The Category Interpretation of Directed Type Theory

Thorsten Altenkirch
University of Nottingham
Nottingham, United Kingdom
thorsten.altenkirch@nottingham.ac.uk

Jacob Neumann
University of Nottingham
Nottingham, United Kingdom
jacob.neumann@nottingham.ac.uk

Abstract

The field of *directed type theory* seeks to design type theories capable of reasoning synthetically about (higher) categories, by generalizing the symmetric *identity* types of Martin-Löf Type Theory to asymmetric *Hom-types*. So far, the main approaches to directed type theory proceed in a "somewhat indirect" manner: axiomatizing the directed interval type and defining the machinery of directed type theory in terms of it. In this paper, we take the first step towards a 'directed homotopy type theory *without* the directed interval' by developing a directed analogue of Hofmann and Streicher's groupoid model—the *category model*. The directed type theory this models is adequate for synthetic 1-category theory; the extent to which this approach extends to higher dimensions is the subject of ongoing investigation. Moreover, since the semantics of this theory are expressed in terms of *categories with families*, these results promise to interface elegantly with current research into generalized algebraic theories and (higher) observational type theory.

Keywords

semantics, directed type theory, homotopy type theory, category theory

ACM Reference Format:
Thorsten Altenkirch and Jacob Neumann. 2024. The Category Interpretation of Directed Type Theory. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 12 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

One of the central constructs of Martin-Löf Type Theory (MLTT) [22, 23] are its *identity types*: for any given terms t,t' of the same type, we have a type Id(t,t') encoding the proposition that t and t' are identical; the elements of this type (if there are any) are proofs that t equals t'. Because the formation of identity types can be iterated—for p,q: Id(t,t') we can form the type Id(p,q), and so on—a natural question arose: are identity proofs unique? That is, given two terms of type Id(t,t'), i.e. two proofs, p and q, that t and t' are identical, can we always construct a proof that p and q must also be identical?

One of the most important results in the history of type theory is Hofmann and Streicher's proof [16] answering this question in the negative: the *uniqueness of identity proofs* principle (UIP) is independent of the rules of Martin-Löf Type Theory. To show that UIP (or its equivalent formulation, Streicher's Axiom K) is not provable, they construct a countermodel, the *groupoid model of type theory*, which models all the rules of MLTT but violates UIP. While such an independence result is, on its own, quite remarkable, this paper also is notable for setting into motion several lines of thought which would later be absolutely central to homotopy type theory. Perhaps most significantly, it contains the statement of (a special case of) Voevodsky's univalence axiom, prefiguring the observation that univalence is necessary for managing iterated identities in the absence of UIP.

Another essential insight stemming from this work is the notion of *types as synthetic (higher) groupoids*. Like any reasonable notion of 'identity', identity in MLTT is reflexive, symmetric, and transitive. This means that every term t comes with a term refl_t : Id(t,t), that identity proofs of type Id(t,t') can be 'inverted' to get proofs of type Id(t',t), and that we can 'compose' terms of Id(t,t') with terms of Id(t',t'') to get terms of type Id(t,t''). Mathematically, this is the data of a higher groupoid: the objects of the groupoid are the terms of the type and Id(t,t') is the collection of morphisms from t to t'. It is a *higher* groupoid (a weak ω-groupoid) because of the iteration of identity types: Id(t,t') is itself a type, and therefore has the groupoidal structure of identity between *its* terms, and those identities have identities between them, and so on. This creates an exciting possibility: that one can use the language of MLTT (which is essentially a programming language, and hence relatively tractable) to reason directly about weak ω-groupoids (which are difficult to define and work with using standard mathematical foundations). This *synthetic theory of higher groupoids* (and the closely-related

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference'17, July 2017, Washington, DC, USA
© 2024 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

synthetic homotopy theory) serves as a key motivation for homotopy type theory.

The usual way of defining a groupoid is as a category where every morphism is invertible. Thus, categories are a *generalization* of groupoids, in the same way that monoids generalize groups and that preorders generalize equivalence relations. There are homotopy type theoretic treatments of category theory (e.g. [2]), but the 'categories' in such a theory are a very different *kind* of thing than the above-mentioned weak ω-groupoids: the former is *analytic*, i.e. the structure of a category must be meticulously defined in terms of the type theory, conditions like associativity and functoriality must be checked by hand, and it remains very difficult to define higher categories. This is a rather peculiar: why would groupoids and categories be entirely different kinds of structures? Is it possible to rectify this, and generalize homotopy type theory such that types are synthetic higher *categories*? This question is the impetus for *directed type theory*. In a directed (homotopy) type theory, the identity types of MLTT are replaced by *Hom-types*; the terms of type Hom(t,t')—which we'll call *directed paths* from t to t'—still include refl and are closed under composition (corresponding to the identity morphisms and composition in a category), but are *not* assumed to be symmetric/invertible. Such a theory would subsume ordinary, *undirected* type theory and, presumably, include operations for constructing groupoids out of categories, corresponding to the 'localization' and 'core' functors that are left- and right-adjoint (respectively) to the inclusion of groupoids into categories.

However, the project of articulating directed type theory has proven rather difficult. In particular, a number of issues arise relating to *variance*. In undirected type theory, there is no concern over whether terms or types depend on each other *positively* (i.e. covariantly) or *negatively* (contravariantly)—this distinction is invisible in the presence of symmetry. In directed type theory, however, it must be dealt with. Also, it's not obvious which aspects of undirected type theory ought to be 'made directed' and which ought to be left alone. For instance, the groupoid model is so named because it uses groupoids in two key aspects: it interprets contexts as groupoids, and it interprets types as families of groupoids. To define its directed analogue, the *category model*, should we replace groupoids with categories in both instances? Or just one? The literature already includes a variety of different directed type theories, whose features, differences, and relationships are still to be fully understood.

1.1 Related Work

1.1.1 Directed Type Theories We can categorize the various directed type theories according to (i) whether or not they adopt a modal typing discipline to control variance, and, among those that do, (ii) the 'depth' of the modalities and (iii) how many 'dimensions' of directed paths are asserted judgmentally. The various directed type theories also differ in terms of their semantics: some do not have semantics, and those that do adopt different notions of 'model'.

The first work in modern directed type theory is the theory of Licata and Harper [21]. This is a directed type theory which treats variances modally: any variable must be tagged with a *polarity*, either 'positive' or 'negative', and this restricts how the variable may be used. For instance, in the term λx.M : Π(x:A).B, the variable x must bear a negative marking (written x:A^-), since it appears in a negative position (the domain of a function). Moreover, this theory is what we'll call "*deeply-polarized*", meaning there's an operation of 'context negation': for each context Γ, there is a context Γ^{op} containing the same variables as Γ but with the opposite polarity. Finally, this theory is, to use Licata and Harper's terminology, *2-dimensional*, because it has syntax for directed reductions between parallel substitutions. This is interpreted inside the 2-category of categories: contexts are categories, substitutions are functors, and reductions between substitutions are interpreted as natural transformations. The negation operation on contexts is interpreted as the 'opposite category' construction, which fits nicely with the notion that 'negative' means 'contravariant'. However, this interpretation is somewhat *ad-hoc*: the authors do not identify a general model theory of which this is an instance.

Nuyts [25] carries on in a similar vein, but with some important changes. In particular, he notes that the Licata–Harper theory cannot accommodate Martin-Löf identity types without them coinciding with (and therefore enforcing symmetry upon) the directed reductions. Thus, the Licata–Harper theory is expanded to include "isovariance" and "invariance", two further polarities. This theory is much richer and more capable of offering a directed analogue to the various constructions of the HoTT Book [28].

There is, however, one significant way in which the theories of Licata-Harper and Nuyts do not provide the directed analogue of higher-groupoids-as-iterated-identity-types: the Homs in these theories are not Hom-types, but rather Hom-judgments. This means that, in these theories, we cannot iterate Hom: the Licata-Harper theory has exactly two dimensions, because that is how many were added by hand. Noticing this drawback, North [24] outlines a directed type theory which is *not* deeply-polarized, but is *shallowly-polarized*: given a type A, we can form the types A^{op} and A^{core}, representing contravariant and isovariant terms of A, but we do not have similar operations on contexts. North's theory is higher-dimensional in the sense that Hom-formation can be iterated arbitrarily, but is 1-dimensional in the sense of Licata-Harper. Similar to Licata-Harper, North provides an interpretation of the syntax, but not a general model theory.

Ahrens et al. [3] do situate their type theory within a general model theory, namely *comprehension bicategories*. Their theory is again 2-dimensional—directed reductions between substitutions being a key desideratum of theirs. However, this theory does not yet include a system for dealing with variances nor a Hom-type former—the authors promise that these features will be included in future work.

Finally, there are several directed type theories which sidestep the need for an explicit modal typing system for variances. This is essentially achieved by building an undirected base theory which is adequate for axiomatizing the directed interval, and then defining directed type theory in terms of this directed interval. Riehl and Shulman [26] give a directed simplicial type theory in this style, yielding a synthetic theory of ∞-categories. This is among the most mature of directed type theories: it includes a statement of directed univalence, has been extended to include modalities like the opposite category [31], and has recently been implemented in a computer proof assistant, Rzk [19]. Similarly, Weaver and Licata [30] explore the directed analogue of *cubical type theory*, and their theory has also been implemented in a computer proof assistant [20]. In the present work, our focus will be on developing a more elementary semantics which models directed type theory in a more immediate way, and thus won't be focusing to closely on such theories. However, it remains an interesting question to compare them to our approach, and understand the extent to which these theories achieve similar results. Also, since these theories (particularly Riehl-Shulman) have already succeeded at synthetically capturing higher categories, they provide valuable insight for how to generalize our theory to higher dimensions.

1.1.2 Categories with Families and Generalized Algebraic Theories Categories with families (CwFs) were introduced as the semantics of type theory by Dybjer and Hofmann [11, 15], they are a special case of a generalised algebraic theory (GAT) [9]. CwFs are type-theoretic in nature: they separate contexts and types as in the syntax of type theory, giving rise to a more intuitive notion of semantics compared with locally cartesian closed categories (LCCCs). Indeed, many interesting models of type theory (such as the groupoid model) are *not* LCCCs, but are CwFs. Another aspect is that CwF come with explicit choices for all constructions, while LCCCs use universal properties, which only give constructions up to isomorphism. However, Hofmann has showed that we can construct a CwF for each LCCC via an adjoint splitting construction [14], which has been shown to give rise to a biequivalence [10].

There are a number of interesting alternatives to CwFs such as split comprehension categories [17] or natural models [8] which have a more categorical as opposed to a type-theoretic flavor. However, they can be shown to be equivalent to CwFs.

In the context of CwFs, the so-called *initiality conjecture*, i.e. that the syntax of type theory gives rise to the initial CwF, has a simple solution when working in homotopy type theory: the initial CwF can be presented as a Quotient-Inductive-Inductive Type (QIIT) [5]. This definition can be implemented in cubical Agda, using a technique pioneered by Szumi [18]. However, this *solution* to the initiality problem presumes a rich type theory which itself needs to be justified semantically.

1.2 Contribution

In the present work, we define a directed analogue of the groupoid model. We dub this model 'the category model' because it replaces groupoids with categories, in two ways. First, the actual definition of the model interprets contexts as categories and types as families of categories, instead of groupoids and families of groupoids. Second, the theory of this model is a directed type theory, and thus provides a language for synthetic category theory.

To use the categorization of the previous subsection, the directed type theory given here (i) uses a modal typing discipline; (ii) is deeply-polarized; and (iii) is 1-dimensional. Our 'calculus of polarity' will be somewhat like that of Licata–Harper, in that it will allow for the negation of contexts and 'negative context extension', and interpret these using the 'opposite category' operation. However, while Licata and Harper make context- and type-negation inextricable, our polarity calculus will treat them as separate, but closely-related operations.[1]

Our theory will be 1-dimensional in the sense of Licata–Harper, because it will not include syntax for directed reductions between parallel substitutions. Instead, we will use the same Hom-type former as North, which has the key advantage of being iterable. However, while North uses a third polarity, *core types*, to be able to type the identity morphism and state directed path induction, a more versatile solution is available in our semantics: restricting to neutral contexts. What this means is that, while we still work in the category model (where contexts are categories), we only define directed path induction in those contexts which are groupoids. In a neutral context, we obtain principles of directed path induction (similar to North's, modulo the aforementioned change), which, as Riehl and Shulman observe, can be seen as an instance of the Yoneda Lemma. We leave it for future

[1] In the terminology of [7], Licata and Harper's theory is *abstractly polarized*, whereas ours will be *concretely polarized*, which is slightly stronger.

work to develop some system of *zoned contexts*, which reconciles the deep polarity of the category model with our need to work in neutral contexts for directed path induction.

Finally, we begin to demonstrate that this theory is suitable for synthetic category theory, by constructing the composition of directed paths. Unlike in *analytic* category theory, where such constructions must be done explicitly by hand, this construction is given automatically. In future work, we will show that this same phenomenon repeats for the topics of functoriality and naturality. Like the groupoid model, the category model does not satisfy (the appropriate analogue of) UIP—otherwise all the Hom-types would be subsingletons, and we would instead have a language for synthetic *preorder* theory—but it does satisfy UIP "one level up": directed paths between directed paths *are* unique. This theory, therefore, is a language for synthetic *1*-category theory, not higher category theory. But, just as Hofmann and Streicher's work prefigured, but did not achieve, homotopy type theory, we view this as a promising first step towards a future *directed homotopy type theory*.

1.3 Metatheory and Notation

Throughout, we use dependent type theory as our metatheory, writing $=$ to mean definitional or judgmental equality. In a few cases, we use $\equiv$ as our metatheoretic propositional equality, which we take to satisfy UIP. To express dependent functions, we'll use Agda-like notation, including the use of curly braces to indicate implicit arguments which can be omitted. We do not distinguish between curried and uncurried functions, writing arguments either separated by spaces or commas, as convenient. To introduce infix functions, we use underscores to indicate where arguments can be placed.

We make extensive use of basic category-theoretic notions. For a category Γ, we write $|\Gamma|$ to indicate the type of objects of Γ, and, for $\gamma_0, \gamma_1 : |\Gamma|$, write $\Gamma\ [\gamma_0, \gamma_1]$ to denote the set of Γ-morphisms whose domain is γ_0 and whose codomain is γ_1. We won't pay attention to matters of *size*, i.e. whether a given collection constitutes a "small set".

We'll make a number of category-theoretic definitions using snippets of pseudo-Agda. For instance, given categories Δ, Γ, we might define the hom-sets of the category Cat, i.e. the set of functors from Δ to Γ, as follows.

```
record Cat [ Δ, Γ ] : Set where
  field
  obj : |Δ| → |Γ|
  map : {δ₀ δ₁ : |Δ|} → Δ [ δ₀, δ₁ ] → Γ [ obj δ₀, obj δ₁ ]
  fid : {δ : |Δ|} → map (id_δ) ≡ id_obj(δ)
  fcomp : {δ₀ δ₁ δ₂ : |Δ|}{δ₀₁ : Δ [ δ₀, δ₁ ]}{δ₁₂ : [ δ₁, δ₂
      ]}
        → map (δ₁₂ ∘ δ₀₁) ≡ (map δ₁₂) ∘ (map δ₀₁)
```

We then implement some σ: Cat $[\Delta, \Gamma]$ by defining σ.obj, σ.map, etc. We'll take the appropriate extensionality principles (e.g. to prove $\sigma = \sigma'$, show $\sigma.\mathrm{obj}\ \delta = \sigma'.\mathrm{obj}\ \delta$ for all δ, etc.) for granted. In the code snippets themselves, we'll be more pedantic about writing the components σ.obj, σ.map, etc., but in the main body of the text we'll just write σ for both.

2 Polarity in the Category Model

As discussed above, the central impetus for the groupoid model was to serve as a *countermodel*, exhibiting that the rules of MLTT cannot possibly prove UIP. Of course, in order to speak of countermodels, one must first have a notion of "model". What *kind of thing* is the groupoid model? The notion of model utilized by Hofmann and Streicher is that of categories with families.[2] One benefit of CwFs (in addition to those listed in 1.1.2) is their modularity: the basic definition of CwF includes exactly enough mathematical structure to model the fundamental mechanisms of a type theory (contexts, substitution, types, terms, and variables), but no more. Any constructs of type theory one wants to study (such as dependent types and identity types) must be constructed *upon* this structural foundation. If we had a specific syntax of directed type theory already in mind, then we could proceed to encode all the desired type- and term-formers in the CwF framework. But we don't—instead we're letting the semantics take the lead, and determining our syntax from there. So we'll define the category model as a CwF, and then explore what further type-theoretic constructs it interprets.

The four basic components of a CwF are the category of contexts and substitutions, the presheaf of types, the presheaf of terms, and the operation of context extension. These are spelled out in pseudo-Agda in Fig. 1. Our presentation is essentially the generalized algebraic theory of CwFs given in [11, Section 2.2], though for brevity we've omitted several specifications, e.g. that Con and Sub form a category, the functoriality of the __[__] operators, that the morphism pairing operation $\langle \_\_, \_\_ \rangle$ is compatible with the composition of substitutions, etc. One condition we will highlight is the 'local representability' condition tying together Ty, Tm, and the $\triangleright$ operation.

$$\mathsf{Sub}\ \Delta\ (\Gamma \triangleright \mathsf{A}) \cong \sum_{\sigma :\ \mathsf{Sub}\ \Delta\ \Gamma} \mathsf{Tm}(\Delta, \mathsf{A}[\sigma]) \qquad \text{(Local-Rep)}$$

This isomorphism is natural in both its arguments, i.e. viewing both sides as either presheaves in Δ or as covariant functors in $(\Gamma, \mathsf{A}) : \int \mathsf{Ty}$. The right-to-left direction of this bijection is the pairing operation of Fig. 1, and the left-to-right

[2] See [15, Section 3.2] for a comparison between CwFs and other notions of model.

direction sends τ to $(\mathsf{p} \circ \tau, \mathsf{v}[\mathsf{p} \circ \tau])$. Intuitively, we think of contexts as *lists of typed variable declarations*, and context extension is the operation appending one more variable onto the end of the context. To define a substitution from Δ to Θ is to "implement" the variables in Θ as appropriately-typed terms, using the variables in Δ. The meaning of this is defined recursively in Θ: if Θ is the empty context, $\bullet$, then the only such substitution is the terminal map $!_\Delta : \mathsf{Sub}\ \Delta\ \bullet$. If Θ is an extended context $\Gamma \triangleright A$, then such a substitution must be a pair $\langle \sigma, s \rangle$, as (Local-Rep) demands. As we'll see, a variant of this isomorphism will be the law connecting the deep and shallow polarities of the category model together.

```
record CwF : Set where
  field
  -- Category of contexts
  Con : Set
  Sub : Con → Con → Set

  -- The empty context (terminal object)
  • : Con
  ! : (Γ : Con) → Sub Γ •

  -- Presheaf of types
  Ty : Con → Set
  __[__] : {Γ Δ : Con}
    → Ty Γ → Sub Δ Γ → Ty Δ

  -- Presheaf of terms
  Tm : (Γ : Con) → Ty Γ → Set
  __[__] : {Γ Δ : Con}{A : Ty Γ}
    → Tm(Γ,A) → (σ : Sub Δ Γ) → Tm(Δ, A[σ])

  -- Context extension
  __▹__ : (Γ : Con) → Ty Γ → Con
  ⟨__,__⟩ : {Γ Δ : Con}{A : Ty Γ}
    → (σ : Sub Δ Γ) → Tm(Δ, A[σ]) → Sub Δ (Γ▹A)
  p : {Γ : Con}{A : Ty Γ} → Sub (Γ▹A) Γ
  v : {Γ : Con}{A : Ty Γ} → Tm(Γ▹A, A[p])
```

Figure 1: The main data of a category with families

It turns out that the groupoid model only relies on its mathematical structures being *groupoids* when giving semantics for identity types: for defining the basic CwF structure, it's never necessary to invert morphisms. Therefore, obtaining the category model as a CwF is very easy: take the definition of the groupoid model, and replace every instance of 'groupoid' with 'category'. This is done for Con, Sub, and the object parts of Ty and Tm in Fig. 2: contexts are categories, substitutions are functors, and types are families of categories indexed by the context. We could just as well define context extension at this point as well—its definition *also* doesn't need to invert any morphisms in Γ, or in any of the categories $A(\gamma)$,[3] to define the category $\Gamma \triangleright A$. However, it will be more elegant to do so in just a moment, when we introduce the negative polarity.

```
Con = Cat

Sub Γ Δ = Cat [ Γ, Δ ]

• : Cat
• = 𝟙  -- the singleton category, with one object, *,
       and only the identity morphism

--  A : Ty Γ means A : Γ → Cat
record Ty (Γ : Con) : Set where
 field
  obj : |Γ| → Cat
  map : {γ₀ γ₁ : |Γ|} → Γ [ γ₀, γ₁ ] → Cat [ obj γ₀, obj
     γ₁ ]
  fid : {γ : |Γ|} → map (id_γ) ≡ id_obj(γ)
  fcomp : {γ₀ γ₁ γ₂ : |Γ|}{γ₀₁ : Γ [ γ₀, γ₁ ]}{γ₁₂ : Γ [ γ₁, γ
    ₂ ]}
        → map (γ₁₂ ∘ γ₀₁) ≡ (map γ₁₂) ∘ (map γ₀₁)

record Tm (Γ : Con) (A : Ty Γ) : Set where
 field
  obj : (γ : |Γ|) → |A.obj(γ)|
  map : {γ₀ γ₁ : |Γ|}
        → (γ₀₁ : Γ [ γ₀, γ₁ ])
        → (A.obj γ₁) [ (A.map(γ₀₁)).obj (obj γ₀),
     obj(γ₁) ]
  fid : {γ : |Γ|} → map (id_γ) ≡ id_obj(γ)
  fcomp : {γ₀ γ₁ γ₂ : |Γ|}{γ₀₁ : Γ [ γ₀, γ₁ ]}{γ₁₂ : Γ [ γ₁, γ
    ₂ ]}
        → map (γ₁₂ ∘ γ₀₁) ≡ (map γ₁₂) ∘ (A.map γ₁₂
    ).map (map γ₀₁)
```

Figure 2: The CwF structure of the category model (besides context extension)

So what additional structure can we interpret in the category model? Well, as discussed in Section 1, we will be

[3] As with functors, we use the more pedantic notation of A.obj and A.map in the pseudo-AGDA snippets, but in the main text we'll write both as just A (e.g. writing $A(\gamma)$ and $A(\gamma_{01})$), since it'll always be clear from context whether we mean the object or morphism part of A; likewise for terms.

```
(__)ᵒᵖ : Cat → Cat
  | Γᵒᵖ | = | Γ |
  Γᵒᵖ [ γ₀ , γ₁ ] = Γ [ γ₁ , γ₀ ]

(__)ᵒᵖ : {Δ Γ : Cat} → Cat [ Δ , Γ ] → Cat [ Δᵒᵖ , Γᵒᵖ ]
  Fᵒᵖ.obj δ = F.obj δ
  Fᵒᵖ.map δ₀₁ = F.map δ₀₁
```

Figure 3: The 'opposite' operation on categories

particularly interested in the 'opposite category' operation, which is precisely defined in Fig. 3. Note that $(\_\_)^{op}$ is not just an operation on categories, but that it has an action on functors as well, i.e. it is an endofunctor on Cat. Now, recall that there's two key ways we use Cat in the category model (and Grpd in the groupoid model): Cat is the category of contexts, and a type A in context Γ is a functor from Γ into Cat.[4] Therefore, the opposite category construction will manifest in the theory as an operation on contexts (and substitutions), as well as an operation on types. The former is what we referred to as 'deep' polarity in Section 1, while the latter is 'shallow' polarity.

Proposition 2.1. *The category model validates the following rules.*

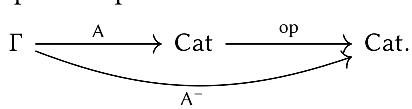
$$\frac{\Gamma : \mathit{Con}}{\Gamma^- : \mathit{Con}} \quad \text{(Con-Neg)}$$

$$\frac{\sigma : \mathit{Sub}\ \Delta\ \Gamma}{\sigma^- : \mathit{Sub}\ \Delta^-\ \Gamma^-} \quad \text{(Sub-Neg)}$$

$$\frac{A : \mathit{Ty}\ \Gamma}{A^- : \mathit{Ty}\ \Gamma} \quad \text{(Ty-Neg)}$$

Proof. Fig. 4 □

The interpretation of (Con-Neg) is just $(\_\_)^{op}$ itself, whereas (Ty-Neg) is post-composition with it:



Now, let's observe a few things about these negation operations. First, notice that negation has no effect on the empty context: $\bullet^- = \bullet$. In general, any groupoid will be isomorphic to its opposite—hence why this operation is not studied on the groupoid model. Next, observe that negation is definitionally self-inverse: $(\Gamma^-)^- = \Gamma$, and $(A^-)^- = A$, and $(\sigma^-)^- = \sigma$. Finally, we note that type negation distributes over substitution: for any σ: Sub Δ Γ and A : Ty Γ,

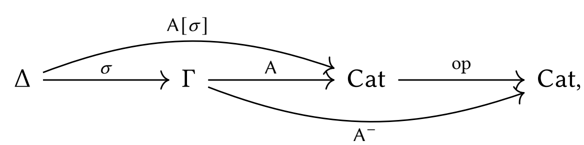
$$(A[\sigma])^- = A^-[\sigma].$$

[4]This is a more succinct statement of the definition of Ty(Γ) given in Fig. 2.

```
-- Context negation
(__)⁻ : Con → Con
  Γ⁻ = Γᵒᵖ
(__)⁻ : {Δ Γ : Con} → Sub Δ Γ → Sub Δ⁻ Γ⁻
  F⁻ = Fᵒᵖ

-- Type negation
(__)⁻ : {Γ : Con} → Ty Γ → Ty Γ
  A⁻.obj γ   = (A.obj γ)ᵒᵖ
  A⁻.map γ₀₁ = (A.map γ₀₁)ᵒᵖ

-- Positive extension
__▹⁺__ : (Γ : Con) → Ty Γ → Cat
  |Γ ▹⁺ A| = Σ (γ : |Γ|). |A.obj γ|
  (Γ ▹⁺ A) [ (γ₀, a₀) , (γ₁, a₁) ] =
     Σ (γ₀₁ : Γ [ γ₀, γ₁ ]). (A.obj γ₁) [ (A.map γ₀₁).obj
       a₀, a₁ ]

-- Negative extension
__▹⁻__ : (Γ : Con) → Ty Γ⁻ → Cat
  |Γ ▹⁻ A| = Σ (γ : |Γ|). |A.obj γ|
  (Γ ▹⁻ A) [ (γ₀, a₀) , (γ₁, a₁) ] =
     Σ (γ₀₁ : Γ [ γ₀, γ₁ ]). (A.obj γ₀) [ a₀, (A.map γ₀₁
       ).obj a₁ ]
```

Figure 4: Polarized structure of the category model

This is because substitution is defined by pre-composition,



and composition is associative.

Above, we did not yet mention the definition of context extension in the category model. This is because there are *two* context extension operations in the category model: one positive, one negative.

Proposition 2.2. *The category model validates the following rules.*

$$\frac{\Gamma : \mathit{Con} \quad A : \mathit{Ty}\ \Gamma}{\Gamma \triangleright^+ A : \mathit{Con}} \quad \text{(Extend}^+\text{)}$$

$$\frac{\Gamma : \mathit{Con} \quad A : \mathit{Ty}\ \Gamma^-}{\Gamma \triangleright^- A : \mathit{Con}} \quad \text{(Extend}^-\text{)}$$

such that there are bijections, natural in Δ *and in* (Γ, A)*:*

$$Sub\ \Delta\ (\Gamma \triangleright^{+} A) \cong \sum_{\sigma :\ Sub\ \Delta\ \Gamma} Tm(\Delta, A[\sigma]) \qquad \text{(Local-Rep}^{+}\text{)}$$

$$Sub\ \Delta\ (\Gamma \triangleright^{-} A) \cong \sum_{\sigma :\ Sub\ \Delta\ \Gamma} Tm(\Delta^{-}, A[\sigma^{-}]^{-}). \qquad \text{(Local-Rep}^{-}\text{)}$$

The semantics for these extension operators are given in Fig. 4, and the required bijections are easy to verify. Hofmann and Streicher define context extension in the groupoid model using the same definition as (Local-Rep^{+}), but, since groupoids are self-dual, (Local-Rep^{-}) would be equivalent. This construction is well-known: as Hofmann and Streicher note, the definition of $\Gamma \triangleright^{+}$ A is "the total category of the co-fibration obtained by applying the Grothendieck construction to A" [16, Section 4.5]. The definition of $\Gamma \triangleright^{-}$ A likewise utilizes the Grothendieck construction for contravariant functors—note in the premises of (Local-Rep^{-}) that A : Ty Γ^{-}, i.e. A : $\Gamma^{\mathrm{op}} \to$ Cat.

In the present work, we will not explore in detail the theory of deep polarity: in the next section, we will need to restrict our attention to *neutral contexts*, i.e. groupoid contexts. This is not the same as reverting to the groupoid model—as we'll see, keeping the shallow polarity of A versus A^{-} makes it possible to define asymmetric Hom-types, instead of the symmetric identity types of the groupoid model—but, as mentioned, the deep polarity is trivial when applied to groupoid contexts. Our suspicion is that some system of *zoned contexts* or *modal type theory* which provides syntax for working with *both* polarized and neutral contexts will be the ultimate theory of the category model. But that will have to be left for future work.

3 Hom Types and Directed Path Induction

We now turn our attention to the directed analogue to identity types: *Hom-types*. Our hypothesis is that the category model will accommodate an interpretation of Hom-types, analogous to the groupoid model's interpretation of identity types. Recall that our goal is to do synthetic category theory—terms of these Hom-types ought to behave like the morphisms of a *category* (refl_t as the identity morphism on the term/object t, and a composition operation definable by directed path induction), but not necessarily a *groupoid* (symmetry ought not to hold in general: having a term of type Hom(t,t') shouldn't automatically give you one of type Hom(t',t)). This hypothesis will prove correct, but it will take careful work to actually perform the desired constructions.

In the groupoid model, identity types between terms of type A are interpreted using the hom-sets of the groupoids interpreting A. That is, A : Ty Γ means that A is a functor from the groupoid Γ to the category of groupoids, i.e. A(γ) is a groupoid for each object γ of Γ. To give semantics for the type Id(t,t'), we must also assign a groupoid to each γ in a functorial way. The groupoid that is used is the *discrete groupoid* on the set A(γ) [t(γ), t'(γ)], that is, the groupoid whose only morphisms are identity morphisms.[5] So far, this can be copied exactly to the category model: the object part of Hom(t,t') at γ is the discrete category on the hom-set A(γ) [t(γ), t'(γ)]. However, it is in the *morphism part* that we actually use the assumption that A(γ) is a groupoid: given γ_{01} : Γ [γ_0, γ_1], we have to give a way to turn A(γ_0)-morphisms (t γ_0) $\longrightarrow$ (t' γ_0) into A(γ_1)-morphisms (t γ_1) $\longrightarrow$ (t' γ_1). Given x : A(γ_0) [t(γ_0), t'(γ_0)], we have this situation:

$$\begin{array}{ccc} \mathsf{A}\ \gamma_{01}\ (\mathsf{t}\ \gamma_0) & \xrightarrow{\mathsf{A}\ \gamma_{01}\ \mathsf{x}} & \mathsf{A}\ \gamma_{01}\ (\mathsf{t'}\ \gamma_0) \\ {\scriptstyle \mathsf{t}(\gamma_{01})}\big\downarrow & & \big\downarrow{\scriptstyle \mathsf{t'}(\gamma_{01})} \\ \mathsf{t}\ \gamma_1 & & \mathsf{t'}\ \gamma_1. \end{array}$$

So, in the groupoid model, it's simple to fill in the bottom: just invert the arrow on the left, and compose the three arrows. In the category model, we can't invert arrows. But what we *can* do is have t be a term of type A^{-}, so the left arrow goes up and the construction carries through. This gives us the following rule for Hom-type formation, the same one given in [24, Section 2].

Proposition 3.1. *The category model validates the following rule.*

$$\frac{t\colon Tm(\Gamma, A^{-}) \qquad t'\colon Tm(\Gamma, A)}{Hom(t,t')\colon Ty\ \Gamma} \qquad \text{(Hom-Form)}$$

Proof. Fig. 5 □

```
Hom : Tm(Γ, A⁻) → Tm(Γ, A) → Ty Γ
 (Hom(t,t')).obj γ =
  (A.obj γ) [t.obj γ, t'.obj γ] -- Discrete category

 (Hom(t,t')).map γ01 =
  λ x → (t'.map γ01) ∘ ((A.map γ01).map x) ∘ (t.map γ
    01)
```

Figure 5: Semantics of Hom types in the Category Model

[5] To be fully precise, we'll express this by saying that the hom-set between objects X and Y is our meta-theoretic identity type $X \equiv Y$. So the hom-set between X and itself, $X \equiv X$, will be a singleton (since our meta-theoretic identity type satisfies UIP), whose single element must be the identity morphism on X. For distinct X and Y, the identity type $X \equiv Y$ will be uninhabited. Thus, "the only morphisms are identity morphisms".

This also matches with the goal of synthetic category theory: in category theory, Hom-sets are *contravariant* in their first argument and *covariant* in their second. So asking for t to be a term of A^- and for t' to be a term of A is the correct assignment of variances.

At this stage, we can remark that the category model refutes the directed analogue of UIP. We could use the same counterexample as Hofmann-Streicher [16, Theorem 5.1] taking the group $\mathbb{Z}_2$ as a single-object groupoid. Any given groupoid is a closed type in the empty context of the groupoid model (since functors $\mathbb{1} \to$ Grpd are the same thing as groupoids), and any closed term of such a type is just an object of that groupoid. So therefore the closed type $\mathbb{Z}_2$ has a single term, call it s, but exactly two morphisms in $\mathbb{Z}_2$ [s, s] (corresponding to 0 and 1). Thus the identity type Id(s,s) has exactly two terms, violating UIP. But, since we're in the category model, let's pick a category that's *not* a groupoid. The simplest example is the category with two objects, p and s, and two distinct parallel morphisms from p to s:

$$p \rightrightarrows s$$

which, by the same logic, dictates that Hom(p,s) will not have a unique element. So we've refuted directed UIP. However, this example also serves another purpose for us: showing that symmetry cannot be proved in the category model. Below, we will give rules for directed path induction, our theoretical tool for proving statements about Hom-types. We can rest assured that these rules do not allow us to invert a directed path in Hom(t,t') to get a directed path in the other direction, Hom(t',t). Because, if they did, then we could apply that to either morphism from p to s to get a morphism from s to p, of which there are none. So our Hom-types are genuinely asymmetric in general; we didn't accidentally reinvent symmetric identity types.

To define directed path induction, we'll first need to construct refl. This will prove more challenging. Naïvely, we want refl_t to be a term of type Hom(t,t), reflecting the fact that identity morphisms in a category have the same domain and codomain. But it is seemingly impossible to form the type Hom(t,t), because (Hom-Form) would say that t must be a term of type A *and* a term of type A^-. Can this be? None of the existing rules tell us how a term can be of type A and A^-, nor do they furnish us a way to convert terms of type A into terms of type A^- or vice-versa.

There are two solutions to this impasse: neutralizing the shallow polarity, or neutralizing the deep polarity. The former solution, which is the one adopted by North, utilizes a further extension of our theory, *core types.* Recall from above that the opposite category endofunctor Cat $\to$ Cat gives semantics for both context- and type-negation operations $(\_\_)^-$. We can do the same again, with a different endofunctor: the 'core groupoid' construction. The core operation sends each category Γ to its 'core', core(Γ), the subcategory with all the same objects, but only the Γ-*isomorphisms* as morphisms.[6] We could view this as a 'deep' operation on contexts, $(\_\_)^0$: Con $\to$ Con, but we'll again be more interested in the operation on types:

$$(\_\_)^0 : \{\Gamma : \mathsf{Con}\} \to \mathsf{Ty}\ \Gamma \to \mathsf{Ty}\ \Gamma$$
$$A^0.\mathsf{obj}\ \gamma = \mathsf{core}(A.\mathsf{obj}\ \gamma)$$
$$A^0.\mathsf{map}\ \gamma_{01} = \mathsf{core}(A.\mathsf{map}\ \gamma_{01})$$

This will fix the refl issue. Consider a term $t\colon \mathsf{Tm}(\Gamma, A^0)$. Then, for some $\gamma_1\colon |\Gamma|$, $A^0(\gamma_1)$ is a groupoid. Since it's a groupoid, i.e. we can take the inverse of $A^0(\gamma_1)$-morphisms such as $t(\gamma_{01})$, it's possible to make sense of Hom(t,t) the same way we made sense of Id(t,t) in the groupoid model. More verbosely, we could say there are 'coercion' operations

$$+ : \{\Gamma : \mathsf{Con}\}\{A : \mathsf{Ty}\ \Gamma\} \to \mathsf{Tm}(\Gamma,A^0) \to \mathsf{Tm}(\Gamma,A)$$
$$- : \{\Gamma : \mathsf{Con}\}\{A : \mathsf{Ty}\ \Gamma\} \to \mathsf{Tm}(\Gamma,A^0) \to \mathsf{Tm}(\Gamma,A^-)$$

where the object parts of t, +t, and −t all coincide, and the morphism parts of +t, and −t are the morphism part of t and its inverse, respectively. Then we could state the introduction rule as follows.

$$\frac{t\colon \mathsf{Tm}(\Gamma, A^0)}{\mathsf{refl}_t\colon \mathsf{Tm}(\Gamma, \mathsf{Hom}(-t,+t))}$$

While simple and effective at incorporating refl into the framework of polarized type theory, this fix suffers from a significant drawback. Our ultimate aim is to state a directed analogue of path induction, to allow us to prove a claim about arbitrary directed paths just by proving it about refl. But, if this is the introduction rule for refl, then it seems we can only state path induction based at terms of type A^0; we don't have any tools for reasoning about *arbitrary* directed paths.

In the present work, we'll take a different approach: neutralizing the *deep* polarity and maintaining the shallow polarity. To see what this means, consider the case where Γ is the empty context. In the empty context, there's no trouble typing refl. Since the only morphism of the empty context is the identity morphism, we don't have to worry about the morphism parts of closed types and terms. A closed type A is interpreted as a category; terms of A and terms of A^- are the same thing—objects of A; and Hom(t,t) makes perfect sense—it's the set A [t, t], as a discrete category. In the empty context, path induction *will* allow us to prove things about arbitrary directed paths.

Here's the key point: we can do everything necessary for directed path induction, not just in the empty context,

[6]The morphism part of core as a functor Cat $\to$ Cat , takes a functor $F\colon$ Cat $[\Delta, \Gamma]$ and restricts its domain to the subcategory core(Δ). Since F must preserve isomorphisms by the functor laws, its image is in the core of Γ, hence we have $\mathsf{core}(F)\colon$ Cat $[\mathsf{core}(\Delta), \mathsf{core}(\Gamma)]$.

but in *any groupoid* context. Suppose Γ is a groupoid, and $\mathsf{A} : \mathsf{Ty}\ \Gamma$. Given a "positive term" of type A, that is, some $\mathsf{t} : \mathsf{Tm}(\Gamma, A)$, recall that the morphism part of t applied to some $\gamma_{01} : \Gamma\ [\gamma_0, \gamma_1]$ is of the form

$$\mathsf{t}\ \gamma_{01} :\ (\mathsf{A}\ \gamma_1)\ [\ \mathsf{A}\ \gamma_{01}\ (\mathsf{t}\ \gamma_0), \mathsf{t}(\gamma_1)\ ].$$

If we want to use t negatively, that is, in place of a term of type A^-—as we must do for Hom(t,t) to be well-formed—then we need a way to turn this into something of the form

$$(\mathsf{A}\ \gamma_1)\ [\ \mathsf{t}(\gamma_1), \mathsf{A}\ \gamma_{01}\ (\mathsf{t}\ \gamma_0)\ ]$$

which is the type of morphism parts of terms of A^-, applied to γ_{01}. In the groupoid model (and in the above-mentioned solution using core types), we did this by assuming $\mathsf{A}(\gamma_1)$ is a groupoid (or using the fact that $\mathsf{A}^0(\gamma_1)$ is a groupoid), and thereby taking the inverse of $\mathsf{t}(\gamma_{01})$ to get the desired $\mathsf{A}(\gamma_1)$-morphism. But instead we're assuming that Γ is a groupoid. This will still give us a way to coerce terms of A into terms of A^-: observe that we can instead invert γ_{01} and apply t.map:

$$\mathsf{t}({\gamma_{01}}^{-1}) :\ (\mathsf{A}\ \gamma_0)\ [\ \mathsf{A}\ {\gamma_{01}}^{-1}\ (\mathsf{t}\ \gamma_1), \mathsf{t}(\gamma_0)\ ].$$

Applying $\mathsf{A}(\gamma_{01})$, which is a functor from $\mathsf{A}(\gamma_0)$ to $\mathsf{A}(\gamma_1)$, takes us back into $\mathsf{A}(\gamma_1)$:

$$\begin{aligned}&\mathsf{A}\ \gamma_{01}\ (\mathsf{t}({\gamma_{01}}^{-1}))\\ &\quad : (\mathsf{A}\ \gamma_1)\ [\ \mathsf{A}\ \gamma_{01}\ (\mathsf{A}\ {\gamma_{01}}^{-1}\ (\mathsf{t}\ \gamma_1)), \mathsf{A}\ \gamma_{01}\ (\mathsf{t}(\gamma_0))\ ].\end{aligned}$$

Cleaning up using the functoriality of A, this turns out to have exactly the type we want. So we've successfully found a way to coerce terms of A into terms of type A^-. We can do the opposite, as well, and coerce terms of A^- to A.

Proposition 3.2. *The category model validates the following rules.*

$$\frac{\Gamma : \mathit{NeutCon} \quad t' : \mathit{Tm}(\Gamma, A)}{-t' : \mathit{Tm}(\Gamma, A^-)} \quad (\text{Coe}^-)$$

$$\frac{\Gamma : \mathit{NeutCon} \quad t : \mathit{Tm}(\Gamma, A^-)}{+t : \mathit{Tm}(\Gamma, A)} \quad (\text{Coe}^+)$$

Proof. Fig. 6 □

```
– : {Γ : NeutCon}{A : Ty Γ} → Tm(Γ,A) → Tm(Γ,A⁻)
  (–t').obj γ = t'.obj γ
  (–t').map γ01 = (A.map γ01).map (t'.map(γ01⁻¹))
+ : {Γ : NeutCon}{A : Ty Γ} → Tm(Γ,A⁻) → Tm(Γ,A)
  (+t).obj γ = t.obj γ
  (+t).map γ01 = (A.map γ01).map (t.map(γ01⁻¹))
```

Figure 6: Semantics of coercion in neutral contexts

This makes it possible to define refl: for a given term $\mathsf{t} : \mathsf{Tm}(\Gamma, \mathsf{A}^-)$, we can say $\mathsf{refl}_\mathsf{t} : \mathsf{Tm}(\Gamma, \mathsf{Hom}(\mathsf{t},+\mathsf{t}))$, and for $\mathsf{t}' : \mathsf{Tm}(\Gamma, \mathsf{A})$, we can say $\mathsf{refl}_{\mathsf{t}'} : \mathsf{Tm}(\Gamma, \mathsf{Hom}(-\mathsf{t}',\mathsf{t}'))$.

Proposition 3.3. *The category model validates the following rules.*

$$\frac{\Gamma : \mathit{NeutCon} \quad t : \mathit{Tm}(\Gamma, A^-)}{\mathit{refl}_t : \mathit{Tm}(\Gamma, \mathit{Hom}(t,+t))} \quad (\text{Hom-Intro}^-)$$

$$\frac{\Gamma : \mathit{NeutCon} \quad t' : \mathit{Tm}(\Gamma, A)}{\mathit{refl}_{t'} : \mathit{Tm}(\Gamma, \mathit{Hom}(-t',t'))} \quad (\text{Hom-Intro}^+)$$

Proof. Fig. 7 □

```
-- refl for terms of A⁻
refl : {Γ : NeutCon}{A : Ty Γ}
    → (t : Tm(Γ,A⁻)) → Tm(Γ,Hom(t,+t))
  refl_t.obj γ = id_t(γ)

  refl_t.map γ01 :
    (+t.map γ01) ∘ ((A.map γ01).map id) ∘ (t.map γ01) ≡
      id
  -- follows from functoriality of A, the defn of +t, and
      the groupoid axioms for Γ

-- refl for terms of A
refl : {Γ : NeutCon}{A : Ty Γ}
    → (t' : Tm(Γ,A)) → Tm(Γ,Hom(–t',t'))
  refl_t'.obj γ = id_t'(γ)

  refl_t'.map γ01 :
    (t'.map γ01) ∘ ((A.map γ01).map id) ∘ (–t'.map γ01)
      ≡ id
  -- follows from functoriality of A, the defn of –t',
      and the groupoid axioms for Γ
```

Figure 7: Semantics of refl in neutral contexts

Now on to directed path induction. As we've seen, directed paths in a type A are introduced by two rules, (Hom-Intro$^-$) and (Hom-Intro$^+$)—one for A and one for A^-. These are essentially the same, except they're oppositely polarized. Fittingly, directed paths are *eliminated* by one of two twin J-rules.

Proposition 3.4. *The category model validates the following rules.*

$$\frac{t\colon Tm(\Gamma, A^-) \quad M\colon Ty(\Gamma \rhd^+ (z{:}\,A) \rhd^+ Hom(t,z)) \quad m\colon Tm(\Gamma, M[+t, refl_t]) \qquad t'\colon Tm(\Gamma, A) \quad p\colon Tm(\Gamma, Hom(t,t'))}{J^+_M\ m\ t'\ p\colon Tm(\Gamma, M[t',p])}\ (\text{Hom-Elim}^+)$$

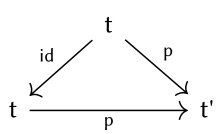
$$\frac{t'\colon Tm(\Gamma, A) \quad M\colon Ty((\Gamma \rhd^+ (z{:}\,A^-) \rhd^+ Hom(z,t'))^-) \quad m\colon Tm(\Gamma, M[-t', refl_{t'}]) \qquad t\colon Tm(\Gamma, A^-) \quad p\colon Tm(\Gamma, Hom(t,t'))}{J^-_M\ m\ t\ p\colon Tm(\Gamma, M[t,p])}\ (\text{Hom-Elim}^-)$$

Proof. Fig. 8 □

The precise statement of directed J given in Fig. 8 are somewhat impenetrable, but somewhat easier to understand in the special case of the empty context.[7] So then A is a category and t is an object of A. Then observe that M is a Cat-valued functor on the coslice category under t

$$t/A \to \mathsf{Cat}$$

and m is an object of the category $M(t, id_t)$. So then, for any arbitrary object t' and morphism $p\colon A\ [t, t']$, we need to get an object of $M(t', p)$. But recall that (t, id_t) is initial in the category t/A: the morphism p is a morphism from (t, id_t) to (t', p):



so $M(p)$ is a functor from $M(t, id_t)$ to $M(t', p)$, which we can apply to m. Conversely, for J^-, we instead have

$$M\colon (A/t')^{op} \to \mathsf{Cat}.$$

We need M to be contravariant here, since identity morphisms are *terminal* in slice categories: p is a A/t'-morphism from (t, p) to $(t', id_{t'})$, and so M p m again gives us what we want.

To see the utility of these rules, let's define composition of directed paths. Suppose we have terms t,t' of A^-, a term t'' of type A, and

$$p\colon Tm(\Gamma, Hom(t,+t')) \quad \text{and} \quad q\colon Tm(\Gamma, Hom(t',t'')).$$

Then (Hom-Elim$^+$) says: in order to define $q \circ p\colon Tm(\Gamma, Hom(t,t''))$, it is sufficient to define $refl_{t'} \circ p\colon Tm(\Gamma, Hom(t,+t'))$. Of course, we define this as p. Thus, we have a notion of composition for our directed paths. An interested reader can confirm that this definition gives us the same semantics as if we defined composition as a primitive, and interpreted it using the composition of the interpreting morphisms in the category. In future work, we will develop this into a more fully-featured synthetic category theory.

[7] This is also how Hofmann and Streicher explain the semantics of path induction in the groupoid model [16, Section 4.10], which is the same construction we're doing here.

4 Conclusion and Future Directions

In this paper, we have defined the category model, a directed analogue of the groupoid model, and shown that it has both a calculus of deep polarity, and, in neutral contexts, adequate facilities for directed path induction and synthetic 1-category theory.

Much remains to be explored about this model. As we have already noted, further work is required to allow for better interplay between neutral and polarized contexts. We suspect our type theory can be presented as an instance of multi-modal type theory [12], with one modality which corresponding to the core functor from categories to groupoids (the right adjoint to the forgetful functor).

The work presented in this paper is only a first step towards a better understanding of directed type theory, which has many exciting applications. One such application of directed type theory is to unify the two principles that reflect that *types are opaque*: univalence and parametricity [27]. The appropriate notion of *directed univalence* (which needs to be formally articulated in our model) says that the directed paths of the universe are functions, and entails normal, unidrected univalence by considering the core of the universe of sets. This principle—combined with the power of directed path induction—ought to entail various "free theorems" (in the sense of [29]), in particular that any polymorphic function is natural. We leave the detailed investigation of this idea to further work.

Another interesting line of thought is to revisit the theory of containers [1] in the directed setting where they correspond to categorified container [13]. We hope that this gives rise to a concise notion of QW types (W-types for QITs) and in a higher dimensional setting HW types (W-types for HITs).

Finally, we ultimately want to move to an untruncated setting, replacing categories with higher categories. We believe that should be possible along the lines of Higher Observational Type Theory (HOTT)[4, 6]. Indeed, we hope that this connection would contribute to our understanding of HOTT.

References

[1] Michael Abbott, Thorsten Altenkirch, and Neil Ghani. Containers: Constructing strictly positive types. *Theoretical Computer Science*, 342(1):3–27, 2005.

[2] Benedikt Ahrens, Krzysztof Kapulkin, and Michael Shulman. Univalent categories and the rezk completion. *Mathematical Structures in Computer Science*, 25(5):1010–1039, 2015.

J^+_M m t' p : Tm(Γ, M[t', p])
(J^+_M m t' p).obj γ : | M.obj(γ, t'.obj γ, p.obj γ) |
(J^+_M m t' p).obj γ = M.obj (id$_\gamma$, p.obj γ) (m.obj γ)

(J^+_M m t' p).map γ_{01} :
(M.obj (γ_1, t'.obj γ_1, p.obj γ_1)) [M.map (γ_{01}, t'.map γ_{01}) ((J^+_M m t' p).obj γ_0), (J^+_M m t' p).obj γ_1]
(J^+_M m t' p).map γ_{01} = (M.map (id$_\gamma$, p.obj γ)).map (m.map γ_{01})

J^-_M m t p : Tm(Γ, M[t, p])
(J^-_M m t p).obj γ : | M.obj(γ, t.obj γ, p.obj γ) |
(J^+_M m t p).obj γ = M.obj (id$_\gamma$, p.obj γ) (m.obj γ)

(J^-_M m t p).map γ_{01} :
(M.obj (γ_1, t.obj γ_1, p.obj γ_1)) [M.map (γ_{01}, t.map γ_{01}) ((J^-_M m t p).obj γ_0), (J^-_M m t p).obj γ_1]
(J^-_M m t p).map γ_{01} = (M.map (id$_\gamma$, p.obj γ)).map (m.map γ_{01})

Figure 8: Semantics of directed path induction in neutral contexts

[3] Benedikt Ahrens, Paige Randall North, and Niels van der Weide. Bicategorical type theory: semantics and syntax. *Mathematical Structures in Computer Science*, 33(10), oct 2023.

[4] Thorsten Altenkirch, Yorgo Chamoun, Ambrus Kaposi, and Michael Shulman. Internal parametricity, without an interval. *Proceedings of the ACM on Programming Languages*, 8(POPL):2340–2369, 2024.

[5] Thorsten Altenkirch and Ambrus Kaposi. Type theory in type theory using quotient inductive types. *ACM SIGPLAN Notices*, 51(1):18–29, 2016.

[6] Thorsten Altenkirch, Ambrus Kaposi, and Michael Shulman. Towards higher observational type theory, 2022. 28th International Conference on Types for Proofs and Programs (TYPES 2022).

[7] Thorsten Altenkirch and Jacob Neumann. Presheaf models of polarized higher-order abstract syntax, 2023. Second International Conference on Homotopy Type Theory (HoTT 2023).

[8] Steve Awodey. Natural models of homotopy type theory. *Mathematical Structures in Computer Science*, 28(2):241–286, 2018.

[9] John Cartmell. Generalised algebraic theories and contextual categories. *Annals of pure and applied logic*, 32:209–243, 1986.

[10] Pierre Clairambault and Peter Dybjer. The biequivalence of locally cartesian closed categories and martin-löf type theories. *Mathematical Structures in Computer Science*, 24(6):e240606, 2014.

[11] Peter Dybjer. Internal type theory. In *International Workshop on Types for Proofs and Programs*, pages 120–134. Springer, 1995.

[12] Daniel Gratzer, GA Kavvos, Andreas Nuyts, and Lars Birkedal. Multimodal dependent type theory. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 492–506, 2020.

[13] Håkon Robbestad Gylterud. Symmetric containers. Master's thesis, 2011.

[14] Martin Hofmann. On the interpretation of type theory in locally cartesian closed categories. In *International Workshop on Computer Science Logic*, pages 427–441. Springer, 1994.

[15] Martin Hofmann. Syntax and semantics of dependent types. In *Extensional Constructs in Intensional Type Theory*, pages 13–54. Springer, 1997.

[16] Martin Hofmann and Thomas Streicher. The groupoid interpretation of type theory. *Twenty-five years of constructive type theory (Venice, 1995)*, 36:83–111, 1995.

[17] Bart Jacobs. Comprehension categories and the semantics of type dependency. *Theoretical Computer Science*, 107(2):169–207, 1993.

[18] Ambrus Kaposi. "Message to the Agda mailing list". "https://lists.chalmers.se/pipermail/agda/2019/011176.html", 2019.

[19] Nikolai Kudasov, Emily Riehl, and Jonathan Weinberger. Formalizing the ∞-categorical yoneda lemma, 2023.

[20] Daniel Licata and Matthew Weaver. Theory and implmentation of bicubical directed type theory, 2023. Second International Conference on Homotopy Type Theory (HoTT 2023).

[21] Daniel R Licata and Robert Harper. 2-dimensional directed dependent type theory. 2011.

[22] 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. Elsevier, 1975.

[23] Per Martin-Löf. Constructive mathematics and computer programming. In L. Jonathan Cohen, Jerzy Łoś, Helmut Pfeiffer, and Klaus-Peter Podewski, editors, *Logic, Methodology and Philosophy of Science VI*, volume 104 of *Studies in Logic and the Foundations of Mathematics*, pages 153–175. Elsevier, 1982.

[24] Paige Randall North. Towards a directed homotopy type theory. *Electronic Notes in Theoretical Computer Science*, 347:223–239, 2019.

[25] Andreas Nuyts. Towards a directed homotopy type theory based on 4 kinds of variance. *Mém. de mast. Katholieke Universiteit Leuven*, 2015.

[26] Emily Riehl and Michael Shulman. A type theory for synthetic ∞-categories. *arXiv preprint arXiv:1705.07442*, 2017.

[27] Filipo Sestini and Thorsten Altenkirch. Naturality for free!—the category interpretation of directed type theory, 2019. The International Conference on Homotopy Type Theory (HoTT 2019).

[28] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. https://homotopytypetheory.org/book, Institute for Advanced Study, 2013.

[29] Philip Wadler. Theorems for free! In *Proceedings of the fourth international conference on Functional programming languages and computer*

architecture, pages 347–359, 1989.

[30] Matthew Z. Weaver and Daniel R. Licata. A constructive model of directed univalence in bicubical sets. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '20, page 915–928, New York, NY, USA, 2020. Association for Computing Machinery.

[31] Jonathan Weinberger and Ulrik Buchholtz. Type-theoretic modalities for synthetic $(\infty, 1)$-categories, 2019. International Conference on Homotopy Type Theory (HoTT 2019).