A Relationally Parametric Model of Dependent Type Theory

By: Robert Atkey, Neil Ghani, and Patricia Johann

Abstract

Reynolds’ theory of relational parametricity captures the invariance of polymorphically typed programs under change of data representation. Reynolds’ original work exploited the typing discipline of the polymorphically typed λ-calculus System F, but there is now considerable interest in extending relational parametricity to type systems that are richer and more expressive than that of System F. This paper constructs parametric models of predicative and impredicative dependent type theory. The significance of our models is twofold. Firstly, in the impredicative variant we are able to deduce the existence of initial algebras for all indexed functors. To our knowledge, ours is the first account of parametricity for dependent types that is able to lift the useful deduction of the existence of initial algebras in parametric models of System F to the dependently typed setting. Secondly, our models offer conceptual clarity by uniformly expressing relational parametricity for dependent types in terms of reflexive graphs, which allows us to unify the interpretations of types and kinds, instead of taking the relational interpretation of types as a primitive notion. Expressing our model in terms of reflexive graphs ensures that it has canonical choices for the interpretations of the standard type constructors of dependent type theory, except for the interpretation of the universe of small types, where we formulate a refined interpretation tailored for relational parametricity. Moreover, our reflexive graph model opens the door to generalisations of relational parametricity, for example to higher-dimensional relational parametricity.

Reynolds’ theory of relational parametricity captures the invariance of polymorphically typed programs under change of data representation [28]. Relational parametricity has been shown to yield a wide range of useful and surprising consequences, including free theorems [35], data type representations [13], optimisation of recursive programs [18], and geometric invariance properties [3]. Reynolds’ original work exploited the typing discipline of the polymorphically typed λ-calculus System F, but there is now considerable interest in extending relational parametricity to type systems that are richer and more expressive than that of System F.

Relational parametricity for systems with type-level computation — of which System $F_\omega$ is the prototypical example — has been studied by Vytiniotis and Weirich [33] and by Atkey [2]. The latter builds on earlier work by Hasegawa [14], who gives general requirements a model must satisfy in order to interpret System $F_\omega$. The extra expressive power of System $F_\omega$ allows for Church Encodings of indexed types. For example, the standard example of a Generalised Algebraic Datatype (GADT) [7] for typed expressions containing constants and pairing can be represented by the following $F_\omega$ type:

$$\lambda \tau. \forall \alpha: \tau \rightarrow \alpha \rightarrow (\forall \sigma. \alpha \rightarrow \sigma \rightarrow \alpha \rightarrow \alpha \rightarrow \alpha \rightarrow \alpha \rightarrow \alpha) \rightarrow \alpha \tau$$

Relational parametricity can be used to show that this type enjoys an initiality property [2], showing that it is isomorphic to the normal GADT of typed expressions.

By introducing a kind of natural numbers, it is also possible to encode in System $F_\omega$ interesting indexed types like length-indexed vectors. However, due to the rigid separation of types and terms in System $F_\omega$, type-level and term-level naturals are not the same thing, which leads to duplication and loss of expressivity. Dependent type theories weaken the distinction between types and terms, allowing terms to appear in types. It is therefore natural to ask whether or not relational parametricity extends to systems with type dependency, and whether or not the consequences of relational parametricity (suitably generalised) carry over as well.

In this paper, we answer both these questions in the affirmative. We build a model of dependent type theory (in predicative and impredicative variants) that naturally extends Reynolds’ original relationally parametric model. We also show that, within the impredicative model, every indexed functor $F$ has an initial algebra, given by the dependently typed Church encoding:

$$\lambda x: X. \Pi A: X \rightarrow U. ((\Pi x: X. T(Ax)) \rightarrow T(Ax)) \rightarrow T(Ax)$$

(The symbol $U$ stands for the universe of small types, and $T$ transforms a small type into a proper type; we define the syntax of dependent types in Section 3.) To our knowledge the model we present in this paper is the first parametric model, syntactic or semantic, of dependent type theory to establish this key theorem.

The question of the existence of relationally parametric models of dependent type theory has already been established as relevant for applications: Chlipala [8] assumes relational parametricity for representing higher-order abstract syntax, and Nanevski et al. [23] assumes properties of type abstraction for enforcing security properties. In this paper we demonstrate that relationally parametric models of dependent type theory do exist, though we leave questions about potential applications to future work.

**Approaches to Understanding Relational Parametricity** Since Reynolds’ original model-theoretic formulation, relational parametricity.
metricity has been studied in a number of settings. Reynolds’ original model was elucidated by Bainbridge et al. [4], who also moved to a PER-based interpretation of types to handle foundational set-theoretic problems, caused by impredicativity, with Reynolds’ original formulation. Robinson and Rosolini [30] studied Reynolds’ model from the point of view of internal categories, again to work around foundational problems with the semantics of impredicativity. More usefully for us, Robinson and Rosolini reformulated Reynolds’ original model in terms of reflexive graphs. Reflexive graphs capture the essence of relationally parametric models: types are simultaneously interpreted as sets and as relations between sets, with a distinguished reflexive relation, called the “equality relation”, from any set to itself, and open types are interpreted as morphisms between reflexive graphs. The model of dependent type theory that we construct in this paper is built on the category of reflexive graphs and reflexive graph morphisms that Robinson and Rosolini used to interpret the types of System F. Reflexive graph models of parametricity have also appeared implicitly in the work of Hasegawa [13, 14], and explicitly in the work of Dunphy and Reddy [11], both in settings without type dependency.

Other approaches to understanding parametricity include Pitts’ operational models of programming languages with polymorphic types and recursion [24]; Plotkin and Abadi’s logic for parametricity [26]; and Wadler’s translation of System F into second-order predicate logic [34].

The study of relational parametricity for systems with dependent types originates with Takeuti [31] and has recently been taken up by Bernardy et al. [5] and Krishnaswami and Dreyer [19]. Both Takeuti and Bernardy et al. study relational parametricity for dependent types via syntactic translations from one type theory to another. Krishnaswami and Dreyer construct a realizability model of the Calculus of Constructions, using Quasi-PERs to simultaneously define the underlying and relational interpretations of types. They demonstrate the adequacy of the Church encoding of the non-indexed initial algebra representing the natural numbers, but do not mention the Church encodings of indexed initial algebras that we consider in this paper.

The distinguishing feature of the parametric model of dependent types developed in this paper is that we follow Robinson and Rosolini in taking the interpretation of types as reflexive graphs as primitive, rather than explicitly seeking a relational interpretation of dependent types, as was done in the work of Takeuti, Bernardy et al. and Krishnaswami and Dreyer. In order to capture relational parametricity, our model interprets the universe U of small types in terms of sets and relations, so that we recover a relational interpretation of types within the setting of reflexive graphs (Section 4.3). By taking reflexive graphs as primitive, we can offer the following two significant contributions:

- At a results level, as we have mentioned above, we are able to show that the impredicative variant of our reflexive graph model contains initial algebras for all indexed functors (Section 5.4).
- The reflexive property of reflexive graphs plays a crucial role in the proof, being the appropriate generalisation of the key identity extension property of relationally parametric models of System F. Our models for both the predicative and impredicative variants also validate free theorems about polymorphic terms (Section 5.1), in the style of Wadler [35].

- At a conceptual level, reflexive graphs offer both canonical choices when building parametric models, and insight into the structure of relational parametricity. Robinson and Rosolini used reflexive graphs to bring clarity to relationally parametric models of System F, and here we use reflexive graphs to bring clarity to relationally parametric models of dependent type theory. Since reflexive graph models are presheaf models, we are provided with canonical choices of much of the structure of dependent types: for example, there is a canonical choice of interpretation of II-types, up to isomorphism. For the interpretation of the universe of small types there is no canonical choice. The obvious choice is to capture a semantic notion of smallness, using a construction due to Hofmann and Streicher. However, we demonstrate that smallness by itself is inadequate for relational parametricity, and we identify two further semantic properties, discreteness and proof-irrelevance, that must be captured by our interpretation of a universe of small types (Section 4.3).

### Structure of the Paper

- In Section 2 we trace a path from the original relationally parametric model of Reynolds, with its explicit use of relations, through to the reflexive graph formulation of parametricity, originally due to Robinson and Rosolini and, separately, Hasegawa. Since most presentations of relational parametricity for System F are explicitly formulated in terms of relations (e.g., Wadler’s “Theorems for Free”), we carefully demonstrate that, at the level of System F, the relational and reflexive graph formulations are equivalent. This gives us the necessary foundation on which to build a parametric model of dependent types.

- Section 3 recalls necessary syntactic and semantic background. We set out the precise variants of Martin-Löf Type Theory (predicative and impredicative) that we will model. We also recall the definition of Categories with Families (CwFs), a sound and complete class of categorical models of dependent type theory.

- We construct our parametric models of dependent types in Section 4 by showing that the category of reflexive graphs forms a CwF. The key feature of our models, apart from our use of reflexive graphs, is in our interpretation of the universe U of small types. We examine the necessary features that the interpretation of U must capture (namely smallness, discreteness, and proof-irrelevance) and build them into the model.

- With our parametric models in hand, we demonstrate several consequences of it in Section 5. We show that our model supports simple free theorems, as well as the construction of initial algebras for arbitrary indexed functors.

- Section 6 concludes and offers directions for future work.

### 2. Relational Parametricity and Reflexive Graphs

In this section we introduce the semantic structures used to construct our parametric model of dependent types. To this end, we first recall in Section 2.1 the “direct” method for building relationally parametric models of System F, as originally proposed by Reynolds [28] and elucidated by Bainbridge et al. [4]. We then show in Section 2.2 how to reformulate this model in terms of reflexive graphs, following Robinson and Rosolini [30].

Any model of System F, or any extension of it, must confront the issue of modelling impredicative type quantification. Reynolds showed that classical set-theoretic models of impredicative polymorphism cannot exist [27]. To overcome this problem, we build our models of impredicative type theories in an impredicative meta-theory. Specifically, we use the Calculus of Inductive Constructions with the predicative Type universe hierarchy, an impredicative universe Set, a (disjoint) impredicative universe Prop, and axioms for functional extensionality, and proof irrelevance and propositional extensionality in Prop. This theory is implemented in the Coq proof assistant [9] with the -iapredicative-set option, and the three axioms postulated. We work informally within this meta-theory, using set-theoretic notation. By taking this synthetic approach, we can dispense with the technicalities of constructing a model within
classical set theory using PERs or internal categories. A fully formalised relationally parametric model of System F has been constructed in this setting by Atkey [1].

2.1 Relational Models of Parametricity for System F

The well-kindred types of System F are shown in Figure 1. By these rules, well-kindred types are either type variables, function types, or universally quantified types. Kinding contexts \( \Gamma \) are lists of type variable/kind pairs \( \alpha : * \) that do not contain duplicate type variable names. In System F, there is only the base kind \( * \) of types.

Interpretation of Types

In the model presented by Bainbridge et al. [4], each well-kindred type \( \Gamma \vdash A : * \) is assigned two interpretations: an underlying interpretation \( \lbrack \Gamma ; A \rbrack_{\Theta} \), that interprets a type as a function on the universe \( U \) of small sets, and a relational interpretation \( \lbrack \Gamma ; A \rbrack_{\Theta, \gamma} \), that interprets a type as a transformer of relations between underlying interpretations. If, for any two sets \( X, Y \in U \), \( \text{Rel}(X, Y) \) denotes the set of binary relations between \( X \) and \( Y \), and if \( \lbrack \Gamma \rbrack \) stands for the number of members of \( \Gamma \), then the underlying interpretation and the relational interpretation of \( \Gamma \vdash A : * \) have the following meta-theoretic properties:

\[
\lbrack \Gamma \vdash [\,]_{\Theta} \rbrack \in U^{[\Gamma]} \rightarrow U
\]
\[
\lbrack \Gamma \vdash [\,]_{\Theta, \gamma} \rbrack \in V \gamma, \gamma_1, \gamma_2 \in U^{[\Gamma]}, \text{Rel}(\gamma_1, \gamma_2) \rightarrow \text{Rel}(\lbrack \Gamma \vdash \Theta \rbrack, \gamma_1, \lbrack \Gamma \vdash \Theta \rbrack, \gamma_2)
\]

The underlying interpretation and the relational interpretation of \( \Gamma \vdash A : * \) are explicitly linked by the appearance of the underlying interpretation in the type of the relational interpretation. The two interpretations are also linked by the identity extension property, which states that the relational interpretation of any type \( \Gamma \vdash A : * \) preserves equality:

\[
\forall \gamma \in U^{[\Gamma]}, \lbrack \Gamma \vdash A : \Theta \rbrack, \gamma \in \text{Eq}_\gamma = \text{Eq}_{\lbrack \Gamma \vdash A : \Theta \rbrack, \gamma} = \text{Eq}_{\lbrack \Gamma \vdash A : \Theta \rbrack, \gamma}
\]

Here, \( \text{Eq}_\gamma \) is the equality relation on the set \( X \), and \( \text{Eq}_{\lbrack \Gamma \vdash A : \Theta \rbrack, \gamma} \) is the tuple \( \{ [E_{\gamma_1}], \ldots, E_{\gamma_n} \} \).

Identity extension is the key property that makes a model relationally parametric, rather than just logical relations over an existing model. We can read it as saying that every element of the interpretation of a closed type is related to itself by the relational interpretation of that type. Therefore, identity extension is similar to the abstraction theorem (see Interpretation of Terms at the end of this subsection), except that it applies to all elements of the denotation of a type, not just the denotations of terms. Consequently, we are able to use the relational interpretation of types to reason about arbitrary elements of the denotations of those types, not just elements that are the interpretations of terms.

As we shall see when we construct the interpretation of the type-theoretic universe of small types in Section 4.3 below, our model of dependent types will satisfy the (suitably generalised) identity extension property for this universe. Identity extension is essential in our proof of the existence of initial algebras in Section 5.4.

A consequence of requiring identity extension is that the underlying interpretation and the relational interpretation must be defined simultaneously. This is due to the interpretation of types of the form \( \Gamma \vdash \forall \alpha : * \vdash A : * \). Indeed, if we do not cut down the underlying interpretations of universal types to include only the “parametric” elements, then we will be unable to prove the identity extension property. This cutting down is shown in Bainbridge et al.’s definitions of the underlying interpretations and relational interpretations for universal types, here adapted to our setting:

\[
\lbrack \Gamma \vdash [\,]_{\Theta, \gamma} \rbrack \in \{ x : X \in U \mid \lbrack \Gamma, \alpha : \Phi \rbrack, \gamma_1 \rightarrow \text{Eq}_{\Theta, \gamma_2} \}
\]

\[
\forall \gamma_1, \gamma_2 \in U^{[\Gamma]}, \lbrack \Gamma \vdash \Theta \rbrack, \gamma_1, \lbrack \Gamma \vdash \Theta \rbrack, \gamma_2 \rightarrow \lbrack \Gamma \vdash A : \Theta \rbrack, \gamma_1 \rightarrow \text{Eq}_{\Theta, \gamma_2}
\]

Because the underlying interpretations and relational interpretations of types are mutually defined, we must simultaneously define \( \lbrack \Gamma \vdash \Theta \rbrack, \gamma_1 \rbrack, \gamma_2 \rbrack \) and prove the following property, which states that the relational interpretations of \( \Theta \) and \( A \) are preserved:

\[
\forall \gamma_1, \gamma_2 \in U^{[\Gamma]}, \lbrack \Gamma \vdash \Theta \rbrack, \gamma_1, \lbrack \Gamma \vdash \Theta \rbrack, \gamma_2 \rightarrow \lbrack \Gamma \vdash A : \Theta \rbrack, \gamma_1 \rightarrow \text{Eq}_{\Theta, \gamma_2}
\]

This preservation of relations property has been variously called Reynolds’ Abstraction Theorem [28], the Fundamental Theorem of Logical Relations [25], and the Parametricity Theorem [21].

2.2 Reflexive Graph Models of Parametricity for System F

The two-level approach of the Reynolds/Bainbridge et al. model, in which types are assigned an underlying interpretation and a relational interpretation, works well for the relatively simple setting of System F. However, it is difficult to see how to extend this approach to allow for more complex features such as higher kinds, which allow function kinds such as \( * \rightarrow * \) and type-level computation, as is found in dependent type theories. The primary difficulty lies in extending the definition of the identity extension property. When we only considered the base kind \( * \), interpreted as the collection of sets \( X \in U \), there was an obvious notion of “equality relation”. But it is not so obvious what “equality relation” should mean when interpreting higher kinds. One complication is that, no matter what our notion of “equality relation” ends up being, an equality relation for a higher-kinded type must actually be a relation transformer. This complication is further compounded when attempting to interpret a dependent type theory in which types may depend on terms as well as types. Fortunately, by using reflexive graphs, as introduced by
Robinson and Rosolini [30], we can construct a parametric model of System F that naturally extends to dependent types.

**Definition 1.** A reflexive graph is a 5-tuple \((\Gamma_o, \Gamma_r, \Gamma_{src}, \Gamma_{tgt}, \Gamma_{refl})\) comprising a pair of sets \(\Gamma_o\) and \(\Gamma_r\), together with three functions

\[
\begin{align*}
\Gamma_o & \to \Gamma_{src} & f_o, \\
\Gamma_r & \to \Gamma_{tgt} & f_r, \\
\Gamma_o & \to \Gamma_{refl} & \gamma, \\
\Gamma_r & \to \Gamma_{refl} & r, \\
\Gamma_o & \to \Gamma_r & \Gamma, \\
\Gamma_r & \to \Gamma_o & \Gamma
\end{align*}
\]

such that \(\Gamma_{src} \circ \Gamma_{refl} = \text{id}_{\Gamma_o} = \Gamma_{tgt} \circ \Gamma_{refl}\). A morphism \(f\) between reflexive graphs \(\Gamma\) and \(\Delta\) is a pair of functions \((f_o, f_r)\) that makes every square the following diagram commute:

\[
\begin{array}{ccc}
\Gamma_o & \xrightarrow{f_o} & \Delta_o \\
\Gamma_r & \xrightarrow{f_r} & \Delta_r \\
\end{array}
\]

The set \(\Gamma_o\) is called the set of objects, and \(\Gamma_r\) is called the set of relations, of the reflexive graph.

We use reflexive graphs for two purposes: to model collections of types, and to model specific types. The first use takes \(\Gamma_o\) to be a set of sets and \(\Gamma_r\) to be a set of relations over those sets. Each relation \(\gamma\) such that \(\Gamma_{src}(\gamma)\), and \(\Gamma_{tgt}(\gamma)\) and each set \(\gamma\) has a distinguished relation \(\Gamma_{refl}(\gamma)\) relating it to itself. This intuition underlies the following example.

**Example 1.** If \(U\) is a universe of small sets, then the kind \(\ast\) of types can be interpreted as a reflexive graph (also called \(\ast\)) as follows:

\[
\begin{align*}
\gamma_o & = U & \gamma_r & = \{ (X, Y, R) | X, Y \in U, R \in \text{Rol}(X,Y) \} \\
\gamma_{refl} & = (X, X, E_{eq_{X}}) & \gamma_{src} & = (X, Y, R) = X & \gamma_{tgt} & = (X, Y, R) = Y
\end{align*}
\]

The objects of this reflexive graph are thus small sets from \(U\), and its relations are ordinary binary relations between such sets. For each \(X \in U\), the function \(\gamma_{refl}\) takes the equality relation to be the distinguished binary relation on \(X\). For this choice of relation the identity extension property is a straightforward consequence of the definition of a reflexive graph morphism.

Our second use of reflexive graphs takes \(\Gamma_o\) to be a specific set and \(\Gamma_r\) to be a reflexive relation on that set, represented as a (multi-set of pairs of elements that are related. Then \(\Gamma_{refl}\) ensures that \(\Gamma_r\) is reflexive, while \(\Gamma_{src}\) and \(\Gamma_{tgt}\) map each pair of related elements to the component elements that are so related. This intuition underlies the following example, which forms the basis of the interpretation of the natural number type we present in Section 4.4.

**Example 2.** A reflexive graph can be built from the set of natural numbers as follows:

\[
\begin{align*}
\text{nat}_o & = \mathbb{N} & \text{nat}_{refl}(n) & = n \\
\text{nat}_r & = \mathbb{N} & \text{nat}_{src}(n) & = n
\end{align*}
\]

The objects of this reflexive graph are just the natural numbers. For any \(n \in \text{nat}_o\), we have \(\text{nat}_{src}(n) = \text{nat}_{tgt}(n) = n\), so two natural numbers are related if and only if they are equal. Generalising the construction of \(\text{nat}\), for any set \(X\) there is a reflexive graph \((X, X, \text{id}_X, \text{id}_X, \text{id}_X)\) that has \(X\) as its set of objects and relates objects in \(X\) if and only if they are equal. In particular, if \(1\) is a terminal object in the category of sets, then \((1, 1, \text{id}_1, \text{id}_1, \text{id}_1)\) is terminal in the category of reflexive graphs.

Reflexive graphs can be equivalently defined as covariant presheaves over the category \(RG = \bullet \xrightarrow{\text{src} \circ \text{refl} = \text{id} = \text{tgt} \circ \text{refl}} \ast\ast\) where \(\text{src} \circ \text{refl}\) is the identity function. Morphisms between reflexive graphs are exactly natural transformations between presheaves in \(\text{Set}^{RG}\).

Viewing reflexive graphs as presheaves ensures that the following well-known properties of categories of presheaves hold for categories of reflexive graphs as well. Firstly, categories of presheaves are always cartesian, so we can use the finite product structure to interpret kinding contexts. Secondly, categories of presheaves are always cartesian closed, which allows for the interpretation of higher kinds [2]. Finally, presheaf categories always form a (non-parametric) model of dependent type theory [15], a fact that will help us construct the parametric model we present in Section 4.

**Interpretation of Types** A well-kindred type \(\Gamma \vdash A : \ast\) is interpreted as a reflexive graph morphism from the \(|\Gamma|\)-fold product of the reflexive graph \(\ast\) from Example 1 to \(\ast\) itself. Just as we did in Example 1, we identify a context with its reflexive graph interpretation and use the same notation for both. Unfolding the definition shows that a reflexive graph morphism is nothing more than a pair comprised of an underlying interpretation and a relational interpretation satisfying the identity extension property, exactly as in the two-level semantics in Section 2.1. For any reflexive graph \(\Gamma\), we think of the homset \(\text{Set}^{RG}(\Gamma, \ast)\) as providing interpretations of types over \(\Gamma\).

**Interpretation of Terms** Just as in Section 2.1, the interpretation of types can be extended to an interpretation of typing contexts by tupling in the current setting. A well-kindred typing context \(\Theta\) is therefore also interpreted as an object of \(\text{Set}^{RG}(\Gamma, \ast)\). In the reflexive graph model, a well-typed term \(\Gamma; \Theta \vdash M : A\) is then interpreted, as before, as a morphism from the interpretation of the typing context \(\Theta\) to the interpretation of the type \(A\), i.e., as a morphism in the category \(\text{Set}^{RG}(\Gamma, \ast)\).

**Definition 2.** Let \(A, B \in \text{Set}^{RG}(\Gamma, \ast)\). A morphism from \(A\) to \(B\) is a function \(M : \gamma_r \in \Gamma_o, A(\gamma_r) \to B(\gamma_r)\) such that for all \(\gamma_r \in \Gamma_o\), if \(A(\gamma_r) = (A_1, A_2, A_3, A_4) \subseteq A_1 \times A_2\) and \(B(\gamma_r) = (B_1, B_2, B_3, B_2) \subseteq B_1 \times B_2\) then for all \((a_1, a_2) \in A\), \((M(\gamma_r)) a_1, M(\gamma_r)) a_2) \in B\). This definition makes each homset \(\text{Set}^{RG}(\Gamma, \ast)\) into a category.

Note that the fundamental theorem of logical relations stated at the very end of Section 2.1 is identical to the condition in Definition 2.1, up to the reformulation of types as reflexive graph morphisms.

3. Martin-Löf Type Theory with a Universe: Syntax and Categorical Semantics

The reflexive graph model of parametricity we present in Section 4 is defined with respect to the specific formulation of dependent type theory that we set out in this section. The type theory we consider is a standard variant of Martin-Löf type theory [22] with a natural number type \(\ast\), dependent product types \(\Pi:\ast.A,B\), and a Tarskistyle universe \(U\) closed under natural numbers and dependent products. Our default assumption is that the universe \(U\) is predicative, but we will also consider an impredicative universe in order to more closely relate our results with (impredicative) System F. Our type theory includes \(\beta\)- and \(\eta\)-equality rules for dependent product types, as well as \(\beta\)-equality rules for the natural numbers. Since our universe is Tarski-style, we have an explicit "decoder" type former \(\ast\) to take terms of type \(\ast\) to actual types.

The type theories we present in this section are close to the theoreies underlying the Agda [6] and Coq [9] systems. Our predicative theory is close to Agda, except that we only have one inductive type (the natural numbers), and we only have one universe, with an explicit universe decoder. The impredicative variant is close to the Calculus of Constructions (the theory underlying the impredicative variant of Coq), except with a natural number type, and an explicit universe decoder. Our use of an explicit universe decoder
is a technical device that means we do not have to deal with coherence issues arising from multiple typing derivations of the same term. Moving to an implicit universe decoder with our models does not present any problems that do not already arise when defining models of dependent types with a universe.

### 3.1 Syntax and Typing

The raw contexts, types, and terms are defined as follows:

\[
\begin{align*}
\Gamma &::= \emptyset \mid \Gamma, x : A \\
A, B, C &::= \Pi x.A.B \mid \text{Nat} \mid U \mid \top M \\
M, N, P &::= x \mid \lambda x.M \mid MN \mid \text{su } M \mid \\
\text{Nrec}(x.A, M^*, x.p.M^*, N) \mid \text{nat } \pi x : M.N
\end{align*}
\]

We will write \(A \to B\) for the type \(\Pi x.A.B\) when \(x\) does not appear free in \(B\), and similarly write \(M \to N\) for the term \(\pi x : M.N\) when \(x\) does not appear free in \(N\). Capture avoiding substitution of terms for variables in pre-syntax types \((A[N/x])\) and terms \((M[N/x])\) is defined in the usual way. In the term \(\text{Nrec}(x.A, M^*, x.p.M^*, N)\), the variable \(x\) is considered bound in \(A\), and the variables \(x\) and \(p\) are considered bound in \(M^*\).

The well-formed types and well-typed terms are defined using a collection of mutually inductively defined judgements. The five different judgements and their intuitive meanings are given in Figure 2. The first three judgements define subsets of the raw syntax that are to be considered “well-formed” under some assumptions. The remaining two judgements define when two types or terms are considered as equal for the purposes of the theory. The rules generating the judgements are given in Figure 3 and type rules for type equality judgements are given in Figure 4. The rules for term equality judgements, that is \(\beta\)- and \(\eta\)-equality for \(\Pi\)-types and \(\beta\)-equality for \(\text{Nat}\) are displayed in Figure 4. The type and term equality judgements are also understood to include the reflexivity, symmetry, transitivity and congruence rules. Note that, due to the type former \(\top\), the congruence rules make type equality depend on term equality. Further discussion about these rules, and close relatives of them, can be found in Hofmann’s survey of the syntax and semantics of dependent type theories [15].

### Impredicativity

The additional rules for considering an impredicative universe are given in Figure 5. These rules replace the rules \(U-\pi\) in Figure 3 and \(\beta-U-\pi\) in Figure 4 describing the \(U\)-former \(\pi x : M.N\). Impredicativity allows the construction of types in \(U\) (i.e., of small types) by quantification over some large type \(A\), which may, in fact, be \(U\) itself. We can therefore construct a universe \(U\) — for example, such that \(\pi a : U. a \to a : U\) — that is not permitted in the predicative theory. When \(U\) is impredicative, our type theory is equivalent to a variant of Coquand and Huet’s Calculus of Constructions [10].

### 3.2 Categories with Families

The syntax of Martin-Löf Type Theory is a complex system of mutually inductive definitions. The mutual dependencies between well-formed types, equality, and well-typed terms severely complicate directly defining and proving sound a semantics for type theory. We address this issue by formulating our reflexive graph model.
in terms of categories with families (CwFs), a notion originally due to Dybjer [12]. CwFs form a sound and complete class of categorical models of many variants of dependent type theory, as explained by Hofmann [15]. Therefore, by demonstrating that the category of reflexive graphs forms a CwF with the appropriate additional structure needed for the type formers we have chosen, we are guaranteed to have soundly modelled the syntax of the type theory presented in the previous section.

We present the definition of a CwF (Definition 3), and then interpret term substitution, such that the following hold:

\[ A\{id_T\} = A \]  
\[ A\{f\}\{g\} = A\{f \circ g\} \]  
\[ M\{id_T\} = M \]  
\[ M\{f\}\{g\} = M\{f \circ g\} \]

Equations 3 and 4 are well-typed by virtue of Equations 1 and 2, respectively.

5. A chosen terminal object 1 in \( C \), interpreting the empty context.
6. For each \( \Gamma \in \mathbb{Ob}(C) \) and \( A \in \mathbb{Ty}(\Gamma) \) an object \( \Gamma.A \in \mathbb{Ob}(C) \) (called the comprehension of \( A \)) such that there is a bijection

\( C(\Delta, \Gamma.A) \cong \{(f, M) \mid f : \Delta \rightarrow \Gamma, M \in \mathbb{Ty}(\Delta, A\{f\})\} \)

that is natural in \( \Delta \).

An alternative presentation of points (1) to (4) of Definition 3 is as a functor \( C^{op} \rightarrow \mathbb{Fam}(\text{SET}) \), where SET is the category of large sets (which includes the category Set as an object).

Given \( f : \Delta \rightarrow \Gamma \) and \( M \in \mathbb{Ty}(\Delta, A\{f\}) \), we write \( \langle f, M \rangle \) for the associated morphism \( f : \Delta \rightarrow \Gamma.A \) in \( C \) given by the isomorphism in point (6) of Definition 3. Conversely, given a morphism \( f : \Delta \rightarrow \Gamma.A \) in \( C \), we write \( f^* : \Delta \rightarrow \Gamma \) and \( f^{#2} \in \mathbb{Ty}(\Delta, A(f^{#1})) \) for the associated morphism and semantic terms. The following definitions derive projection and weakening structure from Definition 3, and also define the semantic counter-part of a simultaneous substitution of a term for a variable.

1. For any \( \Gamma \in \mathbb{Ob}(C) \) and \( A \in \mathbb{Ty}(\Gamma) \), the first projection morphism \( p^\Gamma_A : \Gamma.A \rightarrow \Gamma \) is defined as \( p^\Gamma_A = id_{\Gamma.A} \). This is used to interpret weakening by discarding a single variable.

2. For any \( \Gamma \in \mathbb{Ob}(C) \) and \( A \in \mathbb{Ty}(\Gamma) \), the second projection \( q^\Gamma_A \in \mathbb{Ty}(\Gamma.A, A(p^\Gamma_A)) \) is defined as \( q^\Gamma_A = id_{\Gamma.A} \). This is used to interpret the rule \( \text{VAR} \).

3. For any \( \Gamma, \Delta \in \mathbb{Ob}(C) \), \( A \in \mathbb{Ty}(\Gamma) \) and morphism \( f : \Delta \rightarrow \Gamma \), the weakening of \( f \) by \( A \), \( w_k.A.f : \Delta.A(f) \rightarrow \Gamma.A \) is defined as \( w_k.A.f = \langle f \circ p^\Delta_A(f), q^\Delta_A(f) \rangle \). This is used to lift simultaneous substitutions to work in larger contexts.

4. For any \( \Gamma \in \mathbb{Ob}(C) \), \( A \in \mathbb{Ty}(\Gamma) \) and \( M \in \mathbb{Ty}(\Gamma, A) \), we define the morphism \( \overline{M} : \Gamma \rightarrow \Gamma.A \) as \( \overline{M} = \langle id_{\Gamma}, M \rangle \). Mor-
phisms built from semantic terms are used to build simultaneous substitutions that substitute (semantic) terms for variables.

**Definition 4.** A CwF $C$ supports dependent products if for all semantic contexts $\Gamma \in \text{Ob}(\mathcal{C})$ and semantic types $A \in \text{Ty}(\Gamma)$ and $B \in \text{Ty}(\Gamma.A)$ there exist a semantic type $\Pi AB \in \text{Ty}(\Gamma)$, natural in $\Gamma$, such that there is a bijection $A \longmapsto \text{Ty}(\Gamma,A,B) \cong \text{Ty}(\Gamma,\Pi AB)$, natural in $\Gamma$.

Definition 4 gives a semantic counterpart of the abstraction rule $\lambda AM$. We define the following semantic counterpart of the application rule $\text{App}$. For $\Gamma$, $A$ and $B$ as in Definition 4, and $M \in \text{Ty}(\Gamma,\Pi AB)$ and $N \in \text{Ty}(\Gamma, A)$, we define $\text{App}_{\Gamma,A,B}(M,N) \in \text{Ty}(\Gamma, B(\mathcal{N}))$ by $\text{App}_{\Gamma,A,B}^{\Gamma} = (\lambda N^- (M))(\mathcal{N})$.

**The Natural Number Type** The structure required to model the type of natural numbers directly follows the syntactic structure we presented in Section 3, modulo the more explicit presentation of substitution inherent in the CwF formalism. The following definition gives semantic counterparts to the rules $\epsilon$, $\nu$, and $\text{Nrec}$.

**Definition 5.** A CwF $C$ supports natural numbers if, for all $\Gamma \in \text{Ob}(\mathcal{C})$, there is a semantic type $\text{Nat} \in \text{Ty}(\Gamma)$ (we usually omit the superscript $\Gamma$), semantic terms $\text{zero} \in \text{Ty}(\Gamma, \text{Nat})$, and $\text{succ} \in \text{Ty}(\Gamma, \text{Nat}, \text{Nat})$, and, for each $A \in \text{Ty}(\Gamma, \text{Nat})$, a function on semantic terms $\text{Nrec}_A$ of type $\text{Ty}(\Gamma, A(\text{zero})) \times \text{Ty}(\Gamma, \text{Nat}, A(\text{succ} \circ \text{Nrec}_A \circ A)) \rightarrow \text{Ty}(\Gamma, \text{Nat}, A)$ such that $\text{Nat}$, $\text{zero}$, $\text{succ}$ and $\text{Nrec}$ are natural in $\Gamma$, and the following two equations hold:

$$\text{Nrec}_A(M^\gamma, M^\gamma)(\text{zero}^\gamma) = M^\gamma$$

$$\text{Nrec}_A(M^\gamma, M^\gamma)(\text{succ} \circ \text{Nrec}_A \circ A)^\gamma = M^\gamma \{\text{Nrec}_A(M^\gamma, M^\gamma)\} \quad (5)$$

**Universe** The CwF interpretation of the universe $U$ is just the translation of the syntactic structure into the CwF framework. The following definitions are semantic counterparts to $U$-$\text{Nat}$, $U$-$\pi$, and $U$-$\pi$.

**Definition 6.** A CwF $C$ supports a predicative universe closed under natural numbers and dependent product if (a) for all $\Gamma \in \text{Ob}(\mathcal{C})$ there exist semantic types $U \in \text{Ty}(\Gamma)$ and $T \in \text{Ty}(\Gamma,U)$, both natural in $\Gamma$; (b) there exists a semantic term $\text{nat} \in \text{Ty}(\Gamma, U)$, natural in $\Gamma$ such that $T(\text{nat}) = \text{Nat}$; and (c) for all $M \in \text{Ty}(\Gamma, U)$ and $N \in \text{Ty}(\Gamma.U.\text{Ty}(\mathcal{M}), U)$ there exists a semantic term $\pi MN \in \text{Ty}(\Gamma, U)$, natural in $\Gamma$; such that $T(\pi MN) = \Pi(T(\mathcal{M}))(T(\mathcal{N}))$.

**Definition 7.** A CwF $C$ supports an impredicative universe closed under natural numbers and dependent product if there are $U$, $T$, and $\pi$ as in Definition 6, and for all $A \in \text{Ty}(\Gamma)$ and $M \in \text{Ty}(\Gamma.A, U)$ there exists a semantic term $\pi AM \in \text{Ty}(\Gamma, U)$, natural in $\Gamma$, such that: $T(\pi AM) = \Pi A(T(\mathcal{M}))$.

4. **The Reflexive Graph Model of Type Theory**

Constructing our reflexive graph model of type theory is now a matter of showing that the category of reflexive graphs from Section 2 has the structure of a category with families. The essential tasks are to (a) establish what a family of reflexive graphs over a reflexive graph is, in order to model types (Sections 4.1 and 4.2); and (b) to determine the interpretation of the universe type $U$ (Section 4.3).

Once we have done the former, the interpretations of $\Pi$-types and the natural number type are determined up to isomorphism (Section 4.4). As an intuitive guide for determining the correct definition of a family of reflexive graphs, we use standard results about fibrational models of dependent types. Knowledge of fibrations is not required to understand our final definitions.

4.1 **Families of Reflexive Graphs**

A standard way to model the dependency of types on terms is to use the families fibration $p : \text{Fam}(\text{Set}) \rightarrow \text{Set}$, where the category $\text{Fam}(\text{Set})$ has as objects pairs $(X, P)$, where $X \in \text{Set}$ and $P : X \rightarrow \text{Set}$, and the functor $p$ projects out the first element. The basic idea of the families fibration is that a pair $(X, P)$ represents a context $X$ and a collection $P$ of types indexed by the elements of $X$. However, in a reflexive graph model of parametricity we want to index not by sets, but by reflexive graphs in $\text{Set}^{\text{RG}}$. Fortunately, there is a natural way to do this. By standard results about fibrations (see Exercise 1.8.8 in Jacobs [17]), the functor $p^{\text{RG}} : \text{Fam}(\text{Set})^{\text{RG}} \rightarrow \text{Set}^{\text{RG}}$, defined as $p^{\text{RG}}(F) = p \circ F$, is also a fibration. This hints that objects of $\text{Fam}(\text{Set})^{\text{RG}}$ are the natural interpretations of dependent types in a reflexive graph model of parametricity, and that the functor $p^{\text{RG}}$ should send each such object to the reflexive graph interpreting the context over which the type that object interprets is indexed.

Unpacking the definition of $\text{Fam}(\text{Set})^{\text{RG}}$, we see that an object of $\text{Fam}(\text{Set})^{\text{RG}}$ is equivalent to a pair $(\Gamma, A)$ of a reflexive graph $\Gamma$, together with a family $A$ of reflexive graphs that is over $\Gamma$, in the following sense:

**Definition 8.** Let $\Gamma$ be a reflexive graph. A family of reflexive graphs over $\Gamma$ is a $\tau$-tuple $A = (A_0, A_R, A_{\text{nrec}}, A_{\text{rec}})$, where:

$$A_0 : \Gamma_0 \rightarrow \text{Set}$$

$$A_R : \Gamma_R \rightarrow \text{Set}$$

$$A_{\text{nrec}} : \forall \gamma_0 \in \Gamma_0. A_{\text{nrec}}(\gamma_0) \rightarrow A_R(\Gamma_{\text{nrec}}(\gamma_0))$$

$$A_{\text{rec}} : \forall \gamma_0 \in \Gamma_0. A_R(\gamma_0) \rightarrow A_0(\Gamma_{\text{rec}}(\gamma_0))$$

such that the following equations hold:

$$\forall \gamma_0 \in \Gamma_0. A_{\text{nrec}}(\Gamma_{\text{nrec}}(\gamma_0)) \circ A_{\text{nrec}}(\gamma_0) = \text{id} \quad (7)$$

$$\forall \gamma_0 \in \Gamma_0. A_{\text{rec}}(\Gamma_{\text{rec}}(\gamma_0)) \circ A_{\text{rec}}(\gamma_0) = \text{id} \quad (8)$$

Morphisms between pairs of families of reflexive graphs over the same reflexive graph are derived from the morphisms of $\text{Fam}(\text{Set})^{\text{RG}}$. This yields the following definition:

**Definition 9.** Let $\Gamma$ be a reflexive graph and let $A$ and $B$ be families of reflexive graphs over $\Gamma$. A morphism $M$ from $A$ to $B$ consists of a pair of functions

$$M_0 : \forall \gamma_0 \in \Gamma_0. A_0(\gamma_0) \rightarrow B_0(\gamma_0)$$

$$M_r : \forall \gamma_r \in \Gamma_R. A_R(\gamma_r) \rightarrow B_R(\gamma_r)$$

such that

$$\forall \gamma_0 \in \Gamma_0. a_0 \in A_0(\gamma_0), B_0(\gamma_0)(M_0 \gamma_0 a_0) = M_r (\Gamma_{\text{rec}}(\gamma_0))(A_{\text{rec}}(\gamma_0)(a_0))$$

$$\forall \gamma_r \in \Gamma_R. a_r \in A_R(\gamma_r), B_R(\gamma_r)(M_r \gamma_r a_r) = M_0 (\Gamma_{\text{nrec}}(\gamma_r))(A_{\text{nrec}}(\gamma_r)(a_r))$$

Families of reflexive graphs over a reflexive graph $\Gamma$ and morphisms between them form a category $\text{RG-Fam}(\Gamma)$.

4.2 **Reflexive Graphs as a Category with Families**

We define a CwF structure on the category of reflexive graphs as follows. For each reflexive graph $\Gamma$, we define the collection of...
The category of reflexive graphs has the structure requirements in Definition 3.

A collection of families of reflexive graphs over $\Gamma$ consists of small reflexive graphs and the relations are spans between small reflexive graphs. As Hofmann and Streicher show, this definition exactly captures the small families of reflexive graphs: families of reflexive graphs where the sets of objects and sets of relations are always taken from the universe $U$.

Put mathematically, if we write $RG-Fam(\Gamma)$ for the subcategory of small families of reflexive graphs over some reflexive graph $\Gamma$, then we have $\text{Tm}(\Gamma; U^{HS}) \cong \text{Ob}(RG-Fam(\Gamma))$.

However, despite precisely capturing smallness, Hofmann and Streicher’s universe is too big for our purposes. We want to lift the consequences of relational parametricity from System F to the setting of dependent types. To do this, we need to accurately replicate the salient features of the kind $\star$ of System F types in our interpretation of the type-theoretic universe $U$.

Recall the definition of the reflexive graph $\ast$ representing the System F kind of types from Example 1. This reflexive graph has the following three properties, only the first of which is captured by the Hofmann-Streicher universe. Firstly, the sets of objects and relations are small, in the sense of belonging to the universe $U$ of small sets. Secondly, the reflexive structure of $\ast$ indicates that the equality relation is distinguished amongst all possible binary relations on sets; this is the essential identity extension property. As we saw in Example 2, reflexive graphs in which only equal objects are related can be modelled by taking the sets of objects and relations to be equal. Thirdly, the relations between sets are proof-irrelevant: since a relation $R$ from $X$ to $Y$ is a subset of $X \times Y$, and since each pair $(x, y)$ is either in the subset or not, there is at most one way that $x$ and $y$ can be related. Proof-irrelevance is essential for reasoning using a relationally parametric model: we use the relational level to reason about equality at the object level, but we have nothing to reason about the relational level with, so we have nothing to reason about the relational level with, so proof-irrelevance is needed to make sure that equality at this level is trivial. This insight is formalised in Lemma 2 below, and heavily used in Section 5. We note in passing that thinking along these lines motivates the formulation of infinite reflexive graphs (a.k.a., reflexive globular sets), where there is always another level to reason about the one below.

The three properties just discussed motivate this definition:

**Definition 11.** Let $\Gamma$ be a reflexive graph. A family $A$ of reflexive graphs over $\Gamma$ is:

1. small if, for all $\gamma \in \Gamma$, $A/O(\gamma) \in U$ and for all $\tau \in \Gamma$, $A/R(\tau) \in U$;
2. discrete if, for all $\gamma \in \Gamma$, the reflexive graph on the left of the following diagram is isomorphic to a reflexive graph generated by a set (on the right of the diagram):

$$\text{A}_{\Omega}(\gamma_0) \quad \text{id} \quad X \quad \text{id} \quad \text{A}_{R}(\Gamma_0(\gamma_0))$$

3. proof-irrelevant if, for all $\gamma \in \Gamma$, $A_\Omega(\gamma_0) : A_R(\gamma_0) \rightarrow A_\Omega(\Gamma_0(\gamma_0))$ is injective. We write $RG-Fam_{\ast}(\Gamma)$ for the full subcategory of $RG-Fam(\Gamma)$ consisting of small, discrete, proof-irrelevant reflexive graphs.
A Small, Discrete, Proof Irrelevant Universe

We now remedy this by justifying the following representation result:

Proposition 2. Let \( \Gamma \) be a reflexive graph. There is an equivalence of categories: \( \text{Set}^{\Gamma}(*,*), \mathbf{Rg-Fam}_{\text{sdpi}}(\Gamma) \).

Proposition 2 shows that the reflexive graph model of dependent types can internally represent the semantic System F types and terms, up to isomorphism. This suggests that we can use the reflexive graph * as the basis of our interpretation of U. However, the equivalence of categories in Proposition 2 is too weak to soundly model the type equalities we required for the universe decoder type \( \text{Tm}^{\Gamma}(\mathbf{U}) \). We have, for all \( \gamma \in \Gamma, \text{Ar}_{\mathbf{r}}(\Gamma_{\gamma})(\mathbf{M}_{\gamma}(\gamma)) = \mathbf{M}_{\gamma}(\Gamma_{\gamma}(\gamma)) = \mathbf{N}_{\gamma}(\Gamma_{\gamma}(\gamma)) = \text{Ar}_{\mathbf{r}}(\Gamma_{\gamma})(\mathbf{N}_{\gamma}(\gamma)) \). Similarly, we have \( \text{Ar}_{\mathbf{e}}(\Gamma_{\gamma})(\mathbf{M}_{\gamma}(\gamma)) = \text{Ar}_{\mathbf{e}}(\Gamma_{\gamma})(\mathbf{N}_{\gamma}(\gamma)) \), for all \( \gamma \in \Gamma, \mathbf{r} \). By proof-irrelevance of \( \lambda \), the function \( \text{Ar}_{\mathbf{r}}(\Gamma_{\gamma}), \text{Ar}_{\mathbf{e}}(\Gamma_{\gamma}) \) formed by pairing is injective, hence \( \mathbf{N}_{\gamma} = \mathbf{U} \).

4.4 Interpreting Dependent Products and Natural Numbers

To complete our reflexive graph model of type theory, it remains to provide the interpretations of II-types and the natural number type. Due to the \( \beta \eta \) rules we have chosen for II-types we actually have no choice, up to isomorphism, in how to interpret these types. We spell out the details here to demonstrate the way that type constructors are interpreted in the reflexive graph model.

Dependent Products

The interpretation of II-types directly generalises the interpretation of System \( \mathcal{F} \)-types that we presented in Section 2.1, the additional complication coming from the fact that we need to consider relation transformers rather than just relation preservation (see also Atkey’s relationally parametric model of System \( \mathcal{F} \) [2], which also deals with relation transformers). In the interpretation of II-types, the object level consists of a pair of an object-level function and a relation-level function, connected by three coherence axioms, similar to the definition of semantic terms in Section 4.2. The relation level consists of a pair of object-level interpretations and a relation transformer relating them. Written out in full, the definition is a little daunting, remember that, up to isomorphism, we are essentially forced into this definition.

Definition 12. Let \( \Gamma \) be a reflexive graph, and \( A \in \text{Ty}^{\Gamma}(\mathbf{U}) \) be semantic types in the reflexive graph CwF. Define the semantic type \( \Pi^{\mathbf{r}} A \in \text{Ty}^{\Gamma}(\mathbf{U}) \) as follows:

\[
\Pi^{\mathbf{r}} A = \{ (f_{\gamma}, f_{\gamma}) \mid f_{\gamma} : \forall a_{\gamma} \in \text{Ar}_{\mathbf{r}}(\mathbf{A}_{\gamma}(\gamma)), B_{\mathbf{r}}(\mathbf{A}_{\gamma}(\gamma), a_{\gamma}), a_{\gamma} \in A \}.
\]

\[
\Pi^{\mathbf{e}} A = \{ (f_{\gamma}, f_{\gamma}) \mid (f_{\gamma}, f_{\gamma}) \in \Pi^{\mathbf{r}} A, r_{\gamma} : \forall a_{\gamma} \in \text{Ar}_{\mathbf{r}}(\mathbf{A}_{\gamma}(\gamma)), r_{\gamma} : \forall a_{\gamma} \in \text{Ar}_{\mathbf{r}}(\mathbf{A}_{\gamma}(\gamma), a_{\gamma}), a_{\gamma} \in A \}.
\]

\[
\Pi^{\mathbf{e}} A = \{ (f_{\gamma}, f_{\gamma}) \mid (f_{\gamma}, f_{\gamma}) \in \Pi^{\mathbf{r}} A, r_{\gamma} : \forall a_{\gamma} \in \text{Ar}_{\mathbf{r}}(\mathbf{A}_{\gamma}(\gamma), a_{\gamma}), a_{\gamma} \in A \}.
\]

Lemma 2. Let \( \Gamma \) be a reflexive graph and let \( A \in \text{Ty}^{\Gamma}(\mathbf{U}) \) be a proof-irrelevant family of reflexive graphs. For any pair of semantic terms \( M, N \in \text{Tm}(\Gamma, A) \), \( M_{\gamma} = N_{\gamma} \) implies \( M_{\gamma} = N_{\gamma} \).

Proof. Since \( M \) and \( N \) are semantic terms, and \( M_{\gamma} = N_{\gamma} \), we have, for all \( \gamma \in \Gamma, \text{Ar}_{\mathbf{r}}(\gamma)(\mathbf{M}_{\gamma}^{\mathbf{r}})(\gamma) = \mathbf{M}_{\gamma}(\Gamma_{\gamma}(\gamma)) = N_{\gamma}(\Gamma_{\gamma}(\gamma)) = \text{Ar}_{\mathbf{r}}(\gamma)(\mathbf{N}_{\gamma}(\gamma)) \). Similarly, we have \( \text{Ar}_{\mathbf{e}}(\gamma)(\mathbf{M}_{\gamma}^{\mathbf{e}})(\gamma) = \text{Ar}_{\mathbf{e}}(\gamma)(\mathbf{N}_{\gamma}(\gamma)) \), for all \( \gamma \in \Gamma, \mathbf{r} \). By proof-irrelevance of \( \lambda \), the function \( \text{Ar}_{\mathbf{r}}(\gamma), \text{Ar}_{\mathbf{e}}(\gamma) \) formed by pairing is injective, hence \( M_{\gamma} = N_{\gamma} \).
and, conversely, given \( N \in \text{Ty}(\Gamma, \Pi AB) \), we define \((\Lambda^1 N)\in\text{Ty}(\Gamma, A, B)\) as follows:

\[
(\Lambda^1 N)\sigma(\gamma_o, a_o) = let\ (f_o, f_r) = N_o(\gamma_o)\ in\ f_o(a_o)\ 
(\Lambda^1 N)(\gamma, a_r) = let\ (f^o(\gamma), f^r(\gamma, r)) = N_r(\gamma_r)\ in\ r(a_r)
\]

**Proposition 3.** The reflexive graph \( \text{CwF} \) supports dependent products (Definition 4), using the structure defined in Definition 12.

If the families \( A \in \text{Ty}(\Gamma) \) and \( B \in \text{Ty}(\Gamma, A) \) are small, then so is the family \( \Pi AB \in \text{Ty}(\Gamma) \). This is a consequence of assuming that the set-theoretic universe \( \mathcal{U} \) is a universe of small sets, and so is closed under dependent products and tupling. It is straightforward to check that for closure under discreteness and proof-irrelevance, only the family \( B \) need be discrete and proof-irrelevant.

**Lemma 3.** Let \( A \in \text{Ty}(\Gamma) \) and \( B \in \text{Ty}(\Gamma, A) \). If \( B \) is discrete and proof-irrelevant, then so is \( \Pi AB \in \text{Ty}(\Gamma) \).

**Proposition 4.** Part (c) of Definition 6 is fulfilled by the definition \( \pi MN = \text{repr}(\Pi(T([M]))(T([N]))) \).

If we further assume that our collection of small sets \( \mathcal{U} \) is closed under large products, then we also have an interpretation of an impredicative type-theoretic universe. By Lemma 3, only the codomain type \( B \) need be discrete and proof-irrelevant for \( \Pi AB \) to be discrete and proof-irrelevant. Therefore, if \( \mathcal{U} \) is closed under large products, then \( U \) is closed under impredicative quantification.

**Proposition 5.** Under the assumption of an impredicative universe \( \mathcal{U} \), Definition 7 is fulfilled by \( \pi AM = \text{repr}(\Pi A(\text{Ty}(\Gamma))) \).

The Natural Number Type

For each reflexive graph \( \Gamma \), we define a semantic type \( \text{Nat} \in \text{Ty}(\Gamma) \) as follows, reusing the definition of the reflexive graph of natural numbers from Example 2:

\[
\begin{align*}
\text{Nat}_o(\gamma_0) &= N \\
\text{Nat}_R(\gamma_r) &= N
\end{align*}
\]

Since the family \( \text{Nat} \) does not actually depend on its argument, this definition is trivially natural in \( \Gamma \).

The zero and successor semantic terms \( \text{zero} \in \text{Ty}(\Gamma, \text{Nat}) \) and \( \text{succ} \in \text{Ty}(\Gamma, \text{Nat}, \text{Nat}) \) are defined using the corresponding structure of the natural numbers:

\[
\begin{align*}
\text{zero}_o(\gamma_0) &= 0 \\
\text{zero}_r(\gamma_r) &= 0 \\
\text{succ}_o(\gamma_o, n) &= n + 1 \\
\text{succ}_r(\gamma_r, n) &= n + 1
\end{align*}
\]

It is almost immediate that these are well-defined semantic terms, given the almost trivial structure of the type semantic type \( \text{Nat} \). Likewise, naturality in \( \Gamma \) is trivial.

The natural number recursor \( \text{Nrec} \) is given, for \( A \in \text{Ty}(\Gamma, \text{Nat}) \), \( M^2 \in \text{Ty}(\Gamma, A(\text{zero})) \), and \( M^0 \in \text{Ty}(\Gamma, \text{Nat}, A, A(\text{whNat}\, \text{P(\text{Nat}} \circ \text{succ} \circ \text{P(\text{Nat}})) \), by recursion on the natural number argument:

\[
\begin{align*}
(\text{Nrec}_A(M^\ast, M^\ast))(\alpha_0, 0) &= M^2_o(\alpha_0) \\
(\text{Nrec}_A(M^\ast, M^\ast))(\alpha_r, 0) &= M^2_r(\alpha_r) \\
(\text{Nrec}_A(M^\ast, M^\ast))(\alpha_r, n + 1) &= M^2_o(\alpha_r, n) + 1 \\
(\text{Nrec}_A(M^\ast, M^\ast))(\alpha_r, n + 1) &= M^2_r(\alpha_r, n + 1) \times \text{(Nrec}_A(M^\ast, M^\ast))(\alpha_r, n) \times \text{(Nrec}_A(M^\ast, M^\ast))(\alpha_r, n)
\end{align*}
\]

Equations (5) and (6) specifying the interaction between \( \text{Nrec} \) and \( \text{zero} \) and \( \text{succ} \) hold almost by definition. Naturality in \( \Gamma \) is proved by induction on the natural number argument.

**Proposition 6.** The \( \text{CwF} \) formed from the category of reflexive graphs supports natural numbers (Definition 5). Moreover, the families \( \text{Nat} \in \text{Ty}(\Gamma) \) are small, discrete and proof irrelevant, so part (b) of Definition 6 is fulfilled by \( \text{nat} = \text{repr}(\text{Nat}) \).

4.5 Main Theorem

By the results of the preceding subsections, we have shown that:

**Theorem 1.** The category of reflexive graphs can be given the structure of a \( \text{CwF} \) that supports dependent products and natural numbers. Assuming the existence of a universe \( \mathcal{U} \) of small sets, the category of reflexive graphs supports a predicative universe closed under natural numbers and dependent products, and if \( \mathcal{U} \) is closed under large products then the \( \text{CwF} \) supports an impredicative universe closed under natural numbers and dependent products.

As we argued in the paragraph before Definition 11, our interpretation of the type-theoretic universe \( \mathcal{U} \) supports the identity extension property we identified in Section 2.1 as crucial for relationally parametric models. Thus, we are justified in referring to the model we have constructed here as relationally parametric.

5. Consequences of Parametricity

Having built our parametric model of dependent type theory, we now investigate some of the consequences of our construction. Our careful construction of the interpretation of the type-theoretic universe \( \mathcal{U} \) as capturing the small, discrete, and proof-irrelevant families now bears fruit: we recover many of the interesting consequences of relationally parametricity that have been observed in the non-dependently typed settings of System F and System F\( \omega \).

As a practical matter, calculating directly within the model we have constructed in the previous section is complicated by the “projection-based” presentation of relational interpretations forced by the reflexive graph formalism. We therefore use the following notational shorthands when reasoning within the model:

1. If \( A \in \text{Ty}(\Gamma) \) is a proof-irrelevant family of reflexive graphs, we write \((a, a') \in A_{\omega}(\gamma_r)\) if there exists \( a_r \in A(\gamma_r) \) such that \( A_{\omega}(\gamma_r)(a_r) = a \). By proof-irrelevance the pair \((a, a')\) completely determines \( a_r \) if it exists.
2. If \( f \in (\Pi AB)(\gamma_o) \), we identify \( f \) with its first component, so we can treat it as a function \( \forall a_o \in A_o(\gamma_o). B_o(\gamma_o, a_o) \).
3. We will be liberal in confusing the curried and uncurried versions of the semantic interpretations of inhabitants of dependent products. This decreases the number of parentheses required.

We first prove a simple free theorem, in order to demonstrate reasoning within the model, and the utility of the construction of our interpretation of the type-theoretic universe as only containing proof-irrelevant families. We then go on to adapt Atkey’s proof of the existence of higher-kinded initial algebras [2] to the impredicative dependently typed setting. This proof also relies on proof-irrelevance, and also makes essential use of the discreteness of members of \( U \) in Lemma 7.

5.1 A Free Theorem

We begin by proving a simple free theorem in the style of Wadler [35]. This example shows that the ordinary consequences of relational parametricity are preserved when moving from polymorphic calculi to the richer world of dependent type theory with a universe.

**Theorem 2.** If \( \Gamma \vdash M : \Pi A \vdash \text{Ty} \) then, for any \( \Gamma \vdash X : U \), \( \Gamma \vdash Y : U \), \( \Gamma \vdash f : TX \rightarrow TY \), and \( \Gamma \vdash x : TX \), the equation \( \Gamma \vdash f (M X x) = M Y (f x) \) : \( TY \) is sound when interpreted in the models constructed in Section 4.

**Proof.** Unwinding the definitions, we see that we must establish the following two equations, in which we have reused the letters \( \Gamma \), \( M \), \( X \), \( Y \), \( f \) and \( x \) to stand for the semantic interpretations of the
corresponding syntactic objects, we have applied currying where convenient, and we have identified the small types \( X \) and \( Y \) with their decodings via \( T \):

\[
\forall \gamma_o \in \Gamma_o. \ f_o(\gamma_o, \ M_o(\gamma_o, X_o(\gamma_o), x_o(\gamma_o))) = M_o(\gamma_o, Y_o(\gamma_o), f_o(\gamma_o, x_o(\gamma_o)))
\]

\[
\forall \gamma_r \in \Gamma_r. \ f_r(\gamma_r, M_r(\gamma_r, X_r(\gamma_r), x_r(\gamma_r))) = M_r(\gamma_r, Y_r(\gamma_r), f_r(\gamma_r, x_r(\gamma_r)))
\]

The interpretation of the type \( T Y \) is a proof-inrelevant family, so by Lemma 2 we only need to prove the first of the above equations, from which the second necessarily follows. For a given \( \gamma_o \), we use the relational interpretation \( M_o \) of \( \Gamma_o \), instantiated at \( \Gamma_{rel}(\gamma_o) \):

\[
M_o(\Gamma_{rel}(\gamma_o), - - ) : \forall R \in U_R(\Gamma_{rel}(\gamma_o)). \ T_R(\Gamma_{rel}(\gamma_o), R) \rightarrow T_R(\Gamma_{rel}(\gamma_o), R)
\]

Again unwinding the definitions, and using the fact that \( M_o \) is connected to \( M_r \) via the equations for sources and targets of relations in (10) and (11), we have the following property:

\[
\forall A, A' \in U_{\Omega}(\gamma), \ R \subseteq A \times A', (a, a') \in R. \ (M_o(\gamma_o, a, a'), M_r(\gamma_r, a', a')) \in R
\]

We instantiate \( A \) and \( A' \) with the small discrete reflexive graphs \( X \) and \( Y \), respectively, at \( \gamma_o \). We take \( R = \{(x, y) \mid f_o(\gamma_o, x) = y\} \). Then \( (x_o(\gamma_o), f_o(\gamma_o, x_o(\gamma_o))) \in R \), so \( (M_o(\gamma_o, x_o(\gamma_o)), M_r(\gamma_r, x_o(\gamma_o), x_r(\gamma_o))) \in R \), and the desired equation holds thus by definition of \( R \).

### 5.2 Categories of Indexed Types and Indexed Functors

We now start on our proof that the relationally parametric model that we have constructed supports initial algebras for all indexed functors. To this end, we first define the appropriate category of indexed small types where the carriers of indexed initial algebras reside. The construction of our category of indexed small types is carried out with respect to some fixed context \( \Gamma \). For each judgement \( \Gamma \vdash X \) type, we define the category of \( X \)-indexed small types to have as objects terms \( \Gamma \vdash A : X \rightarrow \text{U} \) and to define morphisms between objects \( \Gamma \vdash A : X \rightarrow \text{U} \) and \( \Gamma \vdash B : X \rightarrow \text{U} \) to be terms \( \Gamma \vdash f : \Pi X. X \cdot (A x) \rightarrow (B x) \). Identities and composition are defined in the obvious way. Composition is associative due to the \( \eta \)-rule for dependent products.

We write \( A \Rightarrow B \) to stand for the type \( \Pi X. X \cdot (A x) \rightarrow (B x) \). We will also use the type \( \Gamma, a : X \rightarrow \text{U}, b : X \rightarrow \text{U} \Rightarrow \Pi X. X \cdot (a x) \rightarrow (b x) \) type, i.e., the type of morphisms where the domain and codomain are abstracted as variables. As a notational convenience, we write “morphism” as shorthand for this type, and similarly for its semantic interpretation morph \( \in T_Y(\Gamma, \Pi X. X \cdot (A x) \rightarrow (B x)) \).

Morphisms in the category of \( X \)-indexed small types induce relations between their domain and codomain objects, as given by the following definition. This definition is the \( X \)-indexed generalisation of the functional relation from the free theorem in Section 5.1.

**Definition 13.** Let \( \Gamma \vdash A : X \rightarrow \text{U} \) and \( \Gamma \vdash B : X \rightarrow \text{U} \). Given a morphism \( \Gamma \vdash f : A \Rightarrow B \), we define \( \langle f \rangle \), called the graph relation of \( f \), to be the following element of \( (\Pi X. U) \cdot R \):

\[
\lambda \gamma_r \in \Gamma_r. \ (A_o(\Gamma_{rel}(\gamma_r)), B_o(\Gamma_{rel}(\gamma_r))), \lambda x_r \in X_o(\gamma_r). \ (A_o(\Gamma_{rel}(\gamma_r))(X_o(x_r)), (\gamma_r)(x_r)), B_o(\Gamma_{rel}(\gamma_r))(X_o(x_r)), \{ (a, b) \mid f_o(\Gamma_{rel}(\gamma_r), X_o(x_r), (\gamma_r)(x_r), a, b) \in B_o(\Gamma_{rel}(\gamma_r))(x_r) \})
\]

The following lemma is a direct consequence of Definition 13.

**Lemma 4.** Let \( \Gamma \vdash A : X \rightarrow \text{U}, \Gamma \vdash B : X \rightarrow \text{U} \), and \( \Gamma \vdash f : A \Rightarrow B \). Then for all \( \gamma_o \in \Gamma_o \), \( f_o(\gamma_o, \text{id}) \in \text{morph}_R(\Gamma_{rel}(\gamma_o), (f)(\Gamma_{rel}(\gamma_o))), B_o(\Gamma_{rel}(\gamma_o)) \).

**Indexed Functors** \( X \)-indexed initial algebras are defined in terms of \( X \)-indexed functors, which we now define. An \( X \)-indexed functor is a pair \((F, Fmap)\) of terms

\[
\Gamma \vdash f : X \rightarrow (Y \rightarrow X) \rightarrow X \rightarrow (Y \rightarrow X) \rightarrow X \rightarrow (Y \rightarrow X)
\]

where \( Fmap \) preserves the identities and composition of the category of \( X \)-indexed small types when interpreted in the reflexive graph model. Indexed functors interact nicely with graph relations, as captured by the following **Graph Lemma**:

**Lemma 5.** Let \( \Gamma \vdash f : A \Rightarrow B \). For all \( \gamma_o \in \Gamma_o \) and \( \gamma_r \in \Gamma_r \cdot (\gamma_o) \cdot x_r \in X_R(\Gamma_{rel}(\gamma_o)) \), if \( (a, b) \in F_r(\Gamma_{rel}(\gamma_o), (f)(\Gamma_{rel}(\gamma_o)))(x_r) \) then \( (a, b) \in (Fmap A f)(\Gamma_{rel}(\gamma_o))(x_r) \).

### 5.3 The Category of \( F \)-algebras

For an \( X \)-indexed functor \((F, Fmap)\), \( F \)-algebras are initial objects in the category of \( F \)-algebras, which we now define. Fix an \( X \)-indexed functor \((F, Fmap)\). The objects of the category of \( F \)-algebras are pairs of terms \( (A, k^A) \) where \( \Gamma \vdash A : X \rightarrow \text{U} \) and \( \Gamma \vdash k^A : FA \Rightarrow A \). Morphisms between \( (A, k^A) \) and \( (B, k^B) \) are terms \( \Gamma \vdash h : A \Rightarrow B \) such that, if \( \circ \) is the composition of the \( X \)-indexed small types, then

\[
\Gamma \vdash k^B \circ Fmap A h \circ k^A : FA \Rightarrow B
\]

An initial \( F \)-algebra is an initial object in the category of \( F \)-algebras, i.e., an \( F \)-algebra \((\mu F, in)\) such that there exists a term

\[
\Gamma \vdash fold : \Pi A : X \rightarrow \text{U}. (FA \Rightarrow A) \rightarrow (\mu F \Rightarrow A)
\]

such that the following two equations hold. The first equation states that \( fold \) always yields \( F \)-algebra morphisms:

\[
\Gamma \vdash fold A k^A \cdot x (\text{in } x M) = k^A x \cdot (Fmap \mu F A (fold A k^A) x M \cdot (T(\text{A} x))
\]

The second equation states that the \( F \)-algebra morphisms generated by \( fold \) are unique, as required by initiality:

\[
\Gamma \vdash A : X \rightarrow \text{U} \big\Rightarrow \Gamma \vdash k^A : FA \Rightarrow A \\
\Gamma \vdash h : \mu F \Rightarrow A \big\Rightarrow h \text{ is an } F \text{-algebra morphism}
\]

\[
\Gamma \vdash h \circ fold A k^A : \mu F \Rightarrow A
\]

Morphisms between \( F \)-algebras have the following relational property in terms of their graph relations. This property will be useful when we prove that initial \( F \)-algebras always exist.

**Lemma 6.** Let \((A, k^A)\) and \((B, k^B)\) be \( F \)-algebras. If \( h \) is an \( F \)-algebra morphism from \((A, k^A)\) to \((B, k^B)\), then for all \( \gamma_o \in \Gamma_o \),

\[
(k^B o \gamma_o, k^A o \gamma_o) \in \text{morph}_R(\Gamma_{rel}(\gamma_o), F_o(\Gamma_{rel}(\gamma_o), (h)(\Gamma_{rel}(\gamma_o))), (h)(\Gamma_{rel}(\gamma_o)))
\]

### 5.4 Construction of Initial \( F \)-algebras

Having defined all the necessary background, we now show that, within the impredicative version of our relationally parametric model of dependent types, initial \( F \)-algebras exist for all \( X \)-indexed functors \((F, Fmap)\). Given an \( X \)-indexed functor
\(F, \text{Fmap})\), we make the following definitions:

\[
\mu F = \lambda x : X. \Pi A : X \to U. (FA \Rightarrow A) \Rightarrow T(A x)
\]

\[
fold = \lambda A : X \to U. \lambda x : A. \lambda A'. \lambda F : T(A x) \to A. \lambda k : A k^A \Rightarrow k^A \in \lambda x : X. \lambda e : T(\mu F x). e A k^A.
\]

\[
in = \lambda x : X. \lambda e : T(\mu F x). \lambda A : X \to U. \lambda A'. \lambda F : T(A x) \to A. \lambda k : A k^A \Rightarrow k^A.
\]

\[k^A x : F \mu F A (fold A k^A) \times e\]

Note that, up to notational changes induced by our presentation of the universe \(U\), and the more general setting, these definitions are identical to those given by Atkey [2] for constructing higher-kindened initial algebras in a relationally parametric model of System \(F\).

The proof that these definitions actually give an initial \(F\)-algebra follows from two applications of the following lemma:

**Lemma 7.** The following equation holds when interpreted in the reflexive graph model:

\[
\Gamma \vdash A : X \to U \quad \Gamma \vdash k^A : F A \Rightarrow A \\
\Gamma \vdash B : X \to U \quad \Gamma \vdash k^B : F B \Rightarrow B \\
\Gamma \vdash h : A \Rightarrow B \\
\Gamma \vdash x : X \\
\Gamma \vdash M : T(\mu F x)
\]

\[
\Gamma \vdash h x (M A k^A) = M B k^B : T(B x)
\]

**Proof.** By Lemma 2 we need only prove the object-level part of the equation. Given \(\gamma_0 \in \Gamma\), we instantiate \(M\) with \(\Gamma_{\gamma_0}(\gamma_0)\), the graph relation \((h) (\Gamma_{\gamma_0}(\gamma_0))\), and the pair \((k^A, x_\gamma), (k^B, x_\gamma)\), by Lemma 6, to obtain:

\[
(M_\gamma(\gamma_0, A_\gamma(\gamma_0), k^A(\gamma_0)), M_\gamma(\gamma_0, B_\gamma(\gamma_0), k^B(\gamma_0))) \in (h)(\Gamma_{\gamma_0}(\gamma_0))(X_{\gamma_0}(\gamma_0)(x_\gamma(\gamma_0)))
\]

Unfolding the graph relation, this statement is equivalent to:

\[
(h_\gamma(\gamma_0, x_\gamma(\gamma_0)), M_\gamma(\gamma_0, A_\gamma(\gamma_0), k^A(\gamma_0)), M_\gamma(\gamma_0, B_\gamma(\gamma_0), k^B(\gamma_0))) \in B_{\gamma}(\Gamma_{\gamma_0}(\gamma_0), X_{\gamma_0}(\gamma_0)(x_\gamma(\gamma_0))).\]

By the identity extension property that is built into the reflexive graph model, the relation \(B_{\gamma}(\Gamma_{\gamma_0}(\gamma_0), X_{\gamma_0}(\gamma_0)(x_\gamma(\gamma_0)))\) is to \(U_{\gamma_0}(\gamma_0, x_\gamma(\gamma_0))\). By the discreteness of elements of the universe \(U\), this relation is, up to isomorphism, the equality relation on the small set \(B_{\gamma}(\gamma_0, x_\gamma(\gamma_0))\). We thus have:

\[
h_\gamma(\gamma_0, x_\gamma(\gamma_0)), M_\gamma(\gamma_0, A_\gamma(\gamma_0), k^A(\gamma_0)) = M_\gamma(\gamma_0, B_\gamma(\gamma_0), k^B(\gamma_0))
\]

We can now state the following key property of our model:

**Theorem 3.** Any \(X\)-indexed functor \((F, \text{Fmap})\) has an initial \(F\)-algebra.

**Proof.** We must verify that the two equations in Section 5.3 hold. The first equation, stating that \(fold\) always generates \(F\)-algebra morphisms, follows directly by using the \(\beta\)-reduction rules of the type theory. The second equation requires relational parametricity. Expanding the definitions, we see that we need to prove that \(h x M = M A k^A\). By Lemma 7, together with the fact that \(h\) is an \(F\)-algebra morphism, we have that \(h x (M (\mu F x)) = M A k^A\). Using Lemma 7 again, this time with an arbitrary \(F\)-algebra \((B, k^B)\), and setting \(h = fold B k^B\), yields \(M \mu F x = M B k^B\). Extensionality therefore gives \(M \mu F x = M\), and thus \(h x M = M A k^A\), as required.

Despite the considerably more general setting, the above proof is almost identical to that given by Atkey for higher-kindened initial algebras in a relationally parametric model of System \(F\). The proofs of other results presented by Atkey — e.g., the existence of an extensional equality type and existentials — also translate to the impredicative dependently typed setting with little change.

6. **Conclusion and Directions for Future Work**

We have presented a parametric model of dependent types that has two significant features. First, our model is based on reflexive graphs, which naturally generalise existing relationally parametric models of System \(F\): just as the construction of a parametric model for System \(F\) can be seen as moving from a model based on Set to a model based on Set\(^\text{RG}\), so our parametric model of dependent types can be seen as moving from a fibration \(p : \text{Fam}(\text{Set}) \to \text{Set}\) to a fibration \(p^{\text{RG}} : \text{Fam}(\text{Set})^{\text{RG}} \to \text{Set}^{\text{RG}}\). Second, our model supports the derivation of key results, such as the existence of initial algebras. We believe ours is the first parametric model, syntactic or semantic, to establish the dependently-typed generalisation of this key property of relationally parametric models of System \(F\).

The most exciting avenue for future work is in comparing the reflexive graph model we have presented here with the strikingly similar groupoid and \(\in\)-groupoid models of type theory [32]. Reflexive graphs can be seen as “categories without composition”, whereas groupoids are categories with inverses. In light of the many useful results derivable from relational parametricity, further investigation of the reduced amount of structure of reflexive graphs looks promising (Robinson has already investigated “Parametricity as Isomorphism”: parametricity where types are interpreted as groupoids [29]). In particular, Voevodsky’s univalence axiom states that isomorphic types are actually equal. But Reynolds’ relational parametricity reveals that often types need only be related, not isomorphic, for them to be indistinguishable to programs. Is there a Reynoldsian version of univalence waiting to be formulated?

**References**


