf}_3 \equiv \mathsf{pr}_1(g(\mathsf{surf}_2)(l))$. Thus, we are essentially trying to prove that

$$g\big(\mathsf{hpf}(\mathsf{pr}_1[g(\mathsf{surf}_2)(l)])\big)(\mathsf{b}_1) = g(\mathsf{surf}_2)(l)$$

This follows from a general fact about maps into a fiber. Consider a lower dimensional analogue of this situation. Let $x : \mathbb{S}^2$ and consider a path u in $\mathcal{H}(x)$. Then $g(x)(u) : \mathsf{fib}_{\mathsf{hpf}}(x)$. So $\mathsf{hpf}(\mathsf{pr}_1(g(x)(u))) = x$. Generalizing this idea to higher dimensions lends that

$$\mathsf{pr}_1\big[g\big(\mathsf{hpf}(\mathsf{pr}_1[g(\mathsf{surf}_2)(l)])\big)(\mathsf{b}_1)\big] = \mathsf{pr}_1(g(\mathsf{surf}_2)(l))$$

Computing that the second components match is similar. Giving a fully rigourous calculation of this in the type theory ends up being quite technical. The reason is that, in order to apply path induction (which is essentially the only way to prove the claim for paths), we have to generalize the situation to an arbitary path. This makes the formulae involved far more complicated than when dealing with loops, as our specific situation does. Thus we ommit the full computation.

Theorem 6.3.1. *We have that* $\prod_{x:\mathbb{S}^2} \mathcal{H}(x) \simeq \mathsf{fib}(x)$.

Proof. This follows from piecing together the maps and homotopies we have defined. □

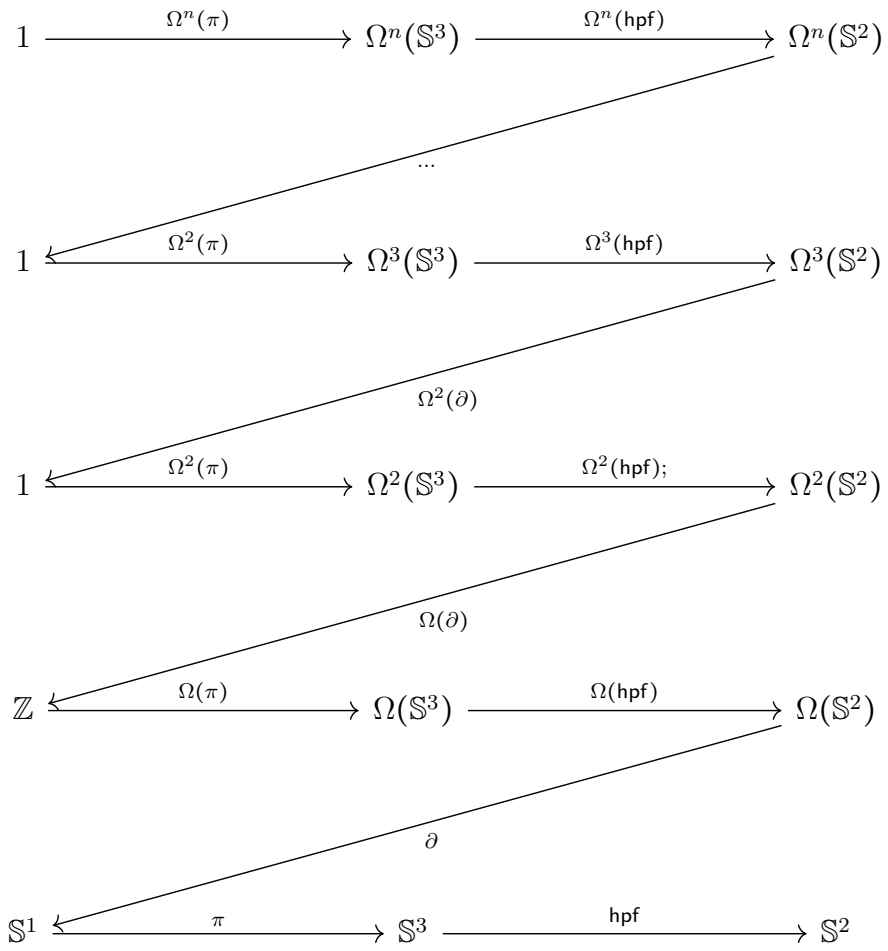
6.4 The corollaries

Theorem 6.3.1 implies all intresting results related to homotopy groups. Here we collect these results.

Theorem 6.4.1. *There is a map* $\mathbb{S}^3 \to \mathbb{S}^2$ *that sends the generating loop* surf_3 *of* $\mathbb{S}^3$ *to* eh *and whose fiber is* $\mathbb{S}^1$.

Proof. Take hpf and apply the equivalence of theorem 6.3.1 to the base point of surf_2. □

Now consider the fiber sequence of hpf:



Theorem 6.4.2. *We have an equvalence* $\Omega^3(\mathsf{hpf}) : \Omega^3(\mathbb{S}^3) \simeq \Omega^3(\mathbb{S}^2)$.

Proof. This is a general fact about fiber sequences. Since this is a fiber sequence, $\Omega^3(\mathbb{S}^3)$ is equivalent to the fiber of $\Omega^2(\partial)$. Since $\Omega^2(\partial)$ has codomain 1, we its fiber is equivalent to its domain. □

Corollary 6.4.2.1. *The 3-loop* eh *generates* $\Omega^3(\mathbb{S}^2)$.

Proof. We already know that surf_3 generates $\Omega^3(\mathbb{S}^3)$. By the above theorem, the map $\Omega^3(\mathsf{hpf})$, which sends surf_3 to eh, is an equivalence and equivalences preserve generators. □

Corollary 6.4.2.2. *We have that* $\pi_3(\mathbb{S}^2) \cong \mathbb{Z}$ *and is generated by the equivalence class of* eh.

Proof. This follows from the above results after applying truncations (somewhat analogous to quotienting by equivalence relations) and using the fact that $\pi_3(\mathbb{S}^3) \cong \mathbb{Z}$. □

It also turns out to be quite easy to show that π is homotopic to the constant map by using $\mathbb{S}^1$-induction. Since this is a fiber sequence, we have that $\Omega(\mathbb{S}^2) \simeq \mathsf{fib}_\pi(\mathsf{N}_2)$. But $\mathsf{fib}_\pi(\mathsf{N}_3) \simeq \mathsf{fib}_{\mathsf{const}}(\mathsf{N}_3) \equiv (\sum_{x:\mathbb{S}^1} \mathsf{N}_3 = \mathsf{N}_3) \equiv \mathbb{S}^1 \times \Omega(\mathbb{S}^3)$.

Theorem 6.4.3. *We have an equivalence* $\Omega(\mathbb{S}^2) \simeq \mathbb{S}^1 \times \Omega(\mathbb{S}^3)$.

Bibliography

[1] The Univalent Foundations Program. Homotopy Type Theory: Univalent Foundations of Mathematics. `Https://Homotopytypetheory.org/Book`, 2013.