Eliminating reversals from cubical type theories

Citation for the original published paper (version of record):
Cavallo, E., Sattler, C. (2026). Eliminating reversals from cubical type theories. Leibniz International Proceedings in Informatics, LIPIcs, 380: 27:1-27:7. http://dx.doi.org/10.4230/LIPIcs.LICS.2026.27

Eliminating Reversals from Cubical Type Theories

Evan Cavallo
Department of Computer Science and Engineering, University of Gothenburg, Sweden
Department of Computer Science and Engineering, Chalmers University of Technology, Sweden

Christian Sattler
Department of Computer Science and Engineering, Chalmers University of Technology, Sweden
Department of Computer Science and Engineering, University of Gothenburg, Sweden

Abstract

Cubical type theories are designed around an abstract unit interval from which types of paths, used to represent equalities, are defined. Varying the operations available on this interval yields different type theories. A reversal is an involutive operator on the interval that swaps its two endpoints. We show that for cubical type theories with self-dual interval theories, such as the minimal theory of two endpoints or the theory of a bounded distributive lattice, the extension of the theory with a reversal that internalizes the duality is a conservative extension. The key tool is a "twist construction": the product of an interval and its dual is again an interval with a reversal given by swapping coordinates.

Our conservativity result applies to "opaque" cubical type theories, without strict equations reducing the filling operator at concrete type formers or eliminators from higher inductive types at path constructors. Using the same twist construction, we also construct models of strict cubical type theory with reversals in categories of cubical sets without reversals. We thereby give the first model of a theory with reversals whose homotopy theory corresponds to that of topological spaces.

2012 ACM Subject Classification Theory of computation → Type theory; Theory of computation → Constructive mathematics

Keywords and phrases Dependent type theory, univalence, cubical type theory

Digital Object Identifier 10.4230/LIPIcs.LICS.2026.27

Related Version *Preprint Version*: `https://arxiv.org/abs/2605.15080`

Funding *Evan Cavallo*: Knut and Alice Wallenberg Foundation (KAW), Grant No. 2019.0116.
Christian Sattler: US Air Force Office of Scientific Research, award number FA9550-24-1-0302.

1 Introduction

Cubical type theories [12, 3, 2] extend Martin-Löf's dependent type theory [24] with an abstract unit interval $\mathbb{I}$ which behaves much like a type. Types of *paths* $a_0 \sim^A a_1$, i.e., of terms $i : \mathbb{I} \vdash a(i) : A$ varying over the interval with fixed endpoints $a(0) = a_0$ and $a(1) = a_1$, play the role of equality types. As equality types, path types are remarkably well-behaved. For example, they natively satisfy function extensionality: equalities of functions correspond to families of pointwise equalities. With additional type formers, cubical type theories can also support Voevodsky's univalence axiom and higher inductive types (HITs) [13, 9], making them models of homotopy type theory (HoTT) [39].

Path types satisfy different strict equations than Martin-Löf's identity types. On the one hand, they do not support a J eliminator with a strict computation rule [12, §9.1]. On the other hand, for example, one has an operator witnessing that functions $f : A \to B$ preserve paths, $\mathsf{cong}_f \coloneqq \lambda p.\lambda i.f(p(i)) : (a_0 \sim^A a_1) \to (f(a_0) \sim^B f(a_1))$, that commutes *strictly* with function composition: $\mathsf{cong}_g \circ \mathsf{cong}_f = \mathsf{cong}_{g \circ f}$. Such equations make cubical type theory a convenient setting for *synthetic homotopy theory* (see, e.g., Mörtberg and Pujet [25]), homotopy theory developed in the language of type theory, which can involve complicated manipulations with iterated identity/path types.

The range of strict equations satisfied by a cubical type theory's path types depends on its *interval theory*, the collection of operations available on $\mathbb{I}$. Given a *reversal* operator $i : \mathbb{I} \vdash \neg i : \mathbb{I}$ such that $\neg 0 = 1$, $\neg 1 = 0$, and $\neg\neg i = i$, we can define a path inversion operator

$$\mathsf{sym} \coloneqq \lambda p.\lambda i.p(\neg i) : (a_0 \sim^A a_1) \to (a_1 \sim^A a_0)$$

that is strictly involutive ($\mathsf{sym} \circ \mathsf{sym} = \mathrm{id}$) and commutes strictly with the action of functions ($\mathsf{cong}_f \circ \mathsf{sym} = \mathsf{sym} \circ \mathsf{cong}_f$). *Connections* $i : \mathbb{I}, j : \mathbb{I} \vdash i \wedge j : \mathbb{I}$ and $i : \mathbb{I}, j : \mathbb{I} \vdash i \vee j : \mathbb{I}$ behaving like the min and max functions on the topological interval are similarly useful for higher-dimensional manipulations. Cohen, Coquand, Huber, and Mörtberg's original cubical type theory [12] includes $\neg$, $\wedge$, and $\vee$ with the equational theory of the free De Morgan algebra. On the other hand, Angiuli, Favonia, and Harper's theory [3] demonstrates that none of these operators is *necessary* to set up a well-behaved cubical type theory.

While convenient for the user of the type theory, additional operations on the interval are less convenient for the semanticist. To justify the project of synthetic homotopy theory, a cubical type theory should at least have a model in ∞-*groupoids*, an abstract description of the homotopy theory of topological spaces. Constructive models classically equivalent to ∞-groupoids were found first for cubical type theory without any interval operations by Awodey, Cavallo, Coquand, Riehl, and Sattler [6] and then for the theory with one connection $\vee$ by Cavallo and Sattler [11]. Most recently, the second-named author announced [32] a model constructively equivalent to ∞-groupoids that can interpret cubical type theory with two connections, $\wedge$ and $\vee$, and the equations of a bounded distributive lattice. However, none of these models interpret a reversal. This is a particularly unfortunate state of affairs because Cubical Agda [41], the most widely used proof assistant for cubical type theory, is based on Cohen et al.'s type theory and thus includes $\neg$ along with $\vee$ and $\wedge$, and its substantial standard library [36] relies extensively on these operators.

1.1 Contributions

We show that a reversal is an essentially harmless extension to cubical type theory.

The key fact is that when $\mathbb{I}$ is an interval object with endpoints 0 and 1, its square $\mathbb{I} \times \mathbb{I}$ is an interval object with endpoints $(0, 1)$ and $(1, 0)$ and a reversal $\neg(i_0, i_1) \coloneqq (i_1, i_0)$ that swaps the axes of the square. When $\mathbb{I}$ has connections defining a distributive lattice $(\mathbb{I}, \wedge, \vee, 0, 1)$, $\mathbb{I} \times \mathbb{I}$ is a De Morgan algebra with connections given by $(i_0, i_1) \wedge (j_0, j_1) \coloneqq (i_0 \wedge j_0, i_1 \vee j_1)$ and $(i_0, i_1) \vee (j_0, j_1) \coloneqq (i_0 \vee j_0, i_1 \wedge j_1)$. In general, when $\mathbb{I}$ has some self-dual algebraic structure (in a sense we make precise in Section 4.1), $\mathbb{I} \times \mathbb{I}$ has the same structure as well as a reversal. A variety of constructions in this mold appear in the algebraic literature (e.g., in lattice and order theory), where they are called *twist constructions*. This name originates with Kracht [23], who applies it to a construction of Nelson algebras from Heyting algebras taken from Vakarelov [40]. Fidel and Brignole [17] and Rivieccio [30, §7] consider the case of building De Morgan algebras from distributive lattices, which is of particular interest to us.

We derive two main results from this simple construction.

1.1.1 Conservativity for opaque cubical type theory

First, we prove that a reversal is a conservative extension for "opaque" cubical type theories with self-dual interval theories. Similar to a theory considered by Coquand, Huber, and Sattler [14], these opaque theories are cubical type theories where certain strict equations are either omitted or replaced with terms of path type. Specifically, we

(a) omit equations that reduce uses of the filling operator at concrete type formers, and
(b) weaken equations for the reduction of HIT eliminators on path constructors to paths.

The equations of (a) always hold up to paths, by higher dimensional-instances of filling, so omitting them also amounts to weakening them to paths.

Building on the twist construction, we define a *twist interpretation* from opaque cubical type theories with reversals to corresponding theories without reversals, interpreting judgments $\Gamma \vdash i : \mathbb{I}$ as pairs $(\Gamma \vdash i_0 : \mathbb{I}, \Gamma \vdash i_1 : \mathbb{I})$ and path types as *square* types – encoded as iterated path types – with fixed values at the points $(0, 1)$ and $(1, 0)$. We use this translation to prove a conservativity result: for a context of term variables Γ and type $\Gamma \vdash A$ in the base theory and a term $\Gamma \vdash_\neg N : A$ in the extended theory, there is a term $\Gamma \vdash M : A$ in the base theory with a path $\Gamma \vdash_\neg P : M \sim^A N$. Similarly, if $\Gamma \vdash_\neg B$ is a type in the extended theory, then there is a type $\Gamma \vdash A$ with an equivalence $\Gamma \vdash_\neg E : A \simeq B$.

Coquand, Huber, and Sattler [14] prove *homotopy canonicity* for their opaque cubical type theory: every closed term of natural number type is connected by a path to a concrete numeral. Our hope is that similar techniques can be used to show that strict cubical type theories may be conservative over their opaque counterparts in general. Some progress towards a framework for coherence theorems dealing with strictification of equations has been made by Bocquet [7]. With such a result, the program of relating cubical theories with different interval structure could be conducted in the simpler world of opaque theories, as we do here for reversals, then mechanically extended to strict theories.

1.1.2 Models for strict cubical type theory with reversals in spaces

In lieu of a hoped-for conservativity result for strict over opaque cubical type theories, we separately use the same basic twist construction to build concrete models of strict cubical type theory with reversals. We work with the parameterized model construction of Angiuli, Brunerie, Coquand, Harper, Favonia, and Licata (ABCHFL) [2], showing that any model of cubical type theory given by this construction can be upgraded to a model of a cubical type theory with reversals in the same target category.

Combining this with our prior work showing that the homotopy theory of a certain ABCHFL model is classically equivalent to ∞-groupoids [11], we obtain a model of strict cubical type theory with a reversal whose homotopy theory is classically equivalent to ∞-groupoids. This is the first known model of its kind. Although Cohen, Coquand, Huber, and Mörtberg [12] give a model of their type theory, which includes reversals, this model and others like it have pathological homotopy theories [31]. Our models avoid these pathologies.

To obtain a similar model for strict cubical type theory with reversals *and* connections, we would need an input ABCHFL model of the theory with connections in ∞-groupoids. At present, no such model is known: while the theory with connections has an ABCHFL model in cartesian cubical sets with connections, it is an open problem to characterize its homotopy theory, as discussed by Streicher and Weinberger [33]. The second-named author has claimed a model of cubical type theory with connections whose homotopy theory is that of ∞-groupoids [32], but it is not a direct instance of the ABCHFL construction. We expect that our work adapts to this model, but we leave this for future work.

1.2 Outline

In § 2, we set the stage to study cubical type theories in generality by reviewing Uemura's framework of *second-order generalized algebraic theories* and their semantics based on *representable map categories* [37, 38]. We recall Kapulkin and Lumsdaine's definition of *weak equivalence* for models of type theory with Σ and identity types [22], which we will use to state our conservativity theorem. In § 3, we define cubical type theory in these terms.

We begin studying opaque cubical type theory in § 4, where we define the extension of a self-dual interval theory with a reversal and the *twist interpretation* from a cubical type theory extended with a reversal back to the original theory. In § 5, we define the *representable map category of spans* and develop tools to relate pairs of interpretations between cubical type theories. We use these in § 6 to prove a general theorem for deriving weak equivalences from interpretations; we apply it with the twist interpretation to deliver the conservativity of reversals over opaque cubical type theories with self-dual interval theories. In § 7, we construct a model of strict cubical type theory with reversals whose homotopy theory is classically that of ∞-groupoids.

2 Type theories and models

2.1 SOGATs

We use Uemura's framework of *second-order generalized algebraic theories (SOGATs)* [37, Chapter 4] and their semantic counterparts, *representable map categories* [37, Chapter 4] [38] (also called *categories with representable maps*). The language of SOGATs is a *logical framework* (cf. Harper, Honsell, and Plotkin [18]) with which we can specify type theories themselves in type-theoretic language. In a SOGAT, we specify the judgment forms of a type theory while marking some as hypothesizable or *representable*. To begin with an example, *basic dependent type theory* $\mathbb{M}\textsc{ltt}$ [37, Example 4.6.1] is specified by two sorts, one for types and one for terms:

$$\mathsf{Ty} \;:\; () \Rightarrow \square \qquad\qquad \mathsf{Tm} \;:\; (\mathtt{A} : () \to \mathsf{Ty}) \Rightarrow \star$$

The type sort Ty takes no parameters ($() \Rightarrow \ldots$) and is *non-representable* ($\square$), so hypotheses "$A : \mathsf{Ty}$" cannot appear in the contexts of the type theory being specified. The term sort Tm takes one type ($\mathtt{A} : () \to \mathsf{Ty}$) as a parameter and is *representable* ($\star$), meaning we allow hypotheses "$a : \mathsf{Tm}(A)$".

In general, a SOGAT consists of declarations $\Phi \Rightarrow s$ where Φ is an *environment* and s is either $\square$, $\star$, a previously introduced sort, or an equation between expressions such a sort. An environment is a list $(\mathtt{A}_1 : \Gamma_1 \to e_1, \ldots, \mathtt{A}_n : \Gamma_n \to e_n)$ of metavariables $\mathtt{A}_i$ that take a *context* Γ_i and output an e_i which is either a previously defined sort or an equation between expressions in such a sort. Finally, a *context* is a list $(\mathtt{a}_1 : A_1, \ldots, \mathtt{a}_n : A_n)$ of variables $\mathtt{a}_i$ of representable sorts A_i. Everything can depend on what precedes it. We refer to Uemura [37, Chapter 4] for a much more precise description.

▶ **Notation 1.** *We omit empty contexts* ($() \to \ldots$) *and environments* ($() \Rightarrow \ldots$)*, writing, e.g.,* $\mathsf{Ty} : \square$ *and* $\mathsf{Tm} : (\mathtt{A} : \mathsf{Ty}) \Rightarrow \star$*. We also omit variable names when what follows does not depend on them, as in* $\mathsf{Tm} : \mathsf{Ty} \Rightarrow \star$*. We surround arguments in square brackets when we intend to leave them implicit, as in the following example of* Σ *types. Specifically to* $\mathbb{M}\textsc{ltt}$*: we leave the* Tm *operator implicit and write* $\mathtt{a} : \mathtt{A}$ *rather than* $\mathtt{a} : \mathsf{Tm}(\mathtt{A})$*.*

▶ **Notation 2** (cf. [37, Remark 5.4.6])**.** *Any context can be treated as an environment; one level up, any environment* Φ *over a SOGAT* T *can be treated as an extension of* T *with new declarations. We write* $T[\Phi]$ *for the extended SOGAT combining* T *with* Φ*.*

Uemura gives encodings of standard type formers of Martin-Löf type theory such as (negative) unit types, (negative) dependent sums, and dependent products [37, §4.6.1]. For example, dependent sums are specified by the declarations

$$\begin{array}{lcl}\Sigma & : & (\mathtt{A} : \mathsf{Ty}, \mathtt{B} : \mathtt{A} \to \mathsf{Ty}) \Rightarrow \mathsf{Ty} \\ \mathsf{fst} & : & ([\mathtt{A} : \mathsf{Ty}, \mathtt{B} : \mathtt{A} \to \mathsf{Ty}], \Sigma(\mathtt{A}, \mathtt{B})) \Rightarrow \mathtt{A} \\ \mathsf{snd} & : & ([\mathtt{A} : \mathsf{Ty}, \mathtt{B} : \mathtt{A} \to \mathsf{Ty}], \Sigma(\mathtt{A}, \mathtt{B})) \Rightarrow \mathtt{B}(\mathtt{a}) \\ \mathsf{pair} & : & ([\mathtt{A} : \mathsf{Ty}, \mathtt{B} : \mathtt{A} \to \mathsf{Ty}], \mathtt{a} : \mathtt{A}, \mathtt{b} : \mathtt{B}(\mathtt{a})) \Rightarrow \Sigma(\mathtt{A}, \mathtt{B})\end{array}$$

and, over $\Phi_\Sigma = (\mathtt{A} : \mathsf{Ty}, \mathtt{B} : \mathtt{A} \to \mathsf{Ty})$, equations

$$\begin{array}{lcl}\_ & : & (\Phi_\Sigma, \mathtt{a} : \mathtt{A}, \mathtt{b} : \mathtt{B}(\mathtt{a})) \Rightarrow \mathsf{fst}(\mathsf{pair}(\mathtt{a}, \mathtt{b})) \equiv \mathtt{a} : \mathtt{A} \\ \_ & : & (\Phi_\Sigma, \mathtt{a} : \mathtt{A}, \mathtt{b} : \mathtt{B}(\mathtt{a})) \Rightarrow \mathsf{snd}(\mathsf{pair}(\mathtt{a}, \mathtt{b})) \equiv \mathtt{b} : \mathtt{B}(\mathtt{a}) \\ \_ & : & (\Phi_\Sigma, \mathtt{s} : \Sigma(\mathtt{A}, \mathtt{B})) \Rightarrow \mathtt{s} \equiv \mathsf{pair}(\mathsf{fst}(\mathtt{s}), \mathsf{snd}(\mathtt{s})) : \Sigma(\mathtt{A}, \mathtt{B})\end{array}$$

▶ **Notation 3.** *We write* $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$ *for the extension of* $\mathbb{M}\mathrm{LTT}$ *with* Σ *types, unit types (which we think of as nullary* Σ *types), and identity types [37, Examples 4.6.4–4.6.6]. We write* $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id},\Pi}$ *for its further extension with* Π *types [37, Example 4.6.3].*

▶ **Notation 4.** *We write* $\Sigma\mathtt{a}{:}A.\ B$ *as shorthand for* $\Sigma(A, \langle\mathtt{a}\rangle B)$*, where* $\langle\mathtt{a}\rangle$ *denotes abstraction over the variable* $\mathtt{a}$*. If* B *does not depend on* $\mathtt{a}$*, we write* $A \times B$*. We write* (a, b) *for the pairing* $\mathsf{pair}(a, b)$ *and* $s.1$ *and* $s.2$ *for* $\mathsf{fst}(s)$ *and* $\mathsf{snd}(s)$*, respectively. For* Π *types, we write* $\Pi\mathtt{a}{:}A.\ B$ *for* $\Pi(A, \langle\mathtt{a}\rangle B)$ *and* $A \to B$ *when* B *does not depend on* $\mathtt{a}$*. For identity types, we write the type* $\mathsf{Id}(A, a_0, a_1)$ *of identities in* A *from* a_0 *to* a_1 *as* $a_0 \asymp^A a_1$ *or* $a_0 \asymp a_1$*.*

▶ **Notation 5.** *The unit type and dependent sums justify types of dependent* n*-tuples for* $n \geq 0$*. We write these with tupling* $(a_1, \ldots, a_n)$ *and projections* $s.1, \ldots, s.n$*.*

2.2 Representable map categories

To specify the notion of *model* of a SOGAT, Uemura first introduces *representable map categories*, also called *categories with representable maps*.

▶ **Definition 6** (Uemura [37, Definition 3.2.1])**.** *A* representable map category (RMC) *is a finite limit category* $\mathbb{R}$ *equipped with a class of morphisms, the* representable maps*, such that*

(a) *the representable maps are closed under pullback, and*

(b) *for each representable map* $f : Y \to X$*, the pullback functor* $f^* : \mathbb{R}/X \to \mathbb{R}/Y$ *has a right adjoint* $f_* : \mathbb{R}/Y \to \mathbb{R}/X$ *(called* pushforward*).*

We use the arrow style $\rightarrowtriangle$ *to indicate representable maps. A* representable map functor *or* RMC functor $F : \mathbb{R} \to \mathbb{S}$ *between representable map categories is a functor that preserves finite limits, representable maps, and pushforwards along representable maps.*

▶ **Example 7** (Uemura [37, Example 3.2.2])**.** Let $\mathcal{C}$ be a small category. The category of presheaves $\mathrm{PSh}(\mathcal{C})$ becomes an RMC when equipped with the class of morphisms $f : B \to A$ such that for every map $a : \text{ɹ}c \to A$ from a representable presheaf, there is a pullback square

$$\begin{array}{ccc} \text{ɹ}d & \dashrightarrow & B \\ \vdots & \lrcorner & \downarrow f \\ \downarrow & & \\ \text{ɹ}c & \xrightarrow[a]{} & A \end{array}$$

for some $d \in \mathcal{C}$.

The collection of representable map categories, representable map functors between them, and natural isomorphisms defines a (2,1)-category **RMC**. Each SOGAT T induces a "*syntactic*" RMC $\mathbb{C}\mathrm{L}(T)$ [37, §4.8] whose objects are environments Φ over T and whose

morphisms are instantiations: an *instantiation* $I \colon \Phi \to \Psi$ where $\Psi = (\mathtt{A}_1 : \Gamma_1 \to e_1, \ldots, \mathtt{A}_n : \Gamma_n \to e_n)$ is an assignment $(\mathtt{A}_1 \coloneqq \langle \vec{\mathsf{a}_1} \rangle t_1, \ldots, \mathtt{A}_n \coloneqq \langle \vec{\mathsf{a}_n} \rangle t_n)$ sending each metavariable $\mathtt{A}_i$ in the target to an expression $t_i : e_i$ in context $\vec{\mathsf{a}_i} : \Gamma_i$ over $T[\Phi]$. An instantiation is representable when it is isomorphic to the projection $\Phi.\Gamma \to \Phi$ for an extension of an environment Φ by a context Γ. For concrete SOGATs, we usually suppress $\mathbb{C}\textsc{l}(-)$ and use the same name for the SOGAT and its induced RMC.

The RMC $\mathbb{C}\textsc{l}(T)$ has a (2, 1)-categorical universal property that characterizes RMC functors $\mathbb{C}\textsc{l}(T) \to \mathbb{R}$ up to isomorphism as *interpretations* [37, Theorem 4.8.18]. An interpretation of T in $\mathbb{S}$ is a specification of the image of each declaration of T inside $\mathbb{S}$. For example, an RMC functor $F \colon \mathbb{M}\textsc{ltt} \to \mathbb{S}$ is determined up to isomorphism by an object $F\mathsf{Ty} \in \mathbb{S}$ and a representable map $F\pi_{\mathsf{Tm}} \colon F\mathsf{Tm} \twoheadrightarrow F\mathsf{Ty}$, which specifies the image of $\pi_{\mathsf{Tm}} \colon (\mathtt{A} : \mathsf{Ty}, \mathsf{a} : \mathsf{Tm}(\mathtt{A})) \to (\mathtt{A} : \mathsf{Ty})$. As a special case, we can speak of interpretations of a SOGAT T in another SOGAT S as interpretations of T in $\mathbb{C}\textsc{l}(S)$.

2.3 Models

An interpretation $\mathbb{C}\textsc{l}(T) \to \mathbb{S}$ is a model of a SOGAT as a *second-order* theory. To recover a notion of *first-order* model, corresponding for example to categories with families [16] for $\mathbb{M}\textsc{ltt}$, Uemura uses presheaf categories with representable maps:

▶ **Definition 8** ([37, §3.2.4])**.** *A* model $\mathcal{M} = (\mathcal{C}, M)$ *of an RMC* $\mathbb{R}$ *is a category* $\mathcal{C}$ *with a terminal object and an RMC functor* $M \colon \mathbb{R} \to \mathrm{PSh}(\mathcal{C})$ *to the presheaf RMC of Example 7. We write* $\mathcal{M}(\star)$ *for* $\mathcal{C}$ *and* $\mathcal{M}(X) \coloneqq MX \in \mathrm{PSh}(\mathcal{M}(\star))$ *for* $X \in \mathbb{R}$*.*

A morphism $\mathcal{F} = (F, \alpha) \colon \mathcal{M} \to \mathcal{N}$ *between models is a functor* $F \colon \mathcal{M}(\star) \to \mathcal{N}(\star)$ *and family of natural transformations* $\alpha_X \colon \mathcal{M}(X) \to F^*\mathcal{N}(X)$*, natural in* $X \in \mathbb{R}$*, such that for each representable* $f \colon Y \twoheadrightarrow X$*, the naturality square for* α *at* f *satisfies a Beck–Chevalley condition. For* $c \in \mathcal{M}(\star)$*, we write* $\mathcal{F}(c) \in \mathcal{N}(\star)$ *for* Fc*. For* $x \colon \yoneda c \to \mathcal{M}(X)$ *in* $\mathrm{PSh}(\mathcal{C})$*, we write* $\mathcal{F}_X(x) \colon \yoneda \mathcal{F}(c) \to \mathcal{N}(X)$ *for the map corresponding by Yoneda to* $\alpha_X \circ x \colon \yoneda c \to F^*\mathcal{N}(X)$*.*

Models of $\mathbb{M}\textsc{ltt}$ in this sense correspond directly to natural models as defined by Awodey [4], and thereby to categories with families: $\mathcal{M}(\star)$ interprets the context judgment, and $\mathcal{M}(\mathsf{Ty})$ and $\mathcal{M}(\mathsf{Tm})$ the type and term judgments over a context. With an appropriate notion of 2-morphism, the collection of models of an RMC $\mathbb{R}$ forms a (2,1)-category $\mathbf{Mod}(\mathbb{R})$.

▶ **Definition 9** ([37, Definitions 5.1.4 & 5.1.6])**.** *The class of* contextual objects *in* $\mathcal{M}(\star)$ *for a model* $\mathcal{M} \in \mathbf{Mod}(\mathbb{R})$ *is inductively generated as follows:*

1. *terminal objects* $1 \in \mathcal{M}(\star)$ *are contextual;*
2. *for each contextual* $c \in \mathcal{M}(\star)$*, representable* $f \colon Y \twoheadrightarrow X$ *in* $\mathbb{R}$*, and pullback square*

$$\begin{array}{ccc} \yoneda d & \longrightarrow & \mathcal{M}(Y) \\ \downarrow & \lrcorner & \downarrow{\scriptstyle \mathcal{M}(f)} \\ \yoneda c & \longrightarrow & \mathcal{M}(X) \end{array}$$

with $d \in \mathcal{M}(\star)$*, the object* d *is contextual.*

A model is democratic *when all of its objects are contextual. The* heart *(or* contextual core*)* $\mathcal{M}^\heartsuit \in \mathbf{Mod}(\mathbb{R})$ *of* $\mathcal{M}$ *is defined by taking* $\mathcal{M}^\heartsuit(\star)$ *to be the full subcategory of contextual objects in* $\mathcal{M}(\star)$ *and* $\mathcal{M}^\heartsuit(X)$ *to be the restriction of* $\mathcal{M}(X)$ *to a presheaf on* $\mathcal{M}^\heartsuit(\star)$*.*

2.4 Weak equivalences

Kapulkin and Lumsdaine [22, Definition 3.1] define *weak equivalences* of contextual categories with identity types. We translate their definition into Uemura's framework as a property of morphisms in $\mathbf{Mod}(\mathbb{M}_{\mathrm{LTT}_{\Sigma,\mathsf{Id}}})$. First, we define the environment $\mathsf{Ty}^{\simeq}$ of *1-to-1 correspondences*, pairs of types connected by a type-valued relation that associates each element of one type with a unique element of the other. This is one way of defining *equivalence* between types [39, Exercise 4.2]. Similarly, we have an environment $\mathsf{Tm}^{\asymp}$ of pairs of identified elements within a type.

▶ **Definition 10.** *Over* $\mathtt{A} : \mathsf{Ty}$, *define* $\Phi_{\mathsf{isContr}}(\mathtt{A}) \coloneqq (\mathtt{a}_0{:}\mathtt{A}, \mathtt{p} : (\mathtt{a}_1{:}\mathtt{A}) \to \mathtt{a}_0 \asymp^{\mathtt{A}} \mathtt{a}_1)$. *Write* $\mathsf{Ty}^{\simeq} \in \mathbb{M}_{\mathrm{LTT}_{\Sigma,\mathsf{Id}}}$ *for*

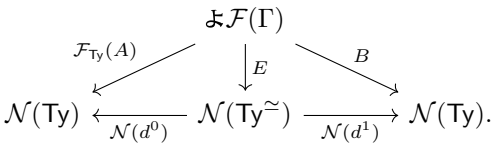
$$(\mathtt{A} : \mathsf{Ty}, \mathtt{A}' : \mathsf{Ty}, \overline{\mathtt{A}} : (\mathtt{a} : \mathtt{A}, \mathtt{a}' : \mathtt{A}') \to \mathsf{Ty},$$
$$\_ : (\mathtt{a} : \mathtt{A}) \to \Phi_{\mathsf{isContr}}(\Sigma \mathtt{a}'{:}\mathtt{A}'.\ \overline{\mathtt{A}}(\mathtt{a}, \mathtt{a}')),\ \_ : (\mathtt{a}' : \mathtt{A}') \to \Phi_{\mathsf{isContr}}(\Sigma \mathtt{a}{:}\mathtt{A}.\ \overline{\mathtt{A}}(\mathtt{a}, \mathtt{a}')))$$

and $d^0, d^1 : \mathsf{Ty}^{\simeq} \to \mathsf{Ty}$ *for the maps projecting* $\mathtt{A}$ *and* $\mathtt{A}'$ *respectively.*

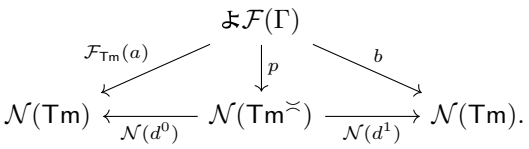
▶ **Definition 11.** *Set* $\mathsf{Tm}^{\asymp} \coloneqq (\mathtt{A} : \mathsf{Ty}, \mathtt{a} : \mathtt{A}, \mathtt{a}' : \mathtt{A}, \overline{\mathtt{a}} : \mathtt{a} \asymp^{\mathtt{A}} \mathtt{a}')$ *and write* $d^0, d^1 : \mathsf{Tm}^{\asymp} \to \mathsf{Tm}$ *for the maps projecting* $(\mathtt{A}, \mathtt{a})$ *and* $(\mathtt{A}, \mathtt{a}')$.

▶ **Definition 12.** *A morphism* $\mathcal{F} : \mathcal{M} \to \mathcal{N}$ *in* $\mathbf{Mod}(\mathbb{M}_{\mathrm{LTT}_{\Sigma,\mathsf{Id}}})$ *is a* weak equivalence *if the following hold for all* $\Gamma \in \mathcal{M}(\star)$*:*

(a) *weak type lifting: for every* $B : \text{ɬ}\mathcal{F}(\Gamma) \to \mathcal{N}(\mathsf{Ty})$*, there exist* $A : \text{ɬ}\Gamma \to \mathcal{M}(\mathsf{Ty})$ *and* $E : \text{ɬ}\mathcal{F}(\Gamma) \to \mathcal{N}(\mathsf{Ty}^{\simeq})$ *fitting in a commutative diagram*



(b) *weak term lifting: for every* $A : \text{ɬ}\mathcal{F}(\Gamma) \to \mathcal{N}(\mathsf{Ty})$ *and* $b : \text{ɬ}\Gamma \to \mathcal{M}(\mathsf{Tm})$ *with* $\pi_{\mathsf{Tm}} b = \mathcal{F}_{\mathsf{Ty}}(A)$*, there exist* $a : \text{ɬ}\Gamma \to \mathcal{M}(\mathsf{Tm})$ *with* $\pi_{\mathsf{Tm}} a = A$ *and* $p : \text{ɬ}\mathcal{F}(\Gamma) \to \mathcal{N}(\mathsf{Tm}^{\asymp})$ *fitting in a commutative diagram*



Though we state Definition 12 for arbitrary models, it is generally only well-behaved for democratic models. In informal turnstile notation, $\mathcal{F} : \mathcal{M} \to \mathcal{N}$ is a weak equivalence when

(a) for every type $\mathcal{F}(\Gamma) \vdash_{\mathcal{N}} B$ in the target model, there is a type $\Gamma \vdash_{\mathcal{M}} A$ in the source model whose image by $\mathcal{F}$ is equivalent to B, and

(b) for every term $\mathcal{F}(\Gamma) \vdash_{\mathcal{N}} b : \mathcal{F}_{\mathsf{Ty}}(A)$ in the target model, there is a term $\Gamma \vdash_{\mathcal{M}} a : A$ whose image by $\mathcal{F}$ is identified with b.

We apply the notion of weak equivalence to "syntactic" models of $\mathbb{M}_{\mathrm{LTT}_{\Sigma,\mathsf{Id}}}$, coming from extensions of the SOGAT of $\mathbb{M}_{\mathrm{LTT}_{\Sigma,\mathsf{Id}}}$, in order to speak about conservativity relations between type theories (cf. for example Isaev [20], Bocquet [7], Kapulkin and Li [21]).

▶ **Definition 13.** *For an RMC functor* $F : \mathbb{R} \to \mathbb{S}$*, we write* $\mathbf{0}_F \coloneqq (\mathbb{R}, \text{ɬ} \circ F)^{\heartsuit} \in \mathbf{Mod}(\mathbb{R})$ *for the heart of the model of* $\mathbb{R}$ *given by the RMC functor* $\mathbb{R} \xrightarrow{F} \mathbb{S} \xrightarrow{\text{ɬ}} \mathrm{PSh}(\mathbb{S})$*. When* F *is understood from context, we write* $\mathbf{0}_{\mathbb{S}} \in \mathbf{Mod}(\mathbb{R})$.

A morphism $G\colon (\mathbb{S}, F) \to (\mathbb{S}', F')$ in the coslice (2,1)-category $\mathbb{R}/\mathbf{RMC}$ induces a morphism $\mathbf{0}_G\colon \mathbf{0}_F \to \mathbf{0}_{F'}$ of models of $\mathbb{R}$. A special case of the above construction is its application to the identity $\mathrm{Id}\colon \mathbb{R} \to \mathbb{R}$: the model $\mathbf{0}_{\mathbb{R}} \in \mathbf{Mod}(\mathbb{R})$ is a bi-initial object in $\mathbf{Mod}(\mathbb{R})$, the *initial model* of $\mathbb{R}$ [37, §5.4.1].

3 Cubical type theories

3.1 The interval

Before defining cubical type theory, we first introduce a simple SOGAT specifying the interval alone. This will let us easily speak about cubical type theories with different interval theories.

▶ **Definition 14.** *The SOGAT* $\mathbb{I}\textsc{nt}$ *of an* interval *has one representable sort with two points:*

$$\mathbb{I} \;:\; () \Rightarrow \star \qquad\qquad \mathsf{0}, \mathsf{1} \;:\; () \Rightarrow \mathbb{I}$$

▶ **Definition 15.** *An* interval theory *is an environment* $\Phi \in \mathbb{I}\textsc{nt}$.

Per § 2.1, a context in $\mathbb{I}\textsc{nt}$ is simply a list $(\mathtt{i}_1 : \mathbb{I}, \ldots, \mathtt{i}_n : \mathbb{I})$. An environment Φ consists of declarations of the form $\mathtt{r} : \Gamma \to \mathbb{I}$ and $\_ : \Gamma \to r_1 \equiv r_2 : \mathbb{I}$. In other words, Φ specifies a single-sorted algebraic theory extending the theory of two points $\mathsf{0}$, $\mathsf{1}$.

▶ **Example 16.** The *cartesian interval theory* Φ_{cart} is the trivial environment $1 \coloneqq () \in \mathbb{I}\textsc{nt}$. The *distributive lattice interval theory* Φ_{DL} is the environment beginning with

$$\begin{array}{lcl} (-\wedge-), (-\vee-) & : & (\mathtt{i} : \mathbb{I}, \mathtt{j} : \mathbb{I}) \to \mathbb{I} \\ \_ & : & (\mathtt{i}\ \mathtt{j}\ \mathtt{k} : \mathbb{I}) \to \mathtt{i} \wedge (\mathtt{j} \vee \mathtt{k}) \equiv (\mathtt{i} \wedge \mathtt{j}) \vee (\mathtt{i} \wedge \mathtt{k}) : \mathbb{I} \end{array}$$

and continuing with the other equations of a bounded distributive lattice, as enumerated for example by Buchholtz and Morehouse [8, Table 1]: associativity and commutativity of $\wedge$ and $\vee$, unit laws $\mathtt{i} \wedge \mathsf{1} \equiv \mathtt{i}$ and $\mathtt{i} \vee \mathsf{0} \equiv \mathtt{i}$, and absorption laws $\mathtt{i} \wedge (\mathtt{i} \vee \mathtt{j}) \equiv \mathtt{i}$ and $\mathtt{i} \vee (\mathtt{i} \wedge \mathtt{j}) \equiv \mathtt{i}$.

3.2 Cofibrations

In addition to Ty, Tm, and $\mathbb{I}$, cubical type theory has a sort of *cofibrations* and, over the sort of cofibrations, a representable sort for *cofibration truth*:

$$\mathsf{Cof} \;:\; \square \qquad\qquad \mathsf{True} \;:\; (\mathsf{P} : \mathsf{Cof}) \Rightarrow \star$$

We write $\mathbb{C}\textsc{of}$ for the SOGAT consisting of these two sorts. In fact we have $\mathbb{C}\textsc{of} \simeq \mathbb{M}\textsc{ltt}$, but it will be useful to have distinct notation for this sub-SOGAT of our cubical type theories.

3.3 Opaque cubical type theory

We define opaque cubical type theory, $\mathbb{C}\textsc{tt}$, as a mutual extension of the SOGATs of Martin-Löf type theory with Σ, Π, and identity types ($\mathbb{M}\textsc{ltt}_{\Sigma,\mathsf{Id},\Pi}$), an interval ($\mathbb{I}\textsc{nt}$), and a cofibration classifier ($\mathbb{C}\textsc{of}$). We roughly follow Uemura's encodings of cubical type theory [37, §4.6.3] [38, Example 5.14]. We introduce the declarations of $\mathbb{C}\textsc{tt}$ (beyond those of $\mathbb{M}\textsc{ltt}_{\Sigma,\mathsf{Id},\Pi}$, $\mathbb{I}\textsc{nt}$, and $\mathbb{C}\textsc{of}$) in stages over the course of this section (§ 3.3).

▶ Remark 17. Given that path types serve as equality types in cubical type theories, it may seem strange that we include Martin-Löf's identity types in $\mathbb{C}\textsc{tt}$, though their coexistence is semantically justified [12, §9.1] [9, §3.3] [2, §2.16]. We do so partly in order to reuse Kapulkin

and Lumsdaine's tools for comparing type theories [22], though we could have redeveloped these with path types. The more technical reason is that we want to include (higher) inductive types in $\mathbb{C}\textsc{tt}$. The span interpretation (§ 5) that we use to prove conservativity interprets inductive types as inductive families [15], and we use identity types to define these families. This is the one place where identities cannot straightforwardly be replaced with paths.

3.3.1 Cofibrations

In $\mathbb{C}\textsc{tt}$, we add new operators and equations for the sorts of $\mathbb{C}\textsc{of}$. A cofibration can be thought of as a constraint on interval terms. Cofibration truth is a *strict proposition* in the sense that any two witnesses to truth of a cofibration are strictly equal:

$$\_ \;:\; (\mathtt{P} : \mathsf{Cof}, \mathtt{u}\ \mathtt{v} : \mathsf{True}(\mathtt{P})) \Rightarrow \mathtt{u} \equiv \mathtt{v} : \mathsf{True}(\mathtt{P})$$

As with Tm, we will leave the True operator implicit. Cofibrations are closed under finite conjunction ($\top$, $\cap$) and disjunction ($\bot$, $\cup$):

$$\begin{array}{lcl@{\qquad}lcl}
\top, \bot & : & \mathsf{Cof} & _ & : & (\mathtt{P}\ \mathtt{Q} : \mathsf{Cof}, \mathtt{P}, \mathtt{Q}) \Rightarrow \mathtt{P} \cap \mathtt{Q} \\
(- \cap -) & : & (\mathtt{P}\ \mathtt{Q} : \mathsf{Cof}) \Rightarrow \mathsf{Cof} & _ & : & (\mathtt{P}\ \mathtt{Q} : \mathsf{Cof}, \mathtt{P} \cap \mathtt{Q}) \Rightarrow \mathtt{P} \\
(- \cup -) & : & (\mathtt{P}\ \mathtt{Q} : \mathsf{Cof}) \Rightarrow \mathsf{Cof} & _ & : & (\mathtt{P}\ \mathtt{Q} : \mathsf{Cof}, \mathtt{P} \cap \mathtt{Q}) \Rightarrow \mathtt{Q} \\
_ & : & \top & _ & : & (\mathtt{P}\ \mathtt{Q} : \mathsf{Cof}, \mathtt{P}) \Rightarrow \mathtt{P} \cup \mathtt{Q} \\
_ & : & (\mathtt{P} : \mathsf{Cof}, \bot) \Rightarrow \mathtt{P} & _ & : & (\mathtt{P}\ \mathtt{Q} : \mathsf{Cof}, \mathtt{Q}) \Rightarrow \mathtt{P} \cup \mathtt{Q} \\
 & & & _ & : & (\mathtt{P}\ \mathtt{Q}\ \mathtt{R} : \mathsf{Cof}, \mathtt{P} \to \mathtt{R}, \mathtt{Q} \to \mathtt{R}, \mathtt{P} \cup \mathtt{Q}) \Rightarrow \mathtt{R}
\end{array}$$

Eliminators for the nullary and binary disjunction ($\mathsf{elim}^{\mathsf{Ty}}_{\bot}$, $\mathsf{elim}^{\mathsf{Tm}}_{\bot}$, $\mathsf{elim}^{\mathsf{Ty}}_{\cup}$, $\mathsf{elim}^{\mathsf{Tm}}_{\cup}$) allow us to define types and terms by case analysis. We abbreviate

$$\begin{array}{lcl}
\Phi_{\cup\mathsf{Ty}} & = & (\mathtt{P}\ \mathtt{Q} : \mathsf{Cof}, \mathtt{A} : [\mathtt{P}] \to \mathsf{Ty}, \mathtt{B} : [\mathtt{Q}] \to \mathsf{Ty}, [\mathtt{P} \cap \mathtt{Q} \to \mathtt{A} \equiv \mathtt{B} : \mathsf{Ty}]) \\
\Phi_{\cup\mathsf{Tm}} & = & (\mathtt{P}\ \mathtt{Q} : \mathsf{Cof}, \mathtt{A} : [\mathtt{P} \cup \mathtt{Q}] \to \mathsf{Ty}, \mathtt{a} : [\mathtt{P}] \to \mathtt{A}, \mathtt{b} : [\mathtt{Q}] \to \mathtt{A}, [\mathtt{P} \cap \mathtt{Q} \to \mathtt{a} \equiv \mathtt{b} : \mathtt{A}])
\end{array}$$

and specify

$$\begin{array}{lcl}
\mathsf{elim}^{\mathsf{Ty}}_{\bot} & : & [\bot] \Rightarrow \mathsf{Ty} \\
\mathsf{elim}^{\mathsf{Tm}}_{\bot} & : & (\mathtt{A} : [\bot] \to \mathsf{Ty}, [\bot]) \Rightarrow \mathtt{A} \\
\mathsf{elim}^{\mathsf{Ty}}_{\cup} & : & (\Phi_{\cup\mathsf{Ty}}, [\mathtt{P} \cup \mathtt{Q}]) \Rightarrow \mathsf{Ty} \\
_ & : & (\Phi_{\cup\mathsf{Ty}}, \mathtt{P}) \Rightarrow \mathsf{elim}^{\mathsf{Ty}}_{\cup}(\mathtt{P}, \mathtt{Q}, \mathtt{A}, \mathtt{B}) \equiv \mathtt{A} : \mathsf{Ty} \\
_ & : & (\Phi_{\cup\mathsf{Ty}}, \mathtt{Q}) \Rightarrow \mathsf{elim}^{\mathsf{Ty}}_{\cup}(\mathtt{P}, \mathtt{Q}, \mathtt{A}, \mathtt{B}) \equiv \mathtt{B} : \mathsf{Ty} \\
\mathsf{elim}^{\mathsf{Tm}}_{\cup} & : & (\Phi_{\cup\mathsf{Tm}}, [\mathtt{P} \cup \mathtt{Q}]) \Rightarrow \mathtt{A} \\
_ & : & (\Phi_{\cup\mathsf{Tm}}, \mathtt{P}) \Rightarrow \mathsf{elim}^{\mathsf{Tm}}_{\cup}(\mathtt{P}, \mathtt{Q}, \mathtt{A}, \mathtt{a}, \mathtt{b}) \equiv \mathtt{a} : \mathtt{A} \\
_ & : & (\Phi_{\cup\mathsf{Tm}}, \mathtt{Q}) \Rightarrow \mathsf{elim}^{\mathsf{Tm}}_{\cup}(\mathtt{P}, \mathtt{Q}, \mathtt{A}, \mathtt{a}, \mathtt{b}) \equiv \mathtt{b} : \mathtt{A}
\end{array}$$

The basic cofibrations are equations on interval terms, which we write with $\approx$. The two endpoints 0 and 1 are distinct, and we can convert between $\approx$ and strict equality $\equiv$.

$$\begin{array}{lcl@{\qquad}lcl}
- \approx - & : & (\mathtt{i}\ \mathtt{j} : \mathbb{I}) \Rightarrow \mathsf{Cof} & _ & : & (\mathtt{i} : \mathbb{I}) \Rightarrow \mathtt{i} \approx \mathtt{i} \\
_ & : & (0 \approx 1) \Rightarrow \bot & _ & : & (\mathtt{i}\ \mathtt{j} : \mathbb{I}, \mathtt{i} \approx \mathtt{j}) \Rightarrow \mathtt{i} \equiv \mathtt{j} : \mathbb{I}
\end{array}$$

▶ Remark 18. We could have included various algebraic laws for cofibrations, such as $\mathtt{P} \cap \mathtt{Q} \equiv \mathtt{Q} \cap \mathtt{P}$, or *cofibration extensionality* $(\mathtt{P} : \mathsf{Cof}, \mathtt{Q} : \mathsf{Cof}, \mathtt{P} \to \mathtt{Q}, \mathtt{Q} \to \mathtt{P}) \to \mathtt{P} \equiv \mathtt{Q} : \mathsf{Cof}$. Our proofs go through for such variations without much change.

3.3.2 Filling

Cofibrations are used to specify the *filling operator*. We first introduce the abbreviation Φ_{fill} for the environment

$$(\mathtt{A} : \mathbb{I} \to \mathsf{Ty}, \mathtt{P} : \mathsf{Cof}, \mathtt{a} : ([\mathtt{P}], \mathtt{i} : \mathbb{I}) \to \mathtt{A}(\mathtt{i}), \mathtt{j} : \mathbb{I}, \mathtt{a}_0 : \mathtt{A}(\mathtt{j}), [\mathtt{P} \to \mathtt{a}(\mathtt{j}) \equiv \mathtt{a}_0 : \mathtt{A}(\mathtt{j})]).$$

This environment specifies a line ($\mathbb{I}$-indexed family) of types $\mathtt{A}$ and a "partial" line of terms $\mathtt{a}$ over it, defined whenever some cofibration $\mathtt{P}$ is true, together with a fully-defined term $\mathtt{a}_0$ at some index $\mathtt{A}(\mathtt{j})$ that coincides with $\mathtt{a}(\mathtt{j})$ when $\mathtt{P}$ holds. Given this input, the filling operator outputs a line $(\mathtt{k} : \mathbb{I}) \to \mathtt{A}(\mathtt{k})$ that "extends" both $\mathtt{a}$ and $\mathtt{a}_0$ in the following sense.

$$\begin{array}{lcl} \mathsf{fill} & : & (\Phi_{\mathsf{fill}}, \mathtt{k} : \mathbb{I}) \Rightarrow \mathtt{A}(\mathtt{k}) \\ \_ & : & (\Phi_{\mathsf{fill}}, \mathtt{k} : \mathbb{I}, \mathtt{P}) \Rightarrow \mathsf{fill}(\mathtt{A}, \mathtt{P}, \mathtt{a}, \mathtt{j}, \mathtt{a}_0, \mathtt{k}) \equiv \mathtt{a}(\mathtt{k}) : \mathtt{A}(\mathtt{k}) \\ \_ & : & (\Phi_{\mathsf{fill}}) \Rightarrow \mathsf{fill}(\mathtt{A}, \mathtt{P}, \mathtt{a}, \mathtt{j}, \mathtt{a}_0, \mathtt{j}) \equiv \mathtt{a}_0 : \mathtt{A}(\mathtt{j}) \end{array}$$

The special case where $\mathtt{P} = \bot$ is called *coercion* by Angiuli et al. [2, §2.7] and converts a term at some index $\mathtt{a}_0 : \mathtt{A}(\mathtt{j})$ to a term at any other index $\mathtt{A}(\mathtt{k})$.

▶ **Notation 19.** *Over the environment* $(\mathtt{A} : (\mathtt{i} : \mathbb{I}) \to \mathsf{Ty}, \mathtt{j} : \mathbb{I}, \mathtt{a}_0 : \mathtt{A}(\mathtt{j}), \mathtt{k} : \mathbb{I})$*, write* $\mathsf{coe}(\mathtt{A}, \mathtt{j}, \mathtt{a}_0, \mathtt{k}) \coloneqq \mathsf{fill}(\mathtt{A}, \bot, \langle \mathtt{i} \rangle \mathsf{elim}_{\bot}^{\mathsf{Tm}}(\mathtt{A}(\mathtt{i})), \mathtt{j}, \mathtt{a}_0, \mathtt{k}) : \mathtt{A}(\mathtt{k})$*.*

▶ **Notation 20.** *We write* $\mathsf{fill}^{j \to k}(A, [P_1 \mapsto a_1, \ldots, P_n \mapsto a_n], a_0)$ *for* $\mathsf{fill}(A, P, a, j, a_0, k)$ *where* $P = P_1 \cup \cdots \cup P_n$ *(with some choice of parentheses) and* a *is defined from* $a_1, \ldots, a_n$ *by cases using* $\mathsf{elim}_{\cup}^{\mathsf{Tm}}$*. We write* $\mathsf{coe}^{j \to k}(A, a_0)$ *for* $\mathsf{coe}(A, j, a_0, k)$*.*

▶ Remark 21. This definition of fill is a suitable base for strict cubical type theory over arbitrary interval theories. In the presence of certain interval structure, it can be reduced to special cases. For theories with connections, $\mathsf{fill}^{0 \to 1}$ and $\mathsf{fill}^{1 \to 0}$ suffice; see Cavallo, Mörtberg, and Swan [10, Theorem 14 with Lemma 8]. With two connections and a reversal, this can be further reduced to $\mathsf{fill}^{0 \to 1}$, as in Cohen et al.'s type theory [12]; see Angiuli et al. [2, §3.4]. We refer to Cavallo, Mörtberg, and Swan [10] for more detailed comparisons.

3.3.3 Paths

A path is an $\mathbb{I}$-indexed term taking two fixed values at the endpoints $\mathsf{0}, \mathsf{1} : \mathbb{I}$. Path types internalize paths:

$$\begin{array}{lcl} \mathsf{Path} & : & (\mathtt{A} : (\mathtt{i} : \mathbb{I}) \to \mathsf{Ty}, \mathtt{a}_0 : \mathtt{A}(\mathsf{0}), \mathtt{a}_1 : \mathtt{A}(\mathsf{1})) \Rightarrow \mathsf{Ty} \\ \lambda^{\mathbb{I}} & : & ([\mathtt{A} : (\mathtt{i} : \mathbb{I}) \to \mathsf{Ty}], \mathtt{a} : (\mathtt{i} : \mathbb{I}) \to \mathtt{A}(i)) \Rightarrow \mathsf{Path}(\mathtt{A}, \mathtt{a}(\mathsf{0}), \mathtt{a}(\mathsf{1})) \\ - \mathbin{@} - & : & ([\mathtt{A} : (\mathtt{i} : \mathbb{I}) \to \mathsf{Ty}, \mathtt{a}_0 : \mathtt{A}(\mathsf{0}), \mathtt{a}_1 : \mathtt{A}(\mathsf{1})], \mathtt{p} : \mathsf{Path}(\mathtt{A}, \mathtt{a}_0, \mathtt{a}_1), \mathtt{i} : \mathbb{I}) \Rightarrow \mathtt{A}(\mathtt{i}) \end{array}$$

The equations for path types state that $\lambda^{\mathbb{I}}(\mathtt{a}) \mathbin{@} \mathtt{i} \equiv \mathtt{a}(\mathtt{i})$ and $\mathtt{p} \equiv \lambda^{\mathbb{I}}(\langle \mathtt{i} \rangle \mathtt{p} \mathbin{@} \mathtt{i})$, as for function types, as well as that $\mathtt{p} \mathbin{@} \mathsf{0} \equiv \mathtt{a}_0$ and $\mathtt{p} \mathbin{@} \mathsf{1} \equiv \mathtt{a}_1$ for $\mathtt{p} : \mathsf{Path}(\mathtt{A}, \mathtt{a}(\mathsf{0}), \mathtt{a}(\mathsf{1}))$. See Uemura [37, §4.6.3, Type constructors] for a fully formal presentation.

▶ **Notation 22.** *We write* $\lambda \mathtt{i}.a$ *as shorthand for* $\lambda^{\mathbb{I}} \langle \mathtt{i} \rangle a$*. We abbreviate "non-dependent" path types* $\mathsf{Path}(\langle \_ \rangle A, a_0, a_1)$*, where the line of types is constant, as* $a_0 \sim^A a_1$ *or simply* $a_0 \sim a_1$*.*

3.3.4 Glue types

In cubical type theories, univalence is not an axiom but is instead derived from a type former that can construct $\mathbb{I}$-indexed types from equivalences. Universes closed under this type former can be shown to be univalent. Following Cohen et al. [12], Angiuli et al. implement univalence using so-called glue types [2, §2.11]; Angiuli, Favonia, and Harper's V-types are an alternative solution [3, §5.6].

First, we define equivalences [39, Definition 4.4.1] using path types. Over $\mathtt{A} : \mathsf{Ty}$, define $\mathsf{isContr}(\mathtt{A}) \coloneqq \Sigma \mathtt{a}_0{:}\mathtt{A}.\ \Pi \mathtt{a}_1{:}\mathtt{A}.\ \mathtt{a}_0 \sim^{\mathtt{A}} \mathtt{a}_1 : \mathsf{Ty}$. Over $([\mathtt{A}\ \mathtt{B} : \mathsf{Ty}], \mathtt{f} : \mathtt{A} \to \mathtt{B})$, define $\mathsf{isEquiv}(\mathtt{f}) \coloneqq \Pi \mathtt{b} : \mathtt{B}.\mathsf{isContr}(\Sigma \mathtt{a} : \mathtt{A}.\ \mathtt{f}(\mathtt{a}) \sim \mathtt{b}) : \mathsf{Ty}$. Finally, over $(\mathtt{A}\ \mathtt{B} : \mathsf{Ty})$, define the type of *equivalences* $(\mathtt{A} \simeq \mathtt{B}) \coloneqq \Sigma \mathtt{f} : \mathtt{A} \to \mathtt{B}.\ \mathsf{isEquiv}(\mathtt{f})$. The Glue type former takes a type $\mathtt{A}$, a cofibration $\mathtt{P}$, and a partial type $\mathtt{T}$ and equivalence $\mathtt{e} : \mathtt{T} \simeq \mathtt{A}$ defined when $\mathtt{P}$ holds. Its output is a total type that reduces to $\mathtt{T}$ when $\mathtt{P}$ holds.

$$\begin{array}{lcl} \mathsf{Glue} & : & (\mathtt{A} : \mathsf{Ty}, \mathtt{P} : \mathsf{Cof}, \mathtt{T} : [\mathtt{P}] \to \mathsf{Ty}, \mathtt{e} : [\mathtt{P}] \to \mathtt{T} \simeq \mathtt{A}) \Rightarrow \mathsf{Ty} \\ \_ & : & (\mathtt{A} : \mathsf{Ty}, \mathtt{P} : \mathsf{Cof}, \mathtt{T} : [\mathtt{P}] \to \mathsf{Ty}, \mathtt{e} : [\mathtt{P}] \to \mathtt{T} \simeq \mathtt{A}, \mathtt{P}) \Rightarrow \mathsf{Glue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \mathtt{e}) \equiv \mathtt{T} : \mathsf{Ty} \end{array}$$

We now abbreviate $\Phi_{\mathsf{Glue}} = (\mathtt{A} : \mathsf{Ty}, \mathtt{P} : \mathsf{Cof}, \mathtt{T} : [\mathtt{P}] \to \mathsf{Ty}, \mathtt{e} : [\mathtt{P}] \to \mathtt{T} \simeq \mathtt{A})$. The Glue type has an introduction form glue and an elimination form unglue. Each reduces when $\mathtt{P}$ holds, and we have computation and uniqueness equations.

$$\begin{array}{lcl} \mathsf{glue} & : & ([\Phi_{\mathsf{Glue}}], \mathtt{a} : \mathtt{A}, \mathtt{t} : [\mathtt{P}] \to \mathtt{T}, [\mathtt{P} \to \mathtt{e}.1(\mathtt{t}) \equiv \mathtt{a} : \mathtt{A}]) \Rightarrow \mathsf{Glue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \mathtt{e}) \\ \_ & : & ([\Phi_{\mathsf{Glue}}], \mathtt{a} : \mathtt{A}, \mathtt{t} : [\mathtt{P}] \to \mathtt{T}, [\mathtt{P} \to \mathtt{e}.1(\mathtt{t}) \equiv \mathtt{a} : \mathtt{A}], \mathtt{P}) \Rightarrow \mathsf{glue}(\mathtt{a}, \mathtt{t}) \equiv \mathtt{t} : \mathtt{T} \\ \mathsf{unglue} & : & ([\Phi_{\mathsf{Glue}}], \mathtt{g} : \mathsf{Glue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \mathtt{e})) \Rightarrow \mathtt{A} \\ \_ & : & ([\Phi_{\mathsf{Glue}}], \mathtt{g} : \mathsf{Glue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \mathtt{e}), \mathtt{P}) \Rightarrow \mathsf{unglue}(\mathtt{g}) \equiv \mathtt{e}.1(\mathtt{g}) : \mathtt{A} \\ \_ & : & ([\Phi_{\mathsf{Glue}}], \mathtt{a} : \mathtt{A}, \mathtt{t} : [\mathtt{P}] \to \mathtt{T}, [\mathtt{P} \to \mathtt{e}.1(\mathtt{t}) \equiv \mathtt{a} : \mathtt{A}]) \Rightarrow \mathsf{unglue}(\mathsf{glue}(\mathtt{a}, \mathtt{t})) \equiv \mathtt{a} : \mathtt{A} \\ \_ & : & ([\Phi_{\mathsf{Glue}}], \mathtt{g} : \mathsf{Glue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \mathtt{e})) \Rightarrow \mathtt{g} \equiv \mathsf{glue}(\mathsf{unglue}(\mathtt{g}), \mathtt{g}) : \mathsf{Glue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \mathtt{e}) \end{array}$$

The eliminator unglue can be shown to be an equivalence $\mathsf{Glue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \mathtt{e}) \simeq \mathtt{A}$ that reduces to $\mathtt{e}$ when $\mathtt{P}$ holds. Univalence is derived using an instance where $\mathtt{P}$ is $(\mathtt{i} \approx 0 \cup \mathtt{i} \approx 1)$ for some $\mathtt{i} : \mathbb{I}$; see Cohen et al. [12, §7.2] or Angiuli et al. [2, §2.12].

3.3.5 Universe

We include one universe: a type U whose elements are regarded as types via a decoding function El.

$$\mathsf{U} \ : \ \mathsf{Ty} \qquad\qquad \mathsf{El} \ : \ (\mathtt{A} : \mathsf{U}) \Rightarrow \mathsf{Ty}$$

We often leave the coercion El from U to types implicit. For a universe to be useful, it should be closed under type formers such as Σ and Π; for it to be univalent, it should be closed under Glue. We refer to Uemura [37, Example 4.6.11] for an example formulation of these closure conditions, but we omit cases for type formers in the universe in our proofs: handling them always amounts to repeating the construction used for the type formers outside of the universe.

3.3.6 Higher inductive types

We include *suspension* types [39, §6.5] as a representative example of an HIT. See [13, 9] for general descriptions of HITs in cubical type theory. We specify the formation and introduction forms as follows:

$$\begin{array}{lcllcl} \mathsf{Susp} & : & (\mathtt{A} : \mathsf{Ty}) \Rightarrow \mathsf{Ty} & \mathsf{north}, \mathsf{south} & : & [\mathtt{A} : \mathsf{Ty}] \Rightarrow \mathsf{Susp}(\mathtt{A}) \\ & & & \mathsf{merid} & : & ([\mathtt{A} : \mathsf{Ty}], \mathtt{a} : \mathtt{A}) \Rightarrow \mathsf{north} \sim^{\mathsf{Susp}(\mathtt{A})} \mathsf{south} \end{array}$$

For elimination, we fix the environment

$$\begin{array}{lcl} \Phi_{\mathsf{elim}} & = & ([\mathtt{A} : \mathsf{Ty}], \mathtt{C} : (\mathtt{t} : \mathsf{Susp}(\mathtt{A})) \to \mathsf{Ty}, \\ & & \mathtt{n} : \mathtt{C}(\mathsf{north}), \mathtt{s} : \mathtt{C}(\mathsf{south}), \mathtt{m} : (\mathtt{a} : \mathtt{A}) \to \mathsf{Path}(\langle \mathtt{i} \rangle \mathtt{C}(\mathsf{merid}(\mathtt{a})\ @\ \mathtt{i}), \mathtt{n}, \mathtt{s})) \end{array}$$

and specify

$$\begin{array}{lcl}
\mathsf{elim} & : & (\Phi_{\mathsf{elim}}, \mathtt{t} : \mathsf{Susp}(\mathtt{A})) \Rightarrow \mathtt{C}(\mathtt{t}) \\
_ & : & (\Phi_{\mathsf{elim}}) \Rightarrow \mathsf{elim}(\mathtt{C}, \mathtt{n}, \mathtt{s}, \mathtt{m}, \mathsf{north}) \equiv \mathtt{n} : \mathtt{C}(\mathsf{north}) \\
_ & : & (\Phi_{\mathsf{elim}}) \Rightarrow \mathsf{elim}(\mathtt{C}, \mathtt{n}, \mathtt{s}, \mathtt{m}, \mathsf{south}) \equiv \mathtt{s} : \mathtt{C}(\mathsf{south}) \\
\mathsf{merid}\beta & : & (\Phi_{\mathsf{elim}}, \mathtt{a} : \mathtt{A}) \Rightarrow \lambda \mathtt{i}.\mathsf{elim}(\mathtt{C}, \mathtt{n}, \mathtt{s}, \mathtt{m}, \mathsf{merid}(\mathtt{a})\ @\ \mathtt{i}) \sim \mathtt{m}
\end{array}$$

This is an "opaque" suspension type in that $\mathsf{merid}\beta$ constructor is a path rather than a strict equality. This is how HITs are usually formulated in Book HoTT [39, §6.2], strict computation rules being characteristic of cubical type theory.

3.4 Strict cubical type theory

Strict cubical type theories – i.e., cubical type theories as they are usually defined – are designed to satisfy strict canonicity [19, 3], the property that every closed term of type $\mathbb{N}$ is strictly equal to a numeral. This requires two adjustments to our opaque cubical type theory, which we sketch here. A full description of the specific strict theory we model in § 7 can be found in Angiuli et al. [2]. We write $\mathbb{C}\mathrm{TT}_s$ for the extension of the SOGAT $\mathbb{C}\mathrm{TT}$ with the symbols and equations indicated below, following Angiuli et al.'s specification.

First, we add equations for each concrete type former for evaluating applications of the filling operator at that type. For Σ types, for example, we have an equation reducing $\mathsf{fill}(\langle \mathtt{i} \rangle \Sigma \mathtt{a}{:}\mathtt{A}(\mathtt{i}).\mathtt{B}(\mathtt{i}, \mathtt{a}), \mathtt{P}, \mathtt{s}, \mathtt{j}, \mathtt{s}_0, \mathtt{k})$ to a pair of two calls to the filling operator, one over $\mathtt{A}$ and one over an instance of $\mathtt{B}$. For higher inductive types such as Susp, some applications of the filling operator are treated as values (i.e., not reduced), and equations are instead introduced for reducing the eliminator at these values [2, §2.15].

Second, we strictify the path $\mathsf{merid}\beta$, replacing it with a strict equation or, to express strict cubical type theory as an extension of opaque cubical type theory, introducing the strict equation and equating $\mathsf{merid}\beta$ with the reflexive path.

4 The twist interpretation

To prove conservativity of opaque cubical type theories with reversals over the corresponding theories without reversals, we first construct interpretations from the former to the latter. In §§ 5–6 we show that the existence of these interpretations abstractly implies conservativity. As sketched in § 1.1, we exploit *twist constructions*: the fact that the "square" environment $\mathbb{I} \times \mathbb{I} = (\mathtt{i}_0 : \mathbb{I}, \mathtt{i}_1 : \mathbb{I}) \in \mathbb{C}\mathrm{TT}$ is an interval object with a reversal and inherits certain algebraic structure from $\mathbb{I}$. Thus, we call our translation the *twist interpretation*.

4.1 Extension by a reversal

We have an interpretation Flip of $\mathbb{I}\mathrm{NT}$ in itself by taking $\mathrm{Flip}(\mathbb{I}) \coloneqq \mathbb{I}$, $\mathrm{Flip}(0) \coloneqq 1$, and $\mathrm{Flip}(1) \coloneqq 0$. By the (2, 1)-categorical universal property of $\mathbb{C}\mathrm{L}(\mathbb{I}\mathrm{NT})$ [37, Theorem 4.8.18], this determines (up to isomorphism) an RMC functor $\mathrm{Flip} \colon \mathbb{I}\mathrm{NT} \to \mathbb{I}\mathrm{NT}$ with an isomorphism $\theta \colon \mathrm{Flip} \circ \mathrm{Flip} \cong \mathrm{Id}$ satisfying $\theta \circ \mathrm{Flip} = \mathrm{Flip} \circ \theta$ and $\theta_{\mathbb{I}} = \mathrm{id}$.

▶ **Definition 23.** *A* self-dual interval theory (Φ, ϕ) *is an interval theory* Φ *equipped with an isomorphism* $\phi \colon \mathrm{Flip}(\Phi) \cong \Phi$ *such that* $\mathrm{Flip}(\phi) \circ \phi = \theta_\Phi : \mathrm{Flip}(\mathrm{Flip}(\Phi)) \cong \Phi$.

In a self-dual interval theory (Φ, ϕ), the value of ϕ at an operator $\mathtt{r} : \mathbb{I}^n \to \mathbb{I}$ in Φ is an expression $\phi(\mathtt{r}) : \mathbb{I}^n \to \mathbb{I}$ over Φ: the dual of $\mathtt{r}$.

▶ **Example 24.** The cartesian theory Φ_{cart} is self-dual with the trivial isomorphism $1 \cong 1$. The theory Φ_{DL} is self-dual with ϕ defined by $\phi(- \wedge -)(\mathtt{i}, \mathtt{j}) = \mathtt{i} \vee \mathtt{j}$ and vice versa.

▶ **Definition 25.** *Given a self-dual interval theory* (Φ, ϕ)*, its* extension by a reversal $\mathrm{Rev}_\phi\Phi \in \mathbb{I}\textsc{nt}$ *is the extension of* Φ *with*

(a) *an operator* $\neg : \mathbb{I} \to \mathbb{I}$*,*
(b) *equations* $\neg 0 \equiv 1 : \mathbb{I}$*,* $\neg 1 \equiv 0 : \mathbb{I}$*, and* $(\mathtt{i} : \mathbb{I}) \to \neg(\neg(\mathtt{i})) \equiv \mathtt{i} : \mathbb{I}$*, and*
(c) *for each* $\mathtt{r} : (\mathtt{i}_1 : \mathbb{I}, \ldots, \mathtt{i}_n : \mathbb{I}) \to \mathbb{I}$ *in* Φ*, an equation*

$$(\mathtt{i}_1 : \mathbb{I}, \ldots, \mathtt{i}_n : \mathbb{I}) \to \neg(\mathtt{r}(\mathtt{i}_1, \ldots, \mathtt{i}_n)) \equiv \phi(\mathtt{r})(\neg(\mathtt{i}_1), \ldots, \neg(\mathtt{i}_n)) : \mathbb{I}.$$

▶ **Example 26.** The interval theory $\mathrm{Rev}_\phi\Phi_{\text{cart}}$ for $\phi\colon 1 \cong 1$ consists simply of the operator $\neg : \mathbb{I} \to \mathbb{I}$ and equations $\neg 0 \equiv 1 : \mathbb{I}$, $\neg 1 \equiv 0 : \mathbb{I}$, and $(\mathtt{i} : \mathbb{I}) \to \neg(\neg(\mathtt{i})) \equiv \mathtt{i} : \mathbb{I}$. The interval theory $\mathrm{Rev}_\phi\Phi_{\text{DL}}$, for the isomorphism $\phi\colon \Phi_{\text{DL}} \cong \Phi_{\text{DL}}$ from Example 24, is the algebraic theory of a De Morgan algebra bounded by 0 and 1.

▶ **Definition 27** (Twist interpretation of the interval)**.** *For a self-dual interval theory* (Φ, ϕ)*, we define a representable map functor* $\mathrm{T}\colon \mathbb{I}\textsc{nt}[\mathrm{Rev}_\phi\Phi] \to \mathbb{I}\textsc{nt}[\Phi]$ *by the following interpretation:*

1. *On sorts, we set* $\mathrm{T}\mathbb{I} \coloneqq \mathbb{I} \times \mathbb{I}$*.*
2. *We interpret* 0 *as* $(0, 1)$ *and* 1 *as* $(1, 0)$*.*
3. *We interpret each* $\mathtt{r} : (\mathtt{i}_1 : \mathbb{I}, \ldots, \mathtt{i}_n : \mathbb{I}) \to \mathbb{I}$ *in* Φ *by*

$$\mathrm{T}\mathtt{r}((\mathtt{i}_{10}, \mathtt{i}_{11}), \ldots, (\mathtt{i}_{n0}, \mathtt{i}_{n1})) \coloneqq (\mathtt{r}(\mathtt{i}_{10}, \ldots, \mathtt{i}_{n0}), \phi(\mathtt{r})(\mathtt{i}_{11}, \ldots, \mathtt{i}_{n1})).$$

4. *We interpret* $\neg$ *by* $\mathrm{T}\neg((\mathtt{i}_0, \mathtt{i}_1)) \coloneqq (\mathtt{i}_1, \mathtt{i}_0)$*.*

4.2 Interpreting cubical type theory

Cubical type theory being an extension of the theory of an interval, any environment Φ over $\mathbb{I}\textsc{nt}$ can also be regarded as an environment $\iota\Phi$ over $\mathbb{C}\textsc{tt}$, from which we can produce a new SOGAT $\mathbb{C}\textsc{tt}[\iota\Phi]$: cubical type theory with the interval theory Φ.

We now extend $\mathrm{T}\colon \mathbb{I}\textsc{nt}[\mathrm{Rev}_\phi\Phi] \to \mathbb{I}\textsc{nt}[\Phi]$ for a self-dual interval theory (Φ, ϕ) to an interpretation $\mathrm{T}\colon \mathbb{C}\textsc{tt}[\iota\mathrm{Rev}_\phi\Phi] \to \mathbb{C}\textsc{tt}[\iota\Phi]$. The specification of this interpretation occupies the remainder of this section; we summarize in Theorem 42.

▶ **Component 28** (T, sorts)**.** *We set* $\mathrm{T}\mathbb{I} \coloneqq \mathbb{I} \times \mathbb{I}$ *and interpret all other sorts by themselves:* $\mathrm{T}\mathsf{Ty} \coloneqq \mathsf{Ty}$*,* $(\mathrm{T}\mathsf{Tm})(\mathtt{A}) \coloneqq \mathsf{Tm}(\mathtt{A})$*,* $\mathrm{T}\mathsf{Cof} \coloneqq \mathsf{Cof}$*,* $(\mathrm{T}\mathsf{True})(\mathtt{P}) \coloneqq \mathsf{True}(\mathtt{P})$*.*

▶ **Notation 29.** *For infix operators, we use a subscript to denote interpretation, for example writing* $\approx_{\mathrm{T}}$ *instead of* $\mathrm{T}(- \approx -)$*.*

▶ **Component 30** (T, interval theory)**.** *We interpret the operations of* $\mathrm{Rev}_\phi\Phi$ *in* $\mathbb{C}\textsc{tt}[\iota\Phi]$ *as in Definition 27.*

▶ **Component 31** (T, cofibration theory)**.** *We interpret the cofibration-forming operations as follows.*

$$(\mathtt{i}_0, \mathtt{i}_1) \approx_{\mathrm{T}} (\mathtt{j}_0, \mathtt{j}_1) \coloneqq (\mathtt{i}_0 \approx \mathtt{j}_0) \cap (\mathtt{i}_1 \approx \mathtt{j}_1)$$

$$\mathrm{T}\top \coloneqq \top \qquad \mathtt{P} \cap_{\mathrm{T}} \mathtt{Q} \coloneqq \mathtt{P} \cap \mathtt{Q} \qquad \mathrm{T}\bot \coloneqq \bot \qquad \mathtt{P} \cup_{\mathrm{T}} \mathtt{Q} \coloneqq \mathtt{P} \cup \mathtt{Q}$$

These definitions validate the associated axioms for the True *judgment. We interpret the* $\mathsf{elim}^{\mathsf{Ty}}_{\bot}$*,* $\mathsf{elim}^{\mathsf{Tm}}_{\bot}$*,* $\mathsf{elim}^{\mathsf{Ty}}_{\cup}$*, and* $\mathsf{elim}^{\mathsf{Tm}}_{\cup}$ *eliminators as themselves.*

▶ **Component 32** (T, filling). *We interpret filling by iterated filling, first in one component of the interval variable and then in the other, defining* $\mathsf{Tfill}(\mathtt{A}, \mathtt{P}, \mathtt{a}, (\mathtt{j}_0, \mathtt{j}_1), \mathtt{a}_0, (\mathtt{k}_0, \mathtt{k}_1))$ *to be*

$$\mathsf{fill}^{\mathtt{j}_1 \to \mathtt{k}_1}(\langle \mathtt{i}_1 \rangle \mathtt{A}(\mathtt{k}_0, \mathtt{i}_1), [\mathtt{P} \mapsto \langle \mathtt{i}_1 \rangle \mathtt{a}(\mathtt{k}_0, \mathtt{i}_1)], \mathsf{fill}^{\mathtt{j}_0 \to \mathtt{k}_0}(\langle \mathtt{i}_0 \rangle \mathtt{A}(\mathtt{i}_0, \mathtt{j}_1), [\mathtt{P} \mapsto \langle \mathtt{i}_0 \rangle \mathtt{a}(\mathtt{i}_0, \mathtt{j}_1)], \mathtt{a}_0)).$$

We interpret type formers that do not involve the interval, namely Σ and Π types, identity types, and U and El, as themselves. This leaves path types, glue types, and suspensions. We interpret the path type as a type of squares with fixed values at the coordinates $\mathrm{T0} = (0, 1)$ and $\mathrm{T1} = (1, 0)$, encoded as an iterated path type consisting of the two unfixed points at $(0, 0)$ and $(1, 1)$, four 1-dimensional paths forming a boundary, and a 2-dimensional path relating them.

▶ **Component 33** (T, path types). *We define* $\mathrm{T}\mathsf{Path}(\mathtt{A}, \mathtt{a}_{01}, \mathtt{a}_{10})$ *to be the iterated path type*

$$\begin{array}{l}
\Sigma \mathtt{a}_{00}{:}\mathtt{A}(0,0).\ \Sigma \mathtt{a}_{11}{:}\mathtt{A}(1,1). \\
\Sigma \mathtt{p}_{\bullet 0}{:}\mathsf{Path}(\langle \mathtt{i}_0 \rangle \mathtt{A}(\mathtt{i}_0, 0), \mathtt{a}_{00}, \mathtt{a}_{10}).\ \Sigma \mathtt{p}_{\bullet 1}{:}\mathsf{Path}(\langle \mathtt{i}_0 \rangle \mathtt{A}(\mathtt{i}_0, 1), \mathtt{a}_{01}, \mathtt{a}_{11}). \\
\Sigma \mathtt{p}_{0\bullet}{:}\mathsf{Path}(\langle \mathtt{i}_1 \rangle \mathtt{A}(0, \mathtt{i}_1), \mathtt{a}_{00}, \mathtt{a}_{01}).\ \Sigma \mathtt{p}_{1\bullet}{:}\mathsf{Path}(\langle \mathtt{i}_1 \rangle \mathtt{A}(1, \mathtt{i}_1), \mathtt{a}_{10}, \mathtt{a}_{11}). \\
\mathsf{Path}(\langle \mathtt{i}_0 \rangle \mathsf{Path}(\langle \mathtt{i}_1 \rangle \mathtt{A}(\mathtt{i}_0, \mathtt{i}_1), \mathtt{p}_{\bullet 0} \mathbin{@} \mathtt{i}_0, \mathtt{p}_{\bullet 1} \mathbin{@} \mathtt{i}_0), \mathtt{p}_{0\bullet}, \mathtt{p}_{1\bullet})
\end{array}
\qquad
\begin{array}{ccc}
a_{00} & \overset{p_{\bullet 0}}{\longrightarrow} & a_{10} \\
{\scriptstyle p_{0\bullet}}\downarrow & & \downarrow{\scriptstyle p_{1\bullet}} \\
a_{01} & \underset{p_{\bullet 1}}{\longrightarrow} & a_{11}
\end{array}$$

and set $\mathrm{T}\lambda^{\mathbb{I}}(\mathtt{a}) \coloneqq (\_, \_, \_, \_, \_, \lambda \mathtt{i}_0.\lambda \mathtt{i}_1.\mathtt{a}(\mathtt{i}_0, \mathtt{i}_1))$ *and* $\mathtt{t} \mathbin{@_{\mathrm{T}}} (\mathtt{i}_0, \mathtt{i}_1) \coloneqq \mathtt{t}.6 \mathbin{@} \mathtt{i}_0 \mathbin{@} \mathtt{i}_1$*, where the first five components in* $\mathrm{T}\lambda^{\mathbb{I}}$ *are determined by the final one.*

We write $\mathrm{T}\lambda \mathtt{i}_0, \mathtt{i}_1.a$ *as shorthand for* $\mathrm{T}\lambda^{\mathbb{I}}(\langle \mathtt{i}_0, \mathtt{i}_1 \rangle a)$.

▶ Remark 34. This iterated path type could be naturally expressed as an *extension type*. Introduced by Riehl and Shulman for simplicial type theory [28, §2.2] and discussed by Angiuli [1, §3.5] in the context of cubical type theory, these are types of n-cubes with fixed values on some cofibration. In a theory with these types, $\mathrm{T}\mathsf{Path}$ could be defined as an extension type over the cofibration in two variables $\mathtt{i}_0 : \mathbb{I}, \mathtt{i}_1 : \mathbb{I} \vdash (\mathtt{i}_0 \approx 0 \cap \mathtt{i}_1 \approx 1) \cup (\mathtt{i}_0 \approx 1 \cap \mathtt{i}_1 \approx 0)$.

To interpret glue and suspension types, we need to convert between inhabitants of $\mathsf{Path}(\mathtt{C}, \mathtt{c}_0, \mathtt{c}_1)$ and inhabitants of $\mathrm{T}\mathsf{Path}(\langle \mathtt{i}_0, \mathtt{i}_1 \rangle \mathtt{C}(\mathtt{i}_0), \mathtt{c}_0, \mathtt{c}_1)$. First, the easy direction:

▶ **Notation 35.** *Over the environment* $([\mathtt{C} : \mathbb{I} \to \mathsf{Ty}, \mathtt{c}_0 : \mathtt{C}(0), \mathtt{c}_1 : \mathtt{C}(1)], \mathtt{p} : \mathsf{Path}(\mathtt{C}, \mathtt{c}_0, \mathtt{c}_1))$*, we define* $\mathsf{thicken}(\mathtt{p}) \coloneqq \mathrm{T}\lambda \mathtt{i}_0, \mathtt{i}_1.\mathtt{p} \mathbin{@} \mathtt{i}_0 : \mathrm{T}\mathsf{Path}(\langle \mathtt{i}_0, \mathtt{i}_1 \rangle \mathtt{C}(\mathtt{i}_0), \mathtt{c}_0, \mathtt{c}_1)$.

For the inverse, we extract the "anti-diagonal" of a square by inverting it along one axis – a standard construction using the filling operation – and then extracting the diagonal.

▶ **Definition 36** (Path inversion). *Over the environment* $([\mathtt{C} : \mathsf{Ty}, \mathtt{c}_0\ \mathtt{c}_1 : \mathtt{C}], \mathtt{p} : \mathtt{c}_0 \sim^{\mathtt{C}} \mathtt{c}_1)$*, we define* $\mathsf{sym}(\mathtt{p}) \coloneqq \lambda \mathtt{i}.\mathsf{fill}^{1 \to 0}(\langle \_ \rangle \mathtt{C}, [\mathtt{i} \approx 0 \mapsto \langle \_ \rangle \mathtt{c}_1, \mathtt{i} \approx 1 \mapsto \langle \mathtt{j} \rangle \mathtt{p} \mathbin{@} \mathtt{j}], \mathtt{c}_1) : \mathtt{c}_1 \sim^{\mathtt{C}} \mathtt{c}_0$.

▶ **Definition 37.** *Over* $([\mathtt{C} : \mathbb{I} \to \mathsf{Ty}, \mathtt{c}_0 : \mathtt{C}(0), \mathtt{c}_1 : \mathtt{C}(1)], \mathtt{q} : \mathrm{T}\mathsf{Path}(\langle \mathtt{i}_0, \mathtt{i}_1 \rangle \mathtt{C}(\mathtt{i}_0), \mathtt{c}_0, \mathtt{c}_1))$*, we define* $\mathsf{anti}(\mathtt{q}) \coloneqq \lambda \mathtt{i}.\mathsf{sym}(\langle \mathtt{j} \rangle \mathtt{q} \mathbin{@_{\mathrm{T}}} (\mathtt{i}, \mathtt{j})) \mathbin{@} \mathtt{i} : \mathsf{Path}(\mathtt{C}, \mathtt{c}_0, \mathtt{c}_1)$.

To show that these constitute an equivalence, we use the contractibility of dependent singleton types:

▶ **Proposition 38.** *Over the environment* $(\mathtt{C} : \mathbb{I} \to \mathsf{Ty}, \mathtt{c}_0 : \mathtt{C}(0))$*, we have a term of type* $\mathsf{isContr}(\Sigma \mathtt{c}_1{:}\mathtt{C}(1).\mathsf{Path}(\mathtt{C}, \mathtt{c}_0, \mathtt{c}_1))$.

Proof (cf. [1, §3.2]). For the center of contraction, take the pair $\mathtt{s}_0 \coloneqq (\_, \lambda \mathtt{i}.\mathsf{coe}^{0 \to \mathtt{i}}(\mathtt{C}, \mathtt{c}_0))$ (whose first component is determined by its second). Given a singleton $\mathtt{s}$, we have a path $\lambda \mathtt{j}.(\_, \lambda \mathtt{i}.\mathsf{fill}^{0 \to \mathtt{i}}(\mathtt{C}, [\mathtt{j} \approx 0 \mapsto \langle \mathtt{k} \rangle \mathtt{s}_0 \mathbin{@} \mathtt{k}, \mathtt{j} \approx 1 \mapsto \langle \mathtt{k} \rangle \mathtt{s} \mathbin{@} \mathtt{k}], \mathtt{c}_0))$ from $\mathtt{s}_0$ to $\mathtt{s}$. ◀

▶ **Lemma 39.** *Over the environment* $(\mathtt{C} : \mathsf{Ty}, \mathtt{c}_0 : \mathtt{C}, \mathtt{c}_1 : \mathtt{C})$*, the function* $\lambda\mathtt{p}.\mathsf{thicken}(\mathtt{p}) : \mathsf{Path}(\mathtt{C}, \mathtt{c}_0, \mathtt{c}_1) \to \mathrm{T}\mathsf{Path}(\langle \mathtt{i}_0, \mathtt{i}_1\rangle \mathtt{C}(\mathtt{i}_0), \mathtt{c}_0, \mathtt{c}_1)$ *is an equivalence.*

Proof. Our proof of Proposition 38 uses only constructs on which we have already defined T. Thus we can mechanically derive from it a term of type $\mathsf{TisContr}(\Sigma \mathtt{d}_1{:}\mathtt{D}(1,0).\mathsf{TPath}(\mathtt{D}, \mathtt{d}_0, \mathtt{d}_1))$ over $(\mathtt{D} : \mathbb{I} \times \mathbb{I} \to \mathsf{Ty}, \mathtt{d}_0 : \mathtt{D}(0,1))$. Using Definition 37, we can go from TisContr to isContr. Taking $\mathtt{D}(\mathtt{i}_0, \mathtt{i}_1) \coloneqq \mathtt{C}(\mathtt{i}_0)$ and $\mathtt{d}_0 \coloneqq \mathtt{c}_0$ gives $\mathsf{isContr}(\Sigma \mathtt{c}_1{:}\mathtt{C}(1).\mathsf{TPath}(\langle \mathtt{i}_0, \mathtt{i}_1\rangle \mathtt{C}(\mathtt{i}_0), \mathtt{c}_0, \mathtt{c}_1))$. Thus $\lambda\mathtt{s}.(\mathtt{s}.1, \mathsf{thicken}(\mathtt{s}.2)) : \Sigma \mathtt{c}_1{:}\mathtt{C}(1).\mathsf{Path}(\mathtt{C}, \mathtt{c}_0, \mathtt{c}_1) \to \Sigma \mathtt{c}_1{:}\mathtt{C}(1).\mathsf{TPath}(\langle \mathtt{i}_0, \mathtt{i}_1\rangle \mathtt{C}(\mathtt{i}_0), \mathtt{c}_0, \mathtt{c}_1)$ is a map between contractible types and therefore an equivalence. It follows [39, Theorem 4.7.7] that it is also a fiberwise equivalence. ◀

We can use Lemma 39 inside the definition of equivalence to construct TGlue.

▶ **Component 40** (T, glue)**.** *Over* $(\mathtt{A} : \mathsf{Ty}, \mathtt{P} : \mathsf{Cof}, \mathtt{T} : [\mathtt{P}] \to \mathsf{Ty}, \mathtt{e} : [\mathtt{P}] \to \mathtt{T} \simeq_{\mathrm{T}} \mathtt{A})$*, define* $\mathsf{TGlue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \mathtt{e}) \coloneqq \mathsf{Glue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \widehat{\mathtt{e}})$ *where* $\widehat{\mathtt{e}}$ *is derived from* e *by using Lemma 39 to replace each use of* TPath *with* Path*. Set* $\mathsf{Tglue}(\mathtt{a}, \mathtt{t}) \coloneqq \mathsf{glue}(\mathtt{a}, \mathtt{t})$ *and* $\mathsf{Tunglue}(\mathtt{g}) \coloneqq \mathsf{unglue}(\mathtt{g})$*.*

The interpretation of suspensions is similar, but we use thicken and anti more directly.

▶ **Component 41** (T, suspension)**.** *Define*

$$
\begin{array}{lcllcl}
\mathsf{TSusp}(\mathtt{A}) & \coloneqq & \mathsf{Susp}(\mathtt{A}) & \mathsf{Tmerid}(\mathtt{a}) & \coloneqq & \mathsf{thicken}(\mathsf{merid}(\mathtt{a})) \\
\mathsf{Tnorth} & \coloneqq & \mathsf{north} & \mathsf{Telim}(\mathtt{C}, \mathtt{n}, \mathtt{s}, \mathtt{m}, \mathtt{t}) & \coloneqq & \mathsf{elim}(\mathtt{C}, \mathtt{n}, \mathtt{s}, \langle \mathtt{a}\rangle \mathsf{anti}(\mathtt{m}(\mathtt{a})), \mathtt{t}) \\
\mathsf{Tsouth} & \coloneqq & \mathsf{south} & & &
\end{array}
$$

For $\mathsf{Tmerid}\beta(\mathtt{C}, \mathtt{n}, \mathtt{s}, \mathtt{m}, \mathtt{a})$*, we compose* $\mathsf{cong}_{\mathsf{thicken}}(\mathsf{merid}\beta(\mathtt{C}, \mathtt{n}, \mathtt{s}, \langle \mathtt{a}\rangle \mathsf{thicken}^{-1}(\mathtt{m}(\mathtt{a})), \mathtt{a}))$ *with the path* $\mathsf{thicken}(\mathsf{thicken}^{-1}(\mathtt{q})) \sim \mathtt{q}$*, using that* thicken *is an equivalence, then thicken the composed path to get a* T*-path.*

This completes the definition of T, as summarized in the following theorem. We record that it preserves the constructs of $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$ and the cofibration judgments for future use.

▶ **Theorem 42.** *For every self-dual interval theory* (Φ, ϕ)*, there is a representable map functor* $\mathrm{T} \colon \mathbb{C}\mathrm{TT}[\iota \mathrm{Rev}_\phi \Phi] \to \mathbb{C}\mathrm{TT}[\iota\Phi]$ *in the coslice* $(\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id},\mathsf{U}} + \mathbb{C}\mathrm{OF})/\mathbf{RMC}$*.*

5 Spans

Abstracting from the particular case of T, we now develop tools – span RMCs and the *span interpretation* between suitable RMC functors $F, G \colon \mathbb{C}\mathrm{TT}[\iota\Phi] \to \mathbb{C}\mathrm{TT}[\iota\Psi]$ – that we use in § 6 to prove that certain morphisms of models induced by RMC functors are weak equivalences. This construction at the level of RMCs is inspired by and resembles path object constructions at the level of models [22, §5], as well as Tabareau, Tanter, and Sozeau's *univalent parametricity* translation for the Calculus of Inductive Constructions [35].

5.1 The representable map category of spans

We write $\mathrm{Span}(\mathcal{C})$ for the category of *spans in a category* $\mathcal{C}$, i.e., the category of functors from the diagram category $\{0 \leftarrow \mathrm{r} \rightarrow 1\}$ into $\mathcal{C}$. Given $X \in \mathrm{Span}(\mathcal{C})$, we write $d^0 \colon X_{\mathrm{r}} \to X_0$ and $d^1 \colon X_{\mathrm{r}} \to X_1$ for its two projections.

▶ **Proposition 43.** *If* $\mathbb{R}$ *is an RMC, then* $\mathrm{Span}(\mathbb{R})$ *is an RMC when equipped with the class of levelwise representable maps.*

Proof. As a functor category, $\mathbb{R}$ has finite limits computed pointwise in $\mathbb{R}$. In particular, the fact that representable maps are closed under pullback in $\mathbb{R}$ implies the same of $\mathrm{Span}(\mathbb{R})$.

It remains to show that the representable maps in $\mathrm{Span}(\mathbb{R})$ are exponentiable. Let $f\colon Z \to Y$ and $p\colon Y \to X$ be maps in $\mathrm{Span}(\mathbb{R})$ and suppose p is representable. As p_0 and p_1 are exponentiable, we have dependent products $g_0 \coloneqq (p_0)_* f_0\colon \Pi_{p_0} Z_0 \to X_0$ and $g_1 \coloneqq (p_1)_* f_1\colon \Pi_{p_1} Z_1 \to X_1$. Write k for the composite

$$\begin{array}{c} Y_{\mathrm{r}} \times_{X_0 \times X_1} (\Pi_{p_0} Z_0 \times \Pi_{p_1} Z_1) \\ \downarrow{\scriptstyle \mathbb{R}} \\ Y_{\mathrm{r}} \times_{Y_0 \times Y_1} ((Y_0 \times_{X_0} \Pi_{p_0} Z_0) \times (Y_1 \times_{X_1} \Pi_{p_1} Z_1)) \xrightarrow{Y_{\mathrm{r}} \times_{Y_0 \times Y_1} (\epsilon_{Z_0} \times \epsilon_{Z_1}))} Y_{\mathrm{r}} \times_{Y_0 \times Y_1} Z_0 \times Z_1 \end{array}$$

induced by the counits of the (pullback, pushforward) adjunction, and write q for the (representable) pullback

$$\begin{array}{ccc} Y_{\mathrm{r}} \times_{X_0 \times X_1} (\Pi_{p_0} Z_0 \times \Pi_{p_1} Z_1) & \longrightarrow & Y_{\mathrm{r}} \\ {\scriptstyle q}\downarrow & \lrcorner & \downarrow{\scriptstyle p_{\mathrm{r}}} \\ X_{\mathrm{r}} \times_{X_0 \times X_1} (\Pi_{p_0} Z_0 \times \Pi_{p_1} Z_1) & \longrightarrow & X_{\mathrm{r}}. \end{array}$$

Writing the components of $q_* k^*(f_{\mathrm{r}}, (d^0, d^1))\colon \Pi_q k^* Z_{\mathrm{r}} \to X_{\mathrm{r}} \times_{X_0 \times X_1} (\Pi_{p_0} Z_0 \times \Pi_{p_1} Z_1)$ as $\langle g_{\mathrm{r}}, d^0, d^1 \rangle$, the morphism of spans

$$\begin{array}{ccccc} \Pi_{p_0} Z_0 & \xleftarrow{d^0} & \Pi_q k^* Z_{\mathrm{r}} & \xrightarrow{d^1} & \Pi_{p_1} Z_1 \\ {\scriptstyle g_0}\downarrow & & \downarrow{\scriptstyle g_{\mathrm{r}}} & & \downarrow{\scriptstyle g_1} \\ X_0 & \xleftarrow[d^0]{} & X_{\mathrm{r}} & \xrightarrow[d^1]{} & X_1 \end{array}$$

is a pushforward of f along p. ◀

By definition, the projections $\pi_0, \pi_1\colon \mathrm{Span}(\mathbb{R}) \to \mathbb{R}$ are RMC functors. Restricting our attention now to $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$, we define an RMC functor Refl fitting in the diagram

$$\begin{array}{ccccc} & & \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}} & & \\ & \swarrow & {\scriptstyle \mathrm{Refl}}\downarrow & \searrow & \\ \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}} & \xleftarrow[\pi_0]{} & \mathrm{Span}(\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}) & \xrightarrow[\pi_1]{} & \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}} \end{array}$$

by giving an interpretation. For $\Phi \in \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$, we write the span $\mathrm{Refl}\Phi$ as $\Phi \overset{d^0_\Phi}{\leftarrow} \mathrm{P}\Phi \overset{d^1_\Phi}{\to} \Phi$, i.e., with $\mathrm{P}\colon \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}} \to \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$ denoting the composite of Refl with the apex projection.

To interpret the type and term judgments, we will use the environment $\mathsf{Ty}^{\simeq}$ of 1-to-1 correspondences from Definition 10.

▶ **Definition 44.** *Write* $\mathsf{Tm}^{\simeq} \coloneqq ((\mathtt{A}, \mathtt{A}', \overline{\mathtt{A}}, \cdots) : \mathsf{Ty}^{\simeq}, \mathtt{a} : \mathtt{A}, \mathtt{a}' : \mathtt{A}', \overline{\mathtt{a}} : \overline{\mathtt{A}}(\mathtt{a}, \mathtt{a}')) \in \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$ *and* $d^0, d^1\colon \mathsf{Tm}^{\simeq} \to \mathsf{Tm}$ *for the instantiations projecting* $(\mathtt{A}, \mathtt{a})$ *and* $(\mathtt{A}', \mathtt{a}')$ *respectively.*

▶ **Component 45** (Refl, sorts)**.** *For sorts, we define* $\mathrm{Refl}\mathsf{Ty} \coloneqq \{\mathsf{Ty} \overset{d^0}{\leftarrow} \mathsf{Ty}^{\simeq} \overset{d^1}{\to} \mathsf{Ty}\}$ *and* $\mathrm{Refl}\mathsf{Tm} \coloneqq \{\mathsf{Tm} \overset{d^0}{\leftarrow} \mathsf{Tm}^{\simeq} \overset{d^1}{\to} \mathsf{Tm}\}$*, with* $\mathrm{Refl}\pi_{\mathsf{Tm}}\colon \mathrm{Refl}\mathsf{Ty} \twoheadrightarrow \mathrm{Refl}\mathsf{Tm}$ *the evident projection.*

Defining Refl for a type former $T\colon \Phi \Rightarrow \mathsf{Ty}$ now amounts to giving, over the environment $(\mathtt{p} : \mathrm{P}\Phi)$, a 1-to-1 correspondence $\mathrm{R}(\mathtt{p}, -, -)$ between $T(d^0_\Phi(\mathtt{p}))$ and $T(d^1_\Phi(\mathtt{p}))$. Similarly, interpreting a term former $t\colon (\mathtt{x} : \Phi) \Rightarrow \mathsf{Tm}(I(\mathtt{x}))$ amounts to giving over $(\mathtt{p} : \mathrm{P}\Phi)$ an inhabitant of $\mathrm{R}(\mathtt{p}, t(d^0_\Phi(\mathtt{p})), t(d^1_\Phi(\mathtt{p})))$.

▶ **Component 46** (Refl, unit type). *We interpret the unit type former by the 1-to-1 correspondence* $\mathrm{RUnit}(\_,\_,\_) = \mathsf{Unit}$ *and its unique inhabitant by the unique witness.*

▶ **Component 47** (Refl, Σ types). *In the environment consisting of* $\mathtt{A} : \mathsf{Ty}$, $\mathtt{A}' : \mathsf{Ty}$, *a 1-to-1 correspondence* $\overline{\mathtt{A}} : (\mathtt{A}, \mathtt{A}') \to \mathsf{Ty}$, *families* $\mathtt{B} : \mathtt{A} \to \mathsf{Ty}$, $\mathtt{B}' : \mathtt{A}' \to \mathsf{Ty}$, *and a family of 1-to-1 correspondences* $\overline{\mathtt{B}} : ([\mathtt{a} : \mathtt{A}, \mathtt{a}' : \mathtt{A}'], \overline{\mathtt{a}} : \overline{\mathtt{A}}(\mathtt{a}, \mathtt{a}'), \mathtt{b} : \mathtt{B}(\mathtt{a}), \mathtt{b}' : \mathtt{B}'(\mathtt{a}')) \to \mathsf{Ty}$, *we define* $\mathrm{R}\Sigma$ *at* $\mathtt{s} : \Sigma(A, B)$ *and* $\mathtt{s}' : \Sigma(A', B')$ *to be* $\Sigma\overline{\mathtt{a}} : \overline{\mathtt{A}}(\mathtt{s}.1, \mathtt{s}'.1).\ \overline{\mathtt{B}}(\overline{\mathtt{a}}, \mathtt{s}.2, \mathtt{s}'.2)$. *We interpret pairing and projection by pairing and projection in this* Σ *type.*

▶ **Component 48** (Refl, identity types). *In the environment consisting of* $\mathtt{A} : \mathsf{Ty}$, $\mathtt{A}' : \mathsf{Ty}$, *a 1-to-1 correspondence* $\overline{\mathtt{A}} : (\mathtt{A}, \mathtt{A}') \to \mathsf{Ty}$, *terms* $\mathtt{a}_0\ \mathtt{a}_1 : \mathtt{A}$ *and* $\mathtt{a}'_0\ \mathtt{a}'_1 : \mathtt{A}'$, *and* $\overline{\mathtt{a}}_0 : \overline{\mathtt{A}}(\mathtt{a}_0, \mathtt{a}'_0)$ *and* $\overline{\mathtt{a}}_1 : \overline{\mathtt{A}}(\mathtt{a}_1, \mathtt{a}'_1)$, *we define* RId *at* $\mathtt{p} : \mathtt{a}_0 \asymp \mathtt{a}_1$ *and* $\mathtt{p}' : \mathtt{a}'_0 \asymp \mathtt{a}'_1$ *to be the type of identities between* $\overline{\mathtt{a}}_0 : \overline{\mathtt{A}}(\mathtt{a}_0, \mathtt{a}'_0)$ *and* $\overline{\mathtt{a}}_1 : \overline{\mathtt{A}}(\mathtt{a}_1, \mathtt{a}'_1)$ *over* $\mathtt{p}$ *and* $\mathtt{p}'$, *i.e., the type of identities between the transport of* $\overline{\mathtt{a}}_0$ *along these identities and* $\overline{\mathtt{a}}_1$.

This completes the definition of $\mathrm{Refl} : \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}} \to \mathrm{Span}(\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}})$. Given an RMC $i : \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}} \to \mathbb{R}$ under $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$, we can now regard $\mathrm{Span}(\mathbb{R})$ as an RMC under $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$ by way of the composite $\mathrm{Span}(i) \circ \mathrm{Refl} : \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}} \to \mathrm{Span}(\mathbb{R})$. In particular, we have $\mathbf{0}_{\mathrm{Span}(\mathbb{R})} \in \mathbf{Mod}(\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}})$ as in Definition 13.

▶ **Proposition 49.** *For any* $i : \mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}} \to \mathbb{R}$, *the morphisms* $\mathbf{0}_{\pi_0}, \mathbf{0}_{\pi_1} : \mathbf{0}_{\mathrm{Span}(\mathbb{R})} \to \mathbf{0}_{\mathbb{R}}$ *in* $\mathbf{Mod}(\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}})$ *are weak equivalences.*

Proof. We consider $\pi_0 : \mathrm{Span}(\mathbb{R}) \to \mathbb{R}$, the case of π_1 being symmetric. An object of $\Gamma \in \mathbf{0}_{\mathrm{Span}(\mathbb{R})}(\star)$ is a span $\{\Gamma_0 \xleftarrow{d^0} \Gamma_\mathrm{r} \xrightarrow{d^1} \Gamma_1\}$ obtained by iterated context extension of the terminal span with respect to the representable map $\mathsf{Tm}^{\simeq} \to \mathsf{Ty}^{\simeq}$ as described in Definition 9. It follows that Γ_0, Γ_1, and Γ_r are contexts in $\mathbb{R}$, i.e., environments of term hypotheses, and that we have instantiations $\eta^0 : \Gamma_0 \to \Gamma_\mathrm{r}$ and $\eta^1 : \Gamma_1 \to \Gamma_\mathrm{r}$ that are homotopy inverses of $d^0 : \Gamma_\mathrm{r} \to \Gamma_0$ and $d^1 : \Gamma_\mathrm{r} \to \Gamma_1$ at each entry. To show weak type lifting for $\pi_0 : \mathrm{Span}(\mathbb{R}) \to \mathbb{R}$, we are given $B : \Gamma_0 \to \mathsf{Ty}$ in $\mathbb{R}$ and must construct an $A : \Gamma \to \mathrm{Refl}\mathsf{Ty}$ in $\mathrm{Span}(\mathbb{R})$ for which A_0 is equivalent to B. We have

$$\begin{array}{ccccc} \Gamma_0 & \xleftarrow{d^0} & \Gamma_\mathrm{r} & \xrightarrow{d^1} & \Gamma_1 \\ {\scriptstyle B}\downarrow & & & & \downarrow{\scriptstyle Bd^0\eta^1} \\ \mathsf{Ty} & \xleftarrow[d^0]{} & \mathsf{Ty}^{\simeq} & \xrightarrow[d^1]{} & \mathsf{Ty} \end{array}$$

and, since $d^0\eta^1 d^1$ is homotopic to d^0 at each component, we can find a map $\Gamma_r \to \mathsf{Ty}^{\simeq}$ that makes the diagram commute. Taking the result as our A, we have not only an equivalence from A_0 to B but an equality. The construction of term lifting is analogous. ◀

5.2 Relating interpretations using spans

For this section, we fix two RMC functors $F, G : \mathbb{C}\mathrm{TT}[\iota\Phi] \to \mathbb{C}\mathrm{TT}[\iota\Psi]$ in the coslice under the combined sub-SOGAT $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id},\mathsf{U}} + \mathbb{C}\mathrm{OF}$, where $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id},\mathsf{U}}$ is the extension of $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$ by a universe U with El. We construct a third RMC functor $\mathrm{S}^F_G : \mathbb{C}\mathrm{TT}[\iota\Phi] \to \mathrm{Span}(\mathbb{C}\mathrm{TT}[\iota\Psi])$ in the coslice under $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$ that fits in the diagram

$$\begin{array}{ccccc} & & \mathbb{C}\mathrm{TT}[\iota\Phi] & & \\ & {\scriptstyle F}\swarrow & \downarrow{\scriptstyle \mathrm{S}^F_G} & \searrow{\scriptstyle G} & \\ \mathbb{C}\mathrm{TT}[\iota\Psi] & \xleftarrow[\pi_0]{} & \mathrm{Span}(\mathbb{C}\mathrm{TT}[\iota\Psi]) & \xrightarrow[\pi_1]{} & \mathbb{C}\mathrm{TT}[\iota\Psi]. \end{array}$$

The effect of this functor, which we exploit in § 6, is to show that F and G are approximately "the same". Note that we need to know very little about F and G to obtain S^F_G: this reflects that the constructs of $\mathbb{C}\mathrm{TT}[\iota\Phi]$ are all characterized up to equivalence by their universal properties, so an interpretation has little choice in where to send them. It is key here that we are looking at *second-order* models, i.e., RMC functors; we would not have the same result for morphisms of first-order models.

For $\Theta \in \mathbb{C}\mathrm{TT}$, we write the span $\mathrm{S}^F_G\Theta$ as $F\Theta \overset{d^0_\Theta}{\leftarrow} \mathrm{M}^F_G\Theta \overset{d^1_\Theta}{\rightarrow} G\Theta$. Because we intend S^F_G to be a morphism in the coslice under $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$, the interpretations of the constructs of $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$ are determined by the definition of Refl from the previous section.

From this point until the summary statement Theorem 62, we omit the annotations on S^F_G and M^F_G and simply write S and M.

▶ **Component 50** (S, sorts). *Set* $\mathrm{S}\mathsf{TY} \coloneqq \{\mathsf{Ty} \overset{d^0}{\leftarrow} \mathsf{Ty}^{\simeq} \overset{d^1}{\rightarrow} \mathsf{Ty}\}$ *and* $\mathrm{S}\mathsf{Tm} \coloneqq \{\mathsf{Tm} \overset{d^0}{\leftarrow} \mathsf{Tm}^{\simeq} \overset{d^1}{\rightarrow} \mathsf{Tm}\}$ *as required by the definition of* Refl. *For the remaining sorts:*

1. *Set* $\mathrm{S}\mathbb{I} \coloneqq \{F\mathbb{I} \overset{d^0}{\leftarrow} F\mathbb{I} \times G\mathbb{I} \overset{d^1}{\rightarrow} G\mathbb{I}\}$.
2. *Set* $\mathrm{MCof} \coloneqq (\mathtt{P}\ \mathtt{P}'\ \overline{\mathtt{P}} : \mathsf{Cof}, [\overline{\mathtt{P}} \to \mathtt{P}, \overline{\mathtt{P}} \to \mathtt{P}'])$ *with* $d^0_{\mathsf{Cof}}, d^1_{\mathsf{Cof}}$ *projecting* $\mathtt{P}$ *and* $\mathtt{P}'$ *respectively.*
3. *Set* $\mathrm{M}\mathsf{True} \coloneqq ((\mathtt{P}, \mathtt{P}', \overline{\mathtt{P}}) : \mathrm{MCof}, \overline{\mathtt{P}})$ *with* $d^0_{\mathsf{True}}, d^1_{\mathsf{True}}$ *applying the implications* $\overline{\mathtt{P}} \to \mathtt{P}$ *and* $\overline{\mathtt{P}} \to \mathtt{P}'$*, and define* $\mathrm{M}\pi_{\mathsf{True}}$ *to be the evident projection.*

▶ **Component 51** (S, interval theory). *By definition of* $\mathrm{S}\mathbb{I}$*, the interpretation of the interval theory is forced by* F *and* G*. Unfolding, the interpretation* $\mathrm{M}f$ *of each operation* $f\colon \mathbb{I}^n \to \mathbb{I}$ *of the interval theory is* $(F\mathbb{I} \times G\mathbb{I})^n \cong F\mathbb{I}^n \times G\mathbb{I}^n \overset{Ff \times Gf}{\longrightarrow} F\mathbb{I} \times G\mathbb{I}$.

▶ **Component 52** (S, cofibration theory). *We interpret the cofibration operations as follows.*

$$
\begin{aligned}
(\mathtt{i}, \mathtt{x}) \approx_{\mathrm{M}} (\mathtt{j}, \mathtt{y}) &\coloneqq (\mathtt{i} \approx_F \mathtt{j}, \mathtt{x} \approx_G \mathtt{y}, (\mathtt{i} \approx_F \mathtt{j}) \cap (\mathtt{x} \approx_G \mathtt{y})) \\
\mathrm{M}\top &\coloneqq (F\top, G\top, \top) \\
(\mathtt{P}, \mathtt{P}', \overline{\mathtt{P}}) \cap_{\mathrm{M}} (\mathtt{Q}, \mathtt{Q}', \overline{\mathtt{Q}}) &\coloneqq (\mathtt{P} \cap_F \mathtt{Q}, \mathtt{P}' \cap_G \mathtt{Q}', \overline{\mathtt{P}} \cap \overline{\mathtt{Q}}) \\
\mathrm{M}\bot &\coloneqq (F\bot, G\bot, \bot) \\
(\mathtt{P}, \mathtt{P}', \overline{\mathtt{P}}) \cup_{\mathrm{M}} (\mathtt{Q}, \mathtt{Q}', \overline{\mathtt{Q}}) &\coloneqq (\mathtt{P} \cup_F \mathtt{Q}, \mathtt{P}' \cup_G \mathtt{Q}', \overline{\mathtt{P}} \cup \overline{\mathtt{Q}})
\end{aligned}
$$

The axioms for cofibrations ensure that these definitions preserve the implicit requirement that for $(\mathtt{P}, \mathtt{P}', \overline{\mathtt{P}}) : \mathrm{MCof}$ *we have* $\overline{\mathtt{P}} \to \mathtt{P}$ *and* $\overline{\mathtt{P}} \to \mathtt{P}'$*. We use this to interpret the* $\mathsf{elim}^{\mathsf{Ty}}_{\cup}$ *and* $\mathsf{elim}^{\mathsf{Tm}}_{\cup}$ *eliminators. For* $\mathsf{elim}^{\mathsf{Ty}}_{\cup}$*, for example, we are given* $(\mathtt{P}, \mathtt{P}', \overline{\mathtt{P}}) : \mathrm{MCof}$*,* $(\mathtt{Q}, \mathtt{Q}', \overline{\mathtt{Q}}) : \mathrm{MCof}$*, compatible* $\mathtt{A} : [\mathtt{P}] \to \mathsf{Ty}$ *and* $\mathtt{B} : [\mathtt{Q}] \to \mathsf{Ty}$*, compatible* $\mathtt{A}' : [\mathtt{P}'] \to \mathsf{Ty}$ *and* $\mathtt{B}' : [\mathtt{Q}'] \to \mathsf{Ty}$*, and compatible 1-to-1 correspondences* $\overline{\mathtt{A}} : ([\overline{\mathtt{P}}], \mathtt{A}, \mathtt{A}') \to \mathsf{Ty}$ *and* $\overline{\mathtt{B}} : ([\overline{\mathtt{Q}}], \mathtt{B}, \mathtt{B}') \to \mathsf{Ty}$*, and we need to extend these to a 1-to-1 correspondence between* $F\mathsf{elim}^{\mathsf{Ty}}_{\cup}(\mathtt{P}, \mathtt{Q}, \mathtt{A}, \mathtt{B})$ *and* $G\mathsf{elim}^{\mathsf{Ty}}_{\cup}(\mathtt{P}', \mathtt{Q}', \mathtt{A}', \mathtt{B}')$ *assuming* $\overline{\mathtt{P}} \cup \overline{\mathtt{Q}}$*. To do so we case on* $\overline{\mathtt{P}} \cup \overline{\mathtt{Q}}$ *and use that we either have both* $\mathtt{P}$ *and* $\mathtt{P}'$ *or both* $\mathtt{Q}$ *and* $\mathtt{Q}'$ *as a consequence.*

▶ **Component 53** (S, filling). *To define* Sfill*, we are given inputs*

$\mathtt{A}$	:	$F\mathbb{I} \to \mathsf{Ty}$	$(\mathtt{j}, \mathtt{y})$	:	$\mathrm{M}\mathbb{I}$
$\mathtt{A}'$	:	$G\mathbb{I} \to \mathsf{Ty}$	$\mathtt{a}_0$	:	$\mathtt{A}(\mathtt{j})$
$\overline{\mathtt{A}}$	:	$(\mathtt{i} : F\mathbb{I}, \mathtt{x} : G\mathbb{I}, \mathtt{a} : \mathtt{A}(\mathtt{i}), \mathtt{a}' : \mathtt{A}'(\mathtt{x})) \to \mathsf{Ty}$	$\mathtt{a}'_0$	:	$\mathtt{A}'(\mathtt{y})$
$(\mathtt{P}, \mathtt{P}', \overline{\mathtt{P}})$	:	MCof	$\overline{\mathtt{a}}_0$	:	$\overline{\mathtt{A}}(\mathtt{j}, \mathtt{y}, \mathtt{a}_0, \mathtt{a}'_0)$
$\mathtt{a}$	:	$(\mathtt{i} : F\mathbb{I}, \mathtt{P}) \to \mathtt{A}(\mathtt{i})$	$(\mathtt{k}, \mathtt{z})$	:	$\mathrm{M}\mathbb{I}$
$\mathtt{a}'$	:	$(\mathtt{x} : G\mathbb{I}, \mathtt{P}') \to \mathtt{A}'(\mathtt{x})$			
$\overline{\mathtt{a}}$	:	$(\mathtt{i} : F\mathbb{I}, \mathtt{x} : G\mathbb{I}, \overline{\mathtt{P}}) \to \overline{\mathtt{A}}(\mathtt{i}, \mathtt{x}, \mathtt{a}(\mathtt{i}), \mathtt{a}'(\mathtt{x}))$			

satisfying $\mathtt{P} \to \mathtt{a}(\mathtt{j}) \equiv \mathtt{a}_0 : \mathtt{A}(\mathtt{j})$, $\mathtt{P}' \to \mathtt{a}'(\mathtt{y}) \equiv \mathtt{a}'_0 : \mathtt{A}'(\mathtt{y})$, *and* $\overline{\mathtt{P}} \to \overline{\mathtt{a}}(\mathtt{j},\mathtt{y}) \equiv \overline{\mathtt{a}}_0 : \overline{\mathtt{A}}(\mathtt{a}_0, \mathtt{a}'_0)$. *Abbreviating* $\mathtt{a}_+(\mathtt{k}) \coloneqq F\mathsf{fill}^{\mathtt{j}\to\mathtt{k}}(\mathtt{A}, [\mathtt{P} \mapsto \mathtt{a}], \mathtt{a}_0)$ *and* $\mathtt{a}'_+(\mathtt{z}) \coloneqq G\mathsf{fill}^{\mathtt{y}\to\mathtt{z}}(\mathtt{A}', [\mathtt{P}' \mapsto \mathtt{a}'], \mathtt{a}'_0)$, *we must exhibit a term of type* $\overline{\mathtt{A}}(\mathtt{k}, \mathtt{z}, \mathtt{a}_+(\mathtt{k}), \mathtt{a}'_+(\mathtt{z}))$. *We take the iterated filling expression*

$$G\mathsf{fill}^{\mathtt{y}\to\mathtt{z}}(\langle \mathtt{x}\rangle \overline{\mathtt{A}}(\mathtt{k}, \mathtt{x}, \mathtt{a}_+(\mathtt{k}), \mathtt{a}'_+(\mathtt{x})), [\overline{\mathtt{P}} \mapsto \langle \mathtt{x}\rangle \overline{\mathtt{a}}(\mathtt{k}, \mathtt{x})],$$
$$F\mathsf{fill}^{\mathtt{j}\to\mathtt{k}}(\langle \mathtt{i}\rangle \overline{\mathtt{A}}(\mathtt{i}, \mathtt{y}, \mathtt{a}_+(\mathtt{i}), \mathtt{a}'_0), [\overline{\mathtt{P}} \mapsto \langle \mathtt{i}\rangle \overline{\mathtt{a}}(\mathtt{i}, \mathtt{y})], \overline{\mathtt{a}}_0)).$$

▶ **Component 54** (S, Π types). *In the environment consisting of* $\mathtt{A} : \mathsf{Ty}$, $\mathtt{A}' : \mathsf{Ty}$, *a 1-to-1 correspondence* $\overline{\mathtt{A}} : (\mathtt{A}, \mathtt{A}') \to \mathsf{Ty}$, *families* $\mathtt{B} : \mathtt{A} \to \mathsf{Ty}$, $\mathtt{B}' : \mathtt{A}' \to \mathsf{Ty}$, *and 1-to-1 correspondences* $\overline{\mathtt{B}} : ([\mathtt{a} : \mathtt{A}, \mathtt{a}' : \mathtt{A}'], \overline{\mathtt{a}} : \overline{\mathtt{A}}(\mathtt{a}, \mathtt{a}'), \mathtt{b} : \mathtt{B}(\mathtt{a}), \mathtt{b}' : \mathtt{B}'(\mathtt{a}')) \to \mathsf{Ty}$, *we take the relation sending* $\mathtt{f} : F\Pi(\mathtt{A}, \mathtt{B})$ *and* $\mathtt{f}' : G\Pi(\mathtt{A}', \mathtt{B}')$ *to* $\Pi \mathtt{a} : \mathtt{A}.\ \Pi \mathtt{a}' : \mathtt{A}'.\ \Pi \overline{\mathtt{a}} : \overline{\mathtt{A}}(\mathtt{a}, \mathtt{a}').\ \overline{\mathtt{B}}(\overline{\mathtt{a}}, \mathtt{f}(\mathtt{a}), \mathtt{f}'(\mathtt{a}'))$.

For Path types, we exploit the fact that we can convert between non-dependent Path, FPath, and GPath types, which follows from the fact that both types support coercion.

▶ **Lemma 55.** *Over* $([\mathtt{C} : \mathsf{Ty}, \mathtt{c}_0 : \mathtt{C},\ \mathtt{c}_1 : \mathtt{C}], \mathtt{p} : F\mathsf{Path}(\langle \_ \rangle \mathtt{C}, \mathtt{c}_0, \mathtt{c}_1))$, *we have a term* $\mathsf{decode}^F(\mathtt{p}) : \mathtt{c}_0 \sim^{\mathtt{C}} \mathtt{c}_1$.

Proof. We have $F\mathsf{coe} : (\mathtt{A} : (\mathtt{x} : F\mathbb{I}) \to \mathsf{Ty}, \mathtt{y} : F\mathbb{I}, \mathtt{a}_0 : \mathtt{A}(\mathtt{y}), \mathtt{z} : F\mathbb{I}) \Rightarrow \mathtt{A}(\mathtt{z})$, and instantiating with the arguments $(\langle \mathtt{x}\rangle(\mathtt{c}_0 \sim^{\mathtt{C}} \mathtt{p}\ @_F\ \mathtt{x}), F0, (\lambda\_.\mathtt{c}_0), F1)$ yields the desired expression. ◀

▶ **Corollary 56.** *Over* $(\mathtt{C} : F\mathbb{I} \to \mathsf{Ty}, \mathtt{c}_0 : \mathtt{C}(F0))$, $\Sigma \mathtt{c}_1 : \mathtt{C}(F1).F\mathsf{Path}(\mathtt{C}, \mathtt{c}_0, \mathtt{c}_1)$ *is contractible.*

Proof. Applying F to singleton contractibility (Proposition 38), we get over the environment $(\mathtt{C} : F\mathbb{I} \to \mathsf{Ty}, \mathtt{c}_0 : \mathtt{C}(F0))$ a term of type $F\mathsf{isContr}(F\Sigma \mathtt{c}_1 : \mathtt{C}(F1).F\mathsf{Path}(\mathtt{C}, \mathtt{c}_0, \mathtt{c}_1))$. The type formers $F\Sigma$ and $F\Pi$ satisfy the rules for Σ- and Π-types, so we can define equivalences $F\Sigma(A, B) \simeq \Sigma(A, B)$ and $F\Pi(A, B) \simeq \Pi(A, B)$. Combined with Lemma 55, we can therefore derive $\mathsf{isContr}(\Sigma \mathtt{c}_1 : \mathtt{C}(F1).F\mathsf{Path}(\mathtt{C}, \mathtt{c}_0, \mathtt{c}_1))$. ◀

Of course, Corollary 56 also holds when we replace F with G.

▶ **Component 57** (S, path types). *To define* SPath, *we are given* $\mathtt{A} : F\mathbb{I} \to \mathsf{Ty}$ *and* $\mathtt{A}' : G\mathbb{I} \to \mathsf{Ty}$ *with a 1-to-1 correspondence* $\overline{\mathtt{A}} : (\mathtt{i} : F\mathbb{I}, \mathtt{x} : G\mathbb{I}, \mathtt{a} : \mathtt{A}(\mathtt{i}), \mathtt{a}' : \mathtt{A}'(\mathtt{x})) \to \mathsf{Ty}$, *terms* $\mathtt{a}_0 : \mathtt{A}(F0)$ *and* $\mathtt{a}'_0 : \mathtt{A}'(G0)$ *with* $\overline{\mathtt{a}}_{00} : \overline{\mathtt{A}}(F0, G0, \mathtt{a}_0, \mathtt{a}'_0)$, *and terms* $\mathtt{a}_1 : \mathtt{A}(F1)$ *and* $\mathtt{a}'_1 : \mathtt{A}'(G1)$ *with* $\overline{\mathtt{a}}_{11} : \overline{\mathtt{A}}(F1, G1, \mathtt{a}_1, \mathtt{a}'_1)$.

We need to define a 1-to-1 correspondence between $F\mathsf{Path}(\mathtt{A}, \mathtt{a}_0, \mathtt{a}_1)$ *and* $G\mathsf{Path}(\mathtt{A}', \mathtt{a}'_0, \mathtt{a}'_1)$. *We take the relation sending* $\mathtt{p}$ *and* $\mathtt{p}'$ *to the iterated* Σ*-type with components*

$$\begin{aligned}
&\overline{\mathtt{a}}_{10} : \overline{\mathtt{A}}(F1, G0, \mathtt{a}_1, \mathtt{a}'_0).\\
&\overline{\mathtt{a}}_{01} : \overline{\mathtt{A}}(F0, G1, \mathtt{a}_0, \mathtt{a}'_1).\\
&\overline{\mathtt{a}}_{\bullet 0} : F\mathsf{Path}(\langle \mathtt{i}\rangle \overline{\mathtt{A}}(\mathtt{i}, G0, \mathtt{p}\ @_F\ \mathtt{i}, \mathtt{a}'_0), \overline{\mathtt{a}}_{00}, \overline{\mathtt{a}}_{10}).\\
&\overline{\mathtt{a}}_{\bullet 1} : F\mathsf{Path}(\langle \mathtt{i}\rangle \overline{\mathtt{A}}(\mathtt{i}, G1, \mathtt{p}\ @_F\ \mathtt{i}, \mathtt{a}'_1), \overline{\mathtt{a}}_{01}, \overline{\mathtt{a}}_{11}).\\
&\overline{\mathtt{a}}_{0\bullet} : G\mathsf{Path}(\langle \mathtt{x}\rangle \overline{\mathtt{A}}(F0, \mathtt{x}, \mathtt{a}_0, \mathtt{p}'\ @_G\ \mathtt{x}), \overline{\mathtt{a}}_{00}, \overline{\mathtt{a}}_{01}).\\
&\overline{\mathtt{a}}_{1\bullet} : G\mathsf{Path}(\langle \mathtt{x}\rangle \overline{\mathtt{A}}(F1, \mathtt{x}, \mathtt{a}_1, \mathtt{p}'\ @_G\ \mathtt{x}), \overline{\mathtt{a}}_{10}, \overline{\mathtt{a}}_{11}).\\
&\overline{\mathtt{a}}_{\bullet\bullet} : F\mathsf{Path}(\langle \mathtt{i}\rangle G\mathsf{Path}(\langle \mathtt{x}\rangle \overline{\mathtt{A}}(\mathtt{i}, \mathtt{x}, \mathtt{p}\ @_F\ \mathtt{i}, \mathtt{p}'\ @_G\ \mathtt{x}), \overline{\mathtt{a}}_{\bullet 0}\ @_F\ \mathtt{i}, \overline{\mathtt{a}}_{\bullet 1}\ @_F\ \mathtt{i}), \overline{\mathtt{a}}_{0\bullet}, \overline{\mathtt{a}}_{1\bullet}).
\end{aligned} \tag{1}$$

An element consists effectively of a family of witnesses $\overline{\mathtt{a}}_{\bullet\bullet}\ @_F\ \mathtt{i}\ @_G\ \mathtt{x} : \overline{\mathtt{A}}(\mathtt{i}, \mathtt{x}, \mathtt{p}\ @_F\ \mathtt{i}, \mathtt{p}'\ @_G\ \mathtt{x})$ *satisfying* $\overline{\mathtt{a}}_{\bullet\bullet}\ @_F\ F0\ @_G\ G0 \equiv \overline{\mathtt{a}}_{00}$ *and* $\overline{\mathtt{a}}_{\bullet\bullet}\ @_F\ F1\ @_G\ G1 \equiv \overline{\mathtt{a}}_{11}$. *We define* $\mathrm{M}\lambda^{\mathbb{I}}$ *and* $@_{\mathrm{M}}$ *to be abstraction and application of such families.*

It remains to check that this relation is a 1-to-1 correspondence. Fix $\mathtt{p} : F\mathsf{Path}(\mathtt{A}, \mathtt{a}_0, \mathtt{a}_1)$ *and consider the type of pairs of* $\mathtt{p}' : G\mathsf{Path}(\mathtt{A}', \mathtt{a}'_0, \mathtt{a}'_1)$ *with the data* (1). *Given the preceding data, the types of pairs* $(\overline{\mathtt{a}}_{10}, \overline{\mathtt{a}}_{\bullet 0})$ *and* $(\overline{\mathtt{a}}_{1\bullet}, \overline{\mathtt{a}}_{\bullet\bullet})$ *as in* (1) *are dependent* F*-singletons, so*

contractible by Corollary 56. The type of pairs $(\overline{\mathtt{a}}_{01}, \overline{\mathtt{a}}_{0\bullet})$ *is likewise a dependent* G*-singleton and thus contractible. After contracting all of these, we are left with* $\mathtt{p}' : G\mathsf{Path}(\mathtt{A}', \mathtt{a}'_0, \mathtt{a}'_1)$ *and* $\overline{\mathtt{a}}_{0\bullet} : G\mathsf{Path}(\langle \mathtt{x}\rangle \overline{\mathtt{A}}(0, \mathtt{x}, \mathtt{a}_0, \mathtt{p}'\ @_G\ \mathtt{x}), \overline{\mathtt{a}}_{00}, \widetilde{a}_{01})$ *where* $\widetilde{a}_{01}$ *is some expression. The type of such pairs is equivalent to* $G\mathsf{Path}(\langle \mathtt{x}\rangle \Sigma \mathtt{a}' : \mathtt{A}'(\mathtt{x}).\overline{\mathtt{A}}(0, \mathtt{x}, \mathtt{a}_0, \mathtt{a}'), (\mathtt{a}'_0, \overline{\mathtt{a}}_{00}), (\mathtt{a}'_1, \widetilde{a}_{01}))$*, which is a* $G\mathsf{Path}$*-type over a contractible type and thus contractible. A symmetric argument deals with the case where we fix* $\mathtt{p}'$ *and allow* $\mathtt{p}$ *to vary freely.*

▶ **Component 58** (S, universes)**.** *Using the assumption that* $F\mathsf{U} = G\mathsf{U} = \mathsf{U}$*, we interpret the universe* U *by the relation sending* $\mathtt{A} : \mathsf{U}$ *and* $\mathtt{A}' : \mathsf{U}$ *to the type of* U*-valued 1-to-1 correspondences between* $\mathtt{A}$ *and* $\mathtt{A}'$*. That this relation is itself a 1-to-1 correspondence is a consequence of univalence of* U *[39, Theorem 5.8.4(iv)⇒(v)]. We interpret* El*, again using the assumption* $F\mathsf{El}(\mathtt{A}) = G\mathsf{El}(\mathtt{A}) = \mathsf{El}(\mathtt{A})$*, as extracting the 1-to-1 correspondence.*

▶ **Component 59** (S, glue)**.** *To define* SGlue*, we are given inputs*

$$
\begin{array}{lcllcl}
(\mathtt{A}, \mathtt{A}', \overline{\mathtt{A}}) & : & \mathrm{MTy} & \mathtt{e} & : & [\mathtt{P}] \to \mathtt{T} \simeq_F \mathtt{A} \\
(\mathtt{P}, \mathtt{P}', \overline{\mathtt{P}}) & : & \mathrm{MCof} & \mathtt{e}' & : & [\mathtt{P}'] \to \mathtt{T}' \simeq_G \mathtt{A}' \\
\mathtt{T} & : & [\mathtt{P}] \to \mathsf{Ty} & \overline{\mathtt{e}} & : & [\overline{\mathtt{P}}] \to \mathsf{R}_{\simeq}(\overline{\mathtt{A}}, \overline{\mathtt{T}}, \mathtt{e}, \mathtt{e}') \\
\mathtt{T}' & : & [\mathtt{P}'] \to \mathsf{Ty} & & & \\
\overline{\mathtt{T}} & : & ([\overline{\mathtt{P}}], \mathtt{t} : \mathtt{T}, \mathtt{t}' : \mathtt{T}') \to \mathsf{Ty} & & &
\end{array}
$$

where $\overline{\mathtt{A}}$ *and* $\overline{\mathtt{T}}$ *are 1-to-1 correspondences and* $\mathsf{R}_{\simeq}(\overline{\mathtt{A}}, \overline{\mathtt{T}}, -, -)$ *is the 1-to-1 correspondence between* $\mathtt{T} \simeq_F \mathtt{A}$ *and* $\mathtt{T}' \simeq_G \mathtt{A}'$ *given by the span interpretation of* $(- \simeq -)$ *at* $\overline{\mathtt{T}}$ *and* $\overline{\mathtt{A}}$*. We need to define a 1-to-1 correspondence between* $F\mathsf{Glue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \mathtt{e})$ *and* $G\mathsf{Glue}(\mathtt{A}', \mathtt{P}', \mathtt{T}', \mathtt{e}')$*. We take the relation sending* $\mathtt{g} : F\mathsf{Glue}(\mathtt{A}, \mathtt{P}, \mathtt{T}, \mathtt{e})$ *and* $\mathtt{g}' : G\mathsf{Glue}(\mathtt{A}', \mathtt{P}', \mathtt{T}', \mathtt{e}')$ *to*

$$\mathsf{Glue}(\overline{\mathtt{A}}(F\mathsf{unglue}(\mathtt{g}), G\mathsf{unglue}(\mathtt{g}')), \overline{\mathtt{P}}, \overline{\mathtt{T}}(\mathtt{g}, \mathtt{g}'), \widehat{\mathtt{e}})$$

where it remains to define $\widehat{\mathtt{e}} : \overline{\mathtt{T}}(\mathtt{g}, \mathtt{g}') \simeq \overline{\mathtt{A}}(F\mathsf{unglue}(\mathtt{g}), G\mathsf{unglue}(\mathtt{g}'))$ *under* $\overline{\mathtt{P}}$*.*

By the reduction equations for $F\mathsf{unglue}(\mathtt{g})$ *and* $G\mathsf{unglue}(\mathtt{g}')$ *under* $\mathtt{P}$ *and* $\mathtt{P}'$*, the type for* $\widehat{\mathtt{e}}$ *simplifies to* $\overline{\mathtt{T}}(\mathtt{g}, \mathtt{g}') \simeq \overline{\mathtt{A}}(\mathtt{e}.1(\mathtt{g}), \mathtt{e}'.1(\mathtt{g}'))$*. Per the interpretations of* Σ *and* Π *(Components 47 and 54),* $\overline{\mathtt{e}}$ *contains a map* $(\mathtt{t} : \mathtt{T}, \mathtt{t}' : \mathtt{T}') \to \overline{\mathtt{T}}(\mathtt{t}, \mathtt{t}') \to \overline{\mathtt{A}}(\mathtt{e}.1(\mathtt{t}), \mathtt{e}'.1(\mathtt{t}'))$ *as its first component. We take this map, instantiated at* $\mathtt{g}$ *and* $\mathtt{g}'$*, as the forward function of* $\widehat{\mathtt{e}}$*. To see that it is an equivalence, it suffices [29, Theorem 11.1.6] to check that the induced map on total spaces* $(\Sigma \mathtt{t} : \mathtt{T}.\overline{\mathtt{T}}(\mathtt{t}, \mathtt{g}')) \to (\Sigma \mathtt{a} : \mathtt{A}.\overline{\mathtt{A}}(\mathtt{a}, \mathtt{e}'.1(\mathtt{g}')))$ *is an equivalence, as the base map* $\mathtt{e}.1 : \mathtt{T} \to \mathtt{A}$ *is an* F*-equivalence and thus an equivalence. This is the case because* $\overline{\mathtt{A}}$ *and* $\overline{\mathtt{T}}$ *are 1-to-1 correspondences and thus both sides are contractible.*

With this interpretation of Glue*, we can give the interpretations of* glue *and* unglue *as* glue *and* unglue*.*

To interpret suspension, we make essential use of identity types.

▶ **Definition 60.** *Over the environment* $([\mathtt{A} : \mathsf{Ty}, \mathtt{A}' : \mathsf{Ty}], \mathtt{f} : \mathtt{A} \to \mathtt{A}')$*, define the type-valued relation* $\mathsf{Graph}(\mathtt{f}) :\equiv \langle \mathtt{a}, \mathtt{a}'\rangle(\mathtt{f}(\mathtt{a}) \asymp \mathtt{a}') : (\mathtt{a} : \mathtt{A}, \mathtt{a}' : \mathtt{A}') \to \mathsf{Ty}$*.*

For a map $\mathtt{f}$ that is an equivalence, $\mathsf{Graph}(\mathtt{f})$ is a 1-to-1 correspondence. Conversely, a 1-to-1 correspondence $\overline{\mathtt{A}}$ between $\mathtt{A}$ and $\mathtt{A}'$ contains a map $\mathsf{fwd}_{\overline{\mathtt{A}}} : \mathtt{A} \to \mathtt{A}'$ that is an equivalence.

Over the environment $([\mathtt{A} : \mathsf{Ty}, \mathtt{A}' : \mathsf{Ty}], \mathtt{f} : \mathtt{A} \to \mathtt{A}')$, define $\mathsf{map}(\mathtt{f}) : F\mathsf{Susp}(\mathtt{A}) \to G\mathsf{Susp}(\mathtt{A}')$ by $F\mathsf{Susp}$-elimination so that $\mathsf{map}(\mathtt{f})(F\mathsf{north}) = G\mathsf{north}$, $\mathsf{map}(\mathtt{f})(F\mathsf{south}) = G\mathsf{south}$, and $\mathsf{cong}_{\mathsf{map}(\mathtt{f})}(F\mathsf{merid}(\mathtt{a})) \sim G\mathsf{merid}(\mathtt{a}')$. If $f : A \to A'$ is an equivalence, then $\mathsf{map}(f)$ is an equivalence, by the elimination principles for $F\mathsf{Susp}(A)$ and $G\mathsf{Susp}(A')$.

▶ **Component 61** (S, suspension). *Over an environment with* $\mathtt{A}\ \mathtt{A}' : \mathsf{Ty}$ *and a 1-to-1 correspondence* $\overline{\mathtt{A}} : (\mathtt{A}, \mathtt{A}') \to \mathsf{Ty}$*, we must construct a 1-to-1 correspondence between* $F\mathsf{Susp}(\mathtt{A})$ *and* $G\mathsf{Susp}(\mathtt{A}')$*; we take* $\mathsf{Graph}(\mathsf{map}(\mathsf{fwd}_{\overline{\mathtt{A}}}))$*. We interpret* north *and* south *by the reflexive identities* $\mathsf{map}(\mathsf{fwd}_{\overline{\mathtt{A}}})(\mathsf{north}) \asymp \mathsf{north}$ *and* $\mathsf{map}(\mathsf{fwd}_{\overline{\mathtt{A}}})(\mathsf{south}) \asymp \mathsf{south}$*.*

To interpret merid *applied to* $\mathtt{a} : \mathtt{A}$*,* $\mathtt{a}' : \mathtt{A}'$*, and* $\overline{\mathtt{a}} : \overline{\mathtt{A}}(\mathtt{a}, \mathtt{a}')$*, we first convert the path* $\mathsf{cong}_{\mathsf{map}(\mathsf{fwd}_{\overline{\mathtt{A}}})}(F\mathsf{merid}(\mathtt{a})) \sim G\mathsf{merid}(\mathsf{fwd}_{\overline{\mathtt{A}}}(\mathtt{a}))$ *to an identity, then rewrite along the identity* $\mathsf{fwd}_{\overline{\mathtt{A}}}(\mathtt{a}) \asymp \mathtt{a}'$ *obtained from* $\overline{\mathtt{a}}$ *to get an identity* $\mathsf{cong}_{\mathsf{map}(\mathsf{fwd}_{\overline{\mathtt{A}}})}(F\mathsf{merid}(\mathtt{a})) \asymp G\mathsf{merid}(\mathtt{a}')$*. Using* $\mathrm{M}\lambda^{\mathbb{I}}$*, we convert this to an* M*-path in the necessary identity type.*

Now we interpret the eliminator. Over the environment

$$(\mathtt{A} : \mathsf{Ty}, \mathtt{C} : \mathsf{Susp}(\mathtt{A}) \to \mathsf{Ty}, \mathtt{n} : \mathtt{C}(\mathsf{north}), \mathtt{s} : \mathtt{C}(\mathsf{south}), \mathtt{m} : (\mathtt{a} : \mathtt{A}) \to \mathsf{Path}(\langle \mathtt{i} \rangle \mathtt{C}(\mathsf{merid}(\mathtt{a})@\mathtt{i}), \mathtt{n}, \mathtt{s}))$$

we have a type

$$\begin{aligned} \mathsf{D} \quad :\equiv \quad & \Sigma \mathtt{f} : (\Pi \mathtt{t}{:}\mathsf{Susp}(\mathtt{A}).\mathtt{C}(\mathtt{t})).\ \Sigma \mathtt{p_n} : \mathtt{f}(\mathsf{north}) \sim \mathtt{n}.\ \Sigma \mathtt{p_s} : \mathtt{f}(\mathsf{south}) \sim \mathtt{s}. \\ & \mathsf{Path}(\langle \mathtt{j} \rangle \mathsf{Path}(\langle \mathtt{i} \rangle \mathtt{C}(\mathsf{merid}(\mathtt{a})\ @\ \mathtt{i}), \mathtt{p_n}\ @\ \mathtt{j}, \mathtt{p_s}\ @\ \mathtt{j}), (\lambda \mathtt{i}.\mathtt{f}(\mathsf{merid}(\mathtt{a})\ @\ \mathtt{i})), \mathtt{m}) \end{aligned}$$

of dependent functions into C *defined on the constructors* north*,* south*, and* merid *by* n*,* s*, and* m *respectively, up to homotopy. By virtue of the eliminator,* D *is contractible, as are* $F\mathsf{D}$ *and* $G\mathsf{D}$*. Thus every pair of elements from* $F\mathsf{D}$ *and* $G\mathsf{D}$ *is related in* SD*, in particular the pair obtained from* $F\mathsf{elim}$ *and* $G\mathsf{elim}$*. This gives us an almost-interpretation of* elim*: we have an eliminator that may satisfy the point constructor computation rules only up to paths.*

We then correct our almost-interpretation on the point constructors. To interpret the eliminator, we are given type families C *and* $\mathtt{C}'$ *and, over* $(\mathtt{t} : F\mathsf{Susp}(\mathtt{A}), \mathtt{t}' : G\mathsf{Susp}(\mathtt{A}'), \overline{\mathtt{t}} : \mathsf{map}(\mathsf{fwd}_{\overline{\mathtt{A}}})(\mathtt{t}) \asymp \mathtt{t}')$*, 1-to-1 correspondences* $\overline{\mathtt{C}}(\mathtt{t}, \mathtt{t}', \overline{\mathtt{t}}) : (\mathtt{C}(\mathtt{t}), \mathtt{C}'(\mathtt{t}')) \to \mathsf{Ty}$*. We want to relate* $F\mathsf{elim}(\mathtt{C}, \mathtt{n}, \mathtt{s}, \mathtt{m}, \mathtt{t})$ *and* $G\mathsf{elim}(\mathtt{C}', \mathtt{n}', \mathtt{s}', \mathtt{m}', \mathtt{t}'))$ *in* $\overline{\mathtt{C}}(\mathtt{t}, \mathtt{t}', \overline{\mathtt{t}})$ *for all related inputs. We go by* $F\mathsf{Susp}$*-elimination from* t *and identity elimination from* $\overline{\mathtt{t}}$*. For the point cases* $\mathtt{t} = F\mathsf{north}$ *and* $\mathtt{t} = F\mathsf{south}$*, we choose the values that the point computation rules require. For the* $F\mathsf{merid}$ *case, we apply the almost-eliminator to the input data to get a section of* $\overline{\mathtt{C}}$*, evaluate it at the corresponding meridian, then coerce the result along the almost-eliminator's point computation paths to get a path of the correct type.*

This completes the definition of $\mathrm{S}^F_G : \mathbb{C}\mathrm{TT}[\iota\Phi] \to \mathrm{Span}(\mathbb{C}\mathrm{TT}[\iota\Psi])$. In summary:

▶ **Theorem 62.** *Let* $F, G : \mathbb{C}\mathrm{TT}[\iota\Phi] \to \mathbb{C}\mathrm{TT}[\iota\Psi]$ *in the coslice under* $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id},\mathsf{U}} + \mathbb{C}\mathrm{OF}$*. There is a* $\mathrm{S}^F_G : \mathbb{C}\mathrm{TT}[\iota\Phi] \to \mathrm{Span}(\mathbb{C}\mathrm{TT}[\iota\Psi])$ *in* $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}/\mathbf{RMC}$ *such that* $\pi_0\mathrm{S}^F_G \cong F$ *and* $\pi_1\mathrm{S}^F_G \cong G$*.*

6 Conservativity

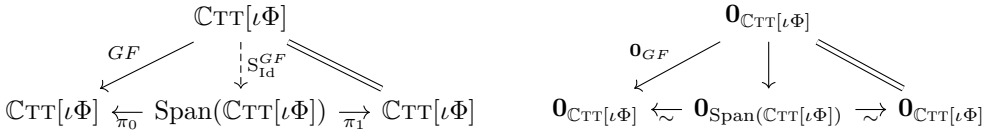
▶ **Proposition 63** (2-out-of-6). *Weak equivalences of democratic models of* $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$ *are closed under 2-out-of-6. That is, given morphisms of democratic models of* $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$ $\mathcal{M} \xrightarrow{\mathcal{F}} \mathcal{N} \xrightarrow{\mathcal{G}} \mathcal{O} \xrightarrow{\mathcal{H}} \mathcal{P}$ *where* $\mathcal{GF}$ *and* $\mathcal{HG}$ *are weak equivalences, the maps* $\mathcal{F}$*,* $\mathcal{G}$*,* $\mathcal{H}$*, and the composite* $\mathcal{HGF}$ *are weak equivalences.*

Proof. See Kapulkin and Lumsdaine [22, Corollary 3.4]. ◀

A corollary of 2-out-of-6 is 2-out-of-3: given composable morphisms $\mathcal{G}$ and $\mathcal{F}$ between democratic models of $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$, if two of the three morphisms $\mathcal{F}$, $\mathcal{G}$, and $\mathcal{GF}$ are weak equivalences, then so is the third.

▶ **Theorem 64.** *For* $F : \mathbb{C}\mathrm{TT}[\iota\Phi] \to \mathbb{C}\mathrm{TT}[\iota\Psi]$ *and* $G : \mathbb{C}\mathrm{TT}[\iota\Psi] \to \mathbb{C}\mathrm{TT}[\iota\Phi]$ *in the coslice under* $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}} + \mathbb{C}\mathrm{OF}$*, the induced morphisms* $\mathbf{0}_F : \mathbf{0}_{\mathbb{C}\mathrm{TT}[\iota\Phi]} \to \mathbf{0}_{\mathbb{C}\mathrm{TT}[\iota\Psi]}$ *and* $\mathbf{0}_G : \mathbf{0}_{\mathbb{C}\mathrm{TT}[\iota\Psi]} \to \mathbf{0}_{\mathbb{C}\mathrm{TT}[\iota\Phi]}$ *are weak equivalences.*

Proof. By Theorem 62, we have an RMC functor fitting in the diagram in $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}/\mathbf{RMC}$ to the left below.



This induces a diagram in $\mathbf{Mod}(\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}})$ as shown to the right, where the morphisms marked $\sim$ are weak equivalences by Proposition 49. By two applications of 2-out-of-3, first in the right triangle and then in the left, it follows that $\mathbf{0}_{GF}$ is a weak equivalence.

By the same argument, $\mathbf{0}_{FG} \colon \mathbf{0}_{\mathbb{C}\mathrm{TT}[\iota\Psi]} \to \mathbf{0}_{\mathbb{C}\mathrm{TT}[\iota\Psi]}$ is a weak equivalence. The claim now follows by 2-out-of-6 applied to the string of morphisms $\mathbf{0}_F \circ \mathbf{0}_G \circ \mathbf{0}_F$. ◀

▶ **Theorem 65** (Conservativity of reversals). *For every self-dual interval theory (Φ, ϕ), the inclusion $\mathbb{C}\mathrm{TT}[\iota\Phi] \to \mathbb{C}\mathrm{TT}[\iota\mathrm{Rev}_\phi\Phi]$ induces a weak equivalence $\mathbf{0}_{\mathbb{C}\mathrm{TT}[\iota\Phi]} \to \mathbf{0}_{\mathbb{C}\mathrm{TT}[\iota\mathrm{Rev}_\phi\Phi]}$ in $\mathbf{Mod}(\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}})$.*

Proof. By Theorem 64 with Theorem 42. ◀

7 Interpreting strict cubical type theory with reversals in spaces

Kapulkin and Lumsdaine [22] show that every democratic model $\mathcal{M} \in \mathbf{Mod}(\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}})$ induces a *fibration category* structure on its category of contexts $\mathcal{M}(\star)$. Such a structure, which is specified by two classes of morphisms in $\mathcal{M}(\star)$ called *fibrations* and *weak equivalences*, induces in turn an $(\infty,1)$-*category* [34] or "homotopy theory". It is in this way that we judge the kind of higher structure described by a model of $\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}}$. The homotopy theory of topological spaces corresponds to one such $(\infty,1)$-category, that of ∞-*groupoids*.

Awodey et al. [6] and Cavallo and Sattler [11] exhibit constructive models $\mathcal{M}$ of strict cubical type theories without reversals whose induced $(\infty,1)$-categories are classically equivalent to the $(\infty,1)$-category of ∞-groupoids. These models are not themselves democratic, so here we mean that their hearts $\mathcal{M}^\heartsuit$ present these $(\infty,1)$-categories in the above sense (Definition 9). Typically, however, these models are analyzed by means of a *Quillen model structure*, another form of presentation of an $(\infty,1)$-category, on $\mathcal{M}$ itself. Such a structure is defined by three classes of maps: *cofibrations*, *weak equivalences*, and *fibrations*.

▶ **Definition 66.** *The* Quillen model structure presented by $\mathcal{M} \in \mathbf{Mod}(\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}})$*, if it exists, is the unique model structure on $\mathcal{M}(\star)$ such that*

(a) *the fibrations are the retracts in $\mathcal{M}(\star)^{\to}$ of context extensions, i.e., of morphisms $p_A \colon \Gamma.A \to \Gamma$ arising as pullbacks in $\mathrm{PSh}(\mathcal{M}(\star))$ of $\mathcal{M}(\pi_{\mathsf{Tm}})$;*

(b) *the unique map $0 \to \Gamma$ is a cofibration for all $\Gamma \in \mathcal{M}(\star)$.*

The uniqueness follows from a result of Joyal [27, Theorem 15.3.1]. A model structure on a category $\mathcal{E}$ induces a fibration category structure on the full subcategory of $X \in \mathcal{E}$ such that $X \to 1$ is a fibration; for a model structure presented by $\mathcal{M} \in \mathbf{Mod}(\mathbb{M}\mathrm{LTT}_{\Sigma,\mathsf{Id}})$, this is exactly $\mathcal{M}^\heartsuit(\star)$, and the induced fibration category is exactly Kapulkin and Lumsdaine's.

Theorem 65 allows us to translate proofs in "opaque" cubical type theory with reversals into proofs that do not use reversals, which can then be interpreted in ∞-groupoids via the aforementioned models. However, it does not allow us to translate proofs in strict cubical type theories. Fortunately, we can also use the twist construction to directly construct models of strict cubical type theory with reversals in ∞-groupoids. In fact, we can reuse existing model constructions of the kind pioneered by Orton and Pitts [26] out of the box.

7.1 Orton–Pitts models

Orton and Pitts [26] give an abstract description of Cohen, Coquand, Huber, and Mörtberg's model of cubical type theory in De Morgan cubical sets [12]. Abstracting from the case of cubical sets, they fix a topos $\mathcal{E}$ equipped with an interval object I and a suitable subobject $\Omega_{\mathrm{cof}} \rightarrowtail \Omega$ of the subobject classifier and isolate axioms on this data sufficient to construct a model of a strict cubical type theory in $\mathcal{E}$ where the interval is interpreted by I and the cofibrations by Ω_{cof}. They assume that the interval I has connections, but Angiuli, Brunerie, Coquand, Harper, Favonia, and Licata (ABCHFL) [2] subsequently gave a similar construction for intervals without such structure. Extracting what we need from their main result and rephrasing in our language, we have:

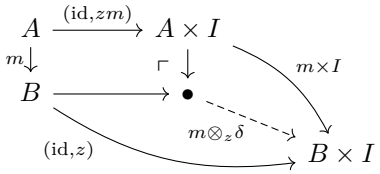
▶ **Proposition 67** ([2, Theorem 2]). *Let $\mathcal{E} = \mathrm{PSh}(\mathcal{C})$ be a presheaf category on a finite product category $\mathcal{C}$, let $I \in \mathcal{E}$ be a representable object with distinct points $0, 1 \colon 1 \to I$, and let $\Omega_{\mathrm{cof}} \rightarrowtail \Omega_{\mathrm{dec}}$ be a subobject of the levelwise decidable subobject classifier in $\mathrm{PSh}(\mathcal{C})$ that classifies the diagonal $I \to I \times I$ and is closed under finite conjunction, finite disjunction, and universal quantification over I. Then there is a model $\mathcal{M}$ of $\mathbb{C}\mathrm{TT}_s$ such that*

(a) $\mathcal{M}(\star) = \mathcal{E}$.
(b) $\mathcal{M}(\mathbb{I}) = \yoneda I \in \mathrm{PSh}(\mathcal{E})$.
(c) *the maps $p_A \colon \Gamma.A \to \Gamma$ arising as pullbacks in $\mathrm{PSh}(\mathcal{E})$ of $\mathcal{M}(\pi_{\mathsf{Tm}})$ are those equipped with a* diagonal Kan composition structure *[2, Definition 1].*

This model interprets the interval theory of all $f \colon I^n \to I$ in $\mathcal{C}$ and equations between them.

We call $(\mathcal{C}, I, 0, 1, \Omega_{\mathrm{cof}})$ satisfying the conditions of Proposition 67 an *ABCHFL setup* and write $\mathcal{M}(\mathcal{C}, 0, 1, I, \Omega_{\mathrm{cof}})$ for the resulting model. The maps with diagonal Kan composition structure can be described by a simple lifting property. Awodey proves the following for cartesian cubical sets, but the same proof applies in the setting of Proposition 67.

▶ **Proposition 68** ([5, Proposition 4.15(2)⇔(3)]). *Let $(\mathcal{C}, I, 0, 1, \Omega_{\mathrm{cof}})$ be an* ABCHFL setup. *A morphism $f \colon Y \to X$ admits a diagonal Kan composition structure if and only if it has the right lifting property against the unique dashed map*



for every $m \colon A \rightarrowtail B$ classified by Ω_{cof} and $z \colon B \to I$.

We call a map an $(I, \Omega_{\mathrm{cof}})$-*fibration* when it satisfies the property in Proposition 68. As a lifting property, it is closed under retracts [27, Lemma 11.1.4].

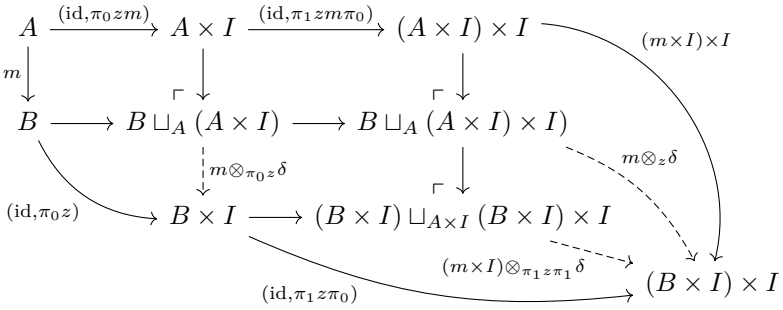
▶ **Proposition 69.** *If $(\mathcal{C}, I, 0, 1, \Omega_{\mathrm{cof}})$ is an ABCHFL setup, then $(\mathcal{C}, I \times I, r, s, \Omega_{\mathrm{cof}})$ is an ABCHFL setup for every $r \neq s \colon 1 \to I \times I$. Moreover, the classes of $(I \times I, \Omega_{\mathrm{cof}})$- and $(I, \Omega_{\mathrm{cof}})$-fibrations coincide.*

Proof. Because $\mathcal{C}$ is a finite product category by assumption, $I \times I$ is also representable. The diagonal $\Delta_{I \times I} \colon I \times I \to (I \times I) \times (I \times I)$ is the conjunction of $(\pi_0 \times \pi_0)^* \Delta_I$ and $(\pi_1 \times \pi_1)^* \Delta_I$, the pullbacks of $\Delta_I \colon I \to I \times I$ along the projections $\pi_0 \times \pi_0, \pi_1 \times \pi_1 \colon (I \times I) \times (I \times I) \to I \times I$. Thus it is classified by Ω_{cof}. Universal quantification of $I \times I$ is iterated universal quantification over I, so Ω_{cof} is closed under this operation. This completes the proof of the first claim.

For the second claim, recall that the class of maps with the left lifting property against a map is closed under retracts, composition, and pushouts along arbitrary maps [27, Lemma 11.1.4]. Given $m\colon A \rightarrowtail B$ classified by Ω_{cof} and $z\colon B \to I$, we can write $m \otimes_z \delta$ as a retract of $m \otimes_{z\times 0} \delta$ where $z \times 0\colon B \to I \times I$:

$$\begin{array}{ccccc} B \sqcup_A (A \times I) & \longrightarrow & B \sqcup_A (A \times (I \times I)) & \longrightarrow & B \sqcup_A (A \times I) \\ \downarrow{\scriptstyle m\otimes_z\delta} & & \downarrow{\scriptstyle m\otimes_{z\times 0}\delta} & & \downarrow{\scriptstyle m\otimes_z\delta} \\ B \times I & \xrightarrow[B\times(I\times 0)]{} & B \times (I \times I) & \xrightarrow[B\times\pi_0]{} & B \times I. \end{array}$$

Thus every $(I \times I, \Omega_{\mathrm{cof}})$-fibration is an $(I, \Omega_{\mathrm{cof}})$-fibration. For the converse, given $m\colon A \rightarrowtail B$ classified by Ω_{cof} and $z\colon B \to I \times I$, we read off the commutative diagram



that $m \otimes_z \delta$ is a composite of a pushout of $m \otimes_{\pi_0 z} \delta$ and $(m \times I) \otimes_{\pi_1 z \pi_1} \delta$. This semantic counterpart to Component 32 shows that all $(I, \Omega_{\mathrm{cof}})$-fibrations are $(I \times I, \Omega_{\mathrm{cof}})$-fibrations. ◀

Using Proposition 69, we can turn any ABCHFL model that presents ∞-groupoids into an ABCHFL model with reversals that also presents ∞-groupoids. A suitable input is the category of presheaves on the *semilattice cube category* $\square_\vee$, i.e., the cartesian cube category with one connection. This is the algebraic theory generated by an object I with points $0, 1\colon 1 \to I$ and a connection $\vee\colon I \times I \to I$ satisfying the axioms of a bounded join-semilattice.

▶ **Proposition 70** ([11, Theorems 4.34 & 7.8])**.** *The ABCHFL model $\mathcal{M}(\square_\vee, I, 0, 1, \Omega_{\mathrm{dec}})$ presents a Quillen model structure. Assuming classical logic, this model structure presents the $(\infty, 1)$-category of ∞-groupoids.*

▶ **Theorem 71.** *The ABCHFL model $\mathcal{M}(\square_\vee, I \times I, (0, 1), (1, 0), \Omega_{\mathrm{dec}})$ interprets strict cubical type theory with reversals and presents a Quillen model structure. Assuming classical logic, this model structure presents the $(\infty, 1)$-category of ∞-groupoids.*

Proof. Propositions 67 and 69 give the existence of the model of type theory. The existence of the model structure can be established by following Awodey's construction [5], but he considers only the cartesian cube category with its canonical interval object. The necessary components appear in generality in Awodey et al. [6, Lemma 3.7.2, Proposition 3.7.3] (compare [6, Theorem 4.4.9]). By Proposition 69 and the uniqueness of the model structure presented by a model of type theory, this model structure is the same as that presented by $\mathcal{M}(\square_\vee, I, 0, 1, \Omega_{\mathrm{dec}})$, so Proposition 70 proves the final claim. ◀

Although the original interval I in $\square_\vee$ has a connection, this is not the case for $I \times I$ with the endpoints $(0, 1)$ and $(1, 0)$: in order to give the definition $(i_0, i_1) \vee (j_0, j_1) \coloneqq (i_0 \vee j_0, i_1 \wedge j_1)$ from § 1.1, we need *both* connections in the base model. Thus we only model cubical type

theory with a reversal, not with a connection. We expect that we can apply the procedure in this section to the second author's model of cubical type theory with two connections [32], even though it is not an ABCHFL model, but we leave this to future work.

References

1 Carlo Angiuli. *Computational Semantics of Cartesian Cubical Type Theory*. PhD thesis, Carnegie Mellon University, 2019. doi:10.1184/R1/16860013.

2 Carlo Angiuli, Guillaume Brunerie, Thierry Coquand, Robert Harper, Kuen-Bang Hou (Favonia), and Daniel R. Licata. Syntax and models of Cartesian cubical type theory. *Mathematical Structures in Computer Science*, 31(4), 2021. doi:10.1017/S0960129521000347.

3 Carlo Angiuli, Kuen-Bang Hou (Favonia), and Robert Harper. Cartesian cubical computational type theory: Constructive reasoning with paths and equalities. In *27th EACSL Annual Conference on Computer Science Logic, CSL 2018, September 4-7, 2018, Birmingham, UK*, pages 6:1–6:17, 2018. doi:10.4230/LIPIcs.CSL.2018.6.

4 Steve Awodey. Natural models of homotopy type theory. *Mathematical Structures in Computer Science*, 28(2):241–286, 2018. doi:10.1017/S0960129516000268.

5 Steve Awodey. *Cartesian Cubical Model Categories*. Springer Nature Switzerland, 2026. doi:10.1007/978-3-032-08730-0.

6 Steve Awodey, Evan Cavallo, Thierry Coquand, Emily Riehl, and Christian Sattler. The equivariant model structure on cartesian cubical sets. *Advances in Mathematics*, 495:110965, 2026. doi:10.1016/j.aim.2026.110965.

7 Rafaël Bocquet. Towards coherence theorems for equational extensions of type theories, 2023. doi:10.48550/arXiv.2304.10343.

8 Ulrik Buchholtz and Edward Morehouse. Varieties of cubical sets. In *Relational and Algebraic Methods in Computer Science - 16th International Conference, RAMiCS 2017, Lyon, France, May 15-18, 2017, Proceedings*, pages 77–92, 2017. doi:10.1007/978-3-319-57418-9_5.

9 Evan Cavallo and Robert Harper. Higher inductive types in cubical computational type theory. *Proceedings of the ACM on Programming Languages*, 3(POPL):1:1–1:27, 2019. doi:10.1145/3290314.

10 Evan Cavallo, Anders Mörtberg, and Andrew W. Swan. Unifying cubical models of univalent type theory. In *28th EACSL Annual Conference on Computer Science Logic, CSL 2020, January 13-16, 2020, Barcelona, Spain*, volume 152 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CSL.2020.14.

11 Evan Cavallo and Christian Sattler. Relative elegance and cartesian cubes with one connection. *Canadian Journal of Mathematics*, pages 1–64, 2025. doi:10.4153/S0008414X25101466.

12 Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. Cubical type theory: A constructive interpretation of the univalence axiom. In *21st International Conference on Types for Proofs and Programs, TYPES 2015, May 18-21, 2015, Tallinn, Estonia*, pages 5:1–5:34, 2015. doi:10.4230/LIPIcs.TYPES.2015.5.

13 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, 2018. doi:10.1145/3209108.3209197.

14 Thierry Coquand, Simon Huber, and Christian Sattler. Canonicity and homotopy canonicity for cubical type theory. *Logical Methods in Computer Science*, 18(1):28:1–28:35, 2022. doi:10.46298/lmcs-18(1:28)2022.

15 Peter Dybjer. Inductive families. *Formal Aspects of Computing*, 6(4):440–465, 1994. doi:10.1007/BF01211308.

16 Peter Dybjer. Internal type theory. In Stefano Berardi and Mario Coppo, editors, *Types for Proofs and Programs: International Workshop, TYPES '95 Torino, Italy, June 5–8, 1995 Selected Papers*, pages 120–134. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996. doi:10.1007/3-540-61780-9_66.

17 Manuel Fidel and Diana Brignole. Estudio algebraico de ciertas lógicas no clásicas mediante producto de álgebras. In *Actas del I Congreso Dr. Antonio A. R. Monteiro, Bahía Blanca*, pages 23–38, 1991. URL: https://www.inmabb-conicet.gob.ar/static/publicaciones/actas/1/03_Brignole.pdf.

18 Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *Journal of the ACM*, 40(1):143–184, 1993. doi:10.1145/138027.138060.

19 Simon Huber. Canonicity for cubical type theory. *Journal of Automated Reasoning*, 63(2):173–210, 2018. doi:10.1007/s10817-018-9469-1.

20 Valery Isaev. Morita equivalences between algebraic dependent type theories, 2020. arXiv:1804.05045.

21 Krzysztof Kapulkin and Yufeng Li. Extensional concepts in intensional type theory, revisited. *Theoretical Computer Science*, 1029:115051, 2025. doi:10.1016/j.tcs.2024.115051.

22 Krzysztof Kapulkin and Peter LeFanu Lumsdaine. The homotopy theory of type theories. *Advances in Mathematics*, 337:1–38, 2018. doi:10.1016/j.aim.2018.08.003.

23 Marcus Kracht. On extensions of intermediate logics by strong negation. *Journal of Philosophical Logic*, 27(1):49–73, 1998. doi:10.1023/a:1004222213212.

24 Per Martin-Löf. An intuitionistic theory of types: predicative part. In H.E. Rose and J.C. Shepherdson, editors, *Logic Colloquium '73*, volume 80 of *Studies in Logic and the Foundations of Mathematics*, pages 73–118. North-Holland, 1975. doi:10.1016/S0049-237X(08)71945-1.

25 Anders Mörtberg and Loïc Pujet. Cubical synthetic homotopy theory. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, pages 158–171, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3372885.3373825.

26 Ian Orton and Andrew M. Pitts. Axioms for modelling cubical type theory in a topos. *Logical Methods in Computer Science*, 14(4), 2018. doi:10.23638/LMCS-14(4:23)2018.

27 Emily Riehl. *Categorical Homotopy Theory*. Cambridge University Press, 2014. doi:10.1017/cbo9781107261457.

28 Emily Riehl and Michael Shulman. A type theory for synthetic ∞-categories. *Higher Structures*, 1(1):116–193, 2017. doi:10.21136/HS.2017.06.

29 Egbert Rijke. *Introduction to Homotopy Type Theory*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2025. doi:10.1017/9781108933568.

30 Umberto Rivieccio. Representation of De Morgan and (semi-)Kleene lattices. *Soft Computing*, 24(12):8685–8716, 2020. doi:10.1007/s00500-020-04885-w.

31 Christian Sattler. Do cubical models of type theory also model homotopy types, 2018. Lecture at the Hausdorff Trimester Program: Types, Sets and Constructions. URL: https://www.youtube.com/watch?v=wkPDyIGmEoA.

32 Christian Sattler. A constructive ∞-groupoid model of homotopy type theory. Invited talk at TYPES 2025. Slides: https://msp.cis.strath.ac.uk/types2025/slides/TYPES2025-slidesSattler.pdf, video: https://youtu.be/eV9y6I2QHEk, 2025.

33 Thomas Streicher and Jonathan Weinberger. Simplicial sets inside cubical sets. *Theory Appl. Categ.*, 37(10):276–286, 2021. doi:10.70930/tac/ob3pmmyi.

34 Karol Szumiło. Frames in cofibration categories. *Journal of Homotopy and Related Structures*, 12(3):577–616, 2016. doi:10.1007/s40062-016-0139-x.

35 Nicolas Tabareau, Éric Tanter, and Matthieu Sozeau. The marriage of univalence and parametricity. *Journal of the ACM*, 68(1):1–44, 2021. doi:10.1145/3429979.

36 The Agda Community. Cubical Agda Library. URL: https://github.com/agda/cubical.

37 Taichi Uemura. *Abstract and Concrete Type Theories*. PhD thesis, University of Amsterdam, 2021. URL: https://eprints.illc.uva.nl/id/eprint/2195/.

38 Taichi Uemura. A general framework for the semantics of type theory. *Mathematical Structures in Computer Science*, 33(3):134–179, 2023. doi:10.1017/s0960129523000208.

39 The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. The Univalent Foundations Program, Institute for Advanced Study, 2013.

40 Dimiter Vakarelov. Notes on N-lattices and constructive logic with strong negation. *Studia Logica*, 36(1-2):109–125, 1977. doi:10.1007/bf02121118.

41 Andrea Vezzosi, Anders Mörtberg, and Andreas Abel. Cubical Agda: A dependently typed programming language with univalence and higher inductive types. *Journal of Functional Programming*, 31, 2021. doi:10.1017/s0956796821000034.