Naive cubical type theory

Bruno Bentzen[1]

[1]Carnegie Mellon University,
Pittsburgh, PA, USA
bbentzen@andrew.cmu.edu

Abstract

This paper proposes a way of doing type theory informally, assuming a cubical style of reasoning. It can thus be viewed as a first step toward a cubical alternative to the program of informalization of type theory carried out in the homotopy type theory book for dependent type theory augmented with axioms for univalence and higher inductive types. We adopt a cartesian cubical type theory proposed by Angiuli, Brunerie, Coquand, Favonia, Harper and Licata as the implicit foundation, confining our presentation to elementary results such as function extensionality, the derivation of weak connections and path induction, the groupoid structure of types and the Eckmman–Hilton duality.

1 Introduction

Cubical type theory [7, 1, 2] is a flavor of higher-dimensional type theory that is more directly amenable to constructive interpretations. It is an alternative to homotopy type theory [12], which, in turn, is based on an extension of dependent type theory with axioms for univalence and higher inductive types. However, unlike the axiomatic approach of homotopy type theory, which is well known to block computation since new canonical terms are postulated without specifying how to compute with them, cubical type theory relies on cubical methods to properly specify the computational behavior of terms at higher dimensions. But, despite being a fully computational presentation of the subject, the appeal to the sophisticated machinery of cubical reasoning makes higher-dimensional type theory even more impenetrable to the uninitiated.

In the spirit of Halmos' seminal book [10], the naive type theory project introduced by Constable [8] is aimed at making type theory more accessible to mathematicians by introducing the subject in an informal but, in principle, formalizable way, that is, by proposing an intuitive presentation independent of any technical appeal to the rules of inference of the formalism. In this sense, "naive" is in contrast with "formal", meaning that naive type theory can be well understood as formal type theory approached from a naive perspective [10, 8].

Ideally, it should be possible to make naive type theory formally precise, so there should be fundamental rules, like formation, introduction, elimination and computation rules, that underlie naive type theory as well. The point here is that we can almost always forget about those technicalities when assuming the naive point of view, according to which a type theory is viewed as an intuitive semantics rather than a collection of axioms and rules of inference.

It is very natural to introduce a complex subject such as higher-dimensional type theory via a naive approach in the sense above. Indeed, the development of naive higher-dimensional type theory results in at least two major benefits:

(1) If higher-dimensional type theory is to be taken seriously as a foundation for mathematics or research program, then, it should be accessible with a minimum of logical formalism to nonexperts.

(2) Pen-and-paper proofs given in a homotopically-inspired informal language for mathematics could be more closely related to practices of working mathematicians such as the identification of isomorphic structures. Moreover, proof mechanization might require significantly less effort, as type theory is the basis of several proof-assistants.

For homotopy type theory, at least, this informalization was accomplished in a recent book [12], in which the theory is systematically developed from scratch with the use of language and notation that are similar to that of ordinary mathematics, but without making no precise reference to the axioms or rules of inference that establish the formal system. Put differently, the so-called homotopy type theory book [12] develops a naive type theory, a rigorous style of doing everyday mathematics informally assuming homotopy type theory as the underlying foundation, which is then used to give informal proofs of theorems from various areas of mathematics, such as logic, set theory, category theory and homotopy theory throughout the book.

This paper can be viewed as an initial effort to introduce readers to the naive way of doing cubical type theory, in the same way that the homotopy type theory book [12] develops and promotes naive homotopy type theory. The main goal here is to propose a naive presentation for cubical type theory that has a similar degree of rigor, while also highlighting the distinctive aspects of the cubical approach to higher-dimensional type theory by means of proofs of some elementary theorems. For that reason, the reader may notice that, in this paper, arguments by path induction, which is used in homotopy type theory as the elimination principle for the path type [12, §1.12], are almost completely ignored in favor of cubical arguments. This often results in diagrammatic proofs that are conceptually simpler, cubically speaking, but may require more ingenuity. Since cubical type theory comes in many different forms, depending on the structure one imposes on the cube category it is based on, we should emphasize that our naive presentation is founded on cartesian cubical type theory [1], a formal theory developed by Angiuli, Brunerie, Coquand, Favonia, Harper and Licata based on the cartesian cube category, that is, the free category with finite products on an interval object [3], a category of cubes which has faces, degeneracies, symmetries, diagonals, but no connections or reversals—unlike the De Morgan cube category and the cubical type theories based on this structure [7, 9].

The remainder of this paper is organized as follows: in the next section, a short introduction to cubical type theory from the naive point of view is given and its semantics is described informally. Then, Kan operations are discussed in some detail. Next, we derive weak connections and give informal proofs of the groupoid structure of types. Subsequently, we consider dependent paths and heterogeneous composition. Finally, proofs of path induction and the Eckmann–Hilton duality are presented, followed by a section containing final conclusions. This paper is intended to be self-contained, but we assume throughout some general familiarity with the concepts of homotopy type theory.

2 The cubical point of view

Naive cubical type theory is the idea that cubes are the basic shapes used to characterize the structure of higher-dimensional mathematical objects. It is, however, grounded on the same homotopical intuition [4, 13] which regards a type A as a space, a term $a : A$ as a point of the space A, a function $f : A \to B$ as a continuous map, a path $p : \mathsf{path}_A(a, b)$ as a path from point a to point b in the space A, a universe type $\mathcal{U}$ as a space, the points of which are spaces, and a type family $P : A \to \mathcal{U}$ as a fibration. Like homotopy type theory, spaces are understood purely from the point of view of homotopy theory, meaning that homotopy equivalent spaces are equal up to a path.

The distinctive cubical perspective of type theory starts with the consideration of an abstraction of the unit interval in the real line, that is, a space consisting of two points, $\mathsf{0}$ and $\mathsf{1}$, which we call the interval type and denote $\mathbb{I}$. Traditionally, a path in a topological space is a continuous function from the unit interval. This point-set topological description is generalized in cubical type theory in the sense that, as will be discussed in the remainder of this section, we represent functions from the interval as paths. In higher dimensions, it turns out that paths can be visualized as higher-dimensional cubes.

2.1 The type of paths

It is useful to have a type of paths. Certainly, the most obvious way of obtaining such a type is to use the function type itself. We shall often refer to the type of functions from the interval $\mathbb{I} \to A$ as the *line type*, and call its terms lines. It is convenient to work with paths with arbitrary endpoints using line types, but as soon as we start considering paths from a particular point to another point, we are in trouble. In order to better deal with paths with fixed endpoints, we need a slightly more sophisticated variant of the function type called the path type: given any type $A : \mathcal{U}$ and terms $a, b : A$, we can form the type of paths starting from a to b in A, which we call their path type, denoted $\mathsf{path}_A(a, b)$.

What can we do with paths? Like functions, given a path $p : \mathsf{path}_A(a, b)$ and an interval variable $i : \mathbb{I}$, we can apply the path to obtain a term of the type A depending on i, denoted $p(i)$ and called the value of p at position i. Moreover, applying a path to one of the interval endpoints should result in the corresponding endpoint of the path. This means that we have the following definitional equalities:

$$p(\mathsf{0}) \equiv a : A \quad \text{and} \quad p(\mathsf{1}) \equiv b : A \quad \text{for} \quad p : \mathsf{path}_A(a, b).$$

The way to construct paths is by abstraction: given a term $a : A$ which may depend on $i : \mathbb{I}$, we write $\lambda i.a$ to indicate a path from $a[\mathsf{0}/i]$ to $a[\mathsf{1}/i]$ in A, where, by convention, the scope of the binding $\lambda -.$ extends over the entire expression to its right unless otherwise noted by the use of parentheses.

It should come as no surprise that we require equalities similar to the ones in the lambda calculus. When a path abstraction term is applied to an interval point, we require a computation that plays the role of β-reduction:

$$(\lambda i.a)(j) \equiv a[j/i]$$

We also expect that "every path is a path abstraction", the uniqueness principle for the path type, so we also consider the η-expansion rule:

$$p \equiv \lambda i.(p(i)) \qquad \text{(when } i \text{ does not occur in } p\text{)}$$

This identifies any path p with the "path that applies p on the interval", thus endowing paths with an extensional aspect. The use of this equality is crucial to the derivation of path induction described in §6.1.

2.2 How we should think of paths

Doing cubical type theory is essentially a problem solving activity in which problems can be solved by cubical methods. Thus, it is essential to stress that higher paths can be represented cubically. The use of diagrams can be very helpful because, as it will be clear soon, many problems can be rigorously stated and solved using diagrammatic arguments.

How can we represent paths as cubes? We visualize a term $a : A$ as a point.

$$\cdot\ a$$

We then think of a term $p : \mathbb{I} \to A$ as a path starting from the point $p(\mathsf{0})$ to the point $p(\mathsf{1})$. Given an interval variable $i : \mathbb{I}$, such a path can be visualized as a line $p(i)$ in the "direction" i that goes from the initial to the terminal point of the path, as shown in the following diagram:

$$p(\mathsf{0}) \xrightarrow{\quad p(i) \quad} p(\mathsf{1}) \qquad\qquad \xrightarrow[i]{}$$

Extending the interpretation to higher dimensions, it is natural to think of a function $h : \mathbb{I} \to \mathbb{I} \to A$ as a homotopy of paths. Such a homotopy consists of two simultaneous continuous deformations and it can be visualized as a square having four paths at each edges, as shown in the following diagram, where the lines $h(0)$ and $h(1)$ are the left and right faces in the direction i, and $\lambda j.h(j,0)$ and $\lambda j.h(j,1)$ the bottom and top faces in the j-direction, respectively:

$$
\begin{tikzcd}
h(0,1) \arrow[rr, "\lambda j.h(j,1)"] & & h(1,1) \\
& h & \\
h(0,0) \arrow[uu, "h(0)"] \arrow[rr, "\lambda j.h(j,0)"'] & & h(1,0) \arrow[uu, "h(1)"']
\end{tikzcd}
$$

In the third dimension, we consider homotopies between homotopies, which, as expected can be pictured as cubes. It is hard enough to visualize paths at even higher dimensions, but, most certainly, the reader can already guess the general pattern here: at dimension $n+1$, we have $n+1$-dimensional paths, which can be visualized as hypercubes with $2(n+1)$ faces formed by n-dimensional cubes. Finally, it is worth noting that this cubical structure applies for types $A : \mathcal{U}$ as well, since types are just terms of a type of universes.

2.3 How can we use paths?

Before proceeding any further it is worthwhile to point out that this functional presentation of paths is a significant departure from the traditional approach taken in homotopy type theory [12], in which the path type is considered an inductive type generated by a reflexivity constructor. The following lemmas will serve to show some fundamental differences of the cubical approach.

We start with the fact that the reflexivity path is definable by considering constant functions from the interval, thus providing us a way to trivially regard terms as degenerate paths.

Lemma 2.3.1. *For every type A and every $a : A$, there exists a path*

$$\mathsf{path}_A(a,a)$$

called the reflexivity path of a and denoted refl_a.

Proof. Suppose that $i : \mathbb{I}$ is a fresh interval variable. By assumption, a does not depend on i, meaning that $\lambda i.a$ gives us a path starting from a to a in A. $\square$

Recall that, in the homotopy interpretation we view functions as continuous maps, so it is natural to expect that functions preserve paths, as shown in the homotopy type theory book [12, §2.2]. Here we state this property with the following lemma for non-dependent functions:

Lemma 2.3.2 (Function application [5, 7])**.** *Given a function $f : A \to B$ and terms $a, b : A$, we have an operation*

$$\mathsf{ap}_f : (\mathsf{path}_A(a,b)) \to (\mathsf{path}_B(f(a), f(b)))$$

such that $\mathsf{ap}_f(\mathsf{refl}_a) \equiv \mathsf{refl}_{f(a)}$.

Proof. Given $p : \mathsf{path}_A(a, b)$ and an interval variable $i : \mathbb{I}$, we have $f(p(i)) : B$, for we can apply $f : A \to B$ to $p(i) : A$. By abstraction, we have a path

$$\mathsf{ap}_f(p) :\equiv \lambda i.f(p(i)).$$

from $f(a)$ to $f(b)$ in B, since p is a path from a to b. Moreover, we have

$$\begin{aligned}\mathsf{ap}_f(\mathsf{refl}_a) &\equiv \lambda i.f(\mathsf{refl}_a(i))\\ &\equiv \lambda i.f(a)\\ &\equiv \mathsf{refl}_{f(a)}\end{aligned}$$

as requested. □

We note that ap_f behaves strictly functorially in the sense that the following equalities hold definitionally [5, 7],

$$\begin{aligned}\mathsf{ap}_{id_A}(p) &\equiv p\\ \mathsf{ap}_{f\circ g}(p) &\equiv \mathsf{ap}_f(\mathsf{ap}_g(p))\\ \mathsf{ap}_{\lambda\_.a}(p) &\equiv \mathsf{refl}_a.\end{aligned}$$

In homotopy type theory [12], in which the path type is defined as an inductive family, those equalities only hold up to homotopy.

Another crucial difference is that, with this notion of path, we are able to prove that two pointwise equal functions are equal up to a path. This is known as function extensionality, a property that cannot be obtained without the use of axioms in homotopy type theory [12, §2.9,§4.9].

Lemma 2.3.3 (Function extensionality [7])**.** *Suppose that $f, g : \prod_{(x:A)} B(x)$ are functions. There is an operation*

$$\mathsf{funext}_{f,g} : \prod_{(x:A)} \mathsf{path}_{B(x)}(f(x), g(x)) \to \mathsf{path}_{\prod_{(x:A)} B(x)}(f, g)$$

Proof. If we are given $H : \prod_{(x:A)} \mathsf{path}_{B(x)}(f(x), g(x))$ and an arbitrary $x : A$, we have a path $H(x)$ from $f(x)$ to $g(x)$ in $B(x)$. For $i : \mathbb{I}$, $H(x)$ at i gives a path from $\lambda x.f(x)$ to $\lambda x.g(x)$ in $A \to B$,

$$\lambda i.\lambda x.(H(x)(i)) : \prod_{(x:A)} B(x).$$

By η-conversion, it can be seen that this path goes from f to g. □

3 There are enough paths

So far we have only discussed general properties of paths. However, just as when working with spaces in terms of simplicial sets one has to consider Kan fibrations to provide enough structure to model spaces, here, in our alternative presentation of spaces in terms of cubical sets, we have to impose certain requirements to ensure that we have enough paths as well.

In order to properly model spaces in a cubical setting, it is helpful to consider two Kan operations, which we call *transport* and *composition* [1, 2]. They tell us what we can do with higher paths, and, to ensure that all terms compute properly, every type must come with its own specific operations, which determine how we should understand a Kan operation on a type in terms of reductions to their constructors or eliminators [1]. The general aspects of transport and composition are discussed in the remainder of this section.

3.1 Transportation along paths

Transport is a cubical generalization of the transport lemma [12, Lem 2.3.1]. Recall that, due to univalence [14], a principle that characterizes the path space of the universe type $\mathcal{U}$, a path between types is (equivalent to) a homotopy equivalence between spaces [12]. In the cubical setting, transport states that, given any path between types $A : \mathbb{I} \to \mathcal{U}$ and any term $a : A(i)$, we have a term of the type $A(j)$, called the transport of a from i to j over A, and denoted by $a_A^{i \rightsquigarrow j} : A(j)$. We also require that static transportations have no effect, which means that, when i is j, we have $a_A^{i \rightsquigarrow i} \equiv a$.

In general, transporting over a constant path is not strictly the same as doing nothing, meaning that $a_{\mathsf{refl}_A}^{i \rightsquigarrow j} \not\equiv a$. This is in contrast with the transport operation of homotopy type theory [12], in which transportation along reflexivity is taken to be the identity function. The following lemma shows in what sense a generalization of this operation is taking place:

Lemma 3.1.1. *Given a type family $C : A \to \mathcal{U}$, terms $a, b : A$ and a path $p : \mathsf{path}_A(a, b)$, there is a function*

$$p_* : C(a) \to C(b).$$

Proof. Assume we have $c : C(a)$. Note first that we have a path of types

$$D(p) :\equiv \lambda i.C(p(i)) : \mathbb{I} \to \mathcal{U}$$

from $C(a)$ to $C(b)$. The transportation of c from $\mathsf{0}$ to $\mathsf{1}$ over $D(p)$ gives us

$$c_{D(p)}^{\mathsf{0} \rightsquigarrow \mathsf{1}} : C(b)$$

as desired. □

Although $p_*(\mathsf{refl}_a)$ is not the identity function up to definitional equality, this can be shown to be the case up to a path.

Lemma 3.1.2. *For a type family $C : A \to \mathcal{U}$ and $a : A$, we have a path*

$$\mathsf{path}_{C(a) \to C(a)}((\mathsf{refl}_a)_*, \mathsf{Id}_{C(a)}).$$

Proof. By function extensionality, we may assume that $c : C(a)$ to find a path from $(\mathsf{refl}_a)_*(c) \equiv c^{0 \rightsquigarrow 1}_{\lambda\_.C(a)}$ to c. Now fix $i : \mathbb{I}$. We observe that $c^{i \rightsquigarrow 1}_{\lambda\_.C(a)} : C(a)$, since $(\lambda\_.C(a))(j) \equiv C(a)$ for any $j : \mathbb{I}$. But recall that the static transportation $c^{1 \rightsquigarrow 1}_{\lambda\_.C(a)}$ has no effect, so this term is definitionally equal to c. In other words, $\lambda i.c^{i \rightsquigarrow 1}_{\lambda\_.C(a)}$ is the path we are looking for. $\square$

3.2 Composition of paths

Composition ensures that every open cube can be filled. For example, at dimension one, if we are given adjacent lines $p, q, r : \mathbb{I} \to A$ such that the initial point of p matches the initial point of q and the terminal point of p matches the initial point of r, composition provides a new line, the composite, that goes from the terminal point of q to the terminal point of r.

$$\begin{tikzcd}
q(1) \arrow[r, dotted] & r(1) \\
q(0) \equiv p(0) \arrow[u, "q(j)"] \arrow[r, "p(i)"'] & p(1) \equiv r(0) \arrow[u, "r(j)"']
\end{tikzcd}
\qquad
\begin{tikzcd}
{} \\
{} \arrow[u, "j"] \arrow[r, "i"'] & {}
\end{tikzcd}$$

In fact, composition states more than that. It asserts the existence of the whole square witnessing the filling of the open box (the square in the diagram above). This is called the *filler* of the composition. When j is 1, the filler gives us the composite, the missing face of the open box. The remaining faces of the filler are the faces of the open box. So, when j is 0, the filler equals p (which we may call the cap of the composition) and when i is 0 or 1, it will be q (the left tube) or r (the right tube), respectively.

To illustrate the use of composition, we shall now prove symmetry and transitivity for paths. Let us consider the former first. In cubical type theories based on a De Morgan structure [7, 9], path inversion comes for free with a primitive (strict) reversal operator but in a cartesian account some work needs to be done to derive it from the Kan operations.

Lemma 3.2.1. *For every type A and every $a, b : A$, there is a function*

$$\mathsf{path}_A(a, b) \to \mathsf{path}_A(b, a)$$

called the inverse function and denoted $p \mapsto p^{-1}$.

Proof. Suppose that $i, j : \mathbb{I}$. The idea of the following proof is to observe that $p : \mathsf{path}_A(a, b)$ gives a "line" $p(j) : A$ in the j-direction from a to b in A. Since its initial point is a, we have an open box whose faces are formed by the lines

$p(j)$ (right tube), $\mathsf{refl}_a(i)$ (cap) and $\mathsf{refl}_a(j)$ (left tube). Note that the latter two lines are degenerate, i.e. by definition, they are the same as a. Degeneracy is indicated using double bars in the following diagram:

$$\begin{tikzcd} b \arrow[r, dotted, "p^{-1}(i)"] & a \arrow[d, equal, "a"] \\ a \arrow[u, "p(j)"] \arrow[r, equal, "a"'] & a \end{tikzcd}$$

By composition, this open box must have a lid (top), an i-line from b to a in A, which gives us, by path abstraction, the required path p^{-1}. $\square$

Fillers of compositions will be relevant to our constructions later on, so it is useful to have a special symbolism to talk about fillers in a convenient way. Thus, we have the following notation:

Definition 3.2.2. *Given a composition scenario and $j : \mathbb{I}$, if p stands for the composite of an open shape, then $\mathsf{fill}_j(p)$ stands for its filler in the j direction.*

When no confusion occurs, we may also write it $\mathsf{fill}(p)$. Thus, the (i, j)-square defined in the proof of Lemma 3.2.1 can be denoted by $\mathsf{fill}(p^{-1}(i))$.

The next lemma defines path concatenation:

Lemma 3.2.3. *For every type A and every $a, b, c : A$, there is a function*

$$\mathsf{path}_A(a, b) \to \mathsf{path}_A(b, c) \to \mathsf{path}_A(a, c)$$

denoted $p \mapsto q \mapsto p \cdot q$. We call $p \cdot q$ the concatenation of p and q.

Proof. Given paths $p : \mathsf{path}_A(a, b)$ and $q : \mathsf{path}_A(b, c)$, we can construct an i-line $p(i)$ from a to b and a j-line $q(j)$ from b to c. Once again, we note that we have an open square,

$$\begin{tikzcd} a \arrow[r, dotted, "p \cdot q(i)"] \arrow[d, equal, "a"'] & c \\ a \arrow[r, "p(i)"'] & b \arrow[u, "q(j)"'] \end{tikzcd}$$

and the path $p \cdot q$ is obtained by path abstraction on its composite. $\square$

We shall make diagrams as explicit as possible throughout the remainder of this paper, but, for the sake of readability, we omit labels for degenerated lines, since the reader should be able to correctly guess the information by checking the endpoints of the line. It is also important to bear in mind that, while we only considered the simplest composition scenario where we compose lines to form squares in the examples above, we can also compose squares to form cubes and, more generally, n-cubes to form $(n+1)$-cubes. We shall deal with compositions at dimension two in §4.1. Composition scenarios at even higher dimensions will not be discussed in this paper.

In addition to the specification of the cap and tubes of an open cube, we also have to admit the specification of diagonals to make composition work properly in a cartesian setting, thereby allowing the diagonal of the filler to be definitionally equal to the designated diagonal [1, 2]. But, as we will not be using diagonals, we do not need to worry about this here.

3.3 The interval is not Kan

We suggested above that every type is Kan. In fact, the interval is the only exception to this rule, since we have been implicitly treating it as a "type", but it actually does not support any Kan operations. Indeed, if the interval were Kan, then the identity path

$$\lambda i.i : \mathsf{path}_{\mathbb{I}}(0, 1)$$

would have an inverse, but what could that be? To deal with this fact, we adopt the convention that the interval can only occur as the antecedent of a function type (hence, the interval may be called a "pretype").

4 Two-dimensional constructions

Now that we have defined path symmetry and concatenation we can show that they satisfy the groupoid laws up to homotopy, which means that we shall be mainly concerned with paths of one dimension higher than the ones we are given. This is the aim of this section. Two-dimensional paths are determined by squares, which represent a mutual identification of lines on the opposing sides, and, as a result, their construction often requires two-extent compositions.

4.1 Weak connections

In cubical type theory, it is useful to have extra kinds of degeneracies known as connections, which can be thought of as meets and joins on the interval. Again, just as reversals, connections are built-in in a De Morgan setting [7, 9]. They are not hard to derive in cartesian cubical type theory, but some of the computation rules only hold up to a path (hence we call them "weak" connections).

Lemma 4.1.1 (Meet)**.** *Suppose $A : \mathcal{U}$, that $a, b : A$ and that $p : \mathsf{path}_A(a, b)$. There is an operation*

$$p(- \wedge -) : \mathbb{I} \to \mathbb{I} \to A$$

such that, for any $i, j : \mathbb{I}$, the following holds:

$$\begin{tikzcd}
a \arrow[rr, "p(i)"] & & b \\
a \arrow[u, equal] \arrow[rr, equal] \arrow[urr, phantom, "p(i \wedge j)"] & & a \arrow[u, "p(j)"']
\end{tikzcd}
\qquad
\begin{tikzcd}
\cdot \\
\cdot \arrow[u, "j"] \arrow[r, "i"'] & \cdot
\end{tikzcd}$$

The proof of this lemma requires a two-extent composition and thus a few remarks are in order here before we proceed. In the one-dimensional case, as we have seen as far, we just have a pair of tubes, so an open square is enough to complete the composition. With two pairs of tubes, each pair corresponding to the dimension in question, we are required to form an open object of the next higher shape, i.e. a cube. In other words, assuming that we want to fill an open (i, j, k)-cube in the j direction, we are expected to determine its (i, k)-face (bottom), two (k, j)-faces (left and right) and two (i, j)-faces (back and front), and those squares must all be adjacent up to definitional equality. If this holds, then the composite is a (i, k)-square. (For the same reasons, performing an n-extent composition requires a construction of an $(n + 1)$-cube).

Finally, it is often convenient to illustrate two-extent compositions using a two-dimensional structure in the form of a cube seen from above. The following diagram indicates how we shall be drawing two-extent compositions in the paper.

$$\textit{bottom}\left\{\begin{tikzcd}
\cdot \arrow[rrr] \arrow[dr] & & & \cdot \arrow[dl] \\
& \cdot \arrow[r] \arrow[r, phantom, shift left=3ex, "\textit{back}"] & \cdot & \\
& \cdot \arrow[u] \arrow[r] \arrow[ur, phantom, "\textit{top (composite)}"] \arrow[r, phantom, shift right=3ex, "\textit{front}"] \arrow[l, phantom, shift left=2ex, "\textit{right}"] & \cdot \arrow[u] \arrow[r, phantom, shift left=2ex, "\textit{left}"] & \\
\cdot \arrow[uuu] \arrow[rrr] \arrow[ur] & & & \cdot \arrow[uuu] \arrow[ul]
\end{tikzcd}\right.$$

When referring to such diagrams we may call the center and outer squares the top (composite) and the bottom faces of the cube (filler), the top and bottom squares the back and front faces, and the left and right squares the left and right faces, respectively.

Proof. Given $p : \mathsf{path}_A(a, b)$, we are to find a (i, j)-square whose top face is $p(i)$, right face is $p(j)$, left and bottom faces are a. First, by composition, we obtain, for any $i, j : \mathbb{I}$, a square that looks like a "halfway" connection, called $p(i \wedge^* j)$.

$$\begin{tikzcd}
a \arrow[rr, dotted, "p^*(i)"] & & b \\
a \arrow[u, equal] \arrow[rr, equal] \arrow[urr, phantom, "p(i \wedge^* j)"] & & a \arrow[u, "p(j)"']
\end{tikzcd}
\qquad
\begin{tikzcd}
\cdot \\
\cdot \arrow[u, "j"] \arrow[r, "i"'] & \cdot
\end{tikzcd}$$

Note that this square is not the one that we are looking for. Its top face is given by a certain path obtained by composition, which we denote p^*, and this path needs not be definitionally equal to p.

But we are able to fix this mismatch in a higher dimension by attaching this square to the top corner of an otherwise degenerated open cube in such a way that $p(- \wedge^* -)$ forms the back and right faces and the back right edge is p^*. This is depicted in the diagram below, where the bottom, front, and left faces of the open cube are a. We complete the proof with a two-extent composition, whose composite is shown as the shaded face in the diagram:

$$\begin{tikzcd}
a \arrow[rrr, equal] \arrow[ddd, equal] \arrow[dr, equal] \arrow[drr, phantom, "p(i \wedge^* j)"] & & & a \arrow[ddd, equal] \arrow[dl, dotted, "p^*(j)"'] \arrow[ddl, phantom, "p(k \wedge^* j)"] \\
& a \arrow[r, "p(i)"'] \arrow[d, equal] & b & \\
& a \arrow[r, equal] \arrow[dl, equal] & a \arrow[u, "p(k)"] \arrow[dr, equal] & \\
a \arrow[rrr, equal] & & & a
\end{tikzcd}$$

∎

Before moving on to the next connection, it is worth pointing out that the connections derived here could be slightly improved by attaching squares to the diagonal face of the open cubes of their compositions, thereby making the diagonal of the connections definitionally equal to p.

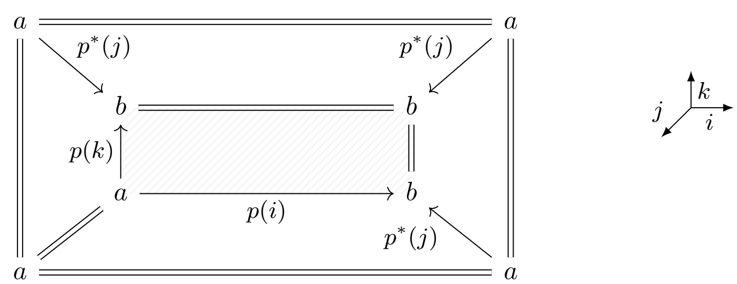
Lemma 4.1.2 (Join)**.** *Given $a, b : A$, for any $p : \mathsf{path}_A(a, b)$, there is a function*

$$p(- \vee -) : \mathbb{I} \to \mathbb{I} \to A$$

such that, for $i, j : \mathbb{I}$, we have:

$$\begin{tikzcd}
b \arrow[r, equal] \arrow[dr, phantom, "p(i \vee j)"] & b \\
a \arrow[u, "p(j)"] \arrow[r, "p(i)"'] & b \arrow[u, equal]
\end{tikzcd}$$

Proof. Using the semi-meet connection constructed in the previous proof, we perform a two-extent composition on an open cube given by semi-meets (front and left), degenerate squares formed from lines (back and right) and points (bottom). The composite gives us the desired square:



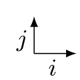
To check that this is a well-formed composition, note that we set a as the bottom (i, k)-face, $p(k \wedge^* j)$ as the left and $p^*(j)$ as the right (k, j)-faces, $p^*(j)$ as the back and $p(i \wedge^* j)$ as the front (k, j)-faces. Those squares are adjacent. □

We hope that the reader is starting to get a feel for proofs by composition and the interplay between two-dimensional paths and squares at this point, so we may omit uses of path abstraction without further comment, and we may also leave it up to the reader to show that the cap and tubes displayed in the composition diagrams respect the relevant adjacency conditions.

4.2 The groupoid laws

From the cubical perspective, path equality (homotopy) is always relative, since, viewed as squares, the only to say that two lines are the same is modulo an identification of two other lines. By contrast, homotopy type theory [12] has a globular approach. But we can simulate globular identifications of paths by considering certain squares whose remaining faces are degenerate lines.

$$\begin{tikzcd}
a \arrow[r, "q(i)"] \arrow[d, equal] & b \arrow[d, equal] \\
a \arrow[r, "p(i)"'] & b
\end{tikzcd}$$



The groupoid laws are stated using this globular representation. Put differently, when we state that reflexivity is a unit for path inversion and concatenation, that inversion is involutive, and concatenation is associative, for example, we mean that there exists a globular identification between them.

Because the proof is simpler, we start by showing that reflexivity is a right and left unit for path concatenation in the next two lemmas.

Lemma 4.2.1. *For every A and every $a, b : A$ we have a path*

$$\mathsf{ru}_p : \mathsf{path}_{\mathsf{path}_A(a,b)}(p, p \cdot \mathsf{refl}_b)$$

for any $p : \mathsf{path}_A(a, b)$.

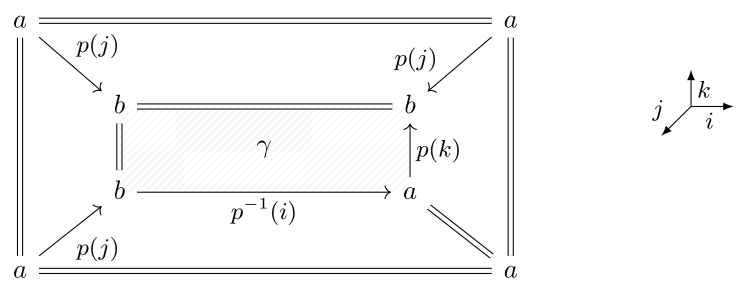
Proof. We need to construct an (i, j)-square having $p(i)$ and $(p \cdot \mathsf{refl}_b)(i)$ as i-lines and a and b as degenerate j-lines. But this already follows from the filler of concatenation defined in the proof of Lemma 3.2.3. $\square$

Lemma 4.2.2. *For every A and every $a, b : A$ we have a path*

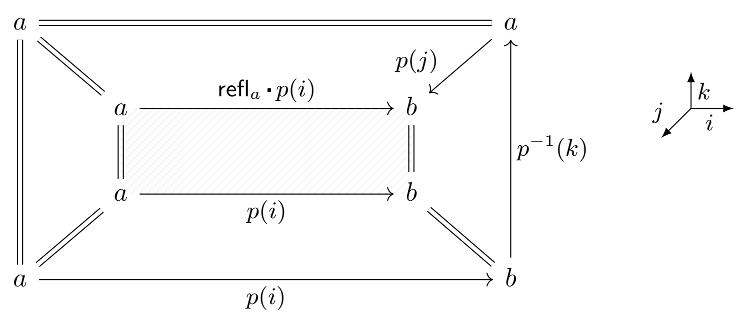
$$\mathsf{lu}_p : \mathsf{path}_{\mathsf{path}_A(a,b)}(p, \mathsf{refl}_a \cdot p)$$

for any $p : \mathsf{path}_A(a, b)$.

Proof. By composition, we define a helper (i, j)-square that goes from $p^{-1}(i)$ to b in the i-direction and from b to $p(j)$ in the j-direction. The composition uses the filler of the path inversion of p (front), meet (right), and degenerate squares formed from lines (back and left) and points (bottom). For future reference, we shall call it γ:



Forming a new open cube, we set γ at the right, the filler of the concatenation of refl_a and p at the back, the filler of the inversion of p at the bottom, and degenerate squares at the other faces.



$\square$

Why is the unit property so much simpler to demonstrate in the right? If we look attentively at the filler of path concatenation (Lemma 3.2.3), for example,

$$\begin{tikzcd}
a \arrow[r, "(p \cdot q)(i)"] & c \\
a \arrow[u, equal] \arrow[r, "p(i)"'] & b \arrow[u, "q(j)"']
\end{tikzcd}$$

we can see that it forms a simultaneous identification that can be pronounced "let p be $p \cdot q$ just in case q is refl_a". Consequently, if we set $q :\equiv \mathsf{refl}_a$, we immediately have a globular path from p to $p \cdot q$. We can thus compare path concatenation with the transitivity operation defined in the homotopy type theory book by path induction on the second argument [12, lem.2.1.2]. The same idea applies to path inversion, "let p^{-1} be refl_a just in case p is refl_a", so it is related to the symmetry operation defined by path induction on p [12, lem.2.1.1].

Next we prove that path inversion indeed is a right and left inverse with respect to concatenation:

Lemma 4.2.3. *For every A and every $a, b : A$ we have a path*

$$\mathsf{rc}_p : \mathsf{path}_{\mathsf{path}_A(a,a)}(\mathsf{refl}_a, p \cdot p^{-1})$$

for any $p : \mathsf{path}_A(a, b)$.

Proof. By composition, we must construct a cube whose composite is an (i, k)-square with $\mathsf{refl}_a(i)$ and $p \cdot p^{-1}(i)$ as i-lines and a in both degenerate k-lines. Now consider the following open (i, j, k)-cube

$$\begin{tikzcd}
a \arrow[rrr, "p(i)"] \arrow[dr, equal] \arrow[ddd, equal] & & & b \arrow[dl, "p^{-1}(j)"'] \arrow[ddd, equal] \\
& a \arrow[r, equal] \arrow[d, equal] & a \arrow[d, equal] & \\
& a \arrow[r, "p \cdot p^{-1}(i)"'] & a & \\
a \arrow[rrr, "p(i)"'] \arrow[ur, equal] & & & b \arrow[ul, "p^{-1}(j)"]
\end{tikzcd}$$

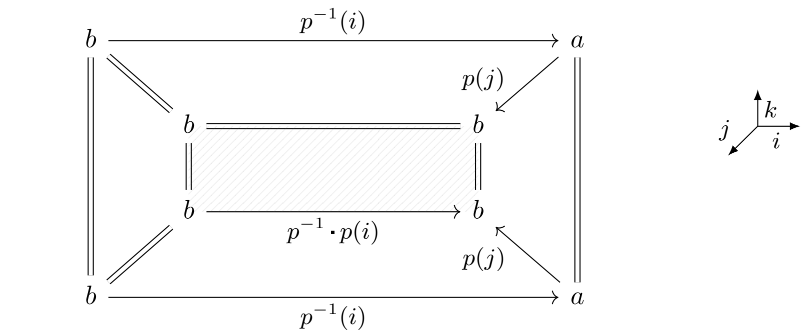
whose bottom, left and right faces are degenerate squares, and back and front squares are respectively the fillers for path inversion and concatenation. □

Lemma 4.2.4. *For every A and every $a, b : A$ we have a path*

$$\mathsf{lc}_p : \mathsf{path}_{\mathsf{path}_A(b,b)}(\mathsf{refl}_b, p^{-1} \cdot p)$$

for any $p : \mathsf{path}_A(a, b)$.

Proof. By composition on the following open cube, whose back face is the γ square defined in the proof of Lemma 4.2.2.



∎

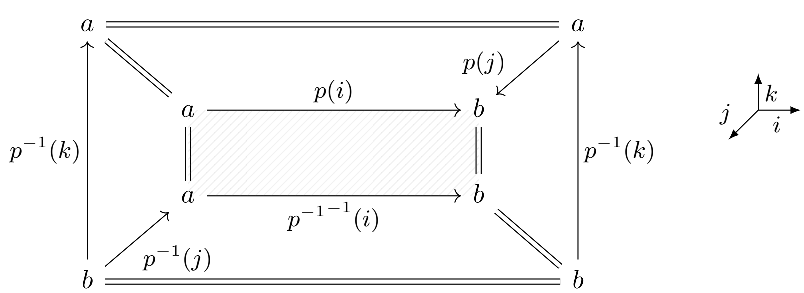
The following lemma states that path inversion is involutive:

Lemma 4.2.5. *For every A and every $a, b : A$, we have a path*

$$\mathsf{inv}_p : \mathsf{path}_{\mathsf{path}_A(a,b)}(p, (p^{-1})^{-1})$$

for any $p : \mathsf{path}_A(a, b)$.

Proof. The proof follows by the use of meets, joins and γ to form the composite:



∎

Last but not least, we want to show that path concatenation is associative.

Lemma 4.2.6. *For every A and every $a, b, c, d : A$, we have a path*

$$\mathsf{assoc}_{p,q,r} : \mathsf{path}_{\mathsf{path}_A(a,d)}((p \cdot q) \cdot r, p \cdot (q \cdot r))$$

for any $p : \mathsf{path}_A(a, b)$, $q : \mathsf{path}_A(b, c)$, $r : \mathsf{path}_A(c, d)$.

Proof. We use the fact that, for any two squares with the same three faces, there is a square showing that the fourth sides are equal. In particular, given the following two squares with definitionally equal bottom, right and left faces

$$\begin{tikzcd} a \arrow[r, "(p\cdot(q\cdot r))(i)"] \arrow[d, equal] \arrow[dr, phantom, "\alpha"] & d \\ a \arrow[r, "p(i)"'] & b \arrow[u, "(q\cdot r)(j)"'] \end{tikzcd} \qquad \begin{tikzcd} a \arrow[r, "((p\cdot q)\cdot r)(i)"] \arrow[d, equal] \arrow[dr, phantom, "\beta"] & d \\ a \arrow[r, "p(i)"'] & b \arrow[u, "(q\cdot r)(j)"'] \end{tikzcd}$$

we can construct the desired path homotopy

$$\begin{tikzcd} a \arrow[rrr, "p(i)"] \arrow[dr, equal] \arrow[ddd, equal] & & & b \arrow[dl, "(q\cdot r)(j)"'] \arrow[ddd, equal] \\ & a \arrow[r, "(p\cdot(q\cdot r))(i)"] \arrow[d, equal] & d \arrow[d, equal] & \\ & a \arrow[r, "((p\cdot q)\cdot r)(i)"'] & d & \\ a \arrow[rrr, "p(i)"'] \arrow[ur, equal] & & & b \arrow[ul, "(q\cdot r)(j)"] \end{tikzcd}$$

Since α is just the filler of path concatenation $p\cdot(q\cdot r)$, it remains to construct β. But this is easy, because looking at the top and right sides of this square, the filler of path concatenation gives us a canonical construction:

$$\begin{tikzcd} a \arrow[rrr, "(p\cdot q)(i)"] \arrow[dr, equal] \arrow[ddd, equal] & & & c \arrow[dl, "r(j)"'] \\ & a \arrow[r, "((p\cdot q)\cdot r)(i)"] \arrow[d, equal] & d & \\ & a \arrow[r, "p(i)"'] & b \arrow[u, "(q\cdot r)(k)"'] \arrow[dr, equal] & \\ a \arrow[rrr, "p(i)"'] \arrow[ur, equal] & & & b \arrow[uuu, "q(j)"'] \end{tikzcd}$$

□

What about the higher groupoid structure of types? What we have shown in this section is that types have a 1-groupoid structure, but since those laws do not hold "on the nose" as definitional equations, they must satisfy some equations of their own. We will not cover this here. Instead we give a proof of path induction in §6.1, which corresponds almost exactly to the elimination rule of the identity type in Martin-Löf type theory (except that the computation rule here only holds up to a path), and conjecture that it should be possible to derive them with an approach similar to [11].

5 Dependent paths

The various operations and laws defined in the previous section all share a fundamental limitation: they are only applicable to a specific class of paths. This restriction is imposed by our implicit requirement that a path be non-dependent, which means that the type $\mathsf{path}_A(a, b)$ is not well-formed unless a and b have exactly the type A. While this restriction is important when trying to understand cubical methods, it rules out the formation of paths in paths between types $A : \mathbb{I} \to \mathcal{U}$, which, for the lack of a better name, we shall call *type lines*, types that may change depending on their endpoints.

5.1 The dependent path type

Given a type line $A : \mathbb{I} \to \mathcal{U}$ and terms $a : A(\mathsf{0})$ and $b : A(\mathsf{1})$, the type of dependent paths from a to b is written $\mathsf{pathd}_A(a, b)$. If the underlying type line is a constant function, that is, the reflexivity path, then the dependent path type is the ordinary path type

$$\mathsf{pathd}_{\lambda\_.A}(a, b) \equiv \mathsf{path}_A(a, b).$$

The rules for the dependent path type arise as straightforward generalizations of the ones for path types. For instance, as with non-dependent paths, we eliminate a term of this type by application, $p(i) : A(i)$, where $p : \mathsf{pathd}_A(a, b)$ and $i : \mathbb{I}$. The constructor of this type is path abstraction, and we say that $\lambda i.a : \mathsf{pathd}_A(a[\mathsf{0}/i], a[\mathsf{1}/i])$ if $a : A(i)$ assuming that $i : \mathbb{I}$, where $a[\mathsf{0}/i] : A(\mathsf{0})$ and $a[\mathsf{1}/i] : A(\mathsf{1})$. Moreover, we also require equalities that are no different from the ones for the ordinary path type. In particular, given $p : \mathsf{pathd}_A(a, b)$ and $i : \mathbb{I}$, we have boundary rules $p(\mathsf{0}) \equiv a : A(\mathsf{0})$ and $p(\mathsf{1}) \equiv a : A(\mathsf{1})$, and the uniqueness rule $\lambda i.(p(i)) \equiv p$ when i does not occur in p.

Why do we need the dependent path type? Very often, we want to work with paths in type lines, that is, squares that are not globular. Consider for example the meet operator defined in Lemma 4.1.1. What, exactly, is its type? Starting with an iterated function type

$$\lambda j.\lambda i.p(i \wedge j) : \mathbb{I} \to \mathbb{I} \to A$$

we can obtain a path in the type of paths from a to $p(j)$

$$\lambda j.\lambda i.p(i \wedge j) : \prod_{(j:\mathbb{I})} \mathsf{path}_A(a, p(j)),$$

because, given a fixed j, $\lambda i.p(i \wedge j)$ varies from the point a to $p(j)$. By repeating the process, we obtain a dependent path from refl_a to p in the type of paths from a to $p(j)$

$$\lambda j.\lambda i.p(i \wedge j) : \mathsf{pathd}_{\lambda i.\mathsf{path}_A(a,p(i))}(\mathsf{refl}_a, p)$$

since $\lambda j.\lambda i.p(i \wedge j)$ goes from

$$\lambda i.p(i \wedge 0) \equiv \lambda i.a$$

to

$$\begin{aligned} \lambda i.p(i \wedge 1) &\equiv \lambda i.p(i) \\ &\equiv p. \end{aligned}$$

5.2 Heterogeneous composition

Let us consider the limitations of non-dependent path inversion first. Assuming that $\mathsf{pathd}_A(a, b)$ is a type, where $a : A(0)$ and $b : A(1)$, the type $\mathsf{pathd}_A(b, a)$ of inverse paths will *not* be well-formed unless it is also the case that $b : A(0)$ and $a : A(1)$. For non-dependent paths this always holds since the type line is a constant function, i.e. $A(0) \equiv A \equiv A(1)$. But this is not true in general, and Lemma 3.2.1 fails to produce inverses for dependent paths.

Fortunately, there is a way to deal with this problem using type universes. Suppose we are given a type line $A : \mathbb{I} \to \mathcal{U}$, that is, a non-dependent path from the type $A(0)$ to $A(1)$ in the universe $\mathcal{U}$,

$$A : \mathsf{path}_{\mathcal{U}}(A(0), A(1)).$$

We can assume that this is a non-dependent path because type universes never depend on interval variables, so we have both $A(0), A(1) : \mathcal{U}$.

Now it can be shown by Lemma 3.2.1 that the following inverse exists:

$$\begin{tikzcd} A(1) \arrow[rr, dotted, "A^{-1}(i)"] & & A(0) \arrow[d, equal] \\ A(0) \arrow[u, "A(j)"] \arrow[rr, equal] & & A(0) \end{tikzcd}$$

In particular, we have

$$A^{-1} : \mathsf{path}_{\mathcal{U}}(A(1), A(0)),$$

which corresponds precisely to the "inverse" type of A, for the initial and terminal points of A^{-1} are respectively $A(1)$ and $A(0)$. Put differently, we have two inferences that hold top/bottom and bottom/top,

$$\frac{a : A^{-1}(1)}{a : A(0)} \quad \text{and} \quad \frac{a : A^{-1}(0)}{a : A(1)}.$$

Thus, assuming that $\mathsf{pathd}_A(a, b)$ is a type, it is now easy to see that $\mathsf{pathd}_{A^{-1}}(b, a)$ will always be a type as well, regardless of whether A is a constant line or not: if $\mathsf{pathd}_A(a, b)$ is a type then we have $a : A(0)$ and $b : A(1)$, meaning that $a : A^{-1}(1)$ and $b : A^{-1}(0)$ must be the case.

This motivates the definition of a dependent path inversion operation:

Lemma 5.2.1. *For every type $A : \mathbb{I} \to \mathcal{U}$ and every $a : A(\mathsf{0})$ and $b : A(\mathsf{1})$, there is a function*

$$\mathsf{pathd}_A(a, b) \to \mathsf{pathd}_{A^{-1}}(b, a)$$

called the (dependent) inverse path and denoted $p \mapsto p^{-1}$.

Proof. Dependent operations are typically introduced by repeating, mutatis mutandis, the proofs of their non-dependent counterparts. More specifically, the results which were previously proven by composition will now follow from a "heterogeneous" variant of composition based on the same open shapes. This adaptation is required because we are dealing with non-constant type lines and dependent paths, and all the faces of an open shape must have the same type.

More concretely, the open box used in the definition of non-dependent path inversion (the one used in the proof of Lemma 3.2.1) is ill-formed in this context since $p(j) : A(j)$ and $a : A(\mathsf{0})$ are lines with different types:

$$\begin{tikzcd} b & & a \\ a \arrow[u, "p(j)"] \arrow[rr, equal] & & a \arrow[u, equal] \end{tikzcd}$$

The solution is not to form an open box living in A, but rather to perform a composition taking place in A^{-1}, using transport on the type line $\lambda j.\mathsf{fill}_j(A^{-1}(i))$, the filler of the inverse of A, to adjust the type of the faces of open box.

For the cap, we want to transport a from $\mathsf{0}$ to $\mathsf{1}$. This gives us the correct type because $\mathsf{fill}_\mathsf{1}(A^{-1}(i)) \equiv A^{-1}(i)$. Now we observe that $a : \mathsf{fill}_\mathsf{0}(A^{-1}(i)) \equiv A(0)$, so we have

$$a^{\mathsf{0} \rightsquigarrow \mathsf{1}}_{\lambda j.\mathsf{fill}_j(A^{-1}(i))} : A^{-1}(i).$$

Now for the left tube, i.e. assuming that $i = \mathsf{0}$, we transport p from j to $\mathsf{1}$. More specifically, since $p(j) : \mathsf{fill}_j(A^{-1}(\mathsf{0})) \equiv A(j)$, and, again, $\mathsf{fill}_\mathsf{1}(A^{-1}(\mathsf{0})) \equiv A^{-1}(i)$, we have

$$(p(j))^{j \rightsquigarrow \mathsf{1}}_{\lambda j.\mathsf{fill}_j(A^{-1}(\mathsf{0}))} : A^{-1}(i).$$

The right tube is treated similarly, except that we transport a (and $i = \mathsf{1}$),

$$a^{j \rightsquigarrow \mathsf{1}}_{\lambda j.\mathsf{fill}_j(A^{-1}(\mathsf{1}))} : A^{-1}(i).$$

It is not hard to show that those lines are adjacent. We want to form a composition from $\mathsf{0}$ to $\mathsf{1}$, so the initial point of the left tube should match the initial point of the cap,

$$\begin{aligned}(\lambda j.(p(j))^{j\rightsquigarrow 1}_{\lambda j.\mathsf{fill}_j(A^{-1}(0))})(0) &\equiv (p(0))^{0\rightsquigarrow 1}_{\lambda j.\mathsf{fill}_j(A^{-1}(0))}\\ &\equiv a^{0\rightsquigarrow 1}_{\lambda j.\mathsf{fill}_j(A^{-1}(0))}\\ &\equiv a^{0\rightsquigarrow 1}_{\lambda j.\mathsf{fill}_j(A^{-1}(i))}\end{aligned}$$

and the initial point of the right tube should be the terminal point of the cap,

$$\begin{aligned}(\lambda j.a^{j\rightsquigarrow 1}_{\lambda j.\mathsf{fill}_j(A^{-1}(1))})(0) &\equiv a^{0\rightsquigarrow 1}_{\lambda j.\mathsf{fill}_j(A^{-1}(1))}\\ &\equiv a^{0\rightsquigarrow 1}_{\lambda j.\mathsf{fill}_j(A^{-1}(i))}.\end{aligned}$$

By composition, we have a line from b to a in $A^{-1}(i)$. This composite has the correct endpoints because

$$(p(1))^{1\rightsquigarrow 1}_{\lambda j.\mathsf{fill}_j(A^{-1}(0))} \equiv p(1) \equiv b$$

and

$$a^{1\rightsquigarrow 1}_{\lambda j.\mathsf{fill}_j(A^{-1}(1))} \equiv a.$$

□

The proof of the previous lemma is based on *heterogeneous composition*, a particular kind of composition in which the types of the cap and composite may differ. It can be obtained from composition and transport. For instance, given a type line $A : \mathbb{I} \to \mathcal{U}$, a heterogeneous composition in A with cap $a : I \to A(0)$ and tubes $a_0, a_1 : \prod_{(i:\mathbb{I})} A(i)$ is just an abbreviation for a compounded composition which combines the two Kan operations into one, a composition with cap $\lambda i.(a(i))^{0\rightsquigarrow 1}_A : I \to A(1)$ and tubes $\lambda j.(a_0(j))^{j\rightsquigarrow 1}_A : I \to A(1)$ and $\lambda j.(a_1(j))^{j\rightsquigarrow 1}_A : I \to A(1)$. The composite is a term of type $I \to A(1)$.

At this point, one may be tempted to think that we can drop the non-dependent path inversion operation from Lemma 3.2.1, since we now already possess a more general (dependent) notion of inversion. On second thought, however, it becomes clear that this is not possible on pain of circularity. To put it another way, a preliminary (non-dependent) inversion operation is absolutely necessary in order to define the type A^{-1}, so that the definition of dependent path inversion is not circular.

Non-dependent path concatenation suffers from a similar limitation. Consider the two-dimensional paths

$$\alpha : \mathsf{pathd}_{\lambda j.\mathsf{path}_A(p'(j),q'(j))}(p,q) \qquad \text{and} \qquad \beta : \mathsf{pathd}_{\lambda j.\mathsf{path}_A(q''(j),r''(j))}(q,r),$$

which correspond to the following two (i,j)-squares in A:

$$
\begin{tikzcd}
a \arrow[r, "q(i)"] & b \\
c \arrow[u, "p'(j)"'] \arrow[r, "p(i)"'] \arrow[ur, phantom, "\alpha(j)(i)"] & d \arrow[u, "q'(j)"']
\end{tikzcd}
\qquad
\begin{tikzcd}
c \arrow[r, "r(i)"] & d \\
e \arrow[u, "q''(j)"'] \arrow[r, "q(i)"'] \arrow[ur, phantom, "\beta(j)(i)"] & f \arrow[u, "r''(j)"']
\end{tikzcd}
\qquad
\begin{tikzcd}
{} \arrow[u, "j"] \arrow[r, "i"'] & {}
\end{tikzcd}
$$

It is clear that we can compose α and β vertically by composing the horizontal lines, as illustrated in the following square:

$$
\begin{tikzcd}
a \arrow[r, "r(i)"] & b \\
e \arrow[u, "(p' \cdot q'')(j)"] \arrow[r, "p(i)"'] \arrow[ur, phantom, "\alpha?\beta(j)(i)"] & f \arrow[u, "(q' \cdot r'')(j)"']
\end{tikzcd}
\qquad
\begin{tikzcd}
{} \arrow[u, "j"] \arrow[r, "i"'] & {}
\end{tikzcd}
$$

but this is not an instance of the composition operation defined in Lemma 3.2.3: to form the concatenated path $\alpha \cdot \beta$, α and β must be paths in the same type and it must be a degenerate line. In this example, the concatenation fails for both reasons, for the equality $\mathsf{path}_A(p'(j), q'(j)) \equiv \mathsf{path}_A(q''(j), r''(j))$ need *not* be the case and those types may depend on the interval variable j.

To overcome this problem we need a dependent path concatenation operation. Following the definition of dependent path inversion, we consider the non-dependent path concatenation of line types first. Suppose that $A, B : \mathbb{I} \to \mathcal{U}$ such that $A(\mathsf{1}) \equiv B(\mathsf{0})$. The diagram

$$
\begin{tikzcd}
A(\mathsf{0}) \arrow[r, dotted, "(A \cdot B)(i)"] \arrow[d, equal] & B(\mathsf{1}) \\
A(\mathsf{0}) \arrow[r, "A(i)"'] & A(\mathsf{1}) \equiv B(\mathsf{0}) \arrow[u, "B(j)"']
\end{tikzcd}
\qquad
\begin{tikzcd}
{} \arrow[u, "j"] \arrow[r, "i"'] & {}
\end{tikzcd}
$$

illustrates the path concatenation of A and B,

$$A \cdot B \ : \ \mathsf{path}_{\mathcal{U}}(A(\mathsf{0}), B(\mathsf{1})).$$

We also have two important inferences that hold top/bottom and bottom/top,

$$\frac{a : (A \cdot B)(0)}{a : A(\mathsf{0})} \quad \text{and} \quad \frac{b : (A \cdot B)(1)}{b : B(\mathsf{1})}.$$

With this, we have all we need to define concatenation for dependent paths:

Lemma 5.2.2. *Suppose that $A, B : \mathbb{I} \to \mathcal{U}$ such that $A(1) \equiv B(0)$. Given any $a : A(0)$, $b : A(1)$ and $c : B(1)$, there is a function*

$$\mathsf{pathd}_A(a, b) \to \mathsf{pathd}_B(b, c) \to \mathsf{pathd}_{A \bullet B}(a, c)$$

written $p \mapsto q \mapsto p \cdot q$ and called the dependent path concatenation function.

Proof. By heterogeneous composition on the open box from Lemma 3.2.3. □

Dependent path concatenation allows us to concatenate α and β from our example above, but it is worth noting that the resulting path need not be definitionally equal to the operation $\alpha ? \beta$ we described. However, it is easy to show by path induction that they are equal up to globular identification.

5.3 More on groupoid laws

Using dependent path inverse and concatenation, it is possible to generalize the groupoid structure from §4.2 deriving laws that hold for dependent paths as well. It is helpful to understand how this works with an example.

Let us examine the involution lemma (Lemma 4.2.5), which actually tells us a fact about constant line types $A : \mathbb{I} \to \mathcal{U}$. It states that, for every $a : A(0)$, $b : A(1)$ and $p : \mathsf{pathd}_A(a, b)$, there is a path

$$\mathsf{path}_{\mathsf{pathd}_A(a,b)}((p^{-1})^{-1}, p).$$

If we were to drop the restriction that A be a constant type line and then use dependent path inversion instead, we would have to state something like

$$\mathsf{pathd}_{\lambda j.\mathsf{pathd}_{\lambda i.?(j)(i)}(a,b)}((p^{-1})^{-1}, p)$$

Note that we run into a problem at this point: we need a path from $(p^{-1})^{-1}$: $\mathsf{pathd}_{(A^{-1})^{-1}}(a, b)$ to $p : \mathsf{pathd}_A(a, b)$, and we must specify in what type this path lives in. But recall that, by Lemma 4.2.5, there is a path

$$\mathsf{inv}_A : \mathsf{path}_{\mathsf{path}_{\mathcal{U}}(A(0),A(1))}((A^{-1})^{-1}, A),$$

so

$$\mathsf{pathd}_{\lambda j.\mathsf{pathd}_{\lambda i.\mathsf{inv}_A(j)(i)}(a,b)}((p^{-1})^{-1}, p)$$

is the path type we are looking for.

Lemma 5.3.1. *For every line type $A : \mathbb{I} \to \mathcal{U}$ with $a : A(0)$ and $b : A(1)$, we have*

$$\mathsf{pathd}_{\lambda j.\mathsf{pathd}_{\lambda i.\mathsf{inv}_A(j)(i)}(a,b)}((p^{-1})^{-1}, p)$$

for any $p : \mathsf{path}_A(a, b)$.

The proof argument is similar to the ones given in the definition of dependent path inversion (Lemma 5.2.1) and composition (Lemma 5.2.2), as it is a straightforward heterogeneous composition on the open cube constructed for the proof of the non-dependent counterpart (Lemma 4.2.5). In fact, all dependent counterparts of the propositions from §4.2 follow the same pattern (they can all be stated by using their non-dependent counterparts and proven by a heterogeneous filling of their open cubes), so we will simply omit those results.

6 Notable properties of paths

Before closing our naive presentation of cubical type theory, we would like to explore a few notable properties of paths from a cubical perspective: path induction and the fact that path concatenation operation on the second loop space is commutative.

6.1 Path induction

We now present a derivation of path induction [12, §1.12.1], a key property that states that paths can be deformed and retracted without changing their essential characteristics. It is also known as J, and, as mentioned before, it serves as the eliminator of the identity type in homotopy type theory.

Theorem 6.1.1 (Path induction)**.** *Given a type $A : \mathcal{U}$, a term $a : A$ and a type family $C : \prod_{(x:A)} \mathsf{path}_A(a, x) \to \mathcal{U}$, we have a function*

$$\mathsf{pathrec} : \prod_{(x:A)} \prod_{(p:\mathsf{path}_A(a,x))} \prod_{(c:C(a,\mathsf{refl}_a))} C(x, p).$$

Proof. We want to construct, for every $x : A$, $p : \mathsf{path}_A(a, x)$ and $c : C(a, \mathsf{refl}_a)$, a term of type $C(x, p)$. To obtain such a term, we shall transport c over a type line $D : \mathbb{I} \to \mathcal{U}$ that goes from $D(\mathsf{0}) :\equiv C(a, \mathsf{refl}_a)$ to $D(\mathsf{1}) :\equiv C(x, p)$.

Recall from Lemma 4.1.1 that we have a meet square that can be regarded as a double identification of a given path with reflexivity.

$$\begin{tikzcd}
a \arrow[r, "p(i)"] & b \\
a \arrow[u, equal] \arrow[r, equal] \arrow[ur, phantom, "p(i \wedge j)"] & a \arrow[u, "p(j)"']
\end{tikzcd}$$

This square induces the desired type line

$$D :\equiv \lambda i.C(p(i), \lambda j.p(i \wedge j)) : \mathbb{I} \to \mathcal{U}$$

because it goes from

$$\begin{aligned} D(0) &\equiv (\lambda i.C(p(i), \lambda j.p(i \wedge j)))0 \\ &\equiv C(p(0), \lambda j.p(0 \wedge j)) \\ &\equiv C(a, \lambda j.(\lambda\_.a)j) \\ &\equiv C(a, \mathsf{refl}_a) \end{aligned}$$

to

$$\begin{aligned} D(1) &\equiv (\lambda i.C(p(i), \lambda j.p(i \wedge j)))1 \\ &\equiv C(p(1), \lambda j.p(1 \wedge j)) \\ &\equiv C(x, \lambda j.(\lambda j.p(j))j) \\ &\equiv C(x, \lambda j.p(j)) \\ &\equiv C(x, p). \end{aligned}$$

Now we complete the proof by transporting $c : D(0)$ from 0 to 1. $\square$

Note that, in the above proof, the η-rule for the path type discussed in §2.1, that is, the requirement that $p \equiv \lambda j.(p(j))$, is crucial to the correct specification of endpoints of the type line we are doing the transportation over. Without this rule, it would not be possible to show that $D(1) \equiv C(x, p)$ and the transportation would give us a term of the wrong type.

In cubical type theory, the computation rule for path induction does not hold "on the nose" like in the case of the elimination rule of the identity type [12]. Put differently, for a fixed type family $C : \prod_{(x:A)} \mathsf{path}_A(a, x) \to \mathcal{U}$, given $a : A$ and $c : C(a, \mathsf{refl}_a)$, in general,

$$\mathsf{pathrec}(a, \mathsf{refl}_a, c) \not\equiv c.$$

But this equality does hold up to a path:

Lemma 6.1.2 (Computation)**.** *For every $a : A$ and $c : C(a, \mathsf{refl}_a)$, we have*

$$\mathsf{path}_{C(a,\mathsf{refl}_a)}(\mathsf{pathrec}(a, \mathsf{refl}_a, c), c)$$

Proof. Unfolding the definition of path induction, we have to find a path from

$$\mathsf{pathrec}(a, \mathsf{refl}_a, c) \equiv c^{0 \rightsquigarrow 1}_{\lambda i.C(a, \lambda j.\mathsf{refl}_a(i \wedge j))} : C(a, \mathsf{refl}_a)$$

to c in the type $C(a, \mathsf{refl}_a)$. Now, since, for any $i : \mathbb{I}$, we have

$$c^{i \rightsquigarrow i}_{\lambda i.C(a, \lambda j.\mathsf{refl}_a(i \wedge j))} \equiv u,$$

the transportation induces a path

$$\lambda i.c^{i \rightsquigarrow 1}_{\lambda i.C(a, \lambda j.\mathsf{refl}_a(i \wedge j))}$$

that goes from $\mathsf{pathrec}(a, \mathsf{refl}_a, c)$ to c in the type $C(a, \lambda j.\mathsf{refl}_a(i \wedge j))$.

This path has the right endpoints, but it still does not live in the correct type. To conclude this proof we fix this type mismatch with a second transportation from i to $\mathsf{0}$ over the type line

$$\lambda i.\mathsf{path}_{C(a,\lambda j.\mathsf{refl}_a(i \wedge j))}(\mathsf{pathrec}(a, \mathsf{refl}_a, c), c).$$

$\square$

In the presence of higher inductive types [6, 9], the usual identity type from homotopy type theory [12] can be recovered as a higher inductive type freely generated by reflexivity [6]. Path induction then acts strictly on reflexivity, since it is given as a specific generator that can be recognized by the eliminator, and this allows for a strict computation rule. This identity type can be shown to be equivalent to the path type [6], meaning that, by univalence, the identity type is the path type up to a path.

6.2 Eckmann–Hilton

Cubically, loops are paths with the same endpoints up to definitional equality. The loop space of a is given by the type $\mathsf{path}_A(a, a)$, the loop space of the loop space of a, which is the space of two-dimensional loops on the constant path at a, is represented by $\mathsf{path}_{\mathsf{path}_A(a,a)}(\mathsf{refl}_a, \mathsf{refl}_a)$. Concatenation of paths in the second loop space is commutative, just as in homotopy type theory [12, §2.1].

Theorem 6.2.1 (Eckmann–Hilton)**.** *For any* $\alpha, \beta : \mathsf{path}_{\mathsf{path}_A(a,a)}(\mathsf{refl}_a, \mathsf{refl}_a)$, *there is a path*

$$\mathsf{path}_{\mathsf{path}_{\mathsf{path}_A(a,a)}(\mathsf{refl}_a, \mathsf{refl}_a)}(\alpha \bullet \beta, \beta \bullet \alpha)$$

Proof. Following the proof of Theorem 2.1.6 in [12], it is easier to prove a stronger statement that holds more generally for two-paths by path induction, and then derive the intended claim as a special case.

Suppose we are given paths $\alpha : \mathsf{path}_{\mathsf{path}_A(a,b)}(p, q)$ and $\beta : \mathsf{path}_{\mathsf{path}_A(b,c)}(r, s)$. First we define a right whiskering operation

$$\alpha \bullet_{\mathsf{r}} r : \mathsf{path}_{\mathsf{path}_A(a,c)}(p \bullet r, q \bullet r)$$

in the obvious way such that the composition holds:

$$\begin{tikzcd}
a \arrow[rrr, "p(i)"] \arrow[ddd, equal] \arrow[dr, equal] & & & b \arrow[dl, "r(j)"'] \arrow[ddd, equal] \\
& a \arrow[r, "(p \bullet r)(i)"] \arrow[d, equal] \arrow[dr, phantom, "(\alpha \bullet_{\mathsf{r}} r)(j)(i)"] & c \arrow[d, equal] & \\
& a \arrow[r, "(q \bullet r)(i)"'] & c & \\
a \arrow[rrr, "q(i)"'] \arrow[ur, equal] & & & b \arrow[ul, "r(j)"]
\end{tikzcd}$$

j k i

Similarly, we define left whiskering

$$p \cdot_{\mathsf{l}} \beta : \mathsf{path}_{\mathsf{path}_A(a,c)}(p \cdot r, p \cdot s)$$

using the composition

$$\begin{tikzcd}
a \arrow[rrr, "p(i)"] \arrow[dr, equal] \arrow[ddd, equal] & & & b \arrow[dl, "r(j)"'] \arrow[ddd, equal] \\
& a \arrow[r, "(p \cdot r)(i)"] \arrow[d, equal] \arrow[dr, phantom, "(p \cdot_{\mathsf{l}} \beta)(j)(i)"] & c \arrow[d, equal] & \\
& a \arrow[r, "(p \cdot s)(i)"'] & c & \\
a \arrow[rrr, "p(i)"'] \arrow[ur, equal] & & & b \arrow[ul, "s(j)"]
\end{tikzcd}$$

it is easy to see by path induction on α, β, p and r that there exists a path

$$\mathsf{path}_{\mathsf{path}_{\mathsf{path}_A(a,c)}(p \cdot r, q \cdot s)}((\alpha \cdot_{\mathsf{r}} r) \cdot (q \cdot_{\mathsf{l}} \beta), (p \cdot_{\mathsf{l}} \beta) \cdot (\alpha \cdot_{\mathsf{r}} s))$$

Now let $p \equiv q \equiv r \equiv s \equiv \mathsf{refl}_a$. Since reflexivity is both a right and left unit for path concatenation (see Lemmas 4.2.1 and 4.2.2), the above proposition demonstrates our intended claim. $\square$

7 Directions for future work

There is much to be done yet in order to provide a cubical alternative to the informal type theory project of the homotopy type theory book [12]. We view this paper as opening up many possibilities for future work, including informal cubical accounts of the higher groupoid structure of type formers, univalence, higher inductive types, homotopy n-types and the development of mathematics such as homotopy theory, category theory or set theory.

Part of the proofs contained in this paper have been formalized in the interactive theorem provers Cubical Agda and Redtt, and are available online.[1]

Acknowledgments The author wishes to thank Carlo Angiuli, Steve Awodey, Evan Cavallo, Robert Harper and Anders Mörtberg for helpful comments on an earlier draft of this paper. This work was supported by the US Air Force Office of Scientific Research (AFOSR) grant FA9550-18-1-0120. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the AFOSR.

[1] See e.g. `https://github.com/agda/cubical/blob/master/Cubical/Foundations/GroupoidLaws.agda` or `https://github.com/RedPRL/redtt/blob/master/library/prelude/path.red`

References

[1] Carlo Angiuli, Guillaume Brunerie, Thierry Coquand, Kuen-Bang Hou (Favonia), Robert Harper, and Daniel Licata. Syntax and models of cartesian cubical type theory, 2019. URL: `https://github.com/dlicata335/cart-cube`. Preprint.

[2] Carlo Angiuli, Kuen-Bang Hou (Favonia), and Robert Harper. Cartesian cubical computational type theory: Constructive reasoning with paths and equalities. *Computer Science Logic 2018*, 2018.

[3] Steve Awodey. A cubical model of homotopy type theory. *Annals of Pure and Applied Logic*, 169(12):1270–1294, 2018.

[4] Steve Awodey and Michael A Warren. Homotopy theoretic models of identity types. *Mathematical Proceedings of the Cambridge Philosophical Society*, 146(1):45–55, 2009.

[5] Marc Bezem, Thierry Coquand, and Simon Huber. A model of type theory in cubical sets. *19th International Conference on Types for Proofs and Programs (TYPES 2013)*, 26:107–128, 2014.

[6] Evan Cavallo and Robert Harper. Computational Higher Type Theory IV: Inductive Types. URL: `https://arxiv.org/pdf/1801.01568.pdf`, 01 2018. Preprint.

[7] Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. Cubical type theory: a constructive interpretation of the univalence axiom. In *TYPES 2015, Bristol*, pages 5:1–5:34. North-Holland, Tallinn, Estonia, 5 2015.

[8] Robert L. Constable. Naïve computational type theory. In Helmut Schwichtenberg and Ralf Steinbrüggen, editors, *Proof and System-Reliability*, pages 213–259. Springer Netherlands, Dordrecht, 2002.

[9] Thierry Coquand, Simon Huber, and Anders Mörtberg. On higher inductive types in cubical type theory. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, pages 255–264, New York, NY, USA, 2018. ACM.

[10] Paul R Halmos. *Naive set theory*. Springer-Verlag, New York, 1974.

[11] Peter LeFanu Lumsdaine. Weak omega-categories from intensional type theory. *Logical Methods in Computer Science*, Volume 6, Issue 3, September 2010.

[12] The Univalent Foundations Program. Homotopy type theory: Univalent foundations of mathematics, 2013.

[13] Vladimir Voevodsky. A very short note on the homotopy λ-calculus. URL: http://www.math.ias.edu/vladimir/files/2006_09_Hlambda.pdf, 2006. Unpublished note.

[14] Vladimir Voevodsky. Notes on type systems. URL: https://www.math.ias.edu/~vladimir/Site3/Univalent_Foundations_files/expressions_current.pdf, 2009. Unpublished note.