On the homotopy groups of spheres in homotopy type theory

Guillaume Brunerie

UNIVERSITÉ DE NICE SOPHIA ANTIPOLIS – UFR Sciences

École Doctorale Sciences Fondamentales et Appliquées

THÈSE

pour obtenir le titre de
Docteur en Sciences
Spécialité MATHÉMATIQUES

présentée et soutenue par
Guillaume BRUNERIE

Sur les groupes d'homotopie des sphères en théorie des types homotopiques

On the homotopy groups of spheres in homotopy type theory

Thèse dirigée par **Carlos SIMPSON**
soutenue le 15 juin 2016

Membres du jury :

M. Denis-Charles CISINSKI	Professeur des universités	*Examinateur*
M. Thierry COQUAND	Professor	*Rapporteur*
M. André HIRSCHOWITZ	Professeur émérite	*Examinateur*
M. André JOYAL	Professeur émérite	*Examinateur*
M. Paul-André MELLIÈS	Chargé de recherche CNRS	*Examinateur*
M. Michael SHULMAN	Assistant Professor	*Rapporteur*
M. Carlos SIMPSON	Directeur de recherche CNRS	*Directeur de thèse*

Laboratoire Jean-Alexandre Dieudonné, Université de Nice, Parc Valrose, 06108 NICE

Abstract

The goal of this thesis is to prove that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$ in homotopy type theory. In particular it is a constructive and purely homotopy-theoretic proof. We first recall the basic concepts of homotopy type theory, and we prove some well-known results about the homotopy groups of spheres: the computation of the homotopy groups of the circle, the triviality of those of the form $\pi_k(\mathbb{S}^n)$ with $k < n$, and the construction of the Hopf fibration. We then move to more advanced tools. In particular, we define the James construction which allows us to prove the Freudenthal suspension theorem and the fact that there exists a natural number n such that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$. Then we study the smash product of spheres, we construct the cohomology ring of a space, and we introduce the Hopf invariant, allowing us to narrow down the n to either 1 or 2. The Hopf invariant also allows us to prove that all the groups of the form $\pi_{4n-1}(\mathbb{S}^{2n})$ are infinite. Finally we construct the Gysin exact sequence, allowing us to compute the cohomology of $\mathbb{C}P^2$ and to prove that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$ and that more generally $\pi_{n+1}(\mathbb{S}^n) \simeq \mathbb{Z}/2\mathbb{Z}$ for every $n \geq 3$.

Keywords: homotopy type theory, homotopy theory, algebraic topology, cohomology, type theory, logic, constructive mathematics

Résumé

L'objectif de cette thèse est de démontrer que $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$ en théorie des types homotopiques. En particulier, c'est une démonstration constructive et purement homotopique. On commence par rappeler les concepts de base de la théorie des types homotopiques et on démontre quelques résultats bien connus sur les groupes d'homotopie des sphères : le calcul des groupes d'homotopie du cercle, le fait que ceux de la forme $\pi_k(\mathbb{S}^n)$ avec $k < n$ sont triviaux et la construction de la fibration de Hopf. On passe ensuite à des outils plus avancés. En particulier, on définit la construction de James, ce qui nous permet de démontrer le théorème de suspension de Freudenthal et le fait qu'il existe un entier naturel n tel que $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$. On étudie ensuite le produit smash des sphères, on construit l'anneau de cohomologie des espaces et on introduit l'invariant de Hopf, ce qui nous permet de montrer que n est égal soit à 1, soit à 2. L'invariant de Hopf nous permet également de montrer que tous les groupes de la forme $\pi_{4n-1}(\mathbb{S}^{2n})$ sont infinis. Finalement, on construit la suite exacte de Gysin, ce qui nous permet de calculer la cohomologie de $\mathbb{C}P^2$ et de démontrer que $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$, et que plus généralement on a $\pi_{n+1}(\mathbb{S}^n) \simeq \mathbb{Z}/2\mathbb{Z}$ pour tout $n \geq 3$.

Mots-clés : théorie des types homotopiques, théorie de l'homotopie, topologie algébrique, cohomologie, théorie des types, logique, mathématiques constructives

Acknowledgments

Five years ago, towards the end of my first year of master studies, I wasn't sure in which domain of mathematics or computer science to continue. I was interested in many subjects, in particular homotopy theory and type theory, so one day I decided to search for "homotopy type theory" on the Internet, thinking that if something with such a name exists it is probably something for me. Apparently I was right.

I would like to thank all the people who helped me and supported me, in particular

- my advisor, Carlos Simpson, for always trusting me, supporting me, and encouraging me,
- the "rapporteurs" of my thesis, Thierry Coquand and Mike Shulman, and also Ulrik Buchholtz for their careful reading and comments on my thesis,
- the university of Nice Sophia Antipolis and the LJAD for letting me work in such a great environment,
- Paul-André Melliès who arranged for me to go to my first conference on homotopy type theory,
- Thierry Coquand, Steve Awodey, and Vladimir Voevodsky for letting me take part in the special year on Univalent Foundations at the Institute for Advanced Study in Princeton,
- all the people I've worked with, in particular Dan Licata and Thierry Coquand, but also Carlo Angiuli, Favonia, Eric Finster, Bob Harper, Simon Huber, André Joyal, Peter Lumsdaine, Egbert Rijke and many others,
- all other PhD students in Nice for making my stay enjoyable, in particular my officemates Arthur, Laurence, and Byron,
- my family for their constant support, in particular my late grandfather Alain who started fueling my mathematical curiosity when I was very young, teaching me how to multiply a number by 10 or 100,
- the European forró community for the countless festivals and dances,
- and finally my girlfriend, Monika, for her presence, her support, and for reading in detail early versions of parts of this text.

Contents

Introduction

The aim of this PhD thesis is to prove the following theorem, whose statement and proof will be explained in due time.

Theorem 1. *We have a group isomorphism*

$$\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}.$$

This is actually a well-known theorem in classical homotopy theory, originally proved by Freudenthal in [Fre37] (see also [Hat02, corollary 4J.4]). The main difference is that in this thesis we work in *homotopy type theory* (also known as *univalent foundations*), which is a new framework for doing mathematics introduced by Vladimir Voevodsky in 2009 and which is particularly well-suited for homotopy theory. From the point of view of a homotopy theorist, the most striking difference between classical homotopy theory and homotopy type theory is that in homotopy type theory *all* constructions are invariant under homotopy equivalences. One of the advantages is that all the constructions and proofs done in this framework are completely independent of the definition of "spaces". In particular, nothing depends on point-set topology or on combinatorics of simplicial sets. Moreover, as we hope the reader will be convinced after reading this thesis, the constructions and proofs have often a more "homotopy-theoretic feel" and are closer to intuition.

However, this also poses a number of challenges as it is not *a priori* obvious which concepts can or cannot be defined in a purely homotopy-invariant way. For instance, even though singular cohomology is homotopy-invariant, the classical definition uses the set of singular cochains which is not homotopy-invariant. Therefore the classical definition cannot be reproduced verbatim in homotopy type theory. An even simpler example is the universal cover of the circle which is classically defined using the exponential function $\mathbb{R} \to \mathbb{S}^1$, but that function is actually homotopic to a constant function. Homotopy type theory gives us a number of tools to work in a completely homotopy-invariant way and in this thesis we show how to prove theorem 1 in homotopy type theory, starting essentially from scratch.

Another advantage of homotopy type theory over classical homotopy theory is that proofs written in homotopy type theory are much more amenable to being formally checked by a computer. While the present work hasn't been formalized yet, many intermediate results (in particular from the first two chapters) have already been formalized

by various people, see for instance the libraries [HoTTCoq] and [Unimath] for Coq, [HoTTAgda] for Agda and [HoTTLean] for Lean.

Content of the thesis The first two chapters of this thesis review basic homotopy type theory. An alternative reference is the book [UF13], but we tried here to be more concise and to keep in mind our end goal. Nevertheless there might be some overlap in the style of presentation between [UF13] and the introduction and the first two chapters of this thesis. Most of the content of the last four chapters is new in homotopy type theory even though the concepts are well-known in classical homotopy theory. The definition of weak ∞-groupoid presented in the first appendix is new as well.

In chapter 1 we introduce all the basic concepts of homotopy type theory, namely all type constructors and in particular the univalence axiom and higher inductive types. We also state the 3×3-lemma and the flattening lemma in sections 1.8 and 1.9, which are two results that we use in various places. Finally we talk about n-truncatedness and truncations. The notion of n-truncated type corresponds to the classical notion of homotopy n-types, i.e. spaces with no homotopical information above dimension n, and truncation is an operation turning any space into an n-truncated space in a universal way. All this is again standard in homotopy type theory.

In chapter 2 we define the homotopy groups of spheres. The group $\pi_k(\mathbb{S}^n)$ is defined as the 0-truncation (i.e. the set of connected components) of the space of k-dimensional loops in $\mathbb{S}^n$. Then we show how to prove that $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$, which is a result originally proven by Michael Shulman in 2011 and which appears in [UF13, section 8.1], cf also [Shu11] and [LS13]. The idea is that, in homotopy type theory, in order to define a fibration we do not give a map from the total space to the base space. Instead we give directly the fibers over every point of the base space. In the case of a fibration over the circle, it is enough to give the fiber over the basepoint of $\mathbb{S}^1$ and the action on the fiber of the loop going around $\mathbb{S}^1$. Here the fiber is the space of integers $\mathbb{Z}$ and the loop of the circle acts on it by the function which adds one. This gives a fibration over $\mathbb{S}^1$ and one can show that its total space is contractible, from which the isomorphism $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$ follows. We then define the notion of connectedness and prove various properties about connected spaces and maps which allow us to prove that $\pi_k(\mathbb{S}^n)$ is trivial for all $k < n$. This result already appears in [UF13, section 8.3] with a more complicated proof, also due to the author. Finally we define the Hopf fibration, which is a fibration over $\mathbb{S}^2$ with fiber $\mathbb{S}^1$ and total space $\mathbb{S}^3$. The idea of the definition of the Hopf fibration is as follows. In order to define a fibration over $\mathbb{S}^2$ it is enough to give the fiber N over the north pole, the fiber S over the south pole and, for every element x of $\mathbb{S}^1$, an equivalence between N and S which describe what happens when we move in the fibration over the meridian corresponding to x. In the case of the Hopf fibration, we take $N, S := \mathbb{S}^1$, and the equivalence between N and S corresponding to x is the operation of multiplication by x. The Hopf fibration was first defined by Peter Lumsdaine, in a slightly different way, but without a proof that its total space is equivalent to $\mathbb{S}^3$. The construction presented here was first written as [UF13, section 8.5].

In chapter 3 we define the James construction following an initial idea of André

Joyal. For every type A we define a family of spaces (J_nA) and we prove that their colimit is equivalent to the loop space of the suspension of A. This is done by defining another space JA and proving that JA is equivalent to both the colimit of (J_nA) and to the loop space of the suspension of A. The James construction gives a sequence of approximations of the loop space of the suspension of A which, in conjunction with the Blakers–Massey theorem, allows us to prove that there is a natural number n such that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$. This number n is defined using Whitehead products, more precisely it is the image of the Whitehead product $[\mathsf{id}_{\mathbb{S}^2}, \mathsf{id}_{\mathbb{S}^2}]$, which is an element of $\pi_3(\mathbb{S}^2)$, by the equivalence $\pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$ constructed using the Hopf fibration.

In chapter 4 we study the smash product and its symmetric monoidal structure. In particular we construct a family of equivalences $\mathbb{S}^n \wedge \mathbb{S}^m \simeq \mathbb{S}^{n+m}$ which is compatible, in some sense, with associativity and commutativity of the smash product. The construction of the symmetric monoidal structure will be essentially admitted, but we give some intuition on how to construct it.

In chapter 5 we first define, for every natural number n, the Eilenberg–MacLane space $K(\mathbb{Z}, n)$ as the n-truncation of the sphere $\mathbb{S}^n$ and the n-th cohomology group of a space X as the 0-truncation of the function space $X \to K(\mathbb{Z}, n)$. We then define the cup product as a map $K(\mathbb{Z}, n) \wedge K(\mathbb{Z}, m) \to K(\mathbb{Z}, n + m)$ by taking the smash product of the two maps $\mathbb{S}^n \to K(\mathbb{Z}, n)$ and $\mathbb{S}^m \to K(\mathbb{Z}, m)$, composing with the equivalence $\mathbb{S}^n \wedge \mathbb{S}^m \simeq \mathbb{S}^{n+m}$, and using some properties of connectivity of maps to show that we can essentially invert it. The properties of the smash product from chapter 4 are then used to prove that the cup product is associative and graded-commutative. We finally define the Hopf invariant of a map $f : \mathbb{S}^{2n-1} \to \mathbb{S}^n$ using the cup product structure on the pushout $\mathbf{1} \sqcup^{\mathbb{S}^{2n-1}} \mathbb{S}^n$, and we prove that for every even n, some particular map $\mathbb{S}^{2n-1} \to \mathbb{S}^n$ coming from the James construction has Hopf invariant 2. This shows that the number n defined in chapter 3 is equal to either 1 or 2 and that the group $\pi_{4n-1}(\mathbb{S}^{2n})$ is infinite for every natural number n.

Finally in chapter 6 we construct the Gysin exact sequence which is a long exact sequence of cohomology groups associated to every fibration where the base space is 1-connected and the fibers are spheres. This exact sequence describes some part of the multiplicative structure of the cohomology of the base space. We then define $\mathbb{C}P^2$ as the pushout $\mathbf{1} \sqcup^{\mathbb{S}^3} \mathbb{S}^2$ for the Hopf map $\mathbb{S}^3 \to \mathbb{S}^2$, we construct a fibration of circles above it in a way similar to the construction of the Hopf fibration, and we compute its cohomology ring using the Gysin exact sequence. This proves that the Hopf invariant of the Hopf map is equal to ± 1 and that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$.

In appendix A we present an elementary definition of weak ∞-groupoids, based on ideas coming from homotopy type theory, together with a proof that every type in homotopy type theory has the structure of a weak ∞-groupoid.

In appendix B we give a self-contained definition of the natural number n defined at the end of chapter 3 which satisfies $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$. The reason is that, as we will see later, computing this number from its definition is an important open problem in homotopy type theory, hence, for the benefit of people trying to solve it, it is convenient to have the complete definition all in one place.

Analytic versus synthetic The main difference between classical homotopy theory and homotopy type theory is that the first one is *analytic* whereas the second one is *synthetic*. To understand the difference between analytic and synthetic homotopy theory, it is helpful to go back to elementary geometry.

Analytic geometry is geometry in the sense of Descartes. The set $\mathbb{R}^2$ is our object of study, points are defined as pairs (x, y) of real numbers and lines are defined as sets of points satisfying an equation of the form $ax + by = c$. Then in order to prove something we use the properties of $\mathbb{R}^2$. For instance we can determine whether two lines intersect by solving a particular system of equations.

In contrast, synthetic geometry is geometry in the sense of Euclid. Points and lines are not defined in terms of other notions, they are just primitive notions, and a collection of axioms stipulating how they are supposed to behave is given. Then in order to prove something we have to use the axioms. For instance we cannot use the equation of a line or the coordinates of a point because lines do not have equations and points do not have coordinates.

Analytic geometry can be used to justify synthetic geometry. Indeed, analytic geometry gives a meaning to the notions of point and line and all the axioms of synthetic geometry can be proved to hold in analytic geometry. Therefore the axioms are consistent and everything which is true is synthetic geometry is also true in analytic geometry. The converse doesn't hold, so one could think that synthetic geometry is less powerful than analytic geometry as less theorems are provable. But from a different point of view, one can also argue that synthetic geometry is actually *more* powerful than analytic geometry because the theorems that can be proved are more general. They are true for any interpretation of the primitive notions for which the axioms are validated, whereas a proof in analytic geometry is by nature only valid in $\mathbb{R}^2$. Another disadvantage of analytic geometry is that because it reduces geometry to the resolution of equations, it is easy to lose track of the geometrical intuition. To sum up, in analytic geometry we give an explicit definition to the concepts we are interested in, and we can prove a lot of things about them, but we are restricted to this particular model, whereas in synthetic geometry we only axiomatize the basic properties of the concepts we are interested in, less theorems are provable, but they have a wider range of applicability and they are closer to the geometrical intuition.

The situation of homotopy theory is very similar. In analytic homotopy theory (or classical homotopy theory), the sphere $\mathbb{S}^n$ is defined as the set $\{(x_0, \dots, x_n) \in \mathbb{R}^{n+1}, x_0^2 + \cdots + x_n^2 = 1\}$ equipped with the appropriate topology, continuous maps are defined as functions preserving the topology in the appropriate way, and $\pi_4(\mathbb{S}^3)$ is defined as the quotient of the set of continuous pointed maps $\mathbb{S}^4 \to \mathbb{S}^3$ by the relation of homotopy. We can then use various techniques to prove that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$, i.e. that $\pi_4(\mathbb{S}^3)$ contains exactly two elements.

In synthetic homotopy theory, which is what this thesis is about, the notion of space does *not* come from topology. Instead it is axiomatized as a primitive notion (under the name *type*) together with primitive notions of *point* of a type and of *path* between two points. In particular, a path is not seen anymore as a continuous function from the

interval, it is a primitive notion. We also introduce a primitive notion of *continuous function*. Note that in classical homotopy theory, we need to define first what is a possibly-non-continuous function before being able to define what a continuous function is, but here we directly take the concept of continuous function as primitive. For us a continuous function is *not* a possibly-non-continuous function which has the additional property of being continuous, indeed there is not any notion of possibly-non-continuous function. Therefore, the adjective "continuous" is superfluous, and we will simply use the word "function" or "map" for what would be called "continuous function" in classical homotopy theory.

Various basic spaces are also axiomatized, for instance the space $\mathbb{N}$ of natural numbers is axiomatized together with an element 0, a function $S : \mathbb{N} \to \mathbb{N}$ and the principle of induction/recursion. The circle is axiomatized together with a point called base, a path called loop from base to base and a similar principle of induction/recursion stating intuitively that the circle is freely generated by base and loop. Similarly, we describe the higher-dimensional spheres $\mathbb{S}^n$ and the set of connected components of a space. Combining all of that with the notion of (continuous) functions mentioned above, we can define $\pi_4(\mathbb{S}^3)$ and we will see that we can still prove that it is isomorphic to the group $\mathbb{Z}/2\mathbb{Z}$.

Type theory Homotopy type theory is a variant of type theory and more precisely of Per Martin–Löf's intuitionistic theory of types (called simply *dependent type theory* here), which was introduced in the 1970s as a foundation for constructive mathematics (cf [ML75]). Constructive mathematics is a philosophy of mathematics based on the idea that in order to prove that a particular object exists, we have to give a method to construct it. It works by restricting the logical principles we are allowed to use and only allows those which are constructive. A proof in constructive mathematics isn't necessarily presented as an algorithm but an algorithm can always be extracted from it. Therefore constructive mathematics rejects principles like the axiom of choice, which asserts the existence of a function without giving a way to compute it, and reasoning by contradiction, which allows us to prove that something exists simply by proving that it cannot not exist. In particular, a proof that there exists a natural number having a specific property has to give (at least implicitly) a method to compute this number. This isn't true in classical mathematics. For instance, let's define $n \in \mathbb{N}$ as the smallest odd perfect number or 0 if no odd perfect number exists. In classical mathematics, this is a correct and complete definition of n, but it doesn't give any way to compute n. Indeed, at the time of writing it isn't known whether n is equal to 0 or not. On the other hand, this would not be considered a valid definition in constructive mathematics because we used the principle of excluded middle (either there exists an odd perfect number or there doesn't exist any) which isn't constructive. There are various flavors of constructive mathematics and note that the one we are using here, homotopy type theory, is not incompatible with classical logic. It would be perfectly possible to add the axiom of choice or excluded middle, but the drawback is that constructivity, which is one of the main advantages of type theory, would be lost.

In dependent type theory the primitive notions are *types* and *elements of types* (or *terms*). We write $u : A$ for the statement that u is an element of type A. Intuitively, one can think of a type as being something like a set, but there are several important differences with traditional set theory. Elements of types do not exist in isolation, they are always elements *of a given type* which is an intrisic part of the nature of the element. The type of an element is always known and it doesn't make sense to "prove" that an element u has type A. It is similar to the fact that it doesn't make sense to "prove" that $x^2 + y^2 = 0$ is an equation. Just looking at it we see that it is an equation and not a matrix. Moreover the type of an element is always unique (modulo computation rules as we will see later). For instance, we cannot say that the number 2 has both type $\mathbb{N}$ and type $\mathbb{Q}$. Instead there are two different elements, one of which is $2_{\mathbb{N}}$ of type $\mathbb{N}$ and the other is $2_{\mathbb{Q}}$ of type $\mathbb{Q}$ (which may both be written as 2 in a mathematical text if there is no risk of confusion) and they satisfy $i(2_{\mathbb{N}}) = 2_{\mathbb{Q}}$ for $i : \mathbb{N} \to \mathbb{Q}$ the canonical inclusion. Similarly, if we are given a rational number $q : \mathbb{Q}$, we cannot ask whether q has type $\mathbb{N}$. By nature q has type $\mathbb{Q}$ which is different from $\mathbb{N}$. What we can ask, however, is whether there exists a natural number $k : \mathbb{N}$ such that $i(k) = q$. This is what proving that q is a natural number would mean.

Mathematics is traditionally based on a two-layer system: the logical layer where propositions and proofs live and the mathematical layer where mathematical objects live. The logical layer is used to reason about the mathematical layer. For instance, constructing a specific mathematical object is an activity carried out in the mathematical layer, while proving a theorem happens in the logical layer. In dependent type theory those two layers are merged into one unique layer where types and their elements live. Apart from representing mathematical objects, types also play the role of (logical) propositions, and their elements play the role of "proofs" or witnesses of those propositions. Proving a given proposition is done by constructing an element of the corresponding type. For example, proving an implication $A \implies B$ corresponds to constructing an element in the function type $A \to B$, i.e. a function taking proofs of A to proofs of B. Proving a conjunction $A \wedge B$ corresponds to constructing an element in the product type $A \times B$, i.e. a pair composed of a proof of A and a proof of B. This correspondence between types and propositions and between elements of types and proofs is known as the *Curry–Howard correspondence*. We will sometimes distinguish between types "seen as propositions" and "seen as types" in order to explain the intuition between various constructions, but the difference between the two is often blurry. For instance, the type $A \simeq B$ can be seen both as the proposition "A and B are isomorphic" and as the type of all isomorphisms between them. Indeed, in constructive mathematics proving that A and B are isomorphic is the same thing as constructing an isomorphism between them.

The word "dependent" in "dependent type theory" refers to the fact that types can depend on elements of other types. Such types are called *dependent types* or *families of types*. Given a type A, having a dependent type B over A means that for every element $a : A$ there is a type $B(a)$. Dependent types are essential for the representation of quantified propositions as we see in chapter 1. For instance, a proposition depending on a natural number $n : \mathbb{N}$ is represented by a type depending on the variable n. A

dependent type B over A where all the types $B(a)$ are seen as propositions is called a *predicate on* A.

The constructivity property of dependent type theory enables one to see it as a programming language. In dependent type theory all primitive constructions have *computation rules* (or *reduction rules*), which essentially explain how to execute the programs of the language. All elements of types can then be seen as programs and can be executed, simply by repeatedly applying the computation rules. Note that in dependent type theory there are no infinite loops. All programs terminate and therefore a result is always obtained when executing a program. From the point of view of mathematics, the computation rules are the defining equations of the primitive constructions, and applying a computation rule corresponds to replacing something by its definition. Two elements u and v of a given type A are said to be *definitionally equal* (or *judgmentally equal*) if they become syntactically equal after replacing everything by their definition, i.e. after executing u and v. An important rule of type theory, known as the *conversion rule*, states that if u is of type A and A is definitionally equal to A', then u has also type A'. In particular, types are unique only up to definitional equality, but definitional equality is decidable because it is simply a matter of repeatedly unfolding the definitions. In the same way as it doesn't make sense to prove that a term u is of type A, it also doesn't make sense to prove that two terms or two types are definitionally equal. This is something that can simply be checked algorithmically.

Given the correspondence between proofs and elements of types it follows that proofs themselves can be executed, which is what gives dependent type theory its constructive nature. For instance, given a proof that there exists a natural number having a certain property, one can execute the proof and the final result will be a pair of the form (n, p) where n is a natural number of the form either 0, 1, 2, ... (i.e. we know its value) and p is a proof that n does satisfy the property. This close relation between type theory and computer science led to the development of *proof assistants* like Coq, Agda or Lean (see [Coq], [Agda], [Lean]). They are essentially type-checkers for dependent type theory together with various features making them easier to use. In a proof assistant, one can state a theorem by defining the corresponding type and then prove it by constructing a term (i.e. writing a program) having this type. If the proof assistant accepts it, it means that the program representing the proof is well-typed and that, therefore, the proof is correct.

Homotopy type theory Dependent type theory is very successful but suffers from a few problems, in particular when it comes to the treatment of equality. Given a type A and two elements $u, v : A$, the proposition "u is equal to v" is reified as a type $u =_A v$ called the *identity type* (whose elements are proofs that u is equal to v). Martin–Löf gave several versions of dependent type theory with different rules for the identity types. In one of them, called *extensional type theory*, the identity types are behaving in a nice way but typing is not decidable, i.e. there is no algorithm checking whether a term has a given type. This is usually an undesirable feature for a type theory. In another one, called *intensional type theory*, the rules of the identity types are different and typing is

decidable. However, the treatment of equality in intensional type theory is sometimes unsatisfactory. For instance, two functions $f, g : A \to B$ can satisfy $f(x) = g(x)$ for every $x : A$ without being equal themselves as functions. Defining the quotient of a set by an equivalence relation is also quite problematic. A different issue is that the principle of *uniqueness of identity proofs*, which states that for any $u, v : A$, any two proofs of $u =_A v$ are equal, isn't provable anymore, which is contrary to the intuition which was behind the identity types. Indeed, the idea of the identity types in Martin–Löf's type theory is that every type represents a set and that $u =_A v$ represents the set having exactly one element if u and v are equal, and the empty set if u and v are different.

Homotopy type theory is based on intensional type theory and resolves this last problem by changing the intuition behind types and the identity types. In homotopy type theory, types are not seen as sets anymore but as *spaces*, dependent types are seen as *fibrations*, and the identity type $u =_A v$ is seen as the space of all *continuous paths* from u to v in the space A. Rather surprisingly, it can be shown that under this interpretation, all rules of intensional type theory are still satisfied. Moreover, in this interpretation, uniqueness of identity proofs isn't a desirable property anymore. Given two points u and v in a space A there can be many non-homotopic paths from u to v and many non-homotopic homotopies between two paths, and so on.

This connection between type theory and homotopy theory was discovered around 2006 independently by Vladimir Voevodsky and by Steve Awodey and Michael Warren in [AW09]. Then in 2009 Vladimir Voevodsky stated the *univalence axiom*, proved its consistency in the simplicial set model, and started the project of formalizing mathematics in this system, intensional type theory with the univalence axiom, named *univalent foundations*. Given a universe Type, i.e. a type whose elements are themselves types, and two elements A and B of Type, the univalence axiom identifies the identity type $A =_{\mathsf{Type}} B$ with the type of equivalences $A \simeq B$. This axiom makes precise the idea that "isomorphic structures have the same properties", which is often used implicitly in mathematics. Note that it is not compatible with the principle of uniqueness of identity proofs because, for instance, it implies that there are two different equalities $\mathbf{2} =_{\mathsf{Type}} \mathbf{2}$ corresponding to the two bijections $\mathbf{2} \simeq \mathbf{2}$ (where $\mathbf{2}$ is the type with two elements). Voevodsky also noticed that the univalence axiom implies function extensionality, i.e. that if $f(x) =_B g(x)$ for all $x : A$, then $f =_{A \to B} g$, and that it makes the definition of quotients possible and well-behaved.

In 2011, the notion of *higher inductive types* started to emerge. Ordinary inductive types are types defined by giving some generators (the *constructors*) and an induction principle making precise the idea that the type is freely generated by the constructors. Higher inductive types are a generalization of ordinary inductive types where we can give not only point-constructors but also path-constructors. For instance, the circle has one point-constructor base and one path-constructor loop which is a path from base to base. In combination with univalence, fibrations can be defined by induction on the base space, which is a very powerful way of defining fibrations. For instance in order to define a fibration over the circle it is enough to give the fiber over base and the action of loop on this fiber (this action must be an equivalence).

One of the drawbacks of homotopy type theory is that by adding the univalence axiom or higher inductive types, we lose the constructivity property which, as we mentioned previously, is an essential feature of type theory. However, unlike the axiom of choice or excluded middle it is widely believed that the univalence axiom and higher inductive types are constructive in some way, and several people are trying to give an alternative description of homotopy type theory in which univalence and higher inductive types compute, see in particular [Coh+15]. A related conjecture is Voevodsky's homotopy canonicity conjecture: for every closed term $n : \mathbb{N}$ constructed using the univalence axiom, there exists a closed term $k : \mathbb{N}$ constructed *without* using the univalence axiom and a proof of $k =_{\mathbb{N}} n$.

Constructivity of $\pi_4(\mathbb{S}^3)$ The first major result of this thesis is corollary 3.4.5 which states that there exists a natural number n such that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$. This statement is quite curious, because it is a statement of the form "there exists a natural number n satisfying a given property" hence according to the constructivity conjecture it should be possible to extract from its proof the value of n. However, nobody has managed to do it so far, mainly because the proof is relatively complicated and that constructivity of the univalence axiom and of higher inductive types isn't very well understood yet. In chapters 4, 5 and 6 we present a proof that this number is equal to 2, but note that this is a mathematical proof, as opposed to a computation extracted from the definition of n, so it doesn't address the constructivity conjecture. However, it shows that we can define and work with cohomology and the Gysin sequence in homotopy type theory, which is interesting in its own right.

Models of homotopy type theory We do not talk much about the relationship between homotopy type theory (synthetic homotopy theory) and classical homotopy theory (analytic homotopy theory) in this thesis, apart from the fact that many definitions and proofs look quite similar to their classical counterpart. A construction of a model of homotopy type theory (minus higher inductive types) in classical homotopy theory is presented in [KLV12] and a proof that they also model higher inductive types is in preparation in [LS]. As we mentioned previously, one of the consequence of working synthetically is that all the work done in this thesis is also valid in any other model of homotopy type theory, not only the classical one. Michael Shulman gave in [Shu15] various other models of homotopy type theory and it is widely believed that any ∞-topos in the sense of Lurie (cf [Lur09]) gives a model of homotopy type theory.

Another very important model is the model of Thierry Coquand et al. described [BCH14], which is a constructive model of homotopy type theory in cubical sets. Note that, in theory, this model should allow us to compute the number n of chapter 3, but this hasn't been done at the time of writing. This model also suggests a different version of homotopy type theory, called *cubical type theory* (cf. [Coh+15]), but in this work we decided to stay with the type theory used in [UF13]. Various squares and cubes are nevertheless used whenever convenient.

Chapter 1

Homotopy type theory

In this first chapter we give an introduction to homotopy type theory and to a few basic results that are used throughout this work. The reader is encouraged to read [UF13] for a more comprehensive presentation of homotopy type theory. Unlike in [UF13], we do not notate definitional equalities differently from propositional equalities, we simply use the terminology "$u = v$ by definition" when we want to insist on the fact that the equality is definitional. We also use the standard notation $u := v$ when *introducing* new definitions. We use the word "proposition" in its standard mathematical meaning. A proposition is either a statement which might be true or false, for instance "the negation of the proposition $1 + 1 = 2$ is the proposition $1 + 1 \neq 2$", or a statement for which we do provide a proof. In particular when we say that a type is "seen as a proposition" or when we state a proposition, it doesn't mean that the type in consideration is assumed to be (-1)-truncated in the sense of section 1.10. We reserve the expression "mere proposition" for such types.

All types are seen as elements of a particular type called Type. For consistency reasons, Type cannot be an element of itself so we have an infinite sequence of universes Type_0, Type_1, Type_2, ..., with $\mathsf{Type}_n : \mathsf{Type}_{n+1}$ for every n. In practice, though, we rarely need to worry about which universe we are in, so from now on we simply write Type for any of the Type_n, as is often done in type theory.

1.1 Function types

We first present *function types*. Given two types A and B, there is a type written

$$A \to B$$

representing the type of functions from A to B. A function can be defined by an explicit formula as follows:

$$f : A \to B,$$
$$f(x) := \Phi[x],$$

where $\Phi[x]$ is a syntactical expression which may use the variable x (and usually do, unless the function is constant) and which is of type B when we assume that x is of type A. We can also use the notation $\lambda x.\Phi[x]$ which is the same thing as the function f above except that it avoids the need to give it a name. Given a function $f : A \to B$ and an element $a : A$, we can apply f to a and we obtain an element of type B,

$$f(a) : B.$$

Moreover, if f is defined as above then $f(a)$ is equal to $\Phi[a/x]$ by definition, where $\Phi[a/x]$ is the expression $\Phi[x]$ where all instances of the variable x have been replaced by a.

When we see A and B as spaces, an element of $A \to B$ should be thought of as a *continuous* function from A to B. When we see A and B as propositions, an element of $A \to B$ is a function turning a proof of A into a proof of B. In other words, it corresponds to a proof of "A implies B". In particular, this means that logical implications are translated into function types in type theory.

An element $P : A \to \mathsf{Type}$ is called a *dependent type* over A and it represents a family of types indexed by A, or a fibration over A if A and all types $P(a)$ are seen as spaces, or a predicate on A if all types $P(a)$ are seen as propositions.

Definition 1.1.1. Given a type A, the *identity function* of A is the function

$$\begin{aligned} \mathsf{id}_A &: A \to A, \\ \mathsf{id}_A(x) &:= x. \end{aligned}$$

Definition 1.1.2. Given three types A, B and C and two functions $f : A \to B$ and $g : B \to C$, the *composition* of f and g is the function

$$\begin{aligned} g \circ f &: A \to C, \\ (g \circ f)(x) &:= g(f(x)). \end{aligned}$$

1.1.1 Dependent functions

A function $f : A \to B$ always returns an element of type B no matter what its argument is. It is possible to generalize function types in order to allow the output type to depend on the value of the input. More precisely, given a type A and a dependent type $B : A \to \mathsf{Type}$, there is a type written

$$(x : A) \to B(x) \quad \text{or} \quad \prod_{x:A} B(x)$$

representing *dependent functions* from A to B, i.e. functions sending an element x of A to an element of the corresponding type $B(x)$. Just as with regular functions, a dependent function can be defined by an explicit formula as follows:

$$\begin{aligned} f &: (x : A) \to B(x), \\ f(x) &:= \Phi[x], \end{aligned}$$

where $\Phi[x]$ is an expression of type $B(x)$, and we can also write it $\lambda x.\Phi[x]$. When we apply a dependent function $f : (x : A) \to B(x)$ to an element $a : A$, we get an element of type $B(a)$ (which depends on a in general)

$$f(a) : B(a).$$

When we see A as a space and B as a fibration over A, a dependent function $f : (x : A) \to B(x)$ should be seen as a continuous section of B. When we see A as a space and B as a predicate on A, the dependent function type $(x : A) \to B(x)$ corresponds to the universally quantified proposition $\forall x : A, B(x)$. Indeed, proving the proposition $\forall x : A, B(x)$ corresponds to proving $B(x)$ for every x in A, which is exactly what a dependent function of type $(x : A) \to B(x)$ does.

For instance, let us assume we have a type A seen as a space and a dependent type B over A seen as a fibration over A. Then a theorem of the form "For every section f of B, if $P(f)$ holds then $Q(f)$ holds" should be interpreted as the type

$$(f : (x : A) \to B(x)) \to (P(f) \to Q(f)).$$

The arrow on the left represents the type of sections of B, the arrow on the right represents the logical implication between $P(f)$ and $Q(f)$ and the arrow in the middle represents universal quantification. A proof of such a theorem is a function taking a function f of type $(x : A) \to B(x)$ (i.e. f is a function taking an argument x of type A and returning a result of type $B(x)$) and returning a function of type $P(f) \to Q(f)$, i.e. which takes an element of type $P(f)$ (a proof that f satisfies P) and returns an element of type $Q(f)$ (a proof that f satisfies Q).

1.1.2 Functions with several arguments

There are several ways to talk about functions with several arguments. Let's say for instance that we are interested in a function f taking two arguments, of types A and B, and returning a result of type C. One way to state it is to say that f has type $(A \times B) \to C$, i.e. f takes one argument of the product type $A \times B$ (that we define in the next section) and returns an element of C. Another way to state it is to say that f has type $A \to (B \to C)$, i.e. f takes one argument of type A and returns another function taking the second argument of type B and returning the result of type C. The two versions turn out to be equivalent, and in general we use the second version (called the *curried* form), as is common in type theory, with the syntax

$$\begin{aligned} f &: A \to B \to C, \\ f(a, b) &:= \Phi[a, b]. \end{aligned}$$

Of course, the type B could be a dependent type over A and the type C could be a dependent type over both A and B, and it can be generalized to functions with more than two arguments.

1.2 Pair types

We now present *pair types*. Given two types A and B, there is a type written

$$A \times B$$

representing the type of pairs consisting in one element of A and one element of B. One can construct an element of $A \times B$ by pairing one element a of A and one element b of B:

$$(a, b) : A \times B.$$

One can deconstruct an element of $A \times B$ as follows. If $P : A \times B \to \mathsf{Type}$ is a dependent type over $A \times B$, then a section of it can be defined by

$$\begin{aligned} f &: (z : A \times B) \to P(z), \\ f((x, y)) &:= f_\times(x, y), \end{aligned}$$

where

$$f_\times : (x : A)(y : B) \to P((x, y)).$$

For instance, we can define the first and the second projection by

$$\begin{aligned} \mathsf{fst} &: A \times B \to A, & \mathsf{snd} &: A \times B \to B, \\ \mathsf{fst}((a, b)) &:= a, & \mathsf{snd}((a, b)) &:= b. \end{aligned}$$

When we see A and B as propositions, the type $A \times B$ represents the *conjunction* of A and B. Indeed proving that "A and B" holds is equivalent to proving that both A and B hold, therefore a proof of "A and B" can be seen as a pair (a, b) where a is a proof of A and b is a proof of B.

1.2.1 Dependent pairs

The second component of an element of $A \times B$ always has type B. It is possible to generalize pair types in order to allow the type of the second component to depend on the value of the first component. More precisely, given a type A and a dependent type B over A, there is a type written

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

representing *dependent pairs*. Such types are often called Σ-*types*. Given $a : A$ and $b : B(a)$, we can construct the dependent pair

$$(a, b) : \sum_{x:A} B(x).$$

One can define a function out of it in the same way as for non-dependent pair types. For instance the first and second projections are defined by

$$\mathsf{fst} : \sum_{x:A} B(x) \to A, \qquad \mathsf{snd} : \left(z : \sum_{x:A} B(x)\right) \to B(\mathsf{fst}(z)),$$
$$\mathsf{fst}((x,y)) := x, \qquad \mathsf{snd}((x,y)) := y.$$

Note that, this time, the second projection is a dependent function because the type of the second component of a dependent pair depends on the first component.

It is possible to nest Σ-types in order to obtain types of arbitrary n-tuples. For instance the type of semigroups can be defined as the type

$$\mathsf{SemiGroup} : \mathsf{Type},$$
$$\mathsf{SemiGroup} := \sum_{G:\mathsf{Type}} \sum_{m:G\to G\to G} ((x,y,z:G) \to m(m(x,y),z) =_G m(x,m(y,z))).$$

In other words, a semigroup is a triple $(G,(m,a))$ where G is a type, m is a function of type $G \to G \to G$ (the multiplication operation), and a is a proof of associativity of m, i.e., a is a function taking three arguments x, y and z of type G and returning an equality between $m(m(x,y),z)$ and $m(x,m(y,z))$. Note that the type of m depends on G, and that in turn the type of a depends on m.

When we see B as a fibration over A, the type $\sum_{x:A} B(x)$ corresponds to the total space of B. This will be used in particular in the flattening lemma in section 1.9. When we see B as a predicate on A, the type $\sum_{x:A} B(x)$ corresponds to the type of elements of A which satisfy B. Note that there is a subtlety here because if for some $a : A$ there are several distinct elements in $B(a)$, then a is counted several times which isn't what we want in general. We can also see $\sum_{x:A} B(x)$ as corresponding to the proposition "there exists an $x : A$ satisfying $B(x)$". Indeed, one can prove this proposition by exhibiting an $x : A$ and a proof that $B(x)$ holds, i.e. an element of $\sum_{x:A} B(x)$. However this would be more accurately called *explicit existence* because it requires us to choose an explicit x satisfying B, which might be a too strong requirement in some cases. We will come back to both problems in section 1.10.3.

1.3 Inductive types

We now present inductive types, which give a wide variety of type formers, including base types. The general idea is that an *inductive type* T is presented by a list of *constructors* which describe all the different ways of constructing elements of T and, in some sense which is made precise by an *induction principle* (or *elimination rule*), the only elements of T are those given by the constructors. In this section, all equalities (introduced by the symbol :=) are equalities by definition. We now give various examples of inductive types.

Natural numbers The canonical example of an inductive type is the type of *natural numbers* $\mathbb{N}$. The two constructors are

$$\begin{aligned} 0 &: \mathbb{N}, \\ \mathsf{S} &: \mathbb{N} \to \mathbb{N}. \end{aligned}$$

In other words there are two ways to construct a natural number: either we take 0 or we take the successor of an already constructed natural number. We use the usual notation $1 := \mathsf{S}(0)$, $2 := \mathsf{S}(\mathsf{S}(0))$, and so on, and we write $n + 1$ for $\mathsf{S}(n)$.

The induction principle states that given a dependent type P over $\mathbb{N}$, we can define a section of it by giving f_0 and f_{S} as follows:

$$\begin{aligned} f &: (n : \mathbb{N}) \to P(n), \\ f(0) &:= f_0, \\ f(n+1) &:= f_{\mathsf{S}}(n, f(n)), \end{aligned}$$

where we have

$$\begin{aligned} f_0 &: P(0), \\ f_{\mathsf{S}} &: (n : \mathbb{N}) \to P(n) \to P(n+1). \end{aligned}$$

For instance, one can define addition and multiplication on natural numbers by

$$\begin{aligned} \mathsf{add} &: \mathbb{N} \to \mathbb{N} \to \mathbb{N}, \\ \mathsf{add}(0, n) &:= n, \\ \mathsf{add}(m+1, n) &:= \mathsf{add}(m, n) + 1, \end{aligned} \qquad \begin{aligned} \mathsf{mul} &: \mathbb{N} \to \mathbb{N} \to \mathbb{N}, \\ \mathsf{mul}(0, n) &:= 0, \\ \mathsf{mul}(m+1, n) &:= \mathsf{add}(\mathsf{mul}(m, n), n). \end{aligned}$$

The unit type The *unit type* is the inductive type $\mathbf{1}$ with one constructor

$$\star_{\mathbf{1}} : \mathbf{1}.$$

Its induction principle states that if $P(x)$ is a dependent type over $x : \mathbf{1}$, then we can construct a section of $P(x)$ by

$$\begin{aligned} f &: (x : \mathbf{1}) \to P(x), \\ f(\star_{\mathbf{1}}) &:= f_{\star_{\mathbf{1}}}, \end{aligned}$$

where $f_{\star_{\mathbf{1}}} : P(\star_{\mathbf{1}})$.

The type of booleans The *type of booleans* or 2*-element type* is the inductive type $\mathbf{2}$ with two constructors

$$\mathsf{true}, \mathsf{false} : \mathbf{2}.$$

Its induction principle states that if $P(x)$ is a dependent type over $x : \mathbf{2}$, then we can construct a section of $P(x)$ by

$$\begin{aligned} f &: (x : \mathbf{2}) \to P(x), \\ f(\mathsf{true}) &:= f_{\mathsf{true}}, \\ f(\mathsf{false}) &:= f_{\mathsf{false}}, \end{aligned}$$

where $f_{\mathsf{true}} : P(\mathsf{true})$ and $f_{\mathsf{false}} : P(\mathsf{false})$.

Disjoint sum Given two types A and B, their *disjoint sum* is the inductive type $A+B$ with the two constructors

$$\begin{aligned} \mathsf{inl} &: A \to A + B, \\ \mathsf{inr} &: B \to A + B. \end{aligned}$$

Its induction principle states that if $P(x)$ is a dependent type over $x : A + B$, then we can construct a section of $P(x)$ by

$$\begin{aligned} f &: (x : A + B) \to P(x), \\ f(\mathsf{inl}(a)) &:= f_{\mathsf{inl}}(a), \\ f(\mathsf{inr}(b)) &:= f_{\mathsf{inr}}(b), \end{aligned}$$

where

$$\begin{aligned} f_{\mathsf{inl}} &: (a : A) \to P(\mathsf{inl}(a)), \\ f_{\mathsf{inr}} &: (b : B) \to P(\mathsf{inr}(b)). \end{aligned}$$

This type can also be used to represent *explicit disjunction*. Indeed, when A and B are seen as propositions, an element of $A + B$ is either a proof of A or a proof of B, hence it's a proof of "A or B". As for explicit existence, the drawback is that it requires us to choose whether we proved the left-hand side or the right-hand side which might be a too strong requirement. We will come back to that in section 1.10.3.

Integers The type of *integers* is the inductive type $\mathbb{Z}$ with the three constructors

$$\begin{aligned} \mathsf{neg} &: \mathbb{N} \to \mathbb{Z}, \\ 0_{\mathbb{Z}} &: \mathbb{Z}, \\ \mathsf{pos} &: \mathbb{N} \to \mathbb{Z}. \end{aligned}$$

An integer is either $0_{\mathbb{Z}}$, $\mathsf{pos}(n)$ for $n : \mathbb{N}$ (which represents $n + 1$) or $\mathsf{neg}(n)$ for $n : \mathbb{N}$ (which represents $-(n + 1)$). There is again an induction principle derived from the

constructors. For instance one can define the operation of adding one by

$$\begin{aligned} \mathsf{succ}_{\mathbb{Z}} &: \mathbb{Z} \to \mathbb{Z}, \\ \mathsf{succ}_{\mathbb{Z}}(\mathsf{neg}(n+1)) &:= \mathsf{neg}(n), \\ \mathsf{succ}_{\mathbb{Z}}(\mathsf{neg}(0)) &:= 0_{\mathbb{Z}}, \\ \mathsf{succ}_{\mathbb{Z}}(0_{\mathbb{Z}}) &:= \mathsf{pos}(0), \\ \mathsf{succ}_{\mathbb{Z}}(\mathsf{pos}(n)) &:= \mathsf{pos}(n+1). \end{aligned}$$

Note that here we have used both the induction principle for $\mathbb{Z}$ and the one for $\mathbb{N}$ (in the case neg).

The empty type The *empty type* is the inductive type $\bot$ (also called $\mathbf{0}$) without any constructor. In particular there is no way to construct an element of it. Its induction principle states that any dependent type over $\bot$ has a section. For instance there is a canonical map $\bot \to A$ for any type A.

The empty type is used to represent the proposition False. Indeed, False is the proposition that doesn't have any proof. It is also used for defining negation. When A is a type seen as a proposition, the type representing its negation is $\neg A := (A \to \bot)$. For instance, we can prove the principle of reasoning by contraposition by

$$\begin{aligned} \mathsf{contraposition} &: (A \to B) \to (\neg B \to \neg A), \\ \mathsf{contraposition}(f, b', a) &:= b'(f(a)). \end{aligned}$$

Here f is a function from A to B, b' is a function from B to $\bot$ and a is a element of A, therefore by applying f and then b' to a we get an element of type $\bot$ which is what we wanted.

General inductive types More generally one can introduce new inductive types whenever needed, but there are a few constraints that have to be satisfied in order for them to make sense and be consistent. In particular:

- Every constructor must end with the type being defined, i.e. a constructor cannot give elements in a different type (we will slightly relax this condition in section 1.7 on higher inductive types).
- The recursive occurrences of the type being defined have to be in strictly positive positions, i.e. they can only appear as the codomain of an argument of the constructors.

We refer to [UF13, chapter 5] for more details on inductive types.

1.4 Identity types

Given a type A and two elements u and v of type A, there is a type $u =_A v$ (or simply $u = v$ when A is clear from the context) called the *identity type* or *equality type*. Its

elements are called *paths*, *equalities* or *identifications*. The idea is that when we see A as a space, the type $u =_A v$ corresponds to the space of (continuous) paths from u to v. To understand the notation, note that if A is just a set seen as a discrete space, then there is a path between u and v if and only if u and v are equal. Therefore, in that case the type $u =_A v$ does correspond to the logical proposition "u is equal to v".

For every point $a : A$, there is a path

$$\mathsf{idp}_a : a =_A a$$

which we call the *constant path at* a. Moreover, we have an "induction principle". Given an point $a : A$, a dependent type $P : (x : A)(p : a =_A x) \to \mathsf{Type}$ and an element $d : P(a, \mathsf{idp}_a)$, there is a function

$$\mathsf{J}_P(d) : (x : A)(p : a =_A x) \to P(x, p)$$

together with an equality (called *the computation rule of* J)

$$\mathsf{J}_P(d)(a, \mathsf{idp}_a) = d. \tag{$\mathsf{J}_{\mathsf{comp}}$}$$

We refer to the use of this operator J as *path induction*. The idea is that when we want to prove/construct something depending on a path p where one endpoint of p is free (the x above), then it is enough to prove/construct it in the case where p is the constant path. It is important that one of the endpoints be free, for instance if we only have $P : (a =_A a) \to \mathsf{Type}$ and $d : P(\mathsf{idp}_a)$, we *cannot* deduce that $P(p)$ hold for every $p : a =_A a$. This last statement is known as the "K rule" and is equivalent to the principle of uniqueness of identity proofs which we do not want.

There is some debate on whether the equality ($\mathsf{J}_{\mathsf{comp}}$) should be taken as a definitional equality or not. On the one hand it was assumed definitional in the original definition of the identity type by Per Martin–Löf in [ML75] and most of the work done in homotopy type theory so far has been done assuming it is definitional, in particular the reference book [UF13]. On the other hand the original motivation for having a definitional equality was that the family of identity types was supposed to be inhabited only by elements of the form idp_a, but homotopy type theory challenges this intuition by adding new elements to the identity types via the univalence axiom and higher inductive types. In this thesis we assume that it holds definitionally, but the only place where we really use it is in the definition of the structure of weak ∞-groupoid, and we conjecture that it is not actually required.

1.4.1 The weak ∞-groupoid structure of types

The path induction principle allows us to equip every type A with a very rich structure making A into what is known as an *weak ∞-groupoid*. In this chapter we give an intuitive presentation by giving many examples of operations part of this structure, and we give a precise definition of weak ∞-groupoids in appendix A.

The first thing to notice is that the identity type operation can be iterated. If we have $p, q : u =_A v$ we can consider the type $p =_{u=_A v} q$, and if $\alpha, \beta : p =_{u=_A v} q$ we can consider

the type $\alpha =_{p=_{u=_A v} q} \beta$, and so on. If we see p and q as proofs of equality between u and v, it seems rather odd to talk about equalities between them, let alone equalities between equalities between them. However, when seeing u and v as two points in a space A and p and q as two paths between u and v, it makes sense to think of $p =_{u=_A v} q$ as the type of homotopies between p and q and then $\alpha =_{p=_{u=_A v} q} \beta$ as the type of homotopies between the homotopies α and β, and so on. Let's now look at several examples of operations (called *coherence operations*) that we can do on those paths and higher paths.

Inverse If $p : a = b$ is a path in A, then there is a path $p^{-1} : b = a$ called the *inverse path* of p. We write this operation as follows:

$$(a, b : A)(p : a = b) \mapsto (p^{-1} : b = a).$$

To define it, the idea is to do a path induction on p, where the dependent type P is $P(b, p) := (b = a)$. We then need to give an element of $P(a, \mathsf{idp}_a)$ (i.e. $a = a$) and we use idp_a. In particular, we get the equality by definition

$$\mathsf{idp}_a^{-1} := \mathsf{idp}_a.$$

If we do not take ($\mathsf{J}_{\mathsf{comp}}$) as an equality by definition, then the equality still holds but only in the sense that there is a path

$$\mathsf{inv\text{-}def} : \mathsf{idp}_a^{-1} = \mathsf{idp}_a.$$

Composition If $p : a = b$ and $q : b = c$ are two paths in A, then there is a path $p \cdot q : a = c$ called the *composition* of p and q. This operation is written as

$$(a, b : A)(p : a = b)(c : A)(q : b = c) \mapsto (p \cdot q : a = c).$$

It is obtained by applying the J rule successively to q and p and we have the equality

$$\mathsf{idp}_a \cdot \mathsf{idp}_a := \mathsf{idp}_a$$

by definition. If ($\mathsf{J}_{\mathsf{comp}}$) is not an equality by definition, then we have instead a term

$$\mathsf{comp\text{-}def} : \mathsf{idp}_a \cdot \mathsf{idp}_a = \mathsf{idp}_a.$$

Associativity If we take three composable paths $p : a = b$, $q : b = c$ and $r : c = d$, then there are two ways to compose them and we define an operation as follows:

$$(a, b : A)(p : a = b)(c : A)(q : b = c)(d : A)(r : c = d) \mapsto (\alpha_{p,q,r} : (p \cdot q) \cdot r = p \cdot (q \cdot r)).$$

By path induction on r, q and p, it suffices to give $\alpha_{\mathsf{idp}_a,\mathsf{idp}_a,\mathsf{idp}_a}$ of type $(\mathsf{idp}_a \cdot \mathsf{idp}_a) \cdot \mathsf{idp}_a = \mathsf{idp}_a \cdot (\mathsf{idp}_a \cdot \mathsf{idp}_a)$. But we have $\mathsf{idp}_a \cdot \mathsf{idp}_a = \mathsf{idp}_a$ by definition, therefore the equality holds by definition and $\mathsf{idp}_{\mathsf{idp}_a}$ fits:

$$\alpha_{\mathsf{idp}_a,\mathsf{idp}_a,\mathsf{idp}_a} := \mathsf{idp}_{\mathsf{idp}_a}.$$

Note that if we do not take ($\mathsf{J}_{\mathsf{comp}}$) as an equality by definition, then it is still possible to find a term of type $(\mathsf{idp}_a \cdot \mathsf{idp}_a) \cdot \mathsf{idp}_a = \mathsf{idp}_a \cdot (\mathsf{idp}_a \cdot \mathsf{idp}_a)$ but it is more complicated to define as we need to explicitly combine several uses of $\mathsf{comp\text{-}def}$. The complexity would increase even more for more complicated coherence operations.

Inverse and identity laws Similarly we have the inverse and identity laws:

$$(a, b : A)(p : a = b) \mapsto (\zeta_p : p \cdot p^{-1} =_{a=a} \mathsf{idp}_a),$$
$$(a, b : A)(p : a = b) \mapsto (\eta_p : p^{-1} \cdot p =_{b=b} \mathsf{idp}_b),$$
$$(a, b : A)(p : a = b) \mapsto (\rho_p : p \cdot \mathsf{idp}_b =_{a=b} p),$$
$$(a, b : A)(p : a = b) \mapsto (\lambda_p : \mathsf{idp}_a \cdot p =_{a=b} p).$$

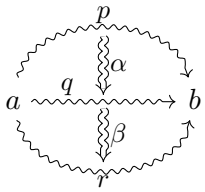
These four operations are defined by path induction on p, using the fact that $\mathsf{idp}_a^{-1} = \mathsf{idp}_a$ and $\mathsf{idp}_a \cdot \mathsf{idp}_a = \mathsf{idp}_a$ by definition, and we have

$$\zeta_{\mathsf{idp}_a} := \mathsf{idp}_{\mathsf{idp}_a}, \qquad \eta_{\mathsf{idp}_a} := \mathsf{idp}_{\mathsf{idp}_a},$$
$$\rho_{\mathsf{idp}_a} := \mathsf{idp}_{\mathsf{idp}_a}, \qquad \lambda_{\mathsf{idp}_a} := \mathsf{idp}_{\mathsf{idp}_a}.$$

Vertical and horizontal composition Higher dimensional paths can be composed in a variety of ways. For 2-dimensional paths, we first consider *vertical composition*, which is the operation

$$(a, b : A)(p, q : a = b)(\alpha : p =_{a=b} q)(r : a = b)(\beta : q =_{a=b} r) \mapsto \alpha \cdot \beta : p =_{a=b} r$$

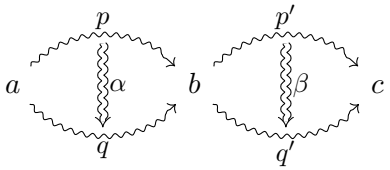
and which corresponds to the diagram



Note that vertical composition is the same thing as regular composition in the type $a = b$, therefore we use the same notation. We also have *horizontal composition*, which is the operation

$$(a, b : A)(p, q : a = b)(\alpha : p =_{a=b} q)$$
$$(c : A)(p', q' : b = c)(\beta : p' =_{b=c} q') \mapsto \alpha \star \beta : (p \cdot p') =_{a=c} (q \cdot q')$$

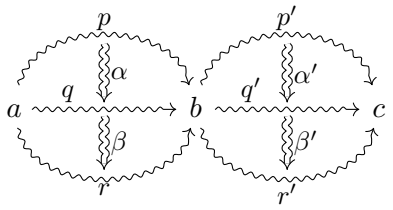
and which corresponds to the diagram



These two operations are defined by successive path inductions and we have

$$\mathsf{idp}_{\mathsf{idp}_a} \cdot \mathsf{idp}_{\mathsf{idp}_a} := \mathsf{idp}_{\mathsf{idp}_a},$$
$$\mathsf{idp}_{\mathsf{idp}_a} \star \mathsf{idp}_{\mathsf{idp}_a} := \mathsf{idp}_{\mathsf{idp}_a}.$$

Exchange law Given four 2-dimensional paths as follows:



we can consider the two compositions

$$(\alpha \cdot \beta) \star (\alpha' \cdot \beta') : p \cdot p' = r \cdot r'$$

and

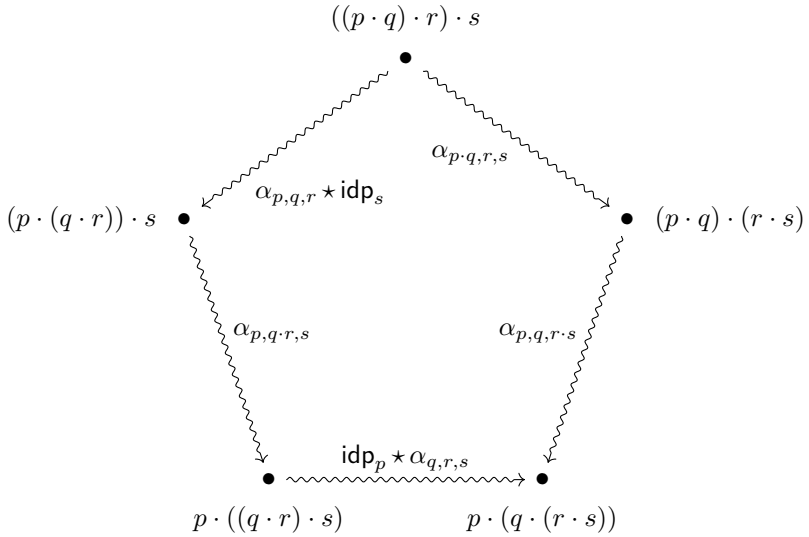
$$(\alpha \star \alpha') \cdot (\beta \star \beta') : p \cdot p' = r \cdot r'.$$

The *exchange law* states that there is a 3-dimensional path equating them and is defined by successive path inductions, using the fact that

$$(\mathsf{idp}_{\mathsf{idp}_a} \cdot \mathsf{idp}_{\mathsf{idp}_a}) \star (\mathsf{idp}_{\mathsf{idp}_a} \cdot \mathsf{idp}_{\mathsf{idp}_a}) = \mathsf{idp}_{\mathsf{idp}_a},$$
$$(\mathsf{idp}_{\mathsf{idp}_a} \star \mathsf{idp}_{\mathsf{idp}_a}) \cdot (\mathsf{idp}_{\mathsf{idp}_a} \star \mathsf{idp}_{\mathsf{idp}_a}) = \mathsf{idp}_{\mathsf{idp}_a}$$

by definition. We use it in the Eckmann–Hilton argument in proposition 2.1.6.

Pentagon of associativities Given four composable paths p, q, r and s, there are two different ways to go from $((p \cdot q) \cdot r) \cdot s$ to $p \cdot (q \cdot (r \cdot s))$, and the following pentagon states that there is a 3-dimensional path between them:



It corresponds to the coherence operation

$$(a, b : A)(p : a = b)(c : A)(q : b = c)(d : A)(r : c = d)(e : A)(s : d = e) \mapsto$$
$$(\pi_p : (\alpha_{p\cdot q,r,s} \cdot \alpha_{p,q,r\cdot s}) = (\alpha_{p,q,r} \star \mathsf{idp}_s) \cdot \alpha_{p,q\cdot r,s} \cdot (\mathsf{idp}_p \star \alpha_{q,r,s})).$$

and it is defined by successive path inductions and using the fact that

$$\alpha_{\mathsf{idp}_a \cdot \mathsf{idp}_a, \mathsf{idp}_a, \mathsf{idp}_a} \cdot \alpha_{\mathsf{idp}_a, \mathsf{idp}_a, \mathsf{idp}_a \cdot \mathsf{idp}_a} = \mathsf{idp}_{\mathsf{idp}_a}$$

and

$$(\alpha_{\mathsf{idp}_a, \mathsf{idp}_a, \mathsf{idp}_a} \star \mathsf{idp}_{\mathsf{idp}_a}) \cdot \alpha_{\mathsf{idp}_a, \mathsf{idp}_a \cdot \mathsf{idp}_a, \mathsf{idp}_a} \cdot (\mathsf{idp}_{\mathsf{idp}_a} \star \alpha_{\mathsf{idp}_a, \mathsf{idp}_a, \mathsf{idp}_a}) = \mathsf{idp}_{\mathsf{idp}_a}$$

by definition.

General coherence operations The operations we just presented are only examples of coherence operations. There are many others and in all dimensions. The structure of a general coherence operation is as follows: the list of arguments needs to form a "contractible" shape in the sense that one can apply J to it until there is only one point $a : A$ left, and the return type can be any iterated identity type between two terms built using the variables and other coherence operations. All coherence operations have the property that they are equal by definition to the iterated constant path when applied only to constant paths. This property allows us to define coherence operations by successive path inductions as we did above in the examples. A precise definition of weak ∞-groupoids together with a proof that path induction gives such a structure on types is presented in appendix A.

1.4.2 Continuity of maps

Given a map $f : A \to B$ and a path $p : a =_A b$ in A, we can *apply* f to p, and obtain a path in B between $f(a)$ and $f(b)$:

$$\mathsf{ap}_f(p) : f(a) =_B f(b).$$

This path is defined by path induction on p and we have the equality

$$\mathsf{ap}_f(\mathsf{idp}_a) = \mathsf{idp}_{f(a)}.$$

Here are some useful properties of ap, which can all be proved by path induction.

- If $f = (\lambda x.a)$ is a constant function, then $\mathsf{ap}_f(p) = \mathsf{idp}_a$ for every p.
- If $f = (\lambda x.x)$ is the identity function, then $\mathsf{ap}_f(p) = p$ for every p.
- If $f = g \circ h$, then $\mathsf{ap}_{g \circ h}(p) = \mathsf{ap}_g(\mathsf{ap}_h(p))$.

Note that we have defined ap only for non-dependent functions, we will see later how to define it for dependent functions as well.

If we see $a =_A b$ as the proposition "a and b are equal", then ap simply states that application of functions respects equality, which is something very reasonable to ask. However, if we see $a =_A b$ as the space of paths from a to b, then ap states that we can

apply functions not only to points but also to paths. One can also apply functions to 2-dimensional paths by

$$\begin{aligned}\mathsf{ap}^2_f &: (p =_{a=_A b} q) \to (\mathsf{ap}_f(p) =_{f(a)=_B f(b)} \mathsf{ap}_f(q)),\\ \mathsf{ap}^2_f(p) &:= \mathsf{ap}_{\mathsf{ap}_f}(p).\end{aligned}$$

The fact that we can apply functions to paths and to higher dimensional paths is compatible with the intuition that, in homotopy type theory, all functions are continuous.

When we see types as weak ∞-groupoid as sketched previously, then any map $f : A \to B$ should be seen as an ∞-*functor*. Indeed , it turns out that (the iterated versions of) ap_f commute with all coherence operations. For instance given any two composable paths $p : a =_A b$ and $q : b =_A c$ in A, we have an equality

$$\mathsf{ap}_f(p \cdot q) = \mathsf{ap}_f(p) \cdot \mathsf{ap}_f(q),$$

which can be proved by path induction on p and q. It also holds for higher coherences but it is more complicated to write. For instance for associativity, given $p : a =_A b$, $q : b =_A c$ and $r : c =_A d$ we have

$$\begin{aligned}\mathsf{ap}^2_f(\alpha_{p,q,r}) &: \mathsf{ap}_f((p \cdot q) \cdot r) = \mathsf{ap}_f(p \cdot (q \cdot r)),\\ \alpha_{\mathsf{ap}_f(p),\mathsf{ap}_f(q),\mathsf{ap}_f(r)} &: ((\mathsf{ap}_f(p) \cdot \mathsf{ap}_f(q)) \cdot \mathsf{ap}_f(r)) = (\mathsf{ap}_f(p) \cdot (\mathsf{ap}_f(q) \cdot \mathsf{ap}_f(r)))\end{aligned}$$

and the commutation of ap^2_f with α states that they are equal after composition with the two paths

$$\begin{aligned}\mathsf{ap}_f((p \cdot q) \cdot r) &= (\mathsf{ap}_f(p) \cdot \mathsf{ap}_f(q)) \cdot \mathsf{ap}_f(r),\\ \mathsf{ap}_f(p \cdot (q \cdot r)) &= \mathsf{ap}_f(p) \cdot (\mathsf{ap}_f(q) \cdot \mathsf{ap}_f(r))\end{aligned}$$

constructed using the fact that ap_f commutes with composition of paths.

1.5 The univalence axiom

Given two types A and B, we can consider the identity type $A =_{\mathsf{Type}} B$ of paths between A and B in Type. An equality between two types allows us to transport elements from one type to the other, we have the two functions

$$\begin{aligned}\mathsf{coe} &: (A =_{\mathsf{Type}} B) \to (A \to B),\\ \mathsf{coe}^{-1} &: (A =_{\mathsf{Type}} B) \to (B \to A),\end{aligned}$$

which are defined by path induction, and which satisfy

$$\begin{aligned}\mathsf{coe}_{\mathsf{idp}_A}(a) &= a,\\ \mathsf{coe}^{-1}_{\mathsf{idp}_A}(a) &= a.\end{aligned}$$

We do not take these equalities as definitional equalities because we never need it and we wish to limit the use of the definitional ($\mathsf{J}_{\mathsf{comp}}$). We can also easily check that those two functions are inverse to each other, again by path induction:

$$\begin{aligned}\mathsf{coeidp} &: (p : A =_{\mathsf{Type}} B)(a : A) \to \mathsf{coe}_p^{-1}(\mathsf{coe}_p(a)) =_A a,\\ \mathsf{coeidp}' &: (p : A =_{\mathsf{Type}} B)(b : B) \to \mathsf{coe}_p(\mathsf{coe}_p^{-1}(b)) =_B b.\end{aligned}$$

Another function related to coe is the transport function. Given a type A and a dependent type $P : A \to \mathsf{Type}$, we have the function

$$\begin{aligned}\mathsf{transport}^P &: (a =_A b) \to (P(a) \to P(b)),\\ \mathsf{transport}_p^P(x) &:= \mathsf{coe}_{\mathsf{ap}_P(p)}(x)\end{aligned}$$

defined using the fact that $\mathsf{ap}_P(p)$ is a path in the universe from $P(a)$ to $P(b)$. We can also transport in the other direction and it gives an equivalence between $P(a)$ and $P(b)$.

The *univalence axiom* states that conversely, any "equivalence" between two types can be turned into a path in the universe. We first define equivalences.

Proposition 1.5.1. *There is a predicate* $\mathsf{isequiv} : (A, B : \mathsf{Type})(f : A \to B) \to \mathsf{Type}$ *where* $\mathsf{isequiv}(f)$ *is read as "*f *is an equivalence", which has the following properties.*

- *A function* f *is an equivalence if and only if there is a function* $g : B \to A$ *such that* $g(f(x)) = x$ *for all* $x : A$ *and* $f(g(y)) = y$ *for all* $y : B$*.*
- *Given a function* $f : A \to B$*, any two elements of* $\mathsf{isequiv}(f)$ *are equal.*

Proof. The first point makes it seem like we could define $\mathsf{isequiv}$ by

$$\mathsf{isequiv}(f) :\overset{?}{=} \sum_{g:B\to A} (((x : A) \to g(f(x)) =_A x) \times ((y : B) \to f(g(y)) =_B y)),$$

but it turns out that this definition does not satisfy the second point of the proposition. One possibility to fix it is to consider separately one left-inverse and one right-inverse, instead of one two-sided inverse:

$$\begin{aligned}\mathsf{isequiv}(f) := &\left(\sum_{g:B\to A} ((x : A) \to g(f(x)) =_A x)\right)\\ &\times \left(\sum_{h:B\to A} ((y : B) \to f(h(y)) =_B y)\right).\end{aligned}$$

The two inverses turn out to be equal and that definition now satisfies the second point of the proposition. We refer to [UF13, section 4.3] for a proof that this definition of equivalences is suitable and to [UF13, chapter 4] for many other equivalent definitions of equivalences. □

We can now state the univalence axiom. We first define the type of equivalences between two types by

$$A \simeq B := \sum_{f:A\to B} \mathsf{isequiv}(f).$$

There is a map $(A =_{\mathsf{Type}} B) \to (A \simeq B)$ sending a path p to the equivalence given by $(\mathsf{coe}_p, \mathsf{coe}_p^{-1}, \mathsf{coeidp}_p, \mathsf{coeidp}'_p)$.

Axiom 1.5.2 (Univalence axiom)**.** *The map*

$$(A =_{\mathsf{Type}} B) \to (A \simeq B)$$

is an equivalence.

In particular, it means that there is a map

$$\mathsf{ua} : (A \simeq B) \to (A =_{\mathsf{Type}} B),$$

which allows us to construct an equality between two types given an equivalence between them, and if $e : A \simeq B$ is an equivalence then $\mathsf{coe}_{\mathsf{ua}(e)}$ is equal to the underlying function of e.

1.6 Dependent paths and squares

The notion of path we described in section 1.4 is *homogeneous* in that it can only be applied between two elements of the same type A. It does not make sense, at least in homotopy type theory, to talk about a path between a point $a : A$ and a point $b : B$ for A and B two unrelated types. However, if A and B are themselves connected by a path in Type, then we can make sense of it.

Proposition 1.6.1. *Given a path $p : A =_{\mathsf{Type}} B$ in* Type *between two types A and B and two terms $a : A$ and $b : B$, there is a type of* heterogeneous paths *between a and b over p written*

$$a =_p b$$

and satisfying the two equivalences

$$(a =_p b) \simeq (\mathsf{coe}_p(a) =_B b),$$
$$(a =_p b) \simeq (a =_A \mathsf{coe}_p^{-1}(b)).$$

Either $(\mathsf{coe}_p(a) =_B b)$ or $(a =_A \mathsf{coe}_p^{-1}(b))$ can be used as the definition of $a =_p b$. We could also define it by path induction on p, or as an inductive family of types. As a special case, if a and b have the same type A and p is the constant path, then the type $a =_{\mathsf{idp}_A} b$ is equivalent to the type $a =_A b$. Therefore homogeneous paths can be seen as a special case of heterogeneous paths via this equivalence.

Heterogeneous paths allow us to define dependent paths, which are useful in many different situations as we will see.

Proposition 1.6.2. *Given a dependent type $B : A \to \mathsf{Type}$ over a type A and a path $p : x =_A y$ in A, for every $u : B(x)$ and $v : B(y)$ there is a type of* dependent paths *in B over p from u to v written*

$$u =_p^B v$$

and satisfying the two equivalences

$$(u =_p^B v) \simeq (\mathsf{transport}^B(p, u) =_{B(y)} v),$$
$$(u =_p^B v) \simeq (u = \mathsf{transport}^B(p^{-1}, v)).$$

We can define the type of dependent paths by

$$(u =_p^B v) := (u =_{\mathsf{ap}_B(p)} v).$$

This makes sense because $\mathsf{ap}_B(p)$ is a path in the universe between $B(x)$ and $B(y)$. As with the type of heterogeneous paths, there are many other ways to define it.

Dependent paths are used in particular to define ap for dependent functions.

Definition 1.6.3. Given a dependent function $f : (x : A) \to B(x)$ and a path $p : a =_A b$ in A, there is a dependent path

$$\mathsf{ap}_f(p) : f(a) =_p^B f(b)$$

defined by path induction and satisfying

$$\mathsf{ap}_f(\mathsf{idp}_a) = \mathsf{idp}_{f(a)}.$$

Note that this last equation uses implicitly the equivalence between $u =_{\mathsf{idp}_a}^B v$ and $u =_{B(a)} v$.

If B does not depend on x, then the types $f(a) =_p^B f(b)$ and $f(a) =_B f(b)$ are equivalent and the element $\mathsf{ap}_f(p)$ defined here and the one defined in section 1.4.2 are identified by this equivalence. Therefore we keep the same notation for both and there will be no ambiguity.

In many cases, we can give a different characterization of $u =_p^B v$ depending on B. A very useful instance of that is when B is an identity type.

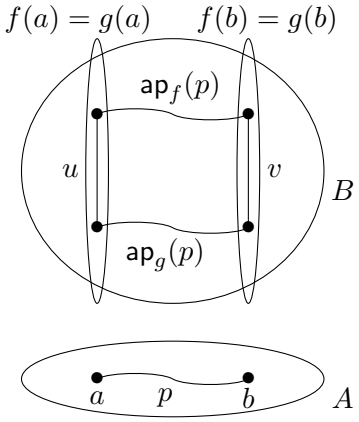
Proposition 1.6.4. *Given $A, B : \mathsf{Type}$, $f, g : A \to B$, $p : a =_A b$, $u : f(a) =_B g(a)$ and $v : f(b) =_B g(b)$, the type*

$$u =_p^{\lambda z. f(z) =_B g(z)} v$$

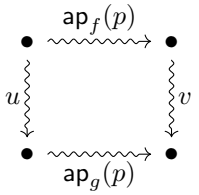
is equivalent to the type

$$(u \cdot \mathsf{ap}_g(p)) = (\mathsf{ap}_f(p) \cdot v).$$

We have the picture



The type $(u \cdot \mathsf{ap}_g(p)) = (\mathsf{ap}_f(p) \cdot v)$ can be seen as the type of *fillers* of the square

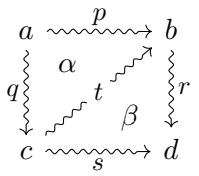


There are many other equivalent ways to define the type of fillers of such a square. We could use for instance any of the following four types (which are all equivalent):

$$\begin{aligned} u &= (\mathsf{ap}_f(p) \cdot v \cdot \mathsf{ap}_g(p)^{-1}), \\ v &= (\mathsf{ap}_f(p)^{-1} \cdot u \cdot \mathsf{ap}_g(p)), \\ \mathsf{ap}_f(p) &= (u \cdot \mathsf{ap}_g(p) \cdot v^{-1}), \\ \mathsf{ap}_g(p) &= (u^{-1} \cdot \mathsf{ap}_f(p) \cdot v). \end{aligned}$$

There is also a direct inductive definition, similar to the definition of the identity type, which states that in order to construct a section of a dependent type over all squares whose upper-left corner is a fixed point $a : A$, it is enough to define it over the identity square ids_a (which has idp_a on all four sides).

We can generalize the notion of coherence operation to include the case of squares or other geometrical shapes. For instance in the diagram



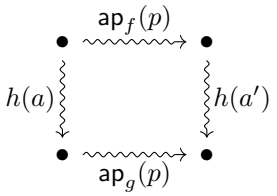
one can compose the two triangles α and β in order to obtain a filler of the square. The coherence operation is the one given by

$$(a, c : A)(q : a = c)(b : A)(p : a = b)(t : c = b)(\alpha : q^{-1} \cdot p = t) \tag{1.6.5}$$

$$(d : A)(s : c = d)(r : b = d)(\beta : t^{-1} \cdot s = r) \mapsto \alpha \boxslash \beta : (p \cdot r) = (q \cdot s). \tag{1.6.6}$$

A type of squares which is often used is the naturality squares of homotopies.

Definition 1.6.7. Given two functions $f, g : A \to B$, a pointwise equality $h : (x : A) \to f(x) =_B g(x)$ between f and g (also called a *homotopy*) and a path $p : a =_A a'$ in A, the *naturality square* of h on p is the filler of the square



obtained by applying proposition 1.6.4 to $\mathsf{ap}_h(p)$. Note that h is a dependent function, so this ap is the dependent one defined in definition 1.6.3.

The next proposition shows that paths in Σ-types can be seen as pairs of paths. There is an operation of pairing of paths and one can take the first and second component of a path in a pair type. Note also that the second path is a dependent path over the first one.

Proposition 1.6.8. *Given a type A, a dependent type $B : A \to \mathsf{Type}$ over A and two elements (a, b) and (a', b') of $\sum_{x:A} B(x)$, there is an equivalence of types*

$$\left((a, b) =_{\sum_{x:A} B(x)} (a', b')\right) \simeq \left(\sum_{p:a=_A a'} b =_p^B b'\right).$$

Proof. The map from the left-hand side to the right-hand side is given by the two maps

$$\begin{aligned} \mathsf{fst}^= &: (r : (a, b) = (a', b')) \to a =_A a', \\ \mathsf{fst}^=(r) &:= \mathsf{ap}_{\mathsf{fst}}(r), \end{aligned}$$

$$\begin{aligned} \mathsf{snd}^= &: (r : (a, b) = (a', b')) \to b =^B_{\mathsf{fst}^=(r)} b', \\ \mathsf{snd}^=(r) &:= \mathsf{ap}_{\mathsf{snd}}(r) \end{aligned}$$

and the map from the right-hand side to the left-hand side is defined by path induction twice. The fact that these two maps are inverse to each other is immediate by path induction. □

For function types we have that paths between functions correspond to homotopies (pointwise equalities).

Proposition 1.6.9 (Function extensionality)**.** *Given a type A, a dependent type $B : A \to \mathsf{Type}$ and two functions $f, g : (x : A) \to B(x)$, we have an equivalence*

$$(f =_{(x:A)\to B(x)} g) \simeq ((x : A) \to f(x) =_{B(x)} g(x)).$$

Proof. The map from the left-hand side to the right-hand side is defined by

$$\begin{aligned}\mathsf{happly} &: (f =_{(x:A)\to B(x)} g) \to ((x : A) \to f(x) =_{B(x)} g(x)),\\ \mathsf{happly}(p)(x) &:= \mathsf{ap}_{\lambda h.h(x)}(p).\end{aligned}$$

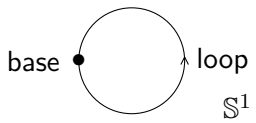
The map in the other direction, which turns a homotopy between f and g into a path between f and g, cannot be defined by path induction as in the previous proposition. But it turns out that we can define it using the univalence axiom, and we can also prove that it is an inverse to happly, see [UF13, section 4.9]. □

1.7 Higher inductive types

In ordinary inductive types, the constructors only generate elements of the type T we are defining. But in homotopy type theory, we are seeing paths and higher paths in T as being somehow still part of T. This leads to the notion of *higher inductive types*, which are similar to inductive types with the difference that there might be *path-constructors* which give new paths or new equalities in the type being defined. Here are some of the most important examples of higher inductive types.

Circle The *circle* $\mathbb{S}^1$ is the simplest non-trivial higher inductive type. It is generated by the two constructors

$$\begin{aligned}&\mathsf{base} : \mathbb{S}^1,\\ &\mathsf{loop} : \mathsf{base} =_{\mathbb{S}^1} \mathsf{base}.\end{aligned}$$



Just as with inductive types, there is an induction principle which states that given a dependent type $P : \mathbb{S}^1 \to \mathsf{Type}$, a function $f : (x : \mathbb{S}^1) \to P(x)$ can be defined by

$$\begin{aligned}f &: (x : \mathbb{S}^1) \to P(x),\\ f(\mathsf{base}) &:= f_{\mathsf{base}},\\ \mathsf{ap}_f(\mathsf{loop}) &:= f_{\mathsf{loop}},\end{aligned}$$

where the terms f_{base} and f_{loop} satisfy

$$\begin{aligned}&f_{\mathsf{base}} : P(\mathsf{base}),\\ &f_{\mathsf{loop}} : f_{\mathsf{base}} =^{P}_{\mathsf{loop}} f_{\mathsf{base}}.\end{aligned}$$

Note that we need to give the value of $\mathsf{ap}_f(\mathsf{loop})$ and not of $f(\mathsf{loop})$ (which would not make sense given that loop is not an element of $\mathbb{S}^1$) and that f_{loop} is a dependent path in P over loop.

There is a subtlety here in that, in the standard presentation of homotopy type theory, the second equation $\mathsf{ap}_f(\mathsf{loop}) := f_{\mathsf{loop}}$ is *not* taken as a definitional equality but only as an equality in the sense of section 1.4. There are two main reasons for that. The

first one is that most of the known models of homotopy type theory do not model it as a definitional equality, and the second one is that most proof assistants do not allow it to be a definitional equality either. These difficulties are being resolved. For instance the cubical model described in [BCH14] models them as definitional equalities and one can now add custom new definitional equalities in Agda as well (see [Agd]), but this is all very much work in progress. There are a few places (for instance in the 3×3-lemma in section 1.8) where it would be very useful to have it as a definitional equality, together with various other equalities, but for proofs written on paper (as opposed to proofs written in a proof assistant) one can usually gloss over these kinds of issues. We use the same symbol $:=$ for aesthetic reasons.

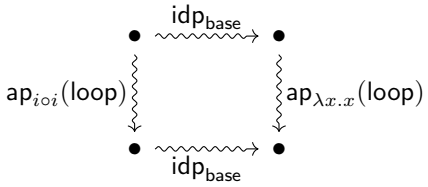
Here are two examples of usage of the induction principle. We first define a function $i : \mathbb{S}^1 \to \mathbb{S}^1$ by

$$\begin{aligned} i &: \mathbb{S}^1 \to \mathbb{S}^1, \\ i(\mathsf{base}) &:= \mathsf{base}, \\ \mathsf{ap}_i(\mathsf{loop}) &:= \mathsf{loop}^{-1}. \end{aligned}$$

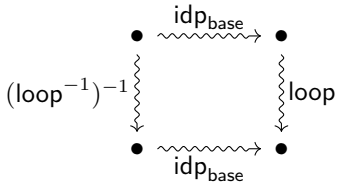
Note that i is a non-dependent function, therefore in the last line we just need to give a path in $\mathbb{S}^1$ from $i(\mathsf{base})$ to itself, and we choose loop^{-1}. We now prove that i is involutive by

$$\begin{aligned} \mathsf{invol} &: (x : \mathbb{S}^1) \to i(i(x)) =_{\mathbb{S}^1} x, \\ \mathsf{invol}(\mathsf{base}) &:= \mathsf{idp}_{\mathsf{base}}, \\ \mathsf{ap}_{\mathsf{invol}}(\mathsf{loop}) &:= \mathsf{invol}_{\mathsf{loop}}, \end{aligned}$$

with $\mathsf{invol}_{\mathsf{loop}}$ to be defined. This time, the dependent type is $P(x) := (i(i(x)) =_{\mathbb{S}^1} x)$. In the case for base we need to prove that $i(i(\mathsf{base})) = \mathsf{base}$, but using the fact that $i(\mathsf{base})$ is equal to base by definition we have $\mathsf{idp}_{\mathsf{base}} : i(i(\mathsf{base})) = \mathsf{base}$. In the case for loop, we need to give a dependent path $\mathsf{invol}_{\mathsf{loop}}$ in P over loop from $\mathsf{idp}_{\mathsf{base}}$ to $\mathsf{idp}_{\mathsf{base}}$. Using proposition 1.6.4 we see that $\mathsf{invol}_{\mathsf{loop}}$ has to be a filler of the square



Using now the fact that $\mathsf{ap}_{\lambda x.x}(p)$ is equal to p, that $\mathsf{ap}_{i \circ i}(p)$ is equal to $\mathsf{ap}_i(\mathsf{ap}_i(p))$ and that $\mathsf{ap}_i(\mathsf{loop})$ is equal to loop^{-1}, we have to fill the square



and this follows from a coherence operation.

Pushouts Let's consider three types A, B, C and two functions $f : C \to A$, $g : C \to B$,

$$A \xleftarrow{f} C \xrightarrow{g} B.$$

Such a diagram is called a *span*. The *pushout* of this span is the higher inductive type $A \sqcup^C B$ generated by the constructors

$$\begin{aligned} \mathsf{inl} &: A \to A \sqcup^C B, \\ \mathsf{inr} &: B \to A \sqcup^C B, \\ \mathsf{push} &: (c : C) \to \mathsf{inl}(f(c)) =_{A \sqcup^C B} \mathsf{inr}(g(c)). \end{aligned}$$

In particular, we have the commutative square

$$\begin{array}{ccc} C & \xrightarrow{g} & B \\ {\scriptstyle f}\downarrow & \mathsf{push} & \downarrow{\scriptstyle \mathsf{inr}} \\ A & \xrightarrow[\mathsf{inl}]{} & A \sqcup^C B \end{array}$$

The idea is that we start with the disjoint sum $A + B$ and for every element c of C, we add a new path from $\mathsf{inl}(f(c))$ to $\mathsf{inr}(g(c))$. What we obtain would rather be called a "homotopy pushout" in classical homotopy theory, but in homotopy type theory this is the only sort of pushout which is possible to define therefore we call them simply "pushouts".

The induction principle states that, given a dependent type $P : A \sqcup^C B \to \mathsf{Type}$, we can define a function $h : (x : A \sqcup^C B) \to P(x)$ by

$$\begin{aligned} h &: (x : A \sqcup^C B) \to P(x), \\ h(\mathsf{inl}(a)) &:= h_{\mathsf{inl}}(a), \\ h(\mathsf{inr}(b)) &:= h_{\mathsf{inr}}(b), \\ \mathsf{ap}_h(\mathsf{push}(c)) &:= h_{\mathsf{push}}(c), \end{aligned}$$

where we have

$$\begin{aligned} h_{\mathsf{inl}} &: (a : A) \to P(\mathsf{inl}(a)), \\ h_{\mathsf{inr}} &: (b : B) \to P(\mathsf{inr}(b)), \\ h_{\mathsf{push}} &: (c : C) \to h_{\mathsf{inl}}(f(c)) =^P_{\mathsf{push}(c)} h_{\mathsf{inr}}(g(c)). \end{aligned}$$

This induction principle looks similar to the usual universal property of pushouts, but there are two differences. On the one hand this is only an existence result, we say nothing about uniqueness a priori. On the other hand we state it for every dependent type whereas the universal property talks only about non-dependent types. It turns out that one can prove the universal property from the induction principle using the fact that uniqueness-up-to-homotopy can be seen as existence of a particular equality which is constructed using the induction principle (see [UF13, section 6.8] and [Soj15]). However

the universal property only implies a weak version of the induction principle, where the equalities of the form $h(\mathsf{inl}(a)) = h_{\mathsf{inl}}(a)$ are only propositional equalities. Moreover, the statement of the induction principle is very natural from the point of view of type theory, while the statement of the universal property looks more contrived.

Many interesting constructions are defined as pushouts, in particular we have the following definitions.

- The *suspension* ΣA of a type A is the pushout of the span

$$\mathbf{1} \longleftarrow A \longrightarrow \mathbf{1}.$$

  We write north, south and $\mathsf{merid}(a)$ for the terms $\mathsf{inl}(\star_{\mathbf{1}})$, $\mathsf{inr}(\star_{\mathbf{1}})$ and $\mathsf{push}(a)$. If A is a pointed type (i.e. a type A equipped with a basepoint $\star_A$), we write ΩA for the type $\star_A = \star_A$ and we define the map

$$\begin{aligned} \varphi_A &: A \to \Omega\Sigma A, \\ \varphi_A(a) &:= \mathsf{merid}(a) \cdot \mathsf{merid}(\star_A)^{-1}, \end{aligned}$$

  (where ΣA is pointed by north) which satisfies

$$\varphi_A(\star_A) = \mathsf{idp}_{\star_A}.$$

- The spheres $\mathbb{S}^n$ for $n : \mathbb{N}$ are defined by induction on n. For $n = 0$ we define $\mathbb{S}^0 := \mathbf{2}$ and for $n + 1$ we define

$$\mathbb{S}^{n+1} := \Sigma\mathbb{S}^n.$$

  Note that $\Sigma\mathbf{2}$ is equivalent to the type $\mathbb{S}^1$ defined above, so we could alternatively define the spheres by iterated suspensions of $\mathbb{S}^1$. In practice, we often use the direct inductive definition of $\mathbb{S}^1$ using base and loop instead of $\Sigma\mathbf{2}$.

- The *join* $A * B$ of two types A and B is the pushout of the span

$$A \xleftarrow{\mathsf{fst}} A \times B \xrightarrow{\mathsf{snd}} B.$$

- The *wedge sum* $A \vee B$ of two pointed types A and B is the pushout of the span

$$A \longleftarrow \mathbf{1} \longrightarrow B,$$

  where the two maps pick the basepoints of A and B

- The *smash product* $A \wedge B$ of two pointed types A and B is the pushout of the span

$$\mathbf{1} \longleftarrow A \vee B \xrightarrow{i^{\vee}_{A,B}} A \times B,$$

where the map on the right is defined by

$$\begin{aligned} i^{\vee}_{A,B} &: A \vee B \to A \times B, \\ i^{\vee}_{A,B}(\mathsf{inl}(a)) &:= (a, \star_B), \\ i^{\vee}_{A,B}(\mathsf{inr}(b)) &:= (\star_A, b), \\ \mathsf{ap}_{i^{\vee}_{A,B}}(\mathsf{push}(\star_{\mathbf{1}})) &:= \mathsf{idp}_{(\star_A,\star_B)}. \end{aligned}$$

The smash product is studied in more detail in chapter 4.

1.8 The 3×3-lemma

The 3×3-lemma is a technical lemma that we will use a few times to construct equivalences between various nested pushouts. Its usefulness has been suggested to me by Eric Finster.

Let's consider the diagram

$$\begin{array}{ccccc} A_{00} & \xleftarrow{f_{01}} & A_{02} & \xrightarrow{f_{03}} & A_{04} \\ {\scriptstyle f_{10}}\uparrow & \swarrow^{H_{11}} & {\scriptstyle f_{12}}\uparrow & \searrow^{H_{13}} & \uparrow{\scriptstyle f_{14}} \\ A_{20} & \xleftarrow[f_{21}]{} & A_{22} & \xrightarrow{f_{23}} & A_{24} \\ {\scriptstyle f_{30}}\downarrow & \swarrow^{H_{31}} & \downarrow{\scriptstyle f_{32}} & \searrow^{H_{33}} & \downarrow{\scriptstyle f_{34}} \\ A_{40} & \xleftarrow[f_{41}]{} & A_{42} & \xrightarrow[f_{43}]{} & A_{44} \end{array} \tag{1.8.1}$$

where the A_{ij} are types, the f_{ij} are maps and the H_{ij} are *fillers* of the squares, i.e. homotopies between compositions of maps (for instance H_{11} has type $(x : A_{22}) \to f_{01}(f_{12}(x)) = f_{10}(f_{21}(x))$). The pushouts of the three columns fit in the diagram

$$A_{\bullet 0} \xleftarrow{f_{\bullet 1}} A_{\bullet 2} \xrightarrow{f_{\bullet 3}} A_{\bullet 4}, \tag{1.8.2}$$

where the map $f_{\bullet 1}$ is defined by

$$\begin{aligned} f_{\bullet 1} &: A_{\bullet 2} \to A_{\bullet 0}, \\ f_{\bullet 1}(\mathsf{inl}(x_{02})) &:= \mathsf{inl}(f_{01}(x_{02})), \\ f_{\bullet 1}(\mathsf{inr}(x_{42})) &:= \mathsf{inr}(f_{41}(x_{42})), \\ \mathsf{ap}_{f_{\bullet 1}}(\mathsf{push}(x_{22})) &:= \mathsf{ap}_{\mathsf{inl}}(H_{11}(x_{22})) \cdot \mathsf{push}(f_{21}(x_{22})) \cdot \mathsf{ap}_{\mathsf{inr}}(H_{31}(x_{22}))^{-1} \end{aligned}$$

and the map $f_{\bullet 3}$ is defined in a similar way. We denote by $A_{\bullet\square}$ the pushout of diagram 1.8.2. We can also first consider the pushout of all three rows of diagram 1.8.1 and then the pushout $A_{\square\bullet}$ of the resulting column.

Lemma 1.8.3 (3×3-lemma)**.** *There is an equivalence*

$$A_{\bullet\square} \simeq A_{\square\bullet}.$$

Sketch of proof. The map $f : A_{\bullet\square} \to A_{\square\bullet}$ basically takes an element of $A_{\bullet\square}$ and swaps the two constructors. For instance, it sends $\mathsf{inl}(\mathsf{inr}(x_{40}))$ to $\mathsf{inr}(\mathsf{inl}(x_{40}))$, and so on. It is defined by

$$\begin{aligned} f &: A_{\bullet\square} \to A_{\square\bullet}, \\ f(\mathsf{inl}(x_0)) &:= f_{\mathsf{inl}}(x_0), \\ f(\mathsf{inr}(x_4)) &:= f_{\mathsf{inr}}(x_4), \\ \mathsf{ap}_f(\mathsf{push}(x_2)) &:= f_{\mathsf{push}}(x_2), \end{aligned}$$

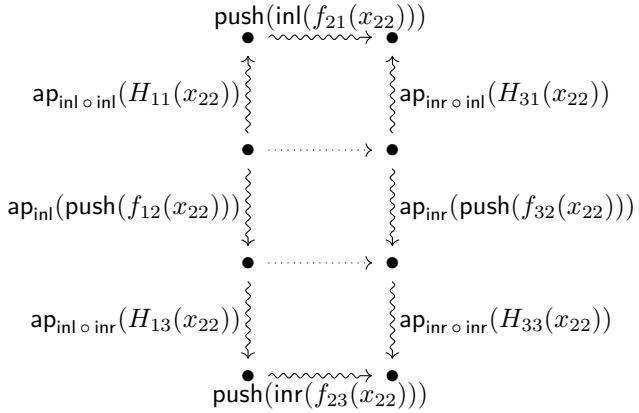
where

$$\begin{aligned} f_{\mathsf{inl}} &: A_{\bullet 0} \to A_{\square\bullet}, \\ f_{\mathsf{inl}}(\mathsf{inl}(x_{00})) &:= \mathsf{inl}(\mathsf{inl}(x_{00})), \\ f_{\mathsf{inl}}(\mathsf{inr}(x_{40})) &:= \mathsf{inr}(\mathsf{inl}(x_{40})), \\ \mathsf{ap}_{f_{\mathsf{inl}}}(\mathsf{push}(x_{20})) &:= \mathsf{push}(\mathsf{inl}(x_{20})), \end{aligned} \qquad \begin{aligned} f_{\mathsf{inr}} &: A_{\bullet 4} \to A_{\square\bullet}, \\ f_{\mathsf{inr}}(\mathsf{inl}(x_{04})) &:= \mathsf{inl}(\mathsf{inr}(x_{04})), \\ f_{\mathsf{inr}}(\mathsf{inr}(x_{44})) &:= \mathsf{inr}(\mathsf{inr}(x_{44})), \\ \mathsf{ap}_{f_{\mathsf{inr}}}(\mathsf{push}(x_{24})) &:= \mathsf{push}(\mathsf{inr}(x_{24})), \end{aligned}$$

and

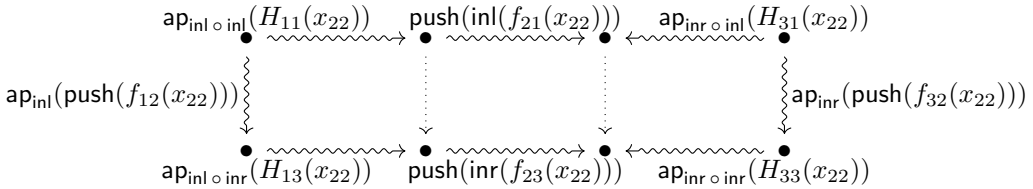
$$\begin{aligned} f_{\mathsf{push}} &: (x_2 : A_{\bullet 2}) \to f_{\mathsf{inl}}(f_{\bullet 1}(x_2)) = f_{\mathsf{inr}}(f_{\bullet 3}(x_2)), \\ f_{\mathsf{push}}(\mathsf{inl}(x_{02})) &:= \mathsf{ap}_{\mathsf{inl}}(\mathsf{push}(x_{02})), \\ f_{\mathsf{push}}(\mathsf{inr}(x_{42})) &:= \mathsf{ap}_{\mathsf{inr}}(\mathsf{push}(x_{42})), \\ \mathsf{ap}_{f_{\mathsf{push}}}(\mathsf{push}(x_{22})) &:= f_{\mathsf{push,push}}. \end{aligned}$$

The last term $f_{\mathsf{push,push}}$ is trickier. We want something filling the middle square of the diagram



where the dotted paths are the composites of the upper and lower squares. What we

have is $\mathsf{ap}_{\mathsf{push}}(\mathsf{push}(x_{22}))$ which fills the middle square of the diagram



where the dotted paths are the composites of the left and right square. We notice that the eight paths around are the same in both diagrams, therefore there is a coherence operation going from one middle square to the other. The inverse map g is then defined in a similar way.

In order to prove that they are inverse to each other, we use again the induction principle twice in order to construct two functions

$$(x : A_{\bullet\square}) \to g(f(x)) = x,$$
$$(y : A_{\square\bullet}) \to f(g(y)) = y.$$

For elements of the form $\mathsf{inl}(\mathsf{inl}(x_{00}))$, $\mathsf{inl}(\mathsf{inr}(x_{40}))$, $\mathsf{inr}(\mathsf{inl}(x_{04}))$ and $\mathsf{inr}(\mathsf{inr}(x_{44}))$ the equality is true by definition, so we use idp. For paths of the form $\mathsf{push}(\mathsf{inl}(x_{02}))$, $\mathsf{push}(\mathsf{inr}(x_{42}))$, $\mathsf{ap}_{\mathsf{inl}}(\mathsf{push}(x_{20}))$ and $\mathsf{ap}_{\mathsf{inr}}(\mathsf{push}(x_{24}))$ it is not true definitionally but not difficult to prove. Finally, for squares of the form $\mathsf{ap}_{\mathsf{push}}(\mathsf{push}(x_{22}))$ it is more tricky because we have to prove an equality along the equalities proved in the previous step, but it can be done. A proof checked in Agda is available at [Bru15]. □

Here are two propositions which are very useful when using the 3×3-lemma.

Proposition 1.8.4. *Given a map $f : A \to B$, the pushout of the diagram*

$$A \xleftarrow{\mathsf{id}_A} A \xrightarrow{f} B$$

is equivalent to B.

Proof. We define $g : B \to A \sqcup^A B$ by $g(b) :\equiv \mathsf{inr}(b)$ and $h : A \sqcup^A B \to B$ by

$$h : A \sqcup^A B \to B,$$
$$h(\mathsf{inl}(a)) :\equiv f(a),$$
$$h(\mathsf{inr}(b)) :\equiv b,$$
$$\mathsf{ap}_h(\mathsf{push}(a)) :\equiv \mathsf{idp}_{f(a)}.$$

It is straightforward to show that these functions are inverse to each other. □

Proposition 1.8.5. *Given two maps $f : C \to A$ and $g : C \to B$ and a type X, the pushout of the diagram*

$$A \times X \xleftarrow{(c,x)\mapsto(f(c),x)} C \times X \xrightarrow{(c,x)\mapsto(g(c),x)} B \times X$$

is equivalent to $(A \sqcup^C B) \times X$.

Proof. We define the two maps

$$\begin{aligned} h_1 : (A \sqcup^C B) \times X &\to (A \times X) \sqcup^{C\times X} (B \times X), \\ h_1(\mathsf{inl}(a), x) &:= \mathsf{inl}(a, x), \\ h_1(\mathsf{inr}(b), x) &:= \mathsf{inr}(b, x), \\ \mathsf{ap}_{h_1(-,x)}(\mathsf{push}(c)) &:= \mathsf{push}(c, x), \end{aligned}$$

$$\begin{aligned} h_2 : (A \times X) \sqcup^{C\times X} (B \times X) &\to (A \sqcup^C B) \times X, \\ h_2(\mathsf{inl}(a, x)) &:= (\mathsf{inl}(a), x), \\ h_2(\mathsf{inr}(b, x)) &:= (\mathsf{inr}(b), x), \\ \mathsf{ap}_{h_2}(\mathsf{push}(c, x)) &:= \mathsf{ap}_{(-,x)}(\mathsf{push}(c)). \end{aligned}$$

It is again easy to prove that those two functions are inverse to each other. □

We now present a proposition that we can prove using the 3 × 3-lemma and which will be used in the next chapter. It has also been formally proved in Agda by Evan Cavallo in [Cav14], directly, without using the 3 × 3-lemma.

Proposition 1.8.6. *Given three types A, B and C, there is an equivalence*

$$(A * B) * C \simeq A * (B * C).$$

Proof. Let's consider the diagram

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

where the arrows are the obvious projections and the four squares are filled by idp. The pushout of the first row is $A * B$, the pushout of the second row is $(A * B) \times C$ according to proposition 1.8.5 and the pushout of the third row is C according to proposition 1.8.4. Moreover, the two maps $(A * B) \times C \to A * B$ and $(A * B) \times C \to C$ are equal to the two projections, as can be checked by hand by induction. Therefore the pushout of the pushouts of the rows is equivalent to $(A * B) * C$.

Similarly, the pushout of the first column is A, the pushout of the second column is $A \times (B * C)$, the pushout of the third column is $B * C$, and the two maps are the two projections, hence the pushout of the pushouts of the columns is equivalent to $A*(B*C)$. Therefore, we have

$$(A * B) * C \simeq A * (B * C).$$ □

Here is another simple application, which could also be proved directly.

Proposition 1.8.7. *For any type A there is an equivalence $\mathbf{2} * A \simeq \Sigma A$.*

Proof. We apply the 3×3-lemma to the diagram

$$\begin{array}{ccccc} \mathbf{1} & \longleftarrow & A & \longrightarrow & A \\ \uparrow & & \uparrow & & \uparrow \\ \mathbf{0} & \longleftarrow & \mathbf{0} & \longrightarrow & A \\ \downarrow & & \downarrow & & \downarrow \\ \mathbf{1} & \longleftarrow & A & \longrightarrow & A \end{array}$$

and we conclude by noticing that $\mathbf{1} + \mathbf{1} \simeq \mathbf{2}$ and that $A + A \simeq \mathbf{2} \times A$. □

Combining the two previous results, we obtain

Proposition 1.8.8. *For every $m, n : \mathbb{N}$, there is an equivalence*

$$\mathbb{S}^m * \mathbb{S}^n \simeq \mathbb{S}^{m+n+1}.$$

Proof. We proceed by induction on m. For $m = 0$ it is proposition 1.8.7, and for $m + 1$ we have

$$\begin{aligned} \mathbb{S}^{m+1} * \mathbb{S}^n &\simeq (\Sigma \mathbb{S}^m) * \mathbb{S}^n \\ &\simeq (\mathbf{2} * \mathbb{S}^m) * (\mathbb{S}^n) \quad \text{by proposition 1.8.7} \\ &\simeq \mathbf{2} * (\mathbb{S}^m * \mathbb{S}^n) \quad \text{by proposition 1.8.6} \\ &\simeq \mathbf{2} * \mathbb{S}^{m+n+1} \quad \text{by induction hypothesis} \\ &\simeq \Sigma \mathbb{S}^{m+n+1} \quad \text{by proposition 1.8.7} \\ &\simeq \mathbb{S}^{(m+1)+n+1}. \end{aligned}$$

□

Another simple application is the following, which we will use in chapter 3.

Proposition 1.8.9. *Given four types A, B, C and D and three functions $f : B \to A$, $g : C \to B$ and $h : C \to D$, there is an equivalence*

$$A \sqcup^B (B \sqcup^C D) \simeq A \sqcup^C D$$

where the map $B \to B \sqcup^C D$ on the left is inl *and the map $C \to A$ on the right is $f \circ g$.*

Proof. We apply the 3×3-lemma to the diagram

$$\begin{array}{ccccc} A & \longleftarrow & \mathbf{0} & \longrightarrow & \mathbf{0} \\ {\scriptstyle f}\uparrow & & \uparrow & & \uparrow \\ B & \longleftarrow & \mathbf{0} & \longrightarrow & \mathbf{0} \\ {\scriptstyle \mathsf{id}_B}\downarrow & & \downarrow & & \downarrow \\ B & \underset{g}{\longleftarrow} & C & \underset{h}{\longrightarrow} & D \end{array}$$

□

All the examples of use of the 3×3-lemma presented so far are simpler to prove than the full version either because it is degenerate or because $H_{11}(x)$, $H_{13}(x)$, $H_{31}(x)$ and $H_{33}(x)$ are all equal to idp. However in proposition 3.3.2 we will use one instance of the 3×3-lemma where $H_{11}(x)$ is not a constant path.

1.9 The flattening lemma

In classical homotopy theory, the usual way to construct a fibration is to define a continuous map $f : E \to B$ between two spaces E and B and prove that it has the property of being a fibration. In homotopy type theory the story is quite different because there is no predicate "being a fibration". Indeed, being a fibration is not a property invariant under homotopy of maps. Instead, fibrations are replaced by dependent types, which encode directly the fact that every point of the base space has a fiber over it. One way to understand why we can see dependent types as fibrations is with the function $\mathsf{transport}$ of section 1.5 which shows that one can transport elements between fibers of a dependent type along a path in the base type, and we could similarly lift any homotopy in the base space. Constructing a fibration in homotopy type theory consists simply in defining a map $P : B \to \mathsf{Type}$. There is no need to "prove" that it is a fibration, but what is non-trivial is to understand its total space. In general, the total space of a fibration $P : B \to \mathsf{Type}$ is the type

$$\sum_{x:B} P(x),$$

but this is not a description which is easy to work with in general.

A special case of fibrations in homotopy type theory are fibrations over higher inductive types which are defined by induction using the univalence axiom. For instance, one can define a fibration P over $\mathbb{S}^1$ by

$$\begin{aligned} P &: \mathbb{S}^1 \to \mathsf{Type}, \\ P(\mathsf{base}) &:= P_{\mathsf{base}}, \\ \mathsf{ap}_P(\mathsf{loop}) &:= \mathsf{ua}(P_{\mathsf{loop}}), \end{aligned}$$

where P_{base} is a type and P_{loop} is an equivalence between P_{base} and itself. Similarly, one can define a fibration over a pushout $A \sqcup^C B$ by

$$\begin{aligned} P &: A \sqcup^C B \to \mathsf{Type}, \\ P(\mathsf{inl}(a)) &:= P_{\mathsf{inl}}(a), \\ P(\mathsf{inr}(b)) &:= P_{\mathsf{inr}}(b), \\ \mathsf{ap}_P(\mathsf{push}(c)) &:= \mathsf{ua}(P_{\mathsf{push}}(c)), \end{aligned}$$

where

$$\begin{aligned} P_{\mathsf{inl}} &: A \to \mathsf{Type}, \\ P_{\mathsf{inr}} &: B \to \mathsf{Type}, \\ P_{\mathsf{push}} &: (c : C) \to P_{\mathsf{inl}}(f(c)) \simeq P_{\mathsf{inr}}(g(c)). \end{aligned}$$

The *flattening lemma* gives us a nice description of the total space of such fibrations. We refer to [UF13, section 6.12] for the general statement and proof, and we state here only these two special cases.

Proposition 1.9.1. *The total space of a fibration $P : \mathbb{S}^1 \to \mathsf{Type}$ defined by P_{base} and P_{loop} as above is equivalent to the higher inductive type T generated by*

$$\widetilde{\mathsf{base}} : P_{\mathsf{base}} \to T,$$
$$\widetilde{\mathsf{loop}} : (x : P_{\mathsf{base}}) \to \widetilde{\mathsf{base}}(x) =_T \widetilde{\mathsf{base}}(P_{\mathsf{loop}}(x)).$$

Proposition 1.9.2. *Given the pushout D of the diagram*

$$A \xleftarrow{f} C \xrightarrow{g} B$$

and $P : D \to \mathsf{Type}$ defined by P_{inl}, P_{inr} and P_{push} as above, the total space of P is equivalent to the pushout of the diagram

$$\sum_{x:A} P_{\mathsf{inl}}(x) \longleftarrow \sum_{z:C} P_{\mathsf{inl}}(f(z)) \longrightarrow \sum_{y:B} P_{\mathsf{inr}}(y),$$

where the left map sends the pair (z, u) to $(f(z), u)$ and the right map sends the pair (z, v) to $(g(z), P_{\mathsf{push}}(z)(v))$.

1.10 Truncatedness and truncations

Finally we introduce n-truncated types and truncations, which are an essential part of this work and of homotopy type theory in general. The notion of n-truncated type was introduced in 2009 by Vladimir Voevodsky under the name "type of h-level $n + 2$" and proved invaluable in the understanding of homotopy type theory. Intuitively, a type is n-truncated if it doesn't contain any interesting information in its k-iterated identity types for $k > n$. From the point of view of homotopy theory they correspond to homotopy n-types. For instance, 0-truncated types corresponds to *sets* (i.e. discrete spaces). We can also make sense of (-1)-truncated types which are types whose elements are all equal, and of (-2)-truncated types which are the contractible types.

1.10.1 Truncatedness of types

Definition 1.10.1. A type A is *contractible* or (-2)*-truncated* if there is a point $a : A$ (sometimes called the *center of contraction*) and an equality from a to x for every $x : A$. In other words, being contractible is defined by the predicate

$$\mathsf{is\text{-}contr}(A) := \sum_{a:A} ((x : A) \to a =_A x).$$

Note that if we read the definition of contractibility as "there exists $a : A$ such that for every $x : A$ there exists a path from a to x", we might think that it is only a definition of connectedness. The error is that the existential quantifier is a *explicit* existential quantifier, which means that the path from a to x should not only exist but it should depend continuously on x, which is much stronger than connectedness.

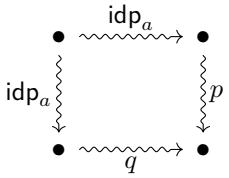
An important example of contractible type is the space of all paths in a type with one endpoint fixed.

Proposition 1.10.2. *Given a type A and a point $a : A$, the type*

$$\sum_{x:A}(a =_A x)$$

is contractible.

Proof. We first notice that the pair (a, idp_a) is an element of this type. We take it as the center of contraction and we now have to prove that for every $x : A$ and $p : a =_A x$, there is a path $(a, \mathsf{idp}_a) = (x, p)$ in $\sum_{x:A}(a =_A x)$. Using propositions 1.6.8 and 1.6.4, we see that we need to find a path $q : a =_A x$ together with a filler of the square



We take $q := p$ and the filler coming from the equality $\mathsf{idp}_{\mathsf{idp}_a \cdot p} : \mathsf{idp}_a \cdot p = \mathsf{idp}_a \cdot q$. □

Definition 1.10.3. For $n \geq -2$, a type A is $(n+1)$-*truncated* if for every $x, y : A$, the type $x =_A y$ is n-truncated.

As a special case, a type which is (-1)-truncated is called a *mere proposition* and a type which is 0-truncated is called a *set*.

Proposition 1.10.4. *We have the following properties.*

- *Being n-truncated is a mere proposition.*
- *A type A satisfies $x =_A y$ for every $x, y : A$ if and only if A is a mere proposition.*
- *If $B : A \to \mathsf{Type}$ is a family of n-truncated types, then $(x : A) \to B(x)$ is itself n-truncated.*
- *If A is n-truncated and $B : A \to \mathsf{Type}$ is a family of n-truncated types, then $\sum_{x:A} B(x)$ is itself n-truncated.*
- *If A is n-truncated and $m \geq n$, then A is m-truncated as well.*
- *If A is n-truncated and $x, y : A$, then $x =_A y$ is n-truncated.*

We refer to [UF13, section 7.1] for a proof of these properties.

Proposition 1.10.5. *The unit type is contractible and the empty type is a proposition.*

Proof. For the unit type, the center of contraction is $\star_{\mathbf{1}}$ and for every $x : \mathbf{1}$ we have $x = \star_{\mathbf{1}}$ because we only need to check it for $x := \star_{\mathbf{1}}$ and in that case we use $\mathsf{idp}_{\star_{\mathbf{1}}}$. Hence, the type $\mathbf{1}$ is contractible.

For the empty type, we have to prove that given $x, y : \bot$ we have $x =_\bot y$, but the induction principle of $\bot$ says that given $x : \bot$ everything is true. Hence, the type $\bot$ is a proposition. □

A large class of sets (0-truncated types) is given by types with decidable equality.

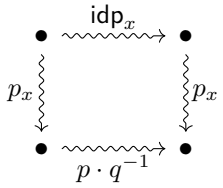
Proposition 1.10.6 (Hedberg's theorem)**.** *If a type A is such that for every $x, y : A$, either $x =_A y$ or $\neg(x =_A y)$ (explicit disjunction), then A is a set.*

We say that a type A *has decidable equality* if it satisfies the hypothesis of the proposition.

Proof. Given $x, y : A$ and $p, q : x =_A y$, we want to prove that $p =_{x=_A y} q$. Let's call d the witness of decidable equality on A. We have

$$d : (x, y : A) \to ((x =_A y) + \neg(x =_A y)).$$

Decidable equality at (x, x) gives us either a loop at x or an element of $\neg(x =_A x)$, but this last case cannot hold because there exists at least one loop at x, namely idp_x. Therefore, for every $x : A$ we have a loop $p_x : x =_A x$, with $d(x, x) = \mathsf{inl}(p_x)$. Let's now consider $\mathsf{ap}_{d(x,-)}(p \cdot q^{-1})$ which is a dependent path from $\mathsf{inl}(p_x)$ to itself over $p \cdot q^{-1}$. Using proposition 1.6.4, we obtain a filler of the square



Therefore, we have

$$p_x = p_x \cdot (p \cdot q^{-1}),$$

which shows that $p = q$. □

Corollary 1.10.7. *The type* **2***, the type* $\mathbb{N}$ *of natural numbers and the type* $\mathbb{Z}$ *of integers are sets.*

Proof. In all cases, we can easily define by induction on x and y a function of type

$$(x, y : A) \to ((x =_A y) + \neg(x =_A y)),$$

and then we apply Hedberg's theorem. □

Note that the circle $\mathbb{S}^1$ does not have decidable equality because we cannot give a path from x to y, for every $x, y : \mathbb{S}^1$, which depends continuously on x and y.

1.10.2 Truncations

Not every type is an n-type for some n, but we can turn any type A into an n-type $\|A\|_n$ in a universal way. This operation is called the *truncation*. There is a map $|-| : A \to \|A\|_n$ and the type $\|A\|_n$ satisfies the following induction principle: given a dependent type $P : \|A\|_n \to \mathsf{Type}$ such that $P(x)$ is n-truncated for every $x : \|A\|_n$, we can define a section of P by defining it only on elements of the form $|a|$:

$$\begin{aligned} g &: (x : \|A\|_n) \to P(x), \\ g(|a|) &:= g_{|-|}(a), \end{aligned}$$

where

$$g_{|-|} : (a : A) \to P(|a|).$$

Note that we can only apply this induction principle if all fibers of P are n-truncated. In particular, it is not possible to use it directly to define a function from $\|A\|_n$ to a type B which is not n-truncated. When we need to go around this limitation, the usual technique is to construct an n-truncated type $\widetilde{B}$ together with a map $\widetilde{B} \to B$ and to use the induction principle for $\widetilde{B}$ instead of B.

We can implement truncations using higher inductive types as described in [UF13, section 7.3] and to some extent in appendix B of this thesis, but in practice we always use the induction principle given above.

Note that truncating a type which is already truncated gives back an equivalent type.

Proposition 1.10.8. *If A is n-truncated, then the map $|-| : A \to \|A\|_n$ is an equivalence.*

Proof. We define $f : \|A\|_n \to A$ using the induction principle of $\|A\|_n$ by

$$\begin{aligned} f &: \|A\|_n \to A, \\ f(|a|) &:= a. \end{aligned}$$

Note that this is allowed because A is n-truncated. We then have $f \circ |-| = \mathsf{id}_A$ by definition and we prove that $|-| \circ f = \mathsf{id}_{\|A\|_n}$ by

$$\begin{aligned} h &: (x : \|A\|_n) \to |f(x)| =_{\|A\|_n} x, \\ h(|a|) &:= \mathsf{idp}_{|a|}. \end{aligned}$$

This is allowed because for every $x : \|A\|_n$ the type $|f(x)| =_{\|A\|_n} x$ is n-truncated, and we use the fact that $|f(|a|)| = |a|$ by definition. Therefore the map $|-| : A \to \|A\|_n$ is an equivalence of inverse f. □

We can truncate maps as well.

Proposition 1.10.9. *Given an integer n and a map $f : A \to B$, there is a map $\|f\|_n : \|A\|_n \to \|B\|_n$ and a filler of the diagram*

$$\begin{array}{ccc} A & \xrightarrow{|-|} & \|A\|_n \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \|f\|_n} \\ B & \xrightarrow[|-|]{} & \|B\|_n \end{array}$$

Proof. The map $\|f\|_n$ is defined by

$$\begin{aligned} \|f\|_n : \|A\|_n &\to \|B\|_n, \\ \|f\|_n (|a|) &:= |f(a)|, \end{aligned}$$

which is allowed because $\|B\|_n$ is n-truncated. □

An important property of truncations is the fact that loop spaces "go under" truncations (see [UF13, theorem 7.3.12] for a proof)

Proposition 1.10.10. *Given a type A, two elements $x, y : A$, and $n \geq -2$, the map*

$$\begin{aligned} f : \|x =_A y\|_n &\to |x| =_{\|A\|_{n+1}} |y|, \\ f(|p|) &:= \mathsf{ap}_{|-|}(p) \end{aligned}$$

is an equivalence.

Note that the right-hand side is an identity type in $\|A\|_{n+1}$, hence $|x| =_{\|A\|_{n+1}} |y|$ is n-truncated so the induction principle applies.

1.10.3 Mere propositions and logic

We saw in section 1.2 that given a type A and a predicate $B : A \to \mathsf{Type}$, the type

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

can be seen as the type of elements of A satisfying B. However, if B is arbitrary, there could be some $a : A$ and $b_0, b_1 : B(a)$ distinct, and (a, b_0) and (a, b_1) would be two different elements of $\sum_{x:A} B(x)$ which means that a would be counted twice. In order for $\sum_{x:A} B(x)$ to accurately represent the type of elements of A satisfying B, we need to assume additionally that $B(x)$ is a mere proposition for every $x : A$. This ensures that each element of A is counted at most once.

The type $\sum_{x:A} B(x)$ can also be seen as the proposition "there exists $x : A$ such that $B(x)$ holds". Now the problem is that even if B is a family of mere propositions, then $\sum_{x:A} B(x)$ may not be a mere proposition anymore because there could be several

elements of A satisfying B. For instance a natural number k is *composite* if there exists $n, m \geq 2$ such that $nm =_{\mathbb{N}} k$, but defining the *type* of composite numbers by

$$\sum_{k:\mathbb{N}} \sum_{n,m:\mathbb{N}} (n \geq 2 \text{ and } m \geq 2 \text{ and } nm =_{\mathbb{N}} k)$$

is incorrect because, for instance, 6 is counted twice as it is equal to both 2×3 and 3×2 (note that the $\sum_{k:\mathbb{N}}$ is interpreted in the sense of the previous paragraph while the $\sum_{n,m:\mathbb{N}}$ is interpreted in the sense of this paragraph). One way to solve this issue is to use the (-1)-truncation and to consider the type

$$\sum_{k:\mathbb{N}} \left\| \sum_{n,m:\mathbb{N}} (n \geq 2 \text{ and } m \geq 2 \text{ and } nm =_{\mathbb{N}} k) \right\|_{-1},$$

which forces the inner Σ-type to be a mere-proposition. Now the number 6 is counted only once because the two proofs that it is composite have been identified in the truncation. Therefore, a statement of the form "there exists $x : A$ such that $B(x)$ holds" can be interpreted in two different ways: either as the type $\sum_{x:A} B(x)$ if we care about the witness x (*explicit existence*) or as the type $\|\sum_{x:A} B(x)\|_{-1}$ if we want a mere proposition (*mere existence*).

The same phenomenon happens for disjunctions. Given two types A and B, the disjunction "A or B" can be interpreted as either $A + B$ if we care about which of A or B holds (*explicit disjunction*) or as $\|A + B\|_{-1}$ if we want a mere proposition (*mere disjunction*). For the universal quantifier and conjunction, however, there is no need to truncate given that a product of mere propositions is a mere proposition.

A related issue arises with equalities. When defining a type of algebraic structures, for instance groups, we usually want to assume that some equalities hold between some elements. But in homotopy type theory, that means that we have to choose *specific* paths, and there might be several such paths. For instance a type together with a binary operation could be seen as group in several different ways if there are several paths witnessing associativity. Instead of using truncations as above, we instead require the carrier type to be a set so that there cannot be several different equalities between its elements. Therefore we take the following definition of group.

Definition 1.10.11. A *group* is a set G (i.e. a 0-truncated type) together with a multiplication operation $m : G \to G \to G$, an inverse operation $i : G \to G$, and a neutral element $e : G$, such that the following equalities hold:

$$\begin{aligned} m(m(x, y), z) &=_G m(x, m(y, z)) \text{ for all } x, y, z : G, \\ m(x, i(x)) &=_G e \text{ for all } x : G, \\ m(i(x), x) &=_G e \text{ for all } x : G, \\ m(x, e) &=_G x \text{ for all } x : G, \\ m(e, x) &=_G x \text{ for all } x : G. \end{aligned}$$

It is easy to see that given a set G and a function $m : G \to G \to G$, the type corresponding to "(G, m) is a group" is a mere proposition. In other words, there is at most one proof that (G, m) is a group. This wouldn't be the case if we hadn't assumed G to be a set.

Chapter 2

First results on homotopy groups of spheres

For every pointed type A and every integer $k \geq 1$, we define a group $\pi_k(A)$ called the k-th homotopy group of A. In some sense, it describes the structure of k-dimensional loops in A. The definition of $\pi_k(A)$ alone isn't very informative, so we usually want to *compute* it, i.e. to prove that it is equivalent to some well-known group, for instance by giving an explicit presentation. Computing homotopy groups is notoriously difficult even for very simple types like the spheres $\mathbb{S}^n$. In classical homotopy theory, one can show that the first few homotopy groups of spheres are those given in table 2.1 on page 48, and in this thesis we show that we can compute some of them in homotopy type theory.

In section 2.2 we compute those in the first column (the homotopy groups of $\mathbb{S}^1$), in section 2.4 we compute all the 0s in the upper-right part ($\pi_k(\mathbb{S}^n)$ for $k < n$), and in section 2.5 we compute $\pi_2(\mathbb{S}^2)$ and show that the second and third column are identical apart for π_2. In the next chapter we prove that the diagonals are constant above the zigzag line (Freudenthal suspension theorem) and that there exists an $n : \mathbb{N}$ such that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$, and in the rest of this thesis we prove that this n is equal to 2.

2.1 Homotopy groups

Definition 2.1.1. A *pointed type* is a type A equipped with a point $\star_A$. Given two pointed types A and B, a *pointed function* between A and B is a function $f : A \to B$ equipped with a path $\star_f : f(\star_A) = \star_B$. We write $A \to_\star B$ for the type of pointed functions between A and B. The type $\mathbb{S}^1$ is pointed by base, and the type ΣA is pointed by north for every type A.

Definition 2.1.2. Given a pointed type A, its *loop space* is the type

$$\Omega A := (\star_A = \star_A)$$

	$\mathbb{S}^1$	$\mathbb{S}^2$	$\mathbb{S}^3$	$\mathbb{S}^4$	$\mathbb{S}^5$	$\mathbb{S}^6$	$\mathbb{S}^7$	$\mathbb{S}^8$	$\mathbb{S}^9$	$\mathbb{S}^{10}$
π_1	$\mathbb{Z}$	0	0	0	0	0	0	0	0	0
π_2	0	$\mathbb{Z}$	0	0	0	0	0	0	0	0
π_3	0	$\mathbb{Z}$	$\mathbb{Z}$	0	0	0	0	0	0	0
π_4	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0	0	0	0	0
π_5	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0	0	0	0
π_6	0	$\mathbb{Z}_{12}$	$\mathbb{Z}_{12}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0	0	0
π_7	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z} \times \mathbb{Z}_{12}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0	0
π_8	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_2^2$	$\mathbb{Z}_{24}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	0
π_9	0	$\mathbb{Z}_3$	$\mathbb{Z}_3$	$\mathbb{Z}_2^2$	$\mathbb{Z}_2$	$\mathbb{Z}_{24}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$	0
π_{10}	0	$\mathbb{Z}_{15}$	$\mathbb{Z}_{15}$	$\mathbb{Z}_{24} \times \mathbb{Z}_3$	$\mathbb{Z}_2$	0	$\mathbb{Z}_{24}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}$
π_{11}	0	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_{15}$	$\mathbb{Z}_2$	$\mathbb{Z}$	0	$\mathbb{Z}_{24}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$
π_{12}	0	$\mathbb{Z}_2^2$	$\mathbb{Z}_2^2$	$\mathbb{Z}_2$	$\mathbb{Z}_{30}$	$\mathbb{Z}_2$	0	0	$\mathbb{Z}_{24}$	$\mathbb{Z}_2$
π_{13}	0	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_2^3$	$\mathbb{Z}_2$	$\mathbb{Z}_{60}$	$\mathbb{Z}_2$	0	0	$\mathbb{Z}_{24}$

Figure 2.1: Homotopy groups of spheres (where $\mathbb{Z}_n$ denotes the group $\mathbb{Z}/n\mathbb{Z}$)

of paths from $\star_A$ to $\star_A$, pointed by the constant path. We can iterate this construction by defining the n-fold iterated loop space of A, for $n : \mathbb{N}$, by

$$\begin{aligned}\Omega^0 A &:= A,\\ \Omega^{n+1} A &:= \Omega(\Omega^n A).\end{aligned}$$

Definition 2.1.3. Given a pointed function $f : A \to_\star B$ between pointed types, its *looping* is

$$\begin{aligned}\Omega f &: \Omega A \to_\star \Omega B,\\ (\Omega f)(p) &:= \star_f^{-1} \cdot \mathsf{ap}_f(p) \cdot \star_f.\end{aligned}$$

Note that $\mathsf{ap}_f(p)$ has type $f(\star_A) = f(\star_A)$, therefore we need to compose it with $\star_f$ on both sides in order to obtain a loop at $\star_B$. The function Ωf is pointed by the proof of

$$\star_f^{-1} \cdot \mathsf{ap}_f(\mathsf{idp}_{\star_A}) \cdot \star_f = \mathsf{idp}_{\star_B}$$

obtained by canceling $\mathsf{ap}_f(\mathsf{idp}_{\star_A})$ (which is equal to $\mathsf{idp}_{f(\star_A)}$) and then canceling both $\star_f$ together.

We can now define the homotopy groups of a pointed type. The idea is that $\pi_n(A)$ is the set of connected components of the type of n-dimensional loops in A. The first homotopy group $\pi_1(A)$ is called the *fundamental group* of A and the $\pi_n(A)$ are called the *higher homotopy groups* of A.

Definition 2.1.4. Given a pointed type A and $n \geq 1$, the *n-th homotopy group of A* is the set

$$\pi_n(A) := \|\Omega^n A\|_0 .$$

It is equipped with the group structure induced by composition of paths, inversion of paths and the constant path.

The homotopy groups of the loop space of a type A are the homotopy groups of A shifted by one.

Proposition 2.1.5. *Given a pointed type A and $n \geq 1$, we have an isomorphism of groups*

$$\pi_n(\Omega A) \simeq \pi_{n+1}(A).$$

Proof. We first construct an equivalence $\Omega^n(\Omega A) \simeq \Omega(\Omega^n A)$ by induction on n. For $n = 0$ we take the identity equivalence and for $n + 1$ we take the composition

$$\begin{aligned}\Omega^{n+1}(\Omega A) &\simeq \Omega(\Omega^n(\Omega A)) && \text{by definition}\\ &\simeq \Omega(\Omega(\Omega^n A)) && \text{by induction hypothesis}\\ &\simeq \Omega(\Omega^{n+1} A).\end{aligned}$$

We then have

$$\begin{aligned}\pi_n(\Omega A) &\simeq \|\Omega^n(\Omega A)\|_0 \\ &\simeq \|\Omega(\Omega^n A)\|_0 \\ &\simeq \left\|\Omega^{n+1}A\right\|_0 \\ &\simeq \pi_{n+1}(A).\end{aligned}$$

It is a group homomorphism because the equivalence $\Omega^n(\Omega A) \simeq \Omega(\Omega^n A)$ preserves composition of paths. □

It turns out that the higher homotopy groups are always abelian, as we prove now.

Proposition 2.1.6 (Eckmann–Hilton argument)**.** *For any pointed type A and $n \geq 2$, the group $\pi_n(A)$ is abelian.*

Proof. We prove by induction on $n \geq 2$ that for every A the group $\pi_n(A)$ is abelian. For $n = 2$ we have to prove that $\alpha \cdot \beta = \beta \cdot \alpha$ for $\alpha, \beta : \mathsf{idp}_{\star_A} =_{\Omega A} \mathsf{idp}_{\star_A}$ and the idea is to use horizontal composition $\star$ and the exchange law. Note that for an arbitrary 2-dimensional path $\gamma : p =_{a=_A b} q$, we do not have $\mathsf{idp}_{\mathsf{idp}_a} \star \gamma = \gamma$. Indeed this is not well-typed as the left-hand side has type $(\mathsf{idp}_b \cdot p) = (\mathsf{idp}_b \cdot q)$. In order to get something having the same type as γ we need to compose on both sides with the paths $\lambda_p : \mathsf{idp}_b \cdot p = p$ and $\lambda_q : \mathsf{idp}_b \cdot q = q$. We get the following two unit laws for horizontal composition, which are proved like other coherence operations:

$$\begin{aligned}&(a, b : A)(p, q : a = b)(\gamma : p = q) \mapsto \gamma = \lambda_p^{-1} \cdot (\mathsf{idp}_{\mathsf{idp}_a} \star \gamma) \cdot \lambda_q, \\ &(a, b : A)(p, q : a = b)(\gamma : p = q) \mapsto \gamma = \rho_p^{-1} \cdot (\gamma \star \mathsf{idp}_{\mathsf{idp}_b}) \cdot \rho_q.\end{aligned}$$

In the case of α and β, both p and q are equal to idp and λ_{idp} and ρ_{idp} are both equal. We write c for short for both of them. Note that c is equal to $\mathsf{idp}_{\mathsf{idp}_{\star_A}}$ by definition if we take $(\mathsf{J}_{\mathsf{comp}})$ definitional, but the computation below does not depend on this fact. Using these two unit laws and the exchange law twice we have

$$\begin{aligned}\alpha \cdot \beta &= \mathsf{c}^{-1} \cdot (\mathsf{idp} \star \alpha) \cdot \mathsf{c} \cdot \mathsf{c}^{-1} \cdot (\beta \star \mathsf{idp}) \cdot \mathsf{c} \\ &= \mathsf{c}^{-1} \cdot (\mathsf{idp} \star \alpha) \cdot (\beta \star \mathsf{idp}) \cdot \mathsf{c} \\ &= \mathsf{c}^{-1} \cdot ((\mathsf{idp} \cdot \beta) \star (\alpha \cdot \mathsf{idp})) \cdot \mathsf{c} \\ &= \mathsf{c}^{-1} \cdot ((\beta \cdot \mathsf{idp}) \star (\mathsf{idp} \cdot \alpha)) \cdot \mathsf{c} \\ &= \mathsf{c}^{-1} \cdot (\beta \star \mathsf{idp}) \cdot (\mathsf{idp} \star \alpha) \cdot \mathsf{c} \\ &= \mathsf{c}^{-1} \cdot (\beta \star \mathsf{idp}) \cdot \mathsf{c} \cdot \mathsf{c}^{-1} \cdot (\mathsf{idp} \star \alpha) \cdot \mathsf{c} \\ &= \beta \cdot \alpha.\end{aligned}$$

Therefore $\pi_2(A)$ is abelian.

For $n > 2$, we know from proposition 2.1.5 that $\pi_{n+1}(A) \simeq \pi_n(\Omega A)$ and $\pi_n(\Omega A)$ is abelian by induction hypothesis, hence $\pi_{n+1}(A)$ is abelian as well. □

We can also define homotopy groups using pointed maps from spheres. We first show the following proposition.

Proposition 2.1.7. *Given two pointed types A and B, there is an equivalence*

$$(\Sigma A \to_\star B) \simeq (A \to_\star \Omega B).$$

Proof. We remind that we defined in section 1.7 a pointed map $\varphi_A : A \to \Omega\Sigma A$ for every type A. We have to define two maps between $\Sigma A \to_\star B$ and $A \to_\star \Omega B$ and prove that they are inverse to each other.

The map from left to right sends $f : \Sigma A \to_\star B$ to $\lambda a.(\Omega f)(\varphi_A(a))$. That map is pointed because φ_A is. The map from right to left sends $g : A \to_\star \Omega B$ to the map $\Sigma A \to_\star B$ sending both north and south to $\star_B$ and the path $\mathsf{merid}(a)$ to $g(a)$. That map is pointed by $\mathsf{idp}_{\star_B}$.

Starting from $f : \Sigma A \to_\star B$, its image by the composite is the map $\widetilde{f}$ sending both north and south to $\star_B$ and the path $\mathsf{merid}(a)$ to $\star_f^{-1} \cdot \mathsf{ap}_f(\varphi_A(a)) \cdot \star_f$. That function is equal to f via $\star_f^{-1}$ for north and $\mathsf{ap}_f(\mathsf{merid}(\star_A))^{-1} \cdot \star_f$ for south.

In the other direction, if we start from $g : A \to_\star \Omega B$, its image is the map

$$\lambda a.(\mathsf{idp}^{-1} \cdot g(a) \cdot g(\star_A)^{-1} \cdot \mathsf{idp}),$$

which is pointwise equal to g because g is pointed. □

It is easy to see that $\mathbf{2} \to_\star A$ is equivalent to A and that $\Sigma\mathbf{2}$ is equivalent to $\mathbb{S}^1$, therefore given that $\mathbb{S}^{n+1}$ is defined to be $\Sigma\mathbb{S}^n$, we have the following by induction on n.

Proposition 2.1.8. *For every pointed type A and $n : \mathbb{N}$, we have*

$$\Omega^n A \simeq (\mathbb{S}^n \to_\star A)$$

and in particular

$$\pi_n(A) \simeq \left\| \mathbb{S}^n \to_\star A \right\|_0.$$

Moreover, one can define the group operation directly on $\left\| \mathbb{S}^n \to_\star A \right\|_0$ as follows.

Definition 2.1.9. For every type A, we define the map

$$\begin{aligned}
\theta_A &: \Sigma A \to \Sigma A \vee \Sigma A, \\
\theta_A(\mathsf{north}) &:= \mathsf{inl}(\mathsf{north}), \\
\theta_A(\mathsf{south}) &:= \mathsf{inr}(\mathsf{north}), \\
\mathsf{ap}_{\theta_A}(\mathsf{merid}(a)) &:= \mathsf{ap}_{\mathsf{inl}}(\varphi_A(a)) \cdot \mathsf{push}(\star_{\mathbf{1}}) \cdot \mathsf{ap}_{\mathsf{inr}}(\varphi_A(a)).
\end{aligned}$$

Given two pointed maps $f : X \to_\star A$ and $g : Y \to_\star A$, we define the map

$$\begin{aligned}
\langle f, g \rangle &: X \vee Y \to_\star A, \\
\langle f, g \rangle(\mathsf{inl}(x)) &:= f(x), \\
\langle f, g \rangle(\mathsf{inr}(y)) &:= g(y), \\
\mathsf{ap}_{\langle f,g \rangle}(\mathsf{push}(\star_{\mathbf{1}})) &:= \star_f \cdot \star_g^{-1}.
\end{aligned}$$

Proposition 2.1.10. *Given two pointed maps $f, g : \mathbb{S}^n \to_\star A$, their multiplication when seen as elements of $\pi_n(A)$ corresponds to the composition $\langle f, g\rangle \circ \theta_{\mathbb{S}^{n-1}}$.*

Proof. Using proposition 2.1.7 we have the equivalence

$$\begin{aligned} e : (\mathbb{S}^n \to_\star A) &\simeq (\mathbb{S}^{n-1} \to_\star \Omega A), \\ e(h) &:= \lambda x.(\star_h^{-1} \cdot \mathsf{ap}_h(\varphi_{\mathbb{S}^{n-1}}(x)) \cdot \star_h). \end{aligned}$$

There is a group structure on the right-hand side given by composition of paths in ΩA, and by proposition 2.1.5 it is the same as the group structure on $\pi_n(A)$. We then have to check that $e(\langle f, g\rangle \circ \theta_{\mathbb{S}^{n-1}})$ is the composition of $e(f)$ and $e(g)$. For every $x : \mathbb{S}^{n-1}$ we have

$$\begin{aligned} e(\langle f, g\rangle \circ \theta_{\mathbb{S}^{n-1}})(x) &= \star_f^{-1} \cdot \mathsf{ap}_{\langle f,g\rangle \circ \theta_{\mathbb{S}^{n-1}}}(\varphi_{\mathbb{S}^{n-1}}(x)) \cdot \star_f \\ &= \star_f^{-1} \cdot \mathsf{ap}_{\langle f,g\rangle}(\mathsf{ap}_{\theta_{\mathbb{S}^{n-1}}}(\varphi_{\mathbb{S}^{n-1}}(x))) \cdot \star_f \\ &= \star_f^{-1} \cdot \mathsf{ap}_{\langle f,g\rangle}(\mathsf{ap}_{\mathsf{inl}}(\varphi_{\mathbb{S}^{n-1}}(x)) \cdot \mathsf{push}(\star_{\mathbf{1}}) \cdot \\ &\qquad \mathsf{ap}_{\mathsf{inr}}(\varphi_{\mathbb{S}^{n-1}}(x)) \cdot \mathsf{push}(\star_{\mathbf{1}})^{-1}) \cdot \star_f \\ &= \star_f^{-1} \cdot \mathsf{ap}_{\langle f,g\rangle}(\mathsf{ap}_{\mathsf{inl}}(\varphi_{\mathbb{S}^{n-1}}(x))) \cdot \mathsf{ap}_{\langle f,g\rangle}(\mathsf{push}(\star_{\mathbf{1}})) \\ &\quad \cdot \mathsf{ap}_{\langle f,g\rangle}(\mathsf{ap}_{\mathsf{inr}}(\varphi_{\mathbb{S}^{n-1}}(x))) \cdot \mathsf{ap}_{\langle f,g\rangle}(\mathsf{push}(\star_{\mathbf{1}}))^{-1} \cdot \star_f \\ &= \star_f^{-1} \cdot \mathsf{ap}_f(\varphi_{\mathbb{S}^{n-1}}(x)) \cdot \star_f \cdot \star_g^{-1} \cdot \mathsf{ap}_g(\varphi_{\mathbb{S}^{n-1}}(x)) \cdot \star_g \\ &= e(f)(x) \cdot e(g)(x). \end{aligned}$$

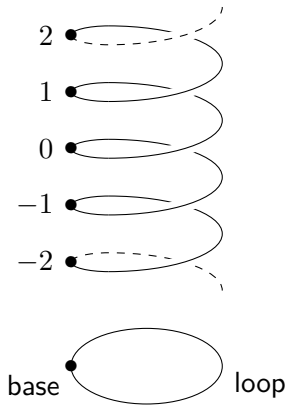
which concludes the proof. □

2.2 Homotopy groups of the circle

In order to compute the homotopy groups of the circle we define a map $U : \mathbb{S}^1 \to \mathsf{Type}$, which corresponds classically to the universal cover of $\mathbb{S}^1$, by

$$\begin{aligned} U : \mathbb{S}^1 &\to \mathsf{Type}, \\ U(\mathsf{base}) &:= \mathbb{Z}, \\ \mathsf{ap}_U(\mathsf{loop}) &:= \mathsf{ua}(\mathsf{succ}_{\mathbb{Z}}). \end{aligned}$$

This definition is valid because the function $\mathsf{succ}_{\mathbb{Z}} : \mathbb{Z} \to \mathbb{Z}$, which adds one to its argument, is an equivalence as is easy to check. Intuitively it means that U is a fibration over $\mathbb{S}^1$ whose fiber over base is the type of integers $\mathbb{Z}$ and such that transporting an integer along loop in that fibration corresponds to adding 1 to it. We have the picture



According to the flattening lemma (proposition 1.9.1), the total space of this fibration is equivalent to the higher inductive type E generated by $\mathbb{Z}$-many points $c_n : E$, and $\mathbb{Z}$-many paths $p_n : c_n =_E c_{n+1}$ relating each point with the next one.

$$\dots \xrightarrow{p_{-2}} \underset{c_{-1}}{\bullet} \xrightarrow{p_{-1}} \underset{c_0}{\bullet} \xrightarrow{p_0} \underset{c_1}{\bullet} \xrightarrow{p_1} \underset{c_2}{\bullet} \xrightarrow{p_2} \dots$$

It is easy to check that this type is contractible. We first construct a path $\ell_n : c_0 = c_n$ by induction on $n : \mathbb{Z}$ as a composition of the paths p_i or their inverses, and then for every $n : \mathbb{Z}$ we prove that $\ell_n \cdot p_n = \ell_{n+1}$, which follows from associativity of composition of paths and from the fact that $p^{-1} \cdot p = \mathsf{idp}$ for any path p. Therefore U is a fibration over $\mathbb{S}^1$ with fiber $\mathbb{Z}$ and a contractible total space and we have the following proposition (which is a consequence of [UF13, theorem 4.7.7] and of proposition 1.10.2)

Proposition 2.2.1. *Given a pointed type A and a dependent type $P : A \to \mathsf{Type}$ whose total space is contractible, for every $a : A$ and $x : P(\star_A)$ the map*

$$\mathsf{transport}^P(-, x) : (\star_A = a) \to P(a)$$

is an equivalence

Therefore we get

Proposition 2.2.2. *We have an equivalence $\Omega\mathbb{S}^1 \simeq \mathbb{Z}$ and*

$$\pi_n(\mathbb{S}^1) \simeq \begin{cases} \mathbb{Z} & \text{for } n = 1, \\ 0 & \text{for } n > 1. \end{cases}$$

Proof. Applying proposition 2.2.1 to U, $a := \mathsf{base}$ and $x := 0$ gives us an equivalence between $\Omega\mathbb{S}^1$ and $U(\mathsf{base})$, and $U(\mathsf{base})$ is equal to $\mathbb{Z}$ by definition. Therefore $\Omega\mathbb{S}^1 \simeq \mathbb{Z}$. Given that $\mathbb{Z}$ is a set by proposition 1.10.7, we have $\|\mathbb{Z}\|_0 \simeq \mathbb{Z}$ and $\Omega^k\mathbb{Z}$ is contractible for every $k \geq 1$, therefore we get the equivalences of types

$$\pi_n(\mathbb{S}^1) \simeq \begin{cases} \mathbb{Z} & \text{for } n = 1, \\ 0 & \text{for } n > 1. \end{cases}$$

Moreover $\mathsf{transport}$ commutes with composition of paths, therefore for every $n : \mathbb{Z}$ the map $\Omega\mathbb{S}^1 \to \mathbb{Z}$ sends loop^n to n, where loop^n is defined by

$$\begin{aligned} \mathsf{loop}^0 &:= \mathsf{idp}_{\mathsf{base}}, \\ \mathsf{loop}^{n+1} &:= \mathsf{loop} \cdot \mathsf{loop}^n, \\ \mathsf{loop}^{-n-1} &:= \mathsf{loop}^{-1} \cdot \mathsf{loop}^{-n}. \end{aligned}$$

This shows that the equivalence $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$ is a group isomorphism for the usual group structure on $\mathbb{Z}$. □

Proposition 2.2.3. *The circle $\mathbb{S}^1$ is 1-truncated.*

Proof. We have to prove that for every $x, y : \mathbb{S}^1$, the type $x =_{\mathbb{S}^1} y$ is a set (i.e. 0-truncated). We proceed by induction on x. Note that we only have to do the case $x = \mathsf{base}$ because for loop it follows immediately from the fact that being a set is a mere proposition. We proceed now by induction on y. For the same reason we only have to do the case $y = \mathsf{base}$. Therefore we now have to prove that $\mathsf{base} =_{\mathbb{S}^1} \mathsf{base}$ is a set, but this follows immediately from the fact that $\Omega\mathbb{S}^1 \simeq \mathbb{Z}$ and that $\mathbb{Z}$ is a set. □

2.3 Connectedness

We introduce now the notion of n-connectedness of types and maps, which can be thought of as dual to the notion of n-truncatedness. A type is n-truncated if it has no homotopical information above dimension n while a type in n-connected if it has no homotopical information *below* dimension n. We take the same convention as in [UF13], which differs from the one in classical homotopy theory. Connectivity of spaces coincides with the classical notion while an n-connected map for us corresponds to an $(n+1)$-connected map for the classical definition.

Definition 2.3.1. Given $f : A \to B$ and $b : B$, the (homotopy) *fiber* of f at b is the type

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

For instance the fiber of the unique map $A \to \mathbf{1}$ is equivalent to A (because $\mathbf{1}$ is contractible), and the fiber at a of a map $f : \mathbf{1} \to A$ is equivalent to the type $f(\star_{\mathbf{1}}) =_A a$.

Definition 2.3.2. A type A is said *n-connected* if its n-truncation $\|A\|_n$ is contractible. A map $f : A \to B$ is said *n-connected* if all of its fibers are n-connected. For short we simply say *connected* for 0-connected.

A type A is n-connected if and only if the canonical map $A \to \mathbf{1}$ is n-connected, given that the only fiber of that map is equivalent to A. On the other hand if A is pointed and n-connected, then the canonical map $\mathbf{1} \to A$ is only $(n-1)$-connected as we show below.

Proposition 2.3.3. *Every type and every map is* (-2)*-connected and every pointed type is* (-1)*-connected.*

Proof. The (-2)-truncation of a type is contractible by definition, hence every type is (-2)-connected and then every map is (-2)-connected as well.

In order to prove that a type A is (-1)-connected, we have to prove that $\|A\|_{-1}$ is contractible. Given that $\|A\|_{-1}$ is a mere proposition, it is enough to prove that it is pointed, and it is the case if A is pointed. □

Proposition 2.3.4. *A pointed type* A *is* n*-connected if and only if the canonical map* $\mathbf{1} \to A$ *is* $(n-1)$*-connected.*

Proof. For every $a : A$, the fiber of $\mathbf{1} \to A$ over a is the type $(a =_A \star_A)$ and according to proposition 1.10.10 we have

$$\|a =_A \star_A\|_{n-1} \simeq |a| =_{\|A\|_n} |\star_A|.$$

Therefore if A is n-connected, then the right-hand side is contractible which shows that the fiber $(a =_A \star_A)$ is $(n-1)$-connected and that the map $\mathbf{1} \to A$ is $(n-1)$-connected. Conversely, if the map $\mathbf{1} \to A$ is $(n-1)$-connected, i.e. $|a| =_{\|A\|_n} |\star_A|$ is contractible for every $a : A$, let's prove that $\|A\|_n$ is contractible. We need to prove that for every $x : \|A\|_n$, we have $x =_{\|A\|_n} |\star_A|$. This type is n-connected, hence it's enough to prove it for x of the form $|a|$, but it's true for such x by assumption. □

Proposition 2.3.5. *Given a pointed type* A*, if* A *is* $(n+1)$*-connected then* ΩA *is* n*-connected.*

Proof. We have $\|\Omega A\|_n \simeq \Omega \|A\|_{n+1}$ which is contractible. □

Proposition 2.3.6. *Given a type* A *and* $m \leq n$*, we have*

$$\|\|A\|_n\|_m \simeq \|A\|_m .$$

Proof. We define two maps f and g by

$$f : \|\|A\|_n\|_m \to \|A\|_m , \qquad f(||a||) := |a|,$$
$$g : \|A\|_m \to \|\|A\|_n\|_m , \qquad g(|a|) := ||a||.$$

Note that in the definition of f we use the induction principle for truncations twice and the fact that $\|A\|_m$ is n-truncated (which follows from $m \leq n$ and proposition 1.10.4). Then we define two homotopies h and k by

$$h : (x : \|\|A\|_n\|_m) \to g(f(x)) = x, \qquad h(||a||) := \mathsf{idp}_{||a||},$$
$$k : (y : \|A\|_m) \to f(g(y)) = y, \qquad k(|a|) := \mathsf{idp}_{|a|},$$

which proves that f and g are inverse to each other. □

Proposition 2.3.7. *Given a type A and $m \leq n$, if A is n-connected then A is m-connected as well.*

Proof. We assume that A is n-connected and we want to prove that $\|A\|_m$ is contractible. From the previous proposition we know that $\|A\|_m$ is equivalent to $\|\|A\|_n\|_m$, and $\|A\|_n$ is contractible by assumption, therefore $\|A\|_m$ is contractible as well. □

The following very important property can be thought of as an "induction principle" for n-connected maps. It is proved in [UF13, lemma 7.5.7]. For readability we use the notation $\prod_{b:B} P(b)$ for the function type $(b : B) \to P(b)$.

Proposition 2.3.8. *For $f : A \to B$ and $P : B \to \mathsf{Type}$, consider the map*

$$\lambda s.s \circ f : \prod_{b:B} P(b) \to \prod_{a:A} P(f(a)).$$

Then the following are equivalent:

- *f is n-connected,*
- *for every family of n-types P, the map $(\lambda s.s \circ f)$ is an equivalence,*
- *for every family of n-types P, the map $(\lambda s.s \circ f)$ has a section.*

One way to use it is as follows. In order to define a map $g : (b : B) \to P(b)$, where P is a family of n-types, it is enough to define g on elements of the form $f(a)$ for $f : A \to B$ an n-connected map. For instance if B is an $(n+1)$-connected pointed type and P is a family of n-types over B, then in order to construct a section of P it is enough to give an element of $P(\star_B)$.

Proposition 2.3.9. *Given two n-connected maps $f : A \to B$ and $g : B \to C$, the composition $g \circ f : A \to C$ is also n-connected.*

Proof. We consider $P : C \to \mathsf{Type}$ a family of n-truncated types over C together with $d : (a : A) \to P(g(f(a)))$. The composition $P \circ g$ is a family of n-truncated types over B and we have $d : (a : A) \to (P \circ g)(f(a))$ therefore, using the fact that f is n-connected, there is a function $d' : (b : B) \to P(g(b))$ such that $d'(f(a)) = d(a)$ for every $a : A$. Using now the fact that g is n-connected, there is a function $d'' : (c : C) \to P(c)$ such that $d''(g(b)) = d'(b)$. We have in particular $d''(g(f(a))) = d'(f(a)) = d(a)$, therefore $g \circ f$ is n-connected. □

Proposition 2.3.10. *Given two maps $f : C \to A$ and $g : C \to B$ such that f is n-connected, the map $\mathsf{inr} : B \to D$ is n-connected, where $D := A \sqcup^C B$.*

$$\begin{array}{ccc} C & \xrightarrow{g} & B \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \mathsf{inr}} \\ A & \xrightarrow[\mathsf{inl}]{} & D \end{array}$$

Proof. Let's consider $P : D \to \mathsf{Type}$ a family of n-truncated types and $h : (b : B) \to P(\mathsf{inr}(b))$. We want to construct a map $k : (d : D) \to P(d)$. We define $Q : A \to \mathsf{Type}$ by $Q(a) := P(\mathsf{inl}(a))$. It's a fibration of n-types and there is an element of $(c : C) \to Q(f(c))$ given by transporting $(\lambda c.h(g(c)))$ in P backwards along the equality $\mathsf{push}(c) : \mathsf{inl}(f(c)) = \mathsf{inr}(g(c))$. Therefore, given that f is n-connected, there is a section $\ell : (a : A) \to Q(a)$ of Q together with a dependent equality $e(c)$ between $\ell(f(c))$ and $h(g(c))$ in P over $\mathsf{push}(c)$.

We can now define k by

$$\begin{aligned} k &: (d : D) \to P(d), \\ k(\mathsf{inl}(a)) &:= \ell(a), \\ k(\mathsf{inr}(b)) &:= h(b), \\ \mathsf{ap}_k(\mathsf{push}(c)) &:= e(c), \end{aligned}$$

and it agrees with h on element of the form $\mathsf{inr}(b)$. Therefore, inr is n-connected. □

Proposition 2.3.11. *For any X, the map $|-| : X \to \|X\|_n$ is n-connected.*

Proof. It's an immediate consequence of proposition 2.3.8 and of the induction principle for truncations. □

2.4 Lower homotopy groups of spheres

We can now prove that all homotopy groups of spheres of the form $\pi_k(\mathbb{S}^n)$ for $k < n$ are trivial.

Proposition 2.4.1. *For every $k, n : \mathbb{N}$ such that $k < n$, we have*

$$\pi_k(\mathbb{S}^n) \simeq \mathbf{1}.$$

It follows from the following proposition, as shown below.

Proposition 2.4.2. *For every $n : \mathbb{N}$, the n-sphere $\mathbb{S}^n$ is $(n-1)$-connected.*

Proof. For $n = 0$ this is true because $\mathbb{S}^0 := \mathbf{2}$ is pointed hence (-1)-connected. If we assume now that $\mathbb{S}^n$ is $(n-1)$-connected, we have the following pushout diagram where the top map is $(n-1)$-connected, hence the bottom map is $(n-1)$-connected as well by proposition 2.3.10.

$$\begin{array}{ccc} \mathbb{S}^n & \longrightarrow & \mathbf{1} \\ \downarrow & \ulcorner & \downarrow \\ \mathbf{1} & \longrightarrow & \mathbb{S}^{n+1} \end{array}$$

Therefore by proposition 2.3.4 the type $\mathbb{S}^{n+1}$ is n-connected, which concludes the proof. □

Proof of proposition 2.4.1. Given $k, n : \mathbb{N}$ such that $k < n$, the type $\mathbb{S}^n$ is $(n-1)$-connected and taking Ω^k decreases the connectivity by k hence $\Omega^k(\mathbb{S}^n)$ is $(n-1-k)$-connected and we have $(n-1-k) \geq 0$ because $k < n$. Therefore $\Omega^k(\mathbb{S}^n)$ is 0-connected, which shows that $\pi_k(\mathbb{S}^n)$ is contractible. □

2.5 The Hopf fibration

Given a fibration $P : B \to \mathsf{Type}$ we often use the notation

$$F \xrightarrow{\ i\ } E \xrightarrow{\ p\ } B,$$

where $F := P(\star_B)$ is the *fiber*, $E := \sum_{x:B} P(x)$ is the *total space*, i sends y to $(\star_B, y)$ and p is the first projection.

In this section we introduce H-spaces and the Hopf construction which is a fibration

$$A \longrightarrow A * A \longrightarrow \Sigma A$$

for any connected H-space A. Applying this construction to the circle gives us the Hopf fibration

$$\mathbb{S}^1 \longrightarrow \mathbb{S}^3 \longrightarrow \mathbb{S}^2.$$

Using the long exact sequence of homotopy groups, it will imply that $\pi_2(\mathbb{S}^2) \simeq \mathbb{Z}$ and that $\pi_n(\mathbb{S}^2) \simeq \pi_n(\mathbb{S}^3)$ for all $n \geq 3$.

Definition 2.5.1. An *H-space* is a pointed type A together with a binary operation $\mu : A \to A \to A$ and equalities

$$\begin{aligned} \mu_l &: (a : A) \to \mu(\star_A, a) =_A a, \\ \mu_r &: (a : A) \to \mu(a, \star_A) =_A a, \\ \mu_{lr} &: \mu_l(\star_A) = \mu_r(\star_A). \end{aligned}$$

For instance every group has an obvious structure of H-space, where μ is given by the multiplication of the group, μ_l and μ_r follow from the unit laws, and μ_{lr} follows from the fact that a group is a set. In the classical definition of H-spaces, the homotopies μ_l and μ_r are required to fix the basepoint (which cannot be expressed in homotopy type theory), and as a consequence the two-dimensional path μ_{lr} is not required anymore.

Proposition 2.5.2. *If A is a connected H-space, then for every $a : A$ the functions $\mu(a, -) : A \to A$ and $\mu(-, a) : A \to A$ are equivalences.*

Proof. What we want to prove is a mere proposition depending on $a : A$. Given that A is pointed and 0-connected, it is enough to prove it for the basepoint $\star_A$. But the functions $\mu(\star_A, -)$ and $\mu(-, \star_A)$ are both equal to the identity function (using μ_l and μ_r), hence they are equivalences. □

Proposition 2.5.3. *Given a connected H-space A, there is a fibration over ΣA with fiber A and whose total space is equivalent to $A * A$.*

Proof. We define $H : \Sigma A \to \mathsf{Type}$ by

$$\begin{aligned} H(\mathsf{north}) &:= A, \\ H(\mathsf{south}) &:= A, \\ \mathsf{ap}_H(\mathsf{merid}(a)) &:= \mathsf{ua}(\mu(-, a)). \end{aligned}$$

The definition is valid because $\mu(-, a)$ is an equivalence by proposition 2.5.2.

By the flattening lemma (proposition 1.9.2), the total space of this fibration is equivalent to the pushout of the span

$$A \xleftarrow{\mathsf{fst}} A \times A \xrightarrow{\mu} A.$$

The map $\alpha : A \times A \to A \times A$ defined by $\alpha(x, y) = (x, \mu(x, y))$ is an equivalence, its inverse is the map $\alpha^{-1}(u, v) = (u, \mu(u, -)^{-1}(v))$, hence we have the equivalence of spans

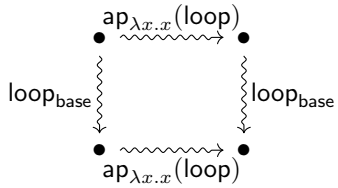
$$\begin{array}{ccccc} A & \xleftarrow{\mathsf{fst}} & A \times A & \xrightarrow{\mu} & A \\ \| & & \downarrow{\scriptstyle \alpha} & & \| \\ A & \xleftarrow{\mathsf{fst}} & A \times A & \xrightarrow{\mathsf{snd}} & A \end{array}$$

This shows that the total space of the fibration we constructed is equivalent to the join $A * A$. □

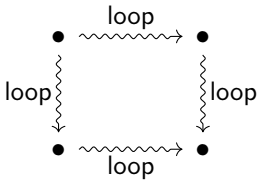
Proposition 2.5.4. *There is a structure of H-space on the circle.*

Proof. We define the map $\mu : \mathbb{S}^1 \to \mathbb{S}^1 \to \mathbb{S}^1$ by induction on the first argument.

- For base, we take $\mu(\mathsf{base}, x) := x$.
- For loop, we have to construct an equality between the identity function and itself in the type $\mathbb{S}^1 \to \mathbb{S}^1$. By function extensionality, it is equivalent to construct a function $\mathsf{loop}\_ : (x : \mathbb{S}^1) \to x =_{\mathbb{S}^1} x$, i.e. a path loop_x from x to itself for every point $x : \mathbb{S}^1$. We proceed again by induction on x.
  - For base, we take $\mathsf{loop}_{\mathsf{base}} := \mathsf{loop}$.
  - For loop, we have to fill the square



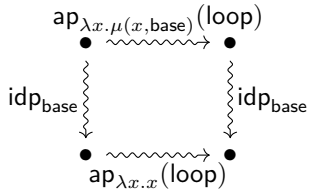
    which is equal to the square



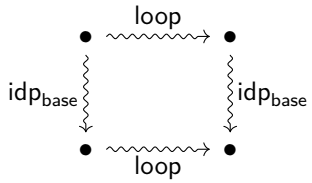
    which is filled by $\mathsf{idp}_{\mathsf{loop} \cdot \mathsf{loop}}$.

We construct μ_l, μ_r and μ_{lr}:

- For μ_l we have $\mu(\mathsf{base}, x) =_{\mathbb{S}^1} x$ by definition, so we take $\mu_l(x) := \mathsf{idp}_{\mathsf{base}}$ for every x.
- For $\mu_r : (x : \mathbb{S}^1) \to \mu(x, \mathsf{base}) =_{\mathbb{S}^1} x$, we proceed by induction on x.
  - For base, it's true by definition, so we take $\mu_r(\mathsf{base}) := \mathsf{idp}_{\mathsf{base}}$.
  - For loop, we have to fill the square



    which is equal to the square



    which is again filled by some coherence operation.

- Finally, both $\mu_l(\mathsf{base})$ and $\mu_r(\mathsf{base})$ are equal to $\mathsf{idp}_{\mathsf{base}}$ by definition, therefore we take $\mu_{lr} := \mathsf{idp}_{\mathsf{idp}_{\mathsf{base}}}$. □

Proposition 2.5.5. *There is a fibration over $\mathbb{S}^2$ with fiber $\mathbb{S}^1$ and whose total space is equivalent to $\mathbb{S}^3$.*

Proof. We apply the Hopf construction to the circle, which is a connected H-space according to the previous proposition. We get a fibration over $\Sigma\mathbb{S}^1$ with fiber $\mathbb{S}^1$ and total space $\mathbb{S}^1 * \mathbb{S}^1$, and we have constructed in proposition 1.8.8 an equivalence $\mathbb{S}^1 * \mathbb{S}^1 \simeq \mathbb{S}^3$ so we get the Hopf fibration. □

This fibration is called the *Hopf fibration* and the induced map $\mathbb{S}^3 \to \mathbb{S}^2$ is called the *Hopf map*.

2.6 The long exact sequence of a fibration

The *long exact sequence of homotopy groups* of a fibration is a long exact sequence relating the homotopy groups of the base space, the fiber space and the total space of a fibration. A construction of it in homotopy type theory is also presented in [UF13, section 8.4]. We present here a different construction.

Definition 2.6.1. A sequence of pointed sets (0-truncated types) and pointed maps

$$\ldots \longrightarrow A_{n-1} \xrightarrow{f_{n-1}} A_n \xrightarrow{f_n} A_{n+1} \longrightarrow \ldots$$

is called a *long exact sequence (of sets)* if for every n and for every $x : A_n$, then x is merely in the image of f_{n-1} if and only if it is in the kernel of f_n. In other words, it means that we have for every $x : A_n$ an equivalence

$$\left\| \sum_{a:A_{n-1}} f_{n-1}(a) =_{A_n} x \right\|_{-1} \simeq (f_n(x) =_{A_{n+1}} \star_{A_{n+1}}) .$$

Note that the left-hand side is a mere proposition by definition, and the right-hand side as well because A_{n+1} is assumed to be a set. Therefore having an equivalence is equivalent to having two functions going back and forth. When all the A_n are groups and all the f_n are group homomorphisms (as is usually the case), we talk about a *long exact sequence of groups*.

When the A_n are not assumed to be sets, we introduce instead the notion of fiber sequence.

Definition 2.6.2. A sequence of pointed types and pointed maps

$$\ldots \longrightarrow A_{n-1} \xrightarrow{f_{n-1}} A_n \xrightarrow{f_n} A_{n+1} \longrightarrow \ldots$$

is called a *fiber sequence* if for every n and for every $x : A_n$, the untruncated types "x is in the image of f_{n-1}" and "x is the kernel of f_n" are equivalent. In other words, it means that we have for every $x : A_n$ an equivalence

$$\left(\sum_{a:A_{n-1}} f_{n-1}(a) =_{A_n} x \right) \simeq (f_n(x) =_{A_{n+1}} \star_{A_{n+1}}) .$$

The relationship between long exact sequences and fiber sequences is given by the following proposition.

Proposition 2.6.3. *Given a fiber sequence*

$$\ldots \longrightarrow A_{n-1} \xrightarrow{f_{n-1}} A_n \xrightarrow{f_n} A_{n+1} \longrightarrow \ldots,$$

then its 0*-truncation*

$$\ldots \longrightarrow \|A_{n-1}\|_0 \xrightarrow{\|f_{n-1}\|_0} \|A_n\|_0 \xrightarrow{\|f_n\|_0} \|A_{n+1}\|_0 \longrightarrow \ldots$$

is a long exact sequence.

Proof. Given $x' : \|A_n\|_0$, we want to construct an element of the type

$$\left\| \sum_{a' : \|A_{n-1}\|_0} \|f_{n-1}\|_0 (a') =_{\|A_n\|_0} x' \right\|_{-1} \simeq \left(\|f_n\|_0 (x') =_{\|A_{n+1}\|_0} | \star_{A_{n+1}} | \right).$$

This type is a set, therefore we can assume that x' is of the form $|x|$ with $x : A_n$. Applying proposition 1.10.10, we see that the right-hand side is equivalent to the type $\|f_n(x) =_{A_{n+1}} \star_{A_{n+1}}\|_{-1}$, and by assumption we have

$$\left(\sum_{a : A_{n-1}} f_{n-1}(a) =_{A_n} x \right) \simeq \left(f_n(x) =_{A_{n+1}} \star_{A_{n+1}} \right).$$

Therefore it is enough to construct an equivalence

$$\left\| \sum_{a:A} B(a) \right\|_{-1} \simeq \left\| \sum_{a' : \|A\|_0} B'(a') \right\|_{-1}$$

given a family of equivalences $e : (a : A) \to \|B(a)\|_{-1} \simeq B'(|a|)$, where A is A_{n-1}, $B(a)$ is $(f_{n-1}(a) =_{A_n} x)$, and $B'(a')$ is $(\|f_{n-1}\|_0 (a') =_{\|A_n\|_0} |x|)$.

From left to right we do an induction on the outer truncation, and we send $|(a, b)|$ to $|(|a|, e_a(|b|))|$. From right to left we do an induction on the outer truncation, an induction on $\|A\|_0$, and, using the equivalence e_a, an induction on $\|B(a)\|_{-1}$, and we send $|(|a|, e_a(|b|))|$ to $|(a, b)|$. □

Definition 2.6.4. Given a pointed type B and a dependent type $P : B \to \mathsf{Type}$ together with a pointing of $F := P(\star_B)$, we define

$$P^\Omega : \Omega B \to \mathsf{Type},$$
$$P^\Omega(p) := (\star_F =_p^P \star_F).$$

Note that the fiber of P^Ω is ΩF and that the total space of P^Ω is the loop space of the total space of P, according to proposition 1.6.8. Moreover there is a map

$$d : \Omega B \to F,$$
$$d(p) := \mathsf{transport}^P(p, \star_F).$$

Let E be the total space of P. By iterating the previous construction, we get the diagram

$$\begin{array}{ccccc} & & \cdots & & \\ & \swarrow & & & \\ \Omega^3 F & \longrightarrow & \Omega^3 E & \longrightarrow & \Omega^3 B \\ & \swarrow & & & \\ \Omega^2 F & \longrightarrow & \Omega^2 E & \longrightarrow & \Omega^2 B \\ & \swarrow & & & \\ \Omega F & \longrightarrow & \Omega E & \longrightarrow & \Omega B, \end{array}$$

and after 0-truncating, we get

$$\begin{array}{ccccc} & & & & \cdots \\ & & & \swarrow & \\ \pi_3(F) & \longrightarrow & \pi_3(E) & \longrightarrow & \pi_3(B) \\ & & & \swarrow & \\ \pi_2(F) & \longrightarrow & \pi_2(E) & \longrightarrow & \pi_2(B) \\ & & & \swarrow & \\ \pi_1(F) & \longrightarrow & \pi_1(E) & \longrightarrow & \pi_1(B). \end{array} \tag{2.6.5}$$

It is easy to see that the maps $\pi_n(F) \to \pi_n(E)$ and $\pi_n(E) \to \pi_n(B)$ are group homomorphisms and for the maps $\pi_{n+1}(B) \to \pi_n(F)$ it follows from the following proposition.

Proposition 2.6.6. *The map*

$$\begin{aligned} d &: \Omega^2 B \to \Omega F, \\ d(p) &:= \mathsf{transport}^{P^\Omega}(p, \mathsf{idp}_{\star_F}) \end{aligned}$$

satisfies $d(p \cdot q) = d(p) \cdot d(q)$ *for every* $p, q : \Omega^2 B$.

Proof. The key observation is that for every $\ell : \Omega B$, $\alpha : \mathsf{idp}_{\star_B} = \ell$ and $r : \Omega F$, we have

$$\mathsf{transport}^{P^\Omega}(\alpha, r) =_{P^\Omega(\ell)} \left(r \cdot \mathsf{transport}^{P^\Omega}(\alpha, \mathsf{idp}_{\star_F})\right),$$

where on the right-hand side we compose the homogeneous path r with the dependent path $\mathsf{transport}^{P^\Omega}(\alpha, \mathsf{idp}_{\star_F})$. This observation is simply proved by path induction on α.

Then we have

$$\begin{aligned} d(p \cdot q) &= \mathsf{transport}^{P^\Omega}(p \cdot q, \mathsf{idp}_{\star_F}) \\ &= \mathsf{transport}^{P^\Omega}(q, \mathsf{transport}^{P^\Omega}(p, \mathsf{idp}_{\star_F})) \\ &= \mathsf{transport}^{P^\Omega}(q, d(p)) \\ &= d(p) \cdot \mathsf{transport}^{P^\Omega}(q, \mathsf{idp}_{\star_F}) \\ &= d(p) \cdot d(q). \end{aligned}$$

□

Proposition 2.6.7. *The sequence 2.6.5 is a long exact sequence of groups.*

Proof. It suffices to prove that the sequence

$$\Omega E \longrightarrow \Omega B \longrightarrow F \longrightarrow E \longrightarrow B$$

is a fiber sequence at E, F and ΩB.

An element (b, f) of E is in the kernel of the map $E \to B$ if and only if $b = \star_B$ and in the image of the map $F \to E$ if and only if there exists $f' : F$ such that $(b, f) = (\star_B, f')$.

They are equivalent because if $(b, f) = (\star_B, f')$ then $b = \star_B$, and conversely if we have $p : b = \star_B$ then we can define f' by transporting f in P along p and we get an equality $(b, f) = (\star_B, f')$ in E.

An element f of F is in the kernel of the map $F \to E$ if and only if $(\star_B, f) = (\star_B, \star_F)$, i.e. that there is a path $p : \star_B = \star_B$ such that $f = \mathsf{transport}^P(p, \star_F)$, which is exactly the condition that f is in the image of $\Omega B \to F$.

A loop $p : \Omega B$ is in the kernel of the map $\Omega B \to F$ if and only if $\mathsf{transport}^P(p, \star_F) = \star_F$, and is in the image of $\Omega E \to \Omega B$ if and only if there exists a dependent path of type $\star_F =^P_p \star_F$. These two conditions are equivalent. □

Proposition 2.6.8. *We have $\pi_2(\mathbb{S}^2) \simeq \mathbb{Z}$ and for every $k \geq 3$ we have $\pi_k(\mathbb{S}^2) \simeq \pi_k(\mathbb{S}^3)$.*

Proof. Using the results already obtained in this chapter, the long exact sequence of homotopy groups of the Hopf fibration is

$$\begin{array}{ccccc} & & & & \cdots \\ & & & \swarrow & \\ 0 & \longrightarrow & \pi_4(\mathbb{S}^3) & \longrightarrow & \pi_4(\mathbb{S}^2) \\ & & & \swarrow & \\ 0 & \longrightarrow & \pi_3(\mathbb{S}^3) & \longrightarrow & \pi_3(\mathbb{S}^2) \\ & & & \swarrow & \\ 0 & \longrightarrow & 0 & \longrightarrow & \pi_2(\mathbb{S}^2) \\ & & & \swarrow & \\ \mathbb{Z} & \longrightarrow & 0 & \longrightarrow & 0 \end{array}$$

therefore the result follows. □

We can also use the long exact sequence of homotopy groups to describe how n-connected maps act on homotopy groups.

Proposition 2.6.9. *Given two pointed types A and B and an n-connected pointed map $f : A \to B$, the map $\pi_k(f) : \pi_k(A) \to \pi_k(B)$ is an isomorphism for $k \leq n$ and a surjection for $k = n + 1$.*

Proof. The dependent type

$$P : B \to \mathsf{Type},$$
$$P(b) := \left(\sum_{a:A} f(a) =_B b \right)$$

corresponds to a fibration

$$\mathsf{fib}_f(\star_B) \longrightarrow A \xrightarrow{f} B.$$

Moreover, $\mathsf{fib}_f(\star_B)$ is n-connected, therefore for $k \leq n$ we have

$$\begin{aligned}\pi_k(\mathsf{fib}_f(\star_B)) &\simeq \Omega^k \left\|\mathsf{fib}_f(\star_B)\right\|_k \\ &\simeq \Omega^k \left\|\left\|\mathsf{fib}_f(\star_B)\right\|_n\right\|_k ,\end{aligned}$$

which is contractible. Therefore, the long exact sequence of homotopy groups for the fibration above has n zeros on the left column, which shows that $\pi_{n+1}(f)$ is surjective and that $\pi_k(f)$ is bijective for $k \leq n$. □

Chapter 3

The James construction

Given a pointed type A, the James construction is a sequence of approximations of $\Omega\Sigma A$ which are often easier to study than $\Omega\Sigma A$. We define the James construction and use it to prove the Freudenthal suspension theorem and the fact that there exists a natural number n such that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$.

3.1 Sequential colimits

In this section we define sequential colimits and we prove a few properties about them.

Definition 3.1.1. Given a family of types $(A_n)_{n:\mathbb{N}}$ and a family of functions $(i_n)_{n:\mathbb{N}}$ with $i_n : A_n \to A_{n+1}$, as in the diagram

$$A_0 \xrightarrow{i_0} A_1 \xrightarrow{i_1} A_2 \xrightarrow{i_2} A_3 \longrightarrow \dots,$$

we define the *sequential colimit* of $(A_n)_{n:\mathbb{N}}$ as the higher inductive type A_∞ generated by the two constructors

$$\begin{aligned} \mathsf{in} &: (n : \mathbb{N}) \to A_n \to A_\infty, \\ \mathsf{push} &: (n : \mathbb{N})(x : A_n) \to \mathsf{in}_n(x) =_{A_\infty} \mathsf{in}_{n+1}(i_n(x)). \end{aligned}$$

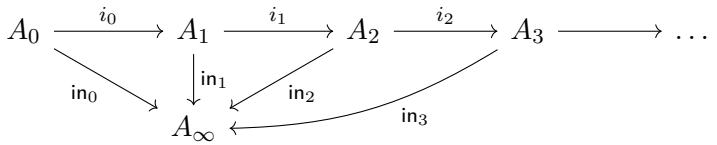
The induction principle for sequential colimits states that given a dependent type $P : A_\infty \to \mathsf{Type}$, a function $f : (x : A_\infty) \to P(x)$ can be defined by

$$\begin{aligned} f &: (x : A_\infty) \to P(x), \\ f(\mathsf{in}_n(x)) &:= f_{\mathsf{in}}(n, x), \\ \mathsf{ap}_f(\mathsf{push}_n(x)) &:= f_{\mathsf{push}}(n, x), \end{aligned}$$

where we have

$$\begin{aligned} f_{\mathsf{in}} &: (n : \mathbb{N})(x : A_n) \to P(\mathsf{in}_n(x)), \\ f_{\mathsf{push}} &: (n : \mathbb{N})(x : A_n) \to f_{\mathsf{in}}(n, x) =^P_{\mathsf{push}_n(x)} f_{\mathsf{in}}(n+1, i_n(x)). \end{aligned}$$

We have the diagram



and the map $\mathsf{in}_0 : A_0 \to A_\infty$ is called the *transfinite composition* of the family $(i_n)_{n:\mathbb{N}}$. The following proposition shows (among other things) that in_k is the transfinite composition of the family $(i_{n+k})_{n:\mathbb{N}}$.

Proposition 3.1.2. *There is an equivalence between A_∞ and the sequential colimit A'_∞ of the diagram*

$$A_1 \xrightarrow{i_1} A_2 \xrightarrow{i_2} A_3 \xrightarrow{i_3} A_4 \longrightarrow \ldots,$$

and the maps $\mathsf{in}'_n : A_{n+1} \to A'_\infty$ correspond to the maps $\mathsf{in}_{n+1} : A_{n+1} \to A_\infty$.

Proof. We write in' and push' for the constructors of A'_∞. The map from A'_∞ to A_∞ sends $\mathsf{in}'_n(x)$ to $\mathsf{in}_{n+1}(x)$ and $\mathsf{push}'_n(x)$ to $\mathsf{push}_{n+1}(x)$, and the map from A_∞ to A'_∞ sends $\mathsf{in}_0(x)$ to $\mathsf{in}'_0(i_0(x))$, $\mathsf{in}_{n+1}(x)$ to $\mathsf{in}'_n(x)$, $\mathsf{push}_0(x)$ to $\mathsf{idp}_{i_0(x)}$ and $\mathsf{push}_{n+1}(x)$ to $\mathsf{push}'_n(x)$. One can easily check that those two functions are inverse to each other. □

The main result of this section is the fact that connectedness is stable under transfinite composition.

Proposition 3.1.3. *If all the maps i_0, i_1, ... are k-connected, then the transfinite composition of $(i_n)_{n:\mathbb{N}}$ is also k-connected.*

Proof. Let's consider $P : A_\infty \to \mathsf{Type}$ a family of k-truncated types and $d_0 : (x : A_0) \to P(\mathsf{in}_0(x))$. Using proposition 2.3.8, it is enough to construct a section d of P which is equal to d_0 on A_0 to conclude that in_0 is k-connected. We define a family of maps $d_n : (x : A_n) \to P(\mathsf{in}_n(x))$ by induction on n, starting with the given d_0 for $n = 0$, as follows. Let's consider

$$P_{n+1} : A_{n+1} \to \mathsf{Type},$$
$$P_{n+1}(x) := P(\mathsf{in}_{n+1}(x)).$$

It is a family of k-truncated types, the map i_n is k-connected, and we have

$$\widetilde{d}_n : (x : A_n) \to P_{n+1}(i_n(x)),$$
$$\widetilde{d}_n(x) := \mathsf{transport}^P(\mathsf{push}_n(x), d_n(x)),$$

therefore, using proposition 2.3.8 again, there is a map $d_{n+1} : (x : A_{n+1}) \to P(\mathsf{in}_{n+1}(x))$ satisfying

$$d_{n+1}(i_n(x)) =^P_{\mathsf{push}_n(x)} d_n(x).$$

The family $(d_n)_{n:\mathbb{N}}$ together with those equalities gives a section of P which is equal to d_0 on A_0. Therefore, the map in_0 is k-connected, which is what we wanted to prove. □

3.2 The James construction

Proposition 3.2.1. *Given a k-connected pointed type A, with $k \geq 0$, there is a sequence of types $(J_n A)_{n:\mathbb{N}}$ and maps $i_n : J_n A \to J_{n+1} A$, as in the diagram*

$$J_0 A \xrightarrow{i_0} J_1 A \xrightarrow{i_1} J_2 A \xrightarrow{i_2} J_3 A \xrightarrow{i_3} \dots,$$

such that for every $n : \mathbb{N}$, the map i_n is $(n(k+1)+(k-1))$-connected, and such that the sequential colimit of $(J_n A)_{n:\mathbb{N}}$ is equivalent to $\Omega\Sigma A$. Moreover, for low values of n we have

- *$J_0 A \simeq \mathbf{1}$,*
- *$J_1 A \simeq A$, the map i_0 is the inclusion of the basepoint, and the induced map from A to $\Omega\Sigma A$ is equal to φ_A,*
- *$J_2 A \simeq (A \times A) \sqcup^{A \vee A} A$, where the two maps are the folding map*

$$\begin{aligned} \nabla_A : A \vee A &\to A, \\ \nabla_A(\mathsf{inl}(x)) &:= x, \\ \nabla_A(\mathsf{inr}(y)) &:= y, \\ \mathsf{ap}_{\nabla_A}(\mathsf{push}(\star_{\mathbf{1}})) &:= \mathsf{idp}_{\star_A} \end{aligned}$$

and the inclusion of the wedge in the product

$$\begin{aligned} i^{\vee}_{A,A} : A \vee A &\to A \times A, \\ i^{\vee}_{A,A}(\mathsf{inl}(a)) &:= (a, \star_A), \\ i^{\vee}_{A,A}(\mathsf{inr}(a)) &:= (\star_A, a), \\ \mathsf{ap}_{i^{\vee}_{A,A}}(\mathsf{push}(\star_{\mathbf{1}})) &:= \mathsf{idp}_{(\star_A, \star_A)}, \end{aligned}$$

and $i_1 : A \to J_2 A$ is equal to inr.

Given that connectedness of maps is downward-closed and stable by transfinite composition, the map in_n from $J_n A$ to the colimit is $(n(k+1)+(k-1))$-connected, hence there is an $(n(k+1)+(k-1))$-connected map from $J_n A$ to $\Omega\Sigma A$ which allows us to study low-dimensional homotopy groups of $\Omega\Sigma A$ by studying those of some $J_n A$ instead. We have in particular the following two corollaries, which are the only two properties of the James construction that we use here.

Corollary 3.2.2 (Freudenthal suspension theorem)**.** *Given a k-connected pointed type A, the map*

$$\varphi_A : A \to \Omega\Sigma A$$

is $2k$-connected.

Corollary 3.2.3. *Given a k-connected pointed type A, there is a $(3k+1)$-connected map*

$$(A \times A) \sqcup^{A \vee A} A \to \Omega\Sigma A.$$

Note that both corollaries are also true in the case $k = -1$ because every map is (-2)-connected.

We will define the types $(J_n A)$ as pushouts, by induction on n, and prove that their colimit $J_\infty A$ is equivalent to another type JA which has a much simpler definition. Intuitively, JA corresponds to the free monoid on A with unit $\star_A$, while $J_n A$ corresponds to the "subtype" of JA consisting of products of length at most n. We then prove that the space $\Omega\Sigma A$ is equivalent to JA, where we crucially use the fact that A is connected.

3.2.1 Definition of the family $(J_n A)$

We define the types $J_n A$ together with three functions

$$\begin{aligned} i_n &: J_n A \to J_{n+1} A, \\ \alpha_n &: A \times J_n A \to J_{n+1} A, \\ \beta_n &: (x : J_n A) \to \alpha_n(\star_A, x) =_{J_{n+1} A} i_n(x), \end{aligned}$$

by induction on n as follows.

- $J_0 A := \mathbf{1}$ (whose unique element is called ϵ),
- $J_1 A := A$, $i_0(\epsilon) := \star_A$, $\alpha_0(a, \epsilon) := a$ and $\beta_0(\epsilon) := \mathsf{idp}_{\star_A}$,
- $J_{n+2} A$, i_{n+1}, α_{n+1} and $\beta_{n+1} := \mathsf{push} \circ \mathsf{inr}$ are defined by the pushout diagram

$$\begin{array}{ccc} (A \times J_n A) \sqcup^{J_n A} J_{n+1} A & \xrightarrow{g} & J_{n+1} A \\ \Big\downarrow{\scriptstyle f} & & \Big\downarrow{\scriptstyle i_{n+1}} \\ A \times J_{n+1} A & \xrightarrow[\alpha_{n+1}]{} & J_{n+2} A \end{array} \tag{3.2.4}$$

  where the pushout at the top-left of the diagram is defined by the maps $x \mapsto (\star_A, x)$ and i_n, and the maps f and g are defined by

$$\begin{aligned} f(\mathsf{inl}(a, x)) &:= (a, i_n(x)), & g(\mathsf{inl}(a, x)) &:= \alpha_n(a, x), \\ f(\mathsf{inr}(y)) &:= (\star_A, y), & g(\mathsf{inr}(y)) &:= y, \\ \mathsf{ap}_f(\mathsf{push}(x)) &:= \mathsf{idp}, & \mathsf{ap}_g(\mathsf{push}(x)) &:= \beta_n(x). \end{aligned}$$

The pushout also consists of

$$\begin{aligned} \gamma_n &: (a : A)(x : J_n A) \to \alpha_{n+1}(a, i_n(x)) =_{J_{n+2} A} i_{n+1}(\alpha_n(a, x)), \\ \gamma_n(a, x) &:= \mathsf{push}(\mathsf{inl}(a, x)) \end{aligned}$$

and

$$\eta_n : (x : J_n A) \to (\gamma_n(\star_A, x) \cdot \mathsf{ap}_{i_{n+1}}(\beta_n(x))) = \beta_{n+1}(i_n(x)),$$

which follows from $\mathsf{ap}_{\mathsf{push}}(\mathsf{push}(x))$ and which is an equality in the type

$$\alpha_{n+1}(\star_A, i_n(x)) =_{J_{n+2}A} i_{n+1}(i_n(x)).$$

Note that, in order to define a map out of $J_{n+2}A$, it is enough to define it for elements of the form $i_{n+1}(x)$ and $\alpha_{n+1}(a, x)$, for paths of the form $\beta_{n+1}(x)$ and $\gamma_n(a, x)$, and for 2-paths of the form $\eta_n(x)$.

3.2.2 Colimit of the family $(J_n A)$

We now prove that the colimit $J_\infty A$ of the family $(J_n A)$ is equivalent to the higher inductive type JA with constructors

$$\begin{aligned}
&\varepsilon : JA,\\
&\alpha : A \to JA \to JA,\\
&\delta : (x : JA) \to x =_{JA} \alpha(\star_A, x).
\end{aligned}$$

In JA we just have a unit called ε, an action of A on JA called α and a proof that multiplying by $\star_A$ is the identity, called δ. The induction principle for JA states that given a dependent type $P : JA \to \mathsf{Type}$, a function $f : (x : JA) \to P(x)$ can be defined by

$$\begin{aligned}
f &: (x : JA) \to P(x),\\
f(\varepsilon) &:= f_\varepsilon,\\
f(\alpha(a, x)) &:= f_\alpha(a, x, f(x)),\\
\mathsf{ap}_f(\delta(x)) &:= f_\delta(x, f(x)),
\end{aligned}$$

where we have

$$\begin{aligned}
&f_\varepsilon : P(\varepsilon),\\
&f_\alpha : (a : A)(x : JA) \to P(x) \to P(\alpha(a, x)),\\
&f_\delta : (x : JA)(y : P(x)) \to y =^P_{\delta(x)} f_\alpha(\star_A, x, y).
\end{aligned}$$

Note that f is used recursively in $f(\alpha(a, x))$ and in $\mathsf{ap}_f(\delta(x))$, because JA is a recursive higher inductive type.

In order to prove that JA and $J_\infty A$ are equivalent, we first define equivalents of γ_n, η_n, in_n and push_n in JA and then equivalents of ε, α, δ, γ and η in $J_\infty A$.

Structure on JA We define the map γ, where we simply apply δ twice, by

$$\begin{aligned}\gamma &: (a : A)(x : JA) \to \alpha(a, \alpha(\star_A, x)) = \alpha(\star_A, \alpha(a, x)),\\ \gamma(a, x) &:= (\mathsf{ap}_{\alpha(a,-)}(\delta(x)))^{-1} \cdot \delta(\alpha(a, x)).\end{aligned}$$

Then we define the map

$$\eta : (x : JA) \to \gamma(\star_A, x) = \mathsf{idp}$$

using naturality of δ on $\delta(x)$, which is the square

$$\begin{array}{ccc} x & \overset{\delta(x)}{\rightsquigarrow} & \alpha(\star_A, x) \\ {\scriptstyle \delta(x)}\downarrow & & \downarrow{\scriptstyle \delta(\alpha(\star_A, x))} \\ \alpha(\star_A, x) & \underset{\mathsf{ap}_{\alpha(\star_A,-)}(\delta(x))}{\rightsquigarrow} & \alpha(\star_A, \alpha(\star_A, x)) \end{array}$$

which shows that $\delta(\alpha(\star_A, x)) = \mathsf{ap}_{\alpha(\star_A,-)}(\delta(x))$, which is what we wanted to prove. We define now (in^J_n) and (push^J_n) by

$$\begin{aligned}\mathsf{in}^J_n &: J_nA \to JA, & \mathsf{push}^J_n &: (x : J_nA) \to \mathsf{in}^J_n(x) = \mathsf{in}^J_{n+1}(i_n(x)),\\ \mathsf{in}^J_0(\epsilon) &:= \varepsilon, & \mathsf{push}^J_n(x) &:= \delta(\mathsf{in}^J_n(x)).\\ \mathsf{in}^J_1(a) &:= \alpha(a, \varepsilon),\\ \mathsf{in}^J_{n+2}(i_{n+1}(x)) &:= \alpha(\star_A, \mathsf{in}^J_{n+1}(x)),\\ \mathsf{in}^J_{n+2}(\alpha_{n+1}(a, x)) &:= \alpha(a, \mathsf{in}^J_{n+1}(x)),\\ \mathsf{ap}_{\mathsf{in}^J_{n+2}}(\beta_n(x)) &:= \mathsf{idp},\\ \mathsf{ap}_{\mathsf{in}^J_{n+2}}(\gamma_n(a, x)) &:= \gamma(a, \mathsf{in}^J_n(x)),\\ \mathsf{ap}^2_{\mathsf{in}^J_{n+2}}(\eta_n(x)) &:= \eta(\mathsf{in}^J_n(x)),\end{aligned}$$

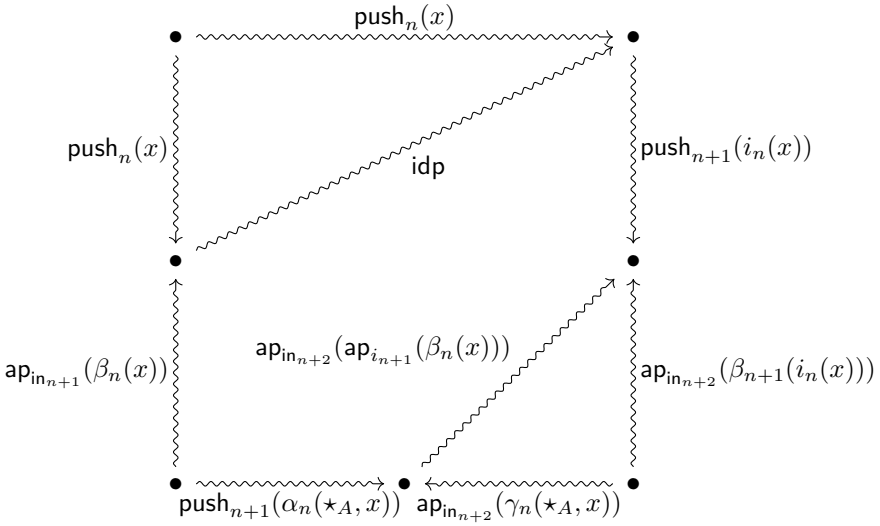
Structure on $J_\infty A$ The equivalent of ε is the term $\varepsilon_\infty := \mathsf{in}_0(\epsilon)$ of type $J_\infty A$. We then define the action of A on $J_\infty A$ by

$$\begin{aligned}\alpha_\infty &: A \to J_\infty A \to J_\infty A,\\ \alpha_\infty(a, \mathsf{in}_n(x)) &:= \mathsf{in}_{n+1}(\alpha_n(a, x)),\\ \mathsf{ap}_{\alpha_\infty(a,-)}(\mathsf{push}_n(x)) &:= \mathsf{push}_{n+1}(\alpha_n(a, x)) \cdot \mathsf{ap}_{\mathsf{in}_{n+2}}(\gamma_n(a, x))^{-1}.\end{aligned}$$

The equivalent of δ is δ_∞ defined by

$$\begin{aligned}\delta_\infty &: (x : J_\infty A) \to x = \alpha_\infty(\star_A, x),\\ \delta_\infty(\mathsf{in}_n(x)) &:= \mathsf{push}_n(x) \cdot \mathsf{ap}_{\mathsf{in}_{n+1}}(\beta_n(x))^{-1},\\ \mathsf{ap}_{\delta_\infty}(\mathsf{push}_n(x)) &:= \delta_\infty^{\mathsf{push}_n}(x),\end{aligned}$$

where $\delta_\infty^{\mathsf{push}_n}(x)$ is the filler of the square



(3.2.5)

where the lower right triangle is filled using $\eta_n(x)$ and the pentagon in the middle is filled using the naturality square of push_{n+1} on $\beta_n(x)$.

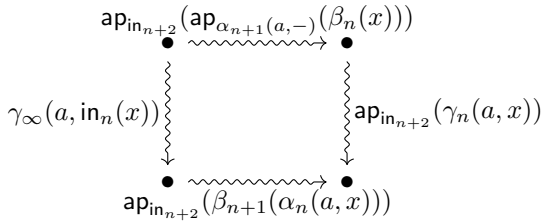
We finally define

$$\begin{aligned}
&\gamma_\infty : (a : A)(x : J_\infty A) \to \alpha_\infty(a, \alpha_\infty(\star_A, x)) = \alpha_\infty(\star_A, \alpha_\infty(a, x)),\\
&\eta_\infty : (x : J_\infty A) \to \gamma_\infty(\star_A, x) = \mathsf{idp}
\end{aligned}$$

in the same way as we defined γ and η, but using δ_∞ instead of δ. In the case of $\gamma_\infty(a, \mathsf{in}_n(x))$ we note that

$$\begin{aligned}
\gamma_\infty(a, \mathsf{in}_n(x)) &= \mathsf{ap}_{\alpha_\infty(a,-)}(\delta_\infty(\mathsf{in}_n(x)))^{-1} \cdot \delta_\infty(\alpha_\infty(a, \mathsf{in}_n(x)))\\
&= \mathsf{ap}_{\alpha_\infty(a,-)}(\mathsf{push}_n(x) \cdot \mathsf{ap}_{\mathsf{in}_{n+1}}(\beta_n(x))^{-1})^{-1} \cdot \delta_\infty(\mathsf{in}_{n+1}(\alpha_n(a, x)))\\
&= (\mathsf{push}_{n+1}(\alpha_n(a, x)) \cdot \mathsf{ap}_{\mathsf{in}_{n+2}}(\gamma_n(a, x))^{-1}\\
&\qquad \cdot \mathsf{ap}_{\mathsf{in}_{n+2}}(\mathsf{ap}_{\alpha_{n+1}(a,-)}(\beta_n(x)))^{-1})^{-1}\\
&\quad \cdot (\mathsf{push}_{n+1}(\alpha_n(a, x)) \cdot \mathsf{ap}_{\mathsf{in}_{n+2}}(\beta_{n+1}(\alpha_n(a, x)))^{-1})\\
&= \mathsf{ap}_{\mathsf{in}_{n+2}}(\mathsf{ap}_{\alpha_{n+1}(a,-)}(\beta_n(x))) \cdot \mathsf{ap}_{\mathsf{in}_{n+2}}(\gamma_n(a, x))\\
&\qquad \cdot \mathsf{ap}_{\mathsf{in}_{n+2}}(\beta_{n+1}(\alpha_n(a, x)))^{-1}.
\end{aligned}$$

Therefore $\gamma_\infty(a, \mathsf{in}_n(x))$ fits in the square

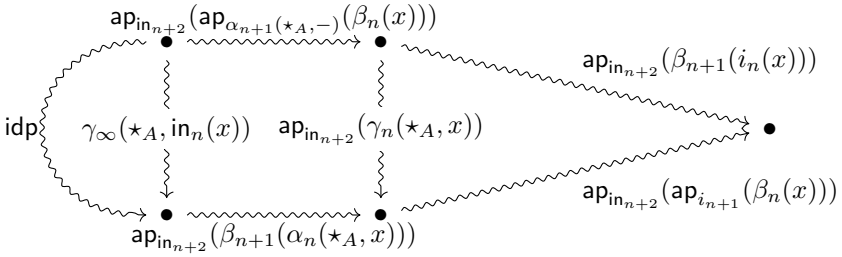


(3.2.6)

For $\eta_\infty(\mathsf{in}_n(x))$, it is defined using $\mathsf{ap}_{\delta_\infty}(\delta_\infty(\mathsf{in}_n(x)))$ and we have

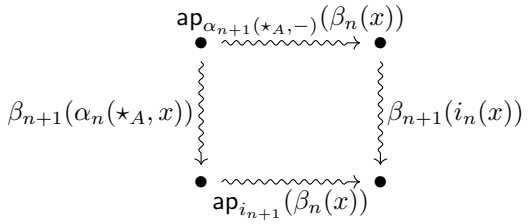
$$\begin{aligned}\mathsf{ap}_{\delta_\infty}(\delta_\infty(\mathsf{in}_n(x))) &= \mathsf{ap}_{\delta_\infty}(\mathsf{push}_n(x) \cdot \mathsf{ap}_{\mathsf{in}_{n+1}}(\beta_n(x)^{-1})) \\ &= \delta_\infty^{\mathsf{push}_n}(x) \cdot \mathsf{ap}_{\lambda x.\, \mathsf{push}_{n+1}(x) \cdot \mathsf{ap}_{\mathsf{in}_{n+2}}(\beta_{n+1}(x)^{-1})}(\beta_n(x)^{-1}).\end{aligned}$$

The $\mathsf{ap}_{\mathsf{push}_{n+1}}(\beta_n(x)^{-1})$ part cancels with the naturality square of push_{n+1} on $\beta_n(x)$ used in $\delta_\infty^{\mathsf{push}_n}(x)$ and the remaining part $\mathsf{ap}_{\mathsf{ap}_{\mathsf{in}_{n+2}}(\beta_{n+1}(-)^{-1})}(\beta_n(x)^{-1})$ is the naturality square of β_{n+1} on $\beta_n(x)$. Therefore $\eta_\infty(\mathsf{in}_n(x))$ fits in the three-dimensional diagram



(3.2.7)

where the half-disc on the left is $\eta_\infty(\mathsf{in}_n(x))$, the square in the middle is square 3.2.6, the triangle on the right is the application of in_{n+2} to $\eta_n(x)$ and the outer diagram is the application of in_{n+2} to the naturality square of β_{n+1} on $\beta_n(x)$, which is



The two maps We can now define the maps back and forth by

$$\begin{aligned} \mathsf{to} : J_\infty A \to JA, \qquad & \mathsf{from} : JA \to J_\infty A, \\ \mathsf{to}(\mathsf{in}_n(x)) := \mathsf{in}_n^J(x), \qquad & \mathsf{from}(\varepsilon) := \varepsilon_\infty, \\ \mathsf{ap}_{\mathsf{to}}\,(\mathsf{push}_n(x)) := \mathsf{push}_n^J(x), \qquad & \mathsf{from}(\alpha(a,x)) := \alpha_\infty(a, \mathsf{from}(x)), \\ & \mathsf{ap}_{\mathsf{from}}(\delta(x)) := \delta_\infty(\mathsf{from}(x)). \end{aligned}$$

First composite Let's prove first that $\mathsf{from} \circ \mathsf{to} = \mathsf{id}_{J_\infty A}$.

By induction on the argument, it is enough to show that for every $n : \mathbb{N}$ and $x : J_n A$, we have $\mathsf{from}(\mathsf{in}_n^J(x)) = \mathsf{in}_n(x)$ and $\mathsf{ap}_{\mathsf{from}}(\mathsf{push}_n^J(x)) = \mathsf{push}_n(x)$ (in the appropriate dependent path type). We proceed by induction on n, and then by induction on x.

- For 0 and ϵ, we have

$$\begin{aligned}\mathsf{from}(\mathsf{in}_0^J(\epsilon)) &= \mathsf{from}(\varepsilon)\\ &= \varepsilon_\infty\\ &= \mathsf{in}_0(\epsilon).\end{aligned}$$

- For 1 and $a : A$, we have

$$\begin{aligned}\mathsf{from}(\mathsf{in}_1^J(a)) &= \mathsf{from}(\alpha(a,\varepsilon))\\ &= \alpha_\infty(a,\mathsf{from}(\varepsilon))\\ &= \mathsf{in}_1(a).\end{aligned}$$

- For $n+2$ and $i_{n+1}(x)$, we have

$$\begin{aligned}\mathsf{from}(\mathsf{in}_{n+2}^J(i_{n+1}(x))) &= \mathsf{from}(\alpha(\star_A,\mathsf{in}_{n+1}^J(x)))\\ &= \alpha_\infty(\star_A,\mathsf{from}(\mathsf{in}_{n+1}^J(x)))\\ &= \alpha_\infty(\star_A,\mathsf{in}_{n+1}(x)) \quad \text{by induction hypothesis}\\ &= \mathsf{in}_{n+2}(\alpha_{n+1}(\star_A,x))\\ &= \mathsf{in}_{n+2}(i_{n+1}(x)) \quad \text{using } \beta_{n+1}(x).\end{aligned}$$

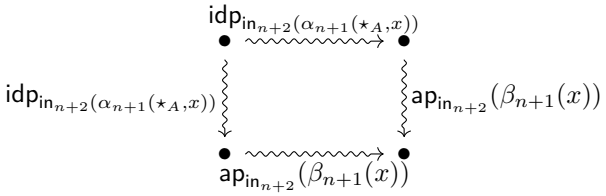
- For $n+2$ and $\alpha_{n+1}(a,x)$, we have

$$\begin{aligned}\mathsf{from}(\mathsf{in}_{n+2}^J(\alpha_{n+1}(a,x))) &= \mathsf{from}(\alpha(a,\mathsf{in}_{n+1}^J(x)))\\ &= \alpha_\infty(a,\mathsf{from}(\mathsf{in}_{n+1}^J(x)))\\ &= \alpha_\infty(a,\mathsf{in}_{n+1}(x)) \quad \text{by induction hypothesis}\\ &= \mathsf{in}_{n+2}(\alpha_{n+1}(a,x)).\end{aligned}$$

- For $n+2$ and $\beta_{n+1}(x)$, we have

$$\begin{aligned}\mathsf{ap}_{\mathsf{from}}(\mathsf{ap}_{\mathsf{in}_{n+2}^J}(\beta_{n+1}(x))) &= \mathsf{ap}_{\mathsf{from}}(\mathsf{idp}_{\alpha(\star_A,\mathsf{in}_{n+1}^J(x))})\\ &= \mathsf{idp}_{\mathsf{from}(\alpha(\star_A,\mathsf{in}_{n+1}^J(x)))}\\ &= \mathsf{idp}_{\alpha_\infty(\star_A,\mathsf{from}(\mathsf{in}_{n+1}^J(x)))}\\ &= \mathsf{idp}_{\alpha_\infty(\star_A,\mathsf{in}_{n+1}(x))} \quad \text{by induction hypothesis}\\ &= \mathsf{idp}_{\mathsf{in}_{n+2}(\alpha_{n+1}(\star_A,x))},\end{aligned}$$

  hence it follows from the fact that the square



can be filled. Here the left side is $\mathsf{ap}_{\mathsf{from}}(\mathsf{ap}_{\mathsf{in}^J_{n+2}}(\beta_{n+1}(x)))$ as computed above, the right side is $\mathsf{ap}_{\mathsf{in}_{n+2}}(\beta_{n+1}(x))$ and the top and bottom side are the equalities $\mathsf{from}(\mathsf{in}^J_{n+2}(x)) = \mathsf{in}_{n+2}(x)$ for $x := \alpha_{n+1}(\star_A, x)$ and $x := i_{n+1}(x)$ respectively (the endpoints of $\beta_{n+1}(x)$) that we constructed above. Note that the paths corresponding to the induction hypothesis do not appear on the diagram because they were taken into account in the computation of $\mathsf{ap}_{\mathsf{from}}(\mathsf{ap}_{\mathsf{in}^J_{n+2}}(\beta_{n+1}(x)))$.

- For $n+2$ and $\gamma_n(a, x)$, we have

$$\begin{aligned}\mathsf{ap}_{\mathsf{from}}(\mathsf{ap}_{\mathsf{in}^J_{n+2}}(\gamma_n(a,x))) &= \mathsf{ap}_{\mathsf{from}}(\gamma(a, \mathsf{in}^J_n(x))) \\ &= \gamma_\infty(a, \mathsf{from}(\mathsf{in}^J_n(x))) \\ &= \gamma_\infty(a, \mathsf{in}_n(x)) \quad \text{by induction hypothesis}\end{aligned}$$

  hence it follows from diagram 3.2.6.

- For $n+2$ and $\eta_n(x)$, we have

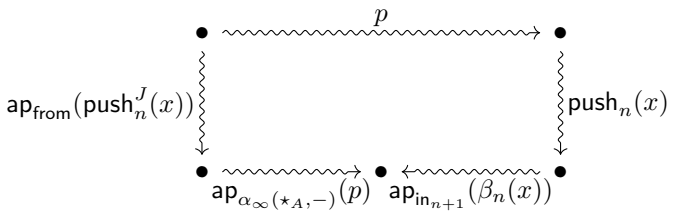
$$\begin{aligned}\mathsf{ap}^2_{\mathsf{from}}(\mathsf{ap}^2_{\mathsf{in}^J_{n+2}}(\eta_n(x))) &= \mathsf{ap}^2_{\mathsf{from}}(\eta(\mathsf{in}^J_n(x))) \\ &= \eta_\infty(\mathsf{from}(\mathsf{in}^J_n(x))) \\ &= \eta_\infty(\mathsf{in}_n(x)) \quad \text{by induction hypothesis}\end{aligned}$$

  hence there is some three-dimensional diagram to fill and it follows from diagram 3.2.7.

We then have to show that for every $n : \mathbb{N}$ and $x : J_n A$, we have an equality between $\mathsf{ap}_{\mathsf{from}}(\mathsf{push}^J_n(x))$ and $\mathsf{push}_n(x)$ along the equalities $\mathsf{from}(\mathsf{in}^J_n(x)) = \mathsf{in}_n(x)$ and $\mathsf{from}(\mathsf{in}^J_{n+1}(i_n(x))) = \mathsf{in}_{n+1}(i_n(x))$ that we just constructed. We have

$$\begin{aligned}\mathsf{ap}_{\mathsf{from}}(\mathsf{push}^J_n(x)) &= \delta_\infty(\mathsf{from}(\mathsf{in}^J_n(x))) \\ &= \delta_\infty(\mathsf{in}_n(x)) \quad \text{by induction hypothesis} \\ &= \mathsf{push}_n(x) \cdot \mathsf{ap}_{\mathsf{in}_{n+1}}(\beta_n(x))^{-1},\end{aligned}$$

hence we have a filler of the square



where p is the equality $\mathsf{from}(\mathsf{in}^J_n(x)) = \mathsf{in}_n(x)$, which concludes this part of the proof.

Second composite Let's now prove that $\mathsf{to} \circ \mathsf{from} = \mathsf{id}_{JA}$.

The idea is very similar, we have to prove that $\mathsf{to}(\varepsilon_\infty) = \varepsilon$ (which is true by definition), that $\mathsf{to}(\alpha_\infty(a, x)) = \alpha(a, \mathsf{to}(x))$, and that $\mathsf{ap}_{\mathsf{to}}(\delta_\infty(x)) = \delta(\mathsf{to}(x))$ along some equalities constructed before. Let's first do the case of α_∞ by induction on x. There are two cases.

- For an element of the form $\mathsf{in}_n(x)$, we have

$$\begin{aligned}\mathsf{to}(\alpha_\infty(a, \mathsf{in}_n(x))) &= \mathsf{to}(\mathsf{in}_{n+1}(\alpha_n(a, x))) \\ &= \alpha(a, \mathsf{in}^J_n(x)) \\ &= \alpha(a, \mathsf{to}(\mathsf{in}_n(x))),\end{aligned}$$

  which is what we wanted.

- For a path of the form $\mathsf{push}_n(x)$, we have

$$\begin{aligned}\mathsf{ap}_{\mathsf{to}}(\mathsf{ap}_{\alpha_\infty(a,-)}(\mathsf{push}_n(x))) &= \mathsf{ap}_{\mathsf{to}}(\mathsf{push}_{n+1}(\alpha_n(a, x)) \cdot \mathsf{ap}_{\mathsf{in}_{n+2}}(\gamma_n(a, x))^{-1}) \\ &= \delta(\mathsf{in}^J_{n+1}(\alpha_n(a, x))) \cdot \gamma(a, \mathsf{in}^J_n(x))^{-1} \\ &= \delta(\alpha(a, \mathsf{in}^J_n(x))) \cdot \gamma(a, \mathsf{in}^J_n(x))^{-1} \\ &= \mathsf{ap}_{\alpha(a,-)}(\delta(\mathsf{in}^J_n(x))) \quad \text{by definition of } \gamma \\ &= \mathsf{ap}_{\alpha(a,-)}(\mathsf{push}^J_n(x)) \\ &= \mathsf{ap}_{\alpha(a,-)}(\mathsf{ap}_{\mathsf{to}}(\mathsf{push}_n(x))),\end{aligned}$$

  which is again what we wanted.

We now prove that the path $\mathsf{ap}_{\mathsf{to}}(\delta_\infty(x)) : \mathsf{to}(x) = \mathsf{to}(\alpha_\infty(\star_A, x))$ composed with the path from $\mathsf{to}(\alpha_\infty(\star_A, x))$ to $\alpha(\star_A, \mathsf{to}(x))$ that we have just constructed is equal to the path $\delta(\mathsf{to}(x)) : \mathsf{to}(x) = \alpha(\star_A, \mathsf{to}(x))$.

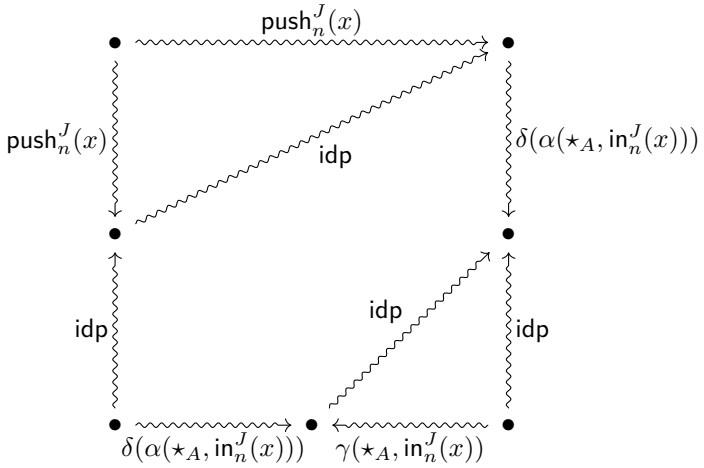
- For an element of the form $\mathsf{in}_n(x)$, we have

$$\begin{aligned}\mathsf{ap}_{\mathsf{to}}(\delta_\infty(\mathsf{in}_n(x))) &= \mathsf{ap}_{\mathsf{to}}(\mathsf{push}_n(x) \cdot \mathsf{ap}_{\mathsf{in}_{n+1}}(\beta_n(x))^{-1}) \\ &= \mathsf{push}^J_n(x) \\ &= \delta(\mathsf{in}^J_n(x)) \\ &= \delta(\mathsf{to}(\mathsf{in}_n(x))),\end{aligned}$$

  which proves the result, as the path from $\mathsf{to}(\alpha_\infty(\star_A, \mathsf{in}_n(x)))$ to $\alpha(\star_A, \mathsf{to}(\mathsf{in}_n(x)))$ is the constant path.

- For a path of the form $\mathsf{push}_n(x)$, we have to compare $\mathsf{ap}^2_{\mathsf{to}}(\mathsf{ap}_{\delta_\infty}(\mathsf{push}_n(x)))$ and

$\mathsf{ap}_\delta(\mathsf{ap}_{\mathsf{to}}(\mathsf{push}_n(x)))$. For the first one, we just apply to to diagram 3.2.5. We obtain



where the bottom right triangle is filled using $\eta(\mathsf{in}_n^J(x))$ and the rest is degenerate. On the other hand, we have

$$\begin{aligned}\mathsf{ap}_\delta(\mathsf{ap}_{\mathsf{to}}(\mathsf{push}_n(x))) &= \mathsf{ap}_\delta(\mathsf{push}_n^J(x)) \\ &= \mathsf{ap}_\delta(\delta(\mathsf{in}_n^J(x)))\end{aligned}$$

and $\eta(\mathsf{in}_n^J(x))$ is defined from $\mathsf{ap}_\delta(\delta(\mathsf{in}_n^J(x)))$ by a coherence operation. Therefore some coherence operation proves that $\mathsf{ap}^2_{\mathsf{to}}(\mathsf{ap}_{\delta_\infty}(\mathsf{push}_n(x)))$ and $\mathsf{ap}_\delta(\mathsf{ap}_{\mathsf{to}}(\mathsf{push}_n(x)))$ are equal.

This concludes the proof that $J_\infty A$ is equivalent to JA.

3.2.3 Equivalence between JA and $\Omega\Sigma A$

We now prove that JA is equivalent to $\Omega\Sigma A$. The function δ shows that the map $\alpha(\star_A, -)$ is equal to the identity function, hence $\alpha(\star_A, -)$ is an equivalence. Given that A is 0-connected it follows that $\alpha(a, -)$ is an equivalence for every a. We define $F : \Sigma A \to \mathsf{Type}$ by

$$\begin{aligned}F(\mathsf{north}) &:= JA, \\ F(\mathsf{south}) &:= JA, \\ \mathsf{ap}_F(\mathsf{merid}\, a) &:= \mathsf{ua}(\alpha(a, -)),\end{aligned}$$

which is a valid definition because $\alpha(a, -)$ is an equivalence.

We now prove that the total space of F is contractible. According to the flattening lemma (proposition 1.9.2), the total space of F is equivalent to the type

$$T := JA \sqcup^{A \times JA} JA,$$

where the two maps $A \times JA \to JA$ are snd and α respectively. We want to construct, for every $x : T$, a path $p(x)$ from $\mathsf{inl}(\varepsilon)$ to x.

- For $\mathsf{inl}(\varepsilon)$, we take the constant path $\mathsf{idp}_{\mathsf{inl}(\varepsilon)}$
- For an element of the form $\mathsf{inl}(\alpha(a,x))$, we take the composition

$$\begin{array}{ccc}
\mathsf{inl}(\alpha(a,x)) & \overset{\mathsf{push}(\star_A,\alpha(a,x))}{\rightsquigarrow} & \mathsf{inr}(\alpha(\star_A,\alpha(a,x))) \\
 & & \Big\downarrow \mathsf{ap}_{\mathsf{inr}}(\delta(\alpha(a,x))) \\
\mathsf{inl}(x) & \overset{\mathsf{push}(a,x)}{\rightsquigarrow} & \mathsf{inr}(\alpha(a,x)) \\
\Big\downarrow p(\mathsf{inl}(x)) & & \\
\mathsf{inl}(\varepsilon). & &
\end{array}$$

- For a path of the form $\mathsf{ap}_{\mathsf{inl}}(\delta(x))$, we need to fill the diagram

$$\begin{array}{ccc}
\mathsf{inl}(\alpha(\star_A,x)) & \overset{\mathsf{push}(\star_A,\alpha(\star_A,x))}{\rightsquigarrow} & \mathsf{inr}(\alpha(\star_A,\alpha(\star_A,x))) \\
\Big\downarrow \mathsf{ap}_{\mathsf{inl}}(\delta(x)) & & \Big\downarrow \mathsf{ap}_{\mathsf{inr}}(\delta(\alpha(\star_A,x))) \\
\mathsf{inl}(x) & \underset{\mathsf{push}(\star_A,x)}{\rightsquigarrow} & \mathsf{inr}(\alpha(\star_A,x))
\end{array}$$

  By naturality of $\mathsf{push}(\star_A,-)$ on the path $\delta(x)$, we get a filler of the similar diagram which has $\mathsf{ap}_{\mathsf{inr}}(\mathsf{ap}_{\alpha(\star_A,-)}(\delta(x)))$ on the right side. Moreover, we know that $\mathsf{ap}_{\mathsf{inr}}(\mathsf{ap}_{\alpha(\star_A,-)}(\delta(x)))$ and $\mathsf{ap}_{\mathsf{inr}}(\delta(\alpha(\star_A,x)))$ are equal via $\eta(x)$, which concludes.

- For a point of the form $\mathsf{inr}(x)$, we take the composition

$$\mathsf{inr}(x) \overset{\mathsf{ap}_{\mathsf{inr}}(\delta(x))}{\rightsquigarrow} \mathsf{inr}(\alpha(\star_A,x)) \overset{\mathsf{push}(\star_A,x)}{\rightsquigarrow} \mathsf{inl}(x) \overset{p(\mathsf{inl}(x))}{\rightsquigarrow} \mathsf{inl}(\varepsilon).$$

- Finally for a path of the form $\mathsf{push}(a,x)$, it is enough to notice that the path from $\mathsf{inr}(\alpha(a,x))$ to $\mathsf{inl}(\varepsilon)$ constructed as above is equal to the composition

$$\mathsf{inr}(\alpha(a,x)) \overset{\mathsf{push}(a,x)}{\rightsquigarrow} \mathsf{inl}(x) \overset{p(\mathsf{inl}(x))}{\rightsquigarrow} \mathsf{inl}(\varepsilon).$$

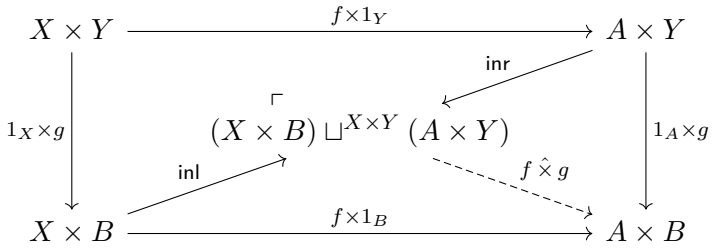
This concludes the proof that T is contractible, and by the same argument as for $\Omega\mathbb{S}^1$ we obtain that $\Omega\Sigma A$ is equivalent to $F(\star_{\Sigma A})$, which is equal to JA by definition.

3.2.4 Connectivity of the maps i_n

Given two maps $f : X \to A$ and $g : Y \to B$, the *pushout-product* of f and g is the map

$$\begin{aligned}
f \mathbin{\hat{\times}} g &: (X \times B) \sqcup^{X\times Y} (A \times Y) \to (A \times B), \\
(f \mathbin{\hat{\times}} g)(\mathsf{inl}(x,b)) &:= (f(x), b), \\
(f \mathbin{\hat{\times}} g)(\mathsf{inr}(a,y)) &:= (a, g(y)), \\
\mathsf{ap}_{f \mathbin{\hat{\times}} g}(\mathsf{push}(x,y)) &:= \mathsf{idp}_{(f(x),g(y))}.
\end{aligned}$$

We have the commutative square



The main result of this subsection is the following.

Proposition 3.2.8. *If f is m-connected and g is n-connected, then $f \hat{\times} g$ is $(m+n+2)$-connected.*

Proof. Let's first recall the following proposition which is a generalization of proposition 2.3.8. It is proved in [UF13, lemma 8.6.1].

Proposition 3.2.9. *If $f : A \to B$ is n-connected and $P : B \to \mathsf{Type}$ is a family of $(n+k)$-types, then the map*

$$\lambda s.s \circ f : \prod_{b:B} P(b) \to \prod_{a:A} P(f(a))$$

is $(k-2)$-truncated (in the sense that all its fibers are $(k-2)$-truncated).

We now prove that $f \hat{\times} g$ is $(m+n+2)$-connected. We consider $P : A \times B \to \mathsf{Type}$ a family of $(n+m+2)$-types together with

$$k : (u : (X \times B) \sqcup^{X \times Y} (A \times Y)) \to P((f \hat{\times} g)(u)).$$

By splitting k in three parts and currying, we have to prove the following proposition.

Proposition 3.2.10. *Suppose we have $P : A \to B \to \mathsf{Type}$ a family of $(n+m+2)$-types together with*

$$\begin{aligned}
&u : (x : X)(b : B) \to P(f(x), b),\\
&v : (a : A)(y : Y) \to P(a, g(y)),\\
&w : (x : X)(y : Y) \to u(x, g(y)) =_{P(f(x),g(y))} v(f(x), y).
\end{aligned}$$

Then there exists a map

$$h : (a : A)(b : B) \to P(a, b)$$

together with homotopies

$$\begin{aligned}
&p : (x : X)(b : B) \to h(f(x), b) = u(x, b),\\
&q : (a : A)(y : Y) \to h(a, g(y)) = v(a, y),\\
&r : (x : X)(y : Y) \to p(x, g(y))^{-1} \cdot q(f(x), y) = w(x, y).
\end{aligned}$$

Proof. Let's define $F : A \to \mathsf{Type}$ by

$$F(a) := \sum_{k:(b:B)\to P(a,b)} ((y : Y) \to k(g(y)) = v(a, y)).$$

For a given $a : A$, the type $F(a)$ is the fiber of the map

$$\lambda s.s \circ g : \prod_{b:B} P(a, b) \to \prod_{y:Y} P(a, g(y))$$

at $v(a, -)$. Given that g is n-connected and P is a family of $(n+m+2)$-truncated types, proposition 3.2.9 shows that $F(a)$ is m-truncated.

For every $x : X$ we have an element of $F(f(x))$ given by $(u(x, -), w(x, -))$. Hence, using the fact that f is m-connected, there is a map $k : (a : A) \to F(a)$ together with a homotopy φ between $k \circ f$ and $\lambda x.(u(x, -), w(x, -))$. We can now define h, p, q, and r by

$$\begin{aligned} h(a, b) &:= \mathsf{fst}(k(a))(b), \\ p(x, b) &:= \mathsf{fst}(\varphi(x))(b), \\ q(a, y) &:= \mathsf{snd}(k(a))(y), \\ r(x, y) &:= \mathsf{snd}(\varphi(x))(y). \end{aligned}$$ □

This concludes the proof that $f \hat{\times} g$ is $(n + m + 2)$-connected. □

We can now compute the connectivity of the maps i_n.

Proposition 3.2.11. *For every $n : \mathbb{N}$, the map i_n is $(n(k + 1) + (k - 1))$-connected.*

Proof. We proceed by induction on n. For 0, the map i_0 is the inclusion of the basepoint of A, hence i_0 is $(k - 1)$-connected because A is k-connected.

For $n + 1$, the map f in diagram 3.2.4 (the diagram defining $J_{n+2}A$) is the pushout-product of i_n and of the map $\mathbf{1} \to A$ (which is $(k - 1)$-connected). Hence f is $((n + 1)(k+1)+(k-1))$-connected by proposition 3.2.8. Therefore, by proposition 2.3.10, the map i_{n+1} is $((n + 1)(k + 1) + (k - 1))$-connected as well. □

This concludes the James construction.

3.3 Whitehead products

In proposition 3.3.1 we give a decomposition of a product of spheres into a pushout of spheres. This allows us to define Whitehead products, which are used in the next section in the definition of the natural number n such that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$.

Proposition 3.3.1. *Given $n, m : \mathbb{N}^*$, there is a map $W_{n,m} : \mathbb{S}^{n+m-1} \to \mathbb{S}^n \vee \mathbb{S}^m$ such that*

$$\mathbb{S}^n \times \mathbb{S}^m \simeq 1 \sqcup^{\mathbb{S}^{n+m-1}} (\mathbb{S}^n \vee \mathbb{S}^m)$$

and such that the induced map $\mathbb{S}^n \vee \mathbb{S}^m \to \mathbb{S}^n \times \mathbb{S}^m$ is $i^{\vee}_{\mathbb{S}^n,\mathbb{S}^m}$.

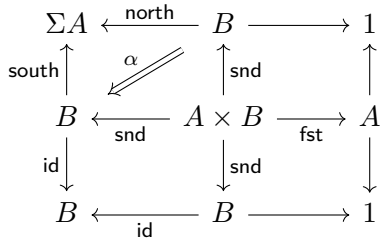
We first prove the following more general version which isn't more complicated to prove.

Proposition 3.3.2. *Given two types A and B, there is a map $W_{A,B} : A * B \to \Sigma A \vee \Sigma B$ such that*

$$\Sigma A \times \Sigma B \simeq 1 \sqcup^{A*B} (\Sigma A \vee \Sigma B)$$

and such that the induced map $\Sigma A \vee \Sigma B \to \Sigma A \times \Sigma B$ is $i^{\vee}_{\Sigma A, \Sigma B}$.

Proof. We use the 3×3-lemma with the diagram



where $\alpha : A \times B \to \mathsf{north} =_{\Sigma A} \mathsf{south}$ is defined by $\alpha(x, y) := \mathsf{merid}(x)$.

The pushout of the top row is equivalent to $\Sigma A \vee \Sigma B$, the pushout of the middle row is equivalent to the join $A * B$ and the pushout of the bottom row is contractible, so the pushout of the pushouts of the rows is equivalent to $1 \sqcup^{A*B} (\Sigma A \vee \Sigma B)$ for the map $A * B \to \Sigma A \vee \Sigma B$ defined by

$$\begin{aligned} W_{A,B} &: A * B \to \Sigma A \vee \Sigma B, \\ W_{A,B}(\mathsf{inl}(a)) &:= \mathsf{inr}(\mathsf{north}), \\ W_{A,B}(\mathsf{inr}(b)) &:= \mathsf{inl}(\mathsf{north}), \\ \mathsf{ap}_{W_{A,B}}(\mathsf{push}(a, b)) &:= \mathsf{ap}_{\mathsf{inr}}(\varphi_B(b)) \cdot \mathsf{push}(\star_{\mathbf{1}}) \cdot \mathsf{ap}_{\mathsf{inl}}(\varphi_A(a)). \end{aligned}$$

The pushouts of the left and of the right columns are both equivalent to ΣA, and the pushout of the middle column is equivalent to $\Sigma A \times B$. Moreover, the horizontal map on the left between $\Sigma A \times B$ and ΣA is equal to fst, as can be proved by induction using the definition of α.The horizontal map on the right is also equal to fst. Hence the pushout of the pushout of the columns is equivalent to $\Sigma A \times \Sigma B$. Therefore we have

$$\Sigma A \times \Sigma B \simeq 1 \sqcup^{A*B} (\Sigma A \vee \Sigma B)$$

and it can be checked that the induced map $\Sigma A \vee \Sigma B \to \Sigma A \times \Sigma B$ is equal to $i^{\vee}_{\Sigma A, \Sigma B}$. □

Proof of proposition 3.3.1. We apply proposition 3.3.2 to $A := \mathbb{S}^{n-1}$ and $B := \mathbb{S}^{m-1}$, and we obtain

$$\mathbb{S}^n \times \mathbb{S}^m \simeq 1 \sqcup^{\mathbb{S}^{n-1} * \mathbb{S}^{m-1}} (\mathbb{S}^n \vee \mathbb{S}^m).$$

Moreover, we have $\mathbb{S}^{n-1} * \mathbb{S}^{m-1} \simeq \mathbb{S}^{n+m-1}$ by proposition 1.8.8, which concludes. □

This allows us to define the following operation on homotopy groups.

Definition 3.3.3. Given a pointed type X and two positive integers n and m, the *Whitehead product* is the function

$$[-,-] : \pi_n(X) \times \pi_m(X) \to \pi_{n+m-1}(X)$$

defined by composition with $W_{n,m}$ when representing elements of homotopy groups as maps from the spheres.

3.4 Application to homotopy groups of spheres

The sphere $\mathbb{S}^n$ is $(n-1)$-connected, therefore by the Freudenthal suspension theorem 3.2.2, the map $\varphi_{\mathbb{S}^n} : \mathbb{S}^n \to \Omega\mathbb{S}^{n+1}$ is $(2n-2)$-connected. On homotopy groups it gives the following result.

Proposition 3.4.1. *For $k, n : \mathbb{N}$, the map $\pi_{n+k}(\mathbb{S}^n) \to \pi_{n+k+1}(\mathbb{S}^{n+1})$ is an isomorphism if $n \geq k+2$ and surjective if $n = k+1$.*

On the table 2.1 of homotopy groups of spheres this corresponds to the fact that the diagonals $\pi_{n+k}(\mathbb{S}^n)$ (for a fixed k) are constant above the zigzag line. In particular, for $k = 0$ and $k = 1$ we have the following result. I already obtained the result for $\pi_n(\mathbb{S}^n)$ in collaboration with Dan Licata in [LB13] but with a different technique.

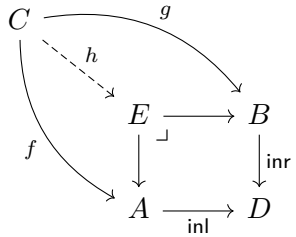
Corollary 3.4.2. *For $n \geq 2$ we have $\pi_n(\mathbb{S}^n) \simeq \pi_2(\mathbb{S}^2) \simeq \mathbb{Z}$. For $n \geq 3$ we have $\pi_{n+1}(\mathbb{S}^n) \simeq \pi_4(\mathbb{S}^3)$ and the map $\pi_3(\mathbb{S}^2) \to \pi_4(\mathbb{S}^3)$ is surjective.*

Note that as we are working constructively this does *not* imply that $\pi_4(\mathbb{S}^3)$ is of the form $\mathbb{Z}/n\mathbb{Z}$ for some $n : \mathbb{Z}$. Indeed, it cannot be proved constructively that every subgroup of $\mathbb{Z}$ is of the form $n\mathbb{Z}$ for some $n : \mathbb{Z}$, as there is no way to compute this n in general. In this case, however, we can use the James construction to give an explicit definition of the kernel of that map. We will need the Blakers–Massey theorem ([Hou+16]), which is stated as [UF13, theorem 8.10.2] and which is proved in Agda in [Hou13].

Proposition 3.4.3 (Blakers–Massey theorem)**.** *Given two maps $f : C \to A$ and $g : C \to B$, we consider $D := A \sqcup^C B$,*

$$E := \sum_{a:A} \sum_{b:B} (\mathsf{inl}(a) =_D \mathsf{inr}(b)),$$

and $h : C \to E$ defined by $h(c) := (f(c), g(c), \mathsf{push}(c))$.



If f is n-connected and g is m-connected, then h is $(n+m)$-connected.

We now prove the following proposition.

Proposition 3.4.4. *For $n \geq 2$, the map $\pi_{2n-1}(\mathbb{S}^n) \to \pi_{2n}(\mathbb{S}^{n+1})$ induced by $\varphi_{\mathbb{S}^n}$ is surjective and its kernel is generated by the Whitehead product $[i_n, i_n]$, where i_n is the generator of $\pi_n(\mathbb{S}^n)$.*

Proof. Applying corollary 3.2.3 to $\mathbb{S}^n$ which is $(n-1)$-connected, we get a $(3n-2)$-connected map from $J_2(\mathbb{S}^n)$ to $\Omega\mathbb{S}^{n+1}$. In particular, given that $2n-1 < 3n-2$, it means that

$$\pi_{2n-1}(J_2(\mathbb{S}^n)) \simeq \pi_{2n-1}(\Omega\mathbb{S}^{n+1}) \simeq \pi_{2n}(\mathbb{S}^{n+1}),$$

so we now study the map $\pi_{2n-1}(\mathbb{S}^n) \to \pi_{2n-1}(J_2(\mathbb{S}^n))$. We know from the James construction that

$$J_2(\mathbb{S}^n) \simeq (\mathbb{S}^n \times \mathbb{S}^n) \sqcup^{\mathbb{S}^n \vee \mathbb{S}^n} \mathbb{S}^n,$$

hence using the decomposition of $\mathbb{S}^n \times \mathbb{S}^n$ given in proposition 3.3.1, we get

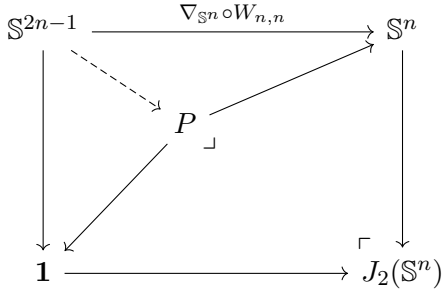
$$J_2(\mathbb{S}^n) \simeq (\mathbf{1} \sqcup^{\mathbb{S}^{2n-1}} (\mathbb{S}^n \vee \mathbb{S}^n)) \sqcup^{\mathbb{S}^n \vee \mathbb{S}^n} \mathbb{S}^n$$

where the map from $\mathbb{S}^n \vee \mathbb{S}^n$ to the pushout on the left is inr (i.e. it's the identity on the second component). Using proposition 1.8.9, this reduces to

$$J_2(\mathbb{S}^n) \simeq \mathbf{1} \sqcup^{\mathbb{S}^{2n-1}} \mathbb{S}^n,$$

where the map $\mathbb{S}^{2n-1} \to \mathbb{S}^n$ is the Whitehead map $W_{n,n} : \mathbb{S}^{2n-1} \to \mathbb{S}^n \vee \mathbb{S}^n$ composed with the folding map $\nabla_{\mathbb{S}^n} : \mathbb{S}^n \vee \mathbb{S}^n \to \mathbb{S}^n$.

We now take the fiber P of the map $\mathbb{S}^n \to J_2(\mathbb{S}^n)$, which is the pullback of the two maps from $\mathbb{S}^n$ and $\mathbf{1}$ to $J_2(\mathbb{S}^n)$



The relevant part of the long exact sequence of homotopy groups for $P \to \mathbb{S}^n \to J_2(\mathbb{S}^n)$ is

$$\pi_{2n-1}(P) \longrightarrow \pi_{2n-1}(\mathbb{S}^n) \longrightarrow \pi_{2n-1}(J_2(\mathbb{S}^n)) \longrightarrow \pi_{2n-2}(P).$$

The map from $\mathbb{S}^{2n-1}$ to $\mathbf{1}$ is $(2n-2)$-connected and the map from $\mathbb{S}^{2n-1}$ to $\mathbb{S}^n$ is $(n-2)$-connected (indeed, every map between two $(n-1)$-connected types is $(n-2)$-connected), hence using the Blakers-Massey theorem, the dashed map from $\mathbb{S}^{2n-1}$ to P is $(3n-4)$-connected. Given that $2n-2 \leq 3n-4$, it follows that $\pi_{2n-2}(P) \simeq \pi_{2n-2}(\mathbb{S}^{2n-1}) \simeq 0$.

Hence $\pi_{2n-1}(J_2(\mathbb{S}^n))$ is the quotient of $\pi_{2n-1}(\mathbb{S}^n)$ by the image of the map $\pi_{2n-1}(P) \to \pi_{2n-1}(\mathbb{S}^n)$. But the dashed map is surjective on π_{2n-1}, so it's the same as the image of the map $\pi_{2n-1}(\mathbb{S}^{2n-1}) \to \pi_{2n-1}(\mathbb{S}^n)$, which is generated by $[i_n, i_n]$, by definition of the Whitehead product.

Therefore, the kernel of the map $\pi_{2n-1}(\mathbb{S}^n) \to \pi_{2n}(\mathbb{S}^{n+1})$ is generated by $[i_n, i_n]$. □

In particular, applying this result to $n = 2$ and using the fact that $\pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$, we get the following corollary.

Corollary 3.4.5. *We have*

$$\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z},$$

where n is the absolute value of the image of $[i_2, i_2]$ by the equivalence $\pi_3(\mathbb{S}^2) \xrightarrow{\sim} \mathbb{Z}$.

This result is quite remarkable in that even though it is a constructive proof, it is not at all obvious how to actually compute this n. At the time of writing, we still haven't managed to extract its value from its definition. A complete and concise definition of this number n is presented in appendix B, for the benefit of someone wanting to implement it in a prospective proof assistant. In the rest of this thesis, we give a mathematical proof in homotopy type theory that $n = 2$.

Chapter 4

Smash products of spheres

In this chapter we prove various properties of the smash product, in particular concerning its symmetric monoidal structure, the smash product of spheres, and the connectedness of the smash product of two maps. The results of this chapter are essential for the definition and the basic properties of the multiplicative structure on cohomology, which we will see in the next chapter.

We recall that the *smash product* of two pointed types A and B is the pushout of the span

$$\mathbf{1} \longleftarrow A \vee B \xrightarrow{i^{\vee}_{A,B}} A \times B.$$

The smash product $A \wedge B$ is pointed by $\star_{A\wedge B} := \mathsf{inl}(\star_{\mathbf{1}})$ and there is a map

$$\begin{aligned} \mathsf{proj} &: A \to B \to A \wedge B, \\ \mathsf{proj}(a,b) &:= \mathsf{inr}((a,b)) \end{aligned}$$

together with two equalities

$$\begin{aligned} \mathsf{proj_r} &: (a:A) \to \mathsf{proj}(a,\star_B) = \star_{A\wedge B}, & \mathsf{proj_l} &: (b:B) \to \mathsf{proj}(\star_A,b) = \star_{A\wedge B}, \\ \mathsf{proj_r}(a) &:= \mathsf{push}(\mathsf{inl}(a)), & \mathsf{proj_l}(b) &:= \mathsf{push}(\mathsf{inr}(b)) \end{aligned}$$

and a 2-dimensional path

$$\begin{aligned} \mathsf{proj_{rl}} &: \mathsf{proj_r}(\star_A) = \mathsf{proj_l}(\star_B), \\ \mathsf{proj_{rl}} &:= \mathsf{ap}_{\mathsf{push}}(\mathsf{push}(\star_{\mathbf{1}})). \end{aligned}$$

In particular, we can also see $A \wedge B$ as the higher inductive type with constructors $\star_{A\wedge B}$, proj, $\mathsf{proj_r}$, $\mathsf{proj_l}$ and $\mathsf{proj_{rl}}$.

4.1 The monoidal structure of the smash product

Let's first define the notion of 1-coherent symmetric monoidal product on pointed types.

Definition 4.1.1. A 1-*coherent symmetric monoidal product on pointed types* is a binary operation $\otimes$ between pointed types equipped with

- a pointed map $f \otimes g : A \otimes B \to A' \otimes B'$ given any two pointed maps $f : A \to A'$ and $g : B \to B'$,
- a pointed equality $\mathsf{id}_A \otimes \mathsf{id}_B = \mathsf{id}_{A \otimes B}$ given any two pointed types A and B,
- a pointed equality $(f' \circ f) \otimes (g' \circ g) = (f' \otimes g') \circ (f \otimes g)$ given any four pointed maps $f : A \to A'$, $f' : A' \to A''$, $g : B \to B'$ and $g' : B' \to B''$,
- a pointed equivalence $\alpha_{A,B,C} : (A \otimes B) \otimes C \xrightarrow{\sim} A \otimes (B \otimes C)$ given any three pointed types A, B and C, and a pointed filler of the square

$$\begin{array}{ccc} (A \otimes B) \otimes C & \xrightarrow{\alpha_{A,B,C}} & A \otimes (B \otimes C) \\ {\scriptstyle (f \otimes g) \otimes h}\downarrow & & \downarrow{\scriptstyle f \otimes (g \otimes h)} \\ (A' \otimes B') \otimes C' & \xrightarrow[\alpha_{A',B',C'}]{} & A' \otimes (B' \otimes C') \end{array}$$

  given any three pointed maps $f : A \to A'$, $g : B \to B'$ and $h : C \to C'$,
- a pointed filler of the pentagon

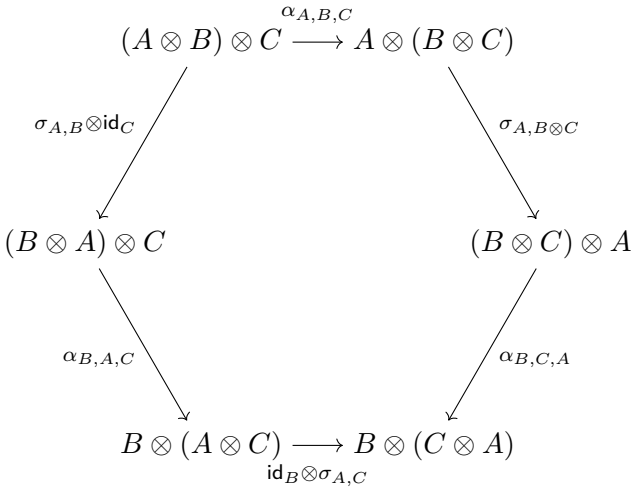
$$\begin{array}{ccccc} & & ((A \otimes B) \otimes C) \otimes D & & \\ & {\scriptstyle \alpha_{A,B,C} \otimes \mathsf{id}_D} \swarrow & & \searrow {\scriptstyle \alpha_{A \otimes B,C,D}} & \\ (A \otimes (B \otimes C)) \otimes D & & & & (A \otimes B) \otimes (C \otimes D) \\ {\scriptstyle \alpha_{A,B \otimes C,D}}\downarrow & & & & \downarrow{\scriptstyle \alpha_{A,B,C \otimes D}} \\ A \otimes ((B \otimes C) \otimes D) & & \xrightarrow[\mathsf{id}_A \otimes \alpha_{B,C,D}]{} & & A \otimes (B \otimes (C \otimes D)) \end{array}$$

  given any four pointed types A, B, C and D,
- a pointed map $\sigma_{A,B} : A \otimes B \to B \otimes A$ (which will turn out to be an equivalence) given any two pointed types A and B, and a pointed filler of the square

$$\begin{array}{ccc} A \otimes B & \xrightarrow{\sigma_{A,B}} & B \otimes A \\ {\scriptstyle f \otimes g}\downarrow & & \downarrow{\scriptstyle g \otimes f} \\ A' \otimes B' & \xrightarrow{\sigma_{A',B'}} & B' \otimes A' \end{array}$$

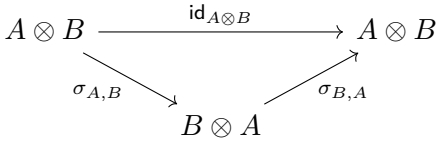
  given any two pointed maps $f : A \to A'$ and $g : B \to B'$,

- a pointed filler of the hexagon



  given any three pointed types A, B and C,

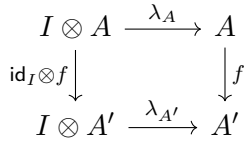
- a pointed filler of the triangle



  given any two pointed types A and B,

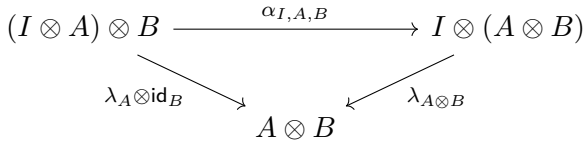
- a pointed type $I : \mathsf{Type}$,

- a pointed equivalence $\lambda_A : I \otimes A \xrightarrow{\sim} A$ given any pointed type A and a pointed filler of the square



  given any pointed map $f : A \to A'$,

- a pointed filler of the triangle



  given any two pointed types A and B.

We call this a 1-*coherent* symmetric monoidal structure because we do not ask the fillers of the diagrams to satisfy any further coherence condition. It's an open question to give a definition in homotopy type theory of the notion of fully coherent (or even only n-coherent) symmetric monoidal structure, but here we only need the 1-coherent structure of the smash product. The following result is the main result of this section even though we essentially admit it.

Proposition 4.1.2. *The smash product is a* 1*-coherent symmetric monoidal product on pointed types.*

Sketch of proof. Putting the unit aside for a moment, we have to define six functions of the form

$$(x : A \wedge B) \to P(x),$$

four of the form

$$(x : (A \wedge B) \wedge C) \to P(x),$$

two of the form

$$(x : A \wedge (B \wedge C)) \to P(x),$$

and one of the form

$$(x : ((A \wedge B) \wedge C) \wedge D) \to P(x),$$

where each time $P(x)$ is either a smash product like $B \wedge A$ or $A \wedge (B \wedge C)$, or an equality in a smash product between combinations of some of those functions.

The idea is that the smash product $A \wedge B$ can be seen as the product $A \times B$ where all elements of the form $(a, \star_B)$ and $(\star_A, b)$ have been identified together. Therefore, in order to define a map out of $A \wedge B$ it should be enough to define it on elements of the form $\mathsf{proj}(a, b)$ in such a way that the image of elements of the form $\mathsf{proj}(a, \star_B)$ and $\mathsf{proj}(\star_A, b)$ are identified to the basepoint in the codomain. But it is not enough to simply give paths from the images of $\mathsf{proj}(a, \star_B)$ and $\mathsf{proj}(\star_A, b)$ to the basepoint of the codomain, because we also need to check that the two induced paths from the image of $\mathsf{proj}(\star_A, \star_B)$ to the basepoint are equal, and this is often the most technical part. Intuitively, however, the idea (that we do not make precise here) is the following. In the type $A \wedge B$, the "subtype" generated (using coherence operations) by $\star_{A\wedge B}$, all $\mathsf{proj}(a, \star_B)$, all $\mathsf{proj}(\star_A, b)$, all $\mathsf{proj_r}(a)$, all $\mathsf{proj_l}(b)$, and $\mathsf{proj_{rl}}$ is "contractible" in the sense that there is a path between any two points, a 2-dimensional path between any two parallel paths, and so on. Similarly, the subtype of $(A \wedge B) \wedge C$ generated by all the $\mathsf{proj}(\mathsf{proj}(a, b), c)$ where either a, b, or c is the basepoint and by all the combinations of $\mathsf{proj_r}$, $\mathsf{proj_l}$, and $\mathsf{proj_{rl}}$ that one can write is contractible in the same sense. Therefore, in order to define (say) a map from $(A \wedge B) \wedge C$ to $A \wedge (B \wedge C)$, it is enough to define it on elements of the form $\mathsf{proj}(\mathsf{proj}(a, b), c)$ in such a way that when either a, b, or c is the basepoint, the image is in the contractible part of $A \wedge (B \wedge C)$. The contractibility takes care of all the other constructors.

For instance, given two pointed maps $f : A \to A'$ and $g : B \to B'$, we define their smash product by

$$f \wedge g : A \wedge B \to A' \wedge B',$$
$$(f \wedge g)(\mathsf{proj}(x, y)) := \mathsf{proj}(f(x), g(y)).$$

The images of the other constructors is as follows. For $\star_{A\wedge B}$ we just take $\star_{A'\wedge B'}$. For $\mathsf{proj_r}(x)$ we need to give a path from $\mathsf{proj}(f(x), g(\star_B))$ to $\star_{A'\wedge B'}$ and the most obvious choice is

$$\mathsf{ap}_{\mathsf{proj}(f(x),-)}(\star_g) \cdot \mathsf{proj_r}(f(x))$$

and similarly for $\mathsf{proj_l}(y)$. For $\mathsf{proj_{rl}}$ we need to fill the square

$$\begin{array}{ccc} \mathsf{proj}(f(\star_A), g(\star_B)) & \rightsquigarrow & \mathsf{proj}(f(\star_A), \star_{B'}) \\ \vdots & & \vdots \\ \mathsf{proj}(\star_{A'}, g(\star_B)) & \rightsquigarrow & \star_{A'\wedge B'} \end{array}$$

and this is done by combining the two squares

$$\begin{array}{ccc} \mathsf{proj}(f(\star_A), g(\star_B)) & \rightsquigarrow & \mathsf{proj}(f(\star_A), \star_{B'}) \\ \vdots & & \vdots \\ \mathsf{proj}(\star_{A'}, g(\star_B)) & \rightsquigarrow & \mathsf{proj}(\star_{A'}, \star_{B'}) \end{array} \qquad \begin{array}{ccc} \mathsf{proj}(\star_{A'}, \star_{B'}) & \rightsquigarrow & \mathsf{proj}(f(\star_A), \star_{B'}) \\ \vdots & \mathsf{proj_r}(\star_{A'}) \searrow & \vdots \\ \mathsf{proj}(\star_{A'}, g(\star_B)) & \mathsf{proj_l}(\star_{B'}) \searrow & \star_{A'\wedge B'} \end{array}$$

where the first one and the two triangles of the second one are filled by naturality squares and the oval shape in the middle of the second one is filled by $\mathsf{proj_{rl}}$.

Similarly, we define associativity by

$$\alpha_{A,B,C} : (A \wedge B) \wedge C \to A \wedge (B \wedge C),$$
$$\alpha_{A,B,C}(\mathsf{proj}(\mathsf{proj}(a, b), c)) := \mathsf{proj}(a, \mathsf{proj}(b, c)),$$

and all generating paths and higher-dimensional paths on the left are sent to appropriate paths and higher-dimensional paths on the right. It works because the image of $\mathsf{proj}(\mathsf{proj}(a, b), c)$ where either a, b or c is the basepoint is in the contractible part of $A \wedge (B \wedge C)$. The commutator $\sigma_{A,B}$ is defined analogously by

$$\sigma_{A,B} : A \wedge B \to B \wedge A,$$
$$\sigma_{A,B}(\mathsf{proj}(a, b)) := \mathsf{proj}(b, a).$$

For functoriality, naturality of associativity, naturality of commutativity, the pentagon, the hexagon, and the triangle, the idea is that we can easily check by hand that it holds by definition for elements of the form $\mathsf{proj}(a, b)$ (or $\mathsf{proj}(\mathsf{proj}(a, b), c)$ or $\mathsf{proj}(\mathsf{proj}(\mathsf{proj}(a, b), c), d)$), therefore by a similar argument of contractibility, they hold also for an arbitrary element of type $A \wedge B$ (or $(A \wedge B) \wedge C$ or $((A \wedge B) \wedge C) \wedge D$).

Finally the unit is $I := \mathbf{2}$, the unitor and its inverse are defined by

$$\begin{aligned} \lambda_A &: \mathbf{2} \wedge A \to A, & \lambda_A^{-1} &: A \to \mathbf{2} \wedge A, \\ \lambda_A(\mathsf{true}, a) &:= \star_A, & \lambda_A^{-1}(a) &:= (\mathsf{false}, a), \\ \lambda_A(\mathsf{false}, a) &:= a, \end{aligned}$$

and a similar reasoning applies for the remaining diagrams. □

4.2 Smash product of spheres

We now study the structure of the smash product on spheres. We first notice that taking the smash product with $\mathbb{S}^1$ is equivalent to taking the suspension.

Proposition 4.2.1. *For every pointed type A, there is an equivalence*

$$\mathbb{S}^1 \wedge A \simeq \Sigma A.$$

Proof. We define the map $f : \mathbb{S}^1 \wedge A \to \Sigma A$ by

$$\begin{aligned} f &: \mathbb{S}^1 \wedge A \to \Sigma A, \\ f(\mathsf{proj}(x, a)) &:= \widetilde{f}(x, a), \end{aligned}$$

where

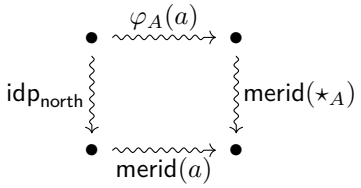
$$\begin{aligned} \widetilde{f} &: \mathbb{S}^1 \to A \to \Sigma A, \\ \widetilde{f}(\mathsf{base}, a) &:= \mathsf{north}, \\ \mathsf{ap}_{\widetilde{f}(-,a)}(\mathsf{loop}) &:= \varphi_A(a). \end{aligned}$$

We also need to give the images by f of the other constructors of $\mathbb{S}^1 \wedge A$. We send the basepoint $\star_{\mathbb{S}^1 \wedge A}$ to north. For $\mathsf{proj_l}(a)$ we can take $\mathsf{idp}_{\mathsf{north}}$ because $f(\mathsf{proj}(\mathsf{base}, a))$ is equal to north by definition. For $\mathsf{proj_r}(x)$ we need to construct a path from $\widetilde{f}(x, \star_A)$ to north for every $x : \mathbb{S}^1$ and we proceed by induction on x. For base they are equal by definition so we take $\mathsf{idp}_{\mathsf{north}}$, and for loop we use the fact that $\varphi_A(\star_A) = \mathsf{idp}_{\mathsf{north}}$. Finally, we have to prove that the two paths from $\widetilde{f}(\mathsf{base}, \star_A)$ to north that we just constructed are equal, but it is true because they are both equal to $\mathsf{idp}_{\mathsf{north}}$.

The inverse function is defined by

$$\begin{aligned} g &: \Sigma A \to \mathbb{S}^1 \wedge A, \\ g(\mathsf{north}) &:= \star_{\mathbb{S}^1 \wedge A}, \\ g(\mathsf{south}) &:= \star_{\mathbb{S}^1 \wedge A}, \\ \mathsf{ap}_g(\mathsf{merid}(a)) &:= \mathsf{proj_l}(a)^{-1} \cdot \mathsf{ap}_{\mathsf{proj}(-,a)}(\mathsf{loop}) \cdot \mathsf{proj_l}(a). \end{aligned}$$

The composition $f \circ g : \Sigma A \to \Sigma A$ sends both north and south to north and sends $\mathsf{merid}(a)$ to $\varphi_A(a)$. Therefore we have $f(g(x)) = x$ via $\mathsf{idp}_{\mathsf{north}}$ for $x = \mathsf{north}$, $\mathsf{merid}(\star_A)$ for $x = \mathsf{south}$, and the filler of the square

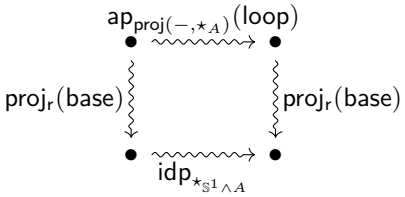


given by the definition of $\varphi_A(a)$ for $\mathsf{merid}(a)$.

For the composition $g \circ f : \mathbb{S}^1 \wedge A \to \mathbb{S}^1 \wedge A$, we first prove that $g(\widetilde{f}(x, a)) = \mathsf{proj}(x, a)$ for all $x : \mathbb{S}^1$ and $a : A$, by induction on x. For base we have $g(\widetilde{f}(\mathsf{base}, a)) = g(\mathsf{north}) = \star_{\mathbb{S}^1 \wedge A}$ by definition, which is equal to $\mathsf{proj}(\mathsf{base}, a)$ along $\mathsf{proj}_{\mathsf{l}}(a)$, and for loop we have

$$\begin{aligned}\mathsf{ap}_g(\mathsf{ap}_{\widetilde{f}(-,a)}(\mathsf{loop})) &= \mathsf{ap}_g(\varphi_A(a)) \\ &= \mathsf{proj}_{\mathsf{l}}(a)^{-1} \cdot \mathsf{ap}_{\mathsf{proj}(-,a)}(\mathsf{loop}) \cdot \mathsf{proj}_{\mathsf{l}}(a) \\ &\quad \cdot \mathsf{proj}_{\mathsf{l}}(\star_A)^{-1} \cdot \mathsf{ap}_{\mathsf{proj}(-,\star_A)}(\mathsf{loop})^{-1} \cdot \mathsf{proj}_{\mathsf{l}}(\star_A).\end{aligned}$$

Moreover the naturality square of $\mathsf{proj}_{\mathsf{r}}$ on loop is



which, in conjunction with the equality $\mathsf{proj}_{\mathsf{rl}} : \mathsf{proj}_{\mathsf{r}}(\mathsf{base}) = \mathsf{proj}_{\mathsf{l}}(\star_A)$, shows that

$$\mathsf{ap}_g(\mathsf{ap}_{\widetilde{f}(-,a)}(\mathsf{loop})) = \mathsf{proj}_{\mathsf{l}}(a)^{-1} \cdot \mathsf{ap}_{\mathsf{proj}(-,a)}(\mathsf{loop}) \cdot \mathsf{proj}_{\mathsf{l}}(a),$$

which is what we wanted. Now we have to prove that $g(f(x)) = x$ for the other constructors of $\mathbb{S}^1 \wedge A$. For the basepoint it's true by definition. The path $\mathsf{proj}_{\mathsf{l}}(a)$ is sent to $\mathsf{idp}_{\star_{\mathbb{S}^1 \wedge A}}$ which is what we want because the equality $g(f(\mathsf{proj}(\mathsf{base}, a))) = \mathsf{proj}(\mathsf{base}, a)$ is already given by $\mathsf{proj}_{\mathsf{l}}(a)$. For $\mathsf{proj}_{\mathsf{r}}(x)$, we have to prove that $\mathsf{ap}_g(\mathsf{ap}_f(\mathsf{proj}_{\mathsf{r}}(x)))$ is equal to the composition of the path $g(f(\mathsf{proj}(x, \star_A))) = \mathsf{proj}(x, \star_A)$ constructed above with the path $\mathsf{proj}_{\mathsf{r}}(x)$, and we proceed by induction on x. For base, the image of $\mathsf{proj}_{\mathsf{r}}(\mathsf{base})$ by $g \circ f$ is the constant path, and the equality $g(f(\mathsf{proj}(x, \star_A))) = \mathsf{proj}(x, \star_A)$ is $\mathsf{proj}_{\mathsf{l}}(\star_A)^{-1}$, therefore the result follows from $\mathsf{proj}_{\mathsf{rl}}$. For loop, the image by $g \circ f$ is a coherence operation (coming from the equality $\varphi_A(\star_A) = \mathsf{idp}_{\mathsf{north}}$) and the equality $g(f(\mathsf{proj}(x, \star_A))) = \mathsf{proj}(x, \star_A)$ is essentially $\mathsf{ap}_{\mathsf{proj}_{\mathsf{l}}}(\mathsf{loop})^{-1}$, therefore it follows. Finally, for $\mathsf{proj}_{\mathsf{rl}}$ it follows automatically as well. □

The previous proposition together with associativity of the smash product shows that $\mathbb{S}^n \wedge \mathbb{S}^m \simeq \mathbb{S}^{n+m}$ for every $n, m : \mathbb{N}$, but we need to prove that this collection of equivalences is compatible with associativity of the smash product in the following sense.

Proposition 4.2.2. *For every $n, m : \mathbb{N}^*$, there is an equivalence*

$$\wedge_{n,m} : \mathbb{S}^n \wedge \mathbb{S}^m \simeq \mathbb{S}^{n+m}$$

together with a filler of the diagram

$$\begin{array}{ccccc}
(\mathbb{S}^n \wedge \mathbb{S}^m) \wedge \mathbb{S}^k & \xrightarrow{(\wedge_{n,m}) \wedge \mathsf{id}_{\mathbb{S}^k}} & \mathbb{S}^{n+m} \wedge \mathbb{S}^k & & \\
 & & & \searrow^{\wedge_{n+m,k}} & \\
\downarrow{\scriptstyle \alpha_{\mathbb{S}^n, \mathbb{S}^m, \mathbb{S}^k}} & & & & \mathbb{S}^{n+m+k} \\
 & & & \nearrow_{\wedge_{n,m+k}} & \\
\mathbb{S}^n \wedge (\mathbb{S}^m \wedge \mathbb{S}^k) & \xrightarrow[\mathsf{id}_{\mathbb{S}^n} \wedge (\wedge_{m,k})]{} & \mathbb{S}^n \wedge \mathbb{S}^{m+k} & &
\end{array}$$

Proof. We define the family of types $(\mathbf{S}^n)_{n:\mathbb{N}^*}$ by

$$\begin{aligned}
\mathbf{S}^1 &:= \mathbb{S}^1, \\
\mathbf{S}^{n+1} &:= \mathbb{S}^1 \wedge \mathbf{S}^n.
\end{aligned}$$

Using proposition 4.2.1 we can easily construct by induction on n an equivalence between $\mathbf{S}^n$ and $\mathbb{S}^n$, therefore it is enough to prove the result for the family $(\mathbf{S}^n)$ instead of $(\mathbb{S}^n)$.

The equivalence $\wedge_{n,m}$ is defined by induction on n.

- For $n = 1$ we take the identity equivalence, as $\mathbf{S}^1 \wedge \mathbf{S}^m$ is equal to $\mathbf{S}^{m+1}$ by definition.

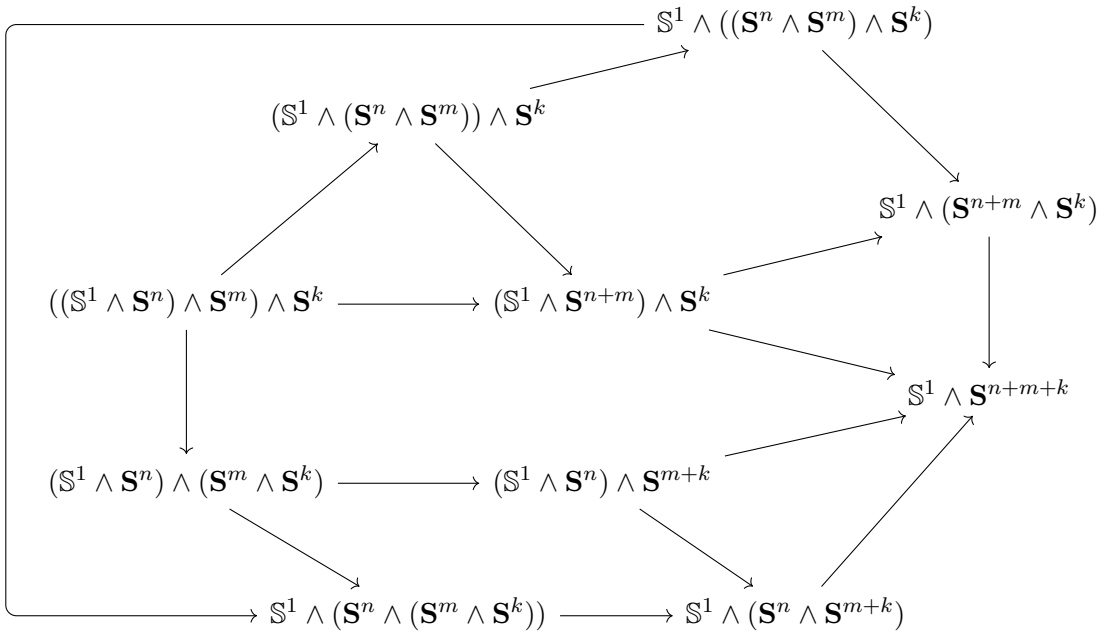
- For $n + 1$ we take the composition

$$\begin{array}{ccccc}
\mathbf{S}^{n+1} \wedge \mathbf{S}^m & \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots & \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots & \cdots\cdots\rightarrow & \mathbf{S}^{n+m+1} \\
\| & & & & \| \\
(\mathbb{S}^1 \wedge \mathbf{S}^n) \wedge \mathbf{S}^m & \xrightarrow{\sim} & \mathbb{S}^1 \wedge (\mathbf{S}^n \wedge \mathbf{S}^m) & \xrightarrow{\sim} & \mathbb{S}^1 \wedge \mathbf{S}^{n+m}
\end{array}$$

  where we used associativity of the smash product and then the smash by $\mathbb{S}^1$ of $\wedge_{n,m}$.

The compatibility for $n = 1$ holds by definition (as both $\wedge_{n,m}$ and $\wedge_{n,m+k}$ are the identity), and for $n+1$ let's consider the following diagram where we omitted the names

of the arrows for readability.



We want to fill the inner pentagon. The three triangles are the definition of the maps of the form $(\mathbb{S}^1 \wedge \mathbf{S}^i) \wedge \mathbf{S}^j \to \mathbb{S}^1 \wedge \mathbf{S}^{i+j}$. The two squares are filled by naturality of the associativity of the smash product. The left part of the diagram is filled by the pentagon of coherences of associativity. Finally the outer diagram is filled by the smash product by $\mathbb{S}^1$ of the induction hypothesis. Therefore, we get a filler of the inner pentagon. □

The definition of the equivalences $\wedge_{n,m}$ is rather indirect, because it goes through the family of types $(\mathbf{S}^n)$. A more direct way to see them is as follows.

Proposition 4.2.3. *Given $n, m : \mathbb{N}^*$, the map $\wedge_{n+1,m}$ of type $(\Sigma\mathbb{S}^n) \wedge \mathbb{S}^m \to \Sigma(\mathbb{S}^{n+m})$ satisfies, for every $x : \mathbb{S}^n$ and $y : \mathbb{S}^m$,*

$$\mathsf{ap}_{\wedge_{n+1,m}}(\mathsf{ap}_{\mathsf{proj}(-,y)}(\varphi_{\mathbb{S}^n}(x))) = \varphi_{\mathbb{S}^{n+m}}(\wedge_{n,m}(\mathsf{proj}(x,y))).$$

Proof. It follows from the definition of $\wedge_{n+1,m}$ as we defined it by first using associativity of the smash product (which corresponds to moving the φ to the outside) and then by applying $\wedge_{n,m}$. □

The compatibility with commutativity of the smash product is a bit more subtle as there is a sign.

Proposition 4.2.4. *For $n, m : \mathbb{N}^*$, there is a filler of the diagram*

$$\begin{array}{ccc} \mathbb{S}^n \wedge \mathbb{S}^m & \xrightarrow{\sigma_{\mathbb{S}^n,\mathbb{S}^m}} & \mathbb{S}^m \wedge \mathbb{S}^n \\ {\scriptstyle \wedge_{n,m}}\downarrow & & \downarrow{\scriptstyle \wedge_{m,n}} \\ \mathbb{S}^{n+m} & \xrightarrow[(-1)^{nm}]{} & \mathbb{S}^{n+m} \end{array}$$

where $(-1)^{nm}$ *is*

- *the identity map, if* nm *is even,*
- *the map* (-1) *sending* $\mathsf{merid}(a)$ *to* $\mathsf{merid}(a)^{-1}$*, if* nm *is odd.*

Proof. As in the proof of the previous proposition, we work with the family $(\mathbf{S}^n)$ instead of $(\mathbb{S}^n)$, the map (-1) corresponding to the map

$$\begin{aligned}(-1) &: \mathbb{S}^1 \wedge \mathbf{S}^n \to \mathbb{S}^1 \wedge \mathbf{S}^n,\\ (-1) &:= i \wedge \mathsf{id}_{\mathbf{S}^n},\end{aligned}$$

where the map $i : \mathbb{S}^1 \to \mathbb{S}^1$ is the map sending base to base and loop to loop^{-1}.

The idea is quite simple. The map exchanging $\mathbf{S}^n$ and $\mathbf{S}^m$ has to swap every $\mathbb{S}^1$-factor of $\mathbf{S}^m$ with every $\mathbb{S}^1$-factor of $\mathbf{S}^n$. Therefore in total we have nm permutations between two $\mathbb{S}^1$ factors, so when nm is even they all cancel, and otherwise one remains. And it turns out that the map swapping the two factors of $\mathbb{S}^1 \wedge \mathbb{S}^1$ is equal to the map (-1) on $\mathbf{S}^2$.

We first consider the case $n = m = 1$.

Proposition 4.2.5. *There is a filler of the diagram*

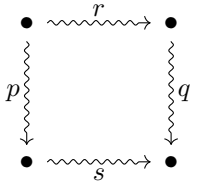
$$\begin{array}{ccc}\mathbf{S}^1 \wedge \mathbf{S}^1 & \xrightarrow{\sigma_{\mathbf{S}^1,\mathbf{S}^1}} & \mathbf{S}^1 \wedge \mathbf{S}^1\\ \| & & \|\\ \mathbf{S}^2 & \xrightarrow[(-1)]{} & \mathbf{S}^2\end{array}$$

Proof. We have to construct an equality between the two functions

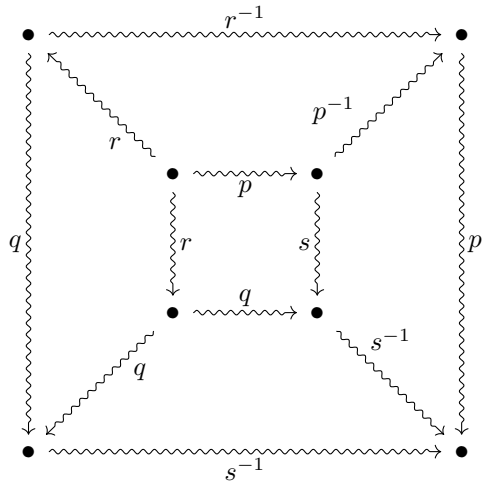
$$\begin{aligned} f &: \mathbf{S}^1 \wedge \mathbf{S}^1 \to \mathbf{S}^1 \wedge \mathbf{S}^1, & g &: \mathbf{S}^1 \wedge \mathbf{S}^1 \to \mathbf{S}^1 \wedge \mathbf{S}^1,\\ f(\mathsf{proj}(x,y)) &:= \mathsf{proj}(y,x), & g(\mathsf{proj}(x,y)) &:= \mathsf{proj}(-x,y).\end{aligned}$$

We prove that they are equal on points of the form $\mathsf{proj}(x, y)$ by double induction on x and y. When either x or y is base, both $f(\mathsf{proj}(x, y))$ and $g(\mathsf{proj}(x, y))$ are equal to $\star_{\mathbf{S}^1\wedge\mathbf{S}^1}$ via either a proj_l or a proj_r. For the case $(\mathsf{loop}, \mathsf{loop})$, the idea is that we are given a square and we want to show that flipping it diagonally (using f) or horizontally (using g) gives equal squares along some coherence operations.

If we start with the square



then we consider the cube



(4.2.6)

The inner and outer faces are the diagonal and horizontal flipping of the square, and the four other sides are defined using the equalities

$$\begin{aligned} r \cdot r^{-1} &= p \cdot p^{-1}, \\ p^{-1} \cdot p &= s \cdot s^{-1}, \\ q \cdot s^{-1} &= q \cdot s^{-1}, \\ r \cdot q &= r \cdot q. \end{aligned}$$

In order to fill this cube we proceed by induction on the whole square. Another way to see it is to do a path induction on p, q and r, and then a fourth path induction on the square seen as the equality $s = p^{-1} \cdot r \cdot q$. Then we only have to fill the cube in the case where the square is the constant square with idp on all four sides. In that case, all of the sides of the cube are constant squares so we can take the constant cube. It concludes the proof that f and g are equal. □

Note that the proof above works because we managed to write the cube 4.2.6 starting from an arbitrary square. There is no way to make a similar cube between the original square and its diagonal flipping, for instance.

We now prove that the maps (-1) are stable under smash product by $\mathbf{S}^m$ on the right and by $\mathbb{S}^1$ on the left.

Proposition 4.2.7. *For every $n, m : \mathbb{N}$, there is a filler of the square*

$$\begin{array}{ccc} \mathbf{S}^n \wedge \mathbf{S}^m & \xrightarrow{(-1)\wedge \mathsf{id}_{\mathbf{S}^m}} & \mathbf{S}^n \wedge \mathbf{S}^m \\ {\scriptstyle \wedge_{n,m}}\downarrow & & \downarrow{\scriptstyle \wedge_{n,m}} \\ \mathbf{S}^{n+m} & \xrightarrow[(-1)]{} & \mathbf{S}^{n+m} \end{array}$$

Proof. We proceed by induction on n. For $n = 1$ it is the definition of the map (-1) on $\mathbf{S}^{m+1}$. For $n + 1$ we take the composition of the two squares

$$\begin{array}{ccc}
(\mathbb{S}^1 \wedge \mathbf{S}^n) \wedge \mathbf{S}^m & \xrightarrow{(i\wedge \mathsf{id}_{\mathbf{S}^n})\wedge \mathsf{id}_{\mathbf{S}^m}} & (\mathbb{S}^1 \wedge \mathbf{S}^n) \wedge \mathbf{S}^m \\
\downarrow{\scriptstyle \alpha_{\mathbb{S}^1,\mathbf{S}^n,\mathbf{S}^m}} & & \downarrow{\scriptstyle \alpha_{\mathbb{S}^1,\mathbf{S}^n,\mathbf{S}^m}} \\
\mathbb{S}^1 \wedge (\mathbf{S}^n \wedge \mathbf{S}^m) & \xrightarrow[i\wedge(\mathsf{id}_{\mathbf{S}^n}\wedge \mathsf{id}_{\mathbf{S}^m})]{} & \mathbb{S}^1 \wedge (\mathbf{S}^n \wedge \mathbf{S}^m) \\
\downarrow{\scriptstyle \mathsf{id}_{\mathbb{S}^1}\wedge(\wedge_{n,m})} & & \downarrow{\scriptstyle \mathsf{id}_{\mathbb{S}^1}\wedge(\wedge_{n,m})} \\
\mathbf{S}^{n+m+1} & \xrightarrow[i\wedge \mathsf{id}_{\mathbf{S}^{n+m}}]{} & \mathbf{S}^{n+m+1}
\end{array}$$

where the top square comes from naturality of associativity of the smash product and the bottom square comes from the fact that $\mathsf{id}_{\mathbf{S}^n} \wedge \mathsf{id}_{\mathbf{S}^m} = \mathsf{id}_{\mathbf{S}^n \wedge \mathbf{S}^m}$. □

Proposition 4.2.8. *For every $n : \mathbb{N}$, the map*

$$\mathsf{id}_{\mathbb{S}^1} \wedge (-1) : \mathbb{S}^1 \wedge \mathbf{S}^n \to \mathbb{S}^1 \wedge \mathbf{S}^n$$

is equal to

$$(-1) : \mathbf{S}^{n+1} \to \mathbf{S}^{n+1}.$$

Proof. We proceed by induction on n. For $n = 1$ the proof is entirely analogous to the proof of proposition 4.2.5, except that we equate the vertical flipping of a square with its horizontal flipping. For $n + 1$ we take the composition of squares

$$\begin{array}{ccc}
\mathbb{S}^1 \wedge (\mathbb{S}^1 \wedge \mathbf{S}^n) & \xrightarrow{\mathsf{id}_{\mathbb{S}^1}\wedge(i\wedge \mathsf{id}_{\mathbf{S}^n})} & \mathbb{S}^1 \wedge (\mathbb{S}^1 \wedge \mathbf{S}^n) \\
\downarrow{\scriptstyle \alpha^{-1}_{\mathbb{S}^1,\mathbb{S}^1,\mathbf{S}^n}} & & \downarrow{\scriptstyle \alpha^{-1}_{\mathbb{S}^1,\mathbb{S}^1,\mathbf{S}^n}} \\
(\mathbb{S}^1 \wedge \mathbb{S}^1) \wedge \mathbf{S}^n & \xrightarrow[(\mathsf{id}_{\mathbb{S}^1}\wedge i)\wedge \mathsf{id}_{\mathbf{S}^n}]{} & (\mathbb{S}^1 \wedge \mathbb{S}^1) \wedge \mathbf{S}^n \\
\downarrow{\scriptstyle \mathsf{id}_{(\mathbb{S}^1\wedge\mathbb{S}^1)\wedge\mathbf{S}^n}} & & \downarrow{\scriptstyle \mathsf{id}_{(\mathbb{S}^1\wedge\mathbb{S}^1)\wedge\mathbf{S}^n}} \\
(\mathbb{S}^1 \wedge \mathbb{S}^1) \wedge \mathbf{S}^n & \xrightarrow[(i\wedge \mathsf{id}_{\mathbb{S}^1})\wedge \mathsf{id}_{\mathbf{S}^n}]{} & (\mathbb{S}^1 \wedge \mathbb{S}^1) \wedge \mathbf{S}^n \\
\downarrow{\scriptstyle \wedge_{2,n}} & & \downarrow{\scriptstyle \wedge_{2,n}} \\
\mathbf{S}^{n+2} & \xrightarrow[(-1)]{} & \mathbf{S}^{n+2}
\end{array}$$

where the top one comes from naturality of associativity of the smash product, the middle one comes from the case $n = 1$ and the bottom one comes from proposition 4.2.7. □

We now come back to the proof of proposition 4.2.4. Given n and m, we assume by induction that we have constructed a filler of the diagram for $(n, 1)$ and for (n, m), and we construct one for $(n, m+1)$. Let's consider the diagram 4.1 where we want to fill the upper square. The outer hexagon is the coherence hexagon between the commutator and the

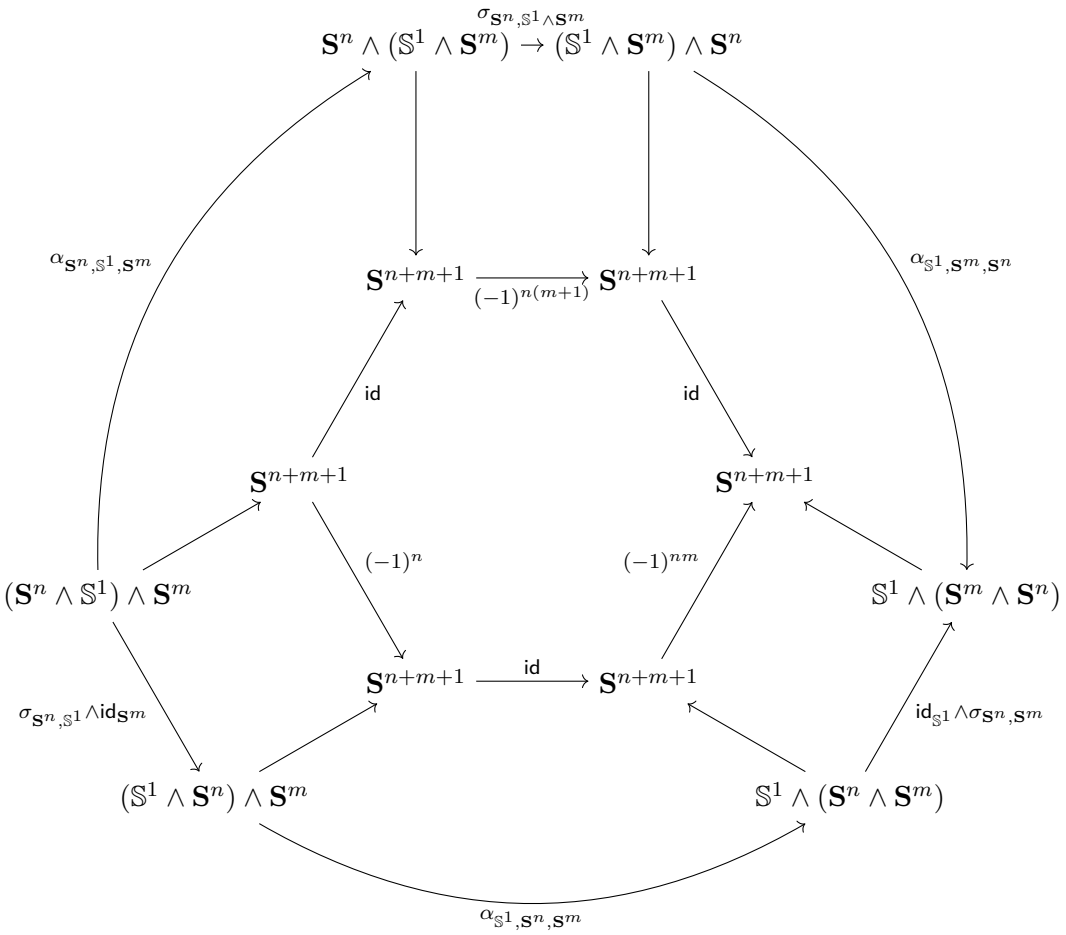


Figure 4.1: Case $(n, m+1)$

associator. The three curved squares come from the compatibility of the maps $\wedge_{i,j}$ with the associator (proposition 4.2.2). The lower left square is the smash product of the case $(n, 1)$ with $\mathbf{S}^m$ and the lower right square is the smash product of $\mathbb{S}^1$ with the case (n, m). Finally, the inner diagram has a filler given by the equality $(-1)^{nm} \circ (-1)^n = (-1)^{n(m+1)}$ which can easily be verified.

Finally, if the property is true for (n, m), then it is true for (m, n) as well. Combining that with the case above and the base case, we obtain that the property is true for every (n, m). □

4.3 Smash product and connectedness

The connectedness of a smash product is given by the following proposition.

Proposition 4.3.1. *Given two pointed types A and B, if A is m-connected and B is*

n-connected then $A \wedge B$ is $(m+n+1)$-connected.

Proof. Notice that the map $i^{\vee}_{A,B} : A \vee B \to A \times B$ is the pushout-product of the two maps $\mathbf{1} \to A$ and $\mathbf{1} \to B$, which are $(m-1)$- and $(n-1)$-connected, therefore proposition 3.2.8 shows that $i^{\vee}_{A,B}$ is $(m+n)$-connected. Given that we have a pushout square

$$\begin{array}{ccc} A \vee B & \xrightarrow{i^{\vee}_{A,B}} & A \times B \\ \downarrow & \ulcorner & \downarrow \\ \mathbf{1} & \longrightarrow & A \wedge B \end{array}$$

and that the top map is $(m+n)$-connected, we conclude that the map $\mathbf{1} \to A \wedge B$ is $(m+n)$-connected as well and therefore $A \wedge B$ is $(m+n+1)$-connected. □

For the smash product of maps, it's not as easy because the answer depends not only on the connectivity of the maps but also on the connectivity of the types. We first prove the following proposition.

Proposition 4.3.2. *Given a pointed map $f : A \to_\star B$ and a pointed type C, if f is n_f-connected and C is n_C-connected then the map $f \wedge \mathsf{id}_C : A \wedge C \to B \wedge C$ is $(n_f + n_C + 1)$-connected.*

As a sanity check, note that if $C = \mathbf{2}$, then $f \wedge \mathsf{id}_C = f$ and $n_C = -1$, and we do have $n_f + n_C + 1 = n_f$.

Proof. We consider $P : B \wedge C \to \mathsf{Type}$ a family of $(n_f + n_C + 1)$-truncated types and

$$d : (x : A \wedge C) \to P((f \wedge \mathsf{id}_C)(x)).$$

According to proposition 3.2.9 and given that f is n_f-connected, if we take an arbitrary $c : C$ then the function

$$\lambda s.s \circ f : \prod_{b:B} P(\mathsf{proj}(b,c)) \to \prod_{a:A} P(\mathsf{proj}(f(a),c))$$

is $(n_C - 1)$-truncated. Note also that there is an element in the codomain given by $d \circ \mathsf{proj}(-,c)$. Let's now consider $Q : C \to \mathsf{Type}$ such that $Q(c)$ is the fiber of the function $\lambda s.s \circ f$ above over $d \circ \mathsf{proj}(-,c)$. We just proved that Q is a family of $(n_C - 1)$-truncated types, therefore in order to construct a section of it it is enough to give an element of $Q(\star_C)$. An element of $Q(\star_C)$ consists of a function $k : (b : B) \to P(\mathsf{proj}(b, \star_C))$ together with an equality between $k(f(a))$ and $d(\mathsf{proj}(a, \star_C))$ for every $a : A$. Therefore the map

$$\lambda b.\, \mathsf{transport}^P(\mathsf{proj_r}(b)^{-1}, d(\star_{A \wedge C}))$$

together with the equality

$$\mathsf{transport}^P(\mathsf{proj_r}(f(a))^{-1}, d(\star_{A \wedge C})) = d(\mathsf{proj}(a, \star_C))$$

given by $\mathsf{ap}_d(\mathsf{proj_r}(a))$ is an element of $Q(\star_C)$.

Putting everything together, what we obtain is a function

$$g : (b : B)(c : C) \to P(\mathsf{proj}(b, c))$$

together with paths, for all $a : A$, $b : B$ and $c : C$,

$$g(f(a), c) = d(\mathsf{proj}(a, c)), \tag{4.3.3}$$

$$g(b, \star_C) = \mathsf{transport}^P(\mathsf{proj_r}(b)^{-1}, d(\star_{A\wedge C})) \tag{4.3.4}$$

which are equal (via a 2-dimensional path) in the case $g(f(a), \star_C)$.

Therefore we can now define $h : (x : B \wedge C) \to P(x)$ as follows. For the basepoint $\star_{B\wedge C}$ we take $d(\star_{A\wedge C})$. For an element of the form $\mathsf{proj}(b, c)$ we take $g(b, c)$. For $\mathsf{proj_r}(b)$ it is automatic by the equation 4.3.4. For $\mathsf{proj_l}(c)$ we prove that $g(\star_B, c)$ is equal to $d(\star_{A\wedge C})$ over $\mathsf{proj_l}(c)$ by the composition

$$g(\star_B, c) \leftsquigarrow g(f(\star_A), c) \overset{4.3.3}{\rightsquigarrow} d(\mathsf{proj}(\star_A, c)) \rightsquigarrow d(\star_{A\wedge C})$$

(note that the last equality is a dependent equality over $\mathsf{proj_l}(c)$). Finally for $\mathsf{proj_{rl}}$ we have to check that in the case $c = \star_C$, the composition we just defined is equal to the one obtained by transporting $d(\star_{A\wedge C})$ along $\mathsf{proj_r}(\star_B)$, and this follows from the 2-dimensional equality for $g(f(a), \star_C)$ mentioned above. □

We can now compute the connectedness of a smash product of maps.

Proposition 4.3.5. *Given two maps $f : A \to A'$ and $g : B \to B'$, if f is n_f-connected, g is n_g-connected, A' is $n_{A'}$-connected and B is n_B-connected, then $f \wedge g$ is k-connected, where*

$$k = \mathsf{min}(n_f + n_B + 1, n_g + n_{A'} + 1).$$

Proof. We have

$$f \wedge g = (f \wedge \mathsf{id}_B) \circ (\mathsf{id}_{A'} \wedge g).$$

□

Chapter 5

Cohomology

In this chapter we introduce the cohomology groups with integer coefficients of a space, together with their additive and multiplicative structure, and we prove that they form a graded ring. This means that for every space X we define a sequence of abelian groups $(H^n(X))_{n:\mathbb{N}}$ and a multiplication operation $\smile : H^i(X) \times H^j(X) \to H^{i+j}(X)$ which is distributive over addition, associative, and graded-commutative in the sense that for every $x : H^i(X)$ and $y : H^j(X)$ we have

$$x \smile y = (-1)^{ij}(y \smile x).$$

We also construct the Mayer–Vietoris sequence, and we compute the cohomology ring of a product of two spheres. We then define the Hopf invariant, which is an invariant of maps of the form $\mathbb{S}^{2n-1} \to \mathbb{S}^n$ constructed using the multiplicative structure of cohomology. Finally we show that the James construction provides functions of Hopf invariant 2 for every even n, which proves that all groups of the form $\pi_{4n-1}(\mathbb{S}^{2n})$ are infinite and that $\pi_4(\mathbb{S}^3)$ is equal to either $\mathbb{Z}/2\mathbb{Z}$ or $\mathbb{Z}/1\mathbb{Z}$.

The usual definition of singular cohomology using singular cochains cannot be reproduced in homotopy type theory because defining the *set* of singular cochains requires taking the underlying set of a topological space which is not an operation invariant under homotopy equivalence, hence it cannot exist in homotopy type theory. We define cohomology instead via the Eilenberg–MacLane spaces $K(\mathbb{Z}, n)$, which can be defined directly very easily by truncating the spheres. It is also possible to define the Eilenberg–MacLane spaces $K(G, n)$ for an arbitrary group G (abelian if $n \geq 2$), as was shown by Dan Licata and Eric Finster in [LF14], and to use them to define cohomology with coefficients in G. However, given that we only use cohomology with integer coefficients in this work and that the definition of $K(\mathbb{Z}, n)$ is much simpler, we restrict ourselves to this case. For readability, we only write K_n instead of $K(\mathbb{Z}, n)$.

It is well-known that this definition of cohomology is the right one in homotopy type theory, but the multiplicative structure and everything which depends on it is new to our knowledge. The Mayer–Vietoris sequence has already been studied by Evan Cavallo in homotopy type theory in [Cav15], but with a rather different approach. His construction starts from the Eilenberg–Steenrod axioms for cohomology and uses extensively the

cubical machinery whereas the construction presented here is more concrete.

5.1 The cohomology ring of a space

Definition 5.1.1. For $n : \mathbb{N}$, the type K_n is the n-truncated and $(n-1)$-connected pointed type defined by

$$\begin{aligned} K_0 &:= \mathbb{Z}, \\ K_n &:= \|\mathbb{S}^n\|_n \text{ for } n \geq 1. \end{aligned}$$

An important property of the family (K_n) is that K_n is equivalent to ΩK_{n+1} for every $n : \mathbb{N}$.

Proposition 5.1.2. *For every $n : \mathbb{N}$, there is an equivalence $\sigma_n : K_n \simeq \Omega K_{n+1}$ and, when $n \geq 1$, a filler of the diagram*

$$\begin{array}{ccc} \mathbb{S}^n & \xrightarrow{\varphi_{\mathbb{S}^n}} & \Omega\Sigma\mathbb{S}^n \\ {\scriptstyle |-|}\downarrow & & \downarrow{\scriptstyle \Omega|-|} \\ K_n & \xrightarrow[\sigma_n]{\sim} & \Omega K_{n+1} \end{array}$$

Proof. We define σ_n by

$$\begin{aligned} \sigma_0(k) &:= (\mathsf{ap}_{|-|}(\mathsf{loop}))^k, \\ \sigma_n(|x|) &:= \mathsf{ap}_{|-|}(\varphi_{\mathbb{S}^n}(x)) \text{ for } n \geq 1, \end{aligned}$$

and the diagram is filled by idp. We now have to prove that σ_n is an equivalence. For $n = 0$, we proved it in section 2.2. For $n \geq 1$, the composition

$$K_n = \|\mathbb{S}^n\|_n \xrightarrow{\|\varphi_{\mathbb{S}^n}\|_n} \left\|\Omega\mathbb{S}^{n+1}\right\|_n \xrightarrow{\sim} \Omega\left\|\mathbb{S}^{n+1}\right\|_{n+1} = \Omega K_{n+1}$$

is equal to σ_n, hence it is enough to prove that $\|\varphi_{\mathbb{S}^n}\|_n$ is an equivalence. For $n = 1$, we know from section 2.5 that the map $d : \Omega\mathbb{S}^2 \to \mathbb{S}^1$ sending p to $\mathsf{transport}^{\mathsf{Hopf}}(p, \mathsf{base})$ has fiber $\Omega\mathbb{S}^3$ and, hence, is 1-connected. Moreover, for every $x : \mathbb{S}^1$ we have

$$\begin{aligned} \mathsf{transport}^{\mathsf{Hopf}}(\varphi_{\mathbb{S}^1}(x), \mathsf{base}) &= \mathsf{transport}^{\mathsf{Hopf}}(\mathsf{merid}_{\mathbb{S}^1}(x) \cdot \mathsf{merid}_{\mathbb{S}^1}(\mathsf{base})^{-1}, \mathsf{base}) \\ &= \mathsf{transport}^{\mathsf{Hopf}}(\mathsf{merid}_{\mathbb{S}^1}(\mathsf{base})^{-1}, \\ &\qquad \mathsf{transport}^{\mathsf{Hopf}}(\mathsf{merid}_{\mathbb{S}^1}(x), \mathsf{base})) \\ &= \mathsf{transport}^{\mathsf{Hopf}}(\mathsf{merid}_{\mathbb{S}^1}(x), \mathsf{base}) \\ &= \mu(\mathsf{base}, x) \\ &= x, \end{aligned}$$

which shows that the composition

$$\mathbb{S}^1 \xrightarrow{\varphi_{\mathbb{S}^1}} \Omega\mathbb{S}^2 \xrightarrow{d} \mathbb{S}^1$$

is equal to the identity function. Given that the map $\|d\|_1$ is an equivalence, the map $\|\varphi_{\mathbb{S}^1}\|_1$ is an equivalence as well, which concludes the case $n = 1$.

For $n \geq 2$, the Freudenthal suspension theorem states that the map $\varphi_{\mathbb{S}^n} : \mathbb{S}^n \to \Omega\mathbb{S}^{n+1}$ is $(2n-2)$-connected, and we have $2n-2 \geq n$; hence, $\|\varphi_{\mathbb{S}^n}\|_n$ is an equivalence. □

Definition 5.1.3. Given a type X and $n : \mathbb{N}$, the *n-th cohomology group* of X is the type

$$H^n(X) := \|X \to K_n\|_0 ,$$

and, if X is pointed, the *n-th reduced cohomology group* of X is the type

$$\widetilde{H}^n(X) := \|X \to_\star K_n\|_0 .$$

Note that for every type X we have $H^n(X) \simeq \widetilde{H}^n(X \sqcup \mathbf{1})$, where $X \sqcup \mathbf{1}$ is pointed by $\mathsf{inr}(\star_{\mathbf{1}})$, given that a map from X to K_n can be seen as a pointed map from $X \sqcup \mathbf{1}$ to K_n. Elements of $\widetilde{H}^n(X)$ can also be seen as elements of $H^n(X)$ by forgetting the pointedness.

We now construct the group structure on cohomology groups. The idea is to construct the whole structure on K_n and then lift it to $H^n(X)$. We write 0 for $\star_{K_n}$.

Proposition 5.1.4. *There are two maps $+ : K_n \times K_n \to K_n$ and $- : K_n \to K_n$, and equalities*

$$\begin{aligned} x + 0 &= x, \\ 0 + x &= x, \\ x + (-x) &= 0, \\ (-x) + x &= 0, \\ (x + y) + z &= x + (y + z), \\ x + y &= y + x, \end{aligned}$$

for every $x, y, z : K_n$.

Proof. We use the equivalence $K_n \simeq \Omega K_{n+1}$. We define the addition of two elements x and y of K_n as the composition of the corresponding loops in ΩK_{n+1}, and the opposite of an element of K_n as the inverse of the corresponding loop. Note that $\star_{K_n}$ corresponds to the constant path of ΩK_{n+1}. The unit laws, inverse laws, and associativity law come from the corresponding facts about paths, and commutativity comes from the Eckmann–Hilton argument (proposition 2.1.6), given that ΩK_{n+1} is equivalent to the double loop space $\Omega^2 K_{n+2}$. □

Definition 5.1.5. Given a type X and a natural number n, we equip $H^n(X)$ with the structure of abelian group given by

$$\begin{aligned} + &: H^n(X) \times H^n(X) \to H^n(X), \\ |\alpha| + |\beta| &:= |\lambda x.(\alpha(x) + \beta(x))|, \end{aligned}$$

$$\begin{aligned} -&: H^n(X) \to H^n(X), \\ -|\alpha| &:= |\lambda x.(-\alpha(x))|, \end{aligned}$$

$$\begin{aligned} 0 &: H^n(X), \\ 0 &:= |\lambda x.0|. \end{aligned}$$

The group laws are proved in a similar fashion. For instance we prove associativity by

$$\begin{aligned} a &: (\alpha, \beta, \gamma : H^n(X)) \to (\alpha + \beta) + \gamma = \alpha + (\beta + \gamma), \\ a(|\alpha|, |\beta|, |\gamma|) &:= \mathsf{ap}_{|-|}(\mathsf{funext}(\lambda x.a'(\alpha(x), \beta(x), \gamma(x)))), \end{aligned}$$

where a' is associativity of $+$ on K_n. We also equip $\widetilde{H}^n(X)$ with the structure of abelian group defined in the exact same way, and the forgetful map from $\widetilde{H}^n(X)$ to $H^n(X)$ is a group homomorphism.

For the multiplicative structure, we first define a map $\smile : K_i \wedge K_j \to K_{i+j}$. It induces a map $K_i \times K_j \to K_{i+j}$ by composition with proj which then induces a product on cohomology groups called the *cup product*:

$$\smile : H^i(X) \times H^j(X) \to H^{i+j}(X).$$

Definition 5.1.6. For every $i, j : \mathbb{N}^*$, there is a map $\smile : K_i \wedge K_j \to K_{i+j}$ together with a filler of the square

$$\begin{array}{ccc} \mathbb{S}^i \wedge \mathbb{S}^j & \xrightarrow{\wedge_{i,j}} & \mathbb{S}^{i+j} \\ {\scriptstyle |-|_i \wedge |-|_j}\downarrow & & \downarrow{\scriptstyle |-|_{i+j}} \\ K_i \wedge K_j & \xrightarrow{\smile} & K_{i+j} \end{array}$$

Proof. For every $k : \mathbb{N}^*$, the type $\mathbb{S}^k$ is $(k-1)$-connected by proposition 2.4.2 and the map $|-|_k : \mathbb{S}^k \to K_k$ is k-connected by proposition 2.3.11. Therefore, by proposition 4.3.5, the map $|-|_i \wedge |-|_j$ on the left of the diagram is $(i+j)$-connected. Moreover, the type K_{i+j} is $(i+j)$-truncated; hence, by proposition 2.3.8, we obtain the map $\smile$ and a filler of the diagram. □

This definition doesn't work when either i or j is equal to 0, because then we have $K_0 = \mathbb{Z}$, which is not equal to $\|\mathbb{S}^0\|_0$. In those cases we define the cup product by

$$\begin{aligned} \smile &: K_0 \wedge K_j \to K_j, \\ n \smile \beta &:= \beta + \cdots + \beta \quad (n \text{ terms}), \end{aligned}$$

$$\begin{aligned} \smile &: K_i \wedge K_0 \to K_i, \\ \alpha \smile m &:= \alpha + \cdots + \alpha \quad (m \text{ terms}). \end{aligned}$$

These two definitions are consistent in the case $i = j = 0$ because multiplication of integers is commutative.

Proposition 5.1.7. *The collection of maps $\smile : K_i \wedge K_j \to K_{i+j}$ we just defined is distributive with respect to addition, graded-commutative, and associative.*

Proof. **Distributivity** Given $x : K_n$ and $y, z : K_m$ we want to prove that

$$x \smile (y + z) = (x \smile y) + (x \smile z).$$

If $m = 0$, it's clear, so we assume now $m > 0$. The idea is to show that we can assume that either y or z is equal to 0. Distributivity is equivalent to giving a filler of the diagram

$$\begin{array}{ccccc} K_n \wedge (K_m \times K_m) & & \xrightarrow{\quad K_n \wedge + \quad} & & K_n \wedge K_m \\ \downarrow & & & & \smile \downarrow \\ (K_n \wedge K_m) \times (K_n \wedge K_m) & \xrightarrow[\smile \times \smile]{} & K_{n+m} \times K_{n+m} & \xrightarrow[+]{} & K_{n+m} \end{array}$$

where the map on the left sends $\mathsf{proj}(x, (y, z))$ to $(\mathsf{proj}(x, y), \mathsf{proj}(x, z))$.

The map $i^{\vee}_{K_m, K_m} : K_m \vee K_m \to K_m \times K_m$ is $(2m-2)$-connected; therefore, the map

$$\begin{aligned} f &: K_n \wedge (K_m \vee K_m) \to K_n \wedge (K_m \times K_m), \\ f(\mathsf{proj}(x, \mathsf{inl}(y))) &:= \mathsf{proj}(x, (y, 0)), \\ f(\mathsf{proj}(x, \mathsf{inr}(z))) &:= \mathsf{proj}(x, (0, z)) \end{aligned}$$

is $(n + 2m - 2)$-connected. Given that what we have to prove is $(n + m - 1)$-truncated and that $n + 2m - 2 \geq n + m - 1$, it is enough to prove commutativity of the diagram for elements given by f. For an element of the form $\mathsf{proj}(x, \mathsf{inl}(y))$, we have to prove

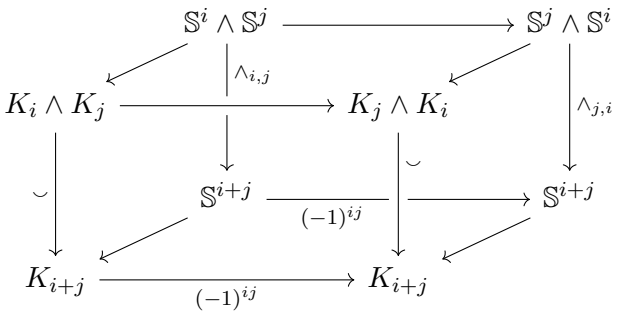
$$x \smile (y + 0) = (x \smile y) + (x \smile 0),$$

which is true because $y + 0 = y$ and $x \smile 0 = 0$. It is true in the same way for elements of the form $\mathsf{proj}(x, \mathsf{inr}(z))$.

Graded commutativity Given $x : K_i$ and $y : K_j$, we want to prove that

$$x \smile y = (-1)^{ij} y \smile x.$$

If either i or j is equal to 0, it is immediate from the definition of the cup product by an element of degree 0. We assume now that i and j are positive and we consider the diagram



where we have to fill the front face. The left and right faces are filled by definition of the cup product. The back face is filled by proposition 4.2.4. The top face is filled by naturality of commutativity of the smash product. The bottom face is filled by definition of (-1) on K_{i+j}. This is not yet enough to fill the front face, it only gives a filler of it after composition with the map $\mathbb{S}^i \wedge \mathbb{S}^j \to K_i \wedge K_j$. But that map is $(i+j)$-connected and we want to construct an equality in K_{i+j} which is $(i+j)$-truncated. Therefore, proposition 2.3.8 gives a filler of the front face.

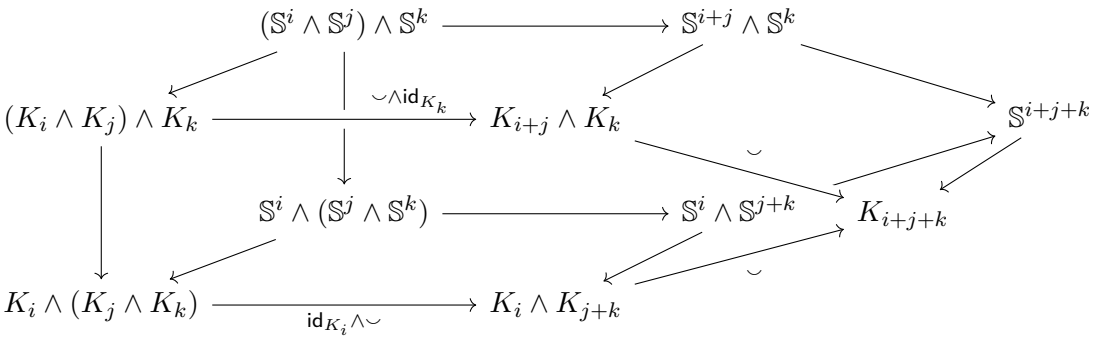
Associativity Given $x : K_i$, $y : K_j$ and $z : K_k$, we want to prove that

$$(x \smile y) \smile z = x \smile (y \smile z).$$

If either i, j, or k is equal to 0, it follows from distributivity. For instance we have

$$\begin{aligned}(n \smile x) \smile y &= (x + \cdots + x) \smile y \\ &= x \smile y + \cdots + x \smile y \\ &= n \smile (x \smile y).\end{aligned}$$

We assume now that i, j, and k are positive and we consider the diagram



We want to fill the front face, and proposition 4.2.2 gives a filler of the back face. The left square is filled by naturality of associativity of the smash product, the top-right and bottom-right squares are filled by definition of the cup product, and the top-left and bottom-left squares are filled by definition of the cup product and naturality of the smash product. The map $(\mathbb{S}^i \wedge \mathbb{S}^j) \wedge \mathbb{S}^k \to (K_i \wedge K_j) \wedge K_k$ is $(i+j+k)$-connected, and we want to construct an equality in K_{i+j+k}, which is $(i+j+k)$-truncated. Therefore, we get a filler of the front face of the diagram. □

This concludes the construction of the structure of graded ring on cohomology. Moreover, as we see now, the operation associating the cohomology ring to a space is a contravariant functor.

Definition 5.1.8. Given $n : \mathbb{N}$ and a map $f : X \to Y$, we define $f^* : H^n(Y) \to H^n(X)$ by

$$\begin{aligned}f^* &: H^n(Y) \to H^n(X), \\ f^*(|\beta|) &:= |\beta \circ f|.\end{aligned}$$

Similarly, if X and Y are pointed and $f : X \to_\star Y$ is a pointed map, we have $f^* : \widetilde{H}^n(Y) \to \widetilde{H}^n(X)$ defined in the same way.

The following proposition is straightforward given that all operations are defined first on K_n and then transferred to $H^n(X)$ by composition.

Proposition 5.1.9. *For every $n, m : \mathbb{N}$, $f : X \to Y$, $g : Y \to Z$, $\beta : H^n(Y)$, $\beta' : H^n(Y)$ and $\gamma : H^m(Y)$, we have*

- $(\mathsf{id}_X)^* = \mathsf{id}_{H^n(X)}$ *and* $(g \circ f)^* = f^* \circ g^*$,
- $f^*(0_{H^n(Y)}) = 0_{H^n(X)}$, $f^*(-\beta) = -f^*(\beta)$ *and* $f^*(\beta + \beta') = f^*(\beta) + f^*(\beta')$,
- $f^*(\beta \smile \gamma) = f^*(\beta) \smile f^*(\gamma)$.

5.2 The Mayer–Vietoris sequence

The *Mayer–Vietoris sequence* is a long exact sequence relating the cohomology groups of a pushout $A \sqcup^C B$ with those of A, B, and C. We prove a "half-reduced" version of the Mayer–Vietoris sequence which allows us to deduce easily both the unreduced version of the Mayer–Vietoris sequence and the long exact sequence of a cofiber.

We consider three types A, B, and C, with A pointed, and two functions $f : C \to A$ and $g : C \to B$. We define $D := A \sqcup^C B$ pointed by $\star_D := \mathsf{inl}(\star_A)$. We define the maps

$$\begin{aligned} i &: \widetilde{H}^n(D) \to \widetilde{H}^n(A) \times H^n(B), & \Delta &: \widetilde{H}^n(A) \times H^n(B) \to H^n(C), \\ i(\delta) &:= (\mathsf{inl}^*(\delta), \mathsf{inr}^*(\delta)), & \Delta(\alpha, \beta) &:= f^*(\alpha) - g^*(\beta), \end{aligned}$$

and

$$\begin{aligned} d &: H^n(C) \to \widetilde{H}^{n+1}(D), \\ d(|\gamma|) &:= \widetilde{d}(\gamma), \end{aligned}$$

where

$$\begin{aligned} \widetilde{d} &: (C \to K_n) \to (D \to_\star K_{n+1}), \\ \widetilde{d}(\gamma)(\mathsf{inl}(a)) &:= 0, \\ \widetilde{d}(\gamma)(\mathsf{inr}(b)) &:= 0, \\ \mathsf{ap}_{\widetilde{d}(\gamma)}(\mathsf{push}(c)) &:= \sigma_n(\gamma(c)), \end{aligned}$$

where in the last line we see $\gamma(c)$ as an element of ΩK_{n+1} via the equivalence $\sigma_n : K_n \simeq \Omega K_{n+1}$. The first thing to note is that i, Δ and d are group homomorphisms. It's obvious for i and Δ, and for d it follows from the fact that

$$\begin{aligned} \mathsf{ap}_{d(|\gamma|+|\gamma'|)}(\mathsf{push}(c)) &= \sigma_n(\gamma(c) + \gamma'(c)) \\ &= \sigma_n(\gamma(c)) \cdot \sigma_n(\gamma'(c)) \\ &= \mathsf{ap}_{d(|\gamma|)}(\mathsf{push}(c)) \cdot \mathsf{ap}_{d(|\gamma'|)}(\mathsf{push}(c)) \\ &= \mathsf{ap}_{d(|\gamma|)+d(|\gamma'|)}(\mathsf{push}(c)). \end{aligned}$$

We then have the following sequence of groups and homomorphisms of groups, which starts at $\widetilde{H}^0(D)$ and then extends infinitely.

$$\begin{array}{ccccc} \widetilde{H}^{n+1}(D) & \xrightarrow{i} & \widetilde{H}^{n+1}(A) \times H^{n+1}(B) & \xrightarrow{\Delta} & H^{n+1}(C) \\ & \overset{d}{\nwarrow} & & & \\ \widetilde{H}^{n}(D) & \xrightarrow{i} & \widetilde{H}^{n}(A) \times H^{n}(B) & \xrightarrow{\Delta} & H^{n}(C) \\ & \overset{d}{\nwarrow} & & & \\ \widetilde{H}^{n-1}(D) & \xrightarrow{i} & \widetilde{H}^{n-1}(A) \times H^{n-1}(B) & \xrightarrow{\Delta} & H^{n-1}(C) \end{array} \tag{5.2.1}$$

Proposition 5.2.2. *The sequence 5.2.1 is a long exact sequence of groups.*

Proof. $\mathsf{Im}(d) \subset \mathsf{Ker}(i)$

The composition $i \circ d$ is equal to 0 because it only applies $\widetilde{d}$ to elements of the form $\mathsf{inl}(a)$ or $\mathsf{inr}(b)$.

$\mathsf{Ker}(i) \subset \mathsf{Im}(d)$

A map f from D to K_{n+1} is in the kernel of i if and only if both $f \circ \mathsf{inl}$ and $f \circ \mathsf{inr}$ are equal to 0. Given such a map we construct a map from C to ΩK_{n+1} by sending c to $\mathsf{ap}_f(\mathsf{push}(c))$ composed with the equalities from the hypothesis. This gives a map from C to K_n after composition with σ_n^{-1} and we can easily check that its image by f is equal to d.

$\mathsf{Im}(i) \subset \mathsf{Ker}(\Delta)$

Given $\delta : D \to K_n$, we have $\Delta(i(|\delta|)) = f^*(\mathsf{inl}^*(|\delta|)) - g^*(\mathsf{inr}^*(|\delta|))$. Moreover we have $f^*(\mathsf{inl}^*(|\delta|)) = \lambda c.\delta(\mathsf{inl}(f(c)))$ and $g^*(\mathsf{inr}^*(|\delta|)) = \lambda c.\delta(\mathsf{inr}(g(c)))$ which are equal via the homotopy $\lambda c.\, \mathsf{ap}_\delta(\mathsf{push}(c))$. Therefore $\Delta(i(|\delta|)) = 0$.

$\mathsf{Ker}(\Delta) \subset \mathsf{Im}(i)$

Given $\alpha : A \to K_n$, $\beta : B \to K_n$, and $\gamma : f^*(|\alpha|) = g^*(|\beta|)$, i.e. $\gamma : (c : C) \to \alpha(f(c)) = \beta(g(c))$, we construct $\delta : D \to K_n$ by

$$\begin{aligned} \delta(\mathsf{inl}(a)) &:= \alpha(a), \\ \delta(\mathsf{inr}(b)) &:= \beta(b), \\ \mathsf{ap}_\delta(\mathsf{push}(c)) &:= \gamma(c). \end{aligned}$$

The image of $|\delta|$ by i is $(|\alpha|, |\beta|)$, which is what we wanted.

$\mathsf{Im}(\Delta) \subset \mathsf{Ker}(d)$

Given $\alpha : A \to K_n$ and $\beta : B \to K_n$, we have

$$\begin{aligned} d(\Delta(|\alpha|, |\beta|)) &= d(f^*(|\alpha|) - g^*(|\beta|)) \\ &= d(f^*(|\alpha|)) - d(g^*(|\beta|)) \\ &= d(|\alpha \circ f|) - d(|\beta \circ g|) \\ &= \widetilde{d}(\alpha \circ f) - \widetilde{d}(\beta \circ g). \end{aligned}$$

We show that $\widetilde{d}(\alpha \circ f)$ is equal to 0, and a similar proof apply to $\widetilde{d}(\beta \circ g)$. We proceed by induction on the argument, which is of type D.

- For an element of the form $\mathsf{inl}(a)$, we use the path $\sigma_n(\alpha(a)) : \Omega K_{n+1}$ between $\widetilde{d}(\alpha \circ f)(\mathsf{inl}(a))$ and 0 and the fact that $\widetilde{d}(\alpha \circ f)(\mathsf{inl}(a))$ is equal to 0 by definition. It might be tempting to take the constant path instead, but then the induction wouldn't work for the $\mathsf{push}(c)$ case.
- For an element of the form $\mathsf{inr}(b)$, we use the constant path.
- For a path of the form $\mathsf{push}(c)$, we have to prove that $\sigma_n(\alpha(f(c)))$ is equal to the constant path, along $\mathsf{ap}_{\widetilde{d}(\alpha \circ f)}(\mathsf{push}(c))$, which is also equal to $\sigma_n(\alpha(f(c)))$, hence it works.

$\mathsf{Ker}(d) \subset \mathsf{Im}(\Delta)$

Given a map $\gamma : C \to K_n$ the hypothesis $\widetilde{d}(\gamma) = 0$ is an equality between two functions from D to K_{n+1}. If we look at what that means for each constructor of D, we get

$$\begin{aligned}
&\alpha : A \to \Omega K_{n+1},\\
&\beta : B \to \Omega K_{n+1},\\
&c : (c : C) \to \alpha(f(c))^{-1} \cdot \sigma_n(\gamma(c)) \cdot \beta(g(c)) = \mathsf{idp}.
\end{aligned}$$

This gives us α and β, and, moreover, we have $f^*(|\alpha|) - g^*(|\beta|) = |\gamma|$ because $\alpha(f(c)) \cdot \beta(g(c))^{-1} = \sigma_n(\gamma(c))$. □

In the special case where we have a pushout of the form $C_f = \mathbf{1} \sqcup^A B$ for some map $f : A \to B$, we obtain the following long exact sequence relating the cohomology of A and B with the reduced cohomology of C_f, where i is the inclusion $B \to C_f$.

$$\begin{array}{ccccccc}
\widetilde{H}^{n+1}(C_f) & \xrightarrow{i^*} & H^{n+1}(B) & \xrightarrow{f^*} & H^{n+1}(A) & & \\
 & \overset{d}{\nwarrow} & & & & & \\
\widetilde{H}^{n}(C_f) & \xrightarrow{i^*} & H^{n}(B) & \xrightarrow{f^*} & H^{n}(A) & & \\
 & \overset{d}{\nwarrow} & & & & & \\
\widetilde{H}^{n-1}(C_f) & \xrightarrow{i^*} & H^{n-1}(B) & \xrightarrow{f^*} & H^{n-1}(A) & &
\end{array}$$

We also obtain the unreduced Mayer–Vietoris sequence, which is formally identical to the Mayer–Vietoris sequence 5.2.1 with the exception that all reduced cohomology groups are replaced by regular cohomology groups. The idea is to apply the Mayer–Vietoris sequence 5.2.1 to $A' := \mathbf{1} + A$ and to notice that $\widetilde{H}^n(A') \simeq H^n(A)$ and that $\widetilde{H}^n(D') \simeq H^n(D)$ because

$$\begin{aligned}
D' &:= (\mathbf{1} + A) \sqcup^C B\\
&\simeq \mathbf{1} + (A \sqcup^C B)\\
&\simeq \mathbf{1} + D.
\end{aligned}$$

5.3 Cohomology of products of spheres

Proposition 5.3.1. *The cohomology groups of the point are*

$$H^k(\mathbf{1}) \simeq \begin{cases} \mathbb{Z} & \text{if } k = 0, \\ 0 & \text{otherwise.} \end{cases}$$

Proof. We have $(\mathbf{1} \to K_k) \simeq K_k$, which is equal to $\mathbb{Z}$ for $k = 0$ and is connected otherwise. □

The cohomology of spheres is easy to compute using the Mayer–Vietoris sequence.

Proposition 5.3.2. *For $n > 0$ we have*

$$H^k(\mathbb{S}^n) \simeq \begin{cases} \mathbb{Z} & \text{if } k = n \text{ or } k = 0, \\ 0 & \text{otherwise.} \end{cases}$$

We write $\mathbf{c}_n$ for the generator of $H^n(\mathbb{S}^n)$.

Proof. It follows easily, by induction, from the application of the Mayer–Vietoris sequence to the pushout $\mathbb{S}^n := \mathbf{1} \sqcup^{\mathbb{S}^{n-1}} \mathbf{1}$. □

Note that the cup product of $\mathbf{c}_n$ with itself in $H^n(\mathbb{S}^n)$ vanishes because it is an element of $H^{2n}(\mathbb{S}^n)$ which is the trivial group. We now look at the cohomology of a product of two spheres.

Proposition 5.3.3. *Given $n, k : \mathbb{N}$, the cohomology groups of $\mathbb{S}^n \times \mathbb{S}^k$ are generated (additively) by*

$$\begin{aligned} 1 &: H^0(\mathbb{S}^n \times \mathbb{S}^k), \\ \mathbf{x} &: H^n(\mathbb{S}^n \times \mathbb{S}^k), \\ \mathbf{y} &: H^k(\mathbb{S}^n \times \mathbb{S}^k), \\ \mathbf{z} &: H^{n+k}(\mathbb{S}^n \times \mathbb{S}^k). \end{aligned}$$

Note that n and k may be equal, which is why we state it in this way. Moreover in the diagram

$$\mathbb{S}^n \xrightarrow{i_n : x \mapsto (x, \star_{\mathbb{S}^k})} \mathbb{S}^n \times \mathbb{S}^k \xrightarrow{p_n : (x,y) \mapsto x} \mathbb{S}^n,$$

the map p_n^* sends $\mathbf{c}_n$ to $\mathbf{x}$ and the map i_n^* sends $\mathbf{x}$ (resp. $\mathbf{y}$) to $\mathbf{c}_n$ (resp. to 0). Similarly, in the diagram

$$\mathbb{S}^k \xrightarrow{i'_k : y \mapsto (\star_{\mathbb{S}^n}, y)} \mathbb{S}^n \times \mathbb{S}^k \xrightarrow{p'_k : (x,y) \mapsto y} \mathbb{S}^k,$$

the map ${p'_k}^*$ sends $\mathbf{c}_k$ to $\mathbf{y}$ and the map ${i'_k}^*$ sends $\mathbf{y}$ (resp. $\mathbf{x}$) to $\mathbf{c}_k$ (resp. to 0). Finally, we have

$$\begin{aligned} \mathbf{x} \smile \mathbf{x} &= 0, \\ \mathbf{y} \smile \mathbf{y} &= 0, \\ \mathbf{x} \smile \mathbf{y} &= \mathbf{z}. \end{aligned}$$

Proof. For the additive structure, we use the Mayer–Vietoris sequence and the fact that

$$\mathbb{S}^n \times \mathbb{S}^k \simeq \mathbf{1} \sqcup^{\mathbb{S}^{n+k-1}} (\mathbb{S}^n \vee \mathbb{S}^k),$$

which gives directly the first result together with the fact that i_n^* and ${i'_k}^*$ send $\mathbf{x}$ and $\mathbf{y}$ to $\mathbf{c}_n$ and $\mathbf{c}_k$ and the other one to 0. The two projections p_n^* and ${p'_k}^*$ send $\mathbf{c}_n$ and $\mathbf{c}_k$ back to $\mathbf{x}$ and $\mathbf{y}$ because we have $p_n \circ i_n = \mathsf{id}_{\mathbb{S}^n}$ and $p'_k \circ i'_k = \mathsf{id}_{\mathbb{S}^k}$. The Mayer–Vietoris sequence also shows that the map $\wedge_{n,k} \circ \mathsf{proj} : \mathbb{S}^n \times \mathbb{S}^k \to \mathbb{S}^n \wedge \mathbb{S}^k \to \mathbb{S}^{n+k}$ induces an isomorphism on H^{n+k}.

By definition, the cup product of $\mathbf{x}$ and $\mathbf{y}$ corresponds to the composition

$$\mathbb{S}^n \times \mathbb{S}^k \xrightarrow{(x,y)\mapsto(x,y)} \mathbb{S}^n \times \mathbb{S}^k \xrightarrow{\mathsf{proj}} \mathbb{S}^n \wedge \mathbb{S}^k \xrightarrow{\sim} \mathbb{S}^{n+k} \xrightarrow{|-|} K_{n+k}.$$

Note that the first map is the identity function because it's the pairing of the two projections, the first projection coming from $\mathbf{x}$ and the second from $\mathbf{y}$. Therefore, we have

$$\mathbf{x} \smile \mathbf{y} = \mathbf{z}.$$

The cup product of $\mathbf{x}$ with itself corresponds to the composition

$$\begin{array}{ccccccccc} \mathbb{S}^n \times \mathbb{S}^k & & \xrightarrow{(x,y)\mapsto(x,x)} & & \mathbb{S}^n \times \mathbb{S}^n & \xrightarrow{\mathsf{proj}} & \mathbb{S}^n \wedge \mathbb{S}^n & \xrightarrow{\sim} & \mathbb{S}^{2n} \xrightarrow{|-|} K_{2n}. \\ & \searrow^{(x,y)\mapsto x} & & \nearrow_{x\mapsto(x,x)} & & & & & \\ & & \mathbb{S}^n & & & & & & \end{array}$$

But the first map factors through $\mathbb{S}^n$ which has no cohomology in dimension $2n$, therefore

$$\mathbf{x} \smile \mathbf{x} = 0,$$

and similarly

$$\mathbf{y} \smile \mathbf{y} = 0. \qquad \square$$

5.4 The Hopf invariant

Given a map $f : \mathbb{S}^k \to \mathbb{S}^n$, we can consider the pushout $\mathbf{1} \sqcup^{\mathbb{S}^k} \mathbb{S}^n$, which has cohomology $\mathbb{Z}$ in dimensions 0, n and $k+1$ (using the Mayer–Vietoris sequence). The cup product structure on this space is often trivial, unless $k = 2n-1$ in which case the square of the generator in dimension n may be a nontrivial multiple of the generator in dimension $2n$. This is what we study in this section.

Definition 5.4.1. Given a pointed map $f : \mathbb{S}^{2n-1} \to \mathbb{S}^n$, we define

$$\begin{aligned} C_f &:= \mathbf{1} \sqcup^{\mathbb{S}^{2n-1}} \mathbb{S}^n, \\ \alpha_f &:= (i^*)^{-1}(\mathbf{c}_n) : H^n(C_f), \\ \beta_f &:= p^*(\mathbf{c}_{2n}) : H^{2n}(C_f), \end{aligned}$$

where $i : \mathbb{S}^n \to C_f$ is the inclusion on the right and $p : C_f \to \mathbb{S}^{2n}$ is the map collapsing the $\mathbb{S}^n$ term in $\mathbf{1} \sqcup^{\mathbb{S}^{2n-1}} \mathbb{S}^n$ to a point. The Mayer–Vietoris sequence shows that the map i^* induces an isomorphism on H^n and that the map p^* induces an isomorphism on H^{2n}, hence the elements α_f and β_f are generators of the respective cohomology groups.

Definition 5.4.2. The *Hopf invariant* of a pointed map $f : \mathbb{S}^{2n-1} \to \mathbb{S}^n$ is the integer $H(f) : \mathbb{Z}$ such that

$$\alpha_f^2 = H(f)\beta_f,$$

where α_f^2 is $\alpha_f \smile \alpha_f$.

Note that, if n is odd, we have $\alpha_f^2 = -\alpha_f^2$ by graded-commutativity; therefore, the Hopf invariant of any map $f : \mathbb{S}^{2n-1} \to \mathbb{S}^n$ is 0.

Proposition 5.4.3. *The Hopf invariant $H : \pi_{2n-1}(\mathbb{S}^n) \to \mathbb{Z}$ is a homomorphism of groups.*

Proof. Let f and g be two pointed maps from $\mathbb{S}^{2n-1}$ to $\mathbb{S}^n$. The sum of f and g when seen as elements of $\pi_{2n-1}(\mathbb{S}^n)$ is represented by the map

$$f + g : \ \mathbb{S}^{2n-1} \xrightarrow{\ \theta\ } \mathbb{S}^{2n-1} \vee \mathbb{S}^{2n-1} \xrightarrow{\ f \vee g\ } \mathbb{S}^n.$$

Let's consider the type $C_{f\vee g}$ defined by the pushout

$$\begin{array}{ccc} \mathbb{S}^{2n-1} \vee \mathbb{S}^{2n-1} & \xrightarrow{\ f\vee g\ } & \mathbb{S}^n \\ \downarrow & & \vdots\, i \\ \mathbf{1} & \dashrightarrow & C_{f\vee g} \end{array}$$

The cohomology groups of $C_{f\vee g}$ are $\mathbb{Z}$ in dimensions 0 and n, $\mathbb{Z}^2$ in dimension $2n$, and the trivial group otherwise. We denote by $\alpha_{f\vee g}$ the generator in dimension n coming from i and by $\beta^f_{f\vee g}$ and $\beta^g_{f\vee g}$ the two generators in dimension $2n$ coming from the two $\mathbb{S}^{2n-1}$ terms. The three maps θ, inl and inr of type $\mathbb{S}^{2n-1} \to \mathbb{S}^{2n-1} \vee \mathbb{S}^{2n-1}$ induce three maps $q : C_{f+g} \to C_{f\vee g}$, $j_f : C_f \to C_{f\vee g}$ and $j_g : C_g \to C_{f\vee g}$ satisfying

$$\begin{array}{lll} q^*(\alpha_{f\vee g}) = \alpha_{f+g}, & j_f^*(\alpha_{f\vee g}) = \alpha_f, & j_g^*(\alpha_{f\vee g}) = \alpha_g, \\ q^*(\beta^f_{f\vee g}) = \beta_{f+g}, & j_f^*(\beta^f_{f\vee g}) = \beta_f, & j_g^*(\beta^f_{f\vee g}) = 0, \\ q^*(\beta^g_{f\vee g}) = \beta_{f+g}, & j_f^*(\beta^g_{f\vee g}) = 0, & j_g^*(\beta^g_{f\vee g}) = \beta_g. \end{array}$$

Given that the square of $\alpha_{f\vee g}$ is an element of $H^{2n}(C_{f\vee g})$, it is a linear combination of $\beta^f_{f\vee g}$ and $\beta^g_{f\vee g}$ i.e. there are two natural numbers x and y such that

$$\alpha^2_{f\vee g} = x\beta^f_{f\vee g} + y\beta^g_{f\vee g}.$$

By applying j_f^* to this equation we obtain $\alpha_f^2 = x\beta_f$, hence $x = H(f)$. Similarly we get $y = H(g)$. Therefore, we can compute the square of α_{f+g}:

$$\begin{aligned}\alpha_{f+g}^2 &= q^*(\alpha_{f\vee g})^2 \\ &= q^*(\alpha_{f\vee g}^2) \\ &= q^*(H(f)\beta_{f\vee g}^f + H(g)\beta_{f\vee g}^g) \\ &= (H(f) + H(g))\beta_{f+g}.\end{aligned}$$

This shows that $H(f+g) = H(f) + H(g)$ and that H is a group homomorphism. □

Proposition 5.4.4. *If n is even, then the Hopf invariant of the map $\nabla_{\mathbb{S}^n} \circ W_{n,n} : \mathbb{S}^{2n-1} \to \mathbb{S}^n$ is equal to 2.*

Proof. We consider the space $C_{\nabla_{\mathbb{S}^n}\circ W_{n,n}}$ and we write $\alpha := \alpha_{\nabla_{\mathbb{S}^n}\circ W_{n,n}}$ and $\beta := \beta_{\nabla_{\mathbb{S}^n}\circ W_{n,n}}$ for short. We saw in proposition 3.4.4 that $C_{\nabla_{\mathbb{S}^n}\circ W_{n,n}}$ is equivalent to the pushout

$$\begin{array}{ccc} \mathbb{S}^n \vee \mathbb{S}^n & \longrightarrow & \mathbb{S}^n \times \mathbb{S}^n \\ \downarrow & & \downarrow q \\ \mathbb{S}^n & \xrightarrow[i]{} & J_2(\mathbb{S}^n) \end{array}$$

via the sequence of equivalences

$$\begin{aligned}C_{\nabla\circ W_{n,n}} &\simeq \mathbf{1} \sqcup^{\mathbb{S}^{2n-1}} \mathbb{S}^n \\ &\simeq (\mathbf{1} \sqcup^{\mathbb{S}^{2n-1}} (\mathbb{S}^n \vee \mathbb{S}^n)) \sqcup^{\mathbb{S}^n\vee\mathbb{S}^n} \mathbb{S}^n \\ &\simeq (\mathbb{S}^n \times \mathbb{S}^n) \sqcup^{\mathbb{S}^n\vee\mathbb{S}^n} \mathbb{S}^n \\ &\simeq J_2(\mathbb{S}^n).\end{aligned}$$

This shows that the map $i^* : H^n(J_2(\mathbb{S}^n)) \to H^n(\mathbb{S}^n)$ sends α to $\mathbf{c}_n$ and the map $q^* : H^{2n}(J_2(\mathbb{S}^n)) \to H^{2n}(\mathbb{S}^n \times \mathbb{S}^n)$ sends β to $\mathbf{x} \smile \mathbf{y}$, where $\mathbf{x}$ and $\mathbf{y}$ are the two generators of $H^n(\mathbb{S}^n \times \mathbb{S}^n)$ given by proposition 5.3.3.

The composition of $q : \mathbb{S}^n \times \mathbb{S}^n \to J_2(\mathbb{S}^n)$ with any of the two inclusions $\mathbb{S}^n \to \mathbb{S}^n \times \mathbb{S}^n$ is homotopic to the map $i : \mathbb{S}^n \to J_2(\mathbb{S}^n)$. Therefore, $q^*(\alpha) = \mathbf{x} + \mathbf{y} : H^n(\mathbb{S}^n \times \mathbb{S}^n)$. Moreover we have $\mathbf{x} \smile \mathbf{y} = \mathbf{y} \smile \mathbf{x}$ by graded-commutativity, because we assumed n even, and then

$$\begin{aligned}q^*(\alpha^2) &= q^*(\alpha)^2 \\ &= (\mathbf{x} + \mathbf{y}) \smile (\mathbf{x} + \mathbf{y}) \\ &= (\mathbf{x} \smile \mathbf{x}) + (\mathbf{x} \smile \mathbf{y}) + (\mathbf{y} \smile \mathbf{x}) + (\mathbf{y} \smile \mathbf{y}) \\ &= 2(\mathbf{x} \smile \mathbf{y}) \\ &= 2q^*(\beta) \\ &= q^*(2\beta).\end{aligned}$$

Finally, we know that q^* is an equivalence on H^{2n}, which shows that $\alpha^2 = 2\beta$ and that the Hopf invariant of the map $\nabla_{\mathbb{S}^n} \circ W_{n,n} : \mathbb{S}^{2n-1} \to \mathbb{S}^n$ is equal to 2. □

This allows us to define new non-trivial elements in the homotopy groups of spheres.

Proposition 5.4.5. *For every $n \geq 1$, the group $\pi_{4n-1}(\mathbb{S}^{2n})$ is infinite.*

Proof. We define a group homomorphism h from $\mathbb{Z}$ to $\pi_{4n-1}(\mathbb{S}^{2n})$ sending 1 to the element corresponding to the function $\nabla \circ W_{2n,2n} : \mathbb{S}^{4n-1} \to \mathbb{S}^{2n}$. For every $n : \mathbb{Z}$ we have $H(h(n)) = 2n$, hence all the $h(n)$ are different, which shows that $\pi_{4n-1}(\mathbb{S}^{2n})$ is infinite. □

Proposition 5.4.6. *The natural number n satisfying $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$ is equal to either 1 or 2.*

Proof. By definition of n in proposition 3.4.5, we have $[i_2, i_2] = \pm n\eta$ in $\pi_3(\mathbb{S}^2)$, for η a generator of $\pi_3(\mathbb{S}^2)$. Applying the Hopf invariant to this equality we get

$$2 = H([i_2, i_2]) = H(\pm n\eta) = \pm nH(\eta),$$

which shows that n divides 2 and that, therefore, it is equal to either 1 or 2. □

What is missing to prove that n is equal to 2 is to know whether there is an element of $\pi_3(\mathbb{S}^2)$ of Hopf invariant 1. More exactly we have the following.

Corollary 5.4.7. *We have $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$ if and only if there exists a map $\mathbb{S}^3 \to \mathbb{S}^2$ of Hopf invariant ± 1. Otherwise $\pi_4(\mathbb{S}^3)$ is the trivial group.*

Proof. If $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$, then the computation above shows that any generator of $\pi_3(\mathbb{S}^2)$ has Hopf invariant ± 1. Conversely, if there is a map $\eta : \mathbb{S}^3 \to \mathbb{S}^2$ of Hopf invariant ± 1, then the Hopf invariant homomorphism $H : \pi_3(\mathbb{S}^2) \to \mathbb{Z}$ sends η to ± 1, which shows that η is a generator of $\pi_3(\mathbb{S}^2)$. The computation above then shows that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$. □

Chapter 6

The Gysin sequence

We now present a tool called *the Gysin sequence*, which gives some information on the cup product structure of a space, given a fibration of spheres over it. More precisely, given a fibration

$$\mathbb{S}^{n-1} \longrightarrow E \xrightarrow{p} B$$

where B is 1-connected, we prove that there is an element $e : H^n(B)$ and a long exact sequence

$$\ldots \longrightarrow H^{i-1}(E) \longrightarrow H^{i-n}(B) \xrightarrow{\smile e} H^i(B) \xrightarrow{p^*} H^i(E) \longrightarrow \ldots,$$

where the middle arrow is the operation of cup product by e. We then define the complex projective plane $\mathbb{C}P^2$ using the Hopf map, and we construct a fibration of circles over $\mathbb{C}P^2$ with total space $\mathbb{S}^5$. The Gysin sequence allows us to compute the cohomology of $\mathbb{C}P^2$, to show that the Hopf invariant of the Hopf map is ± 1, and, therefore, that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$.

The projective spaces $\mathbb{C}P^n$ have also independently been defined in homotopy type theory by Ulrik Buchholtz and Egbert Rijke.

6.1 The Gysin sequence

The following proposition is the main ingredient in the construction of the Gysin sequence. It relates the cup product in dimensions (n, m) with the one in dimensions $(n + 1, m)$.

Proposition 6.1.1. *Given $n, m : \mathbb{N}$, $p : K_n$ and $y : K_m$, we have the equality*

$$\mathsf{ap}_{\lambda x.x \smile y}(\sigma_n(p)) = \sigma_{n+m}(p \smile y).$$

Note that the cup product on the left is the one in dimensions $(n + 1, m)$ while the one on the right is the one in dimensions (n, m).

Proof. For $n = 0$, we have $p : \mathbb{Z}$ and we write $k := p$ in order to remember that it is an integer. We then have $\sigma_0(p) = \mathsf{ap}_{|-|}(\mathsf{loop})^k$ and what we want to prove is

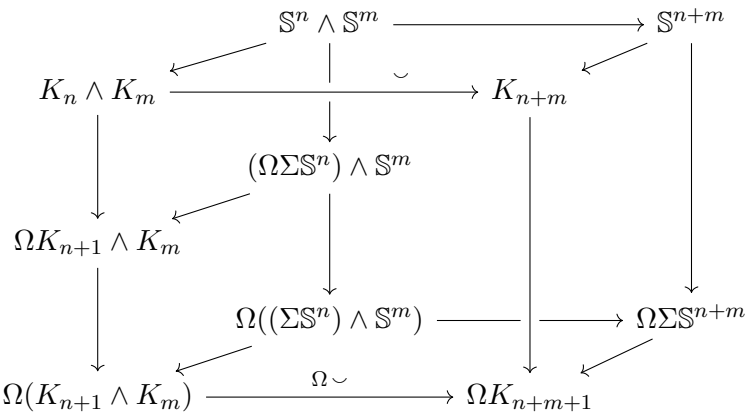
$$\mathsf{ap}_{\lambda x.x \smile y}(\mathsf{ap}_{|-|}(\mathsf{loop})^k) = \sigma_m(ky).$$

The left-hand side is equal to $\mathsf{ap}_{\lambda x.|x| \smile y}(\mathsf{loop})^k$ and the right-hand side is equal to $\sigma_m(y)^k$. Therefore it is enough to prove that $\mathsf{ap}_{\lambda x.|x| \smile y}(\mathsf{loop}) = \sigma_m(y)$ for all $y : \|\mathbb{S}^m\|_m$. What we have to prove is an $(m-1)$-type, so we can assume that y is of the form $|z|$ for $z : \mathbb{S}^m$. Let's consider the diagram

$$\begin{array}{ccc} \Omega(\mathbb{S}^1 \wedge \mathbb{S}^m) & \longrightarrow & \Omega\mathbb{S}^{m+1} \\ \downarrow & & \downarrow \\ \Omega(K_1 \wedge K_m) & \longrightarrow & \Omega K_{m+1} \end{array}$$

which is obtained by looping the diagram defining the cup product in definition 5.1.6. If we start with $\mathsf{ap}_{\mathsf{proj}(-,z)}(\mathsf{loop})$ on the top left and go down and then right, we obtain $\mathsf{ap}_{\lambda x.|x| \smile |z|}(\mathsf{loop})$ while going right and then down gives $\mathsf{ap}_{|-|_m}(\varphi_{\mathbb{S}^m}(z))$ which is equal to $\sigma_m(|z|)$.

For $n > 0$, we want to fill the front face of the diagram



where the two vertical maps of the form $\Omega A \wedge B \to \Omega(A \wedge B)$ send (p, b) to $\mathsf{ap}_{\mathsf{proj}(-,b)}(p)$. The top and bottom squares are filled by definition of the cup product. The rectangle on the right and the upper-left square are filled by definition of σ_n. The lower-left square is filled by naturality of the family of maps $\Omega A \wedge B \to \Omega(A \wedge B)$. The back rectangle is filled by proposition 4.2.3.

Therefore, given that the map $\mathbb{S}^n \wedge \mathbb{S}^m \to K_n \wedge K_m$ is $(n + m)$-connected and that what we want to prove is $(n + m)$-truncated, the result follows. □

The main technical result of this section is the following.

Proposition 6.1.2. *Given a connected pointed type B, a family of pointed types $Q : B \to \mathsf{Type}_\star$ (i.e. a family of types equipped with a section) such that $Q(\star_B) = \mathbb{S}^n$, and a map*

$$c : (b : B) \to (Q(b) \to_\star K_n)$$

such that $c_{\star_B} : \mathbb{S}^n \to_\star K_n$ is a generator of $(\mathbb{S}^n \to_\star K_n) \simeq \mathbb{Z}$, then the map

$$\begin{aligned} \Phi : H^i(B) &\to \|(b : B) \to (Q(b) \to_\star K_{i+n})\|_0\,, \\ \Phi(|\beta|) &:= |\lambda b, x.\beta(b) \smile c_b(x)| \end{aligned}$$

is an equivalence. Moreover, such a c always exists if B is 1-connected.

Proof. The result is actually true for the untruncated map

$$\begin{aligned} \widetilde{\Phi} : (B \to K_i) &\to ((b : B) \to (Q(b) \to_\star K_{i+n})), \\ \widetilde{\Phi}(\beta) &:= \lambda b, x.\beta(b) \smile c_b(x) \end{aligned}$$

and for all the fiberwise maps

$$\begin{aligned} g_b^i : K_i &\to (Q(b) \to_\star K_{i+n}), \\ g_b^i(y) &:= \lambda x.y \smile c_b(x). \end{aligned}$$

Let's prove that all the g_b^i are equivalences. Given that B is connected, it is enough to do it for $b = \star_B$. We proceed by induction on i. For $i = 0$, we have $K_0 = \mathbb{Z}$, and

$$\begin{aligned} g^0_{\star_B} : \mathbb{Z} &\to (\mathbb{S}^n \to_\star K_n) \\ k &\mapsto (\lambda y.k \smile c_{\star_B}(y)) \\ &= \lambda y.k(c_{\star_B}(y)) \\ &= k(\lambda y.c_{\star_B}(y)) \\ &= kc_{\star_B}, \end{aligned}$$

which is an equivalence by assumption.

We now assume that $g^i_{\star_B}$ is an equivalence, and we want to prove that $g^{i+1}_{\star_B}$ is also an equivalence. Note first that K_{i+1} and $\mathbb{S}^n \to_\star K_{i+1+n}$ are both pointed and connected, hence it's enough to prove that $\Omega g^{i+1}_{\star_B}$ is an equivalence in order to show that $g^{i+1}_{\star_B}$ is, by [UF13, theorem 8.8.1]. We will also use the fact that given a function $f : A \to B \to C$ and a path $p : b = b'$, there is an equality

$$\mathsf{ap}_{\lambda y.(\lambda x.f(x,y))}(p) = \mathsf{funext}(\lambda x.\, \mathsf{ap}_{\lambda y.f(x,y)}(p))$$

in the type $(\lambda x.f(x,b)) = (\lambda x.f(x,b'))$. This equality is proven by path induction on p.

Given $p : \Omega K_{i+1}$, we have

$$\begin{aligned}
(\Omega g^{i+1}_{\star_B})(p) &= \mathsf{ap}_{g^{i+1}_{\star_B}}(p) \\
&= \mathsf{ap}_{\lambda y.(\lambda x.y \smile c_{\star_B}(x))}(p) \\
&= \mathsf{funext}(\lambda x.\, \mathsf{ap}_{\lambda y.y \smile c_{\star_B}(x)}(p)) \\
&= \mathsf{funext}(\lambda x.\, \mathsf{ap}_{\lambda y.y \smile c_{\star_B}(x)}(\sigma_n(\sigma_n^{-1}(p)))) \\
&= \mathsf{funext}(\lambda x.\sigma_{n+m}(\sigma_n^{-1}(p) \smile c_{\star_B}(x))) \quad \text{by proposition 6.1.1} \\
&= \mathsf{funext}(\sigma_{n+m} \circ g^i_{\star_B}(\sigma_n^{-1}(p))).
\end{aligned}$$

Therefore, the map $\Omega g^{i+1}_{\star_B}$ is equal to the composition

$$(\Omega g^{i+1}_{\star_B}) = \mathsf{funext} \circ (\lambda f.\sigma_{n+m} \circ f) \circ g^i_{\star_B} \circ \sigma_n^{-1}.$$

All those functions are equivalences, hence $\Omega g^{i+1}_{\star_B}$ and then $g^{i+1}_{\star_B}$ are equivalences as well. We have proved that g^i_b is an equivalence. We now have

$$\widetilde{\Phi}(\beta) = \lambda b.g^i_b(\beta(b)),$$

therefore $\widetilde{\Phi}$ is an equivalence of inverse

$$\widetilde{\Phi}^{-1}(\beta') = \lambda b.(g^i_b)^{-1}(\beta'(b)),$$

and Φ is an equivalence as well.

Let's now prove that c exists whenever B is 1-connected. The type $(Q(b) \to_\star K_n)$ is a set for $b = \star_B$, hence it's a set for every $b : B$, given that B is 0-connected. Now, if B is 1-connected, the map $\mathbf{1} \to B$ is 0-connected, so it's enough to define c on the base point of B, and in this case we choose for $c_{\star_B}$ any generator of $(\mathbb{S}^n \to_\star K_n) \simeq \mathbb{Z}$. □

We now apply this result to the special case of the fibration obtained by taking the fiberwise suspension of a family of spheres. This gives what is known as the *Thom isomorphism*.

Let B be a 0-connected pointed type and $P : B \to \mathsf{Type}$ a fibration over B such that $P(\star_B) = \mathbb{S}^{n-1}$. We define $E := \sum_{x:B} P(x)$ its total space and $\widetilde{E} := \mathbf{1} \sqcup^E B$ the cofiber of the map $E \to B$, and we have a map $i : \mathbb{S}^n \to \widetilde{E}$ corresponding to the fiber $\mathbb{S}^{n-1}$ over $\star_B$. We also define a family of pointed types $Q : B \to \mathsf{Type}_\star$ by $Q(b) := \Sigma(P(b))$ pointed by the north pole, $F := \sum_{x:B} Q(x)$ its total space and $\widetilde{F} := \mathbf{1} \sqcup^B F$ the cofiber of the map $B \to F$ sending b to $(b, \star_{Q(b)})$. It is easy to see that $\widetilde{E}$ and $\widetilde{F}$ are equivalent, with all identified north poles in $\widetilde{F}$ corresponding to the basepoint in $\widetilde{E}$ and all south poles corresponding to the B term in $\widetilde{E}$. For every pointed type A we have an equivalence

$$(\widetilde{F} \to_\star A) \simeq ((b : B) \to (Q(b) \to_\star A))$$

because a pointed map from $\widetilde{F}$ to A corresponds to a family of pointed maps from all $Q(b)$ to A. Therefore, we have an equivalence

$$\iota_k : ((b : B) \to (Q(b) \to_\star K_k)) \simeq (\widetilde{E} \to_\star K_k)$$

for every $k : \mathbb{N}$.

Proposition 6.1.3 (Thom isomorphism)**.** *Given a map* $c : (b : B) \to (Q(b) \to_\star K_n)$ *such that* $c_{\star_B}$ *is a generator of* $H^n(\mathbb{S}^n)$*, the map*

$$\begin{aligned}\Phi : H^i(B) &\to \widetilde{H}^{i+n}(\widetilde{E}),\\ \Phi(|\beta|) &:= |\iota_{i+n}(\lambda b, x.\beta(b) \smile c_b(x))|\end{aligned}$$

is an isomorphism of groups. Moreover, such a c *always exists if* B *is 1-connected.*

Proof. Applying proposition 6.1.2 to Q, we obtain that Φ is an equivalence and that c always exists if B is 1-connected. The map Φ is a group homomorphism by distributivity of the cup product and because ι_{i+n} preserves the group structure. □

We can finally construct the Gysin sequence. Let's consider a 1-connected pointed type B and a fibration $P : B \to \mathsf{Type}$ such that $P(\star_B) = \mathbb{S}^{n-1}$. We write E for the total space of P, $p : E \to B$ for the projection and $\widetilde{E}$ for the cofiber of p. The long exact sequence of the cofiber of p is

$$\ldots \longrightarrow \widetilde{H}^i(\widetilde{E}) \xrightarrow{j^*} H^i(B) \xrightarrow{p^*} H^i(E) \longrightarrow \widetilde{H}^{i+1}(\widetilde{E}) \longrightarrow \ldots,$$

where j is the inclusion $B \to \widetilde{E}$. The Thom isomorphism gives a map $c : (b : B) \to (Q(b) \to_\star K_n)$ together with an isomorphism $\Phi : H^{i-n}(B) \simeq \widetilde{H}^i(\widetilde{E})$, and the induced map $H^{i-n}(B) \to H^i(B)$ is

$$\begin{aligned}H^{i-n}(B) &\to H^i(B),\\ |\beta| &\mapsto j^*(\Phi(|\beta|))\\ &= j^*(|\iota_{i+n}(\lambda b, x.\beta(b) \smile c_b(x))|)\\ &= |\lambda b'.\iota_{i+n}(\lambda b, x.\beta(b) \smile c_b(x))(j(b'))|\\ &= |\lambda b'.(\lambda b, x.\beta(b) \smile c_b(x))(b', \mathsf{south}_{Q(b')})|\\ &= |\lambda b'.\beta(b') \smile c_{b'}(\mathsf{south}_{Q(b')})|\\ &= |\beta| \smile e,\end{aligned}$$

where

$$\begin{aligned}&e : H^n(B),\\ &e := |\lambda b.c_b(\mathsf{south}_{Q(b)})|.\end{aligned}$$

Therefore we get the Gysin sequence.

Proposition 6.1.4 (Gysin sequence)**.** *There is an element* $e : H^n(B)$ *and a long exact sequence*

$$\ldots \longrightarrow H^{i-1}(E) \longrightarrow H^{i-n}(B) \xrightarrow{\smile e} H^i(B) \xrightarrow{p^*} H^i(E) \longrightarrow \ldots \qquad \square$$

6.2 The iterated Hopf construction

We saw in chapter 2 that to every H-space A we can associate a fibration over ΣA with fiber A and total space $A * A$. We now prove that if we assume additionally that the H-space structure is associative, then we can iterate this construction once, by constructing a fibration over $\mathbf{1} \sqcup^{A*A} \Sigma A$ with fiber A and total space $A * A * A$.

Definition 6.2.1. An *associative H-space* is an H-space (A, μ) equipped with a map

$$\alpha : (x, y, z : A) \to \mu(\mu(x, y), z) = \mu(x, \mu(y, z))$$

and a filler of the diagram

$$\begin{array}{ccc} \mu(\mu(\star_A, y), z) & \xrightarrow{\alpha_{\star_A, y, z}} & \mu(\star_A, \mu(y, z)) \\ & {\scriptstyle \mathsf{ap}_{\mu(-,z)}(\mu_l(y))} \searrow \quad \swarrow {\scriptstyle \mu_l(\mu(y,z))} & \\ & \mu(y, z) & \end{array}$$

If A is connected, then the three other triangles follow from this one, but we do not need them here.

Proposition 6.2.2. *Given a connected associative H-space A, there is a fibration over $\mathbf{1} \sqcup^{A*A} \Sigma A$ (where the map $A * A \to \Sigma A$ is induced by the Hopf construction on A) with fiber A, and whose total space is equivalent to $A * A * A$.*

Proof. We recall that the Hopf construction is the fibration $H : \Sigma A \to \mathsf{Type}$ defined by

$$\begin{aligned} H(\mathsf{north}) &:= A, \\ H(\mathsf{south}) &:= A, \\ \mathsf{ap}_H(\mathsf{merid}(y)) &:= \mathsf{ua}(\mu(-, y)). \end{aligned}$$

Its total space is equivalent to the pushout $A \sqcup^{A \times A} A$ of the span

$$A \xleftarrow{\mathsf{fst}} A \times A \xrightarrow{\mu} A$$

via the map

$$\begin{aligned} \psi : A \sqcup^{A \times A} A &\to \sum_{x : \Sigma A} H(x), \\ \psi(\mathsf{inl}(x)) &:= (\mathsf{north}, x), \\ \psi(\mathsf{inr}(z)) &:= (\mathsf{south}, z), \\ \mathsf{ap}_\psi(\mathsf{push}(x, y)) &:= (\mathsf{merid}(y), \psi_{\mathsf{push}}(x, y)), \end{aligned}$$

where $\psi_{\mathsf{push}}(x, y) : x =^H_{\mathsf{merid}(y)} \mu(x, y)$ comes from the fact that $\mathsf{transport}^H(\mathsf{merid}(y), x) = \mu(x, y)$. The induced map $h : A \sqcup^{A \times A} A \to \Sigma A$ is defined by

$$\begin{aligned} h(\mathsf{inl}(x)) &:= \mathsf{north}, \\ h(\mathsf{inr}(z)) &:= \mathsf{south}, \\ \mathsf{ap}_h(\mathsf{push}(x, y)) &:= \mathsf{merid}(y). \end{aligned}$$

We first prove the following proposition.

Proposition 6.2.3. *There is a map* $\nu : (x : A \sqcup^{A\times A} A) \to A \simeq H(h(x))$ *such that for every* $a : A$*, the map*

$$\nu'_a : A \sqcup^{A\times A} A \to \sum_{x:\Sigma A} H(x),$$
$$\nu'_a(x) := (h(x), \nu(x, a))$$

is an equivalence.

Proof. We define ν by

$$\begin{aligned} \nu(\mathsf{inl}(x)) &:= \mu(-, x), \\ \nu(\mathsf{inr}(z)) &:= \mu(-, z), \\ \mathsf{ap}_\nu(\mathsf{push}(x, y)) &:= \nu_{\mathsf{push}}(x, y), \end{aligned}$$

where $\nu_{\mathsf{push}}(x, y)$ is a filling of the square

$$\begin{array}{ccc} A & \xrightarrow{\mathsf{id}_A} & A \\ {\scriptstyle \mu(-,x)}\downarrow & & \downarrow{\scriptstyle \mu(-,\mu(x,y))} \\ A & \xrightarrow[\mu(-,y)]{} & A \end{array}$$

In other words, $\nu_{\mathsf{push}}(x, y)$ proves that for every $a : A$, we have

$$\mu(\mu(a, x), y) = \mu(a, \mu(x, y)),$$

which we have by associativity of the H-space structure.

In order to prove that ν'_a is an equivalence for every $a : A$, it is enough to do it in the case $a = \star_A$ because A is connected. We now show that $\nu'_{\star_A}$ is equal to the function $\psi : A \sqcup^{A\times A} A \to \sum_{x:\Sigma A} H(x)$ by induction on the argument.

- For $\mathsf{inl}(x)$, we have

$$\nu'_{\star_A}(\mathsf{inl}(x)) = (\mathsf{north}, \mu(\star_A, x)) \overset{\text{via } \mu_l(x)}{=} (\mathsf{north}, x) = \psi(\mathsf{inl}(x)).$$

- For $\mathsf{inr}(z)$, we have

$$\nu'_{\star_A}(\mathsf{inr}(z)) = (\mathsf{south}, \mu(\star_A, z)) \overset{\text{via } \mu_l(z)}{=} (\mathsf{south}, z) = \psi(\mathsf{inr}(z)).$$

- For $\mathsf{push}(x, y)$, using the fact that $\mathsf{ap}_H(\mathsf{merid}(y)) = \mu(-, y)$ we have to give a filling of the square

$$\begin{array}{ccc} \mu(\mu(\star_A, x), y) & \longrightarrow & \mu(\star_A, \mu(x, y)) \\ \downarrow & & \downarrow \\ \mu(x, y) & = & \mu(x, y) \end{array}$$

  which follows from the fact that A is an associative H-space. □

We now define $P : \mathbf{1} \sqcup^{A \sqcup^{A\times A} A} \Sigma A \to \mathsf{Type}$ by

$$\begin{aligned} P(\mathsf{inl}(\star_{\mathbf{1}})) &:= A, \\ P(\mathsf{inr}(y)) &:= H(y), \\ \mathsf{ap}_P(\mathsf{push}(x)) &:= \nu(x). \end{aligned}$$

The total space of P is equivalent to the pushout of the span

$$A \xleftarrow{\mathsf{fst}} A \times (A \sqcup^{A\times A} A) \xrightarrow{(a,x)\mapsto \nu'_a(x)} \sum_{x:\Sigma A} H(x),$$

and we have the equivalence of spans

$$\begin{array}{ccccc} A & \xleftarrow{\mathsf{fst}} & A \times (A \sqcup^{A\times A} A) & \xrightarrow{(a,x)\mapsto \nu'_a(x)} & \sum_{x:\Sigma A} H(x) \\ \downarrow{\scriptstyle \mathsf{id}} & & \downarrow{\scriptstyle (a,x)\mapsto(a,\nu'_a(x))} & & \downarrow{\scriptstyle \mathsf{id}} \\ A & \xleftarrow[\mathsf{fst}]{} & A \times \sum_{x:\Sigma A} H(x) & \xrightarrow[\mathsf{snd}]{} & \sum_{x:\Sigma A} H(x) \end{array}$$

where the inverse of the middle map is the map $(a, y) \mapsto (a, {\nu'_a}^{-1}(y))$. Therefore, the total space of P is equivalent to the join of A and $\sum_{x:\Sigma A} H(x)$. But we already know that the total space of H is equivalent to $A * A$; hence, the total space of P is equivalent to $A * A * A$. □

6.3 The complex projective plane

Proposition 6.3.1. *The H-space structure on the circle is associative.*

Proof. We want to prove that for every $x, y, z : \mathbb{S}^1$, we have $\mu(\mu(x, y), z) = \mu(x, \mu(y, z))$. We proceed first by induction on x.

For base, we have $\mu(\mu(\mathsf{base}, y), z) = \mu(y, z) = \mu(\mathsf{base}, \mu(y, z))$ by definition, so we can just use the constant path. For loop, we have to prove that $\mathsf{ap}_{\mu(-,z)}(\mathsf{loop}_y) = \mathsf{loop}_{\mu(y,z)}$. We proceed again by induction on y.

For base we have $\mathsf{ap}_{\mu(-,z)}(\mathsf{loop}_{\mathsf{base}}) = \mathsf{ap}_{\mu(-,z)}(\mathsf{loop}) = \mathsf{loop}_z$ and $\mathsf{loop}_{\mu(\mathsf{base},z)} = \mathsf{loop}_z$, so it's true, and for loop we have to construct a 3-dimensional path in $\mathbb{S}^1$ and $\mathbb{S}^1$ is 1-connected, so it's immediate. Moreover the commutativity of the triangle is immediate because both $\alpha_{\mathsf{base},y,z}$ and $\mu_l(-)$ are constant paths. □

We now define the complex projective plane as $\mathbb{C}P^2 := \mathbf{1} \sqcup^{\mathbb{S}^3} \mathbb{S}^2$, where the map $\mathbb{S}^3 \to \mathbb{S}^2$ is the Hopf map. Its cohomology groups are $\mathbb{Z}$ in dimensions 0, 2 and 4, and the trivial group otherwise, and the iterated Hopf construction on the circle gives us a fibration K over $\mathbb{C}P^2$ with fiber $\mathbb{S}^1$ and total space $\mathbb{S}^1 * \mathbb{S}^1 * \mathbb{S}^1$, which is equivalent to $\mathbb{S}^5$.

Proposition 6.3.2. *The Hopf invariant of the Hopf map $\mathbb{S}^3 \to \mathbb{S}^2$ is equal to ± 1.*

v

Proof. We apply the Gysin sequence to the fibration over $\mathbb{C}P^2$ defined by the iterated Hopf construction as above. We obtain a cohomology class $e : H^2(\mathbb{C}P^2)$ and the two short exact sequences

$$0 \simeq H^1(\mathbb{S}^5) \longrightarrow H^0(\mathbb{C}P^2) \xrightarrow{\smile e} H^2(\mathbb{C}P^2) \longrightarrow H^2(\mathbb{S}^5) \simeq 0,$$

$$0 \simeq H^3(\mathbb{S}^5) \longrightarrow H^2(\mathbb{C}P^2) \xrightarrow{\smile e} H^4(\mathbb{C}P^2) \longrightarrow H^4(\mathbb{S}^5) \simeq 0.$$

From the first one we see that e is a generator of $H^2(\mathbb{C}P^2)$, and from the second one we see that $e \smile e$ is a generator of $H^4(\mathbb{C}P^2)$. Therefore, the Hopf invariant of the Hopf map is equal to ± 1. □

We have found a map $\mathbb{S}^3 \to \mathbb{S}^2$ of Hopf invariant ± 1, so we can conclude.

Corollary 6.3.3. *We have $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$ and more generally $\pi_{n+1}(\mathbb{S}^n) \simeq \mathbb{Z}/2\mathbb{Z}$ for every $n \geq 3$.*

Proof. We apply corollary 5.4.7 to the Hopf map, which is of Hopf invariant ± 1, and we obtain that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$. We then apply the Freudenthal suspension theorem, more precisely corollary 3.4.2, and we obtain that $\pi_{n+1}(\mathbb{S}^n) \simeq \mathbb{Z}/2\mathbb{Z}$ for every $n \geq 3$. □

Conclusion

In this thesis we saw that homotopy type theory is powerful enough to prove that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$. Even though from the point of view of classical homotopy theory this is a well-known result, it was not obvious that the univalence axiom and higher inductive types would suffice to prove it and that a constructive and purely homotopy-theoretic proof exists at all. Moreover, taking into account the fact that it has been only about five years between the definition of the circle in homotopy type theory and the computation of $\pi_4(\mathbb{S}^3)$, the progress has been rather quick, and one can hope that in a few years homotopy type theory will have reached a level comparable to that of classical homotopy theory and will help to obtain completely new results.

Comparison with classical proofs As mentioned before, the main difference between classical homotopy theory and homotopy type theory is that, in homotopy type theory, everything is homotopy-invariant. I have used the book [Hat02], presenting classical algebraic topology, quite regularly during this research, but I often had to find completely different definitions, proofs or statements that would work in homotopy type theory.

In the construction of the universal cover of the circle and of the Hopf fibration, the most obvious difference is that, instead of defining a map from the total space to the base space, we define directly the fibers and how to transport along the fibers, and determining the total space is the non-trivial part. For the universal cover of the circle it is rather transparent, but for the Hopf fibration it wasn't clear to me at first that defining it with the multiplication on $\mathbb{S}^1$ would indeed give the Hopf fibration.

Proving that the smash product is associative isn't easy, neither in homotopy type theory nor in classical homotopy theory, but for very different reasons. In homotopy type theory the problem comes from the fact that we have many different paths and higher paths to manage and that it quickly becomes complicated to take care of all coherences between them. In point-set topology, however, there is already a canonical bijection between the two spaces, but the problem is that it may not be continuous unless we assume that the spaces are nice enough. The difference is that in point-set topology we literally identify various points together, which makes the bijection easy to define but might mess up the topology, whereas in homotopy type theory we add new paths instead of identifying points.

For cohomology we already mentioned the need to use Eilenberg–MacLane spaces

instead of the classical definition via singular cochains, as the set of singular cochains is not a homotopy invariant of a space. The notion of truncations gives a very nice definition of the Eilenberg–MacLane spaces $K(\mathbb{Z}, n)$ and a very nice definition of the cup product. Note that, when working constructively, there are some unexpected phenomena related to cohomology. Indeed, without some version of the axiom of choice, it is not possible to prove the additivity axiom of cohomology. Even the cohomology group $H^1(\mathbb{N}, \mathbb{Z})$ cannot be proved to be trivial, as explained in [Shu13]. It is nice to see that no such problem arises in the computation of $\pi_4(\mathbb{S}^3)$.

Finally, for computing the cohomology of $\mathbb{C}P^2$, I couldn't manage to adapt the very geometrical proof presented for instance in [Hat02, theorem 3.19] and I had to use the Gysin sequence, which is usually considered a more advanced result in classical homotopy theory. Moreover, the construction of the Gysin sequence in [Hat02, section 4.D] is based on the Leray–Hirsch theorem whose proof uses induction on the cells of a CW complex, which is something that cannot be done in homotopy type theory. Instead, the proof presented here is new and based on proposition 6.1.1 which directly relates the cup product in dimensions n and $n + 1$.

Cubical homotopy type theory I first tried to write this thesis using cubical ideas explicitely as much as possible, as was done for instance in my previous work with Dan Licata in [LB15] and in [Cav15], but it turned out not to be such a good idea, much to my disappointment. Even though many squares and cubes appear naturally in homotopy type theory, like for instance the naturality squares of homotopies, there are also many other shapes that can appear, and treating squares differently might not be the wiser choice. For instance trying to write diagram 3.2.5 on page 73 as a Kan composition of squares and cubes is not very natural or useful. Using instead a general notion of composition of diagrams (as described at the end of page 28) seems much more natural in that case. Cubical ideas still have some benefits, in particular [Coh+15] uses cubical sets to give a computational interpretation of homotopy type theory, and in [LB15] and [Cav15] cubes are used mainly to simplify formalizations in Agda. I believe, however, that for informal synthetic homotopy theory as done in this thesis, cubical ideas aren't especially helpful in general.

Future work Maybe the most compelling next target is to formalize the present work in a proof assistant. There might be some issues in the formalization of the James construction and even more so in the monoidal structure of the smash product, which are rather heavy in terms of manipulation of paths and higher paths, but it should be possible to do.

A few results in this thesis have been stated and proved in a restricted form as my main goal was to arrive at the result $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$. For instance one ought to be able to compute the cohomology ring of all the $J_n(\mathbb{S}^k)$, including $J_\infty(\mathbb{S}^k) \simeq \Omega\mathbb{S}^{k+1}$, and of all the $\mathbb{C}P^n$, including $\mathbb{C}P^\infty \simeq K(\mathbb{Z}, 2)$. Moreover we only defined cohomology with integer coefficients but it should be possible to define cohomology with coefficients in an arbitrary group, ring or spectra and to extend most of the results presented here.

Finally the long-term goal is of course to continue the development of synthetic homotopy theory in homotopy type theory. There are many concepts that haven't been much studied yet but seem accessible like K-theory, Steenrod operations, spectral sequences, Toda brackets, and many others. I will definitely keep on pursuing this line of research, and I hope this work will inspire other people to join the study of synthetic homotopy theory as there are so many things waiting to be found.

Appendix A

A type-theoretic definition of weak ∞-groupoids

In this appendix we give a type-theoretic definition of weak ∞-groupoids, and we prove that, using the J rule of dependent type theory, we can equip every type with a structure of weak ∞-groupoid. This definition has been partly inspired by Grothendieck's definition of weak ∞-groupoids as presented by Dimitri Ara and Georges Maltsiniotis, cf for instance [Ara10]. A similar result has also been proved in [BG11] where it is shown that every type in dependent type theory can be equipped with the structure of a weak ω-groupoid in the sense of Batanin–Leinster. The definition of weak ∞-groupoids presented here is more directly related to type theory. It is not known, however, whether this definition is in some sense equivalent to the standard one (Kan complexes).

A.1 Globular sets

Just as a category is defined as a graph together with some structure on it, a weak ∞-groupoid is defined as a globular set (or ∞-graph) together with some structure on it. We recall the definition of globular set.

Definition A.1.1. A *globular set* A is a sequence of sets $(A_n)_{n\in\mathbb{N}}$ and maps $s_n, t_n : A_{n+1} \to A_n$ (source and target) such that for every $n \in \mathbb{N}$ we have:

$$s_n \circ s_{n+1} = s_n \circ t_{n+1},$$
$$t_n \circ s_{n+1} = t_n \circ t_{n+1}.$$

If A is a globular set, we write $\mathsf{Ob}(A)$ for A_0 and we call its elements the *objects* of A. If x and y are two objects of A, there is another globular set, called $\mathsf{Hom}_A(x, y)$, defined by

$$(\mathsf{Hom}_A(x,y))_n = \{\sigma \in A_{n+1} \mid s_0(s_1(\dots(s_n(\sigma))\dots)) = x \text{ and } t_0(t_1(\dots(t_n(\sigma))\dots)) = y\}.$$

The (large) set of all globular sets is denoted by Glob.

A.2 The internal language of weak ∞-groupoids

A weak ∞-groupoid is a globular set equipped with a large number of operations, with a few examples listed in section 1.4.1. In order to describe precisely what are those operations, the idea is to design a minimalist type theory $\mathcal{T}_\infty$ from which we can extract the specification of all the operations. Then we explain how to interpret this type theory into a globular set, and a weak ∞-groupoid will be defined as a globular set together with all the required operations.

There are four kind of things in $\mathcal{T}_\infty$: *contexts*, *types*, *terms*, and *context morphisms* (sequences of terms). We have one base type $\star$, which represents our ambient weak ∞-groupoid, and a type $u \simeq_T v$ for each type T and terms $u, v : T$, which represents an iterated hom-set of the ambient weak ∞-groupoid. Apart from variables, the only way to create new terms is to use one of the coherence operations $\mathsf{coh}_{\Delta.T}(\delta)$ that we describe below. Moreover, there is a special class of contexts called *contractible contexts*, which are used to define the coherence operations.

We use the letters Γ, Θ to represent contexts, Δ to represent contractible contexts, T, U to represent types, t, u, v to represent terms, and γ, θ, δ to represent context morphisms.

The syntax of the language can be summarized by the following grammar.

$$\begin{array}{rlcrl} \Gamma, \Theta := & \mid \varnothing & & \gamma, \theta := & \mid () \\ & \mid (\Gamma, x : T), & & & \mid (\gamma, u), \end{array}$$

$$\begin{array}{rlcrl} T, U := & \mid \star & & t, u, v := & \mid x \\ & \mid u \simeq_T v, & & & \mid \mathsf{coh}_{\Delta.T}(\delta). \end{array}$$

In $\mathsf{coh}_{\Delta.T}(\delta)$, the part $\mathsf{coh}_{\Delta.T}$ is the name of the coherence operation (corresponding for instance to "composition of paths" or "exchange law") while the δ is the list of points, paths, and higher paths to which we apply the coherence operation. The part $\mathsf{coh}_{\Delta.T}$ should be seen as an atomic symbol.

Given a term, type, or context morphism X, a *free variable* of X is a variable which appears at least once in X in a position which is not a subscript of the coh symbol. For instance in the term

$$\mathsf{coh}_{(x,y:\star,f:x\simeq_\star y).y\simeq_\star x}((y, z, g)),$$

the free variables are y, z, and g, but x and f are not free.

If Γ is a context, γ is a context morphism having the same length as Γ and X is a type, term or context morphism such that every free variable of X is declared in Γ, we define $X[\gamma/\Gamma]$ as the type/term/context morphism obtained by replacing every free variable of X by the corresponding term in γ. In particular, for terms of the form $\mathsf{coh}_{\Delta.T}(\delta)$, substitution must be performed only in δ and not in $\Delta.T$.

The five judgment forms of $\mathcal{T}_\infty$ are the following.

- Γ ctx means that Γ is a well-typed context.
- $\Gamma \vdash \gamma' : \Gamma'$ means that γ' is a well-typed context morphism from Γ to Γ'.
- $\Gamma \vdash T$ type means that T is a well-typed type in the context Γ.
- $\Gamma \vdash t : T$ means that t is a well-typed term of type T in the context Γ.
- Δ contr means that the context Δ is a contractible context.

The ten typing rules of $\mathcal{T}_\infty$ are the following.

Contexts

$$\frac{}{\varnothing\ \mathsf{ctx}} \qquad \frac{\Gamma \vdash T\ \mathsf{type}}{(\Gamma, x : T)\ \mathsf{ctx}}\quad (x \notin \Gamma)$$

Context morphisms

$$\frac{}{\Gamma \vdash () : \varnothing} \qquad \frac{\Gamma' \vdash T\ \mathsf{type} \qquad \Gamma \vdash \gamma : \Gamma' \qquad \Gamma \vdash u : T[\gamma/\Gamma']}{\Gamma \vdash (\gamma, u) : (\Gamma', x : T)}\quad (x \notin \Gamma')$$

Types

$$\frac{}{\Gamma \vdash \star\ \mathsf{type}} \qquad \frac{\Gamma \vdash u : T \qquad \Gamma \vdash v : T}{\Gamma \vdash u \simeq_T v\ \mathsf{type}}$$

Terms

$$\frac{}{\Gamma \vdash x : T}\quad ((x : T) \in \Gamma) \qquad \frac{\Delta\ \mathsf{contr} \qquad \Delta \vdash T\ \mathsf{type} \qquad \Gamma \vdash \delta : \Delta}{\Gamma \vdash \mathsf{coh}_{\Delta.T}(\delta) : T[\delta/\Delta]}$$

Contractible contexts

$$\frac{}{(x : \star)\ \mathsf{contr}} \qquad \frac{\Delta\ \mathsf{contr} \qquad \Delta \vdash u : T}{(\Delta, (y : T), (z : u \simeq_T y))\ \mathsf{contr}}\quad (y, z \notin \Delta)$$

All the operations in section 1.4.1 give examples of contractible contexts. The idea is that a coherence operation has a list of arguments, whose shape forms a contractible context Δ, and a return type T, which can be any type in the context Δ using only coherence operations, as they are the only terms available in $\mathcal{T}_\infty$. The definition of weak ∞-groupoid should now be intuitively clear. A weak ∞-groupoid is a globular set together with, for every context Δ and every type T such that Δ contr and $\Delta \vdash T$ type hold, an operation taking a list of arguments of types corresponding to Δ and returning a result of type corresponding to T. Making this definition precise isn't as easy as it seems, though. We first need to prove some syntactic properties of $\mathcal{T}_\infty$, and then we will explain how the syntax is translated into actual operations on a globular set.

Lemma A.2.1 (syntactic substitution)**.** *Let X be a type, a term or a context morphism, Γ and Θ two contexts, γ and θ two context morphisms, T a type and u a term. Then the following syntactic equalities hold whenever all the objects involved are well-defined (we consider that $X[\gamma/\Gamma]$ is not well-defined if there is a variable in X which is not declared in Γ)*

$$(X[\gamma/\Gamma])[\theta/\Theta] = X[\gamma[\theta/\Theta]/\Gamma)]$$
$$X[\gamma/\Gamma] = X[(\gamma, u)/(\Gamma, x : T)]$$

Proof. We proceed by induction on X. If X is a variable it holds by definition of the objects involved, otherwise we apply the induction hypothesis. □

Lemma A.2.2 (weakening)**.** *Let's assume that Γ ctx holds, for $\Gamma = (\Gamma_0, y : B)$. Then we have*

- *If $\Gamma_0 \vdash T$ type holds, then $\Gamma \vdash T$ type holds.*
- *If $\Gamma_0 \vdash u : T$ holds, then $\Gamma \vdash u : T$ holds.*
- *If $\Gamma_0 \vdash \gamma' : \Gamma'$ holds, then $\Gamma \vdash \gamma' : \Gamma'$ holds.*

Proof. We proceed by induction on the typing derivation of the judgment. For the case of a variable we use the fact that Γ ctx holds and that if $(x : T) \in \Gamma_0$, then $(x : T) \in \Gamma$. For the other cases we apply the induction hypothesis. □

Lemma A.2.3. *We have*

- *If Γ ctx holds and $(x : T) \in \Gamma$, then $\Gamma \vdash T$ type holds.*
- *If $\Gamma' \vdash \gamma : \Gamma$ holds and $(x : T) \in \Gamma$, then $\Gamma' \vdash \pi_x^\Gamma(\gamma) : T[\gamma/\Gamma]$ where π_x^Γ is the nth projection, where n is the position of x in Γ (or in other words, $\pi_x^\Gamma(\gamma) = x[\gamma/\Gamma]$).*

Proof. For the first point, we proceed by induction on the typing derivation of the judgment Γ ctx. If Γ is empty, then $(x : T) \in \Gamma$ cannot be true so there is nothing to prove. If $\Gamma = (\Gamma_0, y : T')$, then if x is equal to y, the types T and T' are also equal and we have $\Gamma_0 \vdash T$ type. If x is not equal to y, then $(x : T) \in \Gamma_0$, hence $\Gamma_0 \vdash T$ type by the induction hypothesis. In both cases we get $\Gamma_0 \vdash T$ type, hence $\Gamma \vdash T$ type holds by the weakening lemma.

For the second point, we proceed by induction on the typing derivation of the judgment $\Gamma' \vdash \gamma : \Gamma$. If Γ is empty, then $(x : T) \in \Gamma$ cannot be true so there is nothing to prove. If $\Gamma = (\Gamma_0, y : T')$ and $\gamma = (\gamma_0, u)$, then if x is equal to y, the terms $\pi_x^\Gamma(\gamma)$ and u are also equal and we have $\Gamma' \vdash \pi_x^\Gamma(\gamma) : T[\gamma_0/\Gamma_0]$. If x is not equal to y, then $(x : T) \in \Gamma_0$, hence $\Gamma' \vdash \pi_x^\Gamma(\gamma) : T[\gamma_0/\Gamma_0]$. In both cases we get $\Gamma' \vdash \pi_x^\Gamma(\gamma) : T[\gamma_0/\Gamma_0]$ hence $\Gamma' \vdash \pi_x^\Gamma(\gamma) : T[\gamma/\Gamma]$ holds. □

Lemma A.2.4 (substitution)**.** *Assume Γ' ctx and $\Gamma' \vdash \gamma : \Gamma$ hold. Then:*

- *If $\Gamma \vdash T$ type holds, then $\Gamma' \vdash T[\gamma/\Gamma]$ type holds.*

- *If* $\Gamma \vdash u : T$ *holds, then* $\Gamma' \vdash u[\gamma/\Gamma] : T[\gamma/\Gamma]$ *holds.*
- *If* $\Gamma \vdash \theta : \Theta$ *holds, then* $\Gamma' \vdash \theta[\gamma/\Gamma] : \Theta$ *holds.*

Lemma A.2.5 (compatibility)**.** *Assume* Γ ctx *holds. Then we have*

- *If* $\Gamma \vdash u : T$ *holds, then* $\Gamma \vdash T$ type *holds.*
- *If* $\Gamma \vdash \theta : \Theta$ *holds, then* Θ ctx *holds.*

Proof. The lemmas of substitution and compatibility are proved by a simultaneous induction on the main judgment. For the case of a variable, we use lemma A.2.3. For the other cases we use the induction hypothesis. □

A.3 Syntactic weak ∞-groupoids

We can now define weak ∞-groupoids. The definition is mutually recursive with the interpretation of the syntax, and the various lemmas below are needed to make sure that the interpretation of the syntax is well-defined. Because many coherence operations depend on other coherence operations which may in turn depend on other coherence operations, and so on, we first define a notion of n-partial weak ∞-groupoid and then we define the notion of weak ∞-groupoids.

Definition A.3.1. The *depth* of a type/term/context morphism is defined by

$$\begin{aligned}
\mathsf{depth}(\mathsf{coh}_{\Delta.T}(\delta)) &:= \mathsf{max}(\mathsf{depth}(\delta), \mathsf{depth}(T) + 1, \mathsf{depth}(\Delta) + 1),\\
\mathsf{depth}(u =_T v) &:= \mathsf{max}(\mathsf{depth}(u), \mathsf{depth}(v), \mathsf{depth}(T)),\\
\mathsf{depth}((\gamma, u)) &:= \mathsf{max}(\mathsf{depth}(\gamma), \mathsf{depth}(u)),
\end{aligned}$$

and

$$\mathsf{depth}(\star) = \mathsf{depth}(x) = \mathsf{depth}(()) = 0.$$

In other words, the depth is the highest level of nestedness of coherence operations, where a coherence operation is nested inside $\mathsf{coh}_{\Delta.T}(\delta)$ if it appears in Δ or T.

Definition A.3.2. A 0*-partial weak* ∞*-groupoid* is a globular set, and for $n \geq 0$, an $(n+1)$*-partial weak* ∞*-groupoid* is an n-partial weak ∞-groupoid G together with, for every contractible context Δ and every type T in Δ of depth n, a function

$$\mathbf{coh}^G_{\Delta.T} : (\eta : [\![\Delta]\!]^G) \to \mathsf{Ob}([\![T]\!]^G_\Delta(\eta)),$$

where $[\![\Delta]\!]^G$ and $[\![T]\!]^G_\Delta$ are defined below. This means that for every $\eta \in [\![\Delta]\!]^G$, we have an element $\mathbf{coh}^G_{\Delta.T}(\eta) \in \mathsf{Ob}([\![T]\!]^G_\Delta(\eta))$. Using the structure of n-partial weak ∞-groupoid on G, we can extend $\mathbf{coh}^G_{\Delta.T}$ to the case where T is of depth less than or equal to n.

Definition A.3.3. Given an n-partial weak ∞-groupoid G, we interpret well-typed contexts, context morphisms, types and terms of depth at most n as follows:

- The interpretation of a well-typed context Γ of depth at most n is a set $[\![\Gamma]\!]^G$ defined below.
- The interpretation of a well-typed context morphism θ of depth at most n from Γ to Θ is a map $[\![\theta]\!]^G_{\Gamma,\Theta} : [\![\Gamma]\!]^G \to [\![\Theta]\!]^G$ defined below.
- The interpretation of a well-typed type T of depth at most n in context Γ is a map $[\![T]\!]^G_\Gamma : [\![\Gamma]\!]^G \to \mathsf{Glob}$ defined below.
- The interpretation of a well-typed term u of depth at most n of type T in context Γ is a dependent map $[\![u]\!]^G_{\Gamma,T} : (\gamma : [\![\Gamma]\!]^G) \to \mathsf{Ob}([\![T]\!]^G_\Gamma(\gamma))$ defined below.

We now give the definitions.

- The definition of $[\![\Gamma]\!]^G$ is

$$\begin{aligned}[\![\varnothing]\!]^G &= \{()\},\\ [\![(\Gamma, x : T)]\!]^G &= \{(\gamma, a) \mid \gamma \in [\![\Gamma]\!]^G, a \in \mathsf{Ob}([\![T]\!]^G_\Gamma(\gamma))\}.\end{aligned}$$

- The definition of $[\![\theta]\!]^G_{\Gamma,\Theta} : [\![\Gamma]\!]^G \to [\![\Theta]\!]^G$ is

$$\begin{aligned}[\![()]\!]^G_{\Gamma,\varnothing}(\gamma) &= (),\\ [\![(\theta, u)]\!]^G_{\Gamma,(\Theta,x:T)}(\gamma) &= ([\![\theta]\!]^G_{\Gamma,\Theta}(\gamma), [\![u]\!]^G_{\Gamma,T[\theta/\Theta]}(\gamma)).\end{aligned}$$

- The definition of $[\![T]\!]^G_\Gamma : [\![\Gamma]\!]^G \to \mathsf{Glob}$ is

$$\begin{aligned}[\![\star]\!]^G_\Gamma(\gamma) &= G,\\ [\![u \simeq_T v]\!]^G_\Gamma(\gamma) &= \mathsf{Hom}_{[\![T]\!]^G_\Gamma(\gamma)}([\![u]\!]^G_{\Gamma,T}(\gamma), [\![v]\!]^G_{\Gamma,T}(\gamma)).\end{aligned}$$

- The definition of $[\![u]\!]^G_{\Gamma,T} : (\gamma : [\![\Gamma]\!]^G) \to \mathsf{Ob}([\![T]\!]^G_\Gamma(\gamma))$ is

$$\begin{aligned}[\![x]\!]^G_{\Gamma,T}(\gamma) &= \pi^\Gamma_x(\gamma),\\ [\![\mathsf{coh}_{\Delta.T}(\delta)]\!]^G_{\Gamma,T[\delta/\Delta]}(\gamma) &= \mathbf{coh}^G_{\Delta.T}([\![\delta]\!]^G_{\Gamma,\Theta}(\gamma)).\end{aligned}$$

Note that we use the structure of n-partial weak ∞-groupoid in the last clause.

The fact that the previous definition is well-typed follows from the following three lemmas which are straightforward to prove by induction.

Lemma A.3.4 (semantic weakening)**.** *Let $\Gamma = (\Gamma_0, y : T')$ be a nonempty context and assume that $\vdash \Gamma$ ctx holds. Let $\gamma = (\gamma_0, b)$ be an element of $[\![\Gamma]\!]^G$. Then we have*

- *If $\Gamma_0 \vdash T$ type, then*
$$[\![T]\!]^G_{\Gamma_0}(\gamma_0) = [\![T]\!]^G_\Gamma(\gamma)$$
in Glob.

- *If* $\Gamma_0 \vdash u : T$, *then*
$$[\![u]\!]^G_{\Gamma_0,T}(\gamma_0) = [\![u]\!]^G_{\Gamma,T}(\gamma)$$
in $\mathsf{Ob}([\![T]\!]^G_{\Gamma_0}(\gamma_0))$ *which is equal to* $\mathsf{Ob}([\![T]\!]^G_{\Gamma}(\gamma))$.
- *If* $\Gamma_0 \vdash \theta : \Theta$, *then*
$$[\![\theta]\!]^G_{\Gamma_0,\Theta}(\gamma_0) = [\![\theta]\!]^G_{\Gamma,\Theta}(\gamma)$$
in $[\![\Theta]\!]^G$.

Lemma A.3.5. *We have*

- *If* $\vdash \Gamma$ ctx *and* $(x : T) \in \Gamma$ *then, for every* $\gamma \in [\![\Gamma]\!]^G$,
$$\pi^\Gamma_x(\gamma) \in \mathsf{Ob}([\![T]\!]^G_\Gamma(\gamma)).$$
- *If* $\Gamma \vdash \theta : \Theta$ *and* $(y : T) \in \Theta$ *then, for every* $\gamma \in [\![\Gamma]\!]^G$,
$$[\![\pi^\Theta_y(\theta)]\!]^G_{\Gamma,T[\theta/\Theta]}(\gamma) = \pi^\Theta_y([\![\theta]\!]^G_{\Gamma,\Theta}(\gamma)).$$

Lemma A.3.6 (semantic substitution)**.** *Assume we have* $\Gamma \vdash \theta : \Theta$ *and* $\gamma \in [\![\Gamma]\!]^G$. *Then*

- *If* $\Theta \vdash T$ type, *then*
$$[\![T[\theta/\Theta]]\!]^G_\Gamma(\gamma) = [\![T]\!]^G_\Theta([\![\theta]\!]^G_{\Gamma,\Theta}(\gamma)).$$
- *If* $\Theta \vdash u : T$, *then*
$$[\![u[\theta/\Theta]]\!]^G_{\Gamma,T[\theta/\Theta]}(\gamma) = [\![u]\!]^G_{\Theta,T}([\![\theta]\!]^G_{\Gamma,\Theta}(\gamma)).$$
- *If* $\Theta \vdash \theta' : \Theta'$, *then*
$$[\![\theta'[\theta/\Theta]]\!]^G_{\Gamma,\Theta'}(\gamma) = [\![\theta']\!]^G_{\Theta,\Theta'}([\![\theta]\!]^G_{\Gamma,\Theta}(\gamma)).$$

Finally, we can define weak ∞-groupoids.

Definition A.3.7. A *syntactic weak* ∞*-groupoid* is a globular set G together with for every $n \in \mathbb{N}$, a structure of $(n+1)$-partial weak ∞-groupoid over the previous structure of n-partial weak ∞-groupoid. In particular, that means that for every contractible context Δ and every type T in Δ (without restriction on depth) we have a function
$$\mathbf{coh}^G_{\Delta.T} : ([\![\Delta]\!]^G) \to \mathsf{Ob}([\![T]\!]^G_\Delta(\eta)),$$
where the contexts, context morphisms, types and terms are interpreted as above.

Let's give some examples of coherence operations.

Example A.3.8. The *constant path* coherence operation is
$$\mathbf{coh}_{(x:\star).x \simeq_\star x}.$$

Proof. Given a globular set G we have

$$[\![(x : \star)]\!]^G = \mathsf{Ob}(G),$$

and given $a \in \mathsf{Ob}(G)$ we have

$$[\![x \simeq_\star x]\!]^G_{(x:\star)}(a) = \mathsf{Hom}_G(a, a).$$

Therefore, the operation $\mathbf{coh}^G_{(x:\star).x\simeq_\star x}$ takes an object a of G and returns a morphism from a to a. In particular, such an operation exists only when the graph is a reflexive graph. □

Example A.3.9. The *inverse* coherence operation is

$$\mathbf{coh}_{(x,y:\star)(t:x\simeq_\star y).y\simeq_\star x}.$$

Proof. Given a globular set G we have

$$\begin{aligned}&[\![(x, y : \star)(t : x \simeq_\star y)]\!]^G\\ &\quad = \{(a, b, p) \mid a, b \in \mathsf{Ob}(G), p \in \mathsf{Ob}(\mathsf{Hom}_G(a, b))\},\end{aligned}$$

and given a triple (a, b, p) in this set we have

$$[\![y \simeq_\star x]\!]^G_{(x,y:\star)(t:x\simeq_\star y)}(a, b, p) = \mathsf{Hom}_G(b, a).$$

Therefore, the operation $\mathbf{coh}^G_{(x,y:\star)(t:x\simeq_\star y).y\simeq_\star x}$ takes a morphism p in G between two objects a and b and returns a morphism p^{-1} going from b to a. □

Example A.3.10. The *involutivity of inverse* coherence operation is

$$\mathbf{coh}_{(x,y:\star)(t:x\simeq_\star y).t\simeq_{x\simeq_\star y}(\mathsf{coh}_{(x,y:\star)(t:x\simeq_\star y).y\simeq_\star x}(y,x,\mathsf{coh}_{(x,y:\star)(t:x\simeq_\star y).y\simeq_\star x}(x,y,t)))}.$$

Proof. Here the coherence operation is of depth 1, therefore we need to start with a 1-partial weak ∞-groupoid G. In a 1-partial weak ∞-groupoid we have in particular the

$$\mathbf{coh}_{(x,y:\star)(t:x\simeq_\star y).y\simeq_\star x}(a, b, p)$$

coherence operation described in the previous example. We write it $p \mapsto p^{-1}$ for short. We have

$$\begin{aligned}&[\![(x, y : \star)(t : x \simeq_\star y)]\!]^G\\ &\quad = \{(a, b, p) \mid a, b \in \mathsf{Ob}(G), p \in \mathsf{Ob}(\mathsf{Hom}_G(a, b))\},\end{aligned}$$

and given a triple (a, b, p) in this set we have

$$\begin{aligned}&[\![t \simeq_{x\simeq_\star y} (\mathsf{coh}_{(x,y:\star)(t:x\simeq_\star y).y\simeq_\star x}(y, x, \mathsf{coh}_{(x,y:\star)(t:x\simeq_\star y).y\simeq_\star x}(x, y, t)))]\!]^G_{(x,y:\star)(t:x\simeq_\star y)}(a, b, p)\\ &\quad = \mathsf{Hom}_{\mathsf{Hom}_G(a,b)}(p, (p^{-1})^{-1}).\end{aligned}$$

Therefore, the operation we're looking at takes a morphism p in G between two objects a and b and returns a 2-morphism from p to $(p^{-1})^{-1}$. □

There are many coherence operations which seem completely superfluous like for instance $\mathbf{coh}_{(x,y:\star)(t:x\simeq_\star y).x\simeq_\star y}(a,b,p)$ which gives a morphism from a to b given a morphism p from a to b. There are also various ways to order the arguments, for instance in the diagram for horizontal composition of 2-morphisms, which give different contexts hence different coherence operations. But this is not a problem because it is always possible to prove that they are all equal to one another via another coherence operation.

A.4 The underlying weak ∞-groupoid of a type

We now prove that in a dependent type theory $\mathcal{T}_{\mathsf{ML}}$ with identity types, the J rule, and the definitional computation rule, we can equip every type with a structure of syntactic weak ∞-groupoid. In this section we write $\vdash_\infty$ for the judgments in the type theory $\mathcal{T}_\infty$ described above and $\vdash_{\mathsf{ML}}$ for the judgments in $\mathcal{T}_{\mathsf{ML}}$. We write $\mathsf{Id}_A(u,v)$ for the identity type if $\mathcal{T}_{\mathsf{ML}}$ and $\equiv$ for definitional equality in $\mathcal{T}_{\mathsf{ML}}$.

We first define the iterated constant path and its type.

Definition A.4.1. Given a type $\vdash_{\mathsf{ML}} A : \mathsf{Type}$, a term $\vdash_{\mathsf{ML}} a : A$ and a natural number $n \in \mathbb{N}$, we define a type $\vdash_{\mathsf{ML}} \mathsf{I}_n^{(A,a)} : \mathsf{Type}$ and a term $\vdash_{\mathsf{ML}} \mathsf{i}_n^{(A,a)} : \mathsf{I}_n^{(A,a)}$ by

$$\mathsf{I}_0^{(A,a)} := A, \qquad \mathsf{i}_0^{(A,a)} := a,$$
$$\mathsf{I}_{n+1}^{(A,a)} := \left(\mathsf{i}_n^{(A,a)} =_{\mathsf{I}_n^{(A,a)}} \mathsf{i}_n^{(A,a)}\right), \qquad \mathsf{i}_{n+1}^{(A,a)} := \mathsf{idp}_{\mathsf{i}_n^{(A,a)}}.$$

We now consider a type A of $\mathcal{T}_{\mathsf{ML}}$ and we want to equip it with a structure of weak ∞-groupoid. The idea is that a coherence operation in a contractible context Δ is defined by successive applications of the J rule until we reach the context $(a : A)$. Then we have to prove an equality between two terms in this context and it turns out that all terms defined only using coherence operations in the context $(a : A)$ are definitionally equal to the iterated constant path. Therefore we can define the coherence operation in the case $(a : A)$ by an iterated constant path, and it preserves the invariant that a coherence operation applied to constant paths is definitionally equal to a constant path.

We define an interpretation $(\!|-|\!)$ of $\mathcal{T}_\infty$ into $\mathcal{T}_{\mathsf{ML}}$, define two terms id^Δ and J_Δ for every contractible context Δ, and prove three lemmas stating that, in the context $(a : A)$, all types are definitionally equal to $\mathsf{I}_n^{(A,a)}$ and all terms are definitionally equal to $\mathsf{i}_n^{(A,a)}$. All of that has to be done simultaneously by induction. We also need a substitution and a weakening lemma, stating that the interpretation commutes with the operations of substitution and weakening, but we do not state them here for simplicity. The *dimension* of a type is defined by $\mathsf{dim}(\star) = 0$ and $\mathsf{dim}(u \simeq_T v) = \mathsf{dim}(T) + 1$.

Definition A.4.2. The interpretation of $\mathcal{T}_\infty$ into $\mathcal{T}_{\mathsf{ML}}$ satisfies the following typing rules and is defined below.

- Given $\vdash_\infty \Gamma$, the interpretation $(\!|\Gamma|\!)^A$ is a context in $\mathcal{T}_{\mathsf{ML}}$.
- Given $\Gamma \vdash_\infty \gamma : \Gamma'$, the interpretation $(\!|\gamma|\!)^A_{\Gamma,\Gamma'}$ is a context morphism in $\mathcal{T}_{\mathsf{ML}}$ from $(\!|\Gamma|\!)^A$ to $(\!|\Gamma'|\!)^A$.

- Given $\Gamma \vdash_\infty T$, the interpretation $(\!|T|\!)^A_\Gamma$ is a dependent type in $\mathcal{T}_{\mathsf{ML}}$ over $(\!|\Gamma|\!)^A$.
- Given $\Gamma \vdash_\infty t : T$, the interpretation $(\!|t|\!)^A_{\Gamma,T}$ is a term in $\mathcal{T}_{\mathsf{ML}}$ of type $(x : (\!|\Gamma|\!)^A) \to (\!|T|\!)^A_\Gamma$.

Definition A.4.3. For every contractible context Δ, we define a context morphism

$$(a : A) \vdash_{\mathsf{ML}} \mathsf{id}^\Delta_a : (\!|\Delta|\!)^A$$

and a term

$$(a : A)(P : (\delta : (\!|\Delta|\!)^A) \to \mathsf{Type})(d : (a : A) \to P(\mathsf{id}^\Delta_a)) \vdash_{\mathsf{ML}} \mathsf{J}_\Delta(P, d) : (\delta : (\!|\Delta|\!)^A) \to P(\delta)$$

satisfying the definitional equality

$$\mathsf{J}_\Delta(P, d)(\mathsf{id}^\Delta_a) \equiv d(a).$$

Lemma A.4.4. *For every contractible context Δ and every type $\Delta \vdash_\infty T$, we have the definitional equality*

$$(a : A) \vdash_{\mathsf{ML}} (\!|T|\!)^A_\Delta(\mathsf{id}^\Delta_a) \equiv \mathsf{I}^{(A,a)}_{\mathsf{dim}(T)}.$$

Lemma A.4.5. *For every contractible contexts Δ, Δ' and every context morphism $\Delta' \vdash_\infty \delta : \Delta$, we have the definitional equality*

$$(a : A) \vdash_{\mathsf{ML}} (\!|\delta|\!)^A_{\Delta',\Delta}(\mathsf{id}^{\Delta'}_a) \equiv \mathsf{id}^\Delta_a.$$

Lemma A.4.6. *For every contractible context Δ and every term $\Delta \vdash_\infty t : T$, we have the definitional equality*

$$(a : A) \vdash_{\mathsf{ML}} (\!|t|\!)^A_{\Delta,T}(\mathsf{id}^\Delta_a) \equiv \mathsf{i}^{(A,a)}_{\mathsf{dim}(T)}.$$

Definition of the interpretation. The interpretation is defined as follows:

- The definition of $(\!|\Gamma|\!)^A$ is

$$(\!|\varnothing|\!)^A = \varnothing,$$
$$(\!|(\Gamma, x : T)|\!)^A = (y : (\!|\Gamma|\!)^A, x : (\!|T|\!)^A_\Gamma(y)).$$

- The definition of $(\!|\delta|\!)^A_{\Gamma,\Theta} : (\!|\Gamma|\!)^A \to (\!|\Theta|\!)^A$ is

$$(\!|()|\!)^A_{\Gamma,\varnothing}(\gamma) = (),$$
$$(\!|(\theta, u)|\!)^A_{\Gamma,(\Theta,x:T)}(\gamma) = ((\!|\theta|\!)^A_{\Gamma,\Theta}(\gamma), (\!|u|\!)^A_{\Gamma,T[\theta/\Theta]}(\gamma)).$$

- The definition of $(\gamma : (\!|\Gamma|\!)^A) \vdash_{\mathsf{ML}} (\!|T|\!)^A_\Gamma(\gamma) : \mathsf{Type}$ is

$$(\!|\star|\!)^A_\Gamma(\gamma) = A,$$
$$(\!|u \simeq_T v|\!)^A_\Gamma(\gamma) = \mathsf{Id}_{(\!|T|\!)^A_\Gamma(\gamma)}((\!|u|\!)^A_{\Gamma,T}(\gamma), (\!|v|\!)^A_{\Gamma,T}(\gamma)).$$

- The definition of $(\gamma : (\!|\Gamma|\!)^A) \vdash_{\mathsf{ML}} (\!|u|\!)^A_{\Gamma,T} : (\!|T|\!)^A_\Gamma(\gamma)$ is

$$(\!|x|\!)^A_{\Gamma,x:T}(\gamma) = \pi^\Gamma_x(\gamma),$$
$$(\!|\mathsf{coh}_{\Delta.T}(\delta)|\!)^A_{\Gamma,T[\delta/\Delta]}(\gamma) = \mathsf{J}_\Delta((\!|T|\!)^A_\Delta, (\lambda a.\mathsf{i}^{(A,a)}_{\mathsf{dim}(T)}))((\!|\delta|\!)^A_{\Gamma,\Delta}(\gamma)).$$

The things to note are that the interpretation of $\star$ is the type A we are interested in, that the interpretation of $u \simeq_T v$ uses the identity type in $\mathcal{T}_{\mathsf{ML}}$, and that coherence operations are implemented using J_Δ and $(\lambda a.\mathsf{i}^{(A,a)}_{\mathsf{dim}(T)})$ in the base case, which is well-typed by lemma A.4.4. □

Definition of id^Δ_a *and* J_Δ. For $a : A$, the sequence of terms $\mathsf{id}^\Delta_a : (\!|\Delta|\!)^A$ is defined by induction on Δ by

$$\mathsf{id}^{(x:\star)}_a := (a),$$
$$\mathsf{id}^{(\Delta',y:T,z:u=_T y)}_a := (\mathsf{id}^{\Delta'}_a, (\!|u|\!)^A_{\Delta',T}(\mathsf{id}^{\Delta'}_a), \mathsf{idp}_{(\!|u|\!)^A_{\Delta',T}(\mathsf{id}^{\Delta'}_a)}).$$

We remind that J has type

$$(u : A)(P : (y : A)(z : \mathsf{Id}_A(u, y)) \to \mathsf{Type})(d : P(u, \mathsf{idp}_u))$$
$$\vdash_{\mathsf{ML}} \mathsf{J}(u, P, d) : (v : A)(p : \mathsf{Id}_A(u, v)) \to P(v, p).$$

For $a : A$, $P : (\delta : (\!|\Delta|\!)^A) \to \mathsf{Type}$, and $d : (a : A) \to P(\mathsf{id}^\Delta_a)$, the term

$$\mathsf{J}_\Delta(P, d) : (\delta : (\!|\Delta|\!)^A) \to P(\delta)$$

is defined by induction on Δ by

$$\mathsf{J}_{(x:\star)}(P, d, a) := d(a),$$
$$\mathsf{J}_{(\Delta',y:T,z:u=_T y)}(P, d, \delta', y, z) := \mathsf{J}((\!|u|\!)^A_{\Delta',T}(\delta'), P(\delta'),$$
$$\mathsf{J}_{\Delta'}((\lambda\delta'.P(\delta', (\!|u|\!)^A_{\Delta',T}(\delta'), \mathsf{idp}_{(\!|u|\!)^A_{\Delta',T}(\delta')})), d, \delta'))$$
$$(y, z).$$

In other words, in order to apply J_Δ to (δ', y, z), we first apply the regular J rule to z and then we apply $\mathsf{J}_{\Delta'}$ to δ'. The reduction rule holds because in the case id^Δ_a we have idp for z, hence the reduction rule of the regular J applies, and then $\mathsf{id}^{\Delta'}_a$ for δ' hence the reduction rule for $\mathsf{J}_{\Delta'}$ applies. □

Proof of lemma A.4.4. We want to prove that $(\!|T|\!)^A_\Delta(\mathsf{id}^\Delta_a)$ is definitionally equal to $\mathsf{I}^{(A,a)}_{\mathsf{dim}(T)}$. We proceed by induction on T. If T is $\star$, then it's true as they are both equal to A. If T is of the form $u \simeq_{T'} v$, then it's true by lemma A.4.6. □

Proof of lemma A.4.5. We want to prove that $(\!|\delta|\!)^A_{\Delta',\Delta}(\mathsf{id}^{\Delta'}_a)$ is definitionally equal to id^Δ_a. We proceed by induction on Δ and it follows from lemma A.4.6. □

Proof of lemma A.4.6. We want to prove that $(\!|t|\!)^A_{\Delta,T}(\mathsf{id}^\Delta_a)$ is definitionally equal to $\mathsf{i}^{(A,a)}_{\mathsf{dim}(T)}$. We proceed by induction on Δ and t. If t is a variable, then given the definition of id^Δ_a, the interpretation of t is either a or of the form $(\!|u|\!)^A_{\Delta',T}(\mathsf{id}^{\Delta'}_a)$ or $\mathsf{idp}_{(\!|u|\!)^A_{\Delta',T}(\mathsf{id}^{\Delta'}_a)})$, which are definitionally equal to $\mathsf{i}^{(A,a)}_{\mathsf{dim}(T)}$ by induction hypothesis, as Δ' is strictly smaller that Δ.

If t is of the form $\mathsf{coh}_{\Delta.T}(\delta)$, then we have

$$\begin{aligned}(\!|\mathsf{coh}_{\Delta.T}(\delta)|\!)^A_{\Gamma,T[\delta/\Delta]}(\mathsf{id}^\Gamma_a) &\equiv \mathsf{J}_\Delta((\!|T|\!)^A_\Delta, (\lambda a.\mathsf{i}^{(A,a)}_{\mathsf{dim}(T)}), (\!|\delta|\!)^A_{\Gamma,\Delta}(\mathsf{id}^\Gamma_a))\\ &\equiv \mathsf{J}_\Delta((\!|T|\!)^A_\Delta, (\lambda a.\mathsf{i}^{(A,a)}_{\mathsf{dim}(T)}), \mathsf{id}^\Delta_a)\\ &\equiv \mathsf{i}^{(A,a)}_{\mathsf{dim}(T)},\end{aligned}$$

using lemma A.4.5 and the reduction rule for J_Δ. □

This concludes the proof that we can interpret all the coherence operations as terms of $\mathcal{T}_{\mathsf{ML}}$. This is all we need when using type theory as a foundational system as is done in this thesis, but we can also get an actual set-based weak ∞-groupoid if desired.

Definition A.4.7. Given a type A in $\mathcal{T}_{\mathsf{ML}}$, we define a globular set A^{G} by

$$\begin{aligned}\mathsf{Ob}(A^{\mathsf{G}}) &:= \{t \mid \vdash_{\mathsf{ML}} t : A\},\\ \mathsf{Hom}_{A^{\mathsf{G}}}(t,u) &:= (\mathsf{Id}_A(t,u))^{\mathsf{G}}.\end{aligned}$$

Proposition A.4.8. *Given a type A in $\mathcal{T}_{\mathsf{ML}}$, the globular set A^{G} has a structure of weak ∞-groupoid.*

Proof. Given $(\!|-|\!)$ the interpretation of $\mathcal{T}_\infty$ in $\mathcal{T}_{\mathsf{ML}}$ given in definition A.4.2, we define an interpretation of $\mathcal{T}_\infty$ in the globular set A^{G} in the sense of definition A.3.3 by

$$\begin{aligned}[\![\Gamma]\!]^{A^{\mathsf{G}}} &:= \{\bar{t} \mid \vdash_{\mathsf{ML}} \bar{t} : (\!|\Gamma|\!)^A\},\\ [\![\theta]\!]^{A^{\mathsf{G}}}_{\Gamma,\Theta}(\gamma) &:= (\!|\theta|\!)^A_{\Gamma,\Theta}[\gamma/\Gamma],\\ [\![T]\!]^{A^{\mathsf{G}}}_{\Gamma}(\gamma) &:= \left((\!|T|\!)^A_\Gamma[\gamma/\Gamma]\right)^{\mathsf{G}},\\ [\![u]\!]^{A^{\mathsf{G}}}_{\Gamma,T}(\gamma) &:= (\!|u|\!)^A_{\Gamma,T}[\gamma/\Gamma].\end{aligned}$$

It is easy to check that it satisfies the required properties and it gives an interpretation of all the coherence operations in A^{G}. □

Appendix B

The cardinal of $\pi_4(\mathbb{S}^3)$

The purpose of this appendix is to give a self-contained definition of the natural number $n : \mathbb{N}$ satisfying $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$, in order to serve as a sophisticated test case for candidates for a computational interpretation of univalence and higher inductive types. It doesn't contain the proof that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$ nor the proof that $n = 2$, only the definition of n based on theorem 3.4.5.

If you wrote a proof assistant for homotopy type theory giving a computational interpretation of univalence and higher inductive types, please try to implement the following computation and check that you do get 2 as the result.

B.1 Preliminaries

Here are some basic definitions of homotopy type theory that we use, to fix the notations.

- The universe is Type.
- Given a type A and two elements $a, b : A$, the type of paths from a to b is $a =_A b$ (identity type).
- Given an element $a : A$, the constant path at a is idp_a.
- Given a path $p : a =_A b$, the reverse path of p is $p^{-1} : b =_A a$.
- Given two composable paths $p : a =_A b$ and $q : b =_A c$, their composition is $p \cdot q : a =_A c$ (diagrammatic order).
- Given a function $f : A \to B$ and a path $p : a =_A b$, the application of f to p is $\mathsf{ap}_f(p) : f(a) =_B f(b)$.
- Given a dependent type $P : A \to \mathsf{Type}$, a path $p : a =_A b$ and an element $u : P(a)$, the transport of u along p in P is $\mathsf{transport}^P(p, u) : P(b)$.

B.1.1 Equivalences

Given a function $f : A \to B$, there is a type $\mathsf{isequiv}(f)$ of proofs that f is an equivalence. We do not specify this type here but it should have the following properties:

- A function f is an equivalence if and only if there exists $g : B \to A$ such that $g(f(x)) =_A x$ and $f(g(y)) =_B y$ for every $x : A$ and $y : B$.
- Any two elements of $\mathsf{isequiv}(f)$ are equal.

We write $A \simeq B$ for the type $\sum_{f:A\to B} \mathsf{isequiv}(f)$.

B.1.2 Dependent paths

Given a dependent type $P : A \to \mathsf{Type}$, a path $p : a =_A b$ and two elements $u : P(a)$ and $v : P(b)$, we write $u =^P_p v$ for the type of dependent paths from u to v over p. We assume that $u =^P_{\mathsf{idp}_a} v$ is definitionally equal to $u =_{P(a)} v$. The type of dependent paths $u =^P_p v$ is equivalent to the type $\mathsf{transport}^P(p, u) =_{P(b)} v$. We have the two maps

$$\mathsf{in}^{\mathsf{transport}} : \mathsf{transport}^P(p, u) =_{P(b)} v \to u =^P_p v,$$
$$\mathsf{out}^{\mathsf{transport}} : u =^P_p v \to \mathsf{transport}^P(p, u) =_{P(b)} v,$$

which are inverse to each other. If P is of the form $P(x) := (f(x) =_B g(x))$, then we have two maps

$$\mathsf{in}^{=} : (u \cdot \mathsf{ap}_g(p) =_{f(a)=_B g(b)} \mathsf{ap}_f(p) \cdot v) \to u =^{\lambda x.(f(x)=_B g(x))}_p v,$$
$$\mathsf{out}^{=} : u =^{\lambda x.(f(x)=_B g(x))}_p v \to (u \cdot \mathsf{ap}_g(p) =_{f(a)=_B g(b)} \mathsf{ap}_f(p) \cdot v).$$

If $f : (x : A) \to B(x)$ and $p : a =_A b$, we write $\mathsf{apd}_f(p) : f(a) =^B_p f(b)$ for the image of p by f.

B.1.3 Function extensionality and univalence

We assume unary function extensionality, which is a term of type

$$\mathsf{funext} : \{A : \mathsf{Type}\}\{B : A \to \mathsf{Type}\}\{f, g : (x : A) \to B(x)\}$$
$$(h : (x : A) \to f(x) =_{B(x)} g(x)) \to f =_{(x:A)\to B(x)} g.$$

Moreover, $\mathsf{funext}\{f\}\{g\}$ is a equivalence. We also assume ternary function extensionality

$$\begin{aligned}\mathsf{funext}^3 : \{A, B : \mathsf{Type}\}\{C : A \to \mathsf{Type}\}\{p : a =_A a'\}\\ \{f : (x : C(a)) \to B\}\{g : (y : C(a')) \to B\}\\ \to (\{x : C(a)\}\{y : C(a')\}(z : x =^C_p y) \to f(x) =_B f(y))\\ \to f =^{\lambda z.((x:C(z))\to B)}_p g,\end{aligned}$$

and unary dependent function extensionality

$$\begin{aligned}\mathsf{funext}^{\mathsf{dep}} : \{A, B : \mathsf{Type}\}\{C : A \to B \to \mathsf{Type}\}(p : b =_B b')\{f : (x : A) \to C(x)(b)\}\\ \{g : (x : A) \to C(x)(b')\}\\ \to ((x : A) \to f(x) =_p^{C(x)} g(x)) \to f =_p^{\lambda t.(x:A)\to C(x)(t)} g.\end{aligned}$$

Finally we assume the univalence axiom

$$\mathsf{ua} : (A \simeq B) \to (A =_{\mathsf{Type}} B).$$

The map ua is itself an equivalence, but we won't need to give that a name here.

B.1.4 Truncation levels

A type A is (-2)-*truncated* (or *contractible*) if the type

$$\sum_{a:A}((x : A) \to a =_A x)$$

has an element. For $n \geq -1$, a type A is n-*truncated* if for every $x, y : A$, the type $x =_A y$ is $(n-1)$-truncated. We have the following properties:

- The type of proofs that A is n-truncated is (-1)-truncated.
- To prove that a type A is (-1)-truncated it is enough to construct a term of type $(x, y : A) \to x =_A y$.
- n-truncated types are stable under products.

B.2 The definition

The number n is defined as the absolute value of the image of 1 by the following composition of maps. The types and maps involved are defined below.

$$\begin{array}{c}\mathbb{Z} \xrightarrow{n\mapsto \mathsf{loop}^n} \Omega\mathbb{S}^1 \xrightarrow{\Omega\varphi_{\mathbb{S}^1}} \Omega^2\mathbb{S}^2 \xrightarrow{\Omega^2\varphi_{\mathbb{S}^2}} \Omega^3\mathbb{S}^3 \xrightarrow{\Omega^3 e} \Omega^3(\mathbb{S}^1 * \mathbb{S}^1) \xrightarrow{\Omega^3\alpha} \Omega^3\mathbb{S}^2 \\ \overset{h}{\curvearrowleft} \\ \Omega^3(\mathbb{S}^1 * \mathbb{S}^1) \xrightarrow[\Omega^3(e^{-1})]{} \Omega^3\mathbb{S}^3 \xrightarrow{e_3} \Omega^2\|\mathbb{S}^2\|_2 \xrightarrow[\Omega\kappa_{2,\mathbb{S}^2}]{} \Omega\|\Omega\mathbb{S}^2\|_1 \xrightarrow[\kappa_{1,\Omega\mathbb{S}^2}]{} \|\Omega^2\mathbb{S}^2\|_0 \xrightarrow{e_2} \Omega\mathbb{S}^1 \xrightarrow{e_1} \mathbb{Z}\end{array}$$

We do not repeat the definition of $\mathbb{Z}$ and of the two maps between $\mathbb{Z}$ and $\Omega\mathbb{S}^1$ here.

B.3 Suspensions and spheres

The circle is defined as the higher inductive type $\mathbb{S}^1$ with constructors

$$\begin{aligned}&\mathsf{base} : \mathbb{S}^1,\\ &\mathsf{loop} : \mathsf{base} =_{\mathbb{S}^1} \mathsf{base}.\end{aligned}$$

Given a type A, we define the suspension of A as the higher inductive type ΣA with constructors

$$\begin{aligned}&\mathsf{north} : \Sigma A,\\ &\mathsf{south} : \Sigma A,\\ &\mathsf{merid} : A \to \mathsf{north} =_{\Sigma A} \mathsf{south}.\end{aligned}$$

For $n \geq 1$, we define the $(n+1)$-sphere as the suspension of the n-sphere:

$$\mathbb{S}^{n+1} := \Sigma \mathbb{S}^n.$$

B.4 Pointed types, pointed maps and loop spaces

A pointed type is a type A together with a point $\star_A : A$ (often omitted when clear from the context). The circle is pointed by base and the suspension of any type by north. A pointed map between two pointed types A and B is a map $f : A \to B$ together with an equality $\star_f : f(\star_A) = \star_B$ (also often omitted). The loop space of a pointed type A is the type $\Omega A := (\star_A = \star_A)$ pointed by $\mathsf{idp}_{\star_A}$. We can iterate this operation: $\Omega^1 A := \Omega A$ and $\Omega^{n+1} A := \Omega(\Omega^n A)$. If f is a pointed map between the pointed types A and B, then Ωf is the map between ΩA and ΩB defined by

$$(\Omega f)(p) := \star_f^{-1} \cdot \mathsf{ap}_f(p) \cdot \star_f$$

and pointed by the proof that $\star_f^{-1} \cdot \mathsf{ap}_f(\mathsf{idp}_{\star_A}) \cdot \star_f = \mathsf{idp}_{\star_B}$. We can again iterate this operation as above.

B.5 Loop space of a suspension

Given a pointed type A, we define the map

$$\begin{aligned}\varphi_A &: A \to \Omega(\Sigma A),\\ \varphi_A(a) &:= \mathsf{merid}(a) \cdot \mathsf{merid}(\star_A)^{-1},\end{aligned}$$

pointed by the obvious proof that $\mathsf{merid}(\star_A) \cdot \mathsf{merid}(\star_A)^{-1} = \mathsf{idp}_{\mathsf{north}}$.

B.6 The 3-sphere and the join of two circles

B.6.1 Join and associativity

Given two types A and B, the join of A and B is the higher inductive type $A * B$ with constructors

$$\begin{aligned} \mathsf{inl} &: A \to A * B, \\ \mathsf{inr} &: B \to A * B, \\ \mathsf{push} &: (a : A)(b : B) \to \mathsf{inl}(a) =_{A*B} \mathsf{inr}(b). \end{aligned}$$

If A is pointed then $A * B$ is pointed by $\mathsf{inl}(\star_A)$.

This operation is associative, we have the pointed map

$$\begin{aligned} \alpha_{A,B,C} &: A * (B * C) \to (A * B) * C, \\ \alpha_{A,B,C}(\mathsf{inl}(a)) &:= \mathsf{inl}(\mathsf{inl}(a)), \\ \alpha_{A,B,C}(\mathsf{inr}(\mathsf{inl}(b))) &:= \mathsf{inl}(\mathsf{inr}(b)), \\ \alpha_{A,B,C}(\mathsf{inr}(\mathsf{inr}(c))) &:= \mathsf{inr}(c), \\ \mathsf{ap}_{\alpha_{A,B,C} \circ \mathsf{inr}}(\mathsf{push}(b,c)) &:= \mathsf{push}(\mathsf{inr}(b), c), \\ \mathsf{ap}_{\alpha_{A,B,C}}(\mathsf{push}(a, \mathsf{inl}(b))) &:= \mathsf{ap}_{\mathsf{inl}}(\mathsf{push}(a,b)), \\ \mathsf{ap}_{\alpha_{A,B,C}}(\mathsf{push}(a, \mathsf{inr}(c))) &:= \mathsf{push}(\mathsf{inl}(a), c), \\ \mathsf{apd}_{\mathsf{ap}_{\alpha_{A,B,C}} \circ \mathsf{push}(a,-)}(\mathsf{push}(b,c)) &:= \mathsf{in}^{=}(op(\mathsf{out}^{=}(\mathsf{apd}_{\mathsf{push}(-,c)}(\mathsf{push}(a,b))))). \end{aligned}$$

For the last equation, inside the $\mathsf{in}^{=}$ we need a term of type

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

and the $\mathsf{out}^{=}$ term has type

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

Hence the term op is the path algebra mapping from one type to the other, which isn't complicated to define. The map $\alpha^{-1}_{A,B,C} : (A * B) * C \to A * (B * C)$ is defined in a similar way.

Given two maps $f : A \to A'$ and $g : B \to B'$, we define the map

$$\begin{aligned} f * g &: (A * B) \to (A' * B'), \\ (f * g)(\mathsf{inl}\, a) &:= \mathsf{inl}(f(a)), \\ (f * g)(\mathsf{inr}\, b) &:= \mathsf{inr}(g(b)), \\ \mathsf{ap}_{f*g}(\mathsf{push}(a,b)) &:= \mathsf{push}(f(a), g(b)). \end{aligned}$$

The map $f * g$ is pointed as soon as f is pointed.

B.6.2 Suspension and join with the booleans

The type $\mathbf{2}$ of booleans, with constructors $\mathsf{true}, \mathsf{false} : \mathbf{2}$ is pointed by true. For any type A we define the following two pointed maps, inverse to each other:

$$\begin{aligned}
\psi_A : \Sigma A &\to \mathbf{2} * A,\\
\psi_A(\mathsf{north}) &:= \mathsf{inl}(\mathsf{true}),\\
\psi_A(\mathsf{south}) &:= \mathsf{inl}(\mathsf{false}),\\
\mathsf{ap}_{\psi_A}(\mathsf{merid}(a)) &:= \mathsf{push}(\mathsf{true}, a) \cdot \mathsf{push}(\mathsf{false}, a)^{-1},
\end{aligned}$$

$$\begin{aligned}
\psi_A^{-1} : \mathbf{2} * A &\to \Sigma A,\\
\psi_A^{-1}(\mathsf{inl}(\mathsf{true})) &:= \mathsf{north},\\
\psi_A^{-1}(\mathsf{inl}(\mathsf{false})) &:= \mathsf{south},\\
\psi_A^{-1}(\mathsf{inr}(a)) &:= \mathsf{south},\\
\psi_A^{-1}(\mathsf{push}(\mathsf{true}, a)) &:= \mathsf{merid}(a),\\
\psi_A^{-1}(\mathsf{push}(\mathsf{false}, a)) &:= \mathsf{idp}_{\mathsf{south}}.
\end{aligned}$$

We also define the following two pointed maps, again inverse to each other:

$$\begin{aligned}
c : (\mathbf{2} * \mathbf{2}) &\to \mathbb{S}^1,\\
c(\mathsf{inl}(b)) &:= \mathsf{base},\\
c(\mathsf{inr}(b')) &:= \mathsf{base},\\
\mathsf{ap}_c(\mathsf{push}(\mathsf{true}, \mathsf{true})) &:= \mathsf{loop},\\
\mathsf{ap}_c(\mathsf{push}(b, b')) &:= \mathsf{idp}_{\mathsf{base}},
\end{aligned}$$

$$\begin{aligned}
c^{-1} : \mathbb{S}^1 &\to (\mathbf{2} * \mathbf{2}),\\
c^{-1}(\mathsf{base}) &:= \mathsf{inl}(\mathsf{true}),\\
\mathsf{ap}_{c^{-1}}(\mathsf{loop}) &:= \mathsf{push}(\mathsf{true}, \mathsf{true})\\
&\quad \cdot \mathsf{push}(\mathsf{false}, \mathsf{true})^{-1}\\
&\quad \cdot \mathsf{push}(\mathsf{false}, \mathsf{false})\\
&\quad \cdot \mathsf{push}(\mathsf{true}, \mathsf{false})^{-1}.
\end{aligned}$$

B.6.3 Equivalence between $\mathbb{S}^3$ and $\mathbb{S}^1 * \mathbb{S}^1$

We define the pointed map $e : \mathbb{S}^3 \to \mathbb{S}^1 * \mathbb{S}^1$ as the composition

$$\mathbb{S}^3 \xrightarrow{\psi_{\mathbb{S}^2}} \mathbf{2} * \mathbb{S}^2 \xrightarrow{\mathsf{id}_{\mathbf{2}} * \psi_{\mathbb{S}^1}} \mathbf{2} * (\mathbf{2} * \mathbb{S}^1) \xrightarrow{\alpha_{\mathbf{2},\mathbf{2},\mathbb{S}^1}} (\mathbf{2} * \mathbf{2}) * \mathbb{S}^1 \xrightarrow{c * \mathsf{id}_{\mathbb{S}^1}} \mathbb{S}^1 * \mathbb{S}^1$$

and the pointed map $e^{-1} : \mathbb{S}^1 * \mathbb{S}^1 \to \mathbb{S}^3$ as the opposite composition with the inverse maps.

B.7 The main map

The map $\alpha : \mathbb{S}^1 * \mathbb{S}^1 \to \mathbb{S}^2$ is the main map which appears in the proof that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$; all the other maps appear already in the construction of the Hopf fibration or in the Freudenthal suspension theorem. In the notation of chapter 3 of this thesis, it is the map $\nabla_{\mathbb{S}^2} \circ W_{\mathbb{S}^1,\mathbb{S}^1}$. It is defined by

$$\begin{aligned} \alpha &: \mathbb{S}^1 * \mathbb{S}^1 \to \mathbb{S}^2, \\ \alpha(\mathsf{inl}(x)) &:= \mathsf{north}, \\ \alpha(\mathsf{inr}(y)) &:= \mathsf{north}, \\ \mathsf{ap}_\alpha(\mathsf{push}(x,y)) &:= \mathsf{merid}(y) \cdot \mathsf{merid}(\mathsf{base})^{-1} \cdot \mathsf{merid}(x) \cdot \mathsf{merid}(\mathsf{base})^{-1} \end{aligned}$$

and pointed by $\mathsf{idp}_{\mathsf{north}}$.

B.8 The map defining $\pi_3(\mathbb{S}^2)$

We now want to go from $\Omega^3\mathbb{S}^2$ to $\Omega^3(\mathbb{S}^1 * \mathbb{S}^1)$. We do that by going *back* one of the maps in the long exact sequence of the Hopf fibration, so it's not completely trivial. It's a priori a bit surprising that we can do it with actual loop spaces and not just homotopy groups, but this is because we can use the fact that the double loop space of the fiber $\mathbb{S}^1$ is contractible (which is stronger than just having trivial homotopy groups).

B.8.1 The Hopf fibration

Using the fact that the identity function is an equivalence and that an equality between two equivalences is determined by an equality between the underlying functions, we define the map

$$\begin{aligned} \mu &: \mathbb{S}^1 \to (\mathbb{S}^1 \simeq \mathbb{S}^1), \\ \mu(\mathsf{base}) &:= \lambda x.x, \\ \mathsf{ap}_\mu(\mathsf{loop}) &:= \mathsf{funext}(f), \end{aligned}$$

where

$$\begin{aligned} f &: (x : \mathbb{S}^1) \to x =_{\mathbb{S}^1} x, \\ f(\mathsf{base}) &:= \mathsf{loop}, \\ \mathsf{apd}_f(\mathsf{loop}) &:= \mathsf{in}^{=}(\mathsf{idp}_{\mathsf{loop}\cdot\mathsf{loop}}). \end{aligned}$$

The Hopf fibration is the dependent type

$$\begin{aligned} \mathsf{Hopf} &: \mathbb{S}^2 \to \mathsf{Type}, \\ \mathsf{Hopf}(\mathsf{north}) &:= \mathbb{S}^1, \\ \mathsf{Hopf}(\mathsf{south}) &:= \mathbb{S}^1, \\ \mathsf{ap}_{\mathsf{Hopf}}(\mathsf{merid}(x)) &:= \mathsf{ua}(\mu(x)). \end{aligned}$$

The total space of the Hopf fibration is equivalent to $\mathbb{S}^1 * \mathbb{S}^1$, which means that in particular we have a map

$$\begin{aligned} t &: (x : \mathbb{S}^2) \to (\mathsf{Hopf}(x) \to \mathbb{S}^1 * \mathbb{S}^1), \\ t(\mathsf{north}) &:= \mathsf{inl}, \\ t(\mathsf{south}) &:= \mathsf{inr}, \\ \mathsf{apd}_t(\mathsf{merid}(x)) &:= \mathsf{funext}^3(t^{\mathsf{merid}}(x)), \end{aligned}$$

where

$$\begin{aligned} t^{\mathsf{merid}} &: (x : \mathbb{S}^1)\{y, y' : \mathbb{S}^1\}(p : y =^{\mathsf{Hopf}}_{\mathsf{merid}(x)} y') \to \mathsf{inl}(y) =_{\mathbb{S}^1 * \mathbb{S}^1} \mathsf{inr}(y'), \\ t^{\mathsf{merid}}(x)(p) &:= \mathsf{push}(y, \mu(x)(y)) \cdot \mathsf{ap}_{\mathsf{inr}}(\mathsf{out}^{\mathsf{transport}}(p)), \end{aligned}$$

and then we have

$$\begin{aligned} t' &: \sum_{x:\mathbb{S}^2} \mathsf{Hopf}(x) \to \mathbb{S}^1 * \mathbb{S}^1, \\ t'(x, y) &:= t(x)(y). \end{aligned}$$

B.8.2 Looping a fibration

Let $P : B \to \mathsf{Type}$ be a dependent type over a pointed type B and such that $F := P(\star_B)$ is pointed. The total space of P is pointed by $(\star_B, \star_F)$. We define the dependent type

$$\begin{aligned} P^\Omega &: \Omega B \to \mathsf{Type}, \\ P^\Omega(p) &:= (\star_F =^P_p \star_F). \end{aligned}$$

Note that the fiber of P^Ω (over the basepoint of ΩB, i.e. $\mathsf{idp}_{\star_B}$) is definitionally equal to ΩF, and we point it by $\mathsf{idp}_{\star_F}$.

The total space of P^Ω is equivalent to the loop space of the total space of P, in particular we have a (pointed) map $\sum_{p:\Omega B}(P^\Omega(p)) \to \Omega(\sum_{x:B} P(x))$ given by 1-dimensional pairing. We can then iterate this construction, we write P^{Ω^2} for $(P^\Omega)^\Omega$ and P^{Ω^3} for $(P^{\Omega^2})^\Omega$. Note that if F is $(n+1)$-truncated, then every fiber of P^Ω is n-truncated because every fiber of P^Ω is equivalent to an identity type of F via $\mathsf{in}^{\mathsf{transport}}$ and $\mathsf{out}^{\mathsf{transport}}$.

B.8.3 Looping the Hopf fibration

Let's consider the triple looping of the Hopf fibration:

$$\mathsf{Hopf}^{\Omega^3} : \Omega^3\mathbb{S}^2 \to \mathsf{Type}.$$

The fiber of Hopf is $\mathbb{S}^1$ which is 1-truncated, hence every fiber of Hopf^{Ω^3} is (-2)-truncated, i.e. contractible. Therefore for every $p : \Omega^3\mathbb{S}^2$ there is a $q : \mathsf{Hopf}^{\Omega^3}(p)$, so we get a point (p, q) in $\sum_{p:\Omega^3\mathbb{S}^2}(\mathsf{Hopf}^{\Omega^3}(p))$. Using the maps above, we get a point in $\Omega^3(\sum_{x:\mathbb{S}^2}(\mathsf{Hopf}(x)))$, and then in $\Omega^3(\mathbb{S}^1 * \mathbb{S}^1)$ after applying $\Omega^3 t'$. This gives us the map $h : \Omega^3\mathbb{S}^2 \to \Omega^3(\mathbb{S}^1 * \mathbb{S}^1)$.

B.9 Going back to $\pi_2(\mathbb{S}^2)$

We now start decreasing the dimension by constructing a map $\Omega^3\mathbb{S}^3 \to \Omega^2\|\mathbb{S}^2\|_2$. This is the only place where we really need truncations. In the next subsection we implement truncations using regular higher inductive types. It can safely be skipped in a proof assistant with already a native support for truncations.

B.9.1 Truncations

In this subsection, we define the spheres slightly differently to make the inductive steps simpler. We define $\mathbb{S}^0 := \mathbf{2}$ and $\mathbb{S}^1 := \Sigma\mathbb{S}^0$. We do not need the other spheres at all here, so we keep the same notation for simplicity.

Given a type A and $n \geq -1$, the n-truncation of A is the higher inductive type $\|A\|_n$ with constructors

$$\begin{aligned} |-| &: A \to \|A\|_n, \\ \mathsf{hub} &: (\mathbb{S}^{n+1} \to \|A\|_n) \to \|A\|_n, \\ \mathsf{spoke} &: (f : \mathbb{S}^{n+1} \to \|A\|_n)(t : \mathbb{S}^{n+1}) \to \mathsf{hub}(f) =_{\|A\|_n} f(t). \end{aligned}$$

If A is pointed, then $\|A\|_n$ is pointed by $|\star_A|$.

Given a type B and a map $f : \mathbb{S}^{n+1} \to B$, a *filler* of f is a pair $(\mathsf{hub}_f, \mathsf{spoke}_f)$ with $\mathsf{hub}_f : B$ and $\mathsf{spoke}_f : (t : \mathbb{S}^{n+1}) \to \mathsf{hub}_f =_B f(t)$. If we have a filler for all maps $\mathbb{S}^{n+1} \to B$, we say that *all $(n+1)$-spheres in B are filled*. The hub and spoke constructors of $\|A\|_n$ state exactly that all $(n+1)$-spheres in $\|A\|_n$ are filled. We prove that $\|A\|_n$ is n-truncated and that it has the following universal property: given an n-truncated type B, a map $f : A \to B$ can be extended in a unique way to a map $\widetilde{f} : \|A\|_n \to B$ satisfying $\widetilde{f}(|x|) = f(x)$ definitionally.

We prove first that if a type B has all $(n+1)$-spheres filled, then B is n-truncated, by induction on n. It follows that $\|A\|_n$ is n-truncated.

For $n = -1$, given $x, y : B$, we define $f : \mathbb{S}^0 \to B$ by

$$\begin{aligned} f(\mathsf{true}) &:= x, \\ f(\mathsf{false}) &:= y, \end{aligned}$$

and we assumed that B has all 0-spheres filled. We then have

$$\mathsf{spoke}_f(\mathsf{true})^{-1} \cdot \mathsf{spoke}_f(\mathsf{false}) : x =_B y$$

hence B is (-1)-truncated.

For the induction step, we assume that B has all $(n+2)$-spheres filled and we prove that every identity type of B has all $(n+1)$-spheres filled. Let's take $x, y : B$. We define

the maps

$$\begin{aligned}
\mathsf{lift} &: (\mathbb{S}^{n+1} \to x =_B y) \to (\mathbb{S}^{n+2} \to B),\\
\mathsf{lift}(f)(\mathsf{north}) &:= x,\\
\mathsf{lift}(f)(\mathsf{south}) &:= y,\\
\mathsf{ap}_{\mathsf{lift}(f)}(\mathsf{merid}(t)) &:= f(t),
\end{aligned}$$

$$\begin{aligned}
\mathsf{hub}' &: (\mathbb{S}^{n+1} \to x =_B y) \to x =_B y,\\
\mathsf{hub}'_f &:= \mathsf{spoke}_{\mathsf{lift}(f)}(\mathsf{north})^{-1} \cdot \mathsf{spoke}_{\mathsf{lift}(f)}(\mathsf{south}),
\end{aligned}$$

$$\begin{aligned}
\mathsf{spoke}' &: (f : \mathbb{S}^{n+1} \to x =_B y)(t : \mathbb{S}^{n+1}) \to \mathsf{hub}'_f =_{x=_B y} f(t),\\
\mathsf{spoke}'_f &:= \mathsf{spoke}^{\mathsf{lemma}}(f, \mathsf{merid}(t)),
\end{aligned}$$

where $\mathsf{spoke}^{\mathsf{lemma}}$ has type

$$\begin{aligned}
\mathsf{spoke}^{\mathsf{lemma}} : (f : \mathbb{S}^{n+1} \to x =_B y)\{t, t' : \mathbb{S}^{n+2}\}(p : t =_{\mathbb{S}^{n+2}} t')\\
\to \mathsf{spoke}_{\mathsf{lift}(f)}(t)^{-1} \cdot \mathsf{spoke}_{\mathsf{lift}(f)}(t') =_{(\mathsf{lift}(f)(t) =_B \mathsf{lift}(f)(t'))} \mathsf{ap}_{\mathsf{lift}(f)}(p)
\end{aligned}$$

and is proved by path induction on p. This concludes.

We now prove the converse, i.e. that every n-truncated type has all $(n+1)$-spheres filled. A consequence of that is the non-dependent elimination rule of truncations: if B is n-truncated, then any map $f : A \to B$ can be extended to a map $\widetilde{f} : \|A\|_n \to B$ such that $\widetilde{f}(|x|) := f(x)$, because for the application of $\widetilde{f}$ to the other constructors of $\|A\|_n$ we can just use the hub and spoke structure of B.

For $n = -1$, we know that B is a proposition and we define hub and spoke as follows: hub_f is $f(\mathsf{true})$ and spoke_f is automatic because it's an equality in a proposition.

For $n+1$, by induction hypothesis for every $x, y : B$, we have a $\mathsf{hub}^{x,y}$ and $\mathsf{spoke}^{x,y}$ for the type $x =_B y$. We then define

$$\begin{aligned}
\mathsf{hub} &: (\mathbb{S}^{n+2} \to B) \to B,\\
\mathsf{hub}_f &:= f(\mathsf{north}),
\end{aligned}$$

$$\begin{aligned}
\mathsf{spoke} &: (f : \mathbb{S}^{n+2} \to B)(t : \mathbb{S}^{n+2}) \to \mathsf{hub}_f =_B f(t),\\
\mathsf{spoke}_f(\mathsf{north}) &:= \mathsf{idp}_{f(\mathsf{north})},\\
\mathsf{spoke}_f(\mathsf{south}) &:= \mathsf{ap}_f(\mathsf{merid}(\mathsf{north})),\\
\mathsf{apd}_{\mathsf{spoke}_f}(\mathsf{merid}(t)) &:= \mathsf{in}^{=}(\mathsf{spoke}^{f(\mathsf{north}),f(\mathsf{south})}_{\lambda w.\, \mathsf{ap}_f(\mathsf{merid}(w))}(t)^{-1}\\
&\qquad \cdot \mathsf{spoke}^{f(\mathsf{north}),f(\mathsf{south})}_{\lambda w.\, \mathsf{ap}_f(\mathsf{merid}(w))}(\mathsf{north})),
\end{aligned}$$

which shows that B has all $(n+2)$-spheres filled.

Finally we prove the uniqueness principle: if B is n-truncated and $g, h : \|A\|_n \to B$ are two functions such that $(a : A) \to g(|a|) =_B h(|a|)$, then $(x : \|A\|_n) \to g(x) =_B h(x)$. We prove it by induction on x (using the induction principle derived from the hub and spoke constructors). For the $|-|$ constructor, it's true by assumption. For the hub and spoke constructors, let's take $f : \mathbb{S}^{n+1} \to \|A\|_n$. Using hub and spoke on $\|A\|_n$ we have a filler of f, hence by applying g and h to it we get a filler of $g \circ f$ and a filler of $h \circ f$. Now by induction hypothesis, g and h agree on the image of f, hence $g \circ f = h \circ f$ and by composing the filler of $h \circ f$ with that equality, we get another filler of $g \circ f$, and we have to prove that those two fillers of $g \circ f$ are equal. Let's call $(\mathsf{hub}, \mathsf{spoke})$ and $(\mathsf{hub}', \mathsf{spoke}')$ those two fillers of $g \circ f$ and let's prove that they are equal. The type B is n-truncated, hence its identity types are also n-truncated, hence they have all $(n+1)$-spheres filled. We define the map

$$\begin{aligned} k : \mathbb{S}^{n+1} &\to \mathsf{hub} =_B \mathsf{hub}', \\ k(t) &:= \mathsf{spoke}(t) \cdot \mathsf{spoke}'(t)^{-1}. \end{aligned}$$

That map can be filled, hence we have

$$\begin{aligned} \mathsf{hub}_k &: \mathsf{hub} =_B \mathsf{hub}', \\ \mathsf{spoke}_k &: (t : \mathbb{S}^{n+1}) \to \mathsf{hub}_k =_{\mathsf{hub} =_B \mathsf{hub}'} \mathsf{spoke}(t) \cdot \mathsf{spoke}'(t)^{-1}. \end{aligned}$$

In order to prove that $(\mathsf{hub}, \mathsf{spoke})$ and $(\mathsf{hub}', \mathsf{spoke}')$ are equal, we need

$$\begin{aligned} \mathsf{hub}^= &: \mathsf{hub} =_B \mathsf{hub}', \\ \mathsf{spoke}^= &: \mathsf{spoke} =^{\lambda h.(t:\mathbb{S}^{n+1}) \to h =_B k(t)}_{\mathsf{hub}^=} \mathsf{spoke}'. \end{aligned}$$

We take $\mathsf{hub}^= := \mathsf{hub}_k$ and

$$\mathsf{spoke}^= := \mathsf{funext}^{\mathsf{dep}}(\mathsf{spoke}'^=),$$

where

$$\begin{aligned} \mathsf{spoke}'^= &: (t : \mathbb{S}^{n+1}) \to \mathsf{spoke}(t) =^{\lambda h.h =_B k(t)}_{\mathsf{hub}^=} \mathsf{spoke}'(t), \\ \mathsf{spoke}'^=(t) &:= \mathsf{in}^=(\mathsf{spoke}''^=(t)), \end{aligned}$$

where

$$\mathsf{spoke}''^= : (t : \mathbb{S}^{n+1}) \to \mathsf{spoke}(t) = \mathsf{hub}^= \cdot \mathsf{spoke}'(t)$$

is deduced from spoke_k by some coherence operation.

B.9.2 Truncated higher Hopf fibration

The fibration we define now is similar to the Hopf fibration with $\mathbb{S}^2$ instead of $\mathbb{S}^1$. The main difference is that, unlike for $\mathbb{S}^1$, there is no appropriate multiplication operation

on $\mathbb{S}^2$, there is only one on $\|\mathbb{S}^2\|_2$ so it is a bit more complicated to define. We write $|p|^1$ for $\mathsf{ap}_{\lambda w.|w|}(p)$.

We define the map

$$\begin{aligned}\mu_2 &: \mathbb{S}^2 \to \mathbb{S}^2 \to \|\mathbb{S}^2\|_2,\\ \mu_2(\mathsf{north}) &:= \lambda x.|x|,\\ \mu_2(\mathsf{south}) &:= \mu_2^{\mathsf{south}},\\ \mathsf{ap}_{\mu_2}(\mathsf{merid}(x)) &:= \mathsf{funext}(\mu_2^{\mathsf{merid}}(x)),\end{aligned}$$

where

$$\begin{aligned}\mu_2^{\mathsf{south}} &: \mathbb{S}^2 \to \|\mathbb{S}^2\|_2,\\ \mu_2^{\mathsf{south}}(\mathsf{north}) &:= |\mathsf{south}|,\\ \mu_2^{\mathsf{south}}(\mathsf{south}) &:= |\mathsf{south}|,\\ \mathsf{ap}_{\mu_2^{\mathsf{south}}}(\mathsf{merid}(y)) &:= |\,\mathsf{merid}(\mathsf{base})^{-1} \cdot \mathsf{merid}(y)|^1,\end{aligned}$$

and

$$\begin{aligned}\mu_2^{\mathsf{merid}} &: (x : \mathbb{S}^1)(y : \mathbb{S}^2) \to |y| =_{\|\mathbb{S}^2\|_2} \mu_2^{\mathsf{south}}(y),\\ \mu_2^{\mathsf{merid}}(x)(\mathsf{north}) &:= |\,\mathsf{merid}(x)|^1,\\ \mu_2^{\mathsf{merid}}(x)(\mathsf{south}) &:= |\,\mathsf{merid}(\mathsf{base})^{-1} \cdot \mathsf{merid}(x)|^1,\\ \mathsf{apd}_{\mu_2^{\mathsf{merid}}(x)}(\mathsf{merid}(y)) &:= \mathsf{in}^{=}(\mu_2^{\mathsf{merid},\mathsf{merid}}(x)(y)),\end{aligned}$$

and

$$\begin{aligned}\mu_2^{\mathsf{merid},\mathsf{merid}} : (x, y : \mathbb{S}^1) &\to |\,\mathsf{merid}(x)|^1 \cdot |\,\mathsf{merid}(\mathsf{base})^{-1} \cdot \mathsf{merid}(y)|^1\\ &= |\,\mathsf{merid}(y)|^1 \cdot |\,\mathsf{merid}(\mathsf{base})^{-1} \cdot \mathsf{merid}(x)|^1\end{aligned}$$

is defined as follows:

- When x or y is base, using the fact that $|-|^1$ commutes with composition of paths, it's easy to prove the desired equality.
- When we apply $\mu_2^{\mathsf{merid},\mathsf{merid}}$ to loop and loop, we need to construct a term in a 4-times iterated identity type of $\|\mathbb{S}^2\|_2$. But $\|\mathbb{S}^2\|_2$ is 2-truncated, hence we need to construct something in a (-2)-truncated type which is automatic.

We have now defined μ_2, but in order to use it to construct a fibration, we need to turn it into a map $\widetilde{\mu_2} : \mathbb{S}^2 \to (\|\mathbb{S}^2\|_2 \simeq \|\mathbb{S}^2\|_2)$. For $x : \mathbb{S}^2$, the underlying map of $\widetilde{\mu_2}(x)$ is defined by applying the universal property of truncation to $\mu_2(x)$. In other words, we have

$$\widetilde{\mu_2}(x)(|y|) := \mu_2(x)(y).$$

We prove that $\widetilde{\mu_2}(x)$ is an equivalence by noticing that $\widetilde{\mu_2}(\mathsf{north})$ is equal to the identity function, using the uniqueness principle for maps out of truncations, and then it

follows by induction on x that all of them are equivalences, using the fact that being an equivalence is a proposition.

We can now define the fibration we are interested in by

$$\begin{aligned}\mathsf{tHopf}_3 &: \mathbb{S}^3 \to \mathsf{Type},\\ \mathsf{tHopf}_3(\mathsf{north}) &:\equiv \|\mathbb{S}^2\|_2,\\ \mathsf{tHopf}_3(\mathsf{south}) &:\equiv \|\mathbb{S}^2\|_2,\\ \mathsf{ap}_{\mathsf{tHopf}_3}(\mathsf{merid}(x)) &:\equiv \mathsf{ua}(\widetilde{\mu_2}(x)),\end{aligned}$$

and we have $\mathsf{tHopf}_3^{\Omega^2} : \Omega^2\mathbb{S}^3 \to \mathsf{Type}$ with fiber $\Omega^2\|\mathbb{S}^2\|_2$. A point $p : \Omega^3\mathbb{S}^3$ can be seen as a loop in $\Omega^2\mathbb{S}^3$, and we can transport along it in $\mathsf{tHopf}_3^{\Omega^2}$. We define the map

$$\begin{aligned}e_3 &: \Omega^3\mathbb{S}^3 \to \Omega^2\|\mathbb{S}^2\|_2,\\ e_3(p) &:\equiv \mathsf{transport}^{\mathsf{tHopf}_3^{\Omega^2}}(p, \mathsf{idp}_{\mathsf{idp}_{|\mathsf{north}|}}).\end{aligned}$$

B.10 Loop spaces of truncations

Let A be a pointed type and $n \geq -1$, the goal here is to construct a pointed map $\kappa_{n,A} : \Omega\|A\|_{n+1} \to \|\Omega A\|_n$. Let's first prove that for $n \geq -1$, the type of n-types is $(n+1)$-truncated.

If A and B are two n-types, their identity types in the type of n-types and in Type are equivalent, because being an n-type is a proposition. Using univalence, we get that $A =_{\mathsf{Type}} B$ is equivalent to $A \simeq B$. Using the fact that $\mathsf{isequiv}$ is a mere proposition, the identity type between two equivalences between A and B is the same as the identity type between the underlying functions, and using function extensionality it is equivalent to a product of identity types of B. But B is n-truncated, hence the identity types of $A \simeq B$ are $(n-1)$-truncated, hence $A \simeq B$ is n-truncated, hence the type of n-types is $(n+1)$-truncated.

We can now define the family of n-types

$$\begin{aligned}P_{n,A} &: \|A\|_{n+1} \to \mathsf{Type},\\ P_{n,A}(|x|) &:\equiv \|\star_A =_A x\|_n.\end{aligned}$$

The function $\kappa_{n,A}$ is then defined by

$$\begin{aligned}\kappa_{n,A} &: \Omega\|A\|_{n+1} \to \|\Omega A\|_n,\\ \kappa_{n,A}(p) &:\equiv \mathsf{transport}^{P_{n,A}}(p, |\mathsf{idp}_{\star_A}|).\end{aligned}$$

B.11 Down one more dimension

We're almost done, we now just need to go from $\|\Omega^2\mathbb{S}^2\|_0$ to $\Omega\mathbb{S}^1$. This step is quite easy using the Hopf fibration. Using the fact that $\Omega\mathbb{S}^1$ is a set, we only have to build a map

from $\Omega^2\mathbb{S}^2$ to $\Omega\mathbb{S}^1$. That map is defined by

$$e_2' : \Omega^2\mathbb{S}^2 \to \Omega\mathbb{S}^1,$$
$$e_2'(p) := \mathsf{transport}^{\mathsf{Hopf}^\Omega}(p, \mathsf{idp}_{\mathsf{base}}).$$

This concludes the definition of n.

Bibliography

[Ara10] Dimitri Ara. *Sur les ∞-groupoïdes de Grothendieck et une variante ∞-catégorique*. PhD thesis. 2010. URL: http://iml.univ-mrs.fr/~ara/files/these.pdf.

[AW09] Steve Awodey and Michael A. Warren. *Homotopy theoretic models of identity types*. Mathematical Proceedings of the Cambridge Philosophical Society (2009). arXiv: 0709.0248.

[BCH14] Marc Bezem, Thierry Coquand, and Simon Huber. *A model of type theory in cubical sets* (2014). URL: http://www.cse.chalmers.se/~coquand/mod1.pdf. Preprint.

[BG11] Benno van den Berg and Richard Garner. *Types are weak ω-groupoids*. Proceedings of the London Mathematical Society 102 (2011). arXiv: 0812.0298.

[Cav15] Evan Cavallo. *Synthetic Cohomology in Homotopy Type Theory* (2015). URL: http://www.cs.cmu.edu/~rwh/theses/cavallo.pdf. Preprint.

[Coh+15] Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. *Cubical Type Theory: a constructive interpretation of the univalence axiom*. Preprint (2015). URL: https://www.math.ias.edu/~amortberg/papers/cubicaltt.pdf.

[Fre37] Hans Freudenthal. *Über die Klassen der Sphärenabbildungen, I. Große Dimensionen*. Composition Math. 5 (1937). URL: http://www.math.rochester.edu/people/faculty/doug/otherpapers/freude1.pdf.

[Hat02] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002. URL: https://www.math.cornell.edu/~hatcher/AT/ATpage.html.

[Hou+16] Kuen-Bang Hou, Eric Finster, Daniel R. Licata, and Peter LeFanu Lumsdaine. *A Mechanization of the Blakers-Massey Connectivity Theorem in Homotopy Type Theory*. LICS (2016). arXiv: 1605.03227.

[KLV12] Chris Kapulkin, Peter LeFanu Lumsdaine, and Vladimir Voevodsky. *The Simplicial Model of Univalent Foundations* (2012). arXiv: 1211.2851. Preprint.

[LB13] Daniel R. Licata and Guillaume Brunerie. *$\pi_n(S^n)$ in Homotopy Type Theory*. Invited paper, CPP (2013). URL: http://dlicata.web.wesleyan.edu/pubs/lb13cpp/lb13cpp.pdf.

[LB15] Daniel R. Licata and Guillaume Brunerie. *A Cubical Approach to Synthetic Homotopy Theory*. LICS (2015). URL: http://dlicata.web.wesleyan.edu/pubs/lb15cubicalsynth/lb15cubicalsynth.pdf.

[LF14] Daniel R. Licata and Eric Finster. *Eilenberg-MacLane Spaces in Homotopy Type Theory*. LICS (2014). URL: http://dlicata.web.wesleyan.edu/pubs/lf14em/lf14em.pdf.

[LS] Peter Lumsdaine and Michael Shulman. *Semantics and syntax of higher inductive types*. In preparation.

[LS13] Daniel R. Licata and Michael Shulman. *Calculating the Fundamental Group of the Circle in Homotopy Type Theory*. LICS (2013). arXiv: 1301.3443.

[Lur09] Jacob Lurie. *Higher topos theory*. Princeton University Press, 2009. arXiv: math/0608040.

[ML75] Per Martin–Löf. *An Intuitionistic Theory of Types: Predicative Part*. Proceedings of the Logic Colloquium (1975). URL: http://archive-pml.github.io/martin-lof/pdfs/An-Intuitionistic-Theory-of-Types-Predicative-Part-1975.pdf.

[Shu15] Michael Shulman. *Univalence for inverse EI diagrams* (2015). arXiv: 1508.02410. Preprint.

[Soj15] Kristina Sojakova. *Higher Inductive Types as Homotopy-Initial Algebras*. Proceedings of the Symposium on Principles of Programming Languages (POPL) (2015). arXiv: 1402.0761.

[UF13] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study, Princeton, NJ, 2013. URL: http://homotopytypetheory.org/book.

Online Resources

[Agd] The Agda team. *Agda's documentation*. Chap. Rewriting. URL: http://agda.readthedocs.org/en/latest/language/rewriting.html.

[Agda] The Agda Team. *The Agda Wiki*. URL: http://wiki.portal.chalmers.se/agda/pmwiki.php.

[Bru15] Guillaume Brunerie. *The 3×3-lemma in Agda*. 2015. URL: https://github.com/HoTT/HoTT-Agda/blob/master/homotopy/3x3/Commutes.agda.

[Cav14] Evan Cavallo. *JoinAssocCubical.agda*. 2014. URL: https://github.com/HoTT/HoTT-Agda/blob/master/homotopy/JoinAssocCubical.agda.

[Coq] The Coq Team. *The Coq Proof Assistant*. URL: https://coq.inria.fr/.

[HoTTAgda] *The HoTT-Agda library*. URL: https://github.com/HoTT/HoTT-Agda.

[HoTTCoq] *The HoTT-Coq library*. URL: https://github.com/HoTT/HoTT.

[HoTTLean] *The HoTT-Lean library*. URL: https://github.com/leanprover/lean/tree/master/hott.

[Hou13] Kuen-Bang Hou. *BlakersMassey.agda*. 2013. URL: https://github.com/HoTT/HoTT-Agda/blob/1.0/Homotopy/BlakersMassey.agda.

[Lean] *The Lean Theorem Prover*. URL: https://leanprover.github.io/.

[Shu11] Michael Shulman. *A formal proof that $\pi_1(\mathbb{S}^1) = \mathbb{Z}$*. The Homotopy Type Theory Blog (2011). URL: http://homotopytypetheory.org/2011/04/29/a-formal-proof-that-pi1s1-is-z/.

[Shu13] Michael Shulman. *Cohomology*. The Homotopy Type Theory Blog (2013). URL: `https://homotopytypetheory.org/2013/07/24/cohomology/`.

[Unimath] *The Unimath library*. URL: `https://github.com/UniMath/UniMath`.

Introduction (français)

L'objectif de cette thèse est de démontrer le théorème suivant dont l'énoncé et la démonstration seront expliqués en temps voulu.

Théorème 1. *On a un isomorphisme de groupes*

$$\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}.$$

À proprement parler, c'est un théorème bien connu en théorie de l'homotopie classique, démontré par Freudenthal dans [Fre37] (voir aussi [Hat02, corollary 4J.4]). La principale différence est que dans cette thèse on se place en *théorie des types homotopiques* (aussi connu sous le nom de *fondations univalentes*) qui est un nouveau cadre pour faire des mathématiques, introduit par Vladimir Voevodsky en 2009, et qui est particulièrement bien adapté pour la théorie de l'homotopie. Du point de vue de la théorie de l'homotopie, la différence la plus frappante entre la théorie de l'homotopie classique et la théorie des types homotopiques est qu'en théorie des types homotopiques, *toutes* les constructions sont invariantes par équivalences d'homotopie. Un des avantages est que toutes les constructions et toutes les démonstrations faites dans ce cadre sont complètement indépendantes de la définition de la notion d'« espace ». En particulier, rien ne dépend de la topologie générale ou de la combinatoire des ensembles simpliciaux. De plus, les constructions et les démonstrations ont souvent un aspect plus « homotopique », comme nous espérons que le lecteur en sera convaincu après la lecture de cette thèse.

Cependant, cela pose un certain nombre de défis, étant donné qu'il n'est pas clair a priori quels sont les concepts qui peuvent ou ne peuvent pas être définis d'une façon purement invariante par homotopie. Par exemple, même si la cohomologie singulière est invariante par homotopie, la définition classique utilise l'ensemble des cochaînes singulières qui, lui, n'est pas invariant par homotopie. Par conséquent, la définition classique ne peut pas être reproduite telle quelle en théorie des types homotopiques. Un exemple encore plus simple est le revêtement universel du cercle qui est défini classiquement en utilisant la fonction exponentielle $\mathbb{R} \to \mathbb{S}^1$, mais il se trouve que cette fonction est homotope à une fonction constante. La théorie des types homotopiques nous donne divers outils pour travailler de façon complètement invariante par homotopie et dans cette thèse on montre comment démontrer le théorème 1 en théorie des types homotopiques, en partant essentiellement de zéro.

Un autre avantage de la théorie des types homotopiques par rapport à la théorie de l'homotopie classique est que les démonstrations écrites en théorie des types homoto-

piques sont beaucoup plus appropriées à une vérification formelle sur ordinateur. Même si le travail présenté ici n'a pas encore été formalisé, beaucoup de résultats intermédiaires (en particulier des deux premiers chapitres) ont été formalisés par diverses personnes, voir par exemple les bibliothèques [HoTTCoq] et [Unimath] pour Coq, [HoTTAgda] pour Agda et [HoTTLean] pour Lean.

Contenu de la thèse Les deux premiers chapitres de cette thèse rappellent les bases de la théorie des types homotopiques. Une référence alternative est le livre [UF13], mais on a essayé ici d'être plus concis et de garder en vue notre objectif principal. Cependant, il se peut que le style de présentation soit similaire entre [UF13] et l'introduction et les deux premiers chapitres de cette thèse. La plupart du contenu des quatre derniers chapitres est nouveau en théorie des types homotopiques, même si les concepts sont bien connus en théorie de l'homotopie classique. La définition de la notion d'∞-groupoïde faible présentée dans le premier appendice est également nouvelle.

Dans le chapitre 1, on introduit tous les concepts de base de la théorie des types homotopiques, c'est-à-dire tous les constructeurs de types et en particulier l'axiome d'univalence et les types inductifs supérieurs. On énonce également le lemme 3×3 et le lemme d'aplatissement dans les sections 1.8 et 1.9, qui seront utilisés à plusieurs endroits. Finalement on parle de types et d'applications n-tronqués et de troncations. La notion de type n-tronqué correspond à la notion classique de n-type d'homotopie, c'est-à-dire un espace qui n'a pas de contenu homotopique en dimension supérieure à n, et la troncation est une opération transformant n'importe quel espace en un espace n-tronqué d'une façon universelle. À nouveau, tout ceci est standard en théorie des types homotopiques.

Dans le chapitre 2, on définit les groupes d'homotopie des sphères. Le groupe $\pi_k(\mathbb{S}^n)$ est défini comme étant la 0-troncation (c'est-à-dire l'ensemble des composantes connexes) de l'espace des lacets itérés de dimension k dans $\mathbb{S}^n$. On démontre ensuite que $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$, qui est un résultat initialement dû à Michael Shulman en 2011 et qui apparaît dans [UF13, section 8.1], voir aussi [Shu11] et [LS13]. L'idée est qu'en théorie des types homotopique, afin de définir une fibration on ne donne pas une application de l'espace total vers la base. Au lieu de cela, on donne directement la fibre au dessus de chaque point de la base. Dans le cas d'une fibration au dessus du cercle, il est suffisant de donner la fibre au dessus du point base de $\mathbb{S}^1$ et l'action sur la fibre du lacet faisant le tour du cercle. Dans le cas qui nous intéresse, la fibre est l'espace des entiers relatifs $\mathbb{Z}$ et le lacet faisant le tour du cercle agit dessus en ajoutant 1. Cela donne une fibration au dessus de $\mathbb{S}^1$ et on peut montrer que son espace total est contractile, ce qui donne l'isomorphisme $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$. On définit ensuite la notion de n-connectivité et on démontre diverses propriétés des espaces et des application n-connectés, ce qui nous permet de montrer que $\pi_k(\mathbb{S}^n)$ est trivial pour tout $k < n$. Ce résultat a déjà été obtenue dans [UF13, section 8.3] avec une démonstration un peu plus compliquée également due à l'auteur. Finalement, on définit la fibration de Hopf, qui est une fibration au dessus de $\mathbb{S}^2$, de fibre $\mathbb{S}^1$ et d'espace total $\mathbb{S}^3$. L'idée de la définition de la fibration de Hopf est la suivante. Afin de définir une fibration au dessus de $\mathbb{S}^2$, il suffit de donner la fibre N au dessus du pôle

nord, la fibre S au dessus du pôle sud et, pour tout élément x de $\mathbb{S}^1$, une équivalence entre N et S qui décrit se qui se passe quand on se déplace dans la fibration du pôle nord au pôle sud le long du méridien correspondant à x. Dans le cas de la fibration de Hopf, on prend $N, S := \mathbb{S}^1$ et l'équivalence entre N et S correspondant à x est l'opération de multiplication par x. La fibration de Hopf a été définie pour la première fois par Peter Lumsdaine, d'une façon légèrement différente, mais sans démonstration du fait que son espace total est équivalent à $\mathbb{S}^3$. La construction présentée ici a été également décrite dans [UF13, section 8.5].

Dans le chapitre 3, on définit la fibration de Hopf en suivant une idée suggérée par André Joyal. Pour tout type A, on définit une famille d'espaces $(J_n A)$ et on démontre que leur colimite est équivalente à l'espace des lacets de la suspension de A. Pour cela, on définit un autre espace JA et on démontre que JA est équivalent à la fois à la colimite de $(J_n A)$ et à l'espace des lacets de la suspension de A. La construction de James donne une suite d'approximations de l'espace des lacets de la suspension de A ce qui, avec le théorème de Blakers–Massey, nous permet de montrer qu'il existe un entier naturel n tel que $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$. Cet entier n est défini en utilisant le produit de Whitehead, plus précisément c'est l'image du produit de Whitehead $[\mathsf{id}_{\mathbb{S}^2}, \mathsf{id}_{\mathbb{S}^2}]$, qui est un élément de $\pi_3(\mathbb{S}^2)$, par l'équivalence $\pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$ construite en utilisant la fibration de Hopf.

Dans le chapitre 4, on étudie le produit smash et sa structure monoïdale symétrique. En particulier, on construit une famille d'équivalences $\mathbb{S}^n \wedge \mathbb{S}^m \simeq \mathbb{S}^{n+m}$ compatible, en un certain sens, avec l'associativité et la commutativité du produit smash. La construction de la structure monoïdale symétrique sera essentiellement admise, mais on donne l'intuition de la construction.

Dans le chapitre 5, on commence par définir, pour tout entier naturel n, l'espace d'Eilenberg–MacLane $K(\mathbb{Z}, n)$ comme étant la n-troncation de la sphère $\mathbb{S}^n$, et le $n$ième groupe de cohomologie d'un espace X comme étant la 0-troncation de l'espace de fonctions $X \to K(\mathbb{Z}, n)$. On définit ensuite le produit cup en tant qu'application $K(\mathbb{Z}, n) \wedge K(\mathbb{Z}, m) \to K(\mathbb{Z}, n+m)$ en prenant le produit smash des deux applications $\mathbb{S}^n \to K(\mathbb{Z}, n)$ et $\mathbb{S}^m \to K(\mathbb{Z}, m)$ et en montrant, en utilisant les propriétés de connectivité des applications, qu'on peut essentiellement l'inverser. Les propriétés du produit smash du chapitre 4 sont alors utilisées pour montrer que le produit cup est associatif et commutatif gradué. Finalement, on définit l'invariant de Hopf d'une application $f : \mathbb{S}^{2n-1} \to \mathbb{S}^n$ en utilisant la structure du produit cup sur le pushout $\mathbf{1} \sqcup^{\mathbb{S}^{2n-1}} \mathbb{S}^n$ et on démontre que pour tout nombre pair n, une certaine application $\mathbb{S}^{2n-1} \to \mathbb{S}^n$ provenant de la construction de James a un invariant de Hopf égal à 2. Cela montre en particulier que l'entier n défini au chapitre 3 est égal soit à 1, soit à 2, et que le groupe $\pi_{4n-1}(\mathbb{S}^{2n})$ est infini pour tout entier naturel n.

Finalement, dans le chapitre 6, on construit la suite exacte de Gysin, qui est une suite exacte longue de groupes de cohomologie associée à toute fibration où la base est 1-connexe et les fibres sont des sphères. Cette suite exacte décrit une partie de la structure multiplicative de la cohomologie de la base. On définit ensuite $\mathbb{C}P^2$ comme étant le pushout $\mathbf{1} \sqcup^{\mathbb{S}^3} \mathbb{S}^2$ pour l'application de Hopf $\mathbb{S}^3 \to \mathbb{S}^2$, on construit une fibration de cercles au dessus de $\mathbb{C}P^2$ d'une façon similaire à la construction de la fibration de

Hopf et on calcule son anneau de cohomologie en utilisant la suite exacte de Gysin. Ceci démontre que l'invariant de Hopf de l'application de Hopf est égal à ± 1 et que $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$.

Dans l'appendice A, on présente une définition élémentaire des ∞-groupoïdes faibles, basée sur des idées venant de théorie des types homotopiques, ainsi qu'une démonstration du fait que tout type en théorie des types homotopiques a une structure d'∞-groupoïde faible.

Dans l'appendice B, on donne une définition complète de l'entier n défini à la fin du chapitre 3 et qui satisfait $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$. La raison est que, comme on le verra un peu plus tard, calculer cet entier à partir de sa définition est un problème ouvert important en théorie des types homotopiques et il est donc utile pour les personnes essayant de résoudre ce problème d'avoir la définition complète en un seul endroit.

Analytique et synthétique La différence principale entre la théorie de l'homotopie classique et la théorie des types homotopiques est que la première est *analytique* alors que la deuxième est *synthétique*. Afin de comprendre la différence entre théorie de l'homotopie analytique et synthétique, il est utile de retourner à la géométrie élémentaire.

La géométrie analytique est la géométrie dans le sens de Descartes. L'objet que l'on étudie est l'ensemble $\mathbb{R}^2$, les points sont définis comme étant des couples (x, y) de nombres réels et les droites sont définies comme étant les ensembles de points satisfaisant une équation de la forme $ax + by = c$. Ensuite, afin de démontrer quelque chose, on utilise les propriétés de $\mathbb{R}^2$. Par exemple, on peut déterminer si deux droites se croisent en résolvant un certain système d'équations.

En revanche, la géométrie synthétique est la géométrie dans sens d'Euclide. Les points et les lignes ne sont pas définies en fonction d'autres notions, il s'agit de notions primitives, et on se donne une collection d'axiomes qui spécifie comment elles sont censées se comporter. Ensuite, afin de démontrer quelque chose, on doit utiliser les axiomes. Par exemple on ne peut pas utiliser l'équation d'une droite ou les coordonnées d'un point parce que les droites n'ont pas d'équations et les points n'ont pas de coordonnées.

La géométrie analytique peut être utilisée pour justifier la géométrie synthétique. En effet, la géométrie analytique donne une signification aux notions de point et de ligne et tous les axiomes de la géométrie synthétique peuvent être démontrés en géométrie analytique. Ainsi les axiomes sont consistants et tout ce qui est vrai en géométrie synthétique est également vrai en géométrie analytique. La réciproque n'est pas vraie, donc on pourrait penser que la géométrie synthétique est moins puissante que la géométrie analytique étant donné que moins de théorèmes sont démontrables. Mais d'un autre côté on peut également considérer que la géométrie synthétique est *plus* puissante que la géométrie analytique, parce que les théorèmes qui peuvent être démontrés sont plus généraux. Ils sont valides pour n'importe quelle interprétation des concepts primitifs qui satisfait les axiomes, alors qu'une démonstration en géométrie analytique est par nature valide uniquement dans $\mathbb{R}^2$. Un autre inconvénient de la géométrie analytique est qu'en réduisant la géométrie à la résolution d'équations il est facile de perdre de vue l'intuition géométrique. En résumé, en géométrie analytique on donne une définition explicite

des concepts qui nous intéressent et on peut démontrer beaucoup de choses mais on est restreint à ce modèle particulier, alors qu'en géométrie synthétique on ne fait qu'axiomatiser les propriétés de base des concepts qui nous intéressent, moins de théorèmes sont démontrables mais ils sont applicable plus largement et sont plus proches de l'intuition géométrique.

La situation en théorie de l'homotopie est très similaire. En théorie de l'homotopie analytique (ou théorie de l'homotopie classique), la sphère $\mathbb{S}^n$ est définie comme étant l'ensemble $\{(x_0, \ldots, x_n) \in \mathbb{R}^{n+1}, x_0^2 + \cdots + x_n^2 = 1\}$ équipé de la topologie appropriée, les fonctions continues sont les fonctions qui préservent la topologie de façon appropriée, et $\pi_4(\mathbb{S}^3)$ est défini comme étant le quotient de l'ensemble des fonctions continues pointées $\mathbb{S}^4 \to \mathbb{S}^3$ par la relation d'homotopie. On peut ensuite utiliser diverses techniques pour démontrer que $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$, c'est-à-dire que $\pi_4(\mathbb{S}^3)$ contient exactement deux éléments.

En théorie de l'homotopie synthétique, qui est ce sur quoi cette thèse porte, la notion d'espace ne provient *pas* de la topologie. Au lieu de cela, la notion d'espace est considérée comme une notion primitive (sous le nom *type*) et on a également les notions primitives de *point* et de *chemin* entre deux points. En particulier, un chemin n'est pas vu comme étant une fonction continue depuis l'intervalle, c'est simplement une notion primitive. On introduit aussi une notion primitive de *fonction continue*. Notons qu'en théorie de l'homotopie classique, on doit d'abord définir ce qu'est une fonction possiblement-non-continue avant de pouvoir définir ce qu'est une fonction continue, mais ici on prend directement le concept de fonction continue comme étant primitif. Pour nous, une fonction continue n'est *pas* une fonction possiblement-non-continue qui a la propriété additionnelle d'être continue, il n'y a pas du tout de notion de fonction possiblement-non-continue. Ainsi, l'adjectif « continue » est superflu et on utilisera simplement le mot « fonction » ou « application » pour désigner ce qui correspondrait à une « fonction continue » en théorie de l'homotopie classique.

Divers espaces sont également axiomatisés, par exemple l'espace $\mathbb{N}$ des entiers naturels est axiomatisé avec un élément 0, une fonction $S : \mathbb{N} \to \mathbb{N}$ et le principe d'induction/récursion. Le cercle est axiomatisé avec un point appelé base, un chemin appelé loop de base vers base, et un principe similaire d'induction/récursion qui dit intuitivement que le cercle est librement engendré par base et loop. On décrit de façon similaire les sphères de dimension supérieure $\mathbb{S}^n$ et l'ensemble des composantes connexes d'un espace. En combinant tout ceci avec la notion de fonction (continue) mentionnée ci-dessus, on peut définir $\pi_4(\mathbb{S}^3)$ et on verra qu'on peut encore montrer qu'il est isomorphe au groupe $\mathbb{Z}/2\mathbb{Z}$.

Théorie des types La théorie des types homotopiques est une variante de la théorie des types et plus précisément de la théorie intuitionniste des types de Per Martin–Löf (qu'on appellera simplement *théorie des types dépendants* ici), qui a été introduite dans les années 1970 en tant que fondements des mathématiques constructives (voir [ML75]). Les mathématiques constructives sont une philosophie des mathématiques basée sur l'idée qu'afin de démontrer qu'un certain objet existe, il faut donner une méthode

permettant de le construire. Cela fonctionne en restreignant les principes logiques que l'on peut utiliser et en n'autorisant que ceux qui sont constructifs. Une démonstration en mathématiques constructives n'est pas nécessairement présentée sous la forme d'un algorithme, mais un algorithme peut toujours en être extrait. Par conséquent, les mathématiques constructives rejettent les principes comme l'axiome du choix, qui affirme l'existence d'une fonction sans donner de moyen de la calculer, et le raisonnement par l'absurde qui permet de montrer que quelque chose existe en montrant simplement qu'il ne peut pas ne pas exister. En particulier, une démonstration du fait qu'il existe un entier naturel ayant une certaine propriété doit donner (au moins implicitement) une méthode pour calculer cet entier. Ce n'est pas vrai en mathématiques classiques. Par exemple considérons l'entier $n \in \mathbb{N}$ défini comme étant le plus petit nombre parfait impair, ou 0 s'il n'existe pas de nombre parfait impair. En mathématiques classiques, c'est une définition valide de n mais qui ne donne aucun moyen de le calculer. En effet, au moment où j'écris ces lignes personne ne sait si n est égal à 0 ou pas. En revanche, ce n'est pas considéré une définition valide en mathématiques constructives parce qu'on a utilisé le principe du tiers exclu (soit il existe un nombre parfait impair, soit il n'en existe pas) qui n'est pas constructif. Il y a diverses variantes des mathématiques constructives et notons que celle utilisée ici, la théorie des types homotopiques, n'est pas incompatible avec la logique classique. Il serait tout à fait possible de rajouter l'axiome du choix ou le tiers exclu mais l'inconvénient est que la constructivité, qui est un des principaux avantages de la théorie des types, serait perdue.

En théorie des types dépendants, les notions primitives sont les *types* et les *éléments de types* (ou *termes*). On écrit $u : A$ l'énoncé que u est un élément du type A. Intuitivement, on peut imaginer un type comme étant une sorte d'ensemble, mais il y a plusieurs différences avec la théorie des ensembles traditionnelle. Les éléments de types n'existent pas isolément, il s'agit toujours d'éléments *d'un certain type* qui fait intrinsèquement partie de la nature de l'élément. Le type d'un élément est toujours connu et cela n'a pas de sens de « démontrer » qu'un élément u est de type A. Ceci est similaire au fait que cela n'a pas de sens de « démontrer » que $x^2 + y^2 = 0$ est une équation. Il suffit de la regarder et on voit que c'est une équation et pas une matrice. De plus, le type d'un élément est toujours unique (modulo règles de réduction, comme on le verra un peu plus loin). Par exemple on ne peut pas dire que le nombre 2 a à la fois le type $\mathbb{N}$ et le type $\mathbb{Q}$. Au lieu de cela, il y a deux éléments différents, le premier étant $2_{\mathbb{N}}$ de type $\mathbb{N}$ et le deuxième étant $2_{\mathbb{Q}}$ de type $\mathbb{Q}$ (qui peuvent être écrits tous les deux 2 s'il n'y a pas de risque de confusion), et qui satisfont $i(2_{\mathbb{N}}) = 2_{\mathbb{Q}}$ pour $i : \mathbb{N} \to \mathbb{Q}$ l'inclusion canonique. De façon similaire, étant donné un nombre rationnel $q : \mathbb{Q}$, on ne peut pas demander si q est de type $\mathbb{N}$. Le nombre q est de type $\mathbb{Q}$ par nature et $\mathbb{Q}$ est différent de $\mathbb{N}$. Par contre, ce qu'on peut demander est s'il existe un entier naturel $k : \mathbb{N}$ tel que $i(k) = q$. C'est ce qu'il faudrait faire pour démontrer que q est un entier naturel.

Les mathématiques sont traditionellement basées sur un système à deux niveaux : le niveau logique où vivent les propositions et les démonstrations, et le niveau mathématique où vivent les objets mathématiques. Le niveau logique est utilisée pour raisonner sur le niveau mathématique. Par exemple, construire un certain objet mathématique est

une activité qui se passe dans le niveau mathématique, alors que démontrer un théorème est une activité qui se passe dans le niveau logique. En théorie des types dépendants, ces deux niveaux sont fusionnées en un seul où vivent les types et leurs éléments. En plus de représenter des objets mathématiques, les types jouent aussi les rôle des propositions (logiques) et leurs éléments jouent le rôle des démonstrations de ces propositions. Démontrer une certaine proposition consiste en la construction d'un élément du type correspondant. Par exemple, démontrer une implication $A \implies B$ corresponds à la construction d'un élément dans le type de fonctions $A \to B$, c'est-à-dire c'est une fonction transformant une démonstration de A en une démonstration de B. Démontrer une conjonction $A \wedge B$ correspond à la construction d'un élément dans le type produit $A \times B$, c'est-à-dire c'est un couple formé d'une démonstration de A et d'une démonstration de B. Cette correspondance entre types et propositions et entre éléments de types et démonstrations est connue sous le nom de *correspondance de Curry–Howard*. On distinguera parfois entre les types « vus comme des propositions » et les types « vus comme des types » afin d'expliquer l'intuition derrière certaines constructions, mais la différence entre les deux est souvent floue. Par exemple, le type $A \simeq B$ peut être vu à la fois comme la proposition « A et B sont isomorphes » et comme le type de tous les isomorphismes entre A et B. En effet, en mathématiques constructives, démontrer que A et B sont isomorphes est la même chose que construire un isomorphisme entre les deux.

Le mot « dépendants » dans « théorie des types dépendants » fait référence au fait que les types peuvent dépendre d'éléments d'autres types. De tels types sont appelés des *types dépendants* ou des *familles de types*. Étant donné un type A, avoir un type dépendant B au dessus de A signifie que pour tout élément $a : A$ on a un type $B(a)$. Les types dépendants sont essentiels pour la représentation des propositions quantifiées comme on le verra dans le chapitre 1. Par exemple, une proposition dépendant d'un entier naturel $n : \mathbb{N}$ est représentée par un type dépendant de la variable n. Un type dépendant B au dessus de A où tous les types $B(a)$ sont vus comme des propositions est appelé un *prédicat sur* A.

La propriété de constructivité de la théorie des types dépendants nous permet de le voir comme un langage de programmation. En théorie des types dépendants, toutes les constructions primitives ont des *règles de réduction* qui expliquent essentiellement comment exécuter les programmes du langage. Tous les éléments de types peuvent ainsi être vus comme des programmes et peuvent être exécutés, simplement en appliquant répétitivement les règles de réduction. Il n'y a pas de boucle infinie en théorie des types, tous les programmes terminent et on obtient donc toujours un résultat quand on exécute un programme. Du point de vue mathématique, les règles de réduction sont les équation de définition des notion primitives, et appliquer une règle de réduction correspond à remplacer quelque chose par sa définition. Deux éléments u et v d'un type A sont dits *définitionnellement égaux* s'ils deviennent syntaxiquement égaux après qu'on ait tout remplacé par leurs définitions, c'est-à-dire après avoir exécuté u et v. Une règle importante de la théorie des types, connue sous le nom de *règle de conversion*, stipule que si u est de type A et que A est définitionnellement égal à A', alors u est aussi de type A'. En particulier, les types sont uniques seulement à égalité définitionnelle près, mais l'égalité

définitionnelle est décidable car il s'agit simplement de développer les définitions. De la même façon que cela n'a pas de sens de démontrer qu'un terme u est de type A, cela n'a pas non plus de sens de démontrer que deux termes ou deux types sont définitionnellement égaux. C'est quelque chose qui peut simplement être vérifié algorithmiquement.

Étant donné la correspondance entre démonstrations et éléments de types, il en résulte que les démonstrations elles-mêmes peuvent être exécutées, ce qui donne à la théorie des types dépendants sa nature constructive. Par exemple, étant donné une démonstration qu'il existe un entier naturel ayant une certaine propriété, on peut exécuter la démonstration et le résultat que l'on obtient est un couple de la forme (n, p) où n est un entier naturel de la forme 0, ou 1, ou 2, etc. (c'est-à-dire qu'on connaît sa valeur), et p est une démonstration du fait que n satisfait la propriété. Cette relation étroite entre la théorie des types et l'informatique a donné lieu au développement d'*assistants de preuve* comme Coq, Agda ou Lean (voir [Coq], [Agda], [Lean]). Il s'agit essentiellement de compilateurs pour la théorie des types dépendants avec diverses fonctionnalités pour les rendre plus faciles à utiliser. Dans un assistant de preuve, on peut énoncer un théorème en définissant le type correspondant et on peut ensuite le démontrer en construisant un terme (c'est-à-dire en écrivant un programme) ayant ce type. Si l'assistant de preuve l'accepte, cela signifie que le programme est bien typé et que la démonstration est donc correcte.

Théorie des types homotopiques La théorie des types dépendants est très fructueuse mais elle souffre de quelques problèmes, en particulier en ce qui concerne le traitement de l'égalité. Étant donné un type A et deux éléments $u, v : A$, la proposition « u est égal à v » est réifiée en un type $u =_A v$ qu'on appelle le *type identité* (ses éléments sont les témoins de l'égalité entre u et v). Martin–Löf a donné plusieurs versions de la théorie des types dépendants avec différentes règles pour les types identité. Dans une de ces versions, appelée la *théorie des types extensionnelle*, les types identité se comportent de façon agréable mais le typage n'est pas décidable, c'est-à-dire qu'il n'y a pas d'algorithme vérifiant qu'un terme a un type donné. C'est une propriété en général indésirable pour une théorie des types. Dans une autre version, appelée la *théorie des types intensionnelle*, les règles des types identité sont différentes et le typage est décidable. Cependant, le traitement de l'égalité en théorie des types intensionnelle n'est pas tout le temps satisfaisant. Par exemple, deux fonctions $f, g : A \to B$ peuvent vérifier $f(x) = g(x)$ pour tout $x : A$ sans être égales elles-mêmes en tant que fonctions. Définir le quotient d'un ensemble par une relation d'équivalence est aussi plutôt problématique. Un problème différent est le principe d'*unicité des preuves d'égalités* qui stipule que pour tous $u, v : A$, tous les éléments de $u =_A v$ sont égaux, n'est plus démontrable, ce qui est contraire à l'intuition qui était derrière les types identité. En effet, l'idée des types identité dans la théorie des types de Martin–Löf est que tout type représente un ensemble et que $u =_A v$ représente l'ensemble ayant exactement un élément si u et v sont égaux ou l'ensemble vide si u et v sont différents.

La théorie des types homotopiques est basée sur la théorie des types intensionnelle et résoud ce dernier problème en changeant l'intuition derrière les types et les types identité.

En théorie des types homotopiques, les types ne sont plus vus comme des ensembles mais comme des *espaces*, les types dépendants sont vus comme des *fibrations*, et le type identité $u =_A v$ est vu comme le type des *chemins continus* de u vers v dans l'espace A. De façon plutôt surprenante, on peut montrer que sous cette interprétation toutes les règles de la théorie des types intensionnelle sont toujours vérifiées. De plus, dans cette interprétation, l'unicité des preuves d'égalités n'est plus une propriété désirable. Étant donné deux points u et v dans un espace A, il peut y avoir beaucoup de chemins non homotopes entre u et v et beaucoup d'homotopie non homotopes entre deux chemins, et ainsi de suite.

Cette connection entre la théorie des types et la théorie de l'homotopie a été découverte vers 2006 indépendemment par Vladimir Voevodsky et par Steve Awodey et Michael Warren dans [AW09]. Ensuite, en 2009, Vladimir Voevodsky a énoncé l'*axiome d'univalence*, a démontré sa consistance dans le modèle simplicial et a commencé le projet de formalisation des mathématiques dans ce système, la théorie des types intensionnelle augmentée de l'axiome d'univalence, sous le nom de *fondations univalentes*. Étant donné un univers Type, c'est-à-dire un type dont les éléments sont eux-mêmes des types, et deux éléments A et B de Type, l'axiome d'univalence identifie le type identité $A =_{\mathsf{Type}} B$ avec le type des équivalences $A \simeq B$. Cet axiome rend précise l'idée que « deux structures isomorphes ont les mêmes propriétés », qui est souvent utilisé implicitement en mathématiques. Notons que l'axiome d'univalence n'est pas compatible avec le principe d'unicité des preuves d'égalité, il implique par exemple qu'il y a deux égalités $\mathbf{2} =_{\mathsf{Type}} \mathbf{2}$ différentes correspondant aux deux bijections $\mathbf{2} \simeq \mathbf{2}$ (où $\mathbf{2}$ est le type ayant deux éléments). Voevodsky a aussi remarqué que l'axiome d'univalence implique l'extensionnalité des fonctions, c'est-à-dire que si $f(x) =_B g(x)$ pour tout $x : A$, alors $f =_{A\to B} g$, et qu'il rend possible la définition des quotients.

En 2011, la notion de *type inductif supérieur* a commencé à émerger. Les types inductifs ordinaires sont des types définis en donnant des générateurs (les *constructeurs*) et un principe d'induction qui rend précise l'idée que le type est librement engendré par les constructeurs. Les types inductifs supérieurs sont une généralisation des types inductifs ordinaires où on peut non seulement donner des constructeurs de points, mais aussi des constructeurs de chemins. Par exemple, le cercle a un constructeur de points appelé base et un constructeur de chemins loop qui est un chemin de base vers base. En combinaison avec l'axiome d'univalence, une fibration peut être définie par induction sur l'espace de base, ce qui est un moyen très puissant pour définir des fibrations. Par exemple, afin de définir une fibration sur le cercle il suffit de donner la fibre au dessus de base et l'action de loop sur cette fibre (cette action devant être une équivalence).

Un des inconvénients de la théorie des types homotopiques est qu'en ajoutant l'axiome d'univalence ou les types inductifs supérieurs on perd la propriété de constructivité qui, comme on l'a déjà mentionné, est une propriété essentielle de la théorie des types. Cependant, contrairement au cas de l'axiome du choix ou du tiers exclu, il est généralement admis que l'axiome d'univalence et les types inductifs supérieurs sont constructifs d'une façon ou d'une autre, et diverses personnes cherchent à donner une description alternative de la théorie des types homotopiques dans laquelle on peut calculer avec l'axiome

d'univalence et les types inductifs supérieurs, voir en particulier [Coh+15]. Une conjecture associée est la conjecture de canonicité homotopique de Voevodsky : pour tout terme clos $n : \mathbb{N}$ construit en utilisant l'axiome d'univalence, il existe un terme clos $k : \mathbb{N}$ construit *sans* utiliser l'axiome d'univalence et une démonstration de $k =_{\mathbb{N}} n$.

Constructivité de $\pi_4(\mathbb{S}^3)$ Le premier résultat majeur de cette thèse est le corollaire 3.4.5, qui dit qu'il existe un entier naturel n tel que $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$. Cet énoncé est assez étrange parce que c'est un énoncé de la forme « il existe un entier naturel n satisfaisant une certaine propriété » donc selon la conjecture de constructivité il devrait être possible d'extraire la valeur de n de la démonstration. Cependant, pour l'instant personne n'a réussi à le faire, principalement parce que la démonstration est relativement compliquée et que la constructivité de l'axiome d'univalence et des types inductifs supérieurs n'est pas encore bien comprise. Dans les chapitres 4, 5 et 6 on présente une démonstration du fait que cet entier est égal à 2, mais il s'agit d'une démonstration mathématique et non pas d'un calcul extrait de la définition de n donc cela n'aborde pas la conjecture de constructivité. En revanche cela montre que l'on peut définir et travailler avec la cohomologie et la suite exacte de Gysin en théorie des types homotopiques, ce qui est intéressant en soi.

Modèles de la théorie des types homotopiques On ne va pas beaucoup parler de la relation entre la théorie des types homotopiques (théorie de l'homotopie synthétique) et la théorie de l'homotopie classique (théorie de l'homotopie analytique) dans cette thèse, mis à part le fait que beaucoup de définitions et de démonstrations sont assez similaires à leurs homologues classiques. Une construction d'un modèle de la théorie des types homotopiques (sans types inductifs supérieurs) en théorie de l'homotopie classique est présentée dans [KLV12] et une démonstration qu'il modèle également les types inductifs supérieurs est en préparation dans [LS]. Comme on l'a mentionné plus haut, une des conséquences de travailler de façon synthétique est que tout le travail effectué dans cette thèse est également valide dans n'importe quel autre modèle de la théorie des types homotopiques, pas seulement le modèle classique. Michael Shulman a donné dans [Shu15] divers autres modèles de la théorie des types homotopiques et il est généralement admis que tout ∞-topos au sens de Lurie (voir [Lur09]) devrait donner un modèle de la théorie des types homotopiques.

Un autre modèle très important est le modèle de Thierry Coquand et coauteurs décrit dans [BCH14], qui est un modèle constructif de la théorie des types homotopiques dans les ensemble cubiques. En théorie, ce modèle devrait nous permettre de calculer le nombre n du chapitre 3, mais cela n'a pas encore été fait. Ce modèle suggère également une version différente de la théorie des types homotopiques, appelée *théorie des types cubiques* (voir [Coh+15]), mais dans cette thèse on restera avec la théorie des types décrite dans [UF13]. On utilisera néanmoins divers carrés et cubes à chaque fois que cela sera jugé utile.

Résumé substantiel (français)

1 La théorie des types homotopiques

On commence par introduire tous les types de base et le constructeurs de types de la théorie des types homotopiques. Le premier constructeur de types est le types des *fonctions*. Étant donnés deux types A et B, le type des fonctions de A vers B est noté $A \to B$. On peut appliquer une fonction $f : A \to B$ à un élément $a : A$, on obtient alors un élément $f(a) : B$, et on définit une fonction $f : A \to B$ en donnant la valeur de $f(x) : B$, pour x une variable de type A. Le type des fonctions se généralise au type des *fonctions dépendantes*, noté $(x : A) \to B(x)$, lorsque B est un type dépendant sur A.

On introduit ensuite le type des *paires* $A \times B$, pour deux types A et B. Étant donné $a : A$ et $b : B$ on peut considérer l'élément (a, b) de $A \times B$, et étant donné un élément $u : A \times B$ on peut considérer $\mathsf{fst}(u) : A$ et $\mathsf{snd}(u) : B$. Comme pour le type des fonctions, le type des paires se généralise au type des *paires dépendantes* $\sum_{x:A} B(x)$ où le type du deuxième composant dépend du premier composant.

On introduit ensuite les *types inductifs*. L'idée est qu'un type inductif est librement engendré par un certain nombre de *constructeurs*, ce qui est exprimé par un *principe d'induction*. Par exemple le type vide, le type singleton, l'union disjointe de deux types, le type des entiers naturels et le type des entiers relatifs peuvent être définis comme des types inductifs.

Le quatrième constructeur de types que l'on introduit est le *type identité* ou type des chemins entre deux points. Étant donné un type A et deux éléments $u, v : A$, le type $u =_A v$ représente le type des chemins continus du u vers v, et la principale règle de typage gouvernant les types identité est la règle J. Cette règle permet de munir chaque type d'une structure d'∞-groupoïde faible, par exemple on peut inverser et composer les chemins, cette composition est associative, et ainsi de suite. Une définition plus précise de la notion d'∞-groupoïde est décrite dans l'appendice A.

Une fonction $f : A \to B$ est appelée une *équivalence* s'il existe une fonction $g : B \to A$ telle que les deux composées $g \circ f$ et $f \circ g$ soient homotopes à la fonction identité. Une petite modification de cette définition nous permet de définir le type $A \simeq B$ des équivalences entre A et B ainsi qu'une application $(A =_{\mathsf{Type}} B) \to (A \simeq B)$. L'axiome d'univalence est l'énoncé stipulant que cette application est elle-même une équivalence.

La notion de chemin introduite précédemment est homogène, dans le sens où on ne peut parler d'un chemin entre deux éléments u et v qu'à condition que u et v aient le

même type. On introduit alors la notion de *chemin dépendant*. Étant donné un type dépendant $B : A \to \mathsf{Type}$, un chemin $p : x =_A y$ dans A et deux éléments $u : B(x)$ et $v : B(y)$, le type $u =_p^B v$ représente le type des chemins de u vers v au dessus de p. Cette notion est utilisée en particulier pour définir le type de l'application d'une fonction dépendante à un chemin. Lorsque $B(x)$ est un type identité, il est naturel d'introduire le type des remplissages d'un carré, étant donné quatre chemins correspondant aux quatre côtés. Finalement on a la propriété d'extensionalité des fonctions qui permet de construire une égalité entre deux fonctions f et g étant donné une égalité entre $f(x)$ et $g(x)$ pour tout x.

Les derniers types que l'on introduit sont les *types inductifs supérieurs*. L'idée est similaire aux types inductifs, mis à part la possibilité d'avoir des constructeurs de chemins. Ceci permet de définir de nombreux espace de la théorie de l'homotopie, comme par exemple le cercle $\mathbb{S}^1$ qui est engendré par les deux constructeurs

$$\begin{aligned} &\mathsf{base} : \mathbb{S}^1, \\ &\mathsf{loop} : \mathsf{base} =_{\mathbb{S}^1} \mathsf{base} \end{aligned}$$

et le pushout d'un diagramme de la forme

$$A \xleftarrow{f} C \xrightarrow{g} B$$

qui est le type $A \sqcup^C B$ engendré par les trois constructeurs

$$\begin{aligned} \mathsf{inl} &: A \to A \sqcup^C B, \\ \mathsf{inr} &: B \to A \sqcup^C B, \\ \mathsf{push} &: (c : C) \to \mathsf{inl}(f(c)) =_{A \sqcup^C B} \mathsf{inr}(g(c)). \end{aligned}$$

Les pushouts permettent de définir de nombreux autres types comme la suspension d'un type, les sphères, le join de deux types, la somme pointée et le produit smash. Comme pour les types inductifs ordinaires, les types inductifs supérieurs ont un principe d'induction, qui utilise la notion de chemin dépendant vue ci-dessus.

Étant donné un diagramme 3×3 où l'on peut prendre le pushout de chacune des lignes et de chacune des colonnes (voir le diagramme 1.8.1 de la page 34), le pushout du pushout des lignes est équivalent au pushout du pushout des colonnes. Ce résultat est appelé le *lemme* 3×3 et est utilisé entre autres pour montrer que le join de $\mathbb{S}^n$ et de $\mathbb{S}^m$ est équivalent à $\mathbb{S}^{n+m+1}$.

Étant donné un type inductif T, on peut définir un type dépendant $P : T \to \mathsf{Type}$ en utilisant le principe d'induction de T et l'axiome d'univalence pour les constructeurs de chemin de T. Le *lemme d'aplatissement* est un résultat décrivant l'espace total d'une telle fibration.

On introduit finalement le concept de *type n-tronqué*, c'est à dire intuitivement un type qui n'a pas d'information homotopique en dimension supérieure à n. Un type est (-2)-tronqué (ou contractile) lorsque qu'il possède un point égal à tous les autres points, et un type est $(n+1)$-tronqué lorsque tous ses types identité sont n-tronqués. On

a également une opération de troncation qui transforme un type A en un type n-tronqué $\|A\|_n$ d'une façon universelle, c'est-à-dire qu'on a une application $|-| : A \to \|A\|_n$ et afin de définir une application $f : \|A\|_n \to B$ pour un type B qui est lui-même n-tronqué, il suffit de définir f sur les éléments de la forme $|a|$.

2 Résultats préliminaires sur les groupes d'homotopie des sphères

Étant donné un type A pointé par $\star_A : A$ et un entier $n \geq 1$, on définit l'*espace des lacets itérés* de A par

$$\begin{aligned} \Omega^0 A &:= A, \\ \Omega^{n+1} A &:= \Omega(\Omega^n A), \end{aligned}$$

où $\Omega A := (\star_A = \star_A)$ est l'*espace des lacets* de A. On définit ensuite le *nième groupe d'homotopie* de A par

$$\pi_n(A) := \|\Omega^n A\|_0 .$$

Afin de calculer les groupes d'homotopie du cercle, on définit la fibration

$$\begin{aligned} U : \mathbb{S}^1 &\to \mathsf{Type}, \\ U(\mathsf{base}) &:= \mathbb{Z}, \\ \mathsf{ap}_U(\mathsf{loop}) &:= \mathsf{ua}(\mathsf{succ}_{\mathbb{Z}}). \end{aligned}$$

où on utilise l'axiome d'univalence ua et le fait que $\mathsf{succ}_{\mathbb{Z}}$ est une équivalence de $\mathbb{Z}$. Le lemme d'aplatissement implique que l'espace total de U est contractile, ce qui implique que $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$ et que $\pi_n(\mathbb{S}^1)$ est trivial pour tout $n > 1$.

On définit ensuite la notion de *type n-connexe* comme étant un type dont la n-troncation est contractile, et une fonction est dite *n-connexe* si toutes ses fibres sont n-connexes. Les fonctions n-connexes vérifient le principe d'induction suivant. Une fonction $f : A \to B$ est n-connexe si et seulement si pour toute famille $P : B \to \mathsf{Type}$ de types n-connexes et toute fonction $d : (a : A) \to P(f(a))$ il existe une section $s : (b : B) \to P(b)$ de P telle que pour tout $a : A$ on ait $s(f(a)) = d(a)$. Ceci permet de montrer entre autres que la fonction $|-| : A \to \|A\|_n$ est n-connexe, que la composée de deux fonctions n-connexes est n-connexe et qu'un pushout d'une fonction n-connexe est n-connexe. On en déduit ensuite que la sphère $\mathbb{S}^n$ est $(n-1)$-connexe et qu'en particulier tous les groupes $\pi_k(\mathbb{S}^n)$ avec $k < n$ sont triviaux.

On définit enfin la fibration de Hopf par

$$\begin{aligned} \mathsf{Hopf} : \mathbb{S}^2 &\to \mathsf{Type}, \\ \mathsf{Hopf}(\mathsf{north}) &:= \mathbb{S}^1, \\ \mathsf{Hopf}(\mathsf{south}) &:= \mathbb{S}^1, \\ \mathsf{ap}_{\mathsf{Hopf}}(\mathsf{merid}(x)) &:= \mathsf{ua}(\mu(-, x)), \end{aligned}$$

où $\mu : \mathbb{S}^1 \times \mathbb{S}^1 \to \mathbb{S}^1$ est défini par double induction sur le cercle. Les deux fonctions $\mu(-, \mathsf{base})$ et $\mu(\mathsf{base}, -)$ sont égales à la fonction identité, ce qui montre que $\mu(-, x)$ est une équivalence pour tout $x : \mathbb{S}^1$. Le lemme d'aplatissement ainsi que le fait que $\mu(x, -)$ est une équivalence pour tout $x : \mathbb{S}^1$ nous permet alors de montrer que l'espace total de la fibration de Hopf est équivalent au join $\mathbb{S}^1 * \mathbb{S}^1$ qui est équivalent à $\mathbb{S}^3$. On obtient alors $\pi_2(\mathbb{S}^2) \simeq \mathbb{Z}$ et $\pi_k(\mathbb{S}^2) \simeq \pi_k(\mathbb{S}^3)$ pour tout $k \geq 3$.

3 La construction de James

Étant donné un type pointé connexe A, la *construction de James* donne une suite d'approximation du type $\Omega\Sigma A$. Plus précisément, si A est k-connexe, on obtient une suite de types et de fonctions

$$J_0 A \xrightarrow{i_0} J_1 A \xrightarrow{i_1} J_2 A \xrightarrow{i_2} J_3 A \xrightarrow{i_3} \dots,$$

où pour tout n l'application i_n est $(n(k+1)+(k-1))$-connexe, et telle que la colimite $J_\infty A$ du diagramme est équivalente à $\Omega\Sigma A$. Les types $J_n A$ sont définis par induction sur n par des pushouts. On définit ensuite un type inductif supérieur JA et on démontre que JA est équivalent à la colimite de $(J_n A)_{n:\mathbb{N}}$ ainsi qu'à $\Omega\Sigma A$. Une première conséquence de la construction de James est le théorème de suspension de Freudenthal, qui dit que l'application $A \to \Omega\Sigma A$ est $2k$-connexe lorsque A est k-connexe. En particulier, cela implique que $\pi_n(\mathbb{S}^n) \simeq \pi_2(\mathbb{S}^2) \simeq \mathbb{Z}$ pour tout $n \geq 2$ et que $\pi_{n+1}(\mathbb{S}^n) \simeq \pi_4(\mathbb{S}^3)$ pour tout $n \geq 3$.

On définit ensuite, pour tout $n, m : \mathbb{N}$, une application $W_{n,m} : \mathbb{S}^{n+m-1} \to \mathbb{S}^n \vee \mathbb{S}^m$ qui donne une équivalence

$$\mathbb{S}^n \times \mathbb{S}^m \simeq \mathbf{1} \sqcup^{\mathbb{S}^{n+m-1}} (\mathbb{S}^n \vee \mathbb{S}^m).$$

Cette application induit (par composition) une application $[-,-] : \pi_n(X) \times \pi_m(X) \to \pi_{n+m-1}(X)$ pour tout type X. En combinaison avec la construction de James et le théorème de Blakers–Massey, on démontre alors que $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$, où n est l'image de $[i_2, i_2]$ par l'équivalence $\pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$ et i_2 est le générateur de $\pi_2(\mathbb{S}^2)$.

4 Produits smash de sphères

Le *produit smash* $A \wedge B$ de deux types pointés A et B est le type inductif supérieur défini par les constructeurs

$$\begin{aligned}
\star_{A\wedge B} &: A \wedge B,\\
\mathsf{proj} &: A \to B \to A \wedge B,\\
\mathsf{proj_r} &: (a : A) \to \mathsf{proj}(a, \star_B) = \star_{A\wedge B},\\
\mathsf{proj_l} &: (b : B) \to \mathsf{proj}(\star_A, b) = \star_{A\wedge B},\\
\mathsf{proj_{rl}} &: \mathsf{proj_r}(\star_A) = \mathsf{proj_l}(\star_B).
\end{aligned}$$

Intuitivement, il s'agit du produit $A \times B$ où l'on a contracté les deux axes A et B sur un point. Le produit smash est un produit monoïdal symétrique sur les types pointés, en particulier il est fonctoriel, commutatif, associatif, unitaire et satisfait des hypothèses de naturalité et de cohérence en dimension 1. Le produit smash de sphères satisfait l'équivalence $\mathbb{S}^n \wedge \mathbb{S}^m \simeq \mathbb{S}^{n+m}$. On démontre que cette famille d'équivalences est compatible avec l'associativité du produit smash, dans le sens où les deux équivalences $\mathbb{S}^n \wedge \mathbb{S}^m \wedge \mathbb{S}^n \to \mathbb{S}^{n+m+k}$ provenant des deux parenthésages du codomaine sont égales. En ce qui concerne la commutativité, on obtient un résultat similaire à celui pour l'associativité, mis à part qu'il y a un signe lorsque les deux sphères sont de dimension impaire.

5 La cohomologie

On définit ensuite les groupes de cohomologie (à coefficients entiers) d'un type. La première étape est la définition des espaces d'Eilenberg–MacLane $K(\mathbb{Z}, n)$ (notés plus simplement K_n). Pour $n = 0$, on définit $K_0 := \mathbb{Z}$ et pour $n > 0$ on définit $K_n := \|\mathbb{S}^n\|_n$. La principale propriété de ces espaces est le fait qu'il existe une équivalence $K_n \simeq \Omega K_{n+1}$. La construction de cette équivalence utilise le résultat sur $\pi_1(\mathbb{S}^1)$ pour $n = 0$, la fibration de Hopf pour $n = 1$ et le théorème de suspension de Freudenthal pour $n \geq 2$. Les groupes de cohomologie d'un type X sont alors définis par

$$H^n(X) := \|X \to K_n\|_0$$

et l'équivalence entre K_n et ΩK_{n+1} donne une structure de groupe abélien sur $H^n(X)$.

On définit ensuite le produit cup

$$\smile : H^n(X) \times H^m(X) \to H^{n+m}(X)$$

en utilisant les équivalences $\mathbb{S}^n \wedge \mathbb{S}^m \simeq \mathbb{S}^{n+m}$, et la compatibilité de cette famille d'équivalences avec l'associativité et la commutativité du produit smash permettent de montrer que le produit cup est associatif, commutatif gradué et distributif.

La suite exacte longue de Mayer–Vietoris se déduit assez facilement de la définition de la cohomologie. Cela permet de montrer que $H^k(\mathbb{S}^n)$ est équivalent à $\mathbb{Z}$ pour $k = 0$ et $k = n$, et trivial sinon. On calcule ensuite les groupes de cohomologie de $\mathbb{S}^n \times \mathbb{S}^k$. La suite exacte longue de Mayer–Vietoris montre qu'ils sont engendrés (additivement) par un élément $\mathbf{1}$ de degré 0, un élément $\mathbf{x}$ de degré n, un élément $\mathbf{y}$ de degré k et un élément $\mathbf{z}$ de degré $n + k$. On montre de plus que le produit cup $\mathbf{x} \smile \mathbf{y}$ est égal à $\mathbf{z}$.

Finalement on définit l'invariant de Hopf d'une application $f : \mathbb{S}^{2n-1} \to \mathbb{S}^n$ comme étant l'entier $H(f)$ vérifiant

$$\alpha \smile \alpha = H(f)\beta,$$

où α et β sont les générateurs de la cohomologie en dimensions n et $2n$ du type $C_f := \mathbf{1} \sqcup^{\mathbb{S}^{2n-1}} \mathbb{S}^n$. L'invariant de Hopf est un homomorphisme de groupes $H : \pi_{2n-1}(\mathbb{S}^n) \to \mathbb{Z}$ et on montre que pour tout n, l'invariant de Hopf de $[i_{2n}, i_{2n}] : \pi_{4n-1}(\mathbb{S}^{2n})$ est égal à 2. On en conclut que $\pi_{4n-1}(\mathbb{S}^{2n})$ est infini, et que $\pi_4(\mathbb{S}^3)$ est équivalent soit à $\mathbb{Z}/2\mathbb{Z}$ s'il

existe une application $\mathbb{S}^3 \to \mathbb{S}^2$ d'invariant de Hopf égal à 1, ou est trivial s'il n'existe pas de telle application.

6 La suite de Gysin

Étant donné une fibration de sphères $\mathbb{S}^{n-1}$ sur un type B, on construit un élément $e : H^n(B)$ ainsi que la *suite exacte longue de Gysin* reliant le produit cup $- \smile e$ avec la cohomologie de l'espace total de la fibration. La principale propriété permettant la construction de cette suite exacte longue est le fait que pour $n, m : \mathbb{N}$, $p : K_n$ et $y : K_m$, on a

$$\mathsf{ap}_{\lambda x.x \smile y}(\sigma_n(p)) = \sigma_{n+m}(p \smile y),$$

où σ_n dénote l'équivalence $K_n \simeq \Omega K_{n+1}$. On en déduit que pour tout $i, n : \mathbb{N}$, l'application

$$\begin{aligned} f &: K_i \to (\mathbb{S}^n \to K_{i+n}) \\ f(x) &:= (\lambda y.x \smile |y|) \end{aligned}$$

est une équivalence, ce qui est démontré par récurrence sur i.

On définit ensuite $\mathbb{C}P^2$ comme étant le pushout $\mathbf{1} \sqcup^{\mathbb{S}^3} \mathbb{S}^2$ pour l'application $\eta : \mathbb{S}^3 \to \mathbb{S}^2$ provenant de la fibration de Hopf. D'une façon similaire à la construction de la fibration de Hopf, on construit une fibration sur $\mathbb{C}P^2$ de fibre $\mathbb{S}^1$ et dont l'espace total est équivalent à $\mathbb{S}^5$. En appliquant la suite de Gysin à cette fibration, on en déduit que l'invariant de Hopf de η est égal à ± 1, ce qui montre que

$$\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}.$$

A Une définition des ∞-groupoïdes faibles par la théorie des types

Dans cet appendice on donne une définition de la notion d'∞-groupoïde faible, inspirée de la théorie des types. L'idée est de définir une théorie des types minimaliste dans laquelles les termes que l'on peut définir sont exactement ceux qui existent dans tout ∞-groupoïde. On explique ensuite comment cette théorie des types donne une définition des ∞-groupoïdes faibles et on montre que tout type en théorie des types homotopiques est un ∞-groupoïde faible.

B Le cardinal de $\pi_4(\mathbb{S}^3)$

Dans cet appendice on résume la définition de l'entier n défini à la fin du chapitre 3, afin de faciliter une implémentation future dans un assistant de preuves muni d'une interprétation calculatoire de l'axiome d'univalence et des types inductifs supérieurs.

Conclusion (français)

On a vu dans cette thèse que la théorie des types homotopiques est suffisamment puissante pour démontrer que $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$. Même si du point de vue de la théorie de l'homotopie classique c'est un résultat bien connu, ce n'était pas évident que l'axiome d'univalence et les types inductifs supérieurs suffiraient à le démontrer, ni même qu'une démonstration constructive et purement homotopique existe. De plus, en prenant en compte le fait que seulement cinq années se sont écoulées entre la définition du cercle en théorie des types homotopiques et le calcul de $\pi_4(\mathbb{S}^3)$, le progrès a été plutôt rapide et on peut espérer que d'ici quelques années la théorie des types homotopiques aura atteint un niveau comparable à celui de la théorie de l'homotopique classique et va aider à obtenir des résultats complètements nouveaux.

Comparaison avec les démonstrations classiques Comme on l'a déjà mentionné, la principale différence entre la théorie de l'homotopie classique et la théorie des types homotopiques est qu'en théorie des types homotopique tout est invariant par homotopie. J'ai utilisé le livre [Hat02], qui présente la topologie algébrique classique, assez régulièrement pendant cette recherche, mais j'ai souvent dû trouver des définitions, des démonstrations et des énoncés complètement différents afin de pouvoir les reproduire en théorie des types homotopiques.

Dans la construction du revêtement universel du cercle et de la fibration de Hopf, la différence la plus évidente est qu'au lieu de définir une fonction de l'espace total vers la base, on définit directement les fibres et comment transporter le long des fibres, et déterminer l'espace total est la partie non triviale. Pour le revêtement universel du cercle c'est assez transparent mais pour la fibration de Hopf il n'était pas clair a priori pour moi que la définir avec la multiplication de $\mathbb{S}^1$ donnerait effectivement la fibration de Hopf.

Démontrer que le produit smash est associatif n'est facile ni en théorie des types homotopiques ni en théorie de l'homotopie classique mais pour des raisons très différentes. En théorie des types homotopiques, le problème vient du fait qu'on a beaucoup de chemins et de chemins de dimension supérieure à gérer et que cela devient vite compliqué de s'occuper de toutes les cohérences entre eux. En topologie générale, en revanche, il y a une bijection canonique entre les deux espaces mais le problème est qu'elle peut ne pas être continue sauf si l'on suppose que les deux espaces se comportent suffisamment bien. La différence est qu'en topologie général on identifie littéralement divers points

entre eux, ce qui fait que la bijection est facile à définir mais peut ne pas respecter la topologie, alors qu'en théorie des types homotopiques on rajoute de nouveaux chemins au lieu d'identifier des points.

Pour la cohomologie, on a déjà mentionné le besoin d'utiliser les espaces d'Eilenberg–MacLane au lieu de la définition classique par les cochaînes singulières, étant donné que l'ensemble des cochaînes singulières n'est pas invariant par homotopie. La notion de troncation donne une définition très agréable des espaces d'Eilenberg–MacLane $K(\mathbb{Z}, n)$ et une définition très agréable du produit cup. Notons que quand on travaille de façon constructive certains phénomènes inattendus peuvent se produire en cohomologie. En effet, sans l'axiome du choix il n'est pas possible de démontrer l'axiome d'additivité de la cohomologie. On ne peut même pas démontrer que le groupe de cohomologie $H^1(\mathbb{N}, \mathbb{Z})$ est trivial, comme est expliqué dans [Shu13]. Il est agréable de voir que ce genre de problème ne se pose pas dans le calcul de $\pi_4(\mathbb{S}^3)$.

Finalement, afin de calculer la cohomologie de $\mathbb{C}P^2$ je n'ai pas réussi à adapter la démonstration géométrique présentée par exemple dans [Hat02, theorem 3.19] and j'ai dû utiliser la suite exacte de Gysin qui est en général considérée comme étant un résultat plus avancé en théorie de l'homotopie classique. De plus, la construction de la suite exacte de Gysin présentée dans [Hat02, section 4.D] est basée sur le théorème de Leray–Hirsch dont la démonstration utilise une induction sur les cellules d'un CW complexe, ce qui est impossible à faire en théorie des types homotopiques. La démonstration présentée ici est nouvelle et basée sur la proposition 6.1.1 qui relie directement le produit cup en dimensions n et $n + 1$.

Théorie des types homotopiques cubique J'ai d'abord essayé d'écrire cette thèse en utilisant des idées "cubiques" autant que possible, comme nous l'avions fait par exemple avec Dan Licata dans [LB15] and comme a été fait dans [Cav15], mais cela s'avéra être une mauvaise idée, à ma grande déception. Même si beaucoup de carrés et de cubes apparaissent naturellement en théorie des types homotopiques, comme par exemple le carré de naturalité des homotopies, il y a aussi d'autres formes qui peuvent apparaître, et traiter les carrés différemment n'est peut-être pas la meilleure idée. Par exemple, essayer d'écrire le diagramme 3.2.5 de la page 73 comme une composition de Kan de carrés et de cubes n'est pas très naturel ni utile. Il semble beaucoup plus naturel d'utiliser dans ce cas une notion générale de composition de diagrammes (comme décrite à la fin de la page 28). Les idées cubiques ont quand même des avantages, en particulier [Coh+15] utilise des ensembles cubiques pour donner une interprétation calculatoire de la théorie des types homotopiques, et dans [LB15] et [Cav15] les cubes sont utilisés principalement pour simplifier les formalisations en Agda. Je pense, cependant, que pour la théorie de l'homotopie synthétique informelle, comme dans cette thèse, les idées cubiques ne sont pas spécialement utiles en général.

Travaux futurs Un des principaux objectifs futurs est la formalisation des résultats présentés ici dans un assistant de preuves. Il risque d'y avoir des problèmes dans la formalisation de la construction de James, et encore plus dans la structure monoïdale du

produit smash, où il y a beaucoup de manipulations de chemins et de chemins supérieurs à faire, mais cela devrait être possible.

Quelques résultats de cette thèse ont été énoncés et démontrés sous une forme plutôt restreinte étant donné que mon objectif principal était d'arriver au résultat $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$. Par exemple on devrait pouvoir calculer l'anneau de cohomologie de tous les $J_n(\mathbb{S}^k)$, y compris $J_\infty(\mathbb{S}^k) \simeq \Omega\mathbb{S}^{k+1}$, et de tous les $\mathbb{C}P^n$, y compris $\mathbb{C}P^\infty \simeq K(\mathbb{Z}, 2)$. De plus, on a défini seulement la cohomologie à coefficients entiers mais il devrait être possible de définir la cohomologie à coefficients dans un groupe, un anneau ou un spectre arbitraire, et d'étendre la plupart des résultats présentés ici.

Finalement, le projet à long terme est évidemment de continuer le développement de la théorie de l'homotopie synthétique dans la théorie des types homotopiques. Il y a beaucoup de concepts qui n'ont pas encore été beaucoup étudiés mais qui semblent accessibles, comme la K-théorie, les opérations de Steenrod, les suites spectrales, les crochets de Toda et beaucoup d'autres. Je vais certainement continuer de poursuivre cette ligne de recherche et j'espère que cette thèse va inspirer d'autres personnes à participer à l'exploration de la théorie de l'homotopie synthétique étant donné tout ce qui n'attend qu'à être découvert.