Pre-logical Relations*

Furio Honsell[1,2] and Donald Sannella[1]

[1] Laboratory for Foundations of Computer Science, University of Edinburgh
[2] Dipartimento di Matematica e Informatica, Università di Udine

Abstract. We study a weakening of the notion of logical relations, called *pre-logical relations*, that has many of the features that make logical relations so useful as well as further algebraic properties including composability. The basic idea is simply to require the reverse implication in the definition of logical relations to hold only for pairs of functions that are expressible by the same lambda term. Pre-logical relations are the minimal weakening of logical relations that gives composability for extensional structures and simultaneously the most liberal definition that gives the Basic Lemma. The use of pre-logical relations in place of logical relations gives an improved version of Mitchell's representation independence theorem which characterizes observational equivalence for all signatures rather than just for first-order signatures. Pre-logical relations can be used in place of logical relations to give an account of data refinement where the fact that pre-logical relations compose explains why stepwise refinement is sound.

Keywords: Logical relations, typed lambda calculus, semantics, representation independence, data refinement.

1 Introduction

Logical relations are structure-preserving relations between models of typed lambda calculus.

Definition 1.1. *Let $\mathcal{A}$ and $\mathcal{B}$ be Σ-applicative structures. A* logical relation $\mathcal{R}$ over $\mathcal{A}$ and $\mathcal{B}$ *is a family of relations $\{R^\sigma \subseteq [\![\sigma]\!]^\mathcal{A} \times [\![\sigma]\!]^\mathcal{B}\}_{\sigma \in Types(B)}$ such that:*

- $R^{\sigma\to\tau}(f,g)$ *iff* $\forall a \in [\![\sigma]\!]^\mathcal{A}.\forall b \in [\![\sigma]\!]^\mathcal{B}.R^\sigma(a,b) \Rightarrow R^\tau(App_\mathcal{A}\, f\, a, App_\mathcal{B}\, g\, b)$.
- $R^\sigma([\![c]\!]^\mathcal{A}, [\![c]\!]^\mathcal{B})$ *for every term constant $c : \sigma$ in Σ.*

Logical relations are used extensively in the study of typed lambda calculus and have applications outside lambda calculus, for example to abstract interpretation [Abr90] and data refinement [Ten94]. A good reference for logical relations is [Mit96]. An important but more difficult reference is [Sta85].

The Basic Lemma is the key to many of the applications of logical relations. It says that any logical relation over $\mathcal{A}$ and $\mathcal{B}$ relates the interpretation of each lambda term in $\mathcal{A}$ to its interpretation in $\mathcal{B}$.

Lemma 1.2 (Basic Lemma). *Let $\mathcal{R}$ be a logical relation over Henkin models $\mathcal{A}$ and $\mathcal{B}$. Then for all Γ-environments $\eta_\mathcal{A}, \eta_\mathcal{B}$ such that $R^\Gamma(\eta_\mathcal{A}, \eta_\mathcal{B})$ and every term $\Gamma \triangleright M : \sigma$, $R^\sigma([\![\Gamma \triangleright M : \sigma]\!]^\mathcal{A}_{\eta_\mathcal{A}}, [\![\Gamma \triangleright M : \sigma]\!]^\mathcal{B}_{\eta_\mathcal{B}})$.* □

* This is an extended version of a paper with the same title that appeared in *Proc. CSL'99*, Springer LNCS (1999).

($R^\Gamma(\eta_\mathcal{A}, \eta_\mathcal{B})$ refers to the obvious extension of $\mathcal{R}$ to environments, see page 5 below.)

As structure-preserving relations, logical relations resemble familiar algebraic concepts like homomorphisms and congruence relations but they lack some of the convenient properties of such concepts. In particular, the composition of two logical relations is not in general a logical relation. This calls into question their application to data refinement at least, where one would expect composition to provide an account of stepwise refinement.

We propose a weakening of the notion of logical relations called *pre-logical relations* (Sect. 3) that has many of the features that make logical relations so useful — in particular, the Basic Lemma still holds for pre-logical relations (Lemma 4.1) — but having further algebraic properties including composability (Prop. 5.6). The basic idea is simply to require the reverse implication in the definition of logical relations to hold only for pairs of functions that are expressible by the same lambda term. Pre-logical relations turns out to be the minimal weakening of logical relations that gives composability for extensional structures (Corollary 7.2) and simultaneously the most liberal definition that gives the Basic Lemma. Pre-logical predicates (the unary case of pre-logical relations) coincide with sets that are invariant under Kripke logical relations with varying arity as introduced by Jung and Tiuryn [JT93] (Prop. 6.2). The use of pre-logical relations in place of logical relations gives an improved version of Mitchell's representation independence theorem (Corollaries 8.5 and 8.6 to Theorem 8.4) which characterizes observational equivalence for all signatures rather than just for first-order signatures. Pre-logical relations can be used in place of logical relations in Tennent's account of data refinement in [Ten94] and the fact that pre-logical relations compose explains why stepwise refinement is sound.

Many applications of logical relations follow a standard pattern where the result comes directly from the Basic Lemma once an appropriate logical relation has been defined. Some results in the literature follow similar lines in the sense that a type-indexed family of relations is defined by induction on types and a proof like that of the Basic Lemma is part of the construction, but the family of relations defined is not logical. Examples can be found in Plotkin's and Jung and Tiuryn's lambda-definability results using I-relations [Plo80] and Kripke logical relations with varying arity [JT93] respectively, and Gandy's proof of strong normalization using hereditarily strict monotonic functionals [Gan80]. In each of these cases, the family of relations involved turns out to be a pre-logical relation (Example 3.12, Sect. 6 and Example 3.13) which allows the common pattern to be extended to these cases as well. Since pre-logical relations are more general than logical relations and variants like I-relations, they provide a framework within which these different classes can be compared. Here we begin by studying and comparing their closure properties (Prop. 5.7) with special attention to closure under composition.

The definition of pre-logical relations is not new. In [Sch87], Schoett uses a first-order version of algebraic relations which he calls *correspondences*, and he conjectures (p. 281) that for Henkin models, what we have called pre-logical relations (formulated as in Prop. 3.3) would be closed under composition and yield the Basic Lemma. In [Mit90], Mitchell makes the same suggestion, referring to Schoett and also crediting Abramsky and Plotkin, but as an assertion rather than a conjecture. The idea is not developed any further. An independent but apparently equivalent definition of pre-logical relations over cartesian closed categories is given in [PPS98] where they are called *lax logical relations*. It is shown that these compose and that the Basic Lemma holds, and an axiomatic account is provided. Earlier, a closely related notion called *L-relations* was defined in [KOPTT97] and shown to compose. Another related paper is [Rob96]. There appears to be no previous work on pre-logical relations that goes beyond observing that they compose and that the Basic Lemma holds. Another difference to [PPS98] and [KOPTT97] is that our treatment is elementary rather than categorical, and covers also combinatory logics.

2 Syntax and Semantics

We begin with $\lambda^{\rightarrow}$, the simply-typed lambda calculus having $\rightarrow$ as the only type constructor. Other type constructors will be considered in Sect. 10. We follow the terminology in [Mit96] for the most part, with slightly different notation.

Definition 2.1. *The set $Types(B)$ of* types *over a set B of* base types *(or* type constants*) is given by the grammar $\sigma ::= b \mid \sigma \rightarrow \sigma$ where b ranges over B. A* signature *Σ consists of a set B of type constants and a collection C of typed* term constants *$c : \sigma$.*

Let $\Sigma = \langle B, C \rangle$ be a signature. We assume familiarity with the usual notions of *context* $\Gamma = x_1{:}\sigma_1, \ldots, x_n{:}\sigma_n$ and *Σ-term* M of type σ over a context Γ, written $\Gamma \rhd M : \sigma$, with the meta-variable t reserved for lambda-free Σ-terms. When we need to make Σ explicit we write $\Gamma \rhd_{\Sigma} M : \sigma$. If Γ is empty then we write simply $M : \sigma$. Capture-avoiding substitution $[N/x]M$ is as usual.

Definition 2.2. *A* Σ-applicative structure *$\mathcal{A}$ consists of:*

- *a* carrier *set $[\![\sigma]\!]^{\mathcal{A}}$ for each $\sigma \in Types(B)$;*
- *a function $App_{\mathcal{A}}^{\sigma,\tau} : [\![\sigma \rightarrow \tau]\!]^{\mathcal{A}} \rightarrow [\![\sigma]\!]^{\mathcal{A}} \rightarrow [\![\tau]\!]^{\mathcal{A}}$ for each $\sigma, \tau \in Types(B)$;*
- *an element $[\![c]\!]^{\mathcal{A}} \in [\![\sigma]\!]^{\mathcal{A}}$ for each term constant $c : \sigma$ in Σ.*

We drop the subscripts and superscripts when they are determined by the context, and we sometimes abbreviate $App_{\mathcal{A}}^{\sigma,\tau}\, f\, x$ as $f\, x$. Two elements $f, g \in [\![\sigma \rightarrow \tau]\!]^{\mathcal{A}}$ are said to be extensionally equal *if $App_{\mathcal{A}}^{\sigma,\tau}\, f\, x = App_{\mathcal{A}}^{\sigma,\tau}\, g\, x$ for every $x \in [\![\sigma]\!]^{\mathcal{A}}$. A Σ-applicative structure is* extensional *when extensional equality coincides with identity.*

A Σ-combinatory algebra *is a Σ-applicative structure $\mathcal{A}$ that has elements $K_{\mathcal{A}}^{\sigma,\tau} \in [\![\sigma \rightarrow (\tau \rightarrow \sigma)]\!]^{\mathcal{A}}$ and $S_{\mathcal{A}}^{\rho,\sigma,\tau} \in [\![(\rho \rightarrow \sigma \rightarrow \tau) \rightarrow (\rho \rightarrow \sigma) \rightarrow \rho \rightarrow \tau]\!]^{\mathcal{A}}$ for each $\rho, \sigma, \tau \in Types(B)$ satisfying the equations $K_{\mathcal{A}}^{\sigma,\tau}\, x\, y = x$ and $S_{\mathcal{A}}^{\rho,\sigma,\tau}\, x\, y\, z = (x\, z)(y\, z)$.*

An extensional combinatory algebra is called a Henkin model*. An applicative structure $\mathcal{A}$ is a* full type hierarchy *when $[\![\sigma \rightarrow \tau]\!]^{\mathcal{A}} = [\![\sigma]\!]^{\mathcal{A}} \rightarrow [\![\tau]\!]^{\mathcal{A}}$ (the full set-theoretic function space) for every $\sigma, \tau \in Types(B)$ and then it is obviously a Henkin model.*

In a combinatory algebra, we can extend the definition of lambda-free Σ-terms by allowing them to contain S and K; we call these *combinatory Σ-terms.*

A *Γ-environment* $\eta_{\mathcal{A}}$ assigns elements of an applicative structure $\mathcal{A}$ to variables, with $\eta_{\mathcal{A}}(x) \in [\![\sigma]\!]^{\mathcal{A}}$ for $x : \sigma$ in Γ. A lambda-free Σ-term $\Gamma \rhd t : \sigma$ is interpreted in a Σ-applicative structure $\mathcal{A}$ under a Γ-environment $\eta_{\mathcal{A}}$ in the obvious way, written $[\![\Gamma \rhd t : \sigma]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}$, and this extends immediately to an interpretation of combinatory Σ-terms in combinatory algebras by interpreting K and S as $K_{\mathcal{A}}$ and $S_{\mathcal{A}}$. If t is closed then we write simply $[\![t : \sigma]\!]^{\mathcal{A}}$.

There are various ways of interpreting terms containing lambda abstraction in a combinatory algebra by "compiling" them to combinatory terms so that outermost β holds (see Prop. 2.4 below for what we mean by "outermost β"). In Henkin models, all these compilations yield the same result.

An axiomatic approach to interpreting lambda abstraction requires an applicative structure equipped with an interpretation function that satisfies certain minimal requirements — cf. the notion of *acceptable meaning function* in [Mit96].

Definition 2.3. *A* lambda Σ-applicative structure *consists of a Σ-applicative structure $\mathcal{A}$ together with a function $[\![\cdot]\!]^{\mathcal{A}}$ that maps any term $\Gamma \rhd M : \sigma$ and Γ-environment $\eta_{\mathcal{A}}$ over $\mathcal{A}$ to an element of $[\![\sigma]\!]^{\mathcal{A}}$, such that:*

- $[\![\Gamma \rhd x : \sigma]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}} = \eta_{\mathcal{A}}(x)$
- $[\![\Gamma \rhd c : \sigma]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}} = [\![c]\!]^{\mathcal{A}}$
- $[\![\Gamma \rhd M\,N : \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}} = App_{\mathcal{A}}\,[\![\Gamma \rhd M : \sigma \to \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}\,[\![\Gamma \rhd N : \sigma]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}$
- $[\![\Gamma \rhd \lambda x{:}\sigma.M : \sigma \to \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}} = [\![\Gamma \rhd \lambda y{:}\sigma.[y/x]M : \sigma \to \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}$ *provided* $y \notin \Gamma$
- $[\![\Gamma \rhd M : \sigma]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}} = [\![\Gamma \rhd M : \sigma]\!]^{\mathcal{A}}_{\eta'_{\mathcal{A}}}$ *provided $\eta'_{\mathcal{A}}$ is a Γ-environment such that $\eta_{\mathcal{A}}(x) = \eta'_{\mathcal{A}}(x)$ for all $x \in \Gamma$*
- $[\![\Gamma, x{:}\sigma \rhd M : \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}} = [\![\Gamma \rhd M : \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}$
- $[\![\Gamma, x{:}\sigma \rhd M : \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}[x \mapsto [\![\Gamma \rhd N:\sigma]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}]} = [\![\Gamma \rhd [N/x]M : \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}$

The relationship between lambda applicative structures and combinatory algebras is as follows.

Proposition 2.4. *A lambda applicative structure $\mathcal{A}$ such that $App_{\mathcal{A}}[\![\Gamma \rhd \lambda x{:}\sigma.M : \sigma \to \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}\ a = [\![\Gamma, x{:}\sigma \rhd M : \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}[x \mapsto a]}$ amounts to a combinatory algebra, and vice versa.*

Proof. $\Leftarrow$: *We define $[\![\cdot]\!]^{\mathcal{A}}$ via the standard compilation of lambda terms using K and S to combinatory terms.*

$\Rightarrow$: *$K^{\sigma,\tau}_{\mathcal{A}}$ and $S^{\rho,\sigma,\tau}_{\mathcal{A}}$ are the interpretations of the usual lambda terms.* □

The proof of this proposition shows that the interpretation of lambda terms in combinatory algebras via compilation to combinatory terms satisfies the axioms in Def. 2.3 and the additional property in the proposition. Therefore when viewing a combinatory algebra as a lambda applicative structure, this is the interpretation function we have in mind.

3 Algebraic and Pre-logical Relations

We propose a weakening of the definition of logical relations which is closed under composition and which has most of the attractive properties of logical relations. First we change the two-way implication in the condition on functions to a one-way implication which requires preservation of the relation under application.

Definition 3.1. *Let $\mathcal{A}$ and $\mathcal{B}$ be Σ-applicative structures. An* algebraic relation $\mathcal{R}$ over $\mathcal{A}$ and $\mathcal{B}$ *is a family of relations $\{R^\sigma \subseteq [\![\sigma]\!]^{\mathcal{A}} \times [\![\sigma]\!]^{\mathcal{B}}\}_{\sigma \in \mathit{Types}(B)}$ such that:*

- *If $R^{\sigma\to\tau}(f,g)$ then $\forall a \in [\![\sigma]\!]^{\mathcal{A}}.\forall b \in [\![\sigma]\!]^{\mathcal{B}}.R^\sigma(a,b) \Rightarrow R^\tau(\mathit{App}_{\mathcal{A}}\, f\, a, \mathit{App}_{\mathcal{B}}\, g\, b)$.*
- *$R^\sigma([\![c]\!]^{\mathcal{A}}, [\![c]\!]^{\mathcal{B}})$ for every term constant $c:\sigma$ in Σ.*

In lambda applicative structures, we additionally require the relation to preserve lambda abstraction in a sense that is analogous to the definition of *admissible relation* in [Mit96]. First, we extend a family of relations $\mathcal{R} = \{R^\sigma \subseteq [\![\sigma]\!]^{\mathcal{A}} \times [\![\sigma]\!]^{\mathcal{B}}\}_{\sigma \in \mathit{Types}(B)}$ to a relation on Γ-environments: $R^\Gamma(\eta_{\mathcal{A}}, \eta_{\mathcal{B}})$ if $R^\sigma(\eta_{\mathcal{A}}(x), \eta_{\mathcal{B}}(x))$ for every $x{:}\sigma$ in Γ.

Definition 3.2. *Let $\mathcal{A}$ and $\mathcal{B}$ be lambda Σ-applicative structures. A* pre-logical relation over $\mathcal{A}$ and $\mathcal{B}$ *is an algebraic relation $\mathcal{R}$ such that given Γ-environments $\eta_{\mathcal{A}}$ and $\eta_{\mathcal{B}}$ such that $R^\Gamma(\eta_{\mathcal{A}}, \eta_{\mathcal{B}})$, and a term $\Gamma, x:\sigma \rhd M:\tau$, if $R^\sigma(a,b)$ implies $R^\tau([\![\Gamma, x:\sigma \rhd M:\tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}[x\mapsto a]}, [\![\Gamma, x:\sigma \rhd M:\tau]\!]^{\mathcal{B}}_{\eta_{\mathcal{B}}[x\mapsto b]})$ for all $a \in [\![\sigma]\!]^{\mathcal{A}}$ and $b \in [\![\sigma]\!]^{\mathcal{B}}$, then $R^{\sigma\to\tau}([\![\Gamma \rhd \lambda x{:}\sigma.M : \sigma\to\tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}, [\![\Gamma \rhd \lambda x{:}\sigma.M : \sigma\to\tau]\!]^{\mathcal{B}}_{\eta_{\mathcal{A}}})$.*

This amounts to defining pre-logical relations as simply the class of relations that make the Basic Lemma hold, as we shall see in Lemma 4.1 below. (Indeed, since the Basic Lemma for pre-logical relations is an equivalence rather than a one-way implication, an alternative at this point would be to take the conclusion of the Basic Lemma itself as the definition of pre-logical relations.)

A simpler and therefore more appealing definition is obtained if we consider combinatory algebras, where the requirement above boils down to preservation of S and K:

Proposition 3.3. *Let $\mathcal{A}$ and $\mathcal{B}$ be Σ-combinatory algebras. An algebraic relation $\mathcal{R}$ over $\mathcal{A}$ and $\mathcal{B}$ is pre-logical iff $R(S^{\rho,\sigma,\tau}_{\mathcal{A}}, S^{\rho,\sigma,\tau}_{\mathcal{B}})$ and $R(K^{\sigma,\tau}_{\mathcal{A}}, K^{\sigma,\tau}_{\mathcal{B}})$ for all $\rho,\sigma,\tau \in \mathit{Types}(B)$.*

Proof. *Directly from the definitions.* □

If we incorporate S and K into the signature Σ, then pre-logical relations are just algebraic relations on combinatory algebras. One way of understanding the definition of pre-logical relations is that the reverse implication in the definition of logical relations is required to hold only for pairs of functions that are expressible by the same lambda term. For combinatory algebras these are exactly the pairs of functions that are denoted by the same combinatory term, and thus this requirement is captured by requiring the relation to contain S and K.

The use of the combinators S and K in the above proposition is in some sense arbitrary: the same result would be achieved by taking any other combinatory basis and changing the definition of combinatory algebra and the interpretation function accordingly. It would be straightforward to modify the definitions to accommodate other variants of lambda calculus, for instance λ_I (where in $\lambda x{:}\sigma.M$, the term M is required to contain x) for which a combinatory basis is

B, C, I, S, or linear lambda calculi. For languages that include recursion, such as PCF, one would add a Y combinator.

As usual, the binary case of algebraic resp. pre-logical relations over $\mathcal{A}$ and $\mathcal{B}$ is derived from the unary case of *algebraic* resp. *pre-logical predicates* for the product structure $\mathcal{A} \times \mathcal{B}$. For instance:

Definition 3.4. *Let $\mathcal{A}$ be a Σ-applicative structure. An* algebraic predicate $\mathcal{P}$ over $\mathcal{A}$ *is a family of predicates $\{P^\sigma \subseteq [\![\sigma]\!]^{\mathcal{A}}\}_{\sigma \in Types(B)}$ such that:*

- *If $P^{\sigma \to \tau}(f)$ then $\forall a \in [\![\sigma]\!]^{\mathcal{A}}. P^\sigma(a) \Rightarrow P^\tau(App_{\mathcal{A}}\ f\ a)$.*
- *$P^\sigma([\![c]\!]^{\mathcal{A}})$ for every term constant $c : \sigma$ in Σ.*

For most results about pre-logical relations below there are corresponding results about pre-logical predicates and about algebraic relations and predicates over applicative structures. We omit these for lack of space. Similar comments apply to n-ary relations for $n > 2$.

Here is one result about pre-logical predicates that does not generalize in the obvious way to pre-logical relations:

Proposition 3.5. *A pre-logical predicate over a reachable lambda applicative structure $\mathcal{A}$ is a logical predicate on $\mathcal{A}$. (A lambda Σ-applicative structure $\mathcal{A}$ is* reachable *if for any $a \in [\![\sigma]\!]^{\mathcal{A}}$ there is a closed Σ-term $M : \sigma$ such that $a = [\![M : \sigma]\!]^{\mathcal{A}}$.)*

Proof. *By a straightforward induction, or alternatively as a corollary of Lemma 4.2 below.* □

A pre-logical relation over reachable lambda applicative structures $\mathcal{A}$ and $\mathcal{B}$ is not necessarily a logical relation over $\mathcal{A}$ and $\mathcal{B}$, because what is required to apply Prop. 3.5 is reachability for the product structure $\mathcal{A} \times \mathcal{B}$: for any $\langle a, b\rangle \in [\![\sigma]\!]^{\mathcal{A}\times\mathcal{B}}$ there needs to be a closed Σ-term $M : \sigma$ such that $\langle a, b\rangle = [\![M : \sigma]\!]^{\mathcal{A}\times\mathcal{B}} = \langle [\![M : \sigma]\!]^{\mathcal{A}}, [\![M : \sigma]\!]^{\mathcal{B}}\rangle$.

The fact that pre-logicality is strictly weaker than logicality is demonstrated by the following examples which also provide a number of general methods for defining pre-logical relations.

Example 3.6. For any signature Σ and lambda Σ-applicative structure, the predicate $\mathcal{P}$ defined by

$$P^\sigma(v) \Leftrightarrow v \text{ is the value of a closed } \Sigma\text{-term } M : \sigma$$

is a pre-logical predicate over $\mathcal{A}$. (In fact, $\mathcal{P}$ is the *least* such — see Prop. 5.9 below.) Now, consider the signature Σ containing the type constant nat and term constants $0 : nat$ and $succ : nat \to nat$ and let $\mathcal{A}$ be the full type hierarchy over $\mathbb{N}$ where 0 and $succ$ have their usual interpretations. $\mathcal{P}$ is not a logical predicate over $\mathcal{A}$: any function $f \in [\![nat \to nat]\!]^{\mathcal{A}}$, including functions that are not lambda definable, takes values in P to values in P and so must itself be in P. □

Example 3.7. The identity relation on a lambda applicative structure is a pre-logical relation but it is logical iff the structure is extensional. □

Example 3.8. A *Σ-homomorphism* between lambda Σ-applicative structures $\mathcal{A}$ and $\mathcal{B}$ is a type-indexed family of functions $\{h^\sigma : [\![\sigma]\!]^{\mathcal{A}} \to [\![\sigma]\!]^{\mathcal{B}}\}_{\sigma \in Types(B)}$ such that for any term constant $c : \sigma$ in Σ, $h^\sigma([\![c]\!]^{\mathcal{A}}) = [\![c]\!]^{\mathcal{B}}$, $h^\tau(App_{\mathcal{A}}^{\sigma,\tau}\ f\ a) = App_{\mathcal{B}}^{\sigma,\tau}\ h^{\sigma\to\tau}(f)\ h^\sigma(a)$ and $h^{\sigma\to\tau}([\![\Gamma \triangleright \lambda x{:}\sigma.M : \sigma \to \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}) = [\![\Gamma \triangleright \lambda x{:}\sigma.M : \sigma \to \tau]\!]^{\mathcal{B}}_{h(\eta_{\mathcal{A}})}$ where $h(\eta_{\mathcal{A}}) = \{x \mapsto h^\sigma(\eta_{\mathcal{A}}(x))\}$ for all $x{:}\sigma$ in Γ. Any Σ-homomorphism is a pre-logical relation. In particular, interpretation of terms in a lambda applicative structure with respect to an environment, viewed as a relation from the lambda applicative structure of terms, is a pre-logical relation but is not in general a logical relation. □

Example 3.9. Pre-logical predicates over a lambda applicative structure $\mathcal{A}$ are the natural notion of *substructures* of $\mathcal{A}$ since they require closure under the available operations, see Prop. 4.2 below. When considering extensional structures, we additionally require substructures to be extensional. Logical predicates are unnecessarily constrained for this purpose. □

Example 3.10. Let $\mathcal{A}$ and $\mathcal{B}$ be lambda applicative structures and define $R^\sigma \subseteq [\![\sigma]\!]^{\mathcal{A}} \times [\![\sigma]\!]^{\mathcal{B}}$ by $R^\sigma(a,b)$ for $a \in [\![\sigma]\!]^{\mathcal{A}}, b \in [\![\sigma]\!]^{\mathcal{B}}$ iff there is a closed term $M : \sigma$ such that $[\![M : \sigma]\!]^{\mathcal{A}} = a$ and $[\![M : \sigma]\!]^{\mathcal{B}} = b$. This is a pre-logical relation but it is not in general a logical relation. Generalizing: the inverse of any pre-logical relation is obviously pre-logical and according to Prop. 5.6 below the composition of any two pre-logical relations is pre-logical. Then observe that the above relation is just the composition of closed term interpretation in $\mathcal{B}$ (which is pre-logical according to Example 3.8) and the inverse of closed term interpretation in $\mathcal{A}$. □

Example 3.11. Let $\mathcal{A}_1, \ldots, \mathcal{A}_m$ be lambda applicative structures. There are several ways of generating an n-ary pre-logical relation from an m-ary one $\mathcal{R} \subseteq \mathcal{A}_1 \times \cdots \times \mathcal{A}_m$ when $n < m$. *Projection* of any n-tuple of components yields an n-ary pre-logical relation, for example $\pi_{1,2}(\mathcal{R}) \subseteq \mathcal{A}_1 \times \mathcal{A}_2$. Another notation for this is $\exists$, defined as follows: $(\exists\mathcal{R})^\sigma = \{\langle a_2, \ldots, a_m\rangle \mid \exists a_1 \in [\![\sigma]\!]^{\mathcal{A}_1}.\langle a_1, a_2, \ldots, a_m\rangle \in \mathcal{R}^\sigma\}$. The dual of this is $\forall$, which is another way of reducing the arity of a pre-logical relation: $(\forall\mathcal{R})^\sigma = \{\langle a_2, \ldots, a_m\rangle \mid \forall a_1 \in [\![\sigma]\!]^{\mathcal{A}_1}.\langle a_1, a_2, \ldots, a_m\rangle \in \mathcal{R}^\sigma\}$. Finally, if $\mathcal{A}_i = \mathcal{A}_j$ for some $i, j \leq m$ then we can take the subset of tuples in $\mathcal{R}$ having the same values in positions i and j: $(\mathit{filter}\ \mathcal{R})^\sigma = \{\langle a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_m\rangle \mid \langle a_1, \ldots, a_m\rangle \in \mathcal{R}^\sigma \text{ and } a_i = a_j\}$. This amounts to a projection of the intersection of $\mathcal{R}$ with the cartesian product of the $\mathcal{A}_l$'s for $l \neq i, j$ and the diagonal over $\mathcal{A}_i \times \mathcal{A}_j$. All n-ary pre-logical relations can be generated using any of these: to obtain $\mathcal{S}$, apply any of the above to $\mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_1$ or alternatively, for the case of projection and filtering, to $\{\langle a_1, \ldots, a_n, a_1, \ldots, a_1\rangle \mid \langle a_1, \ldots, a_n\rangle \in \mathcal{S}\}$. If $\mathcal{R}$ is a logical relation then so is $\forall\mathcal{R}$, but the projection and filtering of $\mathcal{R}$ are not logical relations in general. □

Example 3.12. Plotkin's *I-relations* [Plo80] give rise to pre-logical relations. The family of relations on the full type hierarchy consisting of the tuples which are in a given I-relation at a given world (alternatively, at all worlds) is a pre-logical relation which is not in general a logical relation. □

A related example concerning Kripke logical relations with varying arity [JT93] is postponed to Sect. 6 to allow the reader to first become more familiar with pre-logical relations.

Example 3.13. Let $\mathcal{A}$ be an applicative structure. Given order relations R^b on $[\![b]\!]^{\mathcal{A}}$ for each base type b, we can define the *hereditarily monotonic functionals* as the equivalence classes of those elements of $\mathcal{A}$ which are self-related with respect to the following inductively defined family of relations on $\mathcal{A} \times \mathcal{A}$:

$$\begin{aligned} R^{\sigma\to\tau}(f,g) \text{ iff } \forall a \in [\![\sigma]\!]^{\mathcal{A}}.\forall b \in [\![\sigma]\!]^{\mathcal{A}}.R^{\sigma}(a,b) \Rightarrow (&R^{\tau}(App_{\mathcal{A}}\ f\ a, App_{\mathcal{A}}\ g\ a) \\ \wedge &R^{\tau}(App_{\mathcal{A}}\ f\ a, App_{\mathcal{A}}\ f\ b) \\ \wedge &R^{\tau}(App_{\mathcal{A}}\ g\ a, App_{\mathcal{A}}\ g\ b)) \end{aligned}$$

(This defines simultaneously at each type both the class of functionals we are interested in and the order relation itself.) This method defines a pre-logical relation which is not in general a logical relation.

Gandy's *hereditarily strict monotonic functionals* [Gan80] can be defined using the above technique with just a small modification of the clause for functionals.

$$\begin{aligned} R^{\sigma\to\tau}(f,g) \text{ iff } &\forall a \in [\![\sigma]\!]^{\mathcal{A}}.\forall b \in [\![\sigma]\!]^{\mathcal{A}}. \\ &R^{\sigma}(a,b) \Rightarrow (f \neq g \Rightarrow (R^{\tau} \setminus \Delta^{\tau})(App_{\mathcal{A}}\ f\ a, App_{\mathcal{A}}\ g\ a) \\ &\qquad\qquad\quad \wedge \\ &\qquad\qquad\quad a \neq b \Rightarrow ((R^{\tau} \setminus \Delta^{\tau})(App_{\mathcal{A}}\ f\ a, App_{\mathcal{A}}\ f\ b) \\ &\qquad\qquad\qquad\qquad \wedge (R^{\tau} \setminus \Delta^{\tau})(App_{\mathcal{A}}\ g\ a, App_{\mathcal{A}}\ g\ b))) \end{aligned}$$

Again we have a pre-logical relation (with respect to the language of λ_I) which is not in general a logical relation. We can define the continuous functionals, as used in models of PCF, using a similar pattern. □

4 The Basic Lemma

We will now consider the extension of the Basic Lemma to pre-logical relations. In contrast to Lemma 1.2, we get a two-way implication which says that the requirements on pre-logical relations are exactly strong enough to ensure that the Basic Lemma holds. The reverse implication fails for logical relations as Example 3.6 shows (for logical predicates).

Lemma 4.1 (Basic Lemma for pre-logical relations). *Let $\mathcal{R} = \{R^{\sigma} \subseteq [\![\sigma]\!]^{\mathcal{A}} \times [\![\sigma]\!]^{\mathcal{B}}\}_{\sigma \in Types(B)}$ be a family of relations over lambda Σ-applicative structures $\mathcal{A}$ and $\mathcal{B}$. Then $\mathcal{R}$ is a pre-logical relation iff for all Γ-environments $\eta_{\mathcal{A}}, \eta_{\mathcal{B}}$ such that $R^{\Gamma}(\eta_{\mathcal{A}}, \eta_{\mathcal{B}})$ and every Σ-term $\Gamma \rhd M : \sigma$, $R^{\sigma}([\![\Gamma \rhd M : \sigma]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}, [\![\Gamma \rhd M : \sigma]\!]^{\mathcal{B}}_{\eta_{\mathcal{B}}})$.*

Proof. ⇒: *The proof is by induction on the structure of M. For variables, we use the assumption that $R^{\Gamma}(\eta_{\mathcal{A}}, \eta_{\mathcal{B}})$. For constants, we use the fact that pre-logical relations are required to respect constants. For an application $\Gamma \rhd M\ N : \tau$, we use the inductive hypothesis for M and N and the fact that pre-logical relations are closed under application. As for $\Gamma \rhd \lambda x{:}\sigma.M : \sigma \to \tau$, by the induction hypothesis we know that for all $\Gamma, x{:}\sigma$-environments η'_A, η'_B such that*

$R^{\Gamma,x:\sigma}(\eta'_{\mathcal{A}}, \eta'_{\mathcal{B}})$ *we have* $R^{\sigma}([\![\Gamma, x{:}\sigma \rhd M : \sigma]\!]^{\mathcal{A}}_{\eta'_{\mathcal{A}}}, [\![\Gamma, x{:}\sigma \rhd M : \sigma]\!]^{\mathcal{B}}_{\eta'_{\mathcal{B}}})$. *If* $R^{\sigma}(a,b)$ *then applying this to* $\eta'_{\mathcal{A}} = \eta_{\mathcal{A}}[x \mapsto a]$ *and* $\eta'_{\mathcal{B}} = \eta_{\mathcal{B}}[x \mapsto b]$ *and taking the condition in the definition of pre-logical relations gives the desired result.*

$\Leftarrow$*: The first condition of algebraic relations follows by taking* M *to be* $x\ y$ *and* $\eta_{\mathcal{A}} = \{x \mapsto f, y \mapsto a\}$ $\eta_{\mathcal{B}} = \{x \mapsto g, y \mapsto b\}$ *and the second condition for a term constant* c *follows by taking* M *to be* c. *The additional condition for pre-logical relations holds* a fortiori. □

The "only if" direction of this result is the analogue in our setting of the general version of the Basic Lemma in [Mit96], but where $\mathcal{R}$ is only required to be pre-logical. For combinatory algebras, the case of lambda abstraction is handled via conversion to an equivalent combinatory term; comparing this proof with the standard proof of the Basic Lemma exposes the main idea a little more clearly.

If one applies the Basic Lemma for pre-logical relations to Henkin models, the "only if" part of the result is exactly the usual formulation (Lemma 1.2 above), except that $\mathcal{R}$ is only required to be pre-logical.

The Basic Lemma is intimately connected with the concept of lambda definability. This is most apparent in the unary case:

Lemma 4.2 (Basic Lemma for pre-logical predicates). *Let* $\mathcal{P} = \{P^{\sigma} \subseteq [\![\sigma]\!]^{\mathcal{A}}\}_{\sigma \in Types(B)}$ *be a family of predicates over a lambda* Σ*-applicative structure* $\mathcal{A}$. *Then* $\mathcal{P}$ *is a pre-logical predicate iff it is closed under lambda definability:* $P^{\Gamma}(\eta)$ *and* $\Gamma \rhd M : \sigma$ *implies* $P^{\sigma}([\![\Gamma \rhd M : \sigma]\!]^{\mathcal{A}}_{\eta})$. □

5 Properties of Pre-logical Relations

A logical relation on lambda applicative structures is pre-logical provided it is admissible in the following sense.

Definition 5.1 ([Mit96]). *A logical relation* $\mathcal{R}$ *on lambda applicative structures* $\mathcal{A}$ *and* $\mathcal{B}$ *is* admissible *if given* Γ*-environments* $\eta_{\mathcal{A}}$ *and* $\eta_{\mathcal{B}}$ *such that* $R^{\Gamma}(\eta_{\mathcal{A}}, \eta_{\mathcal{B}})$, *and terms* $\Gamma, x{:}\sigma \rhd M, N : \tau$,

$$\forall a \in [\![\sigma]\!]^{\mathcal{A}}, b \in [\![\sigma]\!]^{\mathcal{B}}. R^{\sigma}(a,b) \Rightarrow R^{\tau}([\![\Gamma, x{:}\sigma \rhd M : \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}[x\mapsto a]}, [\![\Gamma, x{:}\sigma \rhd N : \tau]\!]^{\mathcal{B}}_{\eta_{\mathcal{B}}[x\mapsto b]})$$

implies

$$\begin{aligned}\forall a \in [\![\sigma]\!]^{\mathcal{A}}, b \in [\![\sigma]\!]^{\mathcal{B}}. R^{\sigma}(a,b) \Rightarrow R^{\tau}(&App_{\mathcal{A}}\, [\![\Gamma \rhd \lambda x{:}\sigma.M : \sigma \to \tau]\!]^{\mathcal{A}}_{\eta_{\mathcal{A}}}\ a,\\ &App_{\mathcal{B}}\, [\![\Gamma \rhd \lambda x{:}\sigma.N : \sigma \to \tau]\!]^{\mathcal{B}}_{\eta_{\mathcal{B}}}\ b)\end{aligned}$$

Proposition 5.2. *Any admissible logical relation on lambda applicative structures is a pre-logical relation.*

Proof. *Admissibility plus the reverse implication in the definition of logical relations gives the property in the definition of pre-logical relations.* □

Corollary 5.3. *Any logical relation on combinatory algebras is a pre-logical relation.*

Proof. *Logical relations on combinatory algebras are always admissible. Alternatively, apply Prop. 3.3.* □

To understand why the composition of logical relations $\mathcal{R}$ over $\mathcal{A}$ and $\mathcal{B}$ and $\mathcal{S}$ over $\mathcal{B}$ and $\mathcal{C}$ might not be a logical relation, it is instructive to look at examples. When composition fails, the problem is often that the interpretation of some function type in $\mathcal{B}$ has "too few values". But even if we take logical relations over full type hierarchies, where all possible values of function types are present, composition can fail because the required "missing link" in $\mathcal{B}$ is not a function:

Example 5.4. Let Σ contain just two type constants, b and b'. Consider three full type hierarchies $\mathcal{A}, \mathcal{B}, \mathcal{C}$ which interpret b and b' as follows: $[\![b]\!]^{\mathcal{A}} = \{*\} = [\![b']\!]^{\mathcal{A}}$; $[\![b]\!]^{\mathcal{B}} = \{*\}$ and $[\![b']\!]^{\mathcal{B}} = \{\circ, \bullet\}$; $[\![b]\!]^{\mathcal{C}} = \{\circ, \bullet\} = [\![b']\!]^{\mathcal{C}}$. Let $\mathcal{R}$ be the logical relation over $\mathcal{A}$ and $\mathcal{B}$ induced by $R^b = \{\langle *, * \rangle\}$ and $R^{b'} = \{\langle *, \circ \rangle, \langle *, \bullet \rangle\}$ and let $\mathcal{S}$ be the logical relation over $\mathcal{B}$ and $\mathcal{C}$ induced by $S^b = \{\langle *, \circ \rangle, \langle *, \bullet \rangle\}$ and $S^{b'} = \{\langle \circ, \circ \rangle, \langle \bullet, \bullet \rangle\}$. $\mathcal{S} \circ \mathcal{R}$ is not a logical relation because it does not relate the identity function in $[\![b]\!]^{\mathcal{A}} \to [\![b']\!]^{\mathcal{A}}$ to the identity function in $[\![b]\!]^{\mathcal{C}} \to [\![b']\!]^{\mathcal{C}}$. The problem is that the only two functions in $[\![b]\!]^{\mathcal{B}} \to [\![b']\!]^{\mathcal{B}}$ are $\{* \mapsto \circ\}$ and $\{* \mapsto \bullet\}$, and $\mathcal{S}$ does not relate these to the identity in $\mathcal{C}$. □

Example 5.5. In the previous example, add a type constant *bool* and a term constant $c : (b \to b') \to \mathit{bool}$ to Σ. Let $[\![\mathit{bool}]\!]^{\mathcal{A}} = [\![\mathit{bool}]\!]^{\mathcal{B}} = [\![\mathit{bool}]\!]^{\mathcal{C}} = \{\mathit{true}, \mathit{false}\}$ and take $\mathcal{R}^{\mathit{bool}}$ and $\mathcal{S}^{\mathit{bool}}$ to be the identity. In each model, let the interpretation of c take constant functions to *true* and all other functions to *false*. The resulting $\mathcal{R}$ and $\mathcal{S}$ are logical relations. As before, $\mathcal{S} \circ \mathcal{R}$ is not a logical relation but now the restriction of $\mathcal{S} \circ \mathcal{R}$ to base types cannot be lifted to a logical relation either: this would relate the identity function in $[\![b]\!]^{\mathcal{A}} \to [\![b']\!]^{\mathcal{A}}$ (which is a constant function) to every function in $[\![b]\!]^{\mathcal{C}} \to [\![b']\!]^{\mathcal{C}}$, but then the constant function in $\mathcal{A}$ would be related to non-constant functions in $\mathcal{C}$, and so $[\![c]\!]^{\mathcal{A}}$ could not be related to $[\![c]\!]^{\mathcal{C}}$, otherwise *true* would be related to *false*. □

Proposition 5.6. *The composition $\mathcal{S} \circ \mathcal{R}$ of pre-logical relations $\mathcal{R}$ over $\mathcal{A}, \mathcal{B}$ and $\mathcal{S}$ over $\mathcal{B}, \mathcal{C}$ is a pre-logical relation over $\mathcal{A}, \mathcal{C}$.*

Proof. A proof from the definition is not at all straightforward, but Lemma 4.1 says that pre-logicality is equivalent to a property of relations that is obviously closed under composition. □

Obviously, closure under composition opens the way to the use of categories in which pre-logical relations are morphisms. Many properties of pre-logical relations could then be expressed using the language of category theory, but we do not pursue this here.

Composition of relations is definable in terms of product, intersection and projection:

$$\mathcal{S} \circ \mathcal{R} = \pi_{1,3}(\mathcal{A} \times \mathcal{S} \ \cap \ \mathcal{R} \times \mathcal{C})$$

Closure of pre-logical relations under these operations is a more basic property than closure under composition, and is not specific to binary relations. We have:

Proposition 5.7. *Pre-logical relations are closed under intersection, product, projection, permutation and $\forall$. Logical relations are closed under product, permutation and $\forall$ but not under intersection or projection.* □

Counterexample 5.8. To see that logical relations are not closed under intersection, consider two logical predicates $\mathcal{P}, \mathcal{P}'$ over $\mathcal{A}$ whose intersection at base types is empty. For any logical predicate $\mathcal{Q}$ over $\mathcal{A}$ which is empty at base types, it is easy to show by induction that for each type σ, either $\mathcal{Q}^\sigma = [\![\sigma]\!]^{\mathcal{A}}$ or $\mathcal{Q}^\sigma = \emptyset$. But in general $\mathcal{P} \cap \mathcal{P}'$ does not have this property.

Concerning closure under projection, recall $\mathcal{S}$ from Example 5.4 and observe that $\pi_2(\mathcal{S})$ is not a logical predicate: $\pi_2(\mathcal{S})^b = [\![b]\!]^{\mathcal{C}}$ and $\pi_2(\mathcal{S})^{b'} = [\![b']\!]^{\mathcal{C}}$ but $\pi_2(\mathcal{S})^{b \to b'}$ does not contain the identity function. □

Other classes of relations satisfy different closure properties. For instance, I-relations are obviously closed under product, permutation and $\forall$, and it is easy to see that they are also closed under intersection. They are not closed under composition by Example 5.4 *mutatis mutandis*, and since composition is definable in terms of product, intersection and projection it follows that they are not closed under projection.

A consequence of closure under intersection is that given a property P of relations that is preserved under intersection, there is always a *least* pre-logical relation satisfying P. We then have the following lambda-definability result (recall Example 3.6 above):

Proposition 5.9. *The least pre-logical predicate over a given lambda Σ-applicative structure contains exactly those elements that are the values of closed Σ-terms.* □

In a signature with no term constants, a logical relation may be constructed by defining a relation R on base types and using the definition to *"lift"* R inductively to higher types. The situation is different for pre-logical relations: there are in general many pre-logical liftings of a given R, one being of course its lifting to a logical relation (provided this gives an admissible relation). But since the property of lifting a given R is preserved under intersection, the least pre-logical lifting of R is also a well-defined relation. Similarly for the least pre-logical *extension* of a given family of relations, for any signature. Notice that lifting $\mathcal{R}$ to a logical relation is not possible in general for signatures containing higher-order term constants, see Example 5.5 (and see [Mit96] for a way of accommodating least fixed-point operators). Extension is also problematic: the cartesian product $\mathcal{A} \times \mathcal{A}$ is a logical relation that trivially extends any binary relation on $\mathcal{A}$, but this is uninteresting. Further ways of defining pre-logical relations are indicated by the examples at the end of Sect. 3 and in Sect. 6.

It is easy to see that pre-logical relations are not closed under union. And even in a signature with no term constants, the class of pre-logical relations that lift a given relation R on base types cannot be endowed with a lattice structure in general. But the only logical relation in this class is one of its maximal elements under inclusion.

6 Kripke Logical Relations with Varying Arity

In [JT93], Jung and Tiuryn give the following variant of the notion of logical relations:

Definition 6.1 ([JT93]). *Let $\mathcal{C}$ be a small category of sets and let $\mathcal{A}$ be a Henkin model. A* Kripke logical relation with varying arity *(* KLRwVA *for short) over $\mathcal{A}$ is a family of relations R^w_σ indexed by objects w of $\mathcal{C}$ and types σ of Types(B), where the elements of R^w_σ are tuples of elements from $[\![\sigma]\!]^{\mathcal{A}}$ indexed by the elements of w, such that:*

- *If $f : v \to w$ is a map in $\mathcal{C}$ and $R^w_\sigma\langle a_j\rangle_{j\in w}$ then $R^v_\sigma\langle a_{f(i)}\rangle_{i\in v}$.*
- *$R^w_{\sigma\to\tau}\langle g_j\rangle_{j\in w}$ iff $\forall f : v \to w. \forall\langle a_i\rangle_{i\in v}. R^v_\sigma\langle a_i\rangle_{i\in v} \Rightarrow R^v_\tau\langle App_{\mathcal{A}}\, g_{f(i)}\, a_i\rangle_{i\in v}$*
- *$R^w_\sigma\langle[\![c]\!]^{\mathcal{A}}\rangle_{j\in w}$ for every term constant $c : \sigma$ in Σ.*

(This extends Jung and Tiuryn's definition to take term constants into account.)

KLRwVAs give rise to pre-logical relations in a similar way to I-relations, see Example 3.12: the family of relations consisting of the w-indexed tuples which are in a given KLRwVA at world w is a pre-logical relation which is not in general a logical relation, and those elements which are invariant under a given KLRwVA (i.e. $a \in [\![\sigma]\!]^{\mathcal{A}}$ such that $R^w_\sigma\langle a\rangle_{j\in w}$ for all w) also form a pre-logical predicate. More interesting is the fact that every pre-logical relation can be obtained in this way. We give the unary case; the binary and n-ary cases are obtained by instantiating to product structures.

Proposition 6.2. *Let $\mathcal{P} = \{P^\sigma \subseteq [\![\sigma]\!]^{\mathcal{A}}\}_{\sigma\in Types(B)}$ be a family of predicates over a Henkin structure $\mathcal{A}$. $\mathcal{P}$ is a pre-logical predicate iff it is the set of elements of $\mathcal{A}$ which are invariant under some KLRwVA.*

Proof. *$\Rightarrow$: Just as in the proof of Lemma 4 in [JT93], but using terms over Σ expanded by term constants for all values in $\mathcal{A}$. The proof assumes that $\mathcal{A}$ is a full type hierarchy but it extends to Henkin structures by a minor change of notation.*

$\Leftarrow$: From the definitions. □

The modification to Jung and Tiuryn's result hinted at in the proof, together with Theorem 3 in [JT93], can be used to generalize Jung and Tiuryn's lambda-definability result (their Theorem 5) to signatures containing term constants, cf. [Ali95].

7 Pre-logical Relations via Composition of Logical Relations

Our weakening of the definition of logical relations may appear to be *ad hoc*, but for extensional structures it turns out to be the minimal weakening that is closed under composition. There are variants of this result for several different classes of models. We give the version for Henkin models.

Theorem 7.1. *Let $\mathcal{A}$ and $\mathcal{B}$ be Henkin models and let $\mathcal{R}$ be a pre-logical relation over $\mathcal{A}$ and $\mathcal{B}$. Then $\mathcal{R}$ factors into a composition of three logical relations over Henkin models.*

Proof idea. *The key idea is to extend $\mathcal{A}$ and $\mathcal{B}$ to models $\mathcal{A}[Y]$ and $\mathcal{B}[Y]$ in such a way that $\mathcal{R}[Y]$, a minimally-extended version of $\mathcal{R}$, becomes a logical relation on $\mathcal{A}[Y]$ and $\mathcal{B}[Y]$. These models are obtained by adding indeterminates with generic behaviour to the carrier sets of $\mathcal{A}$ and $\mathcal{B}$, which ensures that the condition $\forall a \in [\![\sigma]\!]^{\mathcal{A}[Y]}. \forall b \in [\![\sigma]\!]^{\mathcal{B}[Y]}. R[Y]^\sigma(a, b) \Rightarrow R[Y]^\tau(App_{\mathcal{A}[Y]}\, f\, a, App_{\mathcal{B}[Y]}\, g\, b)$*

holds only for $f \in [\![\sigma \to \tau]\!]^{\mathcal{A}}$ *and* $g \in [\![\sigma \to \tau]\!]^{\mathcal{B}}$ *such that* $R^{\sigma\to\tau}(f,g)$. *Roughly speaking, we have that for any indeterminate* $y : \sigma$, $R[Y]^{\tau}(App_{\mathcal{A}[Y]}fy, App_{\mathcal{B}[Y]}gy)$ *iff* $R^{\sigma\to\tau}(f,g)$. *Then* $\mathcal{R}$ *is the composition of the embedding of* $\mathcal{A}$ *in* $\mathcal{A}[Y]$, $\mathcal{R}[Y]$, *and the inverse of the embedding of* $\mathcal{B}$ *in* $\mathcal{B}[Y]$.

A detailed proof may be found in the appendix. □

It is an open problem whether or not Theorem 7.1 holds if we take the composition of two logical relations rather than three.

Corollary 7.2. *The class of pre-logical relations on Henkin models is the closure under composition of the class of logical relations on such structures.* □

This gives the following lambda-definability result:

Corollary 7.3. *Let* $\mathcal{A}$ *be a Henkin model and* $a \in [\![\sigma]\!]^{\mathcal{A}}$. *Then* $\langle a, a\rangle$ *belongs to all relations over* $\mathcal{A}\times\mathcal{A}$ *obtained by composing logical relations iff* $a = [\![M : \sigma]\!]^{\mathcal{A}}$ *for some closed* Σ*-term* $M : \sigma$.

Proof. *By Corollary 7.2 and a binary version of Prop. 5.9.* □

It appears that the proof of Theorem 7.1 can be generalized to n-ary relations by repeating the same construction in each component and taking an appropriately generalized notion of composition. This yields the following structure theorem:

Theorem 7.4. *The class of pre-logical relations is obtained by closing the class of logical relations under intersection and then taking projections.* □

Corollary 7.2 does not hold if we restrict ourselves to considering just finite full type hierarchies: given an element a of a finite structure, it turns out to be co-r.e. whether the pair $\langle a, a\rangle$ belongs to all binary relations which are obtainable by closing logical relations under intersection and projection (and hence by closure under composition), while by Prop. 5.9 and [Loa9?] it is not co-r.e. whether $\langle a, a\rangle$ belongs to all binary pre-logical relations. In the case of arbitrary full type hierarchies, the question is open: the proof of Theorem 7.1 fails if we take a full type hierarchy in place of $\mathcal{A}[X]$, and we conjecture that Corollary 7.2 does not hold.

For non-extensional structures the notion of pre-logical relations is not the minimal weakening that gives closure under composition. The following variant is the minimal weakening for this case.

Definition 7.5. *An algebraic relation is* extensional *if whenever* $R^{\sigma\to\tau}(f,g)$, f *is extensionally equal to* f' *and* g *is extensionally equal to* g', *we have* $R^{\sigma\to\tau}(f',g')$.

All pre-logical relations over extensional structures are automatically extensional, and all logical relations over applicative structures (even non-extensional ones) are automatically extensional as well.

Proposition 7.6. *Let* $\mathcal{A}$ *and* $\mathcal{B}$ *be combinatory algebras and let* $\mathcal{R}$ *be an extensional pre-logical relation over* $\mathcal{A}$ *and* $\mathcal{B}$. *Then* $\mathcal{R}$ *factors into a composition of three logical relations.*

Proof. *As for Theorem 7.1 except that we need to take the extensional collapse over* $\mathcal{A}$ *and* $\mathcal{B}$ *respectively in the construction of* $\mathcal{A}[X]$ *and* $\mathcal{B}[X]$. *The fact that* $\mathcal{R}$ *is extensional is needed to show that the embeddings are logical relations.* □

Corollary 7.7. *The class of extensional pre-logical relations on combinatory algebras is the closure under composition of the class of logical relations on such structures.* □

These results may suggest that our definition of pre-logical relations on non-extensional structures should be strengthened by requiring the relation to be extensional, but this would make the reverse implication of the Basic Lemma fail. So although the notion of extensional pre-logical relations is the minimal weakening that gives closure under composition, these are stronger than necessary to give the Basic Lemma.

It would be interesting to continue the above investigations to characterize classes of relations generated by logical relations under operations such as intersection, projection or filtering, particularly on full type hierarchies.

8 Representation Independence and Data Refinement

Logical relations have been applied to explain the fact that the behaviour of programs does not depend on the way that data types are represented, but only on what can be observed about them using the operations that are provided. "Behaviour of programs" is captured by the notion of observational equivalence.

Definition 8.1. *Let $\mathcal{A}$ and $\mathcal{B}$ be lambda Σ-applicative structures and let OBS, the* observable types*, be a subset of Types(B). Then $\mathcal{A}$ is* observationally finer than *$\mathcal{B}$ with respect to OBS, written $\mathcal{A} \leq_{OBS} \mathcal{B}$, if for any two closed terms $M, N : \sigma$ for $\sigma \in OBS$ such that $[\![M : \sigma]\!]^{\mathcal{A}} = [\![N : \sigma]\!]^{\mathcal{A}}$ we have $[\![M : \sigma]\!]^{\mathcal{B}} = [\![N : \sigma]\!]^{\mathcal{B}}$.*

$\mathcal{A}$ and $\mathcal{B}$ are observationally equivalent *with respect to OBS, written $\mathcal{A} \equiv_{OBS} \mathcal{B}$, if $\mathcal{A} \leq_{OBS} \mathcal{B}$ and $\mathcal{B} \leq_{OBS} \mathcal{A}$.*

It is usual to take *OBS* to be the "built-in" types for which equality is decidable, for instance *bool* and/or *nat*. Then $\mathcal{A}$ and $\mathcal{B}$ are observationally equivalent iff it is not possible to distinguish between them by performing computational experiments.

Mitchell gives the following representation independence result:

Theorem 8.2 ([Mit96]). *Let Σ be a signature that includes a type constant nat, and let $\mathcal{A}$ and $\mathcal{B}$ be Henkin models, with $[\![nat]\!]^{\mathcal{A}} = [\![nat]\!]^{\mathcal{B}} = \mathbb{N}$. If there is a logical relation $\mathcal{R}$ over $\mathcal{A}$ and $\mathcal{B}$ with R^{nat} the identity relation on natural numbers, then $\mathcal{A} \equiv_{\{nat\}} \mathcal{B}$. Conversely, if $\mathcal{A} \equiv_{\{nat\}} \mathcal{B}$, Σ provides a closed term for each element of $\mathbb{N}$, and Σ only contains first-order functions, then there is a logical relation $\mathcal{R}$ over $\mathcal{A}$ and $\mathcal{B}$ with R^{nat} the identity relation.* □

The following example (Exercise 8.5.6 in [Mit96]) shows that the requirement that Σ contains only first-order functions is necessary.

Example 8.3. Let Σ have type constant *nat* and term constants $0, 1, 2, \ldots : nat$ and $f : (nat \to nat) \to nat$. Let $\mathcal{A}$ be the full type hierarchy over $[\![nat]\!]^{\mathcal{A}} = \mathbb{N}$ with $0, 1, 2, \ldots$ interpreted as usual and $[\![f]\!]^{\mathcal{A}}(g) = 0$ for all $g : \mathbb{N} \to \mathbb{N}$. Let $\mathcal{B}$ be like $\mathcal{A}$ but with $[\![f]\!]^{\mathcal{B}}(g) = 0$ if g is computable and $[\![f]\!]^{\mathcal{B}}(g) = 1$ otherwise. Since the difference between $\mathcal{A}$ and $\mathcal{B}$ cannot be detected by evaluating terms,

$\mathcal{A} \equiv_{\{nat\}} \mathcal{B}$. But there is no logical relation over $\mathcal{A}$ and $\mathcal{B}$ which is the identity relation on nat: if $\mathcal{R}$ is logical then $R^{nat\to nat}(g,g)$ for any $g : \mathbb{N} \to \mathbb{N}$, and then $R^{nat}(App_{\mathcal{A}}\, [\![f]\!]^{\mathcal{A}}\, g, App_{\mathcal{B}}\, [\![f]\!]^{\mathcal{B}}\, g)$, which gives a contradiction if g is non-computable. □

We will strengthen this result by showing that pre-logical relations characterize observational equivalence for *all* signatures. We also generalize to arbitrary sets of observable types but this is much less significant. This characterization is obtained as a corollary of the following theorem which is a strengthening of Lemma 8.2.17 in [Mit96], again made possible by using pre-logical relations in place of logical relations.

Theorem 8.4. *Let $\mathcal{A}$ and $\mathcal{B}$ be lambda Σ-applicative structures and let $OBS \subseteq Types(B)$. Then $\mathcal{A} \leq_{OBS} \mathcal{B}$ iff there exists a pre-logical relation over $\mathcal{A}$ and $\mathcal{B}$ which is a partial function on OBS.*

Proof. *$\Leftarrow$: Suppose that $\mathcal{R}$ is a pre-logical relation over $\mathcal{A}$ and $\mathcal{B}$ which is a partial function on OBS and let $[\![M:\sigma]\!]^{\mathcal{A}} = [\![N:\sigma]\!]^{\mathcal{A}}$ for $\sigma \in OBS$. Apply the Basic Lemma to both sides and use the fact that $\mathcal{R}^{\sigma}$ is a partial function to get $[\![M:\sigma]\!]^{\mathcal{B}} = [\![N:\sigma]\!]^{\mathcal{B}}$.*

$\Rightarrow$: Take the relation defined in Example 3.10. □

(Mitchell's Lemma 8.2.17 is the "if" direction for Henkin models where $OBS = Types(B)$ but $\mathcal{R}$ is required to be logical rather than just pre-logical.)

Corollary 8.5. *Let $\mathcal{A}$ and $\mathcal{B}$ be lambda Σ-applicative structures and let $OBS \subseteq Types(B)$. Then $\mathcal{A} \equiv_{OBS} \mathcal{B}$ iff there exists a pre-logical relation over $\mathcal{A}$ and $\mathcal{B}$ which is a partial function on OBS in both directions.* □

Corollary 8.6. *Let Σ be a signature that includes a type constant nat and let $\mathcal{A}$ and $\mathcal{B}$ be lambda Σ-applicative structures with $[\![nat]\!]^{\mathcal{A}} = [\![nat]\!]^{\mathcal{B}} = \mathbb{N}$ such that Σ provides a closed term for each element of $\mathbb{N}$. There is a pre-logical relation $\mathcal{R}$ over $\mathcal{A}$ and $\mathcal{B}$ with R^{nat} the identity relation on natural numbers iff $\mathcal{A} \equiv_{\{nat\}} \mathcal{B}$.* □

Example 8.7. Revisiting Example 8.3, the pre-logical relation constructed in Example 3.10 has the required property, and it does not relate non-computable functions since they are not lambda definable. □

Following [ST87], it would be interesting to try to characterize finer notions of observational equivalence, e.g. elementary equivalence in first-order logic with equality restricted to observable types. In Example 8.3, note that the sentence $\forall g,h : nat \to nat. f(g) = f(h)$ holds in $\mathcal{A}$ but not in $\mathcal{B}$.

In accounts of data refinement in terms of logical relations such as Sect. 2 of [Ten94], the fact that logical relations do not compose conflicts with the experience that data refinements *do* compose in real life. Example 5.4 can be embellished to give refinements between data structures like lists and sets for which the logical relations underlying the refinement steps do not compose to give a logical relation, yet the data refinements involved do compose at an intuitive level. This failure to justify the soundness of *stepwise* refinement is a serious flaw. If pre-logical relations are used in place of logical relations, then the fact that the composition of pre-logical relations is again a pre-logical relation (Prop. 5.6) explains why stepwise refinement is sound. This opens the way to further development of the foundations of data refinement along the lines of [ST88], but we leave this to a separate future paper, see Sect. 11.

9 Other Applications

There are many other applications of logical relations. Take for instance the proof of strong normalization of $\lambda^{\rightarrow}$ in [Mit96]: one defines an admissible logical predicate on a lambda applicative structure of terms by lifting the predicate on base types consisting of the strongly normalizing terms to higher types, proves that the predicate implies strong normalization, and then applies the general version of the Basic Lemma to give the result. The pattern for proofs of confluence, completeness of leftmost reduction, etc., is the same, sometimes with logical relations in place of logical predicates. There are also constructions that do not involve the Basic Lemma because the relations defined are not logical relations, but that include proofs following the same lines as the proof of the Basic Lemma. Examples include Gandy's proof that the hereditarily strict monotonic functionals model λ_I terms [Gan80], Plotkin's proof that lambda terms satisfy any I-relation [Plo80], and Jung and Tiuryn's proof that lambda terms satisfy any KLRwVA at each arity (Theorem 3 of [JT93]).

All of these can be cast into a common mould by using pre-logical relations rather than logical relations. If a relation or predicate on a lambda applicative structure is logical and admissible, then it is pre-logical, and then the Basic Lemma for pre-logical relations gives the result. Plotkin's, Jung and Tiuryn's, and Gandy's relations can be shown to be pre-logical (in Gandy's case with respect to λ_I), see Example 3.12, Sect. 6 and Example 3.13 respectively, and then the application of the Basic Lemma for pre-logical relations gives the result in these cases as well. In each case, however, the interesting part of the proof is not the application of the Basic Lemma (or the argument that replaces its application in the case of Gandy, Plotkin, and Jung and Tiuryn) but rather the construction of the relation and the proof of its properties. The point of the analysis is not to say that this view makes the job easier but rather to bring forward the common pattern in all of these proofs, which is suggestive of a possible methodology for such proofs.

Definition 9.1. *A family of binary relations* $\{R^\sigma \subseteq [\![\sigma]\!]^{\mathcal{A}} \times [\![\sigma]\!]^{\mathcal{A}}\}_{\sigma \in Types(B)}$ *over a Σ-applicative structure $\mathcal{A}$ is a* partial equivalence relation *(abbreviated* PER*) if it is symmetric and transitive for each type.*

Proposition 9.2. *Let $\mathcal{R}$ be a PER on a Σ-applicative structure $\mathcal{A}$ which is algebraic. Define the* quotient *of $\mathcal{A}$ by $\mathcal{R}$, written $\mathcal{A}/\mathcal{R}$, as follows:*

- $[\![\sigma]\!]^{\mathcal{A}/\mathcal{R}} = [\![\sigma]\!]^{\mathcal{A}}/R^\sigma$, *i.e. the set of $\mathcal{R}$-equivalence classes of objects $a \in [\![\sigma]\!]^{\mathcal{A}}$ such that $R^\sigma(a,a)$.*
- $App^{\sigma,\tau}_{\mathcal{A}/\mathcal{R}}\,[f]_{\mathcal{A}/\mathcal{R}}\,[a]_{\mathcal{A}/\mathcal{R}} = [App^{\sigma,\tau}_{\mathcal{A}}\,f a]_{\mathcal{A}/\mathcal{R}}$
- $[\![c]\!]^{\mathcal{A}/\mathcal{R}} = [c]_{\mathcal{A}/\mathcal{R}}$ *for each term constant $c : \sigma$ in Σ.*

Then:

1. *Let $\mathcal{A}$ be a lambda applicative structure. Then $\mathcal{A}/\mathcal{R}$ is a lambda applicative structure iff $\mathcal{R}$ is pre-logical.*
2. *Let $\mathcal{A}$ be a combinatory algebra. Then $\mathcal{A}/\mathcal{R}$ is a combinatory algebra iff $\mathcal{R}$ is pre-logical.*
3. *$\mathcal{A}/\mathcal{R}$ is an extensional applicative structure iff its restriction to the substructure of $\mathcal{A}$ consisting of the elements in $Dom(\mathcal{R})$ is a logical relation.* □

The last part of the above proposition says that one application of logical relations, that is their use in obtaining extensional structures by quotienting non-extensional structures — the so-called *extensional collapse* — requires a relation that is logical (on a substructure) rather than merely pre-logical.

The above proposition allows us to prove completeness for different classes of structures using the traditional technique of quotienting an applicative structure of terms by a suitable relation defined by provability in a calculus. For non-extensional structures, this is not possible using logical relations because the relation defined by provability is pre-logical or algebraic rather than logical.

At this point one could develop a theory analogous to that of homomorphisms, quotients and substructures in universal algebra, but we refrain from doing this here. One would expect analogues of the usual theorems relating these three notions.

10 Beyond $\boldsymbol{\lambda}^{\rightarrow}$ and Applicative Structures

Up to now we have been working in $\lambda^{\rightarrow}$, the simplest version of typed lambda calculus. We will now briefly indicate how other type constructors could be treated so as to obtain corresponding results for extended languages.

As a template, we shall discuss the case of product types. The syntax of types is extended by adding the type form $\sigma \times \tau$ and the syntax of terms is extended by adding pairing $\langle M, N\rangle$ and projections $\pi_1\ M$ and $\pi_2\ M$. If we regard these as additional term constants in the signature, e.g. $\langle\cdot,\cdot\rangle : \sigma \to \tau \to \sigma \times \tau$ for all σ, τ, rather than as new term forms, then the definition of pre-logical relations remains the same: the condition on term constants says that e.g. $R^{\sigma\to\tau\to\sigma\times\tau}(\llbracket\langle\cdot,\cdot\rangle\rrbracket^{\mathcal{A}}, \llbracket\langle\cdot,\cdot\rangle\rrbracket^{\mathcal{B}})$ and this is all that is required. For models that satisfy surjective pairing, this implies the corresponding condition on logical relations, namely

$$R^{\sigma\times\tau}(a,b) \quad \text{iff} \quad R^{\sigma}(\pi_1\, a, \pi_1\, b) \text{ and } R^{\tau}(\pi_2\, a, \pi_2\, b).$$

The treatment of sum types $\sigma + \tau$ is analogous, but notice that in this case this is also the only way to give the definition for logical relations since applying the usual pattern, namely

$$\begin{aligned}&R^{\sigma+\tau}(a,b) \quad \text{iff}\\ &\qquad \forall\rho.(\,\forall f \in \llbracket\sigma\to\rho\rrbracket^{\mathcal{A}}, f' \in \llbracket\sigma\to\rho\rrbracket^{\mathcal{B}}, g \in \llbracket\tau\to\rho\rrbracket^{\mathcal{A}}, g' \in \llbracket\tau\to\rho\rrbracket^{\mathcal{B}}.\\ &\qquad\qquad R^{\sigma\to\rho}(f,f') \wedge R^{\tau\to\rho}(g,g') \Rightarrow R^{\rho}(\mathit{case}\ a\ f\ g, \mathit{case}\ b\ f'\ g')\,)\end{aligned}$$

does not yield an inductive definition. This demonstrates that the pattern of definition for pre-logical relations is more robust than that for logical relations.

A type constructor that has received less attention in the literature is (finite) powerset, $\mathcal{P}(\sigma)$. The treatment would be like that of products and coproducts, given models containing a standard interpretation of *bool*: we add term constants $\emptyset : \mathcal{P}(\sigma)$, $\{\cdot\} : \sigma \to \mathcal{P}(\sigma)$, $\cup : \mathcal{P}(\sigma) \to \mathcal{P}(\sigma) \to \mathcal{P}(\sigma)$ and $\in : \sigma \to \mathcal{P}(\sigma) \to \mathit{bool}$ and require $R^{P(\sigma)}([\![\emptyset]\!]^{\mathcal{A}}, [\![\emptyset]\!]^{\mathcal{B}})$ etc. For models in which the interpretation of the powerset type and these new constants are as usual, this amounts to imposing the following condition:

$$R^{\mathcal{P}(\sigma)}(\alpha, \beta) \quad \text{iff} \quad \forall a \in \alpha. \exists b \in \beta. R^{\sigma}(a, b) \text{ and } \forall b \in \beta. \exists a \in \alpha. R^{\sigma}(a, b).$$

Note that this is the same pattern used in defining bisimulations.

Various other kinds of types can be considered, including inductive and coinductive data types, universally and existentially quantified types, and various flavours of dependent types. We have not yet considered these in any detail, but we are confident that for any of them, one could take any existing treatment of logical relations and modify it by weakening the condition on functions as above without sacrificing the Basic Lemma. We expect that this would even yield improved results as it has above, but this is just speculation.

A different dimension of generalization is to consider models having additional structure — e.g. Kripke applicative structures [MM91], pre-sheaf models or cartesian closed categories — for which logical relations have been studied. We have not yet examined the details of this generalization but it appears that a corresponding weakening of the definition would lead to analogues of the results above, cf. [PPS98]. It is worth noting that Prop. 6.2 links pre-logical relations to KLRwVAs, which have a formulation in terms of pre-sheaf categories and have been extended to cartesian closed categories in [Ali95].

11 Conclusions and Directions for Future Work

Our feeling is that by introducing the notion of pre-logical relation we have, metaphorically and a little immodestly, removed a "blind spot" in the existing intuition of the use and scope of logical relations and related techniques. This is not to say that some specialists in the field have not previously contemplated generalizations similar to ours, but they have not carried the investigation far enough. We believe that in this paper we have exposed very clearly the fact that in many situations the use of logical relations is unnecessarily restrictive. Using pre-logical relations instead, we get improved statements of some results (e.g. Theorem 8.4 and its corollaries), we encompass constructions that had previously escaped the logical paradigm (e.g. Example 3.13), and we isolate the necessary and sufficient hypotheses for many arguments to go through (e.g. Lemma 4.1). We have given several characterizations of pre-logical relations, summarized in the following theorem (for the unary case):

Theorem 11.1. *Let $\mathcal{P} = \{P^{\sigma} \subseteq [\![\sigma]\!]^{\mathcal{A}}\}_{\sigma \in \mathit{Types}(B)}$ be a family of predicates over a Henkin structure $\mathcal{A}$. The following are equivalent.*

1. *$\mathcal{P}$ is a pre-logical predicate.*
2. *$\mathcal{P}$ is closed under lambda definability.*
3. *$\mathcal{P}$ is the set of elements of $\mathcal{A}$ which are invariant under some KLRwVA.*

4. $\mathcal{P}$ is the set of elements of $\mathcal{A}$ that are invariant under (or alternatively, the projection or $\forall$ of) the composition of (three) logical relations.

Proof. *1⇔2 is Prop. 4.2, 1⇔3 is Prop. 6.2, and 1⇔4 is by Corollary 7.2.* □

The fact that there are so many conceptually independent ways of defining the same class of relations suggests that it is a truly intrinsic notion. Notice that Thm. 11.1(3) gives an inductive flavour to this concept which is not explicit in the definition of pre-logical relation; this apparent lack has been regarded as a weakness of the concept, see e.g. p. 428-429 of [Mit90].

Throughout the paper we have indicated possible directions of future investigation, e.g. with respect to richer type theories. It is plausible that sharper characterizations of representation independence will appear in many different type contexts.

But probably the area where the most benefits will be achieved will be that of the foundations of data refinement. Here we think that a more comprehensive explanation of data refinement would be obtained by combining an account in terms of pre-logical relations with the first-order algebraic treatment in [ST88] which we would expect to extend smoothly to higher-order. Among other improvements, this would result in a non-symmetric refinement relation, giving a better fit with the real-life phenomenon being modelled.

There is a vast literature on logical relations in connection with areas like parametricity, abstract interpretation, etc. A treatment of these topics in terms of pre-logical relations is likely to be as fruitful and illuminating as it has proved to be for the classical example of simply-typed lambda calculus presented here.

Acknowledgements: Thanks to Samson Abramsky, Jo Hannay, Martin Hofmann, Andrew Kennedy, Yoshiki Kinoshita, John Mitchell, Peter O'Hearn, Gordon Plotkin, John Power and Ian Stark for helpful comments. This work has been partially supported by EPSRC grant GR/K63795, an SOEID/RSE Support Research Fellowship, the ESPRIT-funded CoFI and TYPES working groups, and a MURST'97 grant.

References

[Abr90] S. Abramsky. Abstract interpretation, logical relations, and Kan extensions. *Journal of Logic and Computation* 1:5–40 (1990).

[Ali95] M. Alimohamed. A characterization of lambda definability in categorical models of implicit polymorphism. *Theoretical Computer Science* 146:5–23 (1995).

[Gan80] R. Gandy. Proofs of strong normalization. In: *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, 457–477. Academic Press (1980).

[JT93] A. Jung and J. Tiuryn. A new characterization of lambda definability. *Proc. TLCA'93*. Springer LNCS 664, 245–257 (1993).

[KOPTT97] Y. Kinoshita, P. O'Hearn, J. Power, M. Takeyama and R. Tennent. An axiomatic approach to binary logical relations with applications to data refinement. *Proc. TACS'97*, Springer LNCS 1281, 191–212 (1997).

[Loa9?] R. Loader. The undecidability of λ-definability. Church Memorial volume, to appear (199?).

[Mit90] J. Mitchell. Type Systems for Programming Languages. Chapter 8 of *Handbook of Theoretical Computer Science, Vol B.* Elsevier (1990).

[Mit96] J. Mitchell. *Foundations for Programming Languages.* MIT Press (1996).

[MM91] J. Mitchell and E. Moggi. Kripke-style models for typed lambda calculus. *Annals of Pure And Applied Logic* 51:99–124 (1991).

[Plo80] G. Plotkin. Lambda-definability in the full type hierarchy. In: *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, 363–373. Academic Press (1980).

[PPS98] G. Plotkin, J. Power and D. Sannella. A compositional generalisation of logical relations. Draft report, `http://www.dcs.ed.ac.uk/home/dts/pub/laxlogrel.ps` (1998).

[Rob96] E. Robinson. Logical relations and data abstraction. Report 730, Queen Mary and Westfield College (1996).

[ST87] D. Sannella and A. Tarlecki. On observational equivalence and algebraic specifications. *Journal of Computer and System Sciences* 34:150–178 (1987).

[ST88] D. Sannella and A. Tarlecki. Toward formal development of programs from algebraic specifications: implementations revisited. *Acta Informatica* 25:233–281 (1988).

[Sch87] O. Schoett. Data Abstraction and the Correctness of Modular Programming. Ph.D. thesis CST-42-87, Univ. of Edinburgh (1987).

[Sta85] R. Statman. Logical relations and the typed lambda calculus. *Information and Control* 65:85–97 (1985).

[Ten94] R. Tennent. Correctness of data representations in Algol-like languages. In: *A Classical Mind: Essays in Honour of C.A.R. Hoare.* Prentice Hall (1994).

A Proof of Theorem 7.1

Theorem 7.1. *Let $\mathcal{A}$ and $\mathcal{B}$ be Henkin models and let $\mathcal{R}$ be a pre-logical relation over $\mathcal{A}$ and $\mathcal{B}$. Then $\mathcal{R}$ factors into a composition of three logical relations over Henkin models.*

Proof. *See Sect. 7 for the idea of the proof. We proceed as follows:*

- *We define the model $\mathcal{A}[Y]$ and the relation that embeds $\mathcal{A}$ in $\mathcal{A}[Y]$, and show that this is a logical relation (Lemma A.7). The same applies to $\mathcal{B}$ and $\mathcal{B}[Y]$.*
- *We define $\mathcal{R}[Y] \subseteq \mathcal{A}[Y] \times \mathcal{B}[Y]$ and show that it is a logical relation (Lemma A.9).*
- *Finally we show that $\mathcal{R}$ is the composition of the embedding of $\mathcal{A}$ in $\mathcal{A}[Y]$, $\mathcal{R}[Y]$, and the inverse of the embedding of $\mathcal{B}$ in $\mathcal{B}[Y]$ (Lemma A.10).*

Let Y be a set of typed variables containing an infinite number of variables for each type in $Types(B)$.

Definition A.1. *Let $\Sigma\mathcal{A}$ be Σ augmented by a term constant $c_a : \sigma$ for each element $a \in [\![\sigma]\!]^{\mathcal{A}}$. Let $\Sigma[Y]$ (resp. $\Sigma\mathcal{A}[Y]$) be Σ (resp. $\Sigma\mathcal{A}$) augmented by a term constant $c_y : \sigma$, called an* indeterminate*, for each variable $y : \sigma$ in Y.*

Definition A.2. *The Henkin model $\mathcal{A}[Y]$ over $\Sigma[Y]$ is the (closed) $\Sigma\mathcal{A}[Y]$-term model of the simply-typed lambda calculus with β and η, and with δ-reductions describing the behaviour of each constant c_a with respect to other such constants (and with no δ-reductions involving the indeterminates). This calculus is Church-Rosser and strongly normalizing so each term M has a unique normal form $\widehat{M}$. Although elements in $\mathcal{A}[Y]$ are equivalence classes of terms, it is convenient to refer to them using single terms, by which is meant the equivalence class containing that term. Since all terms in a given equivalence class have the same normal form, we will sometimes write $\widehat{a}$ for an element a in $\mathcal{A}[Y]$. If η is a Γ-environment over $\mathcal{A}$ then η_c is the Γ-environment over $\mathcal{A}[Y]$ which assigns each variable x to (the equivalence class of) $c_{\eta(x)}$.*

Lemma A.3. *If $M \to^*_{\beta\eta\delta} N$ for $\Gamma \rhd_{\Sigma\mathcal{A}} M, N : \sigma$, then for any Γ-environment η over $\mathcal{A}$, $[\![M]\!]^{\mathcal{A}}_\eta = [\![N]\!]^{\mathcal{A}}_\eta$ in which we interpret each of the constants c_a as a.*

Proof. By induction on the structure of the derivation of $M \to^*_{\beta\eta\delta} N$. □

Lemma A.4. *Let $\Gamma \rhd_\Sigma M, N : \sigma$. If $[\![N]\!]^{\mathcal{A}[Y]}_{\eta_c} = [\![M]\!]^{\mathcal{A}[Y]}_{\eta'_c}$ then $[\![N]\!]^{\mathcal{A}}_\eta = [\![M]\!]^{\mathcal{A}}_{\eta'}$.*

Proof. If $[\![N]\!]^{\mathcal{A}[Y]}_{\eta_c} = [\![M]\!]^{\mathcal{A}[Y]}_{\eta'_c}$ then $N[\eta_c(x)/x$ for all $x]$ and $M[\eta_c(x)/x$ for all $x]$ reduce to the same normal form. By Lemma A.3, the reduction sequences that yield this normal form can be reproduced in $\mathcal{A}$ to give $[\![N]\!]^{\mathcal{A}}_\eta = [\![M]\!]^{\mathcal{A}}_{\eta'}$. □

Lemma A.5. *Let $\Gamma \rhd_\Sigma M : \sigma$ and $\Gamma \rhd_{\Sigma[Y]} N : \sigma$. If $[\![M]\!]^{\mathcal{A}[Y]}_{\eta_c} = [\![N]\!]^{\mathcal{A}[Y]}_{\eta_c}$ then $\widehat{[\![N]\!]^{\mathcal{A}[Y]}_{\eta_c}}$ is a $\Sigma\mathcal{A}$-term.*

Proof. Adding indeterminates adds no extra reductions. □

Definition A.6. *The embedding emb of $\mathcal{A}$ into $\mathcal{A}[Y]$ is defined as follows:*

$$emb^\sigma = \{\langle [\![M]\!]^{\mathcal{A}}_\eta, [\![M]\!]^{\mathcal{A}[Y]}_{\eta_c}\rangle \mid \Gamma \rhd_\Sigma M : \sigma \text{ and } \eta \text{ a } \Gamma\text{-environment over } \mathcal{A}\}$$

Although we call emb an embedding, this is inaccurate since it is never a set-theoretic function. To be precise, it is the natural lifting to a logical relation of the embedding at base types.

Lemma A.7. *emb is a logical relation.*

Proof. It is easy to see that each constant c in Σ is in relation to itself: just take $M = c$. Furthermore, suppose $emb^{\sigma\to\tau}([\![M]\!]^{\mathcal{A}}_\eta, [\![M]\!]^{\mathcal{A}[Y]}_{\eta_c})$ and $emb^\sigma([\![N]\!]^{\mathcal{A}}_{\eta'}, [\![N]\!]^{\mathcal{A}[Y]}_{\eta'_c})$; we need to show $emb^\tau(App_{\mathcal{A}}\, [\![M]\!]^{\mathcal{A}}_\eta\, [\![N]\!]^{\mathcal{A}}_{\eta'}, App_{\mathcal{A}[Y]}\, [\![M]\!]^{\mathcal{A}[Y]}_{\eta_c}\, [\![N]\!]^{\mathcal{A}[Y]}_{\eta'_c})$. Without loss of generality we can assume that $\eta = \eta'$ and then the result follows from the definition of interpretation: $App_{\mathcal{A}}\, [\![M]\!]^{\mathcal{A}}_\eta\, [\![N]\!]^{\mathcal{A}}_\eta = [\![M\, N]\!]^{\mathcal{A}}_\eta$ and similarly for $\mathcal{A}[Y]$.

It remains to prove the opposite direction of the implication. Suppose that whenever $emb^\sigma(a, d)$, $emb^\tau(App_{\mathcal{A}} f a, App_{\mathcal{A}[Y]} g d)$. We need to show $emb^{\sigma\to\tau}(f, g)$. Consider the normal form $\widehat{g}$ of g. We claim that $\widehat{g}$ is a $\Sigma\mathcal{A}$-term, i.e. it contains no indeterminates. We know that $emb^\sigma(a, [\![x]\!]^{\mathcal{A}[Y]}_{\eta_c})$ with $\eta_c(x) = c_a$. Since $emb^\tau(App_{\mathcal{A}}\ f\ a, App_{\mathcal{A}[Y]}\ g\ [\![x]\!]^{\mathcal{A}[Y]}_{\eta_c})$, there is a term $\Gamma \rhd_\Sigma N : \tau$ and Γ-environment ξ such that $App_{\mathcal{A}[Y]}\ g\ [\![x]\!]^{\mathcal{A}[Y]}_{\eta_c} = [\![N]\!]^{\mathcal{A}[Y]}_{\xi_c}$. W.l.o.g. $\xi = \eta$, so $\widehat{App_{\mathcal{A}[Y]}\ g\ [\![x]\!]^{\mathcal{A}[Y]}_{\eta_c}}$ is a $\Sigma\mathcal{A}$-term by Lemma A.5, and then so is $\widehat{g}$. Therefore, consider the Σ-term M and environment η'_c such that $[\![M]\!]^{\mathcal{A}[Y]}_{\eta'_c} = \widehat{g}$: M is $\widehat{g}$ with each constant c_a replaced by a variable which η'_c maps to c_a. But now we claim that f is extensionally equal to $[\![M]\!]^{\mathcal{A}}_{\eta'}$. We have $emb^\tau(App_{\mathcal{A}} f a, App_{\mathcal{A}[Y]} [\![M]\!]^{\mathcal{A}[Y]}_{\eta'_c} c_a)$ so $emb^\tau(App_{\mathcal{A}}\ f\ a, [\![M\ z]\!]^{\mathcal{A}[Y]}_{\eta'_c[z\mapsto c_a]})$ for a fresh variable z. By the definition of emb, there is a term $\Gamma \rhd_\Sigma M' : \tau$ and Γ-environment η'' such that $[\![M']\!]^{\mathcal{A}}_{\eta''} = App_{\mathcal{A}}\ f\ a$ and $[\![M']\!]^{\mathcal{A}[Y]}_{\eta''_c} = [\![M\ z]\!]^{\mathcal{A}[Y]}_{\eta'_c[z\mapsto c_a]}$. W.l.o.g., $\eta''_c = \eta'_c[z \mapsto c_a]$. Then $[\![M']\!]^{\mathcal{A}}_{\eta''} = [\![M\ z]\!]^{\mathcal{A}}_{\eta''}$ by Lemma A.4, and thus $App_{\mathcal{A}}\ f\ a = App_{\mathcal{A}}\ [\![M]\!]^{\mathcal{A}}_{\eta'}\ a$ which shows that f is extensionally equal to $[\![M]\!]^{\mathcal{A}}_{\eta'}$. So by extensionality, $[\![M]\!]^{\mathcal{A}}_{\eta'} = f$ which gives $emb^{\sigma\to\tau}(f, g)$. □

We will need to refer to both $emb \subseteq \mathcal{A} \times \mathcal{A}[Y]$ and $emb \subseteq \mathcal{B} \times \mathcal{B}[Y]$ so we use the notation $emb_{\mathcal{A}}$ for the former and $emb_{\mathcal{B}}$ for the latter.

Definition A.8. *The relation $\mathcal{R}[Y] \subseteq \mathcal{A}[Y] \times \mathcal{B}[Y]$ is defined as follows:*

$$\mathcal{R}[Y]^\sigma = \{\langle [\![M]\!]^{\mathcal{A}[Y]}_{\eta_c}, [\![M]\!]^{\mathcal{B}[Y]}_{\eta'_c}\rangle \mid \Gamma \rhd_{\Sigma[Y]} M : \sigma \text{ and } \eta, \eta' \text{ are } \Gamma\text{-environments for } \mathcal{A}, \mathcal{B} \text{ such that } R(\eta, \eta')\}$$

Lemma A.9. *$\mathcal{R}[Y]$ is a logical relation.*

Proof. *Each constant c in Σ is in relation to itself: just take $M = c$. Furthermore, suppose $\mathcal{R}[Y]([\![M]\!]^{\mathcal{A}[Y]}_{\eta_c}, [\![M]\!]^{\mathcal{B}[Y]}_{\eta'_c})$ and $\mathcal{R}[Y]([\![N]\!]^{\mathcal{A}[Y]}_{\eta_c}, [\![N]\!]^{\mathcal{B}[Y]}_{\eta'_c})$. (We should assume that we have different environments for $[\![M]\!]^{\mathcal{A}[Y]}$ and $[\![N]\!]^{\mathcal{A}[Y]}$ and for $[\![M]\!]^{\mathcal{B}[Y]}$ and $[\![N]\!]^{\mathcal{B}[Y]}$, but without loss of generality we can assume that the $\mathcal{A}[Y]$-environments are the same and similarly for $\mathcal{B}[Y]$.) We need to show $\mathcal{R}[Y](App_{\mathcal{A}[Y]}\, [\![M]\!]^{\mathcal{A}[Y]}_{\eta_c}\, [\![N]\!]^{\mathcal{A}[Y]}_{\eta_c}, App_{\mathcal{B}[Y]}\, [\![M]\!]^{\mathcal{B}[Y]}_{\eta'_c}\, [\![N]\!]^{\mathcal{B}[Y]}_{\eta'_c})$. We get the result by the definition of interpretation: $App_{\mathcal{A}[Y]}\, [\![M]\!]^{\mathcal{A}[Y]}_{\eta_c}\, [\![N]\!]^{\mathcal{A}[Y]}_{\eta_c} = [\![MN]\!]^{\mathcal{A}[Y]}_{\eta_c}$ and similarly for $\mathcal{B}[Y]$.*

It remains to prove the opposite direction of the implication. Suppose we have $f \in \mathcal{A}[Y]$ and $g \in \mathcal{B}[Y]$ such that for every $\Sigma[Y]$-term P and environments η, η' for $\mathcal{A}$ and $\mathcal{B}$ such that $R(\eta, \eta')$, $\mathcal{R}[Y](App_{\mathcal{A}[Y]} f\, [\![P]\!]^{\mathcal{A}[Y]}_{\eta_c}, App_{\mathcal{B}[Y]} g\, [\![P]\!]^{\mathcal{B}[Y]}_{\eta'_c})$. Then take P to be an indeterminate c_y which does not occur in $\widehat{f}$ or $\widehat{g}$. (We need an infinite number of indeterminates to ensure that there is one that is different from all those that are already used in the terms $\widehat{f}$ and $\widehat{g}$.) We know that there exists a $\Sigma[Y]$-term M and environments ξ, ξ' for $\mathcal{A}$ and $\mathcal{B}$ such that $R(\xi, \xi')$, $App_{\mathcal{A}[Y]}\, f\, [\![c_y]\!]^{\mathcal{A}[Y]}_{\xi_c} = [\![M]\!]^{\mathcal{A}[Y]}_{\xi_c}$ and $App_{\mathcal{B}[Y]}\, g\, [\![c_y]\!]^{\mathcal{B}[Y]}_{\xi'_c} = [\![M]\!]^{\mathcal{B}[Y]}_{\xi'_c}$. Consider the $\Sigma[Y]$-term $Q = \lambda c_y.M$. (Formally speaking, we should really take $\lambda x.M[x/c_y]$ so that we are abstracting a variable rather than a constant, but the meaning is clear.) We claim that f is extensionally equal to $[\![Q]\!]^{\mathcal{A}[Y]}_{\xi_c}$. Let $a \in \mathcal{A}[Y]$. Now consider the reduction sequences from $App_{\mathcal{A}[Y]}\, f\, [\![c_y]\!]^{\mathcal{A}[Y]}_{\xi_c}$ and $App_{\mathcal{A}[Y]}\, [\![\lambda c_y.M]\!]^{\mathcal{A}[Y]}_{\xi_c}\, [\![c_y]\!]^{\mathcal{A}[Y]}_{\xi_c}$ to a common reduct S. If we replace all unbound occurrences of c_y in these reduction sequences with a, we obtain valid reduction sequences from $App_{\mathcal{A}[Y]}\, f\, a$ and $App_{\mathcal{A}[Y]}\, [\![\lambda c_y.M]\!]^{\mathcal{A}[Y]}_{\xi_c}\, a$ to the common reduct $S[a/c_y]$. This works because c_y is "sterile", with generic behaviour. Similary, g is extensionally equal to $[\![Q]\!]^{\mathcal{B}[Y]}_{\xi'_c}$. Therefore, since $\mathcal{A}[Y]$ and $\mathcal{B}[Y]$ are extensional, $\mathcal{R}[Y](f, g)$. □

Lemma A.10. *The composition of $emb_{\mathcal{A}}$, $\mathcal{R}[Y]$ and $(emb_{\mathcal{B}})^{-1}$ is R.*

Proof. *It is easy to see that R is included in the composition. As for the converse, suppose $((emb_{\mathcal{B}})^{-1} \circ \mathcal{R}[Y] \circ emb_{\mathcal{A}})(a, b)$. By definition of $emb_{\mathcal{A}}$ there exists a Σ-term N and environment $\eta\mathcal{A}$ such that $a = [\![N]\!]^{\mathcal{A}}_{\eta\mathcal{A}}$. By definition of $emb_{\mathcal{B}}$ there exists a Σ-term N' and environment $\eta\mathcal{B}$ such that $b = [\![N']\!]^{\mathcal{B}}_{\eta\mathcal{B}}$. By definition of $\mathcal{R}[Y]$ there exists a $\Sigma[Y]$-term M and environments η, η' such that $[\![N]\!]^{\mathcal{A}[Y]}_{\eta\mathcal{A}_c} = [\![M]\!]^{\mathcal{A}[Y]}_{\eta_c}$, $[\![N']\!]^{\mathcal{B}[Y]}_{\eta\mathcal{B}_c} = [\![M]\!]^{\mathcal{B}[Y]}_{\eta'_c}$ and $R(\eta, \eta')$. W.l.o.g. we can assume that $\eta\mathcal{A} = \eta$ and $\eta\mathcal{B} = \eta'$. By Lemma A.5 we know that $\widehat{[\![M]\!]^{\mathcal{A}[Y]}_{\eta_c}}$ and*

$\widehat{[\![M]\!]^{\mathcal{B}[Y]}_{\eta'_c}}$ *are* $\Sigma\mathcal{A}$*-terms. By Lemma A.4 we know that* $a = [\![N]\!]^{\mathcal{A}}_{\eta} = [\![M]\!]^{\mathcal{A}}_{\eta}$ *and* $b = [\![N']\!]^{\mathcal{B}}_{\eta} = [\![M]\!]^{\mathcal{B}}_{\eta'}$*. So by the Basic Lemma,* $R(a, b)$*.* □

□ *(Theorem 7.1)*