A Relationally Parametric Model of Dependent Type Theory

Robert Atkey[1], Neil Ghani[2], and Patricia Johann[3]

[1] bob.atkey@gmail.com

[2] University of Strathclyde, neil.ghani@strath.ac.uk

[3] Appalachian State University, johannp@appstate.edu

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 im-predicative dependent type theory. The significance of our models is twofold. Firstly, in the impredicative variant we are able to de-duce 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

1. Introduction

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 Fω 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 Fω 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 System Fω type:

$$\lambda\tau.\forall\alpha{:}{*}{\to}{*}.(\forall\sigma.\sigma{\to}\alpha\,\sigma){\to}(\forall\sigma,\sigma'.\alpha\,\sigma{\to}\alpha\,\sigma'{\to}\alpha(\sigma{\times}\sigma')){\to}\alpha\,\tau$$

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 Fω interesting indexed types like length-indexed vectors. However, due to the rigid separation of types and terms in System Fω, 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:

$$\lambda x : X.\ \Pi A : X \to \mathsf{U}.\ (\Pi x : X.\mathsf{T}(FAx) \to \mathsf{T}(Ax)) \to \mathsf{T}(A\ x)$$

(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 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 Π-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 `-impredicative-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

$$\frac{(\alpha : *) \in \Gamma}{\Gamma \vdash \alpha : *} \qquad \frac{\Gamma \vdash A : * \qquad \Gamma \vdash B : *}{\Gamma \vdash A \to B : *} \qquad \frac{\Gamma, \alpha : * \vdash A : *}{\Gamma \vdash \forall \alpha{:}*.\, A : *}$$

Figure 1. Well-kinded Types in System F

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-kinded types of System F are shown in Figure 1. By these rules, well-kinded types are either type variables, function types, or universally quantified types. Kinding contexts Γ 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-kinded type $\Gamma \vdash A : *$ is assigned two interpretations: an *underlying interpretation* $[\![\Gamma \vdash A : *]\!]_o$ that interprets a type as a function on the universe $\mathcal{U}$ of small sets, and a *relational interpretation* $[\![\Gamma \vdash A : *]\!]_r$ that interprets a type as a transformer of relations between underlying interpretations. If, for any two sets $X, Y \in \mathcal{U}$, $\mathrm{Rel}(X, Y)$ denotes the set of binary relations between X and Y, and if $|\Gamma|$ stands for the number of members of Γ, then the underlying interpretation and the relational interpretation of $\Gamma \vdash A : *$ have the following meta-theoretic types:

$$[\![\Gamma \vdash A : *]\!]_o \in \mathcal{U}^{|\Gamma|} \to \mathcal{U}$$

$$[\![\Gamma \vdash A : *]\!]_r \in \forall \gamma_1, \gamma_2 \in \mathcal{U}^{|\Gamma|}.\ \mathrm{Rel}^{|\Gamma|}(\gamma_1, \gamma_2) \to \mathrm{Rel}([\![\Gamma \vdash A : *]\!]_o\gamma_1, [\![\Gamma \vdash A : *]\!]_o\gamma_2)$$

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:

$$\forall \gamma \in \mathcal{U}^{|\Gamma|}.\ [\![\Gamma \vdash A : *]\!]_r\ \gamma\ \gamma\ \mathrm{Eq}_\gamma = \mathrm{Eq}_{[\![\Gamma \vdash A : *]\!]_o\gamma}$$

Here, Eq_X is the equality relation on the set X, and Eq_γ for a tuple of sets $\gamma = (X_1, ..., X_{|\Gamma|})$ is the tuple $(\mathrm{Eq}_{X_1}, ..., \mathrm{Eq}_{X_{|\Gamma|}})$.

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 $\Gamma \vdash \forall \alpha{:}*.A : *$. 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:

$$\begin{aligned}&[\![\Gamma \vdash \forall \alpha{:}*.A : *]\!]_o\gamma = \\ &\quad \{x : \forall X \in \mathcal{U}.\ [\![\Gamma, \alpha : * \vdash A : *]\!]_o(\gamma, X) \mid \\ &\qquad \forall X, Y, R \in \mathrm{Rel}(X, Y). \\ &\qquad\quad (x\,X, x\,Y) \in [\![\Gamma, \alpha : * \vdash A : *]\!]_r(\gamma, X)(\gamma, Y)(\mathrm{Eq}_\gamma, R)\}\end{aligned}$$

$$\begin{aligned}&[\![\Gamma \vdash \forall \alpha{:}*.A : *]\!]_r\gamma_1\gamma_2\rho = \\ &\quad \{(x, y) \mid \forall X, Y, R \in \mathrm{Rel}(X, Y). \\ &\qquad (x\,X, y\,Y) \in [\![\Gamma, \alpha : * \vdash A : *]\!]_r(\gamma_1, X)(\gamma_2, Y)(\rho, R)\}\end{aligned}$$

We will refer back to this definition when we construct the interpretation of dependent products (Π-types) in our relationally parametric model of dependent types in Section 4.4.

Interpretation of Terms A well-typed term of System F is determined by the typing judgement $\Gamma; \Theta \vdash M : A$ where Γ is a kinding context, Θ is a typing context consisting of variable name/type pairs $x_i : A_i$ where each A_i is well-kinded with respect to Γ, and A is also well-kinded with respect to Γ. We write $\Gamma \vdash \Theta$ to indicate when all the types in the typing context Θ are well-kinded with respect to the kinding context Γ. We omit the (entirely standard) typing rules that generate the judgement $\Gamma; \Theta \vdash M : A$.

The interpretation of types that we sketched above is extended to an interpretation of typing contexts Θ by tupling together the interpretations of the constituent types and using the standard logical relations interpretation of product types. Each well-typed term $\Gamma; \Theta \vdash M : A$ is interpreted as a family $[\![\Gamma; \Theta \vdash M : A]\!]_o$ of functions from the underlying interpretation of the typing context Θ to the underlying interpretation of the type A, indexed by the underlying interpretations of the free type variables:

$$[\![\Gamma; \Theta \vdash M : A]\!]_o : \forall \gamma \in \mathcal{U}^{|\Gamma|}.\ [\![\Gamma \vdash \Theta]\!]_o\gamma \to [\![\Gamma \vdash A : *]\!]_o\gamma$$

Because the underlying interpretations and relational interpretations of types are mutually defined, we must simultaneously define $[\![\Gamma; \Theta \vdash M : A]\!]_o$ and prove the following property, which states that the relational interpretations of Θ and A are preserved:

$$\begin{aligned}&\forall \gamma_1, \gamma_2 \in \mathcal{U}^{|\Gamma|}, \rho \in \mathrm{Rel}^{|\Gamma|}(\gamma_1, \gamma_2), \\ &\quad \theta_1 \in [\![\Gamma \vdash \Theta]\!]_o\gamma_1, \theta_2 \in [\![\Gamma \vdash \Theta]\!]_o\gamma_2. \\ &\qquad (\theta_1, \theta_2) \in [\![\Gamma \vdash \Theta]\!]_r\gamma_1\gamma_2\rho \Rightarrow \\ &\qquad\quad ([\![\Gamma; \Theta \vdash M : A]\!]_o\ \gamma_1\ \theta_1, [\![\Gamma; \Theta \vdash M : A]\!]_o\ \gamma_2\ \theta_2) \in \\ &\qquad\qquad [\![\Gamma \vdash A : *]\!]_r\gamma_1\gamma_2\rho\end{aligned}$$

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 $* \to *$, 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 $X \in \mathcal{U}$, 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* $(\Gamma_O, \Gamma_R, \Gamma_{\mathit{refl}}, \Gamma_{\mathit{src}}, \Gamma_{\mathit{tgt}})$ *comprising a pair of sets* Γ_O *and* Γ_R*, together with three functions*

$$\begin{tikzcd}
\Gamma_O \arrow[d, "\Gamma_{\mathit{refl}}"] \\
\Gamma_R \arrow[u, bend left=60, "\Gamma_{\mathit{src}}"] \arrow[u, bend right=60, "\Gamma_{\mathit{tgt}}"']
\end{tikzcd}$$

such that $\Gamma_{\mathit{src}} \circ \Gamma_{\mathit{refl}} = \mathrm{id}_{\Gamma_O} = \Gamma_{\mathit{tgt}} \circ \Gamma_{\mathit{refl}}$*. A* morphism *$f$ between reflexive graphs Γ and Δ is a pair of functions (f_o, f_r) that makes every square the following diagram commute:*

$$\begin{tikzcd}
\Gamma_O \arrow[d, "\Gamma_{\mathit{refl}}"] \arrow[rr, "f_o"] & & \Delta_O \arrow[d, "\Delta_{\mathit{refl}}"] \\
\Gamma_R \arrow[u, bend left=60, "\Gamma_{\mathit{src}}"] \arrow[u, bend right=60, "\Gamma_{\mathit{tgt}}"'] \arrow[rr, "f_r"] & & \Delta_R \arrow[u, bend left=60, "\Delta_{\mathit{src}}"] \arrow[u, bend right=60, "\Delta_{\mathit{tgt}}"']
\end{tikzcd}$$

The set Γ_O is called the set of *objects*, and Γ_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 Γ_O to be a set of *sets* and Γ_R to be a set of *relations* over those sets. Each relation $\gamma_r \in \Gamma_R$ is thus a relation between $\Gamma_{\mathit{src}}(\gamma_r)$, and $\Gamma_{\mathit{tgt}}(\gamma_r)$ and each set $\gamma_o \in \Gamma_O$ has a distinguished relation $\Gamma_{\mathit{refl}}(\gamma_o)$ relating γ_o to itself. This intuition underlies the following example.

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

$$*_O = \mathcal{U} \qquad *_R = \{(X, Y, R) \mid X, Y \in \mathcal{U}, R \in \mathrm{Rel}(X, Y)\}$$

$$*_{\mathit{refl}}(X) = (X, X, \mathrm{Eq}_X) \quad *_{\mathit{src}}(X, Y, R) = X \quad *_{\mathit{tgt}}(X, Y, R) = Y$$

The objects of this reflexive graph are thus small sets from $\mathcal{U}$, and its relations are ordinary binary relations between such sets. For each $X \in \mathcal{U}$, the function $*_{\mathit{refl}}$ 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 Γ_O to be a specific set and Γ_R to be a reflexive relation on that set, represented as a (multi-)set of pairs of elements that are related. Then Γ_{refl} ensures that Γ_R is reflexive, while Γ_{src} and Γ_{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:*

$$\begin{array}{ll} \mathit{nat}_O = \mathbb{N} & \mathit{nat}_{\mathit{refl}}(n) = n \\ \mathit{nat}_R = \mathbb{N} & \mathit{nat}_{\mathit{src}}(n) \ = n \\ & \mathit{nat}_{\mathit{tgt}}(n) \ = n \end{array}$$

The objects of this reflexive graph are just the natural numbers. For any $n \in \mathit{nat}_R$, we have $\mathit{nat}_{\mathit{src}}(n) = \mathit{nat}_{\mathit{tgt}}(n) = n$, so two natural numbers are related if and only if they are equal. Generalising the construction of nat, for any set X there is a reflexive graph $(X, X, \mathrm{id}_X, \mathrm{id}_X, \mathrm{id}_X)$ 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 $(1, 1, \mathrm{id}_1, \mathrm{id}_1, \mathrm{id}_1)$ is terminal in the category of reflexive graphs.

Reflexive graphs can be equivalently defined as covariant presheaves over the category $\mathrm{RG} = \bullet \underset{\xleftarrow{\mathit{tgt}}}{\overset{\xleftarrow{\mathit{src}}}{\xrightarrow{\mathit{refl}}}} \bullet$ where $\mathit{src} \circ \mathit{refl} = \mathrm{id} = \mathit{tgt} \circ \mathit{refl}$. Morphisms between reflexive graphs are exactly natural transformations between presheaves in $\mathrm{Set}^{\mathrm{RG}}$. 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-kinded type $\Gamma \vdash A : *$ is interpreted as a reflexive graph morphism from the $|\Gamma|$-fold product of the reflexive graph $*$ from Example 1 to $*$ 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 definitions 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 Γ, we think of the homset $\mathrm{Set}^{\mathrm{RG}}(\Gamma, *)$ as providing interpretations of types over Γ.

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-kinded typing context Γ is therefore also interpreted as an object of $\mathrm{Set}^{\mathrm{RG}}(\Gamma, *)$. 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 Θ to the interpretation of the type A, i.e., as a morphism in the category $\mathrm{Set}^{\mathrm{RG}}(\Gamma, *)$:

Definition 2. *Let $A, B \in \mathrm{Set}^{\mathrm{RG}}(\Gamma, *)$. A morphism from A to B is a function $M : \forall \gamma_o \in \Gamma_O.\ A_o(\gamma_o) \to B_o(\gamma_o)$ such that for all $\gamma_r \in \Gamma_R$, if $A_r(\gamma_r) = (A_1, A_2, R_A \subseteq A_1 \times A_2)$ and $B_r(\gamma_r) = (B_1, B_2, R_B \subseteq B_1 \times B_2)$ then for all $(a_1, a_2) \in R_A$, $(M\ (\Gamma_{\mathit{src}}(\gamma_r))\ a_1, M\ (\Gamma_{\mathit{tgt}}(\gamma_r))\ a_2) \in R_B$. This definition makes each homset $\mathrm{Set}^{\mathrm{RG}}(\Gamma, *)$ 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, 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 nat, dependent product types $\Pi x{:}A.B$, and a Tarski-style 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 β- and η-equality rules for dependent product types, as well as β-equality rules for the natural numbers. Since our universe is Tarski-style, we have an explicit "decoder" type former T to take terms of type U to actual types.

The type theories we present in this section are close to the theories 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

Judgement	Intuitive meaning
Γ ctxt	Γ is a well-formed context
$\Gamma \vdash A$ type	A is a well-formed type in the context Γ
$\Gamma \vdash M : A$	the term M has type A in the context Γ
$\Gamma \vdash A = B$ type	A and B are equal types in the context Γ
$\Gamma \vdash M = N : A$	the terms M and N are equal at the type A in the context Γ

Figure 2. Judgement forms

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:

$$\begin{array}{rcl}\Gamma & ::= & \diamond \mid \Gamma, x : A \\ A, B, C & ::= & \Pi x{:}A.B \mid \mathsf{Nat} \mid \mathsf{U} \mid \mathsf{T}\,M \\ M, N, P & ::= & x \mid \lambda x.M \mid MN \mid \mathsf{ze} \mid \mathsf{su}\,M \mid \\ & & \mathsf{Nrec}(x.A, M^z, x\,p.M^s, N) \mid \mathsf{nat} \mid \pi x{:}M.N\end{array}$$

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 $\mathsf{Nrec}(x.A, M^z, x\,p.M^s, N)$, the variable x is considered bound in A, and the variables x and p are considered bound in M^s.

The well-formed types and well-typed terms are defined using a collection of five 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 β- and η-equality for Π-types and β-equality for 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 T, 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-π in Figure 3 and β-U-π 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 $\pi a{:}\mathsf{U}.\ a \to a : \mathsf{U}$ — 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

Context formation (Γ ctxt)

$$\frac{}{\diamond \text{ ctxt}}\ (\textsc{Emp}) \qquad \frac{\Gamma \text{ ctxt} \quad \Gamma \vdash A \text{ type}}{\Gamma, x : A \text{ ctxt}}\ (\textsc{Ext})$$

Type Formers ($\Gamma \vdash A$ type)

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma, x : A \vdash B \text{ type}}{\Gamma \vdash \Pi x{:}A.\ B \text{ type}}\ (\textsc{Ty-}\Pi)$$

$$\frac{}{\Gamma \vdash \mathsf{Nat} \text{ type}}\ (\textsc{Ty-Nat}) \qquad \frac{}{\Gamma \vdash \mathsf{U} \text{ type}}\ (\textsc{Ty-U})$$

$$\frac{\Gamma \vdash M : \mathsf{U}}{\Gamma \vdash \mathsf{T}M \text{ type}}\ (\textsc{Ty-T})$$

Typing Rules ($\Gamma \vdash M : A$)

$$\frac{\Gamma, x : A, \Gamma' \text{ ctxt}}{\Gamma, x : A, \Gamma' \vdash x : A}\ (\textsc{Var})$$

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash A = B \text{ type}}{\Gamma \vdash M : B}\ (\textsc{Conv})$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x{:}A.M : \Pi x{:}A.B}\ (\textsc{Lam})$$

$$\frac{\Gamma \vdash M : \Pi x{:}A.B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B[N/x]}\ (\textsc{App})$$

$$\frac{}{\Gamma \vdash \mathsf{ze} : \mathsf{Nat}}\ (\textsc{Ze}) \qquad \frac{\Gamma \vdash n : \mathsf{Nat}}{\Gamma \vdash \mathsf{su}\,n : \mathsf{Nat}}\ (\textsc{Su})$$

$$\frac{\begin{array}{c}\Gamma, x : \mathsf{Nat} \vdash A \text{ type} \\ \Gamma \vdash M^z : A[\mathsf{ze}/x] \\ \Gamma, n : \mathsf{Nat}, p : A[n/x] \vdash M^s : A[\mathsf{su}\,n/x] \\ \Gamma \vdash n : \mathsf{Nat}\end{array}}{\Gamma \vdash \mathsf{Nrec}(x.\ A, M^z, n\,p.\ M^s, n) : A[n/x]}\ (\textsc{Nrec})$$

$$\frac{}{\Gamma \vdash \mathsf{nat} : \mathsf{U}}\ (\textsc{U-nat})$$

$$\frac{\Gamma \vdash M : \mathsf{U} \quad \Gamma, x : \mathsf{T}M \vdash N : \mathsf{U}}{\Gamma \vdash \pi x{:}M.N : \mathsf{U}}\ (\textsc{U-}\pi)$$

Figure 3. Typing Rules of Martin-Löf Type Theory with Universe

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma, x : A \vdash M : \mathsf{U}}{\Gamma \vdash \pi x{:}A.M : \mathsf{U}}\ (\text{U-}\pi\text{'})$$

$$\frac{\Gamma \vdash A \text{ type} \quad \Gamma, x : A \vdash M : \mathsf{U}}{\Gamma \vdash \mathsf{T}(\pi x{:}A.\ M) = \Pi x{:}A.\ \mathsf{T}M \text{ type}}\ (\beta\text{-U-}\pi\text{'})$$

Figure 5. Rules for Impredicative Quantification

Type Equality Rules ($\Gamma \vdash A = B$ type)

$$\frac{}{\Gamma \vdash \mathsf{T}\ \mathsf{nat} = \mathsf{Nat}\ \text{type}}\ (\beta\text{-U-NAT}) \qquad \frac{\Gamma \vdash M : \mathsf{U} \qquad \Gamma, x : \mathsf{T}M \vdash N : \mathsf{U}}{\Gamma \vdash \mathsf{T}(\pi x{:}M.N) = \Pi x{:}\mathsf{T}M.\ \mathsf{T}N\ \text{type}}\ (\beta\text{-U-}\pi)$$

elided: type equality is an equivalence relation and a congruence with respect to all type formers.

Term Equality Rules ($\Gamma \vdash M = N : A$)

$$\frac{\Gamma, x : A \vdash M : B \qquad \Gamma \vdash N : A}{\Gamma \vdash (\lambda x{:}A.\ M)N = M[N/x] : B[N/x]}\ (\Pi\text{-}\beta) \qquad \frac{\Gamma \vdash M : \Pi x{:}A.B}{\Gamma \vdash (\lambda x{:}A.\ Mx) = M : \Pi x{:}A.B}\ (\Pi\text{-}\eta)$$

$$\frac{\Gamma, x : \mathsf{Nat} \vdash A\ \text{type} \qquad \Gamma \vdash M^z : A[\mathsf{ze}/x] \qquad \Gamma, n : \mathsf{Nat}, p : A[n/x] \vdash M^s : A[\mathsf{su}\ n/x]}{\Gamma \vdash \mathsf{Nrec}(x.A, M^z, n\,p.M^s, \mathsf{ze}) = M^z : A[\mathsf{ze}/x]}\ (\beta\text{-ZE})$$

$$\frac{\Gamma, x : \mathsf{Nat} \vdash A\ \text{type} \qquad \Gamma \vdash M^z : A[\mathsf{ze}/x] \qquad \Gamma, n : \mathsf{Nat}, p : A[n/x] \vdash M^s : A[\mathsf{su}\ n/x] \qquad \Gamma \vdash n : \mathsf{Nat}}{\Gamma \vdash \begin{array}{l}\mathsf{Nrec}(x.\ A, M^z, n\ p.\ M^s, \mathsf{su}\ n) \\ \quad = M^s[n/n, \mathsf{Nrec}(x.A, M^z, n\,p.M^s, n)/p]\end{array} : P[\mathsf{su}\ n/x]}\ (\beta\text{-SU})$$

elided: term equality is an equivalence relation and a congruence with respect to all term formers.

Figure 4. Type and Term Equality Judgements

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 present the extra structure required for dependent products (Definition 4), a natural number type (Definition 5), and a universe of small types (Definition 6). We loosely follow Hofmann's presentation [15], except that we specialise for our particular type theory.

The Basic Structure CwFs provide enough structure to interpret the contexts, types, and terms of a type theory, along with simultaneous substitutions and their action on types and terms. Judgemental equality on types and terms is modelled directly as equality in the model; in particular, there is no semantic structure corresponding to the rule CONV. Here, we define comprehension in terms of a bijection between sets of morphisms and semantic terms. This is equivalent to Hofmann's definition of comprehension in terms of projections and weakening. We treat projection and weakening as derived structure, as described after the following definition.

Definition 3. *A* Category with Families *(CwF) consists of:*

1. *A category $\mathcal{C}$, whose objects are intended as the semantic interpretations of contexts and whose morphisms are intended as the interpretations of simultaneous substitutions;*
2. *For each $\Gamma \in \mathrm{Ob}(\mathcal{C})$, a collection $\mathrm{Ty}(\Gamma)$ of* semantic types*;*
3. *For each $\Gamma \in \mathrm{Ob}(\mathcal{C})$ and semantic type $A \in \mathrm{Ty}(\Gamma)$, a collection $\mathrm{Tm}(\Gamma, A)$ of* semantic terms*;*
4. *For every morphism $f : \Gamma \to \Delta$ in $\mathcal{C}$, a function*
$$-\{f\} : \mathrm{Ty}(\Delta) \to \mathrm{Ty}(\Gamma)$$
interpreting type substitution, and for every $A \in \mathrm{Ty}(\Delta)$, a function
$$-\{f\} : \mathrm{Tm}(\Delta, A) \to \mathrm{Tm}(\Gamma, A\{f\})$$
interpreting term substitution, such that the following hold:
$$A\{\mathrm{id}_\Gamma\} = A \tag{1}$$
$$A\{f\}\{g\} = A\{f \circ g\} \tag{2}$$
$$M\{\mathrm{id}_\Gamma\} = M \tag{3}$$
$$M\{f\}\{g\} = M\{f \circ g\} \tag{4}$$
Equations 3 and 4 are well-typed by virtue of Equations 1 and 2, respectively.
5. *A chosen terminal object 1 in $\mathcal{C}$, interpreting the empty context;*
6. *For each $\Gamma \in \mathrm{Ob}(\mathcal{C})$ and $A \in \mathrm{Ty}(\Gamma)$ an object $\Gamma.A \in \mathrm{Ob}(\mathcal{C})$ (called the* comprehension of *A) such that there is a bijection*
$$\mathcal{C}(\Delta, \Gamma.A) \cong \{(f, M) \mid f : \Delta \to \Gamma, M \in \mathrm{Tm}(\Delta, A\{f\})\}$$
that is natural in Δ.

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

Given $f : \Delta \to \Gamma$ and $M \in \mathrm{Tm}(\Delta, A\{f\})$, we write $\langle f, M\rangle$ for the associated morphism $\Delta \to \Gamma.A$ in $\mathcal{C}$ given by the isomorphism in point (6) of Definition 3. Conversely, given a morphism $f : \Delta \to \Gamma.A$ in $\mathcal{C}$, we write $f^{\#1} : \Delta \to \Gamma$ and $f^{\#2} \in \mathrm{Tm}(\Delta, A\{f^{\#1}\})$ for the associated morphism and semantic term. The following definitions derive projection and weakening structure from Definition 3, and also define the semantic counterpart of a simultaneous substitution of a term for a variable.

1. For any $\Gamma \in \mathrm{Ob}(\mathcal{C})$ and $A \in \mathrm{Ty}(\Gamma)$, the *first projection* morphism $\mathrm{p}^\Gamma_A : \Gamma.A \to \Gamma$ is defined as $p^\Gamma_A = \mathrm{id}^{\#1}_{\Gamma.A}$. This is used to interpret weakening by discarding a single variable.
2. For any $\Gamma \in \mathrm{Ob}(\mathcal{C})$ and $A \in \mathrm{Ty}(\Gamma)$, the *second projection* $\mathrm{q}^\Gamma_A \in \mathrm{Tm}(\Gamma.A, A\{p^\Gamma_A\})$ is defined as $q^\Gamma_A = \mathrm{id}^{\#2}_{\Gamma.A}$. This is used to interpret the rule VAR.
3. For any $\Gamma, \Delta \in \mathrm{Ob}(\mathcal{C})$, $A \in \mathrm{Ty}(\Gamma)$ and morphism $f : \Delta \to \Gamma$, the *weakening* of f by A, $\mathrm{wk}_A f : \Delta.A\{f\} \to \Gamma.A$ is defined as $\mathrm{wk}_A f = \langle f \circ \mathrm{p}^\Delta_{A\{f\}}, \mathrm{q}^\Delta_{A\{f\}}\rangle$. This is used to lift simultaneous substitutions to work in larger contexts.
4. For any $\Gamma \in \mathrm{Ob}(\mathcal{C})$, $A \in \mathrm{Ty}(\Gamma)$ and $M \in \mathrm{Tm}(\Gamma, A)$, we define the morphism $\overline{M} : \Gamma \to \Gamma.A$ as $\overline{M} = \langle \mathrm{id}_\Gamma, M\rangle$. Mor-

phisms built from semantic terms are used to build simultaneous substitutions that substitute (semantic) terms for variables.

Dependent Products Hofmann's presentation of the structure for interpreting dependent product types in a CwF admits the possibility of not supporting the η-rule. But since the type theory we consider in this paper does include the η-rule Π-η, we can use the following compact definition to describe the requirements ensuring that a CwF supports dependent products.

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

Definition 4 gives a semantic counterpart of the abstraction rule LAM. We define the following semantic counterpart of the application rule APP. For Γ, A and B as in Definition 4, and $M \in \mathrm{Tm}(\Gamma, \Pi AB)$ and $N \in \mathrm{Tm}(\Gamma, A)$, we define $\mathrm{App}^{\Gamma}_{A,B}(M, N) \in \mathrm{Tm}(\Gamma, B\{\overline{N}\})$ by $\mathrm{App}^{\Gamma}_{A,B} = (\Lambda^{-1}(M))\{\overline{N}\}$.

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 ZE, SU, and NREC.

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

$$Nrec_A(M^z, M^s)\{\overline{zero}\} = M^z \tag{5}$$

$$Nrec_A(M^z, M^s)\{\mathrm{wk}_{Nat}\ p^{\Gamma}_{Nat} \circ \overline{succ}\} = M^s\{\overline{Nrec_A(M^z, M^s)}\} \tag{6}$$

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-NAT, U-π, and U-π'. We treat the predicative and impredicative cases separately.

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

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

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 Π-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* $p : \mathrm{Fam(Set)} \to \mathrm{Set}$, where the category Fam(Set) has as objects pairs (X, P), where $X \in \mathrm{Set}$ and $P : X \to \mathrm{Set}$, 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 $\mathrm{Set}^{\mathrm{RG}}$. Fortunately, there is a natural way to do this. By standard results about fibrations (see Exercise 1.8.8 in Jacobs [17]), the functor $p^{\mathrm{RG}} : \mathrm{Fam(Set)}^{\mathrm{RG}} \to \mathrm{Set}^{\mathrm{RG}}$, defined as $p^{\mathrm{RG}}(F) = p \circ F$, is also a fibration. This hints that objects of $\mathrm{Fam(Set)}^{\mathrm{RG}}$ are the natural interpretations of dependent types in a reflexive graph model of parametricity, and that the functor p^{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 $\mathrm{Fam(Set)}^{\mathrm{RG}}$, we see that an object of $\mathrm{Fam(Set)}^{\mathrm{RG}}$ is equivalent to a pair (Γ, A) of a reflexive graph Γ, together with a family A of reflexive graphs that is *over* Γ, in the following sense:

Definition 8. *Let Γ be a reflexive graph. A* family of reflexive graphs over Γ *is a 5-tuple $A = (A_O, A_R, A_{\mathit{refl}}, A_{src}, A_{tgt})$, where:*

$$\begin{aligned}
A_O &: \Gamma_O \to \mathrm{Set}\\
A_R &: \Gamma_R \to \mathrm{Set}\\
A_{\mathit{refl}} &: \forall \gamma_o \in \Gamma_O.\ A_O(\gamma_o) \to A_R(\Gamma_{\mathit{refl}}(\gamma_o))\\
A_{src} &: \forall \gamma_r \in \Gamma_R.\ A_R(\gamma_r) \to A_O(\Gamma_{src}(\gamma_r))\\
A_{tgt} &: \forall \gamma_r \in \Gamma_R.\ A_R(\gamma_r) \to A_O(\Gamma_{tgt}(\gamma_r))
\end{aligned}$$

such that the following equations hold:

$$\forall \gamma_o \in \Gamma_O.\ A_{src}(\Gamma_{\mathit{refl}}(\gamma_o)) \circ A_{\mathit{refl}}(\gamma_o) = \mathrm{id} \tag{7}$$

$$\forall \gamma_o \in \Gamma_O.\ A_{tgt}(\Gamma_{\mathit{refl}}(\gamma_o)) \circ A_{\mathit{refl}}(\gamma_o) = \mathrm{id} \tag{8}$$

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

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

$$\begin{aligned}
M_o &: \forall \gamma_o \in \Gamma_O.\ A_O(\gamma_o) \to B_O(\gamma_o)\\
M_r &: \forall \gamma_r \in \Gamma_R.\ A_R(\gamma_r) \to B_R(\gamma_r)
\end{aligned}$$

such that

$$\begin{aligned}
&\forall \gamma_o \in \Gamma_o, a_o \in A_O(\gamma_o).\\
&\qquad B_{\mathit{refl}}(\gamma_o)(M_o\ \gamma_o\ a_o) = M_r\ (\Gamma_{\mathit{refl}}(\gamma_o))\ (A_{\mathit{refl}}(\gamma_o)(a_o))\\
&\forall \gamma_r \in \Gamma_R, a_r \in A_R(\gamma_r).\\
&\qquad B_{src}(\gamma_r)(M_r\ \gamma_r\ a_r) = M_o\ (\Gamma_{src}(\gamma_r))\ (A_{src}(\gamma_r)(a_r))\\
&\forall \gamma_r \in \Gamma_R, a_r \in A_R(\gamma_r).\\
&\qquad B_{tgt}(\gamma_r)(M_r\ \gamma_r\ a_r) = M_o\ (\Gamma_{tgt}(\gamma_r))\ (A_{tgt}(\gamma_r)(a_r))
\end{aligned}$$

Families of reflexive graphs over a reflexive graph Γ and morphisms between them form a category RG-Fam(Γ).

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 Γ, we define the collection of *semantic types* $\mathrm{Ty}(\Gamma)$ to be the collection of families of reflexive graphs over Γ as defined in Definition 8. Given a reflexive graph Γ and a family A of reflexive graphs over Γ, we define the collection of *semantic terms* $\mathrm{Tm}(\Gamma, A)$ to be the collection of morphisms from the terminal family $(\lambda\gamma_o \in \Gamma_O.*, \lambda\gamma_r \in \Gamma_R.*, \lambda\gamma_o \in \Gamma_O.\mathrm{id}_*, \lambda\gamma_r \in \Gamma_R.\mathrm{id}_*, \lambda\gamma_r \in \Gamma_R.\mathrm{id}_*)$ of reflexive graphs over Γ to A, as in Definition 9. Spelling this definition out, suppressing the contribution of the terminal family, a semantic term $M \in \mathrm{Tm}(\Gamma, A)$ is defined as a pair of functions (M_o, M_r):

$$M_o : \forall\gamma_o \in \Gamma_O.\, A_O(\gamma_o) \qquad M_r : \forall\gamma_r \in \Gamma_R.\, A_R(\gamma_r)$$

such that

$$\forall\gamma_o \in \Gamma_O.\, A_{\mathit{refl}}(\gamma_o)(M_o(\gamma_o)) = M_r(\Gamma_{\mathit{refl}}(\gamma_o)) \tag{9}$$

$$\forall\gamma_r \in \Gamma_R.\, A_{\mathit{src}}(\gamma_r)(M_r(\gamma_r)) = M_o(\Gamma_{\mathit{src}}(\gamma_r)) \tag{10}$$

$$\forall\gamma_r \in \Gamma_R.\, A_{\mathit{tgt}}(\gamma_r)(M_r(\gamma_r)) = M_o(\Gamma_{\mathit{tgt}}(\gamma_r)) \tag{11}$$

For each reflexive graph morphism $f : \Delta \to \Gamma$, substitution $A\{f\}$ in semantic types A and substitution $M\{f\}$ in semantic terms are both defined by pre-composition. Comprehension structure is given by the following definition, which can be seen as the dependent version of the cartesian product of reflexive graphs used in Section 2.2 to interpret the non-dependent kinding and typing contexts of System F.

Definition 10. *Let* Γ *be a reflexive graph, and let* $A \in \mathrm{Ty}(\Gamma)$ *be a semantic type. Define the* comprehension $\Gamma.A$ *as the following reflexive graph:*

$$\begin{array}{ll}
(\Gamma.A)_O & = \{(\gamma_o, a_o) \mid \gamma_o \in \Gamma_O, a_o \in A_O(\gamma_o)\} \\
(\Gamma.A)_R & = \{(\gamma_r, a_r) \mid \gamma_r \in \Gamma_R, a_r \in A_R(\gamma_r)\} \\
(\Gamma.A)_{\mathit{refl}}(\gamma_o, a_o) & = (\Gamma_{\mathit{refl}}(\gamma_o), A_{\mathit{refl}}(\gamma_o)(a_o)) \\
(\Gamma.A)_{\mathit{src}}(\gamma_r, a_r) & = (\Gamma_{\mathit{src}}(\gamma_r), A_{\mathit{src}}(\gamma_r)(a_r)) \\
(\Gamma.A)_{\mathit{tgt}}(\gamma_r, a_r) & = (\Gamma_{\mathit{tgt}}(\gamma_r), A_{\mathit{tgt}}(\gamma_r)(a_r))
\end{array}$$

Given $f : \Delta \to \Gamma$ *and* $M \in \mathrm{Tm}(\Delta, A\{f\})$*, define* $\langle f, M\rangle : \Delta \to \Gamma.A$ *as follows:*

$$\begin{array}{l}
\langle f, M\rangle_o(\delta_o) = (f_o(\delta_o), M_o(f_o(\delta_o))) \\
\langle f, M\rangle_r(\delta_r) = (f_r(\delta_r), M_r(f_r(\delta_r)))
\end{array}$$

Given $f : \Delta \to \Gamma.A$*, define* $f^{\#1} : \Delta \to \Gamma$ *and* $f^{\#2} \in \mathrm{Tm}(\Delta, A\{f^{\#1}\})$ *as follows:*

$$\begin{array}{l}
(f^{\#1})_o(\delta_o) = \text{let } (\gamma_o, a_o) = f_o(\delta_o) \text{ in } \gamma_o \\
(f^{\#1})_r(\delta_r) = \text{let } (\gamma_r, a_r) = f_r(\delta_r) \text{ in } \gamma_r \\
(f^{\#2})_o(\delta_o) = \text{let } (\gamma_o, a_o) = f_o(\delta_o) \text{ in } a_o \\
(f^{\#2})_r(\delta_r) = \text{let } (\gamma_r, a_r) = f_r(\delta_r) \text{ in } a_r
\end{array}$$

The next proposition follows by mechanical checking of the requirements in Definition 3.

Proposition 1. *The category of reflexive graphs has the structure of a CwF, where the collection of semantic types* $\mathrm{Ty}(\Gamma)$ *is the collection of families of reflexive graphs over* Γ*, and the collection of semantic terms* $\mathrm{Tm}(\Gamma, A)$ *is the collection of morphisms from the terminal family of reflexive graphs over* Γ *to* A*.*

4.3 Interpreting the Universe of Small Types

By Definition 6, the basic structure we require to interpret the type-theoretic universe comprises semantic types $U \in \mathrm{Ty}(\Gamma)$ and $T \in \mathrm{Ty}(\Gamma.U)$. So T takes elements of U to semantic types. But which semantic types (i.e., families of reflexive graphs) ought to be elements of U? Unlike in the case of Π-types and the natural number type, where we are constrained in our interpretation up to isomorphism by the equations of the calculus (Section 4.4), we are free to select any collection of semantic types, as long as it contains the natural number type and is closed under Π-types.

Since the reflexive graph model is a covariant presheaf model over the reflexive graph category RG (Section 2.2), a plausible choice would be to use Hofmann and Streicher's general definition of the interpretation of a type-theoretic universe in presheaf models given some set-theoretic universe $\mathcal{U}$ [16]. Spelling out their construction in the setting of reflexive graphs yields:

$$\begin{array}{l}
U^{\mathrm{HS}}_O(\gamma_o) = \text{the set of small reflexive graphs} \\
U^{\mathrm{HS}}_R(\gamma_r) = \{(A, B, R \in \mathcal{U}, R_{\mathit{src}} : R \to A_O, R_{\mathrm{tgt}} : R \to B_O) \mid \\
\qquad\qquad A, B \text{ are small reflexive graphs}\}
\end{array}$$

Thus, the objects of U^{HS} are 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 $\mathcal{U}$. Put mathematically, if we write $\text{RG-Fam}_s(\Gamma)$ for the subcategory of small families of reflexive graphs over some reflexive graph Γ, then we have $\mathrm{Tm}(\Gamma, U^{\mathrm{HS}}) \cong \mathrm{Ob}(\text{RG-Fam}_{\mathrm{s}}(\Gamma))$.

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 $*$ of System F types in our interpretation of the type-theoretic universe U.

Recall the definition of the reflexive graph $*$ 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 $\mathcal{U}$ of small sets. Secondly, the reflexive structure of $*$ 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 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 ∞-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* Γ *be a reflexive graph. A family* A *of reflexive graphs over* Γ *is:*

1. small *if, for all* $\gamma_o \in \Gamma_O$, $A_O(\gamma_o) \in \mathcal{U}$ *and for all* $\gamma_r \in \Gamma_R$, $A_R(\gamma_r) \in \mathcal{U}$*;*
2. discrete *if, for all* $\gamma_o \in \Gamma_O$*, 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):*

$$\begin{tikzcd}
A_O(\gamma_o) \arrow[rr, "\cong"] \arrow[d, "A_{\mathit{refl}}(\gamma_o)"] & & X \arrow[d, "\mathrm{id}"] \\
A_R(\Gamma_{\mathit{refl}}(\gamma_o)) \arrow[rr, "\cong"] \arrow[u, bend left, "A_{\mathit{src}}(\Gamma_{\mathit{refl}}(\gamma_o))"] \arrow[u, bend right, "A_{\mathit{tgt}}(\Gamma_{\mathit{refl}}(\gamma_o))"'] & & X \arrow[u, bend left, "\mathrm{id}"] \arrow[u, bend right, "\mathrm{id}"']
\end{tikzcd}$$

3. proof-irrelevant *if, for all* $\gamma_r \in \Gamma_R$, $\langle A_{\mathit{src}}(\gamma_r), A_{\mathit{tgt}}(\gamma_r)\rangle : A_R(\gamma_r) \to A_O(\Gamma_{\mathit{src}}(\gamma_r)) \times A_O(\Gamma_{\mathit{tgt}}(\gamma_r))$ *is injective.*

We write RG-Fam$_{\mathrm{sdpi}}(\Gamma)$ *for the full subcategory of RG-Fam*(Γ) *consisting of small, discrete, proof-irrelevant reflexive graphs.*

In parts (2) and (3) of this definition, we have chosen to enforce the conditions of discreteness and proof-irrelevance only up to isomorphism. This laxness will be important when showing that the interpretation of the type-theoretic universe of small types is closed under dependent products in Section 4.4. Definition 11 is justified by the following representation result:

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

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 operator T in Figure 4 (specifically, the semantic counterparts of these equalities in Definitions 6 and 7). We now remedy this by refining the definition of our reflexive graph $*$ to provide an interpretation for U with the required properties.

A Small, Discrete, Proof Irrelevant Universe For any reflexive graph Γ we define semantic types $U \in \mathrm{Ty}(\Gamma)$ and $T \in \mathrm{Ty}(\Gamma.\, U)$:

$$\begin{aligned}
U_O(\gamma_o) &= \text{the set of small discrete reflexive graphs} \\
U_R(\gamma_r) &= \{(A, B, R \in \mathcal{U}, R_{src} : R \to A_O, R_{tgt} : R \to B_O \mid \\
&\qquad A, B \text{ are small discrete reflexive graphs}, \\
&\qquad \langle R_{src}, R_{tgt}\rangle : R \to A_O \times B_O \text{ is injective}\}
\end{aligned}$$

$$\begin{aligned}
U_{\mathit{refl}}(\gamma_o)(A) &= (A, A, A_R, A_{src}, A_{tgt}) \\
U_{src}(\gamma_r)(A, B, R, R_{src}, R_{tgt}) &= A \\
U_{tgt}(\gamma_r)(A, B, R, R_{src}, R_{tgt}) &= B
\end{aligned}$$

If A is a discrete reflexive graph, then $\langle A_{src}, A_{tgt}\rangle : A_R \to A_O \times A_O$ is necessarily injective, so the definition of U_{refl} makes sense. We define the semantic types $T \in \mathrm{Ty}(\Gamma.\, U)$ as follows:

$$\begin{aligned}
T_O(\gamma_o, A) &= A_O \\
T_R(\gamma_r, (A, B, R, R_{src}, R_{tgt})) &= R \\
T_{\mathit{refl}}(\gamma_o, A) &= A_{\mathit{refl}} \\
T_{src}(\gamma_r, (A, B, R, R_{src}, R_{tgt})) &= R_{src} \\
T_{tgt}(\gamma_r, (A, B, R, R_{src}, R_{tgt})) &= R_{tgt}
\end{aligned}$$

The definitions of U and T we have just presented fulfil part (a) of Definition 6. The following lemma states that we have successfully internalised the subcategories $\text{RG-Fam}_{\mathrm{sdpi}}(\Gamma)$.

Lemma 1. *For any $M \in \mathrm{Tm}(\Gamma, U)$, the family $T\{\overline{M}\}$ is small, discrete, and proof-irrelevant and this mapping forms a bijection of sets: $T\{\overline{-}\} : \mathrm{Tm}(\Gamma, U) \cong \mathrm{Ob}(\text{RG-Fam}_{\mathrm{sdpi}}(\Gamma))$, natural in Γ.*

The fact that this lemma gives us a bijection is useful because it means that to show that the universe U is closed under some semantic type former, we need only show that the semantic type former is closed under the condition of being a small, discrete, and proof-irrelevant family of reflexive graphs. Below, we define the small counterparts of dependent products and the natural number type – parts (b) and (c) of Definition 6 – using the right-to-left direction of this bijection to obtain a representation in U of a small, discrete, proof-irrelevant family, which we write as *repr* : $\mathrm{Ob}(\text{RG-Fam}_{\mathrm{sdpi}}(\Gamma)) \to \mathrm{Tm}(\Gamma, U)$.

We note the following crucial property of proof-irrelevant families. By proof-irrelevance, there is exactly one way that any two object-level elements can be related. Therefore, for any two terms of proof-irrelevant type, if the object-level interpretations are equal then so are the relation-level interpretations. This lemma plays an important role in showing that the interpretation of the type-theoretic universe we have defined is closed under dependent products (Lemma 3, below), and also in the applications of relational parametricity that we describe in Section 5.

Lemma 2. *Let Γ be a reflexive graph and let $A \in \mathrm{Ty}(\Gamma)$ be a proof-irrelevant family of reflexive graphs. For any pair of semantic terms $M, N \in \mathrm{Tm}(\Gamma, A)$, $M_o = N_o$ implies $M_r = N_r$.*

Proof. Since M and N are semantic terms, and $M_o = N_o$, we have, for all $\gamma_r \in \Gamma_R$, $A_{src}(\gamma_r)(M_r(\gamma_r)) = M_o(\Gamma_{src}(\gamma_r)) = N_o(\Gamma_{src}(\gamma_r)) = A_{src}(\gamma_r)(N_r(\gamma_r))$. Similarly, we have $A_{tgt}(\gamma_r)(M_r(\gamma_r)) = A_{tgt}(\gamma_r)(N_r(\gamma_r))$, for all $\gamma_r \in \Gamma_R$. By proof-irrelevance of A, the function $\langle A_{src}(\gamma_r), A_{tgt}(\gamma_r)\rangle$ formed by pairing is injective, hence $M_r = N_r$. $\square$

4.4 Interpreting Dependent Products and Natural Numbers

To complete our reflexive graph model of type theory, it remains to provide the interpretations of Π-types and the natural number type. Due to the $\beta\eta$-rules we have chosen for Π-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 Π-types directly generalises the interpretation of System F $\forall$-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 Fω [2], which also deals with relation transformers). In the interpretation of Π-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 Γ be a reflexive graph, and $A \in \mathrm{Ty}(\Gamma)$ and $B \in \mathrm{Ty}(\Gamma.A)$ be semantic types in the reflexive graph CwF. Define the semantic type $\Pi AB \in \mathrm{Ty}(\Gamma)$ as follows:*

$$\begin{aligned}
&(\Pi AB)_O(\gamma_o) = \\
&\quad \{\ (f_o, f_r) \mid \\
&\qquad f_o : \forall a_o \in A_O(\gamma_o).\ B_O(\gamma_o, a_o), \\
&\qquad f_r : \forall a_r \in A_R(\Gamma_{\mathit{refl}}(\gamma_o)).\ B_R(\Gamma_{\mathit{refl}}(\gamma_o), a_r), \\
&\qquad \forall a_r \in A_R(\Gamma_{\mathit{refl}}(\gamma_o)).\ B_{src}(\Gamma_{\mathit{refl}}(\gamma_o), a_r)(f_r\ a_r) = \\
&\qquad\qquad\qquad f_o(A_{src}(\Gamma_{\mathit{refl}}(\gamma_o))(a_r)), \\
&\qquad \forall a_r \in A_R(\Gamma_{\mathit{refl}}(\gamma_o)).\ B_{tgt}(\Gamma_{\mathit{refl}}(\gamma_o), a_r)(f_r\ a_r) = \\
&\qquad\qquad\qquad f_o(A_{tgt}(\Gamma_{\mathit{refl}}(\gamma_o))(a_r)), \\
&\qquad \forall a_o \in A_O(\gamma_o).\ B_{\mathit{refl}}(\gamma_o, a_o)(f_o\ a_o) = f_r(A_{\mathit{refl}}(\gamma_o)(a_o))\ \}
\end{aligned}$$

$$\begin{aligned}
&(\Pi AB)_R(\gamma_r) = \\
&\quad \{\ ((f_o^{src}, f_r^{src}), (f_o^{tgt}, f_r^{tgt}), r) \mid \\
&\qquad (f_o^{src}, f_r^{src}) \in (\Pi AB)_O(\Gamma_{src}(\gamma_r)), \\
&\qquad (f_o^{tgt}, f_r^{tgt}) \in (\Pi AB)_O(\Gamma_{tgt}(\gamma_r)), \\
&\qquad r : \forall a_r \in A_R(\gamma_r).\ B_R(\gamma_r, a_r), \\
&\qquad \forall a_r \in A_R(\gamma_r).\ B_{src}(\gamma_r, a_r)(r\ a_r) = f_o^{src}(A_{src}(\gamma_r)(a_r)), \\
&\qquad \forall a_r \in A_R(\gamma_r).\ B_{tgt}(\gamma_r, a_r)(r\ a_r) = f_o^{tgt}(A_{tgt}(\gamma_r)(a_r))\ \}
\end{aligned}$$

$$\begin{aligned}
&(\Pi AB)_{\mathit{refl}}(\gamma_o)(f_o, f_r) = ((f_o, f_r), (f_o, f_r), f_r) \\
&(\Pi AB)_{src}(\gamma_r)(f^{src}, f^{tgt}, r) = f^{src} \\
&(\Pi AB)_{tgt}(\gamma_r)(f^{src}, f^{tgt}, r) = f^{tgt}
\end{aligned}$$

Given a semantic term $M \in \mathrm{Tm}(\Gamma.A, B)$, we define $(\Lambda M) \in \mathrm{Tm}(\Gamma, \Pi AB)$ as follows:

$$\begin{aligned}
&(\Lambda M)_o(\gamma_o) = (\lambda a_o.\ M_o(\gamma_o, a_o), \lambda a_r.\ M_r(\Gamma_{\mathit{refl}}(\gamma_o), a_r)) \\
&(\Lambda M)_r(\gamma_r) = \\
&\qquad ((\Lambda M)_o(\Gamma_{src}(\gamma_r)), (\Lambda M)_o(\Gamma_{tgt}(\gamma_r)), \lambda a_r.\ M_r(\gamma_r, a_r))
\end{aligned}$$

and, conversely, given $N \in \mathrm{Tm}(\Gamma, \Pi AB)$ *, we define* $(\Lambda^{-1}N) \in \mathrm{Tm}(\Gamma.A, B)$ *as follows:*

$$\begin{aligned}(\Lambda^{-1}N)_o(\gamma_o, a_o) &= \textit{let } (f_o, f_r) = N_o(\gamma_o) \textit{ in } f_o(a_o)\\ (\Lambda^{-1}N)_r(\gamma_r, a_r) &= \textit{let } (f^{src}, f^{tgt}, r) = N_r(\gamma_r) \textit{ in } r(a_r)\end{aligned}$$

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

If the families $A \in \mathrm{Ty}(\Gamma)$ and $B \in \mathrm{Ty}(\Gamma.A)$ are small, then so is the family $\Pi AB \in \mathrm{Ty}(\Gamma)$. 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* $A \in \mathrm{Ty}(\Gamma)$ *and* $B \in \mathrm{Ty}(\Gamma.A)$*. If* B *is discrete and proof-irrelevant, then so is* $\Pi AB \in \mathrm{Ty}(\Gamma)$*.*

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

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 ΠAB 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* $\pi AM = \mathit{repr}(\Pi A(T\{\overline{M}\}))$.

The Natural Number Type For each reflexive graph Γ, we define a semantic type $\mathit{Nat} \in \mathrm{Ty}(\Gamma)$ as follows, reusing the definition of the reflexive graph of natural numbers from Example 2:

$$\begin{aligned}\mathit{Nat}_O(\gamma_o) &= \mathbb{N} & \mathit{Nat}_{\mathit{refl}}(\gamma_o)(n) &= n\\ \mathit{Nat}_R(\gamma_r) &= \mathbb{N} & \mathit{Nat}_{\mathit{src}}(\gamma_r)(n) &= n\\ & & \mathit{Nat}_{\mathit{tgt}}(\gamma_r)(n) &= n\end{aligned}$$

Since the family Nat does not actually depend on its argument, this definition is trivially natural in Γ.

The zero and successor semantic terms $\mathit{zero} \in \mathrm{Tm}(\Gamma, \mathit{Nat})$ and $\mathit{succ} \in \mathrm{Tm}(\Gamma.\mathit{Nat}, \mathit{Nat})$ are defined using the corresponding structure of the natural numbers:

$$\begin{aligned}\mathit{zero}_o(\gamma_o) &= 0 & \mathit{succ}_o(\gamma_o, n) &= n + 1\\ \mathit{zero}_r(\gamma_r) &= 0 & \mathit{succ}_r(\gamma_r, n) &= n + 1\end{aligned}$$

It is almost immediate that these are well-defined semantic terms, given the trivial structure of the semantic type Nat. Likewise, naturality in Γ is trivial.

The natural number recursor Nrec is given, for $A \in \mathrm{Ty}(\Gamma.\mathit{Nat})$, $M^z \in \mathrm{Tm}(\Gamma, A\{\overline{\mathit{zero}}\})$, and $M^s \in \mathrm{Tm}(\Gamma.\mathit{Nat}.A, A\{\mathrm{wk}_{\mathit{Nat}}\, \mathrm{p}^{\Gamma}_{\mathit{Nat}} \circ \overline{\mathit{succ}} \circ \mathrm{p}^{\Gamma.\mathit{Nat}}_{A}\})$, by recursion on the natural number argument:

$$\begin{aligned}&(\mathit{Nrec}_A(M^z, M^s))_o(\gamma_o, 0) &&= M^z_o(\gamma_o)\\ &(\mathit{Nrec}_A(M^z, M^s))_r(\gamma_r, 0) &&= M^z_r(\gamma_r)\\ &(\mathit{Nrec}_A(M^z, M^s))_o(\gamma_o, n + 1) = \\ &\qquad\qquad M^s_o(\gamma_o, n, (\mathit{Nrec}_A(M^z, M^s))_o(\gamma_o, n))\\ &(\mathit{Nrec}_A(M^z, M^s))_r(\gamma_r, n + 1) = \\ &\qquad\qquad M^s_r(\gamma_r, n, (\mathit{Nrec}_A(M^z, M^s))_r(\gamma_r, n))\end{aligned}$$

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

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

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 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 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 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 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 Fω.

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 $A \in \mathrm{Ty}(\Gamma)$ is a proof-irrelevant family of reflexive graphs, we write $(a, a') \in A_R(\gamma_r)$ if there exists $a_r \in A_R(\gamma_r)$ such that $A_{\mathit{src}}(\gamma_r)(a_r) = a$ and $A_{\mathit{src}}(\gamma_r)(a_r) = a'$. By proof-irrelevance the pair (a, a') completely determines a_r if it exists.
2. If $f \in (\Pi AB)_O(\gamma_o)$, we identify f with its first component, so we can treat it as a function $\forall a_o \in A_O(\gamma_o).\ B_O(\gamma_o, a_o)$. Similarly, if $f \in (\Pi AB)_R(\gamma_r)$ we identify f with its relation transformer component, which has type $\forall a_r \in A_R(\gamma_r).\ B_R(\gamma_r, a_r)$.
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* $\Gamma \vdash M : \Pi a : \mathsf{U}.\ \mathsf{T}a \to \mathsf{T}a$ *then, for any* $\Gamma \vdash X : \mathsf{U}$, $\Gamma \vdash Y : \mathsf{U}$, $\Gamma \vdash f : \mathsf{T}X \to \mathsf{T}Y$*, and* $\Gamma \vdash x : \mathsf{T}X$*, the equation* $\Gamma \vdash f\ (M\ X\ x) = M\ Y\ (f\ x) : \mathsf{T}Y$ *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 Γ, 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:

$$\begin{aligned}\forall\gamma_o \in \Gamma_O.\ f_o(\gamma_o, M_o(\gamma_o, X_o(\gamma_o), x_o(\gamma_o))) &=\\ M_o(\gamma_o, Y_o(\gamma_o), f_o(\gamma_o, x_o(\gamma_o)))\end{aligned}$$

$$\begin{aligned}\forall\gamma_r \in \Gamma_R.\ f_r(\gamma_r, M_r(\gamma_r, X_r(\gamma_r), x_r(\gamma_r))) &=\\ M_r(\gamma_r, Y_r(\gamma_r), f_r(\gamma_r, x_r(\gamma_r)))\end{aligned}$$

The interpretation of the type $\mathsf{T}Y$ is a proof-irrelevant 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 γ_o, we use the relational interpretation M_r of M, instantiated at $\Gamma_{\mathit{refl}}(\gamma_o)$:

$$\begin{aligned}&M_r(\Gamma_{\mathit{refl}}(\gamma_o), -, -):\\ &\forall R \in U_R(\Gamma_{\mathit{refl}}(\gamma_o)).\ T_R(\Gamma_{\mathit{refl}}(\gamma_o), R) \to T_R(\Gamma_{\mathit{refl}}(\gamma_o), R)\end{aligned}$$

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

$$\begin{aligned}\forall A, A' \in U_O(\gamma_o), R \subseteq A_O \times A'_O, (a, a') \in R.&\\ (M_o(\gamma_o, A, a), M_o(\gamma_o, A', a')) \in R&\end{aligned}$$

We instantiate A and A' with the small discrete reflexive graphs X and Y, respectively, at γ_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 $(M_o(\gamma_o, X_O(\gamma_o), x_o(\gamma_o)),$ $M_o(\gamma_o, Y_O(\gamma_o), f_o(\gamma_o, x_o(\gamma_o)))) \in R$, 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 Γ. For each judgement $\Gamma \vdash X$ type, we define the category of X-indexed small types to have as objects terms $\Gamma \vdash A : X \to \mathsf{U}$, and define morphisms between objects $\Gamma \vdash A : X \to \mathsf{U}$ and $\Gamma \vdash B : X \to \mathsf{U}$ to be terms $\Gamma \vdash f : \Pi x{:}X.\ \mathsf{T}(A\ x) \to \mathsf{T}(B\ x)$. Identities and composition are defined in the obvious way. Composition is associative due to the η-rule for dependent products.

We write $A \Rightarrow B$ to stand for the type $\Pi x{:}X.\ \mathsf{T}(A\ x) \to \mathsf{T}(B\ x)$. We will also use the type $\Gamma, a : X \to \mathsf{U}, b : X \to \mathsf{U} \vdash \Pi x{:}X.\ \mathsf{T}(a\ x) \to \mathsf{T}(b\ x)$ type, i.e., the type of morphisms where the domain and codomain are abstracted as variables. As a notational convenience, we write "morph" as shorthand for this type, and similarly for its semantic interpretation $\mathrm{morph} \in \mathrm{Ty}(\Gamma.\Pi XU.\Pi XU)$.

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* $\Gamma \vdash A : X \to \mathsf{U}$ *and* $\Gamma \vdash B : X \to \mathsf{U}$. *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* $(\Pi XU)_R$:

$$\begin{aligned}&\lambda\gamma_r \in \Gamma_R.\ (A_o(\Gamma_{\mathit{src}}(\gamma_r)), B_o(\Gamma_{\mathit{tgt}}(\gamma_r)),\\ &\quad\lambda x_r \in X_R(\gamma_r).\\ &\quad\quad(A_o(\Gamma_{\mathit{src}}(\gamma_r))(X_{\mathit{src}}(\gamma_r)(x_r)),\\ &\quad\quad B_o(\Gamma_{\mathit{tgt}}(\gamma_r))(X_{\mathit{tgt}}(\gamma_r)(x_r)),\\ &\quad\quad\{(a, b) \mid (f_o(\Gamma_{\mathit{src}}(\gamma_r), X_{\mathit{src}}(\gamma_r)(x_r), a), b) \in\\ &\quad\quad\quad B_r(\gamma_r)(x_r)\}))\end{aligned}$$

The following lemma is a direct consequence of Definition 13.

Lemma 4. *Let* $\Gamma \vdash A : X \to \mathsf{U}$, $\Gamma \vdash B : X \to \mathsf{U}$, *and* $\Gamma \vdash f : A \Rightarrow B$. *Then for all* $\gamma_o \in \Gamma_O$, $(f_o(\gamma_o), \mathrm{id}) \in \mathrm{morph}_R(\Gamma_{\mathit{refl}}(\gamma_o), \langle f \rangle(\Gamma_{\mathit{refl}}(\gamma_o)), B_r(\Gamma_{\mathit{refl}}(\gamma_o)))$.

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

$$\begin{aligned}&\Gamma \vdash F : (X \to \mathsf{U}) \to (X \to \mathsf{U})\\ &\Gamma \vdash \mathit{Fmap} : \Pi A, B{:}X \to \mathsf{U}.\ (A \Rightarrow B) \to (FA \Rightarrow FB)\end{aligned}$$

where *Fmap* preserves the identities and composition of the category of X-indexed small types when interpretated 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* $x_r \in X_R(\Gamma_{\mathit{refl}}(\gamma_o))$, *if* $(a, b) \in F_r(\Gamma_{\mathit{refl}}(\gamma_o), \langle f \rangle(\Gamma_{\mathit{refl}}(\gamma_o)))(x_r)$ *then* $(a, b) \in \langle \mathit{Fmap}\ A\ B\ f \rangle(\Gamma_{\mathit{refl}}(\gamma_o))(x_r)$.

5.3 The Category of F-algebras

For an X-indexed functor (F, Fmap), initial 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 $\Gamma \vdash A : X \to \mathsf{U}$ 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 category of X-indexed small types, then

$$\Gamma \vdash k^B \circ \mathit{Fmap}\ A\ B\ h = h \circ k^A : FA \Rightarrow B$$

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

$$\Gamma \vdash \mathit{fold} : \Pi A : X \to \mathsf{U}.\ (FA \Rightarrow A) \to (\mu F \Rightarrow A)$$

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

$$\frac{\begin{array}{c}\Gamma \vdash A : X \to \mathsf{U}\\ \Gamma \vdash k^A : FA \Rightarrow A \qquad \Gamma \vdash x : X \qquad \Gamma \vdash M : \mathsf{T}(\mu F\ x)\end{array}}{\begin{array}{l}\Gamma \vdash \mathit{fold}\ A\ k^A\ x\ (\mathrm{in}\ x\ M) =\\ \qquad k^A\ x\ (\mathit{Fmap}\ \mu F\ A\ (\mathit{fold}\ A\ k^A)\ x\ M : \mathsf{T}(A\ x)\end{array}}$$

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

$$\frac{\begin{array}{c}\Gamma \vdash A : X \to \mathsf{U} \qquad \Gamma \vdash k^A : FA \Rightarrow A\\ \Gamma \vdash h : \mu F \Rightarrow A \qquad h \text{ is an } F\text{-algebra morphism}\end{array}}{\Gamma \vdash h = \mathit{fold}\ A\ k^A : \mu F \Rightarrow A}$$

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* $\gamma_o \in \Gamma_O$,

$$\begin{aligned}&(k^A_o(\gamma_o), k^B_o(\gamma_o)) \in\\ &\mathrm{morph}_R(\Gamma_{\mathit{refl}}(\gamma_o), F_r(\Gamma_{\mathit{refl}}(\gamma_o), \langle h \rangle(\Gamma_{\mathit{refl}}(\gamma_o))), \langle h \rangle(\Gamma_{\mathit{refl}}(\gamma_o)))\end{aligned}$$

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, Fmap), we make the following definitions:

$$\mu F = \lambda x : X.\, \Pi A : X \to \mathsf{U}.\, (FA \Rightarrow A) \to \mathsf{T}(A\, x)$$

$$\mathit{fold} = \lambda A{:}X \to \mathsf{U}.\, \lambda k^A{:}(FA \Rightarrow A).\, \lambda x{:}X.\, \lambda e{:}\mathsf{T}(\mu F\, x).\, e\ A\ k^A$$

$$\begin{aligned}\mathrm{in} = \lambda x : X.\, \lambda e : \mathsf{T}(F(\mu F)x).\, \lambda A : X \to \mathsf{U}.\, \lambda k^A : (FA \Rightarrow A).\\ k^A\ x\ (\mathit{Fmap}\ \mu F\ A\ (\mathit{fold}\ A\ k^A)\ x\ e)\end{aligned}$$

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-kinded initial algebras in a relationally parametric model of System Fω.

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:*

$$\frac{\begin{array}{c}\Gamma \vdash A : X \to \mathsf{U} \qquad \Gamma \vdash k^A : FA \Rightarrow A\\ \Gamma \vdash B : X \to \mathsf{U} \qquad \Gamma \vdash k^B : FB \Rightarrow B\\ \Gamma \vdash h : A \Rightarrow B \qquad h \textit{ is an } F\textit{-algebra homomorphism}\\ \Gamma \vdash x : X \qquad \Gamma \vdash M : \mathsf{T}(\mu F\ x)\end{array}}{\Gamma \vdash h\ x\ (M\ A\ k^A) = M\ B\ k^B : \mathsf{T}(B\ x)}$$

Proof. By Lemma 2 we need only prove the object-level part of the equation. Given $\gamma_o \in \Gamma_O$, we instantiate M_r with $\Gamma_{\mathit{refl}}(\gamma_o)$, the graph relation $\langle h\rangle(\Gamma_{\mathit{refl}}(\gamma_o))$, and the pair $(k_o^A(\gamma_o), k_o^B(\gamma_o))$, by Lemma 6, to obtain

$$\begin{aligned}(M_o(\gamma_o, A_o(\gamma_o), k_o^A(\gamma_o)), M_o(\gamma_o, B_o(\gamma_o), k_o^B(\gamma_o))) \in\\ \langle h\rangle(\Gamma_{\mathit{refl}}(\gamma_o))(X_{\mathit{refl}}(\gamma_o)(x_o(\gamma_o)))\end{aligned}$$

Unfolding the graph relation, this statement is equivalent to:

$$\begin{aligned}&(h_o(\gamma_o, x_o(\gamma_o), M_o(\gamma_o, A_o(\gamma_o), k_o^A(\gamma_o))), M_o(\gamma_o, B_o(\gamma_o), k_o^B(\gamma_o)))\\ &\in B_r(\Gamma_{\mathit{refl}}(\gamma_o), X_{\mathit{refl}}(\gamma_o)(x_o(\gamma_o)))\end{aligned}$$

Now, by the identity extension property that is built into the reflexive graph model, the relation $B_r(\Gamma_{\mathit{refl}}(\gamma_o), X_{\mathit{refl}}(\gamma_o)(x_o(\gamma_o)))$ is equal to $U_{\mathit{refl}}(\gamma_o, x_o(\gamma_o))(B_o(\gamma_o, x_o(\gamma_o)))$. By the discreteness of elements of the universe U, this relation is, up to isomorphism, the equality relation on the small set $B_o(\gamma_o, x_o(\gamma_o))$. We thus have

$$\begin{aligned}&h_o(\gamma_o, x_o(\gamma_o), M_o(\gamma_o, A_o(\gamma_o), k_o^A(\gamma_o)))\\ =\ &M_o(\gamma_o, B_o(\gamma_o), k_o^B(\gamma_o)) \quad \square\end{aligned}$$

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

Theorem 3. *Any X-indexed functor (F, 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 β-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 $h\ x\ (M\ \mu F\ \mathrm{in}) = M\ A\ k^A$. Using Lemma 7 again, this time with an arbitrary F-algebra (B, k^B), and setting $h = \mathrm{fold}\ B\ k^B$ yields $M\ \mu F\ \mathrm{in}\ B\ k^B = M\ B\ k^B$. Extensionality therefore gives $M\ \mu F\ \mathrm{in} = M$, and thus $h\ x\ M = M\ A\ k^A$, as required. $\square$

Despite the considerably more general setting, the above proof is almost identical to that given by Atkey for higher-kinded initial algebras in a relationally parametric model of System Fω. 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 $\mathrm{Set}^{\mathrm{RG}}$, so our parametric model of dependent types can be seen as moving from a fibration $p : \mathrm{Fam(Set)} \to \mathrm{Set}$ to a fibration $p^{\mathrm{RG}} : \mathrm{Fam(Set)}^{\mathrm{RG}} \to \mathrm{Set}^{\mathrm{RG}}$. 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 ∞-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

[1] R. Atkey. A Deep Embedding of Parametric Polymorphism in Coq. *Proc., Workshop on Mechanising Metatheory*, 2009.

[2] R. Atkey. Relational Parametricity for Higher Kinds. *Proc., Computer Science Logic*, pp. 46–61, 2012.

[3] R. Atkey, P. Johann, and A. Kennedy. Abstraction and Invariance for Algebraically Indexed Types. *Proc., Principles of Programming Languages*, pp. 87–100, 2013.

[4] E. S. Bainbridge, P. J. Freyd, A. Scedrov, and P. J. Scott. Functorial Polymorphism. *Theoretical Computer Science* 70(1), pp. 35–64, 1990.

[5] J.-P. Bernardy, P. Jansson, and R. Paterson. Proofs for Free - Parametricity for Dependent Types. *Journal of Functional Programming* 22(2), pp. 107–152, 2012.

[6] A. Bove, P. Dybjer, and U. Norell. A Brief Overview of Agda – A Functional Language with Dependent Types. *Proc., Theorem Proving in Higher Order Logics*, 2009.

[7] J. Cheney and R. Hinze. A lightweight implementation of generics and dynamics. *Proc., Workshop on Haskell*, pp. 90–104, 2002.

[8] A. Chlipala. Parametric higher-order abstract syntax for mechanized semantics. *Proc., International Conference on Functional Programming*, pp. 143–156, 2008.

[9] The Coq Development Team. The Coq proof assistant reference manual. *LogiCal Project*, Version 8.0, 2004.

[10] T. Coquand and G. P. Huet. The Calculus of Constructions. *Information and Computation* 76(2/3), pp. 95–120, 1988.

[11] B. Dunphy and U. S. Reddy. Parametric Limits. *Proc., Logic in Computer Science*, pp. 242–251, 2004.

[12] P. Dybjer. Internal Type Theory. *Proc., Types for Proofs and Programs*, pp. 120–134, 1996.

[13] R. Hasegawa. Categorical Data Types in Parametric Polymorphism. *Mathematical Structures in Computer Science* 4(1), pp. 71–109, 1994.

[14] R. Hasegawa. Relational Limits in General Polymorphism. *Publications of the Research Institute for Mathematical Sciences* 30, pp. 535–576, 1994.

[15] M. Hofmann. Syntax and Semantics of Dependent Types. In *Semantics and Logics of Computation*, Cambridge University Press, pp. 79–130, 1997.

[16] M. Hofmann and T. Streicher. Lifting Grothendieck Universes. Unpublished manuscript. 199?

[17] B. Jacobs. *Categorical Logic and Type Theory*. Elsevier, 1999.

[18] P. Johann. Short Cut Fusion: Proved and Improved. *Proc., Semantics, Application, and Implementation of Program Generation*, pp. 47–71, 2001.

[19] N. Krishnaswami and D. Dreyer. Internalizing Relational Parametricity in the Extensional Calculus of Constructions. *Proc., Computer Science Logic*, 2013.

[20] S. Mac Lane. *Categories for the Working Mathematician, Second Edition.* Springer-Verlag, 1998.

[21] H. G. Mairson. Outline of a Proof Theory of Parametricity. *Proc., Functional Programming Languages and Computer Architecture*, pp. 313–327, 1991.

[22] P. Martin-Löf. *Intuitionistic Type Theory*. Bibliopolis, 1984.

[23] A. Nanevski, A. Banerjee, D. Garg. Dependent Type Theory for Verification of Information Flow and Access Control Policies. *ACM Transactions on Programming Languages and Systems* 35(2), pp. 6:1–6:41, 2013.

[24] A. M. Pitts. Parametric Polymorphism and Operational Equivalence. *Mathematical Structures in Computer Science* 10, pp. 321–359, 2000.

[25] G. D. Plotkin. Lambda Definability and Logical Relations. Technical Report, University of Edinburgh, 1973.

[26] G. D. Plotkin and M. Abadi. A Logic for Parametric Polymorphism. *Proc., Typed Lambda Calculi and Applications*, pp. 361–375, 1993.

[27] J. C. Reynolds. Polymorphism is Not Set-Theoretic. *Proc., Semantics of Data Types*, pp. 145–156, 1984.

[28] J. C. Reynolds. Types, Abstraction and Parametric Polymorphism. *Information Processing* 83, pp. 513–523, 1983.

[29] E. Robinson. Parametricity as Isomorphism. *Theoretical Computer Science* 136, pp. 163–181, 1994.

[30] E. Robinson and G. Rosolini. Reflexive Graphs and Parametric Polymorphism. *Proc., Logic in Computer Science*, pp. 364–371, 1994.

[31] I. Takeuti. The Theory of Parametricity in the Lambda Cube. Technical Report 1217, Kyoto University, 2001.

[32] The Univalent Foundations Program. *Homotopy Type Theory*. Institute for Advanced Study, 2013.

[33] D. Vytiniotis and S. Weirich Parametricity, Type Equality, and Higher-Order Polymorphism. *Journal of Functional Programming* 20(2), pp. 175–201, 2010.

[34] P. Wadler. The Girard-Reynolds Isomorphism (second edition). *Theoretical Computer Science* 375(1-3), pp. 201–226, 2007.

[35] P. Wadler. Theorems for Free! *Proc., Functional Programming Languages and Computer Architecture*, pp. 347–359, 1989.