Free Theorems for Nested Types

ANONYMOUS AUTHOR(S)

Abstract goes here

1 INTRODUCTION

Suppose we wanted to prove some property of programs over an algebraic data type (ADT) such as that of lists, coded in Agda as

```
data List (A : Set) : Set where
  nil  : List A
  Cons : A → List A → List A
```

A natural approach to the problem uses structural induction on the input data structure in question. This requires knowing not just the definition of the ADT of which the input data structure is an instance, but also the program text for the functions involved in the properties to be proved. For example, to prove by induction that mapping a polymorphic function over a list and then reversing the resulting list is the same as reversing the original list and then mapping the function over the result, we unwind the (recursive) definitions of the `reverse` and `map` functions over lists to according to the inductive structure of the input list. Such data-driven induction proofs over ADTs are so routine that they are often included in, say, undergraduate functional programming courses.

An alternative technique for proving results like the above `map-reverse` property for lists is to use parametricity, a formalization of extensional type-uniformity in polymorphic languages. Parametricity captures the intuition that a polymorphic program must act uniformly on all of its possible type instantiations; it is formalized as the requirement that every polymorphic program preserves all relations between any pair of types that it is instantiated with. Parametricity was originally put forth by Reynolds [Reynolds 1983] for System F [Girard et al. 1989], the formal calculus at the core of all polymorphic functional languages. It was later popularized as Wadler's "theorems for free" [Walder 1989] because it allows the deduction of many properties of programs in such languages solely from their types, i.e., with no knowledge whatsoever of the text of the programs involved. To get interesting free theorems, Wadler's calculus included, implicitly, built-in list types; indeed, most of the free theorems in [Walder 1989] are consequences of naturality for polymorphic list-processing functions. However, parametricity can also be used to prove naturality properties for non-list ADTs, as well as results, like correctness of program optimizations like *short cut fusion* [Gill et al. 1993; Johann 2002, 2003], that go beyond simple naturality.

This paper is about parametricity and free theorems for a polymorphic calculus with explicit syntax not just for ADTs, but for nested types as well. An ADT defines a *family of inductive data types*, one for each input type. For example, the `List` data type definition above defines a collection of data types `List A`, `List B`, `List (A × B)`, `List (List A)`, etc., each independent of all the others. By contrast, a nested type [Bird and Meertens 1998] is an *inductive family of data types* that is defined over, or is defined mutually recursively with, (other) such data types. Since the structures of the data type at one type can depend on those at other types, the entire family of types must be defined at once. Examples of nested types include, trivially, ordinary ADTs, such as list and tree types; simple nested types, such as the data type

```
data PTree (A : Set) : Set where
  pleaf : A → PTree A
  pnode : PTree (A × A) → PTree A
```

2020. 2475-1421/2020/1-ART1 $15.00
https://doi.org/

```
reversePTree : ∀{A : Set} → PTree A → PTree A
reversePTree {A} = foldPTree {A} {PTree}
                     pleaf
                     (λp → pnode (mapPTree swap p))

foldPTree : ∀{A : Set} → {F : Set → Set} →
            ({B : Set} → B → FB) →
            ({B : Set} → F(B × B) → FB) →
            PTree A → F A
foldPTree n c (pleaf x) = n x
foldPTree n c (pnode p) = c (foldPTree n c p)

mapPTree : ∀{A B : Set} → (A → B) → PLeaves A → PLeaves B
mapPTree f (pleaf x) = pleaf (f x)
mapPTree f (pnode p) = pnode (mapPTree (λp → (f(π₁ p), f(π₂ p))) p)

swap : ∀{A : Set} → (A × A) → (A × A)
swap (x, y) = (y, x)
```

Fig. 1. reversePTree and auxiliary functions in Agda

```
reverseBush : ∀{A : Set} → Bush A → Bush A
reverseBush {A} = foldBush {A} {Bush} bnil balg

foldBush : ∀{A : Set} → {F : Set → Set} →
        ({B : Set} → FB) →
        ({B : Set} → B → F (F B) → F B) →
        Bush A → F A
foldBush bn bc bnil = bn
foldBush bn bc (bcons x bb) =
        bc x (foldBush bn bc (mapBush (foldBush bn bc) bb))

mapBush : ∀{A B : Set} → (A → B) → (Bush A) → (Bush B)
mapBush _ bnil = bnil
mapBush f (bcons x bb) = bcons (f x) (mapBush (mapBush f) bb)

balg : ∀{B : Set} → B → Bush (Bush B) → Bush B
balg x bnil = bcons x bnil
balg x (bcons bnil bbbx) = bcons x (bcons bnil bbbx)
balg x (bcons (bcons y bx) bbbx) =
        bcons y (bcons (bcons x bx) bbbx)
```

Fig. 2. reverseBush and auxiliary functions in Agda

of perfect trees, whose recursive occurrences never appear below other type constructors; "deep" nested types [Johann and Polonsky 2020], such as the data type

```
data Forest (A : Set) : Set where
       fempty : Forest A
       fnode : A → PTree (Forest A) → Forest A
```

of perfect forests, whose recursive occurrences appear below type constructors for other nested types; and truly nested types[1], such as the data type

```
data Bush (A : Set) : Set where
       bnil : Bush A
       bcons : A → Bush (Bush A) → Bush A
```

of bushes (also called *bootstrapped heaps* in [Okasaki 1999]), whose recursive occurrences appear below their own type constructors.

Suppose we now want to prove properties of functions over nested types. We might, for example, want to prove a `map-reverse` property for the functions on perfect trees in Figure 1, or for those on bushes[2] in Figure 2. A few well-chosen examples quickly convince us that such a property should indeed hold for perfect trees, and, drawing inspiration from the situation for ADTs, we easily construct a proof by induction on the input perfect tree. To formally establish this result, we could even prove it in Coq or Agda: each of these provers actually generates an induction rule for perfect trees and the generated rule gives the expected result because proving properties of perfect trees requires only that we induct over the top-level perfect tree in the recursive position, leaving any data internal to the input tree untouched.

Unfortunately, it is nowhere near as clear that analogous intuitive or formal inductive arguments can be made for the `map-reverse` property for bushes. Indeed, a proof by induction on the input bush must recursively induct over the bushes that are internal to the top-level bush in the recursive position. This is sufficiently delicate that no induction rule for bushes or other truly nested types was known until very recently, when *deep induction* [Johann and Polonsky 2020] was developed as a way to induct over *all* of the structured data present in an input. Deep induction thus not only gave the first principled and practically useful structural induction rules for bushes and other truly

[1]Nested types that are defined over themselves are known as *truly nested types*.
[2]To define the `foldBush` and `mapBush` functions in Figure 2 it is necessary to turn off Agda's termination checker.

nested types, and has also opened the way for incorporating automatic generation of such rules for (truly) nested data types — and, eventually, even GADTs — into modern proof assistants.

Of course it is great to know that we *can*, at last, prove properties of programs over (truly) nested types by induction. But recalling that inductive proofs over ADTs can sometimes be circumvented in the presence of parametricity, we might naturally ask:

Can we derive properties of functions over (truly) nested types from parametricity?

This paper answers the above question in the affirmative by constructing a parametric model for a polymorphic calculus providing primitives for *constructing* nested types directly via recursion — rather than representing them indirectly by Church encodings as in most polymorphic calculi.

We introduce our calculus in Section 2. At the type level, it is the level-2-truncation of the higher-kinded calculus from [Johann and Polonsky 2019], augmented with a primitive type of natural transformations. To construct nested types, it constructs type expressions not just from standard type variables, but also from type constructor variables of various arities, and includes an explicit μ-construct for type-level recursion with respect to these variables. The class of nested types thus constructed is very robust and includes all (truly) nested types known from the literature. In Section 3 we give set and relational interpretations for the types From Section 2. As is usual when modeling parametricity, types are interpreted as functors from environments interpreting their type variable contexts to set or relations, as appropriate. But in order to ensure that these functors satisfy the cocontinuity properties needed to know that the fixpoints interpreting μ-types exist, set environments must map each k-ary type constructor variable to an appropriately cocontinuous k-ary functor on sets and relation environments must map each k-ary type constructor variable to an appropriately cocontinuous k-ary relation transformer, and these cocontinuity conditions must be threaded throughout the type interpretations in such a way that the resulting model is guaranteed to satisfy an appropriate Identity Extension Lemma (Theorem 22). Properly progagating the cocontinuity conditions turns out to be both subtle and challenging, and Section 4, where it is done, is where the bulk of the work in constructing our model lies. At the term level, our calculus includes primitive constructs for the actions on morphisms of the functors interpreting types, initial algebras for fixpoints of these functors, and structured recursion over elements of these initial algebras (i.e., map, in, and fold constructs, respectively). While our calculus does not support general recursion at the term level, it is strongly normalizing, so does perhaps edge us toward the kind of provably total practical programming language proposed at the end of [Walder 1989]. In Section 5, we give set and relational interpretations for the terms of our calculus. As usual in parametric models, terms are interpreted as natural transformations from interpretations of the term contexts in which they are formed to the interpretations of their types, and these must cohere in what is essentially a fibrational way. Immediately from the definitions of our interpretations we prove in Section **??** a scheme deriving free theorems that are consequences of naturality of functions that are polymorphic over nested types. This scheme is very general, is parameterized over both the data type and the polymorphic function at hand, and has each of the above `map-reverse` theorems as instances. The relationship between naturality and parametricity has long been of interest, and our inclusion of a primitive type of natural transformations makes it possible to clearly delineate those free theorems that are consequences of naturality, and thus would hold even in non-parametric models of polymorphic calculi, from those that use the full power of parametricity to go beyond naturality. In Section 5.4 we prove that our model satisfies an Abstraction Theorem (Theorem 28), and we derive several of this latter kind of free theorem from it in Section 6. Specifically, we state and prove (non-)inhabitation results in Sections 6.1 and 6.2, a free theorem for the type of a filter function on generalized rose trees in Section 6.3, the correctness of short cut fusion for lists in Section 6.4, and its generaliation to nested types in Section 6.5.

There is a long line of work on categorical models of parametricity for System F; see, for example, [Bainbridge et al. 1990; Birkedal and Møgelberg 2005; Dunphy and Reddy 2004; Ghani et al. 2015; Hasegawa 1994; Jacobs 1999; Ma and Reynolds 1992; Robinson and Rosolini 1994]. To our knowledge, all categorical models that treat (algebraic) data types do so via their Church encodings, verifying in the just-constructed parametric model that the Church encoding of each such type is interpreted as the least fixpoint of the (first-order) functor interpreting the type constructor from which its Church encoding was constructed. The present paper draws on this rich tradition of categorical models of parametricity for System F, but treats nested types as well as ADTs, and is the first to treat data types by direct construction via primitives rather than by Church encodings. This requires that we modify the type calculus to ensure that functoriality is guaranteed syntactically, and that functoriality is reflected in the standard model construction so that the existence of the fixpoints by which nested types are to be interpreted is ensured.

Like us, Pitts [Pitts 1998, 2000] extends parametricity from pure System F to System F augmented with primitives for constructing data types directly. Only list types are added in [Pitts 2000], but other polynomial ADTs are easily included as in [Pitts 1998]. Part of Pitts' motivation is to show that ADTs and their Church encodings have the same operational behavior in his system. We cannot even ask this question about our system, which cannot express Church encodings of even simple ADTs such as list, or pair, or sum types, since these are not functorial; nevertheless, we do prove the correctness of short cut fusion for nested types, whose operational analogue is the means by which Pitts proves his equivalence result. It would be particularly interesting to know what operational analogues of functoriality and cocontinuity are needed to extend Pitts' parametricity results from a calculus with primitives for constructing data types modeled as fixpoints of first-order functors to one that also provides such primitives for data types modeled as fixpoints of higher-order functors.

We are not the first to consider parametricity at higher kinds. Atkey [Atkey 2012] constructs a parametric model for full System Fω, but within the impredicative Calculus of Inductive Constructions (iCIC) rather than in a semantic category. Atkey's construction is similar to ours in some ways, but he represents data types using Church encodings rather than constructing them via primitives. Since his model is entirely syntactic, his syntactic "functors", whose associated *fmap* functions representing their functorial actions must be *given* together with their underlying type constructors, cannot be interpreted as semantic functors. The associated Church encodings therefore cannot be interpreted as their fixpoints, so Atkey need not, and does not, impose cocontinuity conditions on his model to ensure that these fixpoints exist. Nevertheless, he does verify the existence of initial algebras for his "functors" in iCIC. Atkey does not indicate which type constructors support the kinds of *fmap* functions needed to turn them into "functors", but we suspect spelling this out would result in a full higher-kinded extension of the calculus we present here.

2 THE CALCULUS

2.1 Types

For each $k \geq 0$, we assume countable sets $\mathbb{T}^k$ of *type constructor variables of arity* k and $\mathbb{F}^k$ of *functorial variables of arity* k, all mutually disjoint. The sets of all type constructor variables and functorial variables are $\mathbb{T} = \bigcup_{k\geq 0} \mathbb{T}^k$ and $\mathbb{F} = \bigcup_{k\geq 0} \mathbb{F}^k$, respectively, and a *type variable* is any element of $\mathbb{T} \cup \mathbb{F}$. We use lower case Greek letters for type variables, writing ϕ^k to indicate that $\phi \in \mathbb{T}^k \cup \mathbb{F}^k$, and omitting the arity indicator k when convenient, unimportant, or clear from context. We reserve letters from the beginning of the alphabet to denote type variables of arity 0, i.e., elements of $\mathbb{T}^0 \cup \mathbb{F}^0$. We write $\overline{\zeta}$ for either a set $\{\zeta_1, ..., \zeta_n\}$ of type constructor variables or a set of functorial variables when the cardinality n of the set is unimportant or clear from context. If

P is a set of type variables we write $P, \overline{\phi}$ for $P \cup \overline{\phi}$ when $P \cap \overline{\phi} = \emptyset$. We omit the vector notation for a singleton set, thus writing ϕ, instead of $\overline{\phi}$, for $\{\phi\}$.

Definition 1. *Let V be a finite subset of $\mathbb{T}$, let P be a finite subset of $\mathbb{F}$, let $\overline{\alpha}$ be a finite subset of $\mathbb{F}^0$ disjoint from P, and let $\phi^k \in \mathbb{F}^k \setminus P$. The sets $\mathcal{T}(V)$ of* type constructor expressions *over V and $\mathcal{F}^P(V)$ of* functorial expressions *over P and V are given by*

$$\mathcal{T}(V) \ ::= \ V \mid \mathsf{Nat}^{\overline{\alpha}}\, \mathcal{F}^{\overline{\alpha}}(V)\, \mathcal{F}^{\overline{\alpha}}(V) \mid V\overline{\mathcal{T}(V)}$$

and

$$\begin{aligned}\mathcal{F}^P(V) \ ::= \ & \mathcal{T}(V) \mid \mathbb{0} \mid \mathbb{1} \mid P\overline{\mathcal{F}^P(V)} \mid V\,\overline{\mathcal{F}^P(V)} \mid \mathcal{F}^P(V) + \mathcal{F}^P(V) \mid \mathcal{F}^P(V) \times \mathcal{F}^P(V) \\ & \mid \left(\mu\phi^{\,k}.\lambda\alpha_1 \ldots \alpha_k.\mathcal{F}^{P,\alpha_1,\ldots,\alpha_k,\phi}(V)\right)\overline{\mathcal{F}^P(V)}\end{aligned}$$

A *type* over P and V is any element of $\mathcal{T}(V) \cup \mathcal{F}^P(V)$.

The notation for types entails that an application $\tau\tau_1 \ldots \tau_k$ is allowed only when τ is a type variable of arity k, or τ is a subexpression of the form $\mu\phi^k.\lambda\alpha_1 \ldots \alpha_k.\tau'$. Moreover, if τ has arity k then τ must be applied to exactly k arguments. Accordingly, an overbar indicates a sequence of subexpressions whose length matches the arity of the type applied to it. The fact that types are always in *η-long normal form* avoids having to consider β-conversion at the level of types. In a subexpression $\mathsf{Nat}^{\overline{\alpha}}\sigma\,\tau$, the Nat operator binds all occurrences of the variables in $\overline{\alpha}$ in σ and τ. Similarly, in a subexpression $\mu\phi^k.\lambda\overline{\alpha}.\tau$, the μ operator binds all occurrences of the variable ϕ, and the λ operator binds all occurrences of the variables in $\overline{\alpha}$, in the body τ.

A *type constructor context* is a finite set Γ of type constructor variables, and a *functorial context* is a finite set Φ of functorial variables. In Definition 2, a judgment of the form $\Gamma; \Phi \vdash \tau : \mathcal{T}$ or $\Gamma; \Phi \vdash \tau : \mathcal{F}$ indicates that the type τ is intended to be functorial in the variables in Φ but not necessarily in the variables in Γ.

Definition 2. *The formation rules for the set $\mathcal{T} \subseteq \bigcup_{V \subseteq \mathbb{T}} \mathcal{T}(V)$ of* well-formed type constructor expressions *are*

$$\frac{}{\Gamma, \alpha^0; \emptyset \vdash \alpha^0 : \mathcal{T}}$$

$$\frac{\Gamma; \overline{\alpha^0} \vdash \sigma : \mathcal{F} \qquad \Gamma; \overline{\alpha^0} \vdash \tau : \mathcal{F}}{\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha^0}}\sigma\,\tau : \mathcal{T}}$$

The formation rules for the set $\mathcal{F} \subseteq \bigcup_{V \subseteq \mathbb{T}, P \subseteq \mathbb{F}} \mathcal{F}^P(V)$ of well-formed functorial expressions *are*

$$\frac{\Gamma; \emptyset \vdash \tau : \mathcal{T}}{\Gamma; \emptyset \vdash \tau : \mathcal{F}} \qquad \frac{}{\Gamma; \Phi, \alpha^0 \vdash \alpha^0 : \mathcal{F}} \qquad \frac{}{\Gamma; \Phi \vdash \mathbb{0} : \mathcal{F}} \qquad \frac{}{\Gamma; \Phi \vdash \mathbb{1} : \mathcal{F}}$$

$$\frac{\phi^k \in \Gamma \cup \Phi \qquad \overline{\Gamma; \Phi \vdash \tau : \mathcal{F}}}{\Gamma; \Phi \vdash \phi^k\overline{\tau} : \mathcal{F}}$$

$$\frac{\Gamma; \Phi, \overline{\alpha}, \phi^k \vdash \tau : \mathcal{F} \qquad \overline{\Gamma; \Phi \vdash \tau : \mathcal{F}}}{\Gamma; \Phi \vdash (\mu\phi^k.\lambda\overline{\alpha}.\,\tau)\overline{\tau} : \mathcal{F}}$$

$$\frac{\Gamma; \Phi \vdash \sigma : \mathcal{F} \qquad \Gamma; \Phi \vdash \tau : \mathcal{F}}{\Gamma; \Phi \vdash \sigma + \tau : \mathcal{F}} \qquad \frac{\Gamma; \Phi \vdash \sigma : \mathcal{F} \qquad \Gamma; \Phi \vdash \tau : \mathcal{F}}{\Gamma; \Phi \vdash \sigma \times \tau : \mathcal{F}}$$

A type τ is well-formed *if it is either a well-formed type constructor expression or a well-formed functorial expression.*

If τ is a closed type we may write $\vdash \tau$, rather than $\emptyset;\emptyset \vdash \tau$, for the judgment that it is well-formed. Definition 2 ensures that the expected weakening rules for well-formed types hold — although weakening does not change the contexts in which Nat-types can be formed. If $\Gamma;\emptyset \vdash \sigma : \mathcal{T}$ and $\Gamma;\emptyset \vdash \tau : \mathcal{T}$, then our rules allow formation of the type $\Gamma;\emptyset \vdash \mathrm{Nat}^{\emptyset}\sigma\,\tau$. Since a type $\Gamma;\emptyset \vdash \mathrm{Nat}^{\overline{\alpha}}\sigma\,\tau$ represents a natural transformation in $\overline{\alpha}$ from σ to τ, the type $\Gamma;\emptyset \vdash \mathrm{Nat}^{\emptyset}\sigma\,\tau$ represents the standard arrow type $\Gamma \vdash \sigma \to \tau$ in our calculus. We similarly represent a standard $\forall$-type $\Gamma;\emptyset \vdash \forall\overline{\alpha}.\tau$ as $\Gamma;\emptyset \vdash \mathrm{Nat}^{\overline{\alpha}}\,\mathbb{1}\,\tau : \mathcal{F}$ in our calculus. However, if $\overline{\alpha}$ is non-empty then τ cannot be of the form $\mathrm{Nat}^{\overline{\beta}}H\,K$ since $\Gamma;\overline{\alpha} \vdash \mathrm{Nat}^{\overline{\beta}}H\,K$ is not a valid type judgment in our calculus (except by weakening).

Definition 2 allows the formation of all of the (closed) nested types from the introduction:

$$\begin{array}{lcl} \mathit{List}\,\alpha & = & \mu\beta.\,\mathbb{1} + \alpha \times \beta \;=\; (\mu\phi.\lambda\beta.\,\mathbb{1} + \beta \times \phi\beta)\,\alpha \\ \mathit{PTree}\,\alpha & = & (\mu\phi.\lambda\beta.\,\beta + \phi\,(\beta \times \beta))\,\alpha \\ \mathit{Forest}\,\alpha & = & (\mu\phi.\lambda\beta.\,\mathbb{1} + \beta \times \mathit{PTree}\,(\phi\beta))\,\alpha \\ \mathit{Bush}\,\alpha & = & (\mu\phi.\lambda\beta.\,\mathbb{1} + \beta \times \phi\,(\phi\beta))\,\alpha \end{array}$$

Each of these types can be considered either functorial in α or not, according to whether $\alpha \in \Gamma$ or $\alpha \in \Phi$. For example, if $\emptyset;\alpha \vdash \mathit{List}\,\alpha$, then the type $\vdash \mathrm{Nat}^{\alpha}\mathbb{1}\,(\mathit{List}\,\alpha) : \mathcal{T}$ is well-formed; if $\alpha;\emptyset \vdash \mathrm{List}\ \alpha$, then it is not. If $\mathit{Tree}\,\alpha\,\gamma = \mu\,\beta.\,\alpha + \beta \times \gamma \times \beta$, then Definition 2 also allows the derivation of, e.g., the type $\gamma;\emptyset \vdash \mathrm{Nat}^{\alpha}(\mathit{List}\,\alpha)\,(\mathit{Tree}\,\alpha\,\gamma)$ representing a natural transformation from lists to trees that is natural in α but not necessarily in γ. We emphasize that types can be functorial in variables of arity greater than 0. For example, the type $\mathit{GRose}\,\phi\,\alpha = \mu\beta.\alpha \times \phi\beta$ can be functorial in ϕ if $\phi \in \Phi$. As usual, whether $\phi \in \Gamma$ or $\phi \in \Phi$ determines whether types such as $\mathrm{Nat}^{\alpha}(\mathit{GRose}\,\phi\,\alpha)\,(\mathit{List}\,\alpha)$ are well-formed. But even if *GRose* is functorial in ϕ, it still cannot be the (co)domain of a Nat type representing a natural transformation in ϕ. This is because our calculus does not allow naturality in variables of arity greater than 0.

Definition 2 explicitly considers types in $\mathcal{T}$ to be types in $\mathcal{F}$ that are functorial in no variables. This allows the formation of types such as $\mathit{List}\,(\sigma \to \tau)$ and $\mathit{PTree}\,(\forall\alpha.\tau)$. Functorial variables in a well-formed type τ can also be demoted to non-functorial status. The proof is by induction on τ.

Lemma 3. *If $\Gamma;\Phi,\phi^k \vdash \tau : \mathcal{F}$, then $\Gamma,\psi^k;\Phi \vdash \tau[\phi^k :== \psi^k] : \mathcal{F}$ is also derivable. Here, $\tau[\phi :== \psi]$ is the textual replacement of ϕ in τ, meaning that all occurences of $\phi\overline{\sigma}$ in τ become $\psi\overline{\sigma}$.*

In addition to textual replacement, we also have a proper substitution operation on types. If τ is a type over P and V, if P and V contain only type variables of arity 0, and if $k = 0$ for every occurrence of ϕ^k bound by μ in τ, then we say that τ is *first-order*; otherwise we say that τ is *second-order*. Substitution for first-order types is the usual capture-avoiding textual substitution. We write $\tau[\alpha := \sigma]$ for the result of substituting σ for α in τ, and $\tau[\alpha_1 := \tau_1, \ldots, \alpha_k := \tau_k]$, or $\tau[\overline{\alpha := \tau}]$ when convenient, for $\tau[\alpha_1 := \tau_1][\alpha_2 := \tau_2, \ldots, \alpha_k := \tau_k]$. Substitution for second-order types is defined below, where we adopt a similar notational convention for vectors of types.

Definition 4. *Def 4 changed again; not correct to substitute along non-functorial variables. If $\phi^k \in \Gamma \cup \Phi$ with $k \geq 1$, if $\Gamma;\Phi \vdash F : \mathcal{F}$, and if $\Gamma;\Phi,\overline{\alpha} \vdash H : \mathcal{F}$ with $|\overline{\alpha}| = k$, then $\Gamma \setminus \phi^k;\Phi \setminus \phi^k \vdash$*

$F[\phi :=_{\overline{\alpha}} H] : \mathcal{F}$, *where the operation* $(\cdot)[\phi := H]$ *of* second-order type substitution *is defined by:*

$$
\begin{array}{lcl}
(\mathsf{Nat}^{\overline{\gamma}}\, G\, K)[\phi :=_{\overline{\alpha}} H] & = & \mathsf{Nat}^{\overline{\gamma}}\,(G[\phi :=_{\overline{\alpha}} H])\,(K[\phi :=_{\overline{\alpha}} H]) \\
\mathbb{1}[\phi :=_{\overline{\alpha}} H] & = & \mathbb{1} \\
\mathbb{0}[\phi :=_{\overline{\alpha}} H] & = & \mathbb{0} \\
(\psi\,\overline{\tau})[\phi :=_{\overline{\alpha}} H] & = & \begin{cases} \psi\,\overline{\tau[\phi :=_{\overline{\alpha}} H]} & \text{if } \psi \neq \phi \\ H[\alpha := \tau[\phi :=_{\overline{\alpha}} H]] & \text{if } \psi = \phi \end{cases} \\
(\sigma + \tau)[\phi :=_{\overline{\alpha}} H] & = & \sigma[\phi :=_{\overline{\alpha}} H] + \tau[\phi :=_{\overline{\alpha}} H] \\
(\sigma \times \tau)[\phi :=_{\overline{\alpha}} H] & = & \sigma[\phi :=_{\overline{\alpha}} H] \times \tau[\phi :=_{\overline{\alpha}} H] \\
((\mu\psi.\lambda\overline{\gamma}.\, G)\,\overline{\tau})[\phi :=_{\overline{\alpha}} H] & = & (\mu\psi.\lambda\overline{\gamma}.\, G[\phi :=_{\overline{\alpha}} H])\,\overline{\tau[\phi :=_{\overline{\alpha}} H]}
\end{array}
$$

We omit the variable subscripts in second-order type constructor substitution when convenient.

2.2 Terms

We assume an infinite set $\mathcal{V}$ of term variables disjoint from $\mathbb{T}$ and $\mathbb{F}$. If Γ be a type constructor context and Φ is a functorial context, then a *term context for* Γ *and* Φ is a finite set of bindings of the form $x : \tau$, where $x \in \mathcal{V}$ and $\Gamma; \Phi \vdash \tau : \mathcal{F}$. We adopt the same conventions for denoting disjoint unions and for vectors in term contexts as for type constructor contexts and functorial contexts.

Definition 5. *Let* Δ *be a term context for* Γ *and* Φ. *The formation rules for the set of* well-formed terms over Δ *are*

$$\frac{\Gamma; \emptyset \vdash \tau : \mathcal{T}}{\Gamma; \emptyset \mid \Delta, x : \tau \vdash x : \tau} \qquad \frac{\Gamma; \Phi \vdash \tau : \mathcal{F}}{\Gamma; \Phi \mid \Delta, x : \tau \vdash x : \tau}$$

$$\frac{}{\Gamma; \Phi \mid \Delta \vdash \top : \mathbb{1}} \qquad \frac{\Gamma; \Phi \mid \Delta \vdash t : \mathbb{0} \qquad \Gamma; \Phi \vdash \tau : \mathcal{F}}{\Gamma; \Phi \mid \Delta \vdash \bot_\tau t : \tau}$$

$$\frac{\Gamma; \Phi \mid \Delta \vdash s : \sigma}{\Gamma; \Phi \mid \Delta \vdash \mathsf{inl}\, s : \sigma + \tau} \qquad \frac{\Gamma; \Phi \mid \Delta \vdash t : \tau}{\Gamma; \Phi \mid \Delta \vdash \mathsf{inr}\, t : \sigma + \tau}$$

$$\frac{\Gamma; \Phi \vdash \tau, \sigma : \mathcal{F} \qquad \Gamma; \Phi \mid \Delta \vdash t : \sigma + \tau \qquad \Gamma; \Phi \mid \Delta, x : \sigma \vdash l : \gamma \qquad \Gamma; \Phi \mid \Delta, y : \tau \vdash r : \gamma}{\Gamma; \Phi \mid \Delta \vdash \mathsf{case}\, t\, \mathsf{of}\, \{\mathsf{inl}\, x \mapsto l;\ \mathsf{inr}\, y \mapsto r\} : \gamma}$$

$$\frac{\Gamma; \Phi \mid \Delta \vdash s : \sigma \qquad \Gamma; \Phi \mid \Delta \vdash t : \tau}{\Gamma; \Phi \mid \Delta \vdash (s, t) : \sigma \times \tau} \qquad \frac{\Gamma; \Phi \mid \Delta \vdash t : \sigma \times \tau}{\Gamma; \Phi \mid \Delta \vdash \pi_1 t : \sigma} \qquad \frac{\Gamma; \Phi \mid \Delta \vdash t : \sigma \times \tau}{\Gamma; \Phi \mid \Delta \vdash \pi_2 t : \tau}$$

$$\frac{\Gamma;\overline{\alpha} \vdash F : \mathcal{F} \qquad \Gamma;\overline{\alpha} \vdash G : \mathcal{F} \qquad \Gamma;\overline{\alpha} \mid \Delta, x : F \vdash t : G}{\Gamma;\emptyset \mid \Delta \vdash L_{\overline{\alpha}} x.t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G}$$

$$\frac{\Gamma;\emptyset \mid \Delta \vdash t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G \qquad \overline{\Gamma;\Phi \vdash \tau : \mathcal{F}} \qquad \Gamma;\Phi \mid \Delta \vdash s : F[\overline{\alpha := \tau}]}{\Gamma;\Phi \mid \Delta \vdash t_{\overline{\tau}} s : G[\overline{\alpha := \tau}]}$$

$$\frac{\Gamma;\overline{\phi},\overline{\gamma} \vdash H : \mathcal{F} \qquad \overline{\Gamma;\overline{\beta},\overline{\gamma} \vdash F : \mathcal{F}} \qquad \overline{\Gamma;\overline{\beta},\overline{\gamma} \vdash G : \mathcal{F}}}{\Gamma;\emptyset \mid \emptyset \vdash \mathsf{map}_H^{\overline{F},\overline{G}} : \mathsf{Nat}^{\emptyset}\ (\overline{\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, F\, G})\ (\mathsf{Nat}^{\overline{\gamma}}\, H[\overline{\phi :=_{\overline{\beta}} F}]\, H[\overline{\phi :=_{\overline{\beta}} G}])}$$

$$\frac{\Gamma;\phi,\overline{\alpha},\overline{\gamma} \vdash H : \mathcal{F}}{\Gamma;\emptyset \mid \emptyset \vdash \mathsf{in}_H : \mathsf{Nat}^{\overline{\beta},\overline{\gamma}} H[\phi :=_{\overline{\beta}} (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\,(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}}$$

$$\frac{\Gamma;\phi,\overline{\alpha},\overline{\gamma} \vdash H : \mathcal{F} \qquad \Gamma;\overline{\beta},\overline{\gamma} \vdash F : \mathcal{F}}{\Gamma;\emptyset \mid \emptyset \vdash \mathsf{fold}_H^F : \mathsf{Nat}^{\emptyset}\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha := \beta}]\, F)\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta})\, F)}$$

In the rule for $L_{\overline{\alpha}} x.t$, the L operator binds all occurrences of the type variables in $\overline{\alpha}$ in the type of the term variable x and in the body t, as well as all occurrences of x in t. In the rule for $t_{\overline{\tau}} s$ there is one functorial expression τ for every functorial variable α. In the rule for $\mathsf{map}_H^{\overline{F},\overline{G}}$ there is one functorial expression F and one functorial expression G for each functorial variable in $\overline{\phi}$. Moreover, for each $\phi^k \in \overline{\phi}$ the number of functorial variables β in the judgments for its corresponding functorial expresssions F and G is k. In the rules for in_H and fold_H^F, the functorial variables in $\overline{\beta}$ are fresh with respect to H, and there is one β for every α. (Recall from above that, in order for the types of in_H and fold_H^F to be well-formed, the length of α must equal the arity of ϕ.) Substitution for terms is the obvious extension of the usual capture-avoiding textual substitution, and Definition 5 ensures that the expected weakening rules for well-formed terms hold.

Using Definition 5 we can represent the `reversePTree` function from Figure 1 in our calculus as

$$\vdash \mathsf{fold}^{PTree\,\alpha}_{\beta+\phi(\beta\times\beta)}\,(\mathsf{in}_{\beta+\phi(\beta\times\beta)} \circ s) : \mathsf{Nat}^{\alpha}(PTree\,\alpha)\,(PTree\,\alpha)$$

where

$$\begin{array}{lcl} \vdash \mathsf{fold}^{PTree\,\alpha}_{\beta+\phi(\beta\times\beta)} & : & \mathsf{Nat}^{\emptyset}(\mathsf{Nat}^{\alpha}(\alpha + PTree\,(\alpha\times\alpha))\,(PTree\,\alpha))\,(\mathsf{Nat}^{\alpha}(PTree\,\alpha)\,(PTree\,\alpha)) \\ \vdash \mathsf{in}_{\beta+\phi(\beta\times\beta)} & : & \mathsf{Nat}^{\alpha}(\alpha + PTree\,(\alpha\times\alpha))\,(PTree\,\alpha) \\ \vdash \mathsf{map}^{\alpha\times\alpha,\alpha\times\alpha}_{PTree\,\alpha} & : & \mathsf{Nat}^{\emptyset}(\mathsf{Nat}^{\alpha}(\alpha\times\alpha)\,(\alpha\times\alpha))\,(\mathsf{Nat}^{\alpha}(PTree\,(\alpha\times\alpha))\,(PTree\,(\alpha\times\alpha))) \end{array}$$

and *swap* and s are the terms

$$\vdash L_{\alpha} p.\,(\pi_2 p, \pi_1 p) : \mathsf{Nat}^{\alpha}(\alpha\times\alpha)\,(\alpha\times\alpha)$$

and

$$\vdash L_{\alpha} t.\,\mathsf{case}\ t\ \mathsf{of}\ \{b \mapsto \mathsf{inl}\ b;\ t' \mapsto \mathsf{inr}\,(\mathsf{map}^{\alpha\times\alpha,\alpha\times\alpha}_{PTree\,\alpha}\ swap\ t')\} : \mathsf{Nat}^{\alpha}(\alpha + PTree\,(\alpha\times\alpha))\,(\alpha + PTree\,(\alpha\times\alpha))$$

respectively. We can similarly represent the `reverseBush` function from Figure 2 as

$$\vdash \mathsf{fold}^{Bush\,\alpha}_{\mathbb{1}+\beta\times\phi(\phi\beta)}\,(\mathsf{in}_{\mathbb{1}+\beta\times\phi(\phi\beta)} \circ (\mathbb{1} + t \circ i \circ i')) : \mathsf{Nat}^{\alpha}(Bush\,\alpha)\,(Bush\,\alpha)$$

where

$$\begin{array}{l} \vdash \mathsf{fold}^{Bush\,\alpha}_{\mathbb{1}+\beta\times\phi(\phi\beta)} : \mathsf{Nat}^{\emptyset}\,(\mathsf{Nat}^{\alpha}\,(\mathbb{1} + \alpha\times Bush\,(Bush\,\alpha)))\,(Bush\,\alpha))\,(\mathsf{Nat}^{\alpha}\,(Bush\,\alpha)\,(Bush\,\alpha)) \\ \vdash \mathsf{in}_{\mathbb{1}+\beta\times\phi(\phi\beta)} : \mathsf{Nat}^{\alpha}\,(\mathbb{1} + \alpha\times Bush\,(Bush\,\alpha))\,(Bush\,\alpha) \end{array}$$

and $bnil$, $bcons$, $\mathsf{in}^{-1}_{\mathbb{1}+\beta\times\phi(\phi\beta)}$, t, i, and i' are the terms

$$\begin{array}{l}
\vdash \mathsf{in}_{\mathbb{1}+\beta\times\phi(\phi\beta)} \circ (L_\alpha\, x.\, \mathsf{inl}\, x) \;:\; \mathsf{Nat}^\alpha\, \mathbb{1}\, (Bush\, \alpha) \\
\vdash \mathsf{in}_{\mathbb{1}+\beta\times\phi(\phi\beta)} \circ (L_\alpha\, x.\, \mathsf{inr}\, x) \;:\; \mathsf{Nat}^\alpha\, (\alpha \times Bush\, (Bush\, \alpha))\, (Bush\, \alpha) \\
\vdash \mathsf{fold}^{(\mathbb{1}+\beta\times\phi(\phi\beta))[\phi:=Bush\,\alpha]}_{\mathbb{1}+\beta\times\phi(\phi\beta)}\, (\mathsf{map}^{(\mathbb{1}+\beta\times\phi(\phi\beta))[\phi:=Bush\,\alpha][\beta:=\alpha],\, Bush\,\alpha}_{\mathbb{1}+\beta\times\phi(\phi\beta)}\, \mathsf{in}_{\mathbb{1}+\beta\times\phi(\phi\beta)}) \\
\quad : \mathsf{Nat}^\alpha\, (Bush\, \alpha)\, (\mathbb{1} + \alpha \times Bush\, (Bush\, \alpha)) \\
\vdash L_\alpha\, (b, s).\, \mathsf{case}\, s\, \{ \quad * \mapsto bcons_\alpha\, b\, (bnil_\alpha\, *); \\
\qquad\qquad (s', u) \mapsto \mathsf{case}\, s'\{ \quad * \mapsto bcons_\alpha\, b\, (bcons_{Bush\,\alpha}\, (bnil_\alpha\, *)\, u); \\
\qquad\qquad\qquad\qquad (b', u') \mapsto bcons_\alpha\, b'(bcons_{Bush\,\alpha}\, (bcons_\alpha\, b\, u)\, u')\}\} \\
\quad : \mathsf{Nat}^\alpha(\alpha \times (\mathbb{1} + (\mathbb{1} + \alpha \times Bush\, (Bush\, \alpha))) \times Bush\, (Bush\, (Bush\, \alpha)))\, (\alpha \times Bush\, (Bush\, \alpha)) \\
\vdash \alpha \times (\mathbb{1} + \mathsf{in}^{-1}_{\mathbb{1}+\beta\times\phi(\phi\beta)} \times Bush\, (Bush\, (Bush\, \alpha))) \\
\quad : \mathsf{Nat}^\alpha(\alpha \times (\mathbb{1} + Bush\, \alpha \times Bush\, (Bush\, (Bush\, \alpha)))) \\
\qquad (\alpha \times (\mathbb{1} + (\mathbb{1} + \alpha \times Bush\, (Bush\, \alpha)) \times Bush\, (Bush\, (Bush\, \alpha)))) \\
\vdash \alpha \times (L_\alpha\, x.\, (\mathsf{in}^{-1}_{\mathbb{1}+\beta\times\phi(\phi\beta)})_{Bush\,\alpha}\, x) \\
\quad : \mathsf{Nat}^\alpha(\alpha \times Bush\, (Bush\, \alpha))\, (\alpha \times (\mathbb{1} + Bush\, (\alpha \times Bush\, (Bush\, (Bush\, \alpha)))))
\end{array}$$

respectively. Here, $\Gamma; \emptyset \mid \Delta \vdash \sigma + \eta : \mathsf{Nat}^{\overline{\alpha}}(\sigma + F)\, (\sigma + G)$ and $\Gamma; \emptyset \mid \Delta \vdash \sigma \times \eta : \mathsf{Nat}^{\overline{\alpha}}(\sigma \times F)\, (\sigma \times G)$ for $\sigma + \eta := L_{\overline{\alpha}}\, x.\, \mathsf{case}\, x\, \mathsf{of}\, \{s \mapsto \mathsf{inl}\, s;\ t \mapsto \mathsf{inr}\, (\eta_{\overline{\alpha}} t)\}$ and $\sigma \times \eta := L_{\overline{\alpha}}\, x.\, (\pi_1 x, \eta_{\overline{\alpha}}(\pi_2 x))$ for $\Gamma; \emptyset \mid \Delta \vdash \eta : \mathsf{Nat}^{\overline{\alpha}} F\, G$ and $\Gamma; \overline{\alpha} \vdash \sigma : \mathcal{F}$.

Because our system is designed to ensure functoriality, and ensuring functoriality does not interact with polymorphism as well as might be hoped, our system cannot express nontrivial recursive functions, such as a concatenation function for perfect trees, that take not just one nested type functorial in a variable α as input, but other inputs functorial in α as well. The fundamental issue is that recursion is expressible only via folds. A fold over a nested type must take a natural transformation as an argument and return a natural transformation, but the polymorphism natural transformations entail leaves no nontrivial way to incorporate data from the second argument to a function with a type like that above into a fold over its first argument. Currying the type also does not help, because the codomain of a fold applied to an algebra must be a functor, and this clearly is not the case for a function that takes more than one input involving the same type variable. Even some recursive functions of a single non-algebraic nested type — e.g., a `reverseBush` function that is a true involution — cannot be expressed as folds because the algebra arguments needed to define them are themselves recursive functions whose type signatures have the same problematic form discussed above. More sophisticated combinators for stylized recursion (e.g., generalized folds) don't mitigate these difficulties, either. Indeed, they appear to be a fundamental consequence of the interaction of functoriality and polymorphism, not just some peculiarity of our particular calculus.

The presence of the "extra" functorial variables in $\overline{\gamma}$ in the rules for $\mathsf{map}^{\overline{F},\overline{G}}_H$, in_H, and fold^F_H also merit special mention. They allows us to map or fold polymorphic functions over nested types. Consider, for example, the function $flatten : \mathsf{Nat}^\beta(PTree\, \beta)\, (List\, \beta)$ that maps perfect trees to lists. Even in the absence of extra variables the instance of map required to map each non-functorial monomorphic instantiation of $flatten$ over a list of perfect trees is well-typed:

$$\frac{\Gamma; \alpha \vdash List\, \alpha \qquad \Gamma; \emptyset \vdash \sigma \qquad \Gamma; \emptyset \vdash \tau \qquad \Gamma; \emptyset \vdash PTree\, \sigma \qquad \Gamma; \emptyset \vdash List\, \tau}{\Gamma; \emptyset \mid \emptyset \vdash \mathsf{map}^{PTree\,\sigma,\, List\,\tau}_{List\,\alpha} : \mathsf{Nat}^\emptyset\, (\mathsf{Nat}^\emptyset\, (PTree\, \sigma)\, (List\, \tau))\, (\mathsf{Nat}^\emptyset\, (List\, (PTree\, \sigma))\, (List\, (List\, \tau)))}$$

But in the absence of $\overline{\gamma}$, the instance

$$\Gamma; \emptyset \mid \emptyset \vdash \mathsf{map}^{PTree\,\beta,\, List\,\beta}_{List\,\alpha} : \mathsf{Nat}^\emptyset\, (\mathsf{Nat}^\beta(PTree\, \beta)\, (List\, \beta))\, (\mathsf{Nat}^\beta\, (List\, (PTree\, \beta))\, (List\, (List\, \beta)))$$

of `map` required to map the *polymorphic flatten* function over a list of perfect trees is not: in that setting the functorial contexts for F and G in the rule for $\mathsf{map}^{F,G}_H$ would have to be empty, but the

fact that the polymorphic *flatten* function is functorial in some variable, say δ, means that it cannot possibly have a type of the form $\mathsf{Nat}^{\emptyset} F\, G$ that would be required for it to be the function input to map. Since untypeability of this instance of map is unsatisfactory in a polymorphic calculus, where we naturally expect to be able to manipulate entire polymorphic functions rather than just their monomorphic instances, we use the "extra" variables in $\overline{\gamma}$ to remedy the situation. Specifically, the rules from Definition 5 ensure that the instance of map needed to map the polymorphic *flatten* function is typeable as follows:

$$\frac{\Gamma; \alpha, \beta \vdash \mathit{List}\,\alpha \qquad \Gamma; \beta \vdash \mathit{PTree}\,\beta \qquad \Gamma; \beta \vdash \mathit{List}\,\beta}{\Gamma; \emptyset \mid \emptyset \vdash \mathsf{map}^{F,G}_{\mathit{List}} : \mathsf{Nat}^{\emptyset}\ (\mathsf{Nat}^{\beta}(\mathit{PTree}\,\beta)\,(\mathit{List}\,\beta))\ (\mathsf{Nat}^{\beta}\,(\mathit{List}\,(\mathit{PTree}\,\beta))\,(\mathit{List}\,(\mathit{List}\,\beta)))}$$

Similar remarks explain the appearance of $\overline{\gamma}$ in the typing rules for in and fold.

3 INTERPRETING TYPES

We denote the category of sets and functions by Set. The category Rel has as its objects triples (A, B, R) where R is a relation between the objects A and B in Set, i.e., a subset of $A \times B$, and has as its morphisms from (A, B, R) to (A', B', R') pairs $(f : A \to A', g : B \to B')$ of morphisms in Set such that $(f\,a, g\,b) \in R'$ whenever $(a, b) \in R$. We write $R : \mathsf{Rel}(A, B)$ in place of (A, B, R) when convenient. If $R : \mathsf{Rel}(A, B)$ we write $\pi_1 R$ and $\pi_2 R$ for the *domain* A of R and the *codomain* B of R, respectively. If $A : \mathsf{Set}$, then we write $\mathsf{Eq}_A = (A, A, \{(x, x) \mid x \in A\})$ for the *equality relation* on A.

The key idea underlying Reynolds' parametricity is to give each type $\tau(\alpha)$ with one free variable α both an *object interpretation* τ_0 taking sets to sets and a *relational interpretation* τ_1 taking relations $R : \mathsf{Rel}(A, B)$ to relations $\tau_1(R) : \mathsf{Rel}(\tau_0(A), \tau_0(B))$, and to interpret each term $t(\alpha, x) : \tau(\alpha)$ with one free term variable $x : \sigma(\alpha)$ as a map t_0 associating to each set A a function $t_0(A) : \sigma_0(A) \to \tau_0(A)$. These interpretations are to be given inductively on the structures of τ and t in such a way that they imply two fundamental theorems. The first is an *Identity Extension Lemma*, which states that $\tau_1(\mathsf{Eq}_A) = \mathsf{Eq}_{\tau_0(A)}$, and is the essential property that makes a model relationally parametric rather than just induced by a logical relation. The second is an *Abstraction Theorem*, which states that, for any $R : \mathsf{Rel}(A, B)$, $(t_0(A), t_0(B))$ is a morphism in Rel from $(\sigma_0(A), \sigma_0(B), \sigma_1(R))$ to $(\tau_0(A), \tau_0(B), \tau_1(R))$. The Identity Extension Lemma is similar to the Abstraction Theorem except that it holds for *all* elements of a type's interpretation, not just those that are interpretations of terms. Similar results are expected to hold for types and terms with any number of free variables.

The key to proving the Identity Extension Lemma (Theorem 22) in our setting is a familiar "cutting down" of the interpretations of universally quantified types, such as our Nat-types, to include only the "parametric" elements. This requires that set interpretations of types are defined simultaneously with their relational interpretations. We give set interpretations for our types in Section 3.1 and give their relational interpretations in Section 3.2. While the set interpretations are relatively straightforward, their relation interpretations are less so, mainly because of the cocontinuity conditions we must impose to ensure that they are well-defined. We take some effort to develop conditions in Section 3.2, which separates Definitions 7 and 16 in space, but otherwise has no impact on the fact that they are given by mutual induction.

3.1 Interpreting Types as Sets

We will interpret the types in our calculus as ω-cocontinuous functors on locally finitely presentable categories [Adámek and Rosický 1994]. Since functor categories of locally finitely presentable categories are again locally finitely presentable, this will ensure, in particular, that the fixed points interpreting μ-types in Set and Rel exist, and thus that both the set and relational interpretations of all of the types in Definition 2 are well-defined [Johann and Polonsky 2019]. To bootstrap this

process, we interpret type variables themselves as ω-cocontinuous functors in Definitions 6 and 14. If $\mathcal{C}$ and $\mathcal{D}$ are locally finitely presentable categories, we write $[\mathcal{C}, \mathcal{D}]$ for the set of ω-cocontinuous functors from $\mathcal{C}$ to $\mathcal{D}$.

Definition 6. *A* set environment *maps each type variable in $\mathbb{T}^k \cup \mathbb{F}^k$ to an element of $[\mathrm{Set}^k, \mathrm{Set}]$. A morphism $f : \rho \to \rho'$ for set environments ρ and ρ' with $\rho|_{\mathbb{T}} = \rho'|_{\mathbb{T}}$ maps each type constructor variable $\psi^k \in \mathbb{T}$ to the identity natural transformation on $\rho\psi^k = \rho'\psi^k$ and each functorial variable $\phi^k \in \mathbb{F}$ to a natural transformation from the k-ary functor $\rho\phi^k$ on* Set *to the k-ary functor $\rho'\phi^k$ on* Set. *Composition of morphisms on set environments is given componentwise, with the identity morphism mapping each set environment to itself. This gives a category of set environments and morphisms between them, which we denote* SetEnv.

When convenient we identify a functor $F : [\mathrm{Set}^0, \mathrm{Set}]$ with the set that is its codomain and consider a set environment to map a type variable of arity 0 to a set. If $\overline{\alpha} = \{\alpha_1, \ldots, \alpha_k\}$ and $\overline{A} = \{A_1, \ldots, A_k\}$, then we write $\rho[\overline{\alpha := A}]$ for the set environment ρ' such that $\rho'\alpha_i = A_i$ for $i = 1, \ldots, k$ and $\rho'\alpha = \rho\alpha$ if $\alpha \notin \{\alpha_1, \ldots, \alpha_k\}$. If ρ is a set environment we write Eq_ρ for the relation environment (see Definition 14) such that $\mathsf{Eq}_\rho \upsilon = \mathsf{Eq}_{\rho\upsilon}$ for every type variable υ. The relational interpretations appearing in the second clause of Definition 7 are given in full in Definition 16.

Definition 7. *The* set interpretation $[\![\cdot]\!]^{\mathrm{Set}} : \mathcal{F} \to [\mathsf{SetEnv}, \mathrm{Set}]$ *is defined by*

$$
\begin{aligned}
[\![\Gamma; \emptyset \vdash \upsilon]\!]^{\mathrm{Set}}\rho &= \rho\upsilon \text{ if } \upsilon \in \mathbb{T}^0 \\
[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathrm{Set}}\rho &= \{\eta : \lambda\overline{A}.\, [\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\alpha := A}] \Rightarrow \lambda\overline{A}.\, [\![\Gamma; \overline{\alpha} \vdash G]\!]^{\mathrm{Set}}\rho[\overline{\alpha := A}] \\
&\qquad \mid \forall \overline{A}, \overline{B} : \mathrm{Set}. \forall \overline{R : \mathrm{Rel}(A, B)}. \\
&\qquad\quad (\eta_{\overline{A}}, \eta_{\overline{B}}) : [\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathrm{Rel}}\mathsf{Eq}_\rho[\overline{\alpha := R}] \to [\![\Gamma; \overline{\alpha} \vdash G]\!]^{\mathrm{Rel}}\mathsf{Eq}_\rho[\overline{\alpha := R}]\} \\
[\![\Gamma; \Phi \vdash \mathbb{0}]\!]^{\mathrm{Set}}\rho &= 0 \\
[\![\Gamma; \Phi \vdash \mathbb{1}]\!]^{\mathrm{Set}}\rho &= 1 \\
[\![\Gamma; \Phi \vdash \phi\overline{\tau}]\!]^{\mathrm{Set}}\rho &= (\rho\phi)\, \overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Set}}\rho} \\
[\![\Gamma; \Phi \vdash \sigma + \tau]\!]^{\mathrm{Set}}\rho &= [\![\Gamma; \Phi \vdash \sigma]\!]^{\mathrm{Set}}\rho + [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Set}}\rho \\
[\![\Gamma; \Phi \vdash \sigma \times \tau]\!]^{\mathrm{Set}}\rho &= [\![\Gamma; \Phi \vdash \sigma]\!]^{\mathrm{Set}}\rho \times [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Set}}\rho \\
[\![\Gamma; \Phi \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\tau}]\!]^{\mathrm{Set}}\rho &= (\mu T^{\mathrm{Set}}_{H,\rho})\, \overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Set}}\rho} \\
\text{where } T^{\mathrm{Set}}_{H,\rho}\, F &= \lambda\overline{A}.[\![\Gamma; \Phi, \phi, \overline{\alpha} \vdash H]\!]^{\mathrm{Set}}\rho[\phi := F][\overline{\alpha := A}] \\
\text{and } T^{\mathrm{Set}}_{H,\rho}\, \eta &= \lambda\overline{A}.[\![\Gamma; \Phi, \phi, \overline{\alpha} \vdash H]\!]^{\mathrm{Set}}id_\rho[\phi := \eta][\overline{\alpha := id_A}]
\end{aligned}
$$

The interpretations in Definition 7 respect weakening, i.e., a type and its weakenings have the same set interpretations. The same holds for the actions of these interpretations on morphisms in Definition 8 below. If ρ is a set environment and $\vdash \tau : \mathcal{F}$ then we may write $[\![\vdash \tau]\!]^{\mathrm{Set}}$ instead of $[\![\vdash \tau]\!]^{\mathrm{Set}}\rho$ since the environment is immaterial. We note that the second clause of Definition 7 does indeed define a set: local finite presentability of Set and ω-cocontinuity of $[\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathrm{Set}}\rho$ ensure that $\{\eta : [\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathrm{Set}}\rho \Rightarrow [\![\Gamma; \overline{\alpha} \vdash G]\!]^{\mathrm{Set}}\rho\}$ (which contains $[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathrm{Set}}\rho$) is a subset of $\left\{([\![\Gamma; \overline{\alpha} \vdash G]\!]^{\mathrm{Set}}\rho[\overline{\alpha := S}])^{([\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\alpha := S}])} \;\middle|\; \overline{S} = (S_1, \ldots, S_{|\overline{\alpha}|}),\ \text{and}\ S_i \text{ is a finite set for } i = 1, \ldots, |\overline{\alpha}|\right\}$. There are countably many choices for tuples $\overline{S}$, and each of these gives rise to a morphism from $[\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\alpha := S}]$ to $[\![\Gamma; \overline{\alpha} \vdash G]\!]^{\mathrm{Set}}\rho[\overline{\alpha := S}]$. But there are only Set-many choices of morphisms between any two objects since Set is locally small. Finally, interpretations of Nat types in Definitions 7 and 16 ensure $[\![\Gamma \vdash \sigma \to \tau]\!]^{\mathrm{Set}}$ and $[\![\Gamma \vdash \forall\alpha.\tau]\!]^{\mathrm{Set}}$ are as expected in any parametric model.

In order to make sense of the last clause in Definition 7, we need to know that, for each $\rho \in \mathsf{SetEnv}$, $T^{\mathsf{Set}}_{H,\rho}$ is an ω-cocontinuous endofunctor on $[\mathsf{Set}^k, \mathsf{Set}]$, and thus admits a fixed point. Since $T^{\mathsf{Set}}_{H,\rho}$ is defined in terms of $[\![\Gamma; \Phi, \phi, \overline{\alpha} \vdash H]\!]^{\mathsf{Set}}$, this means that interpretations of types must be such functors, which in turn means that the actions of set interpretations of types on objects and on morphisms in SetEnv are intertwined. Fortunately, we know from [Johann and Polonsky 2019] that, for every $\Gamma; \overline{\alpha} \vdash \tau : \mathcal{F}$, $[\![\Gamma; \overline{\alpha} \vdash \tau]\!]^{\mathsf{Set}}$ is actually in $[\mathsf{Set}^k, \mathsf{Set}]$ where $k = |\overline{\alpha}|$. This means that for each $[\![\Gamma; \Phi, \phi^k, \overline{\alpha} \vdash H]\!]^{\mathsf{Set}}$, the corresponding operator T^{Set}_H can be extended to a *functor* from SetEnv to $[[\mathsf{Set}^k, \mathsf{Set}], [\mathsf{Set}^k, \mathsf{Set}]]$. The action of T^{Set}_H on an object $\rho \in \mathsf{SetEnv}$ is given by the higher-order functor $T^{\mathsf{Set}}_{H,\rho}$, whose actions on objects (functors in $[\mathsf{Set}^k, \mathsf{Set}]$) and morphisms (natural transformations between such functors) are given in Definition 7. Its action on a morphism $f : \rho \to \rho'$ is the higher-order natural transformation $T^{\mathsf{Set}}_{H,f} : T^{\mathsf{Set}}_{H,\rho} \to T^{\mathsf{Set}}_{H,\rho'}$ whose action on $F : [\mathsf{Set}^k, \mathsf{Set}]$ is the natural transformation $T^{\mathsf{Set}}_{H,f}\, F : T^{\mathsf{Set}}_{H,\rho}\, F \to T^{\mathsf{Set}}_{H,\rho'}\, F$ whose component at $\overline{A}$ is $(T^{\mathsf{Set}}_{H,f}\, F)_{\overline{A}} = [\![\Gamma; \Phi, \phi, \overline{\alpha} \vdash H]\!]^{\mathsf{Set}} f[\phi := id_F][\overline{\alpha := id_A}]$. The next definition uses the functor T^{Set}_H to define the actions of functors interpreting types on morphisms between set environments.

Definition 8. *Let $f : \rho \to \rho'$ for set environments ρ and ρ' (so that $\rho|_{\mathbb{T}} = \rho'|_{\mathbb{T}}$). The action $[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f$ of $[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}}$ on the morphism f is given as follows:*

- *If $\Gamma, \upsilon; \emptyset \vdash \upsilon$ then $[\![\Gamma, \upsilon; \emptyset \vdash \upsilon]\!]^{\mathsf{Set}} f = id_{\rho\upsilon}$*
- *If $\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G$ then $[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathsf{Set}} f = id_{[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathsf{Set}} \rho}$*
- *If $\Gamma; \Phi \vdash \mathbb{0}$ then $[\![\Gamma; \Phi \vdash \mathbb{0}]\!]^{\mathsf{Set}} f = id_0$*
- *If $\Gamma; \Phi \vdash \mathbb{1}$ then $[\![\Gamma; \Phi \vdash \mathbb{1}]\!]^{\mathsf{Set}} f = id_1$*
- *If $\Gamma; \Phi \vdash \phi\overline{\tau}$ then $[\![\Gamma; \Phi \vdash \phi\overline{\tau}]\!]^{\mathsf{Set}} f : [\![\Gamma; \Phi \vdash \phi\overline{\tau}]\!]^{\mathsf{Set}} \rho \to [\![\Gamma; \Phi \vdash \phi\overline{\tau}]\!]^{\mathsf{Set}} \rho' = (\rho\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho} \to (\rho'\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho'}$ is defined by $[\![\Gamma; \Phi \vdash \phi\overline{\tau}]\!]^{\mathsf{Set}} f = (f\phi)_{\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho'}} \circ (\rho\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f} = (\rho'\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f} \circ (f\phi)_{\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho}}$. The latter equality holds because $\rho\phi$ and $\rho'\phi$ are functors and $f\phi : \rho\phi \to \rho'\phi$ is a natural transformation, so the following naturality square commutes:*

$$\begin{array}{ccc} (\rho\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho} & \xrightarrow{(f\phi)_{\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho}}} & (\rho'\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho} \\ {\scriptstyle (\rho\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f}}\downarrow & & \downarrow{\scriptstyle (\rho'\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f}} \\ (\rho\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho'} & \xrightarrow{(f\phi)_{\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho'}}} & (\rho'\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho'} \end{array} \tag{1}$$

- *If $\Gamma; \Phi \vdash \sigma + \tau$ then $[\![\Gamma; \Phi \vdash \sigma + \tau]\!]^{\mathsf{Set}} f$ is defined by $[\![\Gamma; \Phi \vdash \sigma + \tau]\!]^{\mathsf{Set}} f(\mathsf{inl}\, x) = \mathsf{inl}\,([\![\Gamma; \Phi \vdash \sigma]\!]^{\mathsf{Set}} f x)$ and $[\![\Gamma; \Phi \vdash \sigma + \tau]\!]^{\mathsf{Set}} f(\mathsf{inr}\, y) = \mathsf{inr}\,([\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f y)$.*
- *If $\Gamma; \Phi \vdash \sigma \times \tau$ then $[\![\Gamma; \Phi \vdash \sigma \times \tau]\!]^{\mathsf{Set}} f = [\![\Gamma; \Phi \vdash \sigma]\!]^{\mathsf{Set}} f \times [\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f$.*
- *If $\Gamma; \Phi \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\tau}$ then $[\![\Gamma; \Phi \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\tau}]\!]^{\mathsf{Set}} f : [\![\Gamma; \Phi \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\tau}]\!]^{\mathsf{Set}} \rho \to [\![\Gamma; \Phi \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\tau}]\!]^{\mathsf{Set}}$ $= (\mu T^{\mathsf{Set}}_{H,\rho})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho} \to (\mu T^{\mathsf{Set}}_{H,\rho'})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho'}$ is defined by $(\mu T^{\mathsf{Set}}_{H,f})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho'} \circ (\mu T^{\mathsf{Set}}_{H,\rho})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f} = (\mu T^{\mathsf{Set}}_{H,\rho'})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f} \circ (\mu T^{\mathsf{Set}}_{H,f})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho}$. The latter equality holds because $\mu T^{\mathsf{Set}}_{H,\rho}$ and $\mu T^{\mathsf{Set}}_{H,\rho'}$ are functors and $\mu T^{\mathsf{Set}}_{H,f} : \mu T^{\mathsf{Set}}_{H,\rho} \to \mu T^{\mathsf{Set}}_{H,\rho'}$ is a natural transformation, so the following naturality square commutes:*

$$\begin{array}{ccc} (\mu T^{\mathsf{Set}}_{H,\rho})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho} & \xrightarrow{(\mu T^{\mathsf{Set}}_{H,f})_{\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho}}} & (\mu T^{\mathsf{Set}}_{H,\rho'})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho} \\ {\scriptstyle (\mu T^{\mathsf{Set}}_{H,\rho})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f}}\downarrow & & \downarrow{\scriptstyle (\mu T^{\mathsf{Set}}_{H,\rho'})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} f}} \\ (\mu T^{\mathsf{Set}}_{H,\rho})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho'} & \xrightarrow{(\mu T^{\mathsf{Set}}_{H,f})_{\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho'}}} & (\mu T^{\mathsf{Set}}_{H,\rho'})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho'} \end{array} \tag{2}$$

3.2 Interpreting Types as Relations

Definition 9. *A k-ary relation transformer F is a triple (F^1, F^2, F^*), where $F^1, F^2 : [\mathsf{Set}^k, \mathsf{Set}]$ are functors, $F^* : [\mathsf{Rel}^k, \mathsf{Rel}]$ is a functor, if $R_1 : \mathsf{Rel}(A_1, B_1), \ldots, R_k : \mathsf{Rel}(A_k, B_k)$, then $F^*\overline{R} : \mathsf{Rel}(F^1\overline{A}, F^2\overline{B})$, and if $(\alpha_1, \beta_1) \in \mathsf{Hom}_{\mathsf{Rel}}(R_1, S_1), \ldots, (\alpha_k, \beta_k) \in \mathsf{Hom}_{\mathsf{Rel}}(R_k, S_k)$ then $F^*\overline{(\alpha, \beta)} = (F^1\overline{\alpha}, F^2\overline{\beta})$. We define $F\overline{R}$ to be $F^*\overline{R}$ and $F\overline{(\alpha, \beta)}$ to be $F^*\overline{(\alpha, \beta)}$.*

The last clause of Definition 9 expands to: if $\overline{(a, b) \in R}$ implies $\overline{(\alpha\, a, \beta\, b) \in S}$ then $(c, d) \in F^*\overline{R}$ implies $(F^1\overline{\alpha}\, c, F^2\overline{\beta}\, d) \in F^*\overline{S}$. When convenient we identify a 0-ary relation transformer (A, B, R) with $R : \mathsf{Rel}(A, B)$. We may also write $\pi_1 F$ for F^1 and $\pi_2 F$ for F^2. We extend these conventions to relation environments, introduced in Definition 14 below, in the obvious way.

Definition 10. *The category RT_k of k-ary relation transformers is given by the following data:*

- *An object of RT_k is a relation transformer.*
- *A morphism $\delta : (G^1, G^2, G^*) \to (H^1, H^2, H^*)$ in RT_k is a pair of natural transformations (δ^1, δ^2) where $\delta^1 : G^1 \to H^1$, $\delta^2 : G^2 \to H^2$ such that, for all $\overline{R : \mathsf{Rel}(A, B)}$, if $(x, y) \in G^*\overline{R}$ then $(\delta^1_{\overline{A}} x, \delta^2_{\overline{B}} y) \in H^*\overline{R}$.*
- *Identity morphisms and composition are inherited from the category of functors on* Set.

Definition 11. *An endofunctor H on RT_k is a triple $H = (H^1, H^2, H^*)$, where*

- *H^1 and H^2 are functors from $[\mathsf{Set}^k, \mathsf{Set}]$ to $[\mathsf{Set}^k, \mathsf{Set}]$*
- *H^* is a functor from RT_k to $[\mathsf{Rel}^k, \mathsf{Rel}]$*
- *for all $\overline{R : \mathsf{Rel}(A, B)}$, $\pi_1((H^*(\delta^1, \delta^2))_{\overline{R}}) = (H^1\delta^1)_{\overline{A}}$ and $\pi_2((H^*(\delta^1, \delta^2))_{\overline{R}}) = (H^2\delta^2)_{\overline{B}}$*
- *The action of H on objects is given by $H\,(F^1, F^2, F^*) = (H^1F^1,\ H^2F^2,\ H^*(F^1, F^2, F^*))$*
- *The action of H on morphisms is given by $H\,(\delta^1, \delta^2) = (H^1\delta^1, H^2\delta^2)$ for $(\delta^1, \delta^2) : (F^1, F^2, F^*) \to (G^1, G^2, G^*)$*

Since the results of applying an endofunctor H to k-ary relation transformers and morphisms between them must again be k-ary relation transformers and morphisms between them, respectively, Definition 11 implicitly requires that the following three conditions hold: *i*) if $R_1 : \mathsf{Rel}(A_1, B_1), \ldots, R_k : \mathsf{Rel}(A_k, B_k)$, then $H^*(F^1, F^2, F^*)\overline{R} : \mathsf{Rel}(H^1F^1\overline{A}, H^2F^2\overline{B})$; *ii*) if $(\alpha_1, \beta_1) \in \mathsf{Hom}_{\mathsf{Rel}}(R_1, S_1), \ldots, (\alpha_k, \beta_k) \in \mathsf{Hom}_{\mathsf{Rel}}(R_k, S_k)$, then $H^*(F^1, F^2, F^*)\,\overline{(\alpha, \beta)} = (H^1F^1\overline{\alpha}, H^2F^2\overline{\beta})$; and $(\delta^1, \delta^2) : (F^1, F^2, F^*) \to (G^1, G^2, G^*)$ and $R_1 : \mathsf{Rel}(A_1, B_1), \ldots, R_k : \mathsf{Rel}(A_k, B_k)$, then $((H^1\delta^1)_{\overline{A}} x, (H^2\delta^2)_{\overline{B}} y) \in H^*(G^1, G^2, G^*)\overline{R}$ whenever $(x, y) \in H^*(F^1, F^2, F^*)\overline{R}$. Note, however, that this last condition is automatically satisfied because it is implied by the third bullet point of Definition 11.

Definition 12. *If H and K are endofunctors on RT_k, then a* natural transformation $\sigma : H \to K$ *is a pair $\sigma = (\sigma^1, \sigma^2)$, where $\sigma^1 : H^1 \to K^1$ and $\sigma^2 : H^2 \to K^2$ are natural transformations between endofunctors on $[\mathsf{Set}^k, \mathsf{Set}]$ and the component of σ at $F \in RT_k$ is given by $\sigma_F = (\sigma^1_{F^1}, \sigma^2_{F^2})$.*

Definition 12 entails that $\sigma^i_{F^i}$ must be natural in $F^i : [\mathsf{Set}^k, \mathsf{Set}]$, and, for every F, both $(\sigma^1_{F^1})_{\overline{A}}$ and $(\sigma^2_{F^2})_{\overline{A}}$ must be natural in $\overline{A}$. Moreover, since the results of applying σ to k-ary relation transformers must be morphisms of k-ary relation transformers, Definition 12 implicitly requires that $(\sigma_F)_{\overline{R}} = ((\sigma^1_{F^1})_{\overline{A}}, (\sigma^2_{F^2})_{\overline{B}})$ is a morphism in Rel for any k-tuple of relations $\overline{R : \mathsf{Rel}(A, B)}$, i.e., that if $(x, y) \in H^*F\overline{R}$, then $((\sigma^1_{F^1})_{\overline{A}} x, (\sigma^2_{F^2})_{\overline{B}} y) \in K^*F\overline{R}$.

Critically, we can compute ω-directed colimits in RT_k: it is not hard to see that if $\mathcal{D}$ is an ω-directed set then $\varinjlim_{d \in \mathcal{D}} (F^1_d, F^2_d, F^*_d) = (\varinjlim_{d \in \mathcal{D}} F^1_d, \varinjlim_{d \in \mathcal{D}} F^2_d, \varinjlim_{d \in \mathcal{D}} F^*_d)$. We define an endofunctor $T = (T^1, T^2, T^*)$ on RT_k to be *ω-cocontinuous* if T^1 and T^2 are ω-cocontinuous endofunctors on

$[\mathsf{Set}^k, \mathsf{Set}]$ and T^* is an ω-cocontinuous functor from RT_k to $[\mathsf{Rel}^k, \mathsf{Rel}]$, i.e., is in $[RT_k, [\mathsf{Rel}^k, \mathsf{Rel}]]$. Now, for any k, any $A : \mathsf{Set}$, and any $R : \mathsf{Rel}(A, B)$, let K_A^{Set} be the constantly A-valued functor from Set^k to Set and K_R^{Rel} be the constantly R-valued functor from Rel^k to Rel. Also let 0 denote either the initial object of Set or the initial object of Rel, as appropriate. Observing that, for every k, K_0^{Set} is initial in $[\mathsf{Set}^k, \mathsf{Set}]$, and K_0^{Rel} is initial in $[\mathsf{Rel}^k, \mathsf{Rel}]$, we have that, for each k, $K_0 = (K_0^{\mathsf{Set}}, K_0^{\mathsf{Set}}, K_0^{\mathsf{Rel}})$ is initial in RT_k. Thus, if $T = (T^1, T^2, T^*) : RT_k \to RT_k$ is an endofunctor on RT_k then we can define the relation transformer μT to be $\varinjlim_{n \in \mathbb{N}} T^n K_0$. It is not hard to see that μT is given explicitly as

$$\mu T = (\mu T^1, \mu T^2, \varinjlim_{n \in \mathbb{N}} (T^n K_0)^*) \tag{3}$$

and that, as our notation suggests, it really is a fixpoint for T if T is ω-cocontinuous:

Lemma 13. *For any $T : [RT_k, RT_k]$, $\mu T \cong T(\mu T)$.*

The isomorphism is given by the morphisms $(in_1, in_2) : T(\mu T) \to \mu T$ and $(in_1^{-1}, in_2^{-1}) : \mu T \to T(\mu T)$ in RT_k. The latter is always a morphism in RT_k, but the former need not be if T is not ω-cocontinuous.

It is worth noting that the third component in Equation (3) is the colimit in $[\mathsf{Rel}^k, \mathsf{Rel}]$ of third components of relation transformers, rather than a fixpoint of an endofunctor on $[\mathsf{Rel}^k, \mathsf{Rel}]$. That there is an asymmetry between the first two components of μT and its third is an important conceptual observation, and reflects the fact that the third component of an endofunctor on RT_k need not be a functor on all of $[\mathsf{Rel}^k, \mathsf{Rel}]$. In particular, although we can define $T_{H,\rho} F$ for a relation transformer F in Definition 16 below, it is not clear how we could define it for $F : [\mathsf{Rel}^k, \mathsf{Rel}]$.

Definition 14. *A* relation environment *maps each each type variable in $\mathbb{T}^k \cup \mathbb{F}^k$ to a k-ary relation transformer. A morphism $f : \rho \to \rho'$ for relation environments ρ and ρ' with $\rho|_{\mathbb{T}} = \rho'|_{\mathbb{T}}$ maps each type constructor variable $\psi^k \in \mathbb{T}$ to the identity morphism on $\rho\psi^k = \rho'\psi^k$ and each functorial variable $\phi^k \in \mathbb{F}$ to a morphism from the k-ary relation transformer $\rho\phi$ to the k-ary relation transformer $\rho'\phi$. Composition of morphisms on relation environments is given componentwise, with the identity morphism mapping each relation environment to itself. This gives a category of relation environments and morphisms between them, which we denote* RelEnv.

When convenient we identify a 0-ary relation transformer with the relation (transformer) that is its codomain and consider a relation environment to map a type variable of arity 0 to a relation. We write $\rho[\overline{\alpha := R}]$ for the relation environment ρ' such that $\rho'\alpha_i = R_i$ for $i = 1, \ldots, k$ and $\rho'\alpha = \rho\alpha$ if $\alpha \notin \{\alpha_1, \ldots, \alpha_k\}$. If ρ is a relation environment, we write $\pi_1\rho$ and $\pi_2\rho$ for the set environments mapping each type variable ϕ to the functors $(\rho\phi)^1$ and $(\rho\phi)^2$, respectively.

We define, for each k, the notion of an ω-cocontinuous functor from RelEnv to RT_k:

Definition 15. *A functor $H : [\mathsf{RelEnv}, RT_k]$ is a triple $H = (H^1, H^2, H^*)$, where*

- *H^1 and H^2 are objects in $[\mathsf{SetEnv}, [\mathsf{Set}^k, \mathsf{Set}]]$*
- *H^* is a an object in $[\mathsf{RelEnv}, [\mathsf{Rel}^k, \mathsf{Rel}]]$*
- *for all $\overline{R : \mathsf{Rel}(A, B)}$ and morphisms f in* RelEnv*, $\pi_1(H^* f\, \overline{R}) = H^1(\pi_1 f)\, \overline{A}$ and $\pi_2(H^* f\, \overline{R}) = H^2(\pi_2 f)\, \overline{B}$*
- *The action of H on ρ in* RelEnv *is given by $H\rho = (H^1(\pi_1\rho), H^2(\pi_2\rho), H^*\rho)$*
- *The action of H on morphisms $f : \rho \to \rho'$ in* RelEnv *is given by $Hf = (H^1(\pi_1 f), H^2(\pi_2 f))$*

Spelling out the last two bullet points above gives the following analogues of the three conditions immediately following Definition 11: *i*) if $R_1 : \mathsf{Rel}(A_1, B_1), \ldots, R_k : \mathsf{Rel}(A_k, B_k)$, then $H^*\rho\, \overline{R} : \mathsf{Rel}(H^1(\pi_1\rho)\, \overline{A}, H^2(\pi_2\rho)\, \overline{B})$; *ii*) if $(\alpha_1, \beta_1) \in \mathsf{Hom}_{\mathsf{Rel}}(R_1, S_1), \ldots, (\alpha_k, \beta_k) \in \mathsf{Hom}_{\mathsf{Rel}}(R_k, S_k)$, then $H^*\rho\, \overline{(\alpha, \beta)} = (H^1(\pi_1\rho)\, \overline{\alpha}, H^2(\pi_2\rho)\, \overline{\beta})$; and *iii*) if $f : \rho \to \rho'$ and $R_1 : \mathsf{Rel}(A_1, B_1), \ldots, R_k :$

$\mathsf{Rel}(A_k, B_k)$, then $(H^1(\pi_1 f)\,\overline{A}\,x, H^2(\pi_2 f)\,\overline{B}\,y) \in H^*\rho'\,\overline{R}$ whenever $(x,y) \in H^*\rho\,\overline{R}$. As before, the last condition is automatically satisfied because it is implied by the third bullet point of Definition 15.

Considering RelEnv as a product $\Pi_{\phi^k \in \mathbb{T}\cup\mathbb{F}} RT_k$, we extend the computation of ω-directed colimits in RT_k to compute colimits in RelEnv componentwise. We similarly extend the notion of an ω-cocontinuous endofunctor on RT_k componentwise to give a notion of ω-cocontinuity for functors from RelEnv to RT_k. Recalling from the start of this subsection that Definition 16 is given mutually inductively with Definition 7 we can, at last, define:

Definition 16. *The* relational interpretation $[\![\cdot]\!]^{\mathsf{Rel}} : \mathcal{F} \to [\mathsf{RelEnv}, \mathsf{Rel}]$ *is defined by*

$$
\begin{aligned}
[\![\Gamma; \emptyset \vdash v]\!]^{\mathsf{Rel}}\rho &= \rho v \text{ if } v \in \mathbb{T}^0 \\
[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\,F\,G]\!]^{\mathsf{Rel}}\rho &= \{\eta : \lambda\overline{R}.\,[\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathsf{Rel}}\rho[\overline{\alpha := R}] \Rightarrow \lambda\overline{R}.\,[\![\Gamma; \overline{\alpha} \vdash G]\!]^{\mathsf{Rel}}\rho[\overline{\alpha := R}]\} \\
&= \{(t,t') \in [\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\,F\,G]\!]^{\mathsf{Set}}(\pi_1\rho) \times [\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\,F\,G]\!]^{\mathsf{Set}}(\pi_2\rho) \mid \\
&\qquad \forall R_1 : \mathsf{Rel}(A_1, B_1) \ldots R_k : \mathsf{Rel}(A_k, B_k). \\
&\qquad\quad (t_{\overline{A}}, t'_{\overline{B}}) \in ([\![\Gamma; \overline{\alpha} \vdash G]\!]^{\mathsf{Rel}}\rho[\overline{\alpha := R}])^{[\![\Gamma;\overline{\alpha}\vdash F]\!]^{\mathsf{Rel}}\rho[\overline{\alpha:=R}]}\} \\
[\![\Gamma; \Phi \vdash \mathbb{0}]\!]^{\mathsf{Rel}}\rho &= 0 \\
[\![\Gamma; \Phi \vdash \mathbb{1}]\!]^{\mathsf{Rel}}\rho &= 1 \\
[\![\Gamma; \Phi \vdash \phi\overline{\tau}]\!]^{\mathsf{Rel}}\rho &= (\rho\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Rel}}\rho} \\
[\![\Gamma; \Phi \vdash \sigma + \tau]\!]^{\mathsf{Rel}}\rho &= [\![\Gamma; \Phi \vdash \sigma]\!]^{\mathsf{Rel}}\rho + [\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Rel}}\rho \\
[\![\Gamma; \Phi \vdash \sigma \times \tau]\!]^{\mathsf{Rel}}\rho &= [\![\Gamma; \Phi \vdash \sigma]\!]^{\mathsf{Rel}}\rho \times [\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Rel}}\rho \\
[\![\Gamma; \Phi \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\tau}]\!]^{\mathsf{Rel}}\rho &= (\mu T_{H,\rho})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Rel}}\rho} \\
\text{where } T_{H,\rho} &= (T^{\mathsf{Set}}_{H,\pi_1\rho}, T^{\mathsf{Set}}_{H,\pi_2\rho}, T^{\mathsf{Rel}}_{H,\rho}) \\
\text{and } T^{\mathsf{Rel}}_{H,\rho}\,F &= \lambda\overline{R}.[\![\Gamma; \Phi, \phi, \overline{\alpha} \vdash H]\!]^{\mathsf{Rel}}\rho[\phi := F][\overline{\alpha := R}] \\
\text{and } T^{\mathsf{Rel}}_{H,\rho}\,\delta &= \lambda\overline{R}.[\![\Gamma; \Phi, \phi, \overline{\alpha} \vdash H]\!]^{\mathsf{Rel}}id_\rho[\phi := \delta][\overline{\alpha := id_{\overline{R}}}]
\end{aligned}
$$

The interpretations in Definition 16, as well as in Definition 17 below, respect weakening, and also ensure that $[\![\Gamma \vdash \forall\alpha.\tau]\!]^{\mathsf{Rel}}$ are as expected in any parametric model. If ρ is a relational environment and $\vdash \tau : \mathcal{F}$, then we write $[\![\vdash \tau]\!]^{\mathsf{Rel}}$ instead of $[\![\vdash \tau]\!]^{\mathsf{Rel}}\rho$ as for set interpretations. For the last clause in Definition 16 to be well-defined we need T_ρ to be an ω-cocontinuous endofunctor on RT so that, by Lemma 13, it admits a fixed point. Since T_ρ is defined in terms of $[\![\Gamma; \Phi, \phi^k, \overline{\alpha} \vdash H]\!]^{\mathsf{Rel}}$, this means that relational interpretations of types must be ω-cocontinuous functors from RelEnv to RT_0, which in turn entails that the actions of relational interpretations of types on objects and on morphisms in RelEnv are intertwined. As for set interpretations, we know from [Johann and Polonsky 2019] that, for every $\Gamma; \overline{\alpha} \vdash \tau : \mathcal{F}$, $[\![\Gamma; \overline{\alpha} \vdash \tau]\!]^{\mathsf{Set}}$ is actually in $[\mathsf{Rel}^k, \mathsf{Rel}]$ where $k = |\overline{\alpha}|$. We first define the actions of each of these functors on morphisms between environments in Definition 17, and then argue that the functors given by Definitions 16 and 17 are well-defined and have the required properties. To do this, we extend T_H to a *functor* from RelEnv to $[[\mathsf{Rel}^k, \mathsf{Rel}], [\mathsf{Rel}^k, \mathsf{Rel}]]$. Its action on an object $\rho \in \mathsf{RelEnv}$ is given by the higher-order functor $T^{\mathsf{Rel}}_{H,\rho}$ whose actions on objects and morphisms are given in Definition 17. Its action on a morphism $f : \rho \to \rho'$ is the higher-order natural transformation $T_{H,f} : T_{H,\rho} \to T_{H,\rho'}$ whose action on any $F : [\mathsf{Rel}^k, \mathsf{Rel}]$ is the natural transformation $T_{H,f}\,F : T_{H,\rho}\,F \to T_{H,\rho'}\,F$ whose component at $\overline{R}$ is $(T_{H,f}\,F)_{\overline{R}} = [\![\Gamma; \Phi, \phi, \overline{\alpha} \vdash H]\!]^{\mathsf{Rel}}f[\phi := id_F][\overline{\alpha := id_R}]$. The next definition uses the functor T_H to define the actions of functors interpreting types on morphisms between relation environments.

Definition 17. *Let $f : \rho \to \rho'$ for relation environments ρ and ρ' (so that $\rho|_{\mathbb{T}} = \rho'|_{\mathbb{T}}$). The action $[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} f$ of $[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}}$ on the morphism f is given as follows:*

- *If* $\Gamma, v; \emptyset \vdash v$ *then* $[\![\Gamma, v; \emptyset \vdash v]\!]^{\mathrm{Rel}} f = id_{\rho v}$
- *If* $\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G$, *then* $[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathrm{Rel}} f = id_{[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathrm{Rel}} \rho}$
- *If* $\Gamma; \Phi \vdash \mathbb{0}$ *then* $[\![\Gamma; \Phi \vdash \mathbb{0}]\!]^{\mathrm{Rel}} f = id_0$
- *If* $\Gamma; \Phi \vdash \mathbb{1}$ *then* $[\![\Gamma; \Phi \vdash \mathbb{1}]\!]^{\mathrm{Rel}} f = id_1$
- *If* $\Gamma; \Phi \vdash \phi\overline{\tau}$, *then* $[\![\Gamma; \Phi \vdash \phi\overline{\tau}]\!]^{\mathrm{Rel}} f : [\![\Gamma; \Phi \vdash \phi\overline{\tau}]\!]^{\mathrm{Rel}} \rho \to [\![\Gamma; \Phi \vdash \phi\overline{\tau}]\!]^{\mathrm{Rel}} \rho' = (\rho\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} \rho} \to (\rho'\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} \rho'}$ *is defined by* $[\![\Gamma; \Phi \vdash \phi\tau A]\!]^{\mathrm{Rel}} f = (f\phi)_{\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} \rho'}} \circ (\rho\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} f} = (\rho'\phi)\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} f} \circ (f\phi)_{\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} \rho}}$
- *If* $\Gamma; \Phi \vdash \sigma + \tau$ *then* $[\![\Gamma; \Phi \vdash \sigma + \tau]\!]^{\mathrm{Rel}} f$ *is defined by* $[\![\Gamma; \Phi \vdash \sigma + \tau]\!]^{\mathrm{Rel}} f(\mathsf{inl}\, x) = \mathsf{inl}\, ([\![\Gamma; \Phi \vdash \sigma]\!]^{\mathrm{Rel}} f x)$ *and* $[\![\Gamma; \Phi \vdash \sigma + \tau]\!]^{\mathrm{Rel}} f(\mathsf{inr}\, y) = \mathsf{inr}\, ([\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} f y)$
- *If* $\Gamma; \Phi \vdash \sigma \times \tau$ *then* $[\![\Gamma; \Phi \vdash \sigma \times \tau]\!]^{\mathrm{Rel}} f = [\![\Gamma; \Phi \vdash \sigma]\!]^{\mathrm{Rel}} f \times [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} f$
- *If* $\Gamma; \Phi \vdash (\mu\phi^k.\lambda\overline{\alpha}.H)\overline{\tau}$ *then* $[\![\Gamma; \Phi \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\tau}]\!]^{\mathrm{Rel}} f = (\mu T_{H,f})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} \rho'} \circ (\mu T_{H,\rho})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} f} = (\mu T_{H,\rho'})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} f} \circ (\mu T_{H,f})\overline{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}} \rho}$

To see that the functors given by Definitions 16 and 17 are well-defined we must show that, for every H, $T_{H,\rho}\, F$ is a relation transformer for any relation transformer F, and that $T_{H,f}\, F : T_{H,\rho}\, F \to T_{H,\rho'}\, F$ is a morphism of relation transformers for every relation transformer F and every morphism $f : \rho \to \rho'$ in RelEnv. This is an immediate consequence of

Lemma 18. *For every* $\Gamma; \Phi \vdash \tau$, $[\![\Gamma; \Phi \vdash \tau]\!] = ([\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Set}}, [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Set}}, [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}}) \in [\mathsf{RelEnv}, RT_0]$.

The proof is a straightforward induction on the structure of τ, using an appropriate result from [Johann and Polonsky 2019] to deduce ω-cocontinuity of $[\![\Gamma; \Phi \vdash \tau]\!]$ in each case, together with Lemma 13 and Equation 3 in the μ-case.

We can also prove by simultaneous induction that our interpretations of types interact well with demotion of functorial variables. Indeed, we have that, for any ρ, ρ' and $f : \rho \to \rho'$ in SetEnv,

$$[\![\Gamma; \Phi, \phi \vdash F]\!]^{\mathrm{Set}} \rho = [\![\Gamma, \psi; \Phi \vdash F[\phi :== \psi]]\!]^{\mathrm{Set}} \rho[\psi := \rho\phi] \tag{4}$$

$$[\![\Gamma; \Phi, \phi \vdash F]\!]^{\mathrm{Set}} f = [\![\Gamma, \psi; \Phi \vdash F[\phi :== \psi]]\!]^{\mathrm{Set}} f[\psi \mapsto f\phi] \tag{5}$$

Moreover, if $\rho, \rho' : \mathsf{SetEnv}$ are such that $\rho\phi = \rho\psi = \rho'\phi = \rho'\psi$, $f : \rho \to \rho'$ in SetEnv is such that $f\phi = f\psi = id_{\rho\phi}$, $\Gamma; \Phi, \phi^k \vdash F : \mathcal{F}$, $\Gamma; \Phi, \overline{\alpha} \vdash G$ $\Gamma; \Phi, \alpha_1 \ldots \alpha_k \vdash H$, and $\Gamma; \Phi \vdash \tau$, then

$$[\![\Gamma; \Phi \vdash G[\overline{\alpha := \tau}]]\!]^{\mathrm{Set}} \rho = [\![\Gamma; \Phi, \overline{\alpha} \vdash G]\!]^{\mathrm{Set}} \rho[\overline{\alpha := [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Set}} \rho}] \tag{6}$$

$$[\![\Gamma; \Phi \vdash G[\overline{\alpha := \tau}]]\!]^{\mathrm{Set}} f = [\![\Gamma; \Phi, \overline{\alpha} \vdash G]\!]^{\mathrm{Set}} f[\overline{\alpha := [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Set}} f}] \tag{7}$$

$$[\![\Gamma; \Phi \vdash F[\phi := H]]\!]^{\mathrm{Set}} \rho = [\![\Gamma; \Phi, \phi \vdash F]\!]^{\mathrm{Set}} \rho[\phi := \lambda\overline{A}.\, [\![\Gamma; \Phi, \overline{\alpha} \vdash H]\!]^{\mathrm{Set}} \rho[\overline{\alpha := A}]] \tag{8}$$

$$[\![\Gamma; \Phi \vdash F[\phi := H]]\!]^{\mathrm{Set}} f = [\![\Gamma; \Phi, \phi \vdash F]\!]^{\mathrm{Set}} f[\phi := \lambda\overline{A}.\, [\![\Gamma; \Phi, \overline{\alpha} \vdash H]\!]^{\mathrm{Set}} f[\overline{\alpha := id_{\overline{A}}}]] \tag{9}$$

Identities analogous to (4) through (9) hold for relational interpretations as well.

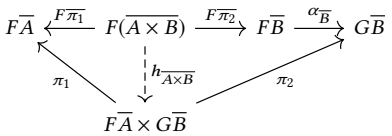
4 THE IDENTITY EXTENSION LEMMA

In most treatments of parametricity, equality relations on sets are given — either directly as diagonal relations, or perhaps via reflexive graphs if kinds are also being tracked — and the graph relations used to validate existence of initial algebras are defined in terms of them. We take a different approach here, giving a categorical definition of graph relations for morphisms (i.e., natural transformations) between functors and define equality relations as particular graph

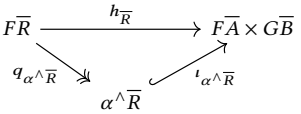
relations. Our definitions specialize to the usual ones for the graph relation for a morphism between sets and the equality relation for a set. In light of its novelty, we spell out our construction in detail.

The standard definition of the graph for a morphism $f : A \to B$ in Set is the relation $\langle f \rangle : \mathsf{Rel}(A, B)$ defined by $(x, y) \in \langle f \rangle$ iff $fx = y$. This definition naturally generalizes to associate to each natural transformation between k-ary functors on Set a k-ary relation transformer as follows:

Definition 19. *If $F, G : \mathsf{Set}^k \to \mathsf{Set}$ and $\alpha : F \to G$ is a natural transformation, then the functor $\langle \alpha \rangle^* : \mathsf{Rel}^k \to \mathsf{Rel}$ is defined as follows. Given $R_1 : \mathsf{Rel}(A_1, B_1), \ldots, R_k : \mathsf{Rel}(A_k, B_k)$, let $\iota_{R_i} : R_i \hookrightarrow A_i \times B_i$, for $i = 1, \ldots, k$, be the inclusion of R_i as a subset of $A_i \times B_i$, let $h_{\overline{A \times B}}$ be the unique morphism making the diagram*



commute, and let $h_{\overline{R}} : F\overline{R} \to F\overline{A} \times G\overline{B}$ be $h_{\overline{A \times B}} \circ F\overline{\iota_R}$. Further, let $\alpha^\wedge\overline{R}$ be the subobject through which $h_{\overline{R}}$ is factorized by the mono-epi factorization system in Set*, as shown in the following diagram:*

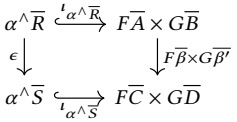


Then $\alpha^\wedge\overline{R} : \mathsf{Rel}(F\overline{A}, G\overline{B})$ by construction, so the action of $\langle \alpha \rangle^*$ on objects can be given by $\langle \alpha \rangle^*\overline{(A, B, R)} = (F\overline{A}, G\overline{B}, \iota_{\alpha^\wedge\overline{R}}\alpha^\wedge\overline{R})$. Its action on morphisms is given by $\langle \alpha \rangle^*\overline{(\beta, \beta')} = (F\overline{\beta}, G\overline{\beta}')$.

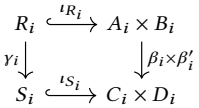
The data in Definition 19 yield the *graph relation transformer for α*, denoted $\langle \alpha \rangle = (F, G, \langle \alpha \rangle^*)$.

Lemma 20. *If $F, G : [\mathsf{Set}^k, \mathsf{Set}]$, and if $\alpha : F \to G$ is a natural transformation, then $\langle \alpha \rangle$ is in RT_k.*

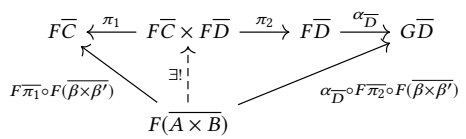
Proof. Clearly, $\langle \alpha \rangle^*$ is ω-cocontinuous, so $\langle \alpha \rangle^* : [\mathsf{Rel}^k, \mathsf{Rel}]$. Now, suppose $\overline{R : \mathsf{Rel}(A, B)}$, $\overline{S : \mathsf{Rel}(C, D)}$, and $\overline{(\beta, \beta') : R \to S}$. We want to show that there exists a morphism $\epsilon : \alpha^\wedge\overline{R} \to \alpha^\wedge\overline{S}$ such that



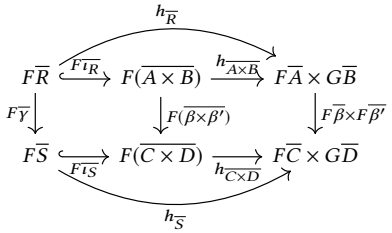
commutes. Since $\overline{(\beta, \beta') : R \to S}$, there exist $\overline{\gamma : R \to S}$ such that each diagram



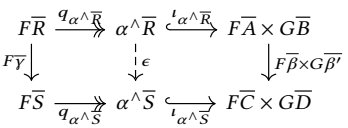
commutes. Moreover, since both $h_{\overline{C \times D}} \circ F(\overline{\beta \times \beta'})$ and $(F\overline{\beta} \times G\overline{\beta'}) \circ h_{\overline{A \times B}}$ make



commute, they must be equal. We therefore get that the right-hand square below commutes, and thus that the entire following diagram does as well:



Finally, by the left-lifting property of $q_{F^\wedge\overline{R}}$ with respect to $\iota_{F^\wedge\overline{S}}$ given by the epi-mono factorization system, there exists an ϵ such that the following diagram commutes:



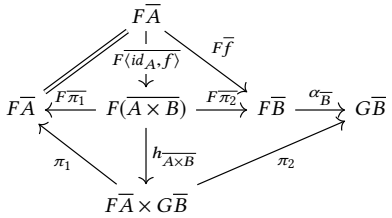
□

If $f : A \to B$ is a morphism in Set then the definition of the graph relation transformer $\langle f \rangle$ for f as a natural transformation between 0-ary functors A and B coincides with its standard definition. Graph relation transformers are thus a reasonable extension of graph relations to functors.

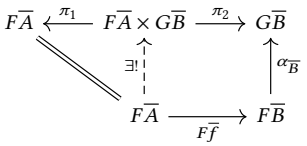
To prove the IEL, we will need to know that the equality relation transformer preserves equality relations; see Equation 10 below. This will follow from the next lemma, which shows how to compute the action of a graph relation transformer on any graph relation.

Lemma 21. *If $\alpha : F \to G$ is a morphism in $[\mathrm{Set}^k, \mathrm{Set}]$ and $f_1 : A_1 \to B_1, \ldots, f_k : A_k \to B_k$, then $\langle \alpha \rangle^* \langle \overline{f} \rangle = \langle G\overline{f} \circ \alpha_{\overline{A}} \rangle = \langle \alpha_{\overline{B}} \circ F\overline{f} \rangle$.*

Proof. Since $h_{\overline{A \times B}}$ is the unique morphism making the bottom triangle of this diagram commute



and since $h_{\overline{\langle f \rangle}} = h_{\overline{A \times B}} \circ F\,\overline{\iota_{\langle f \rangle}} = h_{\overline{A \times B}} \circ F\overline{\langle id_A, f \rangle}$, the universal property of the product



gives $h_{\overline{\langle f \rangle}} = \langle id_{F\overline{A}}, \alpha_{\overline{B}} \circ F\overline{f} \rangle : F\overline{A} \to F\overline{A} \times G\overline{B}$. Moreover, $\langle id_{F\overline{A}}, \alpha_{\overline{B}} \circ F\overline{f} \rangle$ is a monomorphism in Set because $id_{F\overline{A}}$ is, so its epi-mono factorization gives $\iota_{\alpha^\wedge \langle \overline{f} \rangle} = \langle id_{F\overline{A}}, \alpha_{\overline{B}} \circ F\overline{f} \rangle$, and thus $\alpha^\wedge \langle \overline{f} \rangle = F\overline{A}$. Then $\iota_{\alpha^\wedge \langle \overline{f} \rangle} \alpha^\wedge \langle \overline{f} \rangle = \langle id_{F\overline{A}}, \alpha_{\overline{B}} \circ F\overline{f} \rangle (F\overline{A}) = \langle \alpha_{\overline{B}} \circ F\overline{f} \rangle^*$, so that $\langle \alpha \rangle^* \langle \overline{f} \rangle = (F\overline{A}, G\overline{B}, \iota_{\alpha^\wedge \langle \overline{f} \rangle}\, \alpha^\wedge \langle \overline{f} \rangle) = (F\overline{A}, G\overline{B}, \langle \alpha_{\overline{B}} \circ F\overline{f} \rangle^*) = \langle \alpha_{\overline{B}} \circ F\overline{f} \rangle$. Finally, $\alpha_{\overline{B}} \circ F\overline{f} = G\overline{f} \circ \alpha_{\overline{A}}$ by naturality of α. □

The *equality relation transformer* on $F : [\mathsf{Set}^k, \mathsf{Set}]$ is defined to be $\mathsf{Eq}_F = \langle id_F \rangle$. Specifically, $Eq_F = (F, F, \mathsf{Eq}_F^*)$ with $\mathsf{Eq}_F^* = \langle id_F \rangle^*$, and Lemma 21 indeed ensures that

$$\mathsf{Eq}_F^* \overline{\mathsf{Eq}_A} = \langle id_F \rangle^* \langle id_{\overline{A}} \rangle = \langle F id_{\overline{A}} \circ (id_F)_{\overline{A}} \rangle = \langle id_{F\overline{A}} \circ id_{F\overline{A}} \rangle = \langle id_{F\overline{A}} \rangle = \mathsf{Eq}_{F\overline{A}} \tag{10}$$

for all $\overline{A : \mathsf{Set}}$. Graph relation transformers in general, and equality relation transformers in particular, extend to relation environments in the obvious ways. Indeed, if $\rho, \rho' : \mathsf{SetEnv}$ and $f : \rho \longrightarrow \rho'$, then the *graph relation environment* $\langle f \rangle$ is defined pointwise by $\langle f \rangle \phi = \langle f \phi \rangle$ for every ϕ. This entails that $\pi_1 \langle f \rangle = \rho$ and $\pi_2 \langle f \rangle = \rho'$. In particular, the *equality relation environment* Eq_ρ is defined to be $\langle id_\rho \rangle$. This entails that $\mathsf{Eq}_\rho \phi = \mathsf{Eq}_{\rho\phi}$ for every ϕ. With these definitions in hand, we can state and prove both an Identity Extension Lemma and a Graph Lemma for our calculus.

Theorem 22 (IEL). *If* $\rho : \mathsf{SetEnv}$ *and* $\Gamma; \Phi \vdash \tau : \mathcal{F}$ *then* $[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Rel}} \mathsf{Eq}_\rho = \mathsf{Eq}_{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho}$.

The proof is by induction on the structure of τ. Only the application and fixpoint cases are non-routine. Both use Lemma 21. The latter also uses the observation that, for every $n \in \mathbb{N}$, the following intermediate results can be proved by simultaneous induction with Theorem 22: $T^n_{\mathsf{Eq}_\rho} K_0 \, \overline{\mathsf{Eq}_{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho}} = (\mathsf{Eq}_{(T^{\mathsf{Set}}_\rho)^n K_0})^* \overline{\mathsf{Eq}_{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho}}$ and $[\![\Gamma; \Phi, \phi, \overline{\alpha} \vdash H]\!]^{\mathsf{Rel}} \mathsf{Eq}_\rho [\phi := T^n_{\mathsf{Eq}_\rho} K_0][\overline{\alpha := \mathsf{Eq}_{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho}}] = [\![\Gamma; \Phi, \phi, \overline{\alpha} \vdash H]\!]^{\mathsf{Rel}} \mathsf{Eq}_\rho [\phi := \mathsf{Eq}_{(T^{\mathsf{Set}}_\rho)^n K_0}][\overline{\alpha := \mathsf{Eq}_{[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho}}]$.

Lemma 23 (Graph Lemma). *If* $\rho, \rho' : \mathsf{SetEnv}$, $f : \rho \longrightarrow \rho'$, *and* $\Gamma; \Phi \vdash F : \mathcal{F}$, *then* $\langle [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \rangle = [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle$.

Proof. Applying Lemma 18 to the morphisms $(f, id_{\rho'}) : \langle f \rangle \longrightarrow \mathsf{Eq}_{\rho'}$ and $(id_\rho, f) : \mathsf{Eq}_\rho \longrightarrow \langle f \rangle$ of relation environments gives that $([\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f, [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} id_{\rho'}) = [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} (f, id_{\rho'}) : [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle \longrightarrow [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \mathsf{Eq}_{\rho'}$ and $([\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} id_\rho, [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f) = [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} (id_\rho, f) : [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \mathsf{Eq}_\rho \longrightarrow [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle$. Expanding the first equation gives that if $(x, y) \in [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle$ then $([\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \, x, [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} id_{\rho'} \, y) \in [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \mathsf{Eq}_{\rho'}$. Then $[\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} id_{\rho'} \, y = id_{[\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} \rho'} \, y = y$ and $[\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \mathsf{Eq}_{\rho'} = \mathsf{Eq}_{[\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} \rho'}$, so if $(x, y) \in [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle$ then $([\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \, x, y) \in \mathsf{Eq}_{[\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} \rho'}$, i.e., $[\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \, x = y$, i.e., $(x, y) \in \langle [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \rangle$. So, we have that $[\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle \subseteq \langle [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \rangle$. Expanding the second equation gives that if $x \in [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} \rho$ then $([\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} id_\rho \, x, [\![\Gamma; \Phi \vdash$ $[\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle$. Then $[\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} id_\rho \, x = id_{[\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} \rho} x = x$, so for any $x \in [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} \rho$ we have that $(x, [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \, x) \in [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle$. Moreover, $x \in [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} \rho$ if and only if $(x, [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \, x) \in \langle [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \rangle$ and, if $x \in [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} \rho$ then $(x, [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \, x) \in [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle$, so if $(x, [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \, x) \in \langle [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \rangle$ then $(x, [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \, x) \in [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle$, i.e., $\langle [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Set}} f \rangle \subseteq [\![\Gamma; \Phi \vdash F]\!]^{\mathsf{Rel}} \langle f \rangle$.

□

5 INTERPRETING TERMS

Double use of brackets confusing? (Graphs and pairing morphisms)

If $\Delta = x_1 : \tau_1, \ldots, x_n : \tau_n$ is a term context for Γ and Φ, then the interpretations $[\![\Gamma; \Phi \vdash \Delta]\!]^{\mathsf{Set}}$ and $[\![\Gamma; \Phi \vdash \Delta]\!]^{\mathsf{Rel}}$ are defined by

$$\begin{aligned} [\![\Gamma; \Phi \vdash \Delta]\!]^{\mathsf{Set}} &= [\![\Gamma; \Phi \vdash \tau_1]\!]^{\mathsf{Set}} \times \ldots \times [\![\Gamma; \Phi \vdash \tau_n]\!]^{\mathsf{Set}} \\ [\![\Gamma; \Phi \vdash \Delta]\!]^{\mathsf{Rel}} &= [\![\Gamma; \Phi \vdash \tau_1]\!]^{\mathsf{Rel}} \times \ldots \times [\![\Gamma; \Phi \vdash \tau_n]\!]^{\mathsf{Rel}} \end{aligned}$$

Every well-formed term $\Gamma; \Phi \mid \Delta \vdash t : \tau$ then has, for every $\rho \in \mathsf{SetEnv}$, set interpretations $[\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathsf{Set}} \rho$ as natural transformations from $[\![\Gamma; \Phi \vdash \Delta]\!]^{\mathsf{Set}} \rho$ to $[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Set}} \rho$, and, for every $\rho \in \mathsf{RelEnv}$, relational interpretations $[\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathsf{Rel}} \rho$ as natural transformations from $[\![\Gamma; \Phi \vdash \Delta]\!]^{\mathsf{Rel}} \rho$ to $[\![\Gamma; \Phi \vdash \tau]\!]^{\mathsf{Rel}} \rho$. These are given in the next (two) definitions.

Definition 24. *If ρ is a set (resp., relation) environment and $\Gamma;\Phi \mid \Delta \vdash t : \tau$ then $[\![\Gamma;\Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Set}}\rho$ (resp., $[\![\Gamma;\Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Rel}}\rho$) is defined as follows, where* D *is either* Set *or* Rel *as appropriate:*

$$
\begin{array}{lcl}
[\![\Gamma;\emptyset \mid \Delta, x:\tau \vdash x:\tau]\!]^{\mathsf{D}}\rho & = & \pi_{|\Delta|+1} \\
[\![\Gamma;\emptyset \mid \Delta \vdash L_{\overline{\alpha}}x.t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathsf{D}}\rho & = & \mathrm{curry}([\![\Gamma;\overline{\alpha} \mid \Delta, x:F \vdash t : G]\!]^{\mathsf{D}}\rho[\overline{\alpha := \_}]) \\
[\![\Gamma;\Phi \mid \Delta \vdash t_{\overline{\tau}}s : G[\overline{\alpha:=\tau}]]\!]^{\mathsf{D}}\rho & = & \mathrm{eval} \circ \langle \lambda d.\, (d)_{\overline{[\![\Gamma;\Phi\vdash\tau]\!]^{\mathsf{D}}}} \\
 & & \qquad [\![\Gamma;\Phi \mid \Delta \vdash s : F[\overline{\alpha:=\tau}]]\!]^{\mathsf{D}}\rho\rangle \\
 & & \\
[\![\Gamma;\Phi \mid \Delta, x:\tau \vdash x:\tau]\!]^{\mathsf{D}}\rho & = & \pi_{|\Delta|+1} \\
[\![\Gamma;\Phi \mid \Delta \vdash \bot_\tau t : \tau]\!]^{\mathsf{D}}\rho & = & !^0_{[\![\Gamma;\Phi\vdash\tau]\!]^{\mathsf{D}}\rho} \circ [\![\Gamma;\Phi \mid \Delta \vdash t : \mathbb{0}]\!]^{\mathsf{D}}\rho, \text{ where} \\
 & & \quad !^0_{[\![\Gamma;\Phi\vdash\tau]\!]^{\mathsf{D}}\rho} \text{ is the unique morphism from } 0 \\
 & & \quad \text{to } [\![\Gamma;\Phi \vdash \tau]\!]^{\mathsf{D}}\rho \\
[\![\Gamma;\Phi \mid \Delta \vdash \top : \mathbb{1}]\!]^{\mathsf{D}}\rho & = & !^{[\![\Gamma;\Phi\vdash\Delta]\!]^{\mathsf{D}}\rho}_1, \text{ where } !^{[\![\Gamma;\Phi\vdash\Delta]\!]^{\mathsf{D}}\rho}_1 \\
 & & \quad \text{is the unique morphism from } [\![\Gamma;\Phi \vdash \Delta]\!]^{\mathsf{D}}\rho \text{ to} \\
[\![\Gamma;\Phi \mid \Delta \vdash (s,t) : \sigma\times\tau]\!]^{\mathsf{D}}\rho & = & [\![\Gamma;\Phi \mid \Delta \vdash s : \sigma]\!]^{\mathsf{D}}\rho \times [\![\Gamma;\Phi \mid \Delta \vdash t : \tau]\!]^{\mathsf{D}}\rho \\
[\![\Gamma;\Phi \mid \Delta \vdash \pi_1 t : \sigma]\!]^{\mathsf{D}}\rho & = & \pi_1 \circ [\![\Gamma;\Phi \mid \Delta \vdash t : \sigma\times\tau]\!]^{\mathsf{D}}\rho \\
[\![\Gamma;\Phi \mid \Delta \vdash \pi_2 t : \sigma]\!]^{\mathsf{D}}\rho & = & \pi_2 \circ [\![\Gamma;\Phi \mid \Delta \vdash t : \sigma\times\tau]\!]^{\mathsf{D}}\rho \\
[\![\Gamma;\Phi \mid \Delta \vdash \mathrm{case}\, t \,\mathrm{of}\, \{x\mapsto l;\, y\mapsto r\} : \gamma]\!]^{\mathsf{D}}\rho & = & \mathrm{eval} \circ \langle \mathrm{curry}\, [[\![\Gamma;\Phi \mid \Delta, x:\sigma \vdash l : \gamma]\!]^{\mathsf{D}}\rho, \\
 & & \qquad [\![\Gamma;\Phi \mid \Delta, y:\tau \vdash r : \gamma]\!]^{\mathsf{D}}\rho], \\
 & & \quad [\![\Gamma;\Phi \mid \Delta \vdash t : \sigma+\tau]\!]^{\mathsf{D}}\rho\rangle \\
[\![\Gamma;\Phi \mid \Delta \vdash \mathrm{inl}\, s : \sigma+\tau]\!]^{\mathsf{D}}\rho & = & \mathrm{inl} \circ [\![\Gamma;\Phi \mid \Delta \vdash s : \sigma]\!]^{\mathsf{D}}\rho \\
[\![\Gamma;\Phi \mid \Delta \vdash \mathrm{inr}\, t : \sigma+\tau]\!]^{\mathsf{D}}\rho & = & \mathrm{inr} \circ [\![\Gamma;\Phi \mid \Delta \vdash t : \tau]\!]^{\mathsf{D}}\rho \\
[\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{map}^{\overline{F},\overline{G}}_H : \mathsf{Nat}^{\emptyset}\, (\overline{\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, F\, G}) & = & \lambda d\, \overline{\eta}\, \overline{B}.\, [\![\Gamma;\overline{\phi},\overline{\gamma} \vdash H]\!]^{\mathsf{D}}\, id_{\rho[\overline{\gamma:=B}]}[\overline{\phi := \lambda\overline{A}.\eta_{\overline{A}\,\overline{B}}}] \\
\quad (\mathsf{Nat}^{\overline{\gamma}}\, H[\overline{\phi:=_{\overline{\beta}} F}]\, H[\overline{\phi:=_{\overline{\beta}} G}])]\!]^{\mathrm{Set}}\rho & & \\
[\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{in}_H : Nat^{\overline{\beta},\overline{\gamma}}\, H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha:=\beta}] & = & \lambda d\, \overline{B}\, \overline{C}.\, (in_{T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]}})_{\overline{B}} \\
\quad (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathrm{Set}}\rho & & \\
[\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{fold}^F_H : \mathsf{Nat}^{\emptyset}\, (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha:=\beta}]\, F) & = & \lambda d\, \eta\, \overline{B}\, \overline{C}.\, (fold_{T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]}}\, (\lambda\overline{A}.\, \eta_{\overline{A}\,\overline{C}}))_{\overline{B}} \\
\quad (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\, F)]\!]^{\mathrm{Set}}\rho & &
\end{array}
$$

The return type for the semantic fold in the last clause is $[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{D}}\rho[\overline{\gamma:=C}]$. This interpretation gives that $[\![\Gamma;\emptyset \mid \Delta \vdash \lambda x.t : \sigma\to\tau]\!]^{\mathsf{D}}\rho = \mathrm{curry}([\![\Gamma;\emptyset \mid \Delta, x:\sigma \vdash t : \tau]\!]^{\mathsf{D}}\rho)$ and $[\![\Gamma;\emptyset \mid \Delta \vdash st : \tau]\!]^{\mathsf{D}}\rho = \mathrm{eval}\circ\langle [\![\Gamma;\emptyset \mid \Delta \vdash s : \sigma\to\tau]\!]^{\mathsf{D}}\rho, [\![\Gamma;\emptyset \mid \Delta \vdash t : \sigma]\!]^{\mathsf{D}}\rho\rangle$, so it specializes to the standard interpretations for System F terms. If t is closed, i.e., if $\emptyset;\emptyset \mid \emptyset \vdash t : \tau$, then we write $[\![\vdash t : \tau]\!]^{\mathsf{D}}$ instead of $[\![\emptyset;\emptyset \mid \emptyset \vdash t : \tau]\!]^{\mathsf{D}}$.

5.1 Basic Properties of Term Interpretations

The interpretations in Definition 24 respect weakening, i.e., a term and its weakenings all have the same set and relational interpretations. In particular, for any $\rho \in \mathrm{SetEnv}$,

$$[\![\Gamma;\Phi \mid \Delta, x:\sigma \vdash t : \tau]\!]^{\mathrm{Set}}\rho = ([\![\Gamma;\Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Set}}\rho) \circ \pi_\Delta$$

where π_Δ is the projection $[\![\Gamma;\Phi \vdash \Delta, x:\sigma]\!]^{\mathrm{Set}} \to [\![\Gamma;\Phi \vdash \Delta]\!]^{\mathrm{Set}}$, and similarly for relational interpretations. Moreover, if $\Gamma,\alpha;\Phi \mid \Delta \vdash t : \tau$ and $\Gamma;\Phi,\alpha \mid \Delta \vdash t' : \tau$ and $\Gamma;\Phi \vdash \sigma : \mathcal{F}$ then

- $[\![\Gamma;\Phi \mid \Delta[\alpha:=\sigma] \vdash t[\alpha:=\sigma] : \tau[\alpha:=\sigma]]\!]^{\mathrm{Set}}\rho = [\![\Gamma,\alpha;\Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Set}}\rho[\alpha := [\![\Gamma;\Phi \vdash \sigma]\!]^{\mathrm{Set}}\rho]$
- $[\![\Gamma;\Phi \mid \Delta[\alpha:=\sigma] \vdash t'[\alpha:=\sigma] : \tau[\alpha:=\sigma]]\!]^{\mathrm{Set}}\rho = [\![\Gamma;\Phi,\alpha \mid \Delta \vdash t' : \tau]\!]^{\mathrm{Set}}\rho[\alpha := [\![\Gamma;\Phi \vdash \sigma]\!]^{\mathrm{Set}}\rho]$

and if $\Gamma;\Phi \mid \Delta, x:\sigma \vdash t : \tau$ and $\Gamma;\Phi \mid \Delta \vdash s : \sigma$ then

- $\lambda A.\, [\![\Gamma; \Phi \mid \Delta \vdash t[x := s] : \tau]\!]^{\mathrm{Set}} \rho\, A = \lambda A.\, [\![\Gamma; \Phi \mid \Delta, x : \sigma \vdash t : \tau]\!]^{\mathrm{Set}} \rho\, (A, [\![\Gamma; \Phi \mid \Delta \vdash s : \sigma]\!]^{\mathrm{Set}} \rho\, A)$

Direct calculation reveals that the set interpretations of terms also satisfy

- $[\![\Gamma; \Phi \mid \Delta \vdash (L_{\overline{\alpha}} x.t)_{\overline{\tau}} s]\!]^{\mathrm{Set}} = [\![\Gamma; \Phi \mid \Delta \vdash t[\overline{\alpha := \tau}][x := s]]\!]^{\mathrm{Set}}$

Term extensionality with respect to types and terms — i.e., $[\![\Gamma; \Phi \vdash (L_\alpha x.t)_\alpha \top]\!]^{\mathrm{Set}} = [\![\Gamma; \Phi \vdash t]\!]^{\mathrm{Set}}$ and $[\![\Gamma; \Phi \vdash (L_\alpha x.t)_\alpha x]\!]^{\mathrm{Set}} = [\![\Gamma; \Phi \vdash t]\!]^{\mathrm{Set}}$ — follow. Similar properties hold for relational interpretations.

5.2 Properties of Terms of Nat-Type

Define, for $\Gamma; \overline{\alpha} \vdash F$, the term id_F to be $\Gamma; \emptyset \mid \emptyset \vdash L_{\overline{\alpha}} x.x : \mathrm{Nat}^{\overline{\alpha}} F\, F$ and, for terms $\Gamma; \emptyset \mid \Delta \vdash t : \mathrm{Nat}^{\overline{\alpha}} F\, G$ and $\Gamma; \emptyset \mid \Delta \vdash s : \mathrm{Nat}^{\overline{\alpha}} G\, H$, the *composition* $s \circ t$ of t and s to be $\Gamma; \emptyset \mid \Delta \vdash L_{\overline{\alpha}} x.s_{\overline{\alpha}}(t_{\overline{\alpha}} x) : \mathrm{Nat}^{\overline{\alpha}} F\, H$. Then $[\![\Gamma; \emptyset \mid \emptyset \vdash id_F : \mathrm{Nat}^{\overline{\alpha}} F\, F]\!]^{\mathrm{Set}} \rho * = id_{\lambda \overline{A}.[\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathrm{Set}} \rho[\overline{\alpha := A}]}$ for any set environment ρ, and $[\![\Gamma; \emptyset \mid \Delta \vdash s \circ t : \mathrm{Nat}^{\overline{\alpha}} F\, H]\!]^{\mathrm{Set}} = [\![\Gamma; \emptyset \mid \Delta \vdash s : \mathrm{Nat}^{\overline{\alpha}} G\, H]\!]^{\mathrm{Set}} \circ [\![\Gamma; \emptyset \mid \Delta \vdash t : \mathrm{Nat}^{\overline{\alpha}} F\, G]\!]^{\mathrm{Set}}$. Straightforward calculation using these equalities shows that terms of Nat type behave as natural transformations with respect to their source and target functorial types, i.e.,

Theorem 25.

$$\begin{aligned}
&[\![\Gamma; \emptyset \mid x : \mathrm{Nat}^{\overline{\alpha}, \overline{\gamma}} F\, G, \overline{y : \mathrm{Nat}^{\overline{\gamma}} \sigma\, \tau} \vdash ((\mathrm{map}_G^{\overline{\sigma}, \overline{\tau}})_\emptyset \overline{y}) \circ (L_{\overline{\gamma}} z.x_{\overline{\sigma}, \overline{\gamma}} z) : \mathrm{Nat}^{\overline{\gamma}} F[\overline{\alpha := \sigma}]\, G[\overline{\alpha := \tau}]]\!]^{\mathrm{Set}} \\
&= [\![\Gamma; \emptyset \mid x : \mathrm{Nat}^{\overline{\alpha}, \overline{\gamma}} F\, G, \overline{y : \mathrm{Nat}^{\overline{\gamma}} \sigma\, \tau} \vdash (L_{\overline{\gamma}} z.x_{\overline{\tau}, \overline{\gamma}} z) \circ ((\mathrm{map}_F^{\overline{\sigma}, \overline{\tau}})_\emptyset \overline{y}) : \mathrm{Nat}^{\overline{\gamma}} F[\overline{\alpha := \sigma}]\, G[\overline{\alpha := \tau}]]\!]^{\mathrm{Set}}
\end{aligned}$$

When $x = in_H$ and $\xi = \mathrm{Nat}^{\overline{\gamma}} H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \sigma}]\, (\mu\phi.\lambda\overline{\alpha}.H)\overline{\tau}$ Theorem 25 specializes to

$$\begin{aligned}
&[\![\Gamma; \emptyset \mid \overline{y : \mathrm{Nat}^{\overline{\gamma}} \sigma\, \tau} \vdash ((\mathrm{map}^{\overline{\sigma}, \overline{\tau}}_{(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}})_\emptyset \overline{y}) \circ (L_{\overline{\gamma}} z.(\mathrm{in}_H)_{\overline{\sigma}, \overline{\gamma}} z) : \xi]\!]^{\mathrm{Set}} \\
&= [\![\Gamma; \emptyset \mid \overline{y : \mathrm{Nat}^{\overline{\gamma}} \sigma\, \tau} \vdash (L_{\overline{\gamma}} z.(\mathrm{in}_H)_{\overline{\tau}, \overline{\gamma}} z) \circ ((\mathrm{map}^{\overline{\sigma}, \overline{\tau}}_{H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]})_\emptyset \overline{y}) : \xi]\!]^{\mathrm{Set}}
\end{aligned} \tag{11}$$

Theorem 25 gives a family of free theorems that are consequences of naturality, and thus do not require the full power of parametricity. Other specific instances include the three `map`/`reverse` theorems from Section 1. In our setting, the `map`/`reverse` theorem for bushes, e.g., becomes

$$\begin{aligned}
&[\![\emptyset; \alpha \mid y : \mathrm{Nat}^{\emptyset} \sigma\, \tau \vdash ((\mathrm{map}^{\sigma, \tau}_{Bush\, \alpha})_\emptyset y) \circ (L_\emptyset z.reverseBush_\sigma z) : \mathrm{Nat}^{\emptyset} (Bush\, \sigma)\, (Bush\, \tau)]\!]^{\mathrm{Set}} \\
&= [\![\emptyset; \alpha \mid y : \mathrm{Nat}^{\emptyset} \sigma\, \tau \vdash (L_\emptyset z.reverseBush_\tau z) \circ ((\mathrm{map}^{\sigma, \tau}_{Bush\, \alpha})_\emptyset y) : \mathrm{Nat}^{\emptyset} (Bush\, \sigma)\, (Bush\, \tau)]\!]^{\mathrm{Set}}
\end{aligned}$$

for any term $\vdash reverseBush : \mathrm{Nat}^{\alpha} (Bush\, \alpha)\, (Bush\, \alpha)$. We can also prove a theorem for the data type $\emptyset; \alpha, \gamma \vdash Tree\, \alpha\, \gamma$ from Section 2 that is similar but only maps along some of its functorial variables: if $\vdash reverseTree : \mathrm{Nat}^{\alpha, \gamma} (Tree\, \alpha\, \gamma)\, (Tree\, \alpha\, \gamma)$ then

$$\begin{aligned}
&[\![\emptyset; \alpha, \gamma \mid y : \mathrm{Nat}^{\emptyset} \sigma\, \tau \vdash ((\mathrm{map}^{\sigma, \tau}_{Tree\, \alpha\, \gamma})_\emptyset y) \circ (L_\gamma z.reverseTree_{\sigma, \gamma} z) : \mathrm{Nat}^{\gamma} (Tree\, \sigma\, \gamma)\, (Tree\, \tau\, \gamma)]\!]^{\mathrm{Set}} \\
&= [\![\emptyset; \alpha, \gamma \mid y : \mathrm{Nat}^{\emptyset} \sigma\, \tau \vdash (L_\gamma z.reverseTree_{\tau, \gamma} z) \circ ((\mathrm{map}^{\sigma, \tau}_{Tree\, \alpha\, \gamma})_\emptyset y) : \mathrm{Nat}^{\gamma} (Tree\, \sigma\, \gamma)\, (Tree\, \tau\, \gamma)]\!]^{\mathrm{Set}}
\end{aligned}$$

Finally, in Section 2 we observed that $\emptyset; \phi, \alpha \vdash GRose\, \phi\, \alpha$ cannot appear as the (co)domain in a Nat type. As a result, there are no instances of, say, a `map`/`reverse` free theorem for this data type. But $\phi; \alpha \vdash GRose\, \phi\, \alpha$ does support a `map`/`reverse` theorem similar to those from Section 1.

As usual, analogous reuslts hold for relation environments and relational interpretations.

5.3 Properties of Initial Algebraic Constructs

Our semantics interprets map terms as semantic *map* functions. Indeed, if $\Gamma; \overline{\phi}, \overline{\gamma} \vdash H : \mathcal{F}$, $\overline{\Gamma; \overline{\beta}, \overline{\gamma} \vdash F : \mathcal{F}}$ and $\overline{\Gamma; \overline{\beta}, \overline{\gamma} \vdash G : \mathcal{F}}$, then Definition 24 gives that

$$[\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{map}_H^{\overline{F},\overline{G}} : \mathsf{Nat}^{\emptyset}\ (\overline{\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\ F\ G})\ (\mathsf{Nat}^{\overline{\gamma}}\ H[\overline{\phi :=_{\overline{\beta}} F}]\ H[\overline{\phi :=_{\overline{\beta}} G}])]\!]^{\mathsf{Set}}\rho$$
$$= \lambda d\, \overline{\eta}\, \overline{B}.\ [\![\Gamma; \overline{\phi}, \overline{\gamma} \vdash H]\!]^{\mathsf{Set}} id_{\rho[\overline{\gamma:=B}]}[\overline{\phi := \lambda \overline{A}.\eta_{\overline{A}\,\overline{B}}}] \quad (12)$$

In particular, if $\Gamma; \overline{\alpha} \vdash H : \mathcal{F}$, $\overline{\Gamma; \emptyset \vdash \sigma : \mathcal{F}}$, $\overline{\Gamma; \emptyset \vdash \tau : \mathcal{F}}$, and $*$ is the unique element of $[\![\Gamma; \emptyset \vdash \emptyset]\!]^{\mathsf{Set}}\rho$, then

$$\begin{aligned}
& [\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{map}_H^{\overline{\sigma},\overline{\tau}} : \mathsf{Nat}^{\emptyset}\ (\overline{\mathsf{Nat}^{\emptyset}\ \sigma\ \tau})\ (\mathsf{Nat}^{\emptyset}\ H[\overline{\alpha := \sigma}]\ H[\overline{\alpha := \tau}])]\!]^{\mathsf{Set}}\rho\ * \\
= \quad & \overline{\lambda f : [\![\Gamma; \emptyset \vdash \sigma]\!]^{\mathsf{Set}}\rho \to [\![\Gamma; \emptyset \vdash \tau]\!]^{\mathsf{Set}}\rho.\ [\![\Gamma; \overline{\alpha} \vdash H]\!]^{\mathsf{Set}} id_\rho[\overline{\alpha := f}]} \\
= \quad & \overline{\lambda f : [\![\Gamma; \emptyset \vdash \sigma]\!]^{\mathsf{Set}}\rho \to [\![\Gamma; \emptyset \vdash \tau]\!]^{\mathsf{Set}}\rho.\ map_{\lambda \overline{A}.\ [\![\Gamma; \overline{\alpha} \vdash H]\!]^{\mathsf{Set}}\rho[\overline{\alpha:=A}]}\overline{f}} \\
= \quad & map_{\lambda \overline{A}.\ [\![\Gamma; \overline{\alpha} \vdash H]\!]^{\mathsf{Set}}\rho[\overline{\alpha:=A}]}
\end{aligned}$$

Here, the first equality is by Equation 12, the second is obtained by noting that $\lambda \overline{A}.\ [\![\Gamma; \overline{\alpha} \vdash H]\!]^{\mathsf{Set}}\rho[\overline{\alpha := A}]$ is a functor in α, and map_G denotes the action of the semantic functor G on morphisms.

We also have the expected relationships between the interpetations of map, in, and fold:

- If $\Gamma; \overline{\psi}, \overline{\gamma} \vdash H$, $\overline{\Gamma; \overline{\alpha}, \overline{\gamma}, \overline{\phi} \vdash K}$, $\overline{\Gamma; \overline{\beta}, \overline{\gamma} \vdash F}$, and $\overline{\Gamma; \overline{\beta}, \overline{\gamma} \vdash G}$, then

$$[\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{map}_{H[\overline{\psi:=K}]}^{\overline{F},\overline{G}} : \xi]\!]^{\mathsf{Set}} = [\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{map}_H^{\overline{K[\overline{\phi:=F}]},\overline{K[\overline{\phi:=G}]}} \circ \overline{\mathsf{map}_K^{\overline{F},\overline{G}}} : \xi]\!]^{\mathsf{Set}}$$

for $\xi = \mathsf{Nat}^{\emptyset}(\overline{\mathsf{Nat}^{\overline{\alpha},\overline{\beta},\overline{\gamma}}\ F\ G})(\mathsf{Nat}^{\overline{\gamma}}H[\overline{\psi := K}][\overline{\phi := F}]H[\overline{\psi := K}][\overline{\phi := G}])$

- If $\Gamma; \overline{\beta}, \overline{\gamma} \vdash H$, $\Gamma; \overline{\beta}, \overline{\gamma} \vdash K$, $\overline{\Gamma; \overline{\alpha}, \overline{\gamma} \vdash F}$, $\overline{\Gamma; \overline{\alpha}, \overline{\gamma} \vdash G}$, $\overline{\Gamma; \phi, \overline{\psi}, \overline{\gamma} \vdash \tau}$, $\overline{I}$ is the sequence $\overline{F}, H$ and $\overline{J}$ is the sequence $\overline{G}, K$ then

$$[\![\Gamma; \emptyset \mid \emptyset \vdash L_\emptyset(x, \overline{y}).L_{\overline{\gamma}}z.x_{\overline{\tau[\overline{\psi:=G}][\phi:=K]},\overline{\gamma}}\Big(\big((\mathsf{map}_H^{\overline{\tau[\overline{\psi:=F}][\phi:=H]},\overline{\tau[\overline{\psi:=G}][\phi:=K]}})_\emptyset(\overline{(\mathsf{map}_\tau^{\overline{I},\overline{J}})_\emptyset(x,\overline{y})})\big)_{\overline{\gamma}}z\Big) : \xi]\!]^{\mathsf{Set}}$$

$$= [\![\Gamma; \emptyset \mid \emptyset \vdash L_\emptyset(x, \overline{y}).L_{\overline{\gamma}}z.\big((\mathsf{map}_K^{\overline{\tau[\overline{\psi:=F}][\phi:=H]},\overline{\tau[\overline{\psi:=G}][\phi:=K]}})_\emptyset(\overline{(\mathsf{map}_\tau^{\overline{I},\overline{J}})_\emptyset(x,\overline{y})})\big)_{\overline{\gamma}}\Big(x_{\overline{\tau[\overline{\psi:=F}][\phi:=H]},\overline{\gamma}}z\Big) : \xi]\!]^{\mathsf{Set}}$$

for $\xi = \mathsf{Nat}^{\emptyset}(\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}H\,K \times \overline{\mathsf{Nat}^{\overline{\alpha},\overline{\gamma}}F\,G})\,(\mathsf{Nat}^{\overline{\gamma}}H[\overline{\beta := \tau}][\overline{\psi := F}][\phi := H]\,K[\overline{\beta := \tau}][\overline{\psi := G}][\phi := K])$.

- If $\xi = \mathsf{Nat}^{\overline{\beta},\overline{\gamma}}H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\,F$, then

$$[\![\Gamma; \emptyset \mid x : \mathsf{Nat}^{\overline{\beta},\overline{\gamma}}H[\phi := F][\overline{\alpha := \beta}]\,F \vdash ((\mathsf{fold}_{H,F})_\emptyset x) \circ \mathsf{in}_H : \xi]\!]^{\mathsf{Set}}$$
$$= [\![\Gamma; \emptyset \mid x : \mathsf{Nat}^{\overline{\beta},\overline{\gamma}}H[\phi := F][\overline{\alpha := \beta}]\,F \vdash x \circ \big((\mathsf{map}_{H[\overline{\alpha:=\beta}]}^{(\mu\phi.\lambda\overline{\alpha}H)\overline{\beta},F})_\emptyset((\mathsf{fold}_{H,F})_\emptyset x)\big) : \xi]\!]^{\mathsf{Set}}$$

- If $\xi = \mathsf{Nat}^{\overline{\beta},\overline{\gamma}}(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\,(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}$, then

$$[\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{in}_H \circ (\mathsf{fold}_{H,H[\phi:=(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]})_\emptyset((\mathsf{map}_H^{H[\phi:=(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha:=\beta}],(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}})_\emptyset \mathsf{in}_H) : \xi]\!]^{\mathsf{Set}}$$
$$= [\![\Gamma; \emptyset \mid \emptyset \vdash Id_{(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}} : \xi]\!]^{\mathsf{Set}}$$

- If $\xi = \mathsf{Nat}^{\overline{\beta},\overline{\gamma}}H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\,H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]$, then

$$[\![\Gamma; \emptyset \mid \emptyset \vdash (\mathsf{fold}_{H,H[\phi:=(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]})_\emptyset((\mathsf{map}_H^{H[\phi:=(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha:=\beta}],(\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}})_\emptyset \mathsf{in}_H) \circ \mathsf{in}_H : \xi]\!]^{\mathsf{Set}}$$

$= [\![\Gamma; \emptyset \mid \emptyset \vdash Id_{H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]} : \xi]\!]^{\mathrm{Set}}$

Analogous results hold for relational interpretations of terms and relational environments.

5.4 The Abstraction Theorem

To go beyond naturality and get *all* consequences of parametricity, we prove an Abstraction Theorem for our calculus. As usual for such theorems, we prove a more general result in Theorem 26 that handles possibly open terms. We then recover the Abstraction Theorem (Theorem 28) as the special case Theorem 26 for closed terms of closed type.

Theorem 26. *Every well-formed term* $\Gamma; \Phi \mid \Delta \vdash t : \tau$ *induces a natural transformation from* $[\![\Gamma; \Phi \vdash \Delta]\!]$ *to* $[\![\Gamma; \Phi \vdash \tau]\!]$*, i.e., a triple of natural transformations*

$$([\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Set}}, [\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Set}}, [\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Rel}})$$

where

$$[\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Set}} \quad : \quad [\![\Gamma; \Phi \vdash \Delta]\!]^{\mathrm{Set}} \to [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Set}}$$

has as its component at $\rho : \mathsf{SetEnv}$ *a morphism*

$$[\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Set}}\rho \quad : \quad [\![\Gamma; \Phi \vdash \Delta]\!]^{\mathrm{Set}}\rho \to [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Set}}\rho$$

in Set*,*

$$[\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Rel}} \quad : \quad [\![\Gamma; \Phi \vdash \Delta]\!]^{\mathrm{Rel}} \to [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}}$$

has as its component at $\rho : \mathsf{RelEnv}$ *a morphism*

$$[\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Rel}}\rho \quad : \quad [\![\Gamma; \Phi \vdash \Delta]\!]^{\mathrm{Rel}}\rho \to [\![\Gamma; \Phi \vdash \tau]\!]^{\mathrm{Rel}}\rho$$

in Rel*, and, for all* $\rho : \mathsf{RelEnv}$*,*

$$[\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Rel}}\rho = ([\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Set}}(\pi_1\rho), [\![\Gamma; \Phi \mid \Delta \vdash t : \tau]\!]^{\mathrm{Set}}(\pi_2\rho)) \tag{13}$$

Proof. By structural induction on the type judgment for t. The only interesting cases are the cases for abstraction, application, map, in, and fold so we omit the others. The challenging lies in showing that the set and relational interpretations of each type judgment are natural transformations (also satisfying the appropriate equality preservation condition in the case of the set interpretations). We give proofs for the set interpretations only; those for the relational interpretations are similar. The proof that Equation 13 holds in each case is by direct calculation, using the facts that projections are surjective and that the set and relational interpretations are defined "in parallel", i.e., are fibred.

- $\Gamma; \emptyset \mid \Delta \vdash L_{\overline{\alpha}}x.t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G$ Since Φ is empty, to see that $[\![\Gamma; \emptyset \mid \Delta \vdash L_{\overline{\alpha}}x.t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathrm{Set}}$ is a natural transformation from $[\![\Gamma; \emptyset \vdash \Delta]\!]^{\mathrm{Set}}$ to $[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathrm{Set}}$ we need only show that, for every $\rho : \mathsf{SetEnv}$, $[\![\Gamma; \emptyset \mid \Delta \vdash L_{\overline{\alpha}}x.t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathrm{Set}}\rho$ is a morphism in Set from $[\![\Gamma; \emptyset \vdash \Delta]\!]^{\mathrm{Set}}\rho$ to $[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathrm{Set}}\rho$, i.e., that, for each $\overline{A : \mathrm{Set}}$ and each $d : [\![\Gamma; \emptyset \vdash \Delta]\!]^{\mathrm{Set}}\rho = [\![\Gamma; \overline{\alpha} \vdash \Delta]\!]^{\mathrm{Set}}\rho[\overline{\alpha := A}]$, we have $([\![\Gamma; \emptyset \mid \Delta \vdash L_{\overline{\alpha}}x.t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathrm{Set}}\rho\, d)_{\overline{A}} : [\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\alpha := A}] \to [\![\Gamma; \overline{\alpha} \vdash G]\!]^{\mathrm{Set}}\rho[\overline{\alpha := A}]$. But this follows easily from the induction hypothesis. That these maps comprise a natural transformation $\eta : [\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\alpha := \_}] \to [\![\Gamma; \overline{\alpha} \vdash G]\!]^{\mathrm{Set}}\rho[\overline{\alpha := \_}]$ is clear since $\eta_{\overline{A}} = \mathsf{curry}([\![\Gamma; \overline{\alpha} \mid \Delta, x : F \vdash t : G]\!]^{\mathrm{Set}}\rho[\overline{\alpha := A}])\, d$ is the component at $\overline{A}$ of the partial specialization to d of the natural transformation $[\![\Gamma; \overline{\alpha} \mid \Delta, x : F \vdash t : G]\!]^{\mathrm{Set}}\rho[\overline{\alpha := \_}]$. To see that the components of η also satisfy the additional condition needed for η to be in $[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G]\!]^{\mathrm{Set}}\rho$, let $\overline{R : \mathsf{Rel}(A, B)}$ and suppose $(u, v) \in [\![\Gamma; \overline{\alpha} \vdash F]\!]^{\mathrm{Rel}}\mathsf{Eq}_\rho[\overline{\alpha := R}] =$

$(\llbracket \Gamma; \overline{\alpha} \vdash F \rrbracket^{\mathsf{Set}} \rho[\overline{\alpha := A}], \llbracket \Gamma; \overline{\alpha} \vdash F \rrbracket^{\mathsf{Set}} \rho[\overline{\alpha := B}])$. Then the induction hypothesis and $(d, d) \in \llbracket \Gamma; \emptyset \vdash \Delta \rrbracket^{\mathsf{Rel}} \mathsf{Eq}_\rho = \llbracket \Gamma; \emptyset \vdash \Delta \rrbracket^{\mathsf{Rel}} \mathsf{Eq}_\rho[\overline{\alpha := R}]$ ensure that

$$
\begin{array}{ll}
 & (\eta_{\overline{A}} u, \eta_{\overline{B}} v) \\
= & (\mathsf{curry}\,(\llbracket \Gamma; \overline{\alpha} \mid \Delta, x : F \vdash t : G \rrbracket^{\mathsf{Set}} \rho[\overline{\alpha := A}])\, d\, u, \mathsf{curry}\,(\llbracket \Gamma; \overline{\alpha} \mid \Delta, x : F \vdash t : G \rrbracket^{\mathsf{Set}} \rho[\overline{\alpha := B}])\, d\, v) \\
= & \mathsf{curry}\,(\llbracket \Gamma; \overline{\alpha} \mid \Delta, x : F \vdash t : G \rrbracket^{\mathsf{Rel}} \mathsf{Eq}_\rho[\overline{\alpha := R}])\, (d, d)\, (u, v) \\
: & \llbracket \Gamma; \overline{\alpha} \vdash G \rrbracket^{\mathsf{Rel}} \mathsf{Eq}_\rho[\overline{\alpha := R}]
\end{array}
$$

- $\Gamma; \Phi \mid \Delta \vdash t_{\overline{\tau}} s : G[\overline{\alpha := \tau}]$ To see that $\llbracket \Gamma; \Phi \mid \Delta \vdash t_{\overline{\tau}} s : G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}}$ is a natural transformation from $\llbracket \Gamma; \Phi \vdash \Delta \rrbracket^{\mathsf{Set}}$ to $\llbracket \Gamma; \Phi \vdash G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}}$ we must show that, for every $\rho : \mathsf{SetEnv}$, $\llbracket \Gamma; \Phi \mid \Delta \vdash t_{\overline{\tau}} s : G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho$ is a morphism from $\llbracket \Gamma; \Phi \vdash \Delta \rrbracket^{\mathsf{Set}} \rho$ to $\llbracket \Gamma; \Phi \vdash G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho$, and that this family of morphisms is natural in ρ. Let $d : \llbracket \Gamma; \Phi \vdash \Delta \rrbracket^{\mathsf{Set}} \rho$. Then

$$
\begin{array}{ll}
 & \llbracket \Gamma; \Phi \mid \Delta \vdash t_{\overline{\tau}} s : G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}}\, \rho\, d \\
= & (\mathsf{eval} \circ \langle (\llbracket \Gamma; \emptyset \mid \Delta \vdash t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} \rho\ \_)_{\overline{\llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}}, \llbracket \Gamma; \Phi \mid \Delta \vdash s : F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho \rangle)\, d \\
= & \mathsf{eval}((\llbracket \Gamma; \emptyset \mid \Delta \vdash t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} \rho\ \_)_{\overline{\llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}}\, d, \llbracket \Gamma; \Phi \mid \Delta \vdash s : F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho\, d) \\
= & \mathsf{eval}((\llbracket \Gamma; \emptyset \mid \Delta \vdash t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} \rho\ d)_{\overline{\llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}}, \llbracket \Gamma; \Phi \mid \Delta \vdash s : F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho\, d)
\end{array}
$$

The induction hypothesis ensures that $(\llbracket \Gamma; \emptyset \mid \Delta \vdash t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} \rho\ \ d)_{\overline{\llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}}$ has type $\llbracket \Gamma; \overline{\alpha} \vdash F \rrbracket^{\mathsf{Set}} \rho[\overline{\alpha := \llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}] \to \llbracket \Gamma; \overline{\alpha} \vdash G \rrbracket^{\mathsf{Set}} \rho[\overline{\alpha := \llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}]$. Since, in addition, $\llbracket \Gamma; \Phi \mid \Delta \vdash s : F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho\, d : \llbracket \Gamma; \Phi \vdash F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho = \llbracket \Gamma; \Phi, \overline{\alpha} \vdash F \rrbracket^{\mathsf{Set}} \rho[\overline{\alpha := \llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}] = \llbracket \Gamma; \overline{\alpha} \vdash F \rrbracket^{\mathsf{Set}} \rho[\overline{\alpha := \llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}]$ by Equation 6, we have that $\llbracket \Gamma; \Phi \mid \Delta \vdash t_{\overline{\tau}} s : G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}}\, \rho\, d : \llbracket \Gamma; \overline{\alpha} \vdash G \rrbracket^{\mathsf{Set}} \rho[\overline{\alpha := \llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}] = \llbracket \Gamma; \Phi \vdash G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho$, as desired.

To see that the family of maps comprising $\llbracket \Gamma; \Phi \mid \Delta \vdash t_{\overline{\tau}} s : G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}}$ is natural in ρ we need to show that, if $f : \rho \to \rho'$ in SetEnv, then the following diagram commutes, where $g = \llbracket \Gamma; \emptyset \mid \Delta \vdash t : \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}}$ and $h = \llbracket \Gamma; \Phi \mid \Delta \vdash s : F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}}$:

$$
\begin{array}{ccc}
\llbracket \Gamma; \Phi \vdash \Delta \rrbracket^{\mathsf{Set}} \rho & \xrightarrow{\llbracket \Gamma; \Phi \vdash \Delta \rrbracket^{\mathsf{Set}} f} & \llbracket \Gamma; \Phi \vdash \Delta \rrbracket^{\mathsf{Set}} \rho' \\
\downarrow{\scriptstyle \langle g\rho, h\rho \rangle} & \llbracket \Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} f \times \llbracket \Gamma; \Phi \vdash F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} f & \downarrow{\scriptstyle \langle g\rho', h\rho' \rangle} \\
\llbracket \Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} \rho \times \llbracket \Gamma; \Phi \vdash F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho & \longrightarrow & \llbracket \Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} \rho' \times \llbracket \Gamma; \Phi \vdash F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho' \\
\downarrow{\scriptstyle \mathsf{eval} \circ ((-)_{\overline{\llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}} \times id)} & & \downarrow{\scriptstyle \mathsf{eval} \circ ((-)_{\overline{\llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho'}} \times id)} \\
\llbracket \Gamma; \Phi \vdash G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho & \xrightarrow[\llbracket \Gamma; \Phi \vdash G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} f]{} & \llbracket \Gamma; \Phi \vdash G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho'
\end{array}
$$

The top diagram commutes because g and h are natural in ρ by the induction hypothesis. To see that the bottom diagram commutes, we first note that since $\rho|_\Gamma = \rho'|_\Gamma$, $\Gamma; \overline{\alpha} \vdash F : \mathcal{F}$, and $\Gamma; \overline{\alpha} \vdash G : \mathcal{F}$, we can replace the instance of f in $\llbracket \Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} f$ with id. Then, since functoriality ensures that $\llbracket \Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} id = id$, we need only show that, for all $\eta \in \llbracket \Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} \rho$ and $x \in \llbracket \Gamma; \Phi \vdash F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho$, $\llbracket \Gamma; \Phi \vdash G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} f(\eta_{\overline{\llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}} x) = \eta_{\overline{\llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho'}}(\llbracket \Gamma; \Phi \vdash F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} f x)$ holds, i.e., for all $\eta \in \llbracket \Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\alpha}}\, F\, G \rrbracket^{\mathsf{Set}} \rho$, we have $\llbracket \Gamma; \Phi \vdash G[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} f \circ \eta_{\overline{\llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}} = \eta_{\overline{\llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho'}} \circ \llbracket \Gamma; \Phi \vdash F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} f$. But this follows from the naturality of η: the only variables in the functorial contexts for F and G are $\overline{\alpha}$, so $\llbracket \Gamma; \Phi \vdash F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} \rho = \llbracket \Gamma; \overline{\alpha} \vdash F \rrbracket^{\mathsf{Set}} \rho[\overline{\alpha := \llbracket \Gamma; \Phi \vdash \tau \rrbracket^{\mathsf{Set}} \rho}]$ and $\llbracket \Gamma; \Phi \vdash F[\overline{\alpha := \tau}] \rrbracket^{\mathsf{Set}} f =$

$[\![\Gamma;\overline{\alpha} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\alpha := [\![\Gamma;\Phi \vdash \tau]\!]^{\mathrm{Set}}f}]$, and similarly for G, which ensures the commutativity of

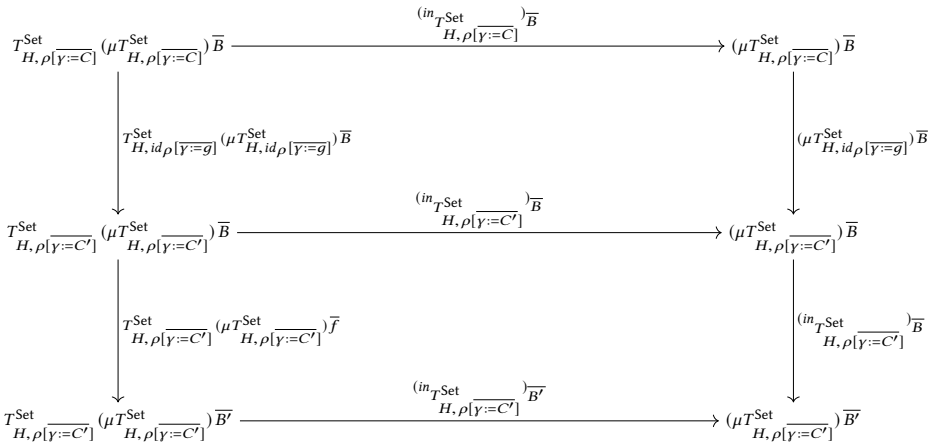
$$\begin{array}{ccc} [\![\Gamma;\overline{\alpha} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\alpha := [\![\Gamma;\Phi \vdash \tau]\!]^{\mathrm{Set}}\rho}] & \xrightarrow{\eta_{\overline{[\![\Gamma;\Phi\vdash\tau]\!]^{\mathrm{Set}}\rho}}} & [\![\Gamma;\overline{\alpha} \vdash G]\!]^{\mathrm{Set}}\rho[\overline{\alpha := [\![\Gamma;\Phi \vdash \tau]\!]^{\mathrm{Set}}\rho}] \\ \downarrow{\scriptstyle [\![\Gamma;\overline{\alpha}\vdash F]\!]^{\mathrm{Set}}id_\rho[\overline{\alpha:=[\![\Gamma;\Phi\vdash\tau]\!]^{\mathrm{Set}}f}]} & & \downarrow{\scriptstyle [\![\Gamma;\overline{\alpha}\vdash G]\!]^{\mathrm{Set}}id_\rho[\overline{\alpha:=[\![\Gamma;\Phi\vdash\tau]\!]^{\mathrm{Set}}f}]} \\ [\![\Gamma;\overline{\alpha} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\alpha := [\![\Gamma;\Phi \vdash \tau]\!]^{\mathrm{Set}}\rho'}] & \xrightarrow{\eta_{\overline{[\![\Gamma;\Phi\vdash\tau]\!]^{\mathrm{Set}}\rho'}}} & [\![\Gamma;\overline{\alpha} \vdash G]\!]^{\mathrm{Set}}\rho[\overline{\alpha := [\![\Gamma;\Phi \vdash \tau]\!]^{\mathrm{Set}}\rho'}] \end{array}$$

- $\Gamma;\emptyset \mid \emptyset \vdash \mathrm{map}_H^{\overline{F},\overline{G}} : \mathrm{Nat}^{\emptyset}\ (\overline{\mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\ F\,G})\ (\mathrm{Nat}^{\overline{\gamma}}\ H[\overline{\phi :=_{\overline{\beta}} F}]\ H[\overline{\phi :=_{\overline{\beta}} G}])$ Since Φ is empty, to see that $[\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{map}_H^{\overline{F},\overline{G}} : \mathrm{Nat}^{\emptyset}\ (\overline{\mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\ F\,G})\ (\mathrm{Nat}^{\overline{\gamma}}\ H[\overline{\phi :=_{\overline{\beta}} F}]\ H[\overline{\phi :=_{\overline{\beta}} G}])]\!]^{\mathrm{Set}}$ is a natural transformation from $[\![\Gamma;\emptyset \vdash \emptyset]\!]^{\mathrm{Set}}$ to $[\![\mathrm{Nat}^{\emptyset}\ (\overline{\mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\ F\,G})\ (\mathrm{Nat}^{\overline{\gamma}}\ H[\overline{\phi :=_{\overline{\beta}} F}]\ H[\overline{\phi :=_{\overline{\beta}} G}])]\!]^{\mathrm{Set}}$ we need only show that, for all $\rho : \mathrm{SetEnv}$, the unique $d : [\![\Gamma;\emptyset \vdash \emptyset]\!]^{\mathrm{Set}}\rho$, and all $\eta : [\![\Gamma;\emptyset \vdash \mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\ F\,G]\!]^{\mathrm{Set}}\rho$, we have $[\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{map}_H^{\overline{F},\overline{G}} : \mathrm{Nat}^{\emptyset}\ (\overline{\mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\ F\,G})\ (\mathrm{Nat}^{\overline{\gamma}}\ H[\overline{\phi :=_{\overline{\beta}} F}]\ H[\overline{\phi :=_{\overline{\beta}} G}])]\!]^{\mathrm{Set}}\ \rho\ d\ \overline{\eta} : [\![\Gamma;\emptyset \vdash \mathrm{Nat}^{\overline{\gamma}}\ H[\overline{\phi :=_{\overline{\beta}} F}]\ H[\overline{\phi :=_{\overline{\beta}} G}]]\!]^{\mathrm{Set}}\rho$. For this, we first note that $[\![\Gamma;\overline{\phi},\overline{\gamma} \vdash H]\!]^{\mathrm{Set}}$ is a functor from SetEnv to Set and, for any $\overline{B}$, $id_{\rho[\overline{\gamma:=B}]}[\overline{\phi := \lambda\overline{A}.\eta_{\overline{A}\,\overline{B}}}]$ is a morphism in SetEnv from
$$\rho[\overline{\gamma := B}][\overline{\phi := \lambda\overline{A}.[\![\Gamma;\overline{\gamma},\overline{\beta} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\gamma := B}][\overline{\beta := A}]}]$$
to
$$\rho[\overline{\gamma := B}][\overline{\phi := \lambda\overline{A}.[\![\Gamma;\overline{\gamma},\overline{\beta} \vdash G]\!]^{\mathrm{Set}}\rho[\overline{\gamma := B}][\overline{\beta := A}]}]$$
so that $([\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{map}_H^{\overline{F},\overline{G}} : \mathrm{Nat}^{\emptyset}\ (\overline{\mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\ F\,G})\ (\mathrm{Nat}^{\overline{\gamma}}\ H[\overline{\phi :=_{\overline{\beta}} F}]\ H[\overline{\phi :=_{\overline{\beta}} G}])]\!]^{\mathrm{Set}}\ \rho\ d\ \overline{\eta})_{\overline{B}} = [\![\Gamma;\overline{\phi},\overline{\gamma} \vdash H]\!]^{\mathrm{Set}}id_{\rho[\overline{\gamma:=B}]}[\overline{\phi := \lambda\overline{A}.\eta_{\overline{A}\,\overline{B}}}]$ is indeed a morphism from $[\![\Gamma;\overline{\gamma} \vdash H[\overline{\phi := F}]]\!]^{\mathrm{Set}}\rho[\overline{\gamma := B}]$ to $[\![\Gamma;\overline{\gamma} \vdash H[\overline{\phi := G}]]\!]^{\mathrm{Set}}\rho[\overline{\gamma := B}]$. This family of morphisms is natural in $\overline{B}$: if $\overline{f : B \to B'}$ then, writing t for $[\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{map}_H^{\overline{F},\overline{G}} : \mathrm{Nat}^{\emptyset}\ (\overline{\mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\ F\,G})\ (\mathrm{Nat}^{\overline{\gamma}}\ H[\overline{\phi :=_{\overline{\beta}} F}]\ H[\overline{\phi :=_{\overline{\beta}} G}])]\!]^{\mathrm{Set}}\ \rho\ d\ \overline{\eta}$, naturality of each η, together with the fact that composition of environments is computed componentwise, ensure that the following naturality diagram for t commutes:

$$\begin{array}{ccc} [\![\Gamma;\overline{\gamma} \vdash H[\overline{\phi := F}]]\!]^{\mathrm{Set}}\rho[\overline{\gamma := B}] & \xrightarrow{t_{\overline{B}}} & [\![\Gamma;\overline{\gamma} \vdash H[\overline{\phi := G}]]\!]^{\mathrm{Set}}\rho[\overline{\gamma := B}] \\ \downarrow{\scriptstyle [\![\Gamma;\overline{\gamma}\vdash H[\overline{\phi:=F}]]\!]^{\mathrm{Set}}id_\rho[\overline{\gamma:=f}]} & & \downarrow{\scriptstyle [\![\Gamma;\overline{\gamma}\vdash H[\overline{\phi:=G}]]\!]^{\mathrm{Set}}id_\rho[\overline{\gamma:=f}]} \\ [\![\Gamma;\overline{\gamma} \vdash H[\overline{\phi := F}]]\!]^{\mathrm{Set}}\rho[\overline{\gamma := B'}] & \xrightarrow{t_{\overline{B'}}} & [\![\Gamma;\overline{\gamma} \vdash H[\overline{\phi := G}]]\!]^{\mathrm{Set}}\rho[\overline{\gamma := B'}] \end{array}$$

That, for all $\rho : \mathrm{SetEnv}$ and $d : [\![\Gamma;\emptyset \vdash \emptyset]\!]^{\mathrm{Set}}\rho$, and all $\eta : \overline{[\![\Gamma;\emptyset \vdash \mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\ F\,G]\!]^{\mathrm{Set}}\rho}$,
$$[\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{map}_H^{\overline{F},\overline{G}} : \mathrm{Nat}^{\emptyset}\ (\overline{\mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\ F\,G})\ (\mathrm{Nat}^{\overline{\gamma}}\ H[\overline{\phi :=_{\overline{\beta}} F}]\ H[\overline{\phi :=_{\overline{\beta}} G}])]\!]^{\mathrm{Set}}\ \rho\ d\ \overline{\eta}$$
satisfies the additional condition needed for it to be in $[\![\Gamma;\emptyset \vdash \mathrm{Nat}^{\overline{\gamma}}\ H[\overline{\phi :=_{\overline{\beta}} F}]\ H[\overline{\phi :=_{\overline{\beta}} G}]]\!]^{\mathrm{Set}}\rho$, follows from the fact that each η satisfies the extra condition needed for it to be in its corresponding $[\![\Gamma;\emptyset \vdash \mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\ F\,G]\!]^{\mathrm{Set}}\rho$.
- $\Gamma;\emptyset \mid \emptyset \vdash \mathrm{in}_H : Nat^{\overline{\beta},\overline{\gamma}}\ H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}$ To see that if $d : [\![\Gamma;\emptyset \vdash \emptyset]\!]^{\mathrm{Set}}\rho$ then $[\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{in}_H : Nat^{\overline{\beta},\overline{\gamma}}\ H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathrm{Set}}\ \rho\ d$ is in $[\![\Gamma;\emptyset \vdash Nat^{\overline{\beta},\overline{\gamma}}\ H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathrm{Set}}\ \rho$, we first note that, for all $\overline{B}$ and $\overline{C}$, $([\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{in}_H : Nat^{\overline{\beta},\overline{\gamma}}\ H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathrm{Set}}\ \rho\ d)_{\overline{B}\,\overline{C}} = (in_{T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]}})_{\overline{B}}$ maps $[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]]\!]^{\mathrm{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}] = T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]}\ (\mu T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]})\ \overline{B}$

to $[\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathsf{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}] = (\mu T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C}]})\,\overline{B}$. Secondly, we observe that $[\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{in}_H : Nat^{\overline{\beta},\overline{\gamma}}\, H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathsf{Set}}\ \rho\, d = \lambda\overline{B}\,\overline{C}.\,(in_{T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C}]}})_{\overline{B}}$ is natural in $\overline{B}$ and $\overline{C}$, since naturality of *in* with respect to its functor argument and naturality of $in_{T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C'}]}}$ ensure that the following diagram commutes for all $\overline{f : B \to B'}$ and $\overline{g : C \to C'}$:



That, for all $\rho : \mathsf{SetEnv}$ and $d : [\![\Gamma; \emptyset \vdash \emptyset]\!]^{\mathsf{Set}}\rho$,

$$[\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{in}_H : Nat^{\overline{\beta},\overline{\gamma}}\, H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathsf{Set}}\ \rho\, d$$

satisfies the additional property needed for it to be in

$$[\![\Gamma; \emptyset \vdash Nat^{\overline{\beta},\overline{\gamma}}\, H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathsf{Set}}\ \rho$$

let $\overline{R : \mathsf{Rel}(B, B')}$ and $\overline{S : \mathsf{Rel}(C, C')}$ follows from the fact that

$$\begin{aligned} &(([\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{in}_H : Nat^{\overline{\beta},\overline{\gamma}}\, H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathsf{Set}}\ \rho\, d)_{\overline{B},\overline{C}},\\ &\qquad ([\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{in}_H : Nat^{\overline{\beta},\overline{\gamma}}\, H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathsf{Set}}\ \rho\, d)_{\overline{B'},\overline{C'}})\\ =\quad &((in_{T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C}]}})_{\overline{B}}, (in_{T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C'}]}})_{\overline{B'}}) \end{aligned}$$

has type

$$\begin{aligned} &(T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C}]}\,(\mu T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C}]})\,\overline{B} \to (\mu T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C}]})\,\overline{B},\\ &\qquad T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C'}]}\,(\mu T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C'}]})\,\overline{B'} \to (\mu T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C'}]})\,\overline{B'})\\ =\quad &[\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash H[\phi := (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}][\overline{\alpha := \beta}]]\!]^{\mathsf{Rel}}\mathsf{Eq}_\rho[\overline{\beta := R}][\overline{\gamma := S}] \to\\ &\qquad [\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathsf{Rel}}\mathsf{Eq}_\rho[\overline{\beta := R}][\overline{\gamma := S}] \end{aligned}$$

- $\dfrac{\Gamma; \emptyset \mid \emptyset \vdash \mathsf{fold}^F_H : \mathsf{Nat}^\emptyset\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha := \beta}]\, F)\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\ F)}{\text{to see that } [\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{fold}^F_H : \mathsf{Nat}^\emptyset\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha := \beta}]\, F)\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\ F)]\!]^{\mathsf{Set}}}$ Since Φ is empty, is a natural transformation $[\![\Gamma; \emptyset \vdash \emptyset]\!]^{\mathsf{Set}}$ to

$$[\![\Gamma; \emptyset \vdash \mathsf{Nat}^\emptyset\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha := \beta}]\, F)\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\ (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\ F]\!]^{\mathsf{Set}}$$

we need only show that, for all ρ : SetEnv, the unique d : $[\![\Gamma; \emptyset \vdash \emptyset]\!]^{\mathrm{Set}}\rho$, and all η : $[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha := \beta}]\, F]\!]^{\mathrm{Set}}\rho$,

$$[\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{fold}^F_H : \mathsf{Nat}^{\emptyset}\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha := \beta}]\, F)\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\, F]\!]^{\mathrm{Set}}\ \rho\, d\, \eta$$

has type $[\![\Gamma; \emptyset \vdash \mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\, F]\!]^{\mathrm{Set}}\ \rho$ i.e., for any $\overline{B}$ and $\overline{C}$,

$$([\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{fold}^F_H : \mathsf{Nat}^{\emptyset}\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha := \beta}]\, F)\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\, F]\!]^{\mathrm{Set}}\ \rho\, d\, \eta)_{\overline{B}\,\overline{C}}$$

is a morphism from $[\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathrm{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}] = (\mu T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]})\overline{B}$ to $[\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}]$. To see this, note that η is a natural transformation from

$$\begin{aligned} & \lambda\overline{B}\,\overline{C}.\, [\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash H[\phi := F][\overline{\alpha := \beta}]]\!]^{\mathrm{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}] \\ = \quad & \lambda\overline{B}\,\overline{C}.\, T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]}\ (\lambda\overline{A}.\, [\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\beta := A}][\overline{\gamma := C}])\ \overline{B} \end{aligned}$$

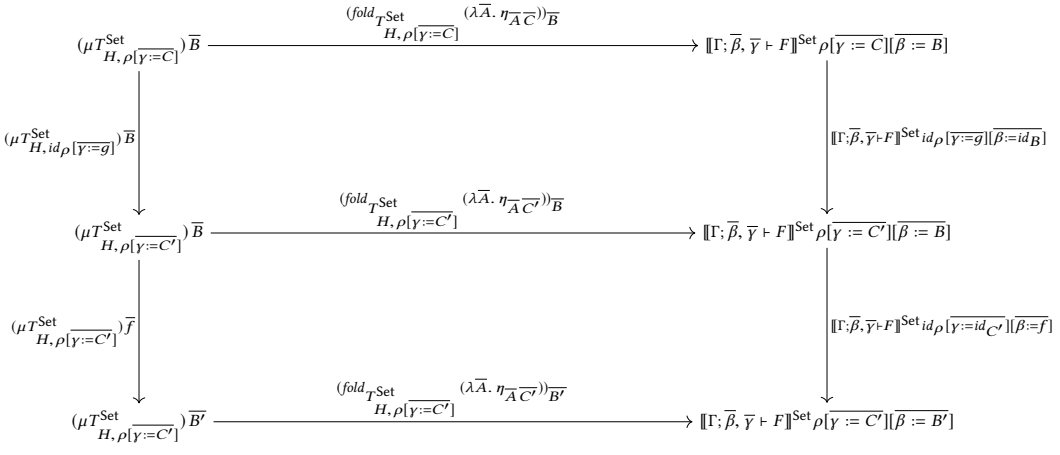
to

$$\begin{aligned} & \lambda\overline{B}\,\overline{C}.\, (\lambda\overline{A}.\, [\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\beta := A}][\overline{\gamma := C}])\overline{B} \\ = \quad & \lambda\overline{B}\,\overline{C}.\, [\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}] \end{aligned}$$

and thus for each $\overline{B}$ and $\overline{C}$,

$$([\![\Gamma; \emptyset \mid \emptyset \vdash \mathsf{fold}^F_H : \mathsf{Nat}^{\emptyset}\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha := \beta}]\, F)\ (\mathsf{Nat}^{\overline{\beta},\overline{\gamma}}\, (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\, F]\!]^{\mathrm{Set}}\ \rho\, d\, \eta)_{\overline{B}\,\overline{C}}$$

is a morphism from $[\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathrm{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}] = (\mu T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]})\overline{B}$ to $[\![\Gamma; \overline{\beta}, \overline{\gamma} \vdash F]\!]^{\mathrm{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}]$. To see that this family of morphisms is natural in $\overline{B}$ and $\overline{C}$, we observe that the following diagram commutes for all $\overline{f : B \to B'}$ and $\overline{g : C \to C'}$:



Indeed, naturality of $fold_{T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C'}]}}\ (\lambda\overline{A}.\, \eta_{\overline{A}\,\overline{C'}})$ ensures that the bottom diagram commutes. To see that the top one commutes we first observe that, given a natural transformation $\Theta : H \to K : [\mathrm{Set}^k, \mathrm{Set}] \to [\mathrm{Set}^k, \mathrm{Set}]$, the fixpoint natural transformation $\mu\Theta : \mu H \to \mu K : \mathrm{Set}^k \to \mathrm{Set}$ is defined to be $fold_H(\Theta\,(\mu K) \circ in_K)$, i.e., the unique morphism making the

following diagram commute:

$$\begin{array}{ccc} H(\mu H) & \xrightarrow{H(\mu\Theta)} & H(\mu K) \\ \Big\downarrow{\scriptstyle in_H} & & \Big\downarrow{\scriptstyle \Theta(\mu K)} \\ \mu H & \xrightarrow[\mu\Theta]{} & K(\mu K) \\ & & \Big\downarrow{\scriptstyle in_K} \\ & & \mu K \end{array}$$

Taking $\Theta = T^{\mathsf{Set}}_{H,f} : T^{\mathsf{Set}}_{H,\rho} \to T^{\mathsf{Set}}_{H,\rho'}$ thus gives that, for any $f : \rho \to \rho'$ in SetEnv,

$$in_{T^{\mathsf{Set}}_{H,\rho'}} \circ T^{\mathsf{Set}}_{H,f}(\mu T^{\mathsf{Set}}_{H,\rho'}) \circ T^{\mathsf{Set}}_{H,\rho}(\mu T^{\mathsf{Set}}_{H,f}) \;=\; \mu T^{\mathsf{Set}}_{H,f} \circ in_{T^{\mathsf{Set}}_{H,\rho}} \tag{14}$$

Next, note that the action of the functor $\lambda\overline{B}.\lambda\overline{C}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash H[\phi := F][\overline{\alpha := \beta}]]\!]^{\mathsf{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}]$ on the morphisms $\overline{f : B \to B'}, \overline{g : C \to C'}$ is given by

$$\begin{array}{lll} & [\![\Gamma;\overline{\beta},\overline{\gamma} \vdash H[\phi := F][\overline{\alpha := \beta}]]\!]^{\mathsf{Set}} id_\rho[\overline{\beta := f}][\overline{\gamma := g}] \\ = & [\![\Gamma;\phi,\overline{\alpha},\overline{\gamma} \vdash H]\!]^{\mathsf{Set}} id_\rho[\overline{\alpha := f}][\overline{\gamma := g}][\phi := \lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}} id_{\rho[\overline{\beta:=A}]}[\overline{\gamma := g}]] \\ = & [\![\Gamma;\phi,\overline{\alpha},\overline{\gamma} \vdash H]\!]^{\mathsf{Set}} id_{\rho[\overline{\gamma:=C'}][\phi:=\lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma}\vdash F]\!]^{\mathsf{Set}}\rho[\overline{\beta:=A}][\overline{\gamma:=C'}]]}[\overline{\alpha := f}] \\ & \quad \circ [\![\Gamma;\phi,\overline{\alpha},\overline{\gamma} \vdash H]\!]^{\mathsf{Set}} id_{\rho[\overline{\alpha:=B}][\phi:=\lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma}\vdash F]\!]^{\mathsf{Set}}\rho[\overline{\beta:=A}][\overline{\gamma:=C'}]]}[\overline{\gamma := g}] \\ & \quad \circ [\![\Gamma;\phi,\overline{\alpha},\overline{\gamma} \vdash H]\!]^{\mathsf{Set}} id_{\rho[\overline{\alpha:=B}][\overline{\gamma:=C}]}[\phi := \lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}} id_{\rho[\overline{\beta:=A}]}[\overline{\gamma := g}]] \\ = & T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C'}]}(\lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}}\rho[\overline{\beta := A}][\overline{\gamma := C'}])\overline{f} \\ & \quad \circ \big(T^{\mathsf{Set}}_{H,id_\rho[\overline{\gamma:=g}]}(\lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}}\rho[\overline{\beta := A}][\overline{\gamma := C'}])\big)_{\overline{B}} \\ & \quad \circ \big(T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C}]}(\lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}} id_{\rho[\overline{\beta:=A}]}[\overline{\gamma := g}])\big)_{\overline{B}} \end{array}$$

So if η is a natural transformation such that $\eta_{\overline{B},\overline{C}}$ has type

$$[\![\Gamma;\overline{\alpha},\overline{\gamma} \vdash H[\phi := F][\overline{\alpha := \beta}]]\!]^{\mathsf{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}] \to [\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}}\rho[\overline{\beta := B}][\overline{\gamma := C}]$$

then, by naturality,

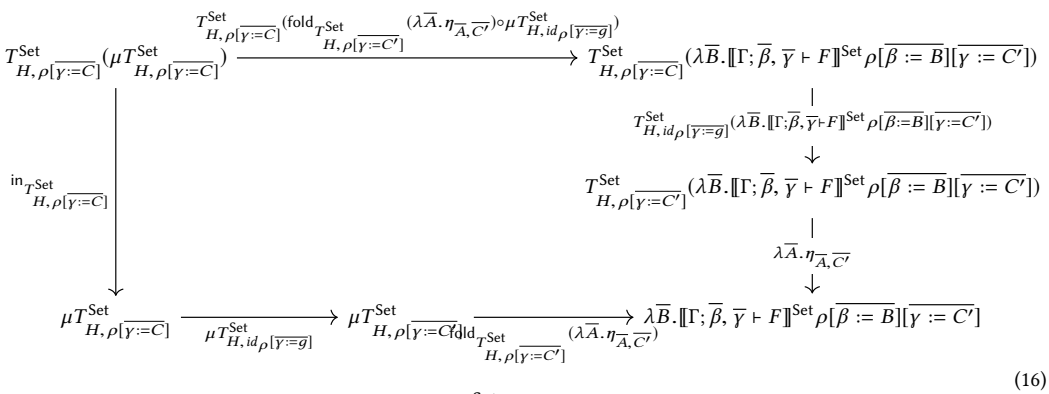
$$\begin{array}{lll} & [\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}} id_\rho[\overline{\beta := f}][\overline{\gamma := g}] \circ \eta_{\overline{B},\overline{C}} \\ = & \eta_{\overline{B'},\overline{C'}} \circ T^{\mathsf{Set}}_{\rho[\overline{\gamma:=C'}]}(\lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}}\rho[\overline{\beta := A}][\overline{\gamma := C'}])\overline{f} \\ & \circ \big(T^{\mathsf{Set}}_{H,id_\rho[\overline{\gamma:=g}]}(\lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}}\rho[\overline{\beta := A}][\overline{\gamma := C'}])\big)_{\overline{B}} \\ & \circ \big(T^{\mathsf{Set}}_{\rho[\overline{\gamma:=C}]}(\lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}} id_{\rho[\overline{\beta:=A}]}[\overline{\gamma := g}])\big)_{\overline{B}} \end{array}$$

As a special case when $\overline{f = id_B}$ we have

$$\begin{array}{lll} & \lambda\overline{B}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}} id_{\rho[\overline{\beta:=B}]}[\overline{\gamma := g}] \circ \lambda\overline{B}.\eta_{\overline{B},\overline{C}} \\ = & \lambda\overline{B}.\eta_{\overline{B},\overline{C'}} \circ \sigma^{\mathsf{Set}}_{id_\rho[\overline{\gamma:=g}]}(\lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}}\rho[\overline{\beta := A}][\overline{\gamma := C'}]) \\ & \qquad \circ T^{\mathsf{Set}}_{\rho[\overline{\gamma:=C}]}(\lambda\overline{A}.[\![\Gamma;\overline{\beta},\overline{\gamma} \vdash F]\!]^{\mathsf{Set}} id_{\rho[\overline{\beta:=A}]}[\overline{\gamma := g}]) \end{array} \tag{15}$$

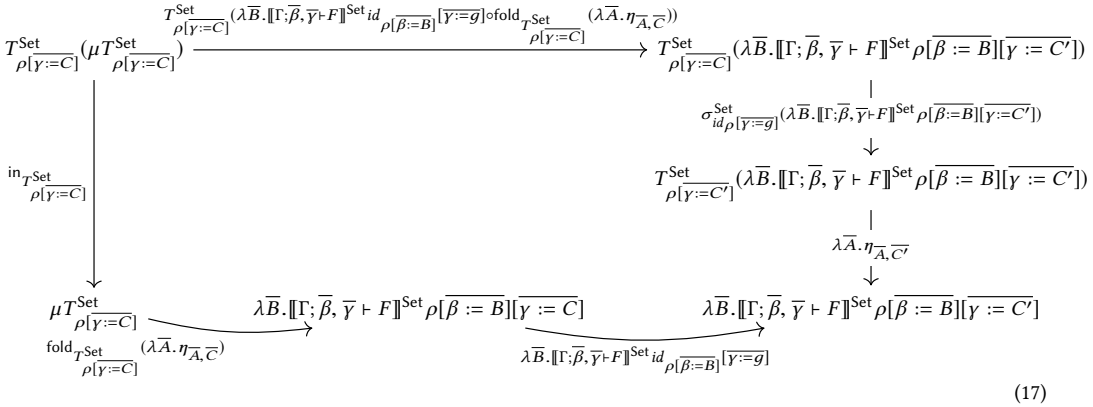
Finally, to see that the top diagram in the diagram on page 28 commutes we first note that functoriality of $T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C}]}$, naturality of $T^{\mathsf{Set}}_{H,id_\rho[\overline{\gamma:=g}]}$, the universal property of $\mathrm{fold}_{T^{\mathsf{Set}}_{H,\rho[\overline{\gamma:=C'}]}}(\lambda\overline{A}.\eta_{\overline{A},\overline{C'}})$

and Equation 14 ensure that the following diagram commutes:



(16)

Next, we note that functoriality of $T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]}$, Equation 15, and the universal property of $\mathrm{fold}_{T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]}}(\lambda\overline{A}.\eta_{\overline{A},\overline{C}})$ ensure that the following diagram commutes:



(17)

Combining the equations entailed by 16 and 17, we get that the top diagram in the diagram on page 28 commutes, as desired. To see that, for all $\rho : \mathrm{SetEnv}$, $d \in [\![\Gamma;\emptyset \vdash \emptyset]\!]^{\mathrm{Set}}\rho$, and $\eta : [\![\Gamma;\emptyset \vdash \mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha := \beta}]\, F]\!]^{\mathrm{Set}}\rho$,

$$[\![\Gamma;\emptyset \mid \emptyset \vdash \mathrm{fold}^{F}_{H} : \mathrm{Nat}^{\emptyset}\ (\mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\, H[\phi :=_{\overline{\beta}} F][\overline{\alpha := \beta}]\, F)\ (\mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\, (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\, F]\!]^{\mathrm{Set}}\ \rho\, d\, \eta$$

satisfies the additional condition needed for it to be in $[\![\Gamma;\emptyset \vdash \mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\, (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}\ F]\!]^{\mathrm{Set}}\ \rho$, let $\overline{R : \mathrm{Rel}(B, B')}$ and $\overline{S : \mathrm{Rel}(C, C')}$. Since η satisfies the additional condition needed for it to be in $[\![\Gamma;\emptyset \vdash \mathrm{Nat}^{\overline{\beta},\overline{\gamma}}\, (H[\phi := F][\overline{\alpha := \beta}])\, F]\!]^{\mathrm{Set}}\rho$,

$$(\,(\mathit{fold}_{T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C}]}}(\lambda\overline{A}.\,\eta_{\overline{A}\,\overline{C}}))_{\overline{B}},\ (\mathit{fold}_{T^{\mathrm{Set}}_{H,\rho[\overline{\gamma:=C'}]}}(\lambda\overline{A}.\eta_{\overline{A}\,\overline{C'}}))_{\overline{B'}}\,)$$

has type

$$\begin{aligned}
& (\mu T_{H,\mathrm{Eq}_\rho[\overline{\gamma:=S}]})\,\overline{R} \to [\![\Gamma;\overline{\gamma},\overline{\beta} \vdash F]\!]^{\mathrm{Rel}}\mathrm{Eq}_\rho[\overline{\gamma := S}][\overline{\beta := R}] \\
= \quad & (\mu T_{H,\mathrm{Eq}_\rho[\overline{\gamma:=S}]})\,\overline{[\![\Gamma;\overline{\gamma},\overline{\beta} \vdash \beta]\!]^{\mathrm{Rel}}\mathrm{Eq}_\rho[\overline{\gamma := S}][\overline{\beta := R}]} \to [\![\Gamma;\overline{\gamma},\overline{\beta} \vdash F]\!]^{\mathrm{Rel}}\mathrm{Eq}_\rho[\overline{\gamma := S}][\overline{\beta := R}] \\
= \quad & [\![\Gamma;\overline{\gamma},\overline{\beta} \vdash (\mu\phi.\lambda\overline{\alpha}.H)\overline{\beta}]\!]^{\mathrm{Rel}}\mathrm{Eq}_\rho[\overline{\gamma := S}][\overline{\beta := R}] \to [\![\Gamma;\overline{\gamma},\overline{\beta} \vdash F]\!]^{\mathrm{Rel}}\mathrm{Eq}_\rho[\overline{\gamma := S}][\overline{\beta := R}]
\end{aligned}$$

□

The following theorem is an immediate consequence of Theorem 26:

Theorem 27. *If* $\Gamma;\Phi \mid \Delta \vdash t : \tau$ *and* $\rho \in \mathsf{RelEnv}$, *and if* $(a,b) \in [\![\Gamma;\Phi \vdash \Delta]\!]^{\mathsf{Rel}}\rho$, *then* $([\![\Gamma;\Phi \mid \Delta \vdash t : \tau]\!]^{\mathsf{Set}}(\pi_1\rho)\,a\,, [\![\Gamma;\Phi \mid \Delta \vdash t : \tau]\!]^{\mathsf{Set}}(\pi_2\rho)\,b) \in [\![\Gamma;\Phi \vdash \tau]\!]^{\mathsf{Rel}}\rho$

Finally, the Abstraction Theorem is the instantiation of Theorem 27 to closed terms of closed type:

Theorem 28. *If* $\vdash t : \tau$, *then* $([\![\vdash t : \tau]\!]^{\mathsf{Set}}, [\![\vdash t : \tau]\!]^{\mathsf{Set}}) \in [\![\vdash \tau]\!]^{\mathsf{Rel}}$.

6 FREE THEOREMS FOR NESTED TYPES

6.1 Free Theorem for Type of Polymorphic Bottom

Suppose $\vdash g : \mathsf{Nat}^{\alpha}\,\mathbb{1}\,\alpha$, let $G^{\mathsf{Set}} = [\![\vdash g : \mathsf{Nat}^{\alpha}\,\mathbb{1}\,\alpha]\!]^{\mathsf{Set}}$, and let $G^{\mathsf{Rel}} = [\![\vdash g : \mathsf{Nat}^{\alpha}\,\mathbb{1}\,\alpha]\!]^{\mathsf{Rel}}$. By Theorem 28, $(G^{\mathsf{Set}}(\pi_1\rho), G^{\mathsf{Set}}(\pi_2\rho)) = G^{\mathsf{Rel}}\rho$. Thus, for all $\rho \in \mathsf{RelEnv}$ and any $(a,b) \in [\![\vdash \emptyset]\!]^{\mathsf{Rel}}\rho = 1$, eliding the only possible instantiations of a and b gives that $(G^{\mathsf{Set}}, G^{\mathsf{Set}}) = (G^{\mathsf{Set}}(\pi_1\rho), G^{\mathsf{Set}}(\pi_2\rho)) \in [\![\vdash \mathsf{Nat}^{\alpha}\,\mathbb{1}\,\alpha]\!]^{\mathsf{Rel}}\rho = \{\eta : K_1 \Rightarrow id\} = \{(\eta_1 : K_1 \Rightarrow id, \eta_2 : K_1 \Rightarrow id)\}$ That is, G^{Set} is a natural transformation from the constantly 1-valued functor to the identity functor in Set. In particular, for every $S : \mathsf{Set}$, $G^{\mathsf{Set}}_S : 1 \to S$. Note, however, that if $S = \emptyset$, then there can be no such morphism, so no such natural transformation can exist, and thus no term $\vdash g : \mathsf{Nat}^{\alpha}\,\mathbb{1}\,\alpha$ can exist. That is, our calculus does not admit any terms with the closed type $\mathsf{Nat}^{\alpha}\,\mathbb{1}\,\alpha$ of the polymorphic bottom.

6.2 Free Theorem for Type of Polymorphic Identity

Suppose $\vdash g : \mathsf{Nat}^{\alpha}\,\alpha\,\alpha$, let $G^{\mathsf{Set}} = [\![\vdash g : \mathsf{Nat}^{\alpha}\,\alpha\,\alpha]\!]^{\mathsf{Set}}$, and let $G^{\mathsf{Rel}} = [\![\vdash g : \mathsf{Nat}^{\alpha}\,\alpha\,\alpha]\!]^{\mathsf{Rel}}$. By Theorem 28, $(G^{\mathsf{Set}}(\pi_1\rho), G^{\mathsf{Set}}(\pi_2\rho)) = G^{\mathsf{Rel}}\rho$. Thus, for all $\rho \in \mathsf{RelEnv}$ and any $(a,b) \in [\![\vdash \emptyset]\!]^{\mathsf{Rel}}\rho = 1$, eliding the only possible instantiations of a and b gives that $(G^{\mathsf{Set}}, G^{\mathsf{Set}}) = (G^{\mathsf{Set}}(\pi_1\rho), G^{\mathsf{Set}}(\pi_2\rho)) \in [\![\vdash \mathsf{Nat}^{\alpha}\,\alpha\,\alpha]\!]^{\mathsf{Rel}}\rho = \{\eta : id \Rightarrow id\} = \{(\eta_1 : id \Rightarrow id, \eta_2 : id \Rightarrow id)\}$ That is, G^{Set} is a natural transformation from the identity functor on Set to itself. Now let S be any set. If $S = \emptyset$, then there is exactly one morphism $id_S : S \to S$, so $G^{\mathsf{Set}}_S : S \to S$ must be id_S. If $S \neq \emptyset$, then if a is any element of S and $K_a : S \to S$ is the constantly a-valued morphism on S, then instantiating the naturality square implied by the above equality gives that $G^{\mathsf{Set}}_S \circ K_a = K_a \circ G^{\mathsf{Set}}_S$, i.e., $G^{\mathsf{Set}}_S\,a = a$, i.e., $G^{\mathsf{Set}}_S = id_S$. Putting these two cases together we have that for every $S : \mathsf{Set}$, $G^{\mathsf{Set}}_S = id_S$, i.e., G^{Set} is the identity natural transformation for the identity functor on Set. So every closed term g of closed type $\mathsf{Nat}^{\alpha}\,\alpha\,\alpha$ always denotes the identity natural transformation for the identity functor on Set, i.e., every closed term g of type $\mathsf{Nat}^{\alpha}\,\alpha\,\alpha$ denotes the polymorphic identity function.

6.3 Standard Free Theorems for ADTs and Their Analogues for Nested types

We can derive in our system even those free theorems for polymorphic functions over ADTs that are not consequences of naturality. We can, e.g., prove the free theorem for *filter*'s type as follows:

Theorem 29. *If* $g : A \to B$, $\rho : \mathsf{RelEnv}$, $\rho\alpha = (A, B, \langle g \rangle)$, $(a,b) \in [\![\alpha;\emptyset \vdash \Delta]\!]^{\mathsf{Rel}}\rho$, $(s \circ g, s) \in [\![\alpha;\emptyset \vdash \mathsf{Nat}^{\emptyset}\,\alpha\;\mathit{Bool}]\!]^{\mathsf{Rel}}\rho$, *and* $\mathit{filter} = [\![\alpha;\emptyset \mid \Delta \vdash t : \mathsf{Nat}^{\emptyset}(\mathsf{Nat}^{\emptyset}\,\alpha\;\mathit{Bool})\,(\mathsf{Nat}^{\emptyset}(\mathit{List}\,\alpha)\,(\mathit{List}\,\alpha))]\!]^{\mathsf{Set}}$ *for some* t, *then*

$$\mathit{map}_{\mathit{List}}\,g \circ \mathit{filter}\,(\pi_1\rho)\,a\,(s \circ g) = \mathit{filter}\,(\pi_2\rho)\,b\,s \circ \mathit{map}_{\mathit{List}}\,g$$

Proof. By Theorem 27,

$$(\mathit{filter}\,(\pi_1\rho)\,a, \mathit{filter}\,(\pi_2\rho)\,b) \in [\![\alpha;\emptyset \vdash \mathsf{Nat}^{\emptyset}(\mathsf{Nat}^{\emptyset}\,\alpha\;\mathit{Bool})\,(\mathsf{Nat}^{\emptyset}(\mathit{List}\,\alpha)\,(\mathit{List}\,\alpha))]\!]^{\mathsf{Rel}}\rho$$

so if $(s,s') \in [\![\alpha;\emptyset \vdash \mathsf{Nat}^{\emptyset}\alpha\;\mathit{Bool}]\!]^{\mathsf{Rel}}\rho = \rho\alpha \to \mathsf{Eq}_{\mathit{Bool}}$ and $(xs,xs') \in [\![\alpha;\emptyset \vdash \mathit{List}\,\alpha]\!]^{\mathsf{Rel}}\rho$ then

$$(\mathit{filter}\,(\pi_1\rho)\,a\,s\,xs, \mathit{filter}\,(\pi_2\rho)\,b\,s'\,xs') \in [\![\alpha;\emptyset \vdash \mathit{List}\,\alpha]\!]^{\mathsf{Rel}}\rho \tag{18}$$

If $\rho\alpha = (A, B, \langle g\rangle)$, then $[\![\alpha; \emptyset \vdash \mathit{List}\,\alpha]\!]^{\mathrm{Rel}}\rho = \langle \mathit{map}_{\mathit{List}}\, g\rangle$ by Lemma 23 and Equation 4. Moreover, $xs' = \mathit{map}_{\mathit{List}}\, g\, xs$ and $(s', s) \in \langle g\rangle \to \mathrm{Eq}_{\mathit{Bool}}$, so $s' = s \circ g$. Instantiating Equation 18 thus gives the desired result. □

But rose trees are essentially first-order. Not for perfect trees or bushes... Say more?

A similar proof establishes the analogous result for, say, generalized rose trees:

Theorem 30. *If* $g : A \to B, F, G : \mathrm{Set} \to \mathrm{Set}$, $\eta : F \to G$ *in* Set, $\rho : \mathrm{RelEnv}$, $\rho\alpha = (A, B, \langle g\rangle)$, $\rho\psi = (F, G, \langle \eta\rangle)$, $(a, b) \in [\![\alpha, \psi; \emptyset \vdash \Delta]\!]^{\mathrm{Rel}}\rho$, $(s \circ g, s) \in [\![\alpha; \emptyset \vdash \mathrm{Nat}^{\emptyset}\alpha\, \mathit{Bool}]\!]^{\mathrm{Rel}}\rho$, *and*

$$\mathit{filter} = [\![\alpha, \psi; \emptyset \mid \Delta \vdash t : \mathrm{Nat}^{\emptyset}(\mathrm{Nat}^{\emptyset}\,\alpha\,\mathit{Bool})\,(\mathrm{Nat}^{\emptyset}(\mathit{GRose}\,\psi\,\alpha)\,(\mathit{GRose}\,\psi\,(\alpha + \mathbb{1})))]\!]^{\mathrm{Set}}$$

for some t, *then*

$$\mathit{map}_{\mathit{GRose}}\,\eta\,(g + 1) \circ \mathit{filter}\,(\pi_1\rho)\,a\,(s \circ g) = \mathit{filter}\,(\pi_2\rho)\,b\,s \circ \mathit{map}_{\mathit{GRose}}\,\eta\, g$$

6.4 Short Cut Fusion for Lists

We can recover standard short cut fusion for lists [Gill et al. 1993] in our system:

Theorem 31. *If* $\vdash \tau : \mathcal{F}$, $\vdash \tau' : \mathcal{F}$, *and* $G = [\![\beta; \emptyset \mid \emptyset \vdash g : \mathrm{Nat}^{\emptyset}(\mathrm{Nat}^{\emptyset}(\mathbb{1} + \tau \times \beta)\,\beta)\,\beta]\!]^{\mathrm{Set}}$ *for some* g, *then*

$$\mathit{fold}_{1+\tau\times\_}\, n\, c\, (G\,(\mathit{List}\,[\![\vdash \tau]\!]^{\mathrm{Set}})\,\mathit{nil}\,\mathit{cons}) = G\,[\![\vdash \tau']\!]^{\mathrm{Set}}\, n\, c$$

Proof. Theorem 28 gives that, for any $\rho : \mathrm{RelEnv}$,

$$\begin{array}{rcl}(G\,(\pi_1\rho), G\,(\pi_2\rho)) & \in & [\![\beta; \emptyset \vdash \mathrm{Nat}^{\emptyset}(\mathrm{Nat}^{\emptyset}(\mathbb{1} + \tau \times \beta)\,\beta)\,\beta]\!]^{\mathrm{Rel}}\rho \\ & \cong & (((([\![\vdash \tau]\!]^{\mathrm{Rel}}\rho \times \rho\beta) \to \rho\beta) \times \rho\beta) \to \rho\beta)\end{array}$$

so if $(c', c) \in [\![\vdash \tau]\!]^{\mathrm{Rel}}\rho \times \rho\beta \to \rho\beta$ and $(n', n) \in \rho\beta$ then $(G\,(\pi_1\rho)\,n'\,c', G\,(\pi_2\rho)\,n\,c) \in \rho\beta$. In addition,

$$[\![\vdash \mathrm{fold}^{\tau'}_{\mathbb{1}+\tau\times\beta} : \mathrm{Nat}^{\emptyset}(\mathrm{Nat}^{\emptyset}(\mathbb{1} + \tau \times \tau')\,\tau')\,(\mathrm{Nat}^{\emptyset}(\mu\alpha.\mathbb{1} + \tau \times \alpha)\,\tau')]\!]^{\mathrm{Set}} = \mathit{fold}_{1+\tau\times\_}$$

so that if $c \in [\![\vdash \tau]\!]^{\mathrm{Set}} \times [\![\vdash \tau']\!]^{\mathrm{Set}} \to [\![\vdash \tau']\!]^{\mathrm{Set}}$ and $n \in [\![\vdash \tau']\!]^{\mathrm{Set}}$, then $(n, c) \in [\![\vdash \mathrm{Nat}^{\emptyset}(\mathbb{1} + \tau \times \tau')\,\tau']\!]^{\mathrm{Set}}$. The instantiation $\pi_1\rho\beta = [\![\vdash \mu\alpha.\mathbb{1} + \tau \times \alpha]\!]^{\mathrm{Set}} = \mathit{List}\,[\![\vdash \tau]\!]^{\mathrm{Set}}$, $\pi_2\rho\beta = [\![\vdash \tau']\!]^{\mathrm{Set}}$, $\rho\beta = \langle \mathit{fold}_{1+\tau\times\_}\, n\, c\rangle : \mathrm{Rel}(\pi_1\rho\beta, \pi_2\rho\beta)$, $c' = \mathit{cons}$, and $n' = \mathit{nil}$ thus gives that $(G\,(\mathit{List}\,[\![\vdash \tau]\!]^{\mathrm{Set}})\,\mathit{nil}\,\mathit{cons}, G\,[\![\vdash \tau']\!]^{\mathrm{Set}}\, n\, c) \in \langle \mathit{fold}_{1+\tau\times\_}\, n\, c\rangle$, i.e., that $\mathit{fold}_{1+\tau\times\_}\, n\, c\, (G\,(\mathit{List}\,[\![\vdash \tau]\!]^{\mathrm{Set}})\,\mathit{nil}\,\mathit{cons}) = G\,[\![\vdash \tau']\!]^{\mathrm{Set}}\, n\, c$. □

Using Equation 4 we can extend short cut fusion results to arbitrary ADTs, as in [Johann 2002].

6.5 Short Cut Fusion for Arbitrary Nested Types

Using our system we can, finally, formally prove correctness of the categorically inspired short cut fusion for nested types from [Johann and Ghani 2010]. Indeed, we have:

Theorem 32. *If* $\emptyset; \phi, \alpha \vdash F : \mathcal{F}$, $\emptyset; \alpha \vdash K : \mathcal{F}$, $H : [\mathrm{Set}, \mathrm{Set}] \to [\mathrm{Set}, \mathrm{Set}]$ *is defined by*

$$H\, f\, x \quad = \quad [\![\emptyset; \phi, \alpha \vdash F]\!]^{\mathrm{Set}}[\phi := f][\alpha := x]$$

and

$$G = [\![\phi; \emptyset \mid \emptyset \vdash g : \mathrm{Nat}^{\emptyset}\,(\mathrm{Nat}^{\alpha}\,F\,(\phi\alpha))\,(\mathrm{Nat}^{\alpha}\,\mathbb{1}\,(\phi\alpha))]\!]^{\mathrm{Set}}$$

for some g, *then, for every* $B \in H[\![\emptyset; \alpha \vdash K]\!]^{\mathrm{Set}} \to [\![\emptyset; \alpha \vdash K]\!]^{\mathrm{Set}}$, $\mathit{fold}_H\, B\,(G\,\mu H\, \mathit{in}_H) = G\,[\![\emptyset; \alpha \vdash K]\!]^{\mathrm{Set}}\, B$.

Proof. Theorem 28 gives that, for any $\rho : \mathrm{RelEnv}$,

$$\begin{array}{rcl}(G\,(\pi_1\rho), G\,(\pi_2\rho)) & \in & [\![\phi; \emptyset \vdash \mathrm{Nat}^{\emptyset}(\mathrm{Nat}^{\alpha}F\,(\phi\alpha))\,(\mathrm{Nat}^{\alpha}\,\mathbb{1}\,(\phi\alpha))]\!]^{\mathrm{Rel}}\rho \\ & = & [\![\phi; \emptyset \vdash \mathrm{Nat}^{\alpha}F\,(\phi\alpha)]\!]^{\mathrm{Rel}}\rho \to [\![\phi; \emptyset \vdash \mathrm{Nat}^{\alpha}\,\mathbb{1}\,(\phi\alpha)]\!]^{\mathrm{Rel}}\rho \\ & = & [\![\phi; \emptyset \vdash \mathrm{Nat}^{\alpha}F\,(\phi\alpha)]\!]^{\mathrm{Rel}}\rho \to \rho\phi\end{array}$$

so if $(A,B) \in [\![\phi;\emptyset \vdash \mathsf{Nat}^{\alpha} F\,(\phi\alpha)]\!]^{\mathsf{Rel}}\rho$ then $(G\,(\pi_1\rho)\,A, G\,(\pi_2\rho)\,B) \in \rho\phi$. In addition,

$$[\![\vdash \mathsf{fold}_F^K : \mathsf{Nat}^{\emptyset}\,(\mathsf{Nat}^{\alpha} F[\phi := K]\,K)\,(\mathsf{Nat}^{\alpha}((\mu\phi.\lambda\alpha.F)\alpha)\,K)]\!]^{\mathsf{Set}} = \mathit{fold}_H$$

Consider the instantiation $A = \mathit{in}_H : H(\mu H) \Rightarrow \mu H$, $B : H[\![\emptyset;\alpha \vdash K]\!]^{\mathsf{Set}} \Rightarrow [\![\emptyset;\alpha \vdash K]\!]^{\mathsf{Set}}$, $\rho\phi = \langle \mathit{fold}_H\, B\rangle$, $\pi_1\rho\phi = \mu H$, $\pi_2\rho\phi = [\![\emptyset;\alpha \vdash K]\!]^{\mathsf{Set}}$, $\rho\phi : \mathsf{Rel}(\pi_1\rho\phi, \pi_2\rho\phi)$, $A : [\![\phi;\emptyset \vdash \mathsf{Nat}^{\alpha} F\,(\phi\alpha)]\!]^{\mathsf{Set}}(\pi_1\rho)$, and $B : [\![\phi;\emptyset \vdash \mathsf{Nat}^{\alpha} F\,(\phi\alpha)]\!]^{\mathsf{Set}}(\pi_2\rho)$. Equation 4 ensures that $A = \mathit{in}_H : H(\mu H) \Rightarrow \mu H = [\![\phi;\emptyset \vdash \mathsf{Nat}^{\alpha} F\,(\phi\alpha)]\!]^{\mathsf{Set}}(\pi_1\rho)$, and Equation 4 and Lemma 23 together give that

$$\begin{array}{rcll}(A,B) = (\mathit{in}_H, B) & \in & [\![\phi;\emptyset \vdash \mathsf{Nat}^{\alpha} F\,(\phi\alpha)]\!]^{\mathsf{Rel}}\rho \\ & = & \lambda A.[\![\phi;\alpha \vdash F]\!]^{\mathsf{Rel}}[\phi := \langle \mathit{fold}_H\, B\rangle][\alpha := A] \Rightarrow \langle \mathit{fold}_H\, B\rangle \\ & = & [\![\emptyset;\phi,\alpha \vdash F]\!]^{\mathsf{Rel}}\langle \mathit{fold}_H\, B\rangle \Rightarrow \langle \mathit{fold}_H\, B\rangle \\ & = & \langle [\![\emptyset;\phi,\alpha \vdash F]\!]^{\mathsf{Set}}\,(\mathit{fold}_H\, B)\rangle \Rightarrow \langle \mathit{fold}_H\, B\rangle \\ & = & \langle \mathit{map}_H\,(\mathit{fold}_H\, B)\rangle \Rightarrow \langle \mathit{fold}_H\, B\rangle \end{array}$$

since if $(x,y) \in \langle \mathit{map}_H\,(\mathit{fold}_H\, B)\rangle$, then $\mathit{fold}_H\, B\,(\mathit{in}_H\, x) = B\, y = B\,(\mathit{map}_H\,(\mathit{fold}_H\, B)\, x)$ by the definition of fold_H as a (indeed, the unique) morphism from in_H to B. Thus, $(G\,(\pi_1\rho)\,A, G\,(\pi_2\rho)\,B) \in \langle \mathit{fold}_H\, B\rangle$, i.e., $\mathit{fold}_H\, B\,(G\,(\pi_1\rho)\,\mathit{in}_H) = G\,(\pi_2\rho)\,B$. But since ϕ is the only free variable in G, this simplifies to $\mathit{fold}_H\, B\,(G\,\mu H\,\mathit{in}_H) = G\,[\![\emptyset;\alpha \vdash K]\!]^{\mathsf{Set}}\, B$. □

As in [Johann and Ghani 2010], replacing $\mathbb{1}$ with any term $\emptyset;\alpha \vdash c$ generalizes Theorem 32 to deliver more general a free theorem whose conclusion is $\mathit{fold}_H\, B \circ G\,\mu H\,\mathit{in}_H = G\,[\![\emptyset;\alpha \vdash K]\!]^{\mathsf{Set}}\, B$.

7 CONCLUSION AND DIRECTIONS FOR FUTURE WORK

Term-level fixpoints from Daniel: In order to interpret term-level fixpoints, we would need to add some sort of domain structure to the sets and relations which are currently being used to interpret types. With this added structure, we should be able to interpret recursive function definitions in the usual way (as least fixed points with respect to the domain structure). An obvious way to add this structure might be to replace Set with the category of omega-CPOs in our model, but this category is not locally finitely presentable as is required.

We don't need to verify exuistence of initial algebras (as is usual) since this is built into our system.

We have forall-types but not all.

Can do everything in abstract locally presentable cartesian closed category.

Give definitions for arb lpccc, but compute free theorems in Set/Rel.

Future Work (in progress): extend calculus to GADTs

Add more polymorphisms (all foralls), even though most free theorems only use one level (or maybe two, like short cut).

Couldn't do this before [Johann and Polonsky 2019] because we didn't know before that nested types (and then some) always have well-defined interpretations in locally finitely presentable categories like Set and Rel. In fact, could extend results here to "locally presentable fibrations", where these are yet to be defined, but would at least have locally presentable base and total categories with the locally presentable structure preserved by the fibration and appropriate reflection of the total category in the base (as in Alex's effects paper?).

fixed points at term level ala Pitts:

I'm not sure if we actually want to do this or not (because not all fixed points will necessarily exist, and this will have consequences, and because of the restriction to strict functions it will entail and the fact that I haven't yet thought about how those will thread through our calculus), but another idea we might consider is adding fixed points at the term level. This is done for System F in Wadler's section 7. [It is also done in Pitts' Parametric Polymorphism and Operational Equivalence paper, where the term introduction rule is SEE SOURCE

But of course the model there is entirely operational rather than categorical.]

On the other hand, check out the last paragraph of Wadler's paper, which suggests that (at least in 1989) there was some interest in calculi like ours:

"The desire to derive theorems from types therefore suggests that it would be valuable to explore programming languages that prohibit recursion or only allow its restricted use. In theory, this is well understood; we have already noted that any computable function that is provably total in second-order Peano arithmetic can be defined in the pure polymorphic lambda calculus, without using the fixpoint as a primitive. However, practical languages based on this notion remain terra incognita."

Of course we are not working in System F, but perhaps the same desire applies here, and we do get the same kinds of theorems as Wadler, many just as a consequence of our interpretation, without invoking parametricity.

In any case, I guess Wadler could be cited as justifying restricting our attention to our calculus since it could be seen as a way to have a practical language that ensures that all programs are terminating.

REFERENCES

J. Adámek and J. Rosický. 1994. *Locally Presentable and Accessible Categories.* Cambridge University Press.

R. Atkey. 2012. Relational Parametricity for Higher Kinds. In *Computer Science Logic.* 46–61.

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

R. Bird and L. Meertens. 1998. Nested datatypes. In *Mathematics of Program Construction.* 52–67.

L. Birkedal and R. E. Møgelberg. 2005. Categorical models for Abadi and Plotkin's logic for parametricity. *Mathematical Structures in Computer Science* 15 (2005), 709–772.

B. Dunphy and U. Reddy. 2004. Parametric limits. In *Logic in Computer Science.*

N. Ghani, P. Johann, Fr. Nordvall Forsberg, F. Orsanigo, and T. Revell. 2015. Bifibrational Functorial Semantics for Parametric Polymorphism. In *Mathematical Foundations of Program Semantics.* 165–181.

A. Gill, J. Launchbury, and S.L. Peyton Jones. 1993. A short cut to deforestation. In *Functional Programming Languages and Computer Architecture, Proceedings.* 223–232.

J.-Y. Girard, P. Taylor, and Y. Lafont. 1989. *Proofs and Types.* Cambridge University Press.

R. Hasegawa. 1994. Categorical data types in parametric polymorphism. *Mathematical Structures in Computer Science* 4 (1994), 71–109.

B. Jacobs. 1999. *Categorical Logic and Type Theory.* Elsevier.

P. Johann. 2002. A Generalization of Short-Cut Fusion and Its Correctness Proof. *Higher-Order and Symbolic Computation* 15 (2002), 273–300.

P. Johann. 2003. Short cut fusion is correct. *Journal of Functional Programming* 13 (2003), 797–814.

P. Johann and N. Ghani. 2010. Haskell Programming with Nested Types: A Principled Approach. *Higher-Order and Symbolic Computation* 22(2) (2010), 155–189.

P. Johann and A. Polonsky. 2019. Higher-kinded Data Types: Syntax and Semantics. In *Logic in Computer Science.* 1–13. https://doi.org/10.1109/LICS.2019.8785657

P. Johann and A. Polonsky. 2020. Deep Induction: Induction Rules for (Truly) Nested Types. In *Foundations of Software Science and Computation Structures.*

Q. Ma and J. C. Reynolds. 1992. Types, abstractions, and parametric polymorphism, part 2. In *Mathematical Foundations of Program Semantics.* 1–40.

C. Okasaki. 1999. *Purely Functional Data Structures.* Cambridge University Press.

A. Pitts. 1998. Parametric polymorphism, recursive types, and operational equivalence. (1998).

A. Pitts. 2000. Parametric polymorphism and operational equivalence. *Mathematical Structures in Computer Science* 10 (2000), 1–39.

J. C. Reynolds. 1983. Types, abstraction, and parametric polymorphism. *Information Processing* 83(1) (1983), 513–523.

E. Robinson and G. Rosolini. 1994. Reflexive graphs and parametric polymorphism. In *Logic in Computer Science.* 364–371.

P. Walder. 1989. Theorems for free!. In *Functional Programming Languages and Computer Architecture, Proceedings.* 347–359.