Deriving Logical Relations from Interpretations of Predicate Logic

Claudio Hermida[1] Uday S. Reddy[2]

University of Birmingham

Edmund P. Robinson[3,4]

Queen Mary, University of London

Abstract

This paper extends the results of Hermida's thesis about logical predicates to more general logical relations and a wider collection of types. The extension of type constructors from types to logical relations is derived from an interpretation of those constructors on a model of predicate logic. This is then further extended to n-ary relations by pullback. Hermida's theory shows how right adjoints in the category of fibrations are composed from a combination of Cartesian lifting and a local adjunction. This result is generalised to make it more applicable to left adjoints, and then shown to be stable under pullback, deriving an account of n-ary relations from standard predicate logic. A brief discussion of lifting monads to predicates includes the existence of an initial such lifting, generalising existing results.

Keywords: logical relations, fibrations, categorical type theory, monadic types

1 Introduction

The purpose of this paper is to illustrate a link between logical relations and predicate logics. We begin by revisiting the conditions under which the predicates in a predicate logic for reasoning about a type theory can be shown to carry the structure of that type theory. We then show that similar structure on binary and more generally n-ary relations can be derived from that on unary predicates. Finally, we take a look at structure defined via monads. The core of this material is already known. We extend the results to bring out more clearly what is needed, handle n-ary predicates via change of base, and discuss monadic types.

In order to do this we make use of the tools of category theory. From this perspective, predicate logic is formalised as a fibration over a category representing the type theory. More accurately the base can be taken to have objects representing contexts consisting of variable:type bindings, and morphisms given by substitutions. We concentrate first on type theoretic structure that is defined by adjunction (products, function spaces and sums), but we then discuss monadic structure.

Logical relations were introduced by Gordon Plotkin [13,15]. But he is generous in giving credit to others, notably Hans Laüchli, Mike Gordon and Robert Milne. Milne had come up with a similar idea in connection with proving the equivalence of different semantics for programming languages (under the name of inclusive predicates) [12]. It is Milne's insight, that these structures can be used to establish a formal link between different semantics, that has led to most of the modern work using them.

[1] Email: claudio.hermida@gmail.com

[2] Email: u.s.reddy@bham.ac.uk

[3] Email: e.p.robinson@qmul.ac.uk

[4] This work was partially supported by EPSRC grant EP/R006865/1, Interface Reasoning for Interacting Systems

Since this time, the basic idea has been interpreted in a variety of settings. Calculi have been introduced for working with them, [14] and others, and the idea has been explained in terms of structure-preserving functor from categories of logical relations to the underlying types.

In this paper we build on that, starting out with Hermida's work on logical predicates and how the structure of adjunctions between fibrations leads to an account of the interpretation of type theory in logical predicates.

We mine Hermida's work slightly to show that we do not always need the full structure of fibrations. The key point where this has impact in the case of coproducts. Hermida's account along with successors, requires all cocartesian liftings. This is a strong condition that does not hold in all the syntactic fibrations corresponding to logics.

We then prove a change of base theorem for Hermida's account, and use it to derive the formation of the same type constructions in n-ary relations. The example of function spaces shows that this is not as straightforward as first appears.

We conclude the paper with a discussion of monadic types. These behave differently from the conventional type-theoretic structures. We emphasise this with examples: monads on Set preserves monics, and hence extend to functors on Pred, but not all are fibred (the continuation monad); even though we immediately get an extension to Pred we need more work to get one for Rel (continuations again and also some algebraic theories like Mal'cev algebras). We also present a sufficient condition to have an initial lifting of a monad, relating it to recent work of Kammar and McDermott [8].

This work forms part of a programme, starting with [6], in which we attempt to relate the theory of logical relations to other forms of programme logic and standard forms of mathematical reasoning. The work in this paper supports deriving logical relations from predicates in the standard way that we expect. An issue with this approach is that it requires relations to have a single logical metastructure, rather than to give a systematic link between two different metastructures. This can be mitigated to some extent because the structures considered are closed under products, so can always be combined into a single entity. The work also contains the start of a treatment of monadic types. We expect to put more detail on that in future work.

2 Predicates and relations

To illustrate the concepts we use the categories of unary and binary relations: Pred and Rel.

Definition 2.1 [Pred] The objects of the category Pred are pairs (P, A) where A is a set and P is a subset of A. A morphism $(P, A) \to (Q, B)$ is a function $f : A \to B$ such that $\forall a \in A. a \in P \Rightarrow f(a) \in Q$. Identities and composition are inherited from Set.

Pred also has a logical reading. We can take (P, A) as a predicate on the type A, and associate it with a judgement of the form $a : A \vdash P(a)$ (read "in the context $a : A$, $P(a)$ is a proposition"). A morphism $(a : A \vdash P(a)) \to (b : B \vdash Q(b))$ has two parts: a substitution $b \mapsto t(a)$, and the logical consequence $P(a) \Rightarrow Q(t(a))$ (read "whenever $P(a)$ holds, then so does $Q(t(a))$).

Definition 2.2 [Rel and Rel$_n$] The objects of the category Rel are triples (R, A_1, A_2) where A_1 and A_2 are sets and R is a subset of $A_1 \times A_2$ (a relation between A_1 and A_2). A morphism $(R, A_1, A_2) \to (S, B_1, B_2)$ is a pair of functions $f_1 : A_1 \to B_1$ and $f_2 : A_2 \to B_2$ such that $\forall a_1 \in A_1, a_2 \in A_2. (a_1, a_2) \in R \Rightarrow (f_1(a_1), f_2(a_2)) \in S$. Identities and composition are inherited from Set $\times$ Set.

$$\begin{tikzcd}
P \arrow[r, dashed] \arrow[d] & Q \arrow[d] \\
A \arrow[r, "f"] & B
\end{tikzcd}
\qquad
\begin{tikzcd}
R \arrow[r, dashed] \arrow[d] & S \arrow[d] \\
A_1 \times A_2 \arrow[r, "f_1 \times f_2"] & B_1 \times B_2
\end{tikzcd}$$

Rel$_n$ is the obvious generalisation of Rel to n-ary relations.

Pred has a forgetful functor $p : \mathsf{Pred} \to \mathsf{Set}$, $p(P, A) = A$, and similarly Rel has a forgetful functor $q : \mathsf{Rel} \to \mathsf{Set} \times \mathsf{Set}$, $q(R, A_1, A_2) = (A_1, A_2)$. These functors carry a good deal of structure and are critical to a deeper understanding of the constructions.

Moreover, both Pred and Rel are cartesian closed categories.

2.1 Cartesian closed structure

The terminal object in Pred is $(1, 1)$, a singleton set with itself as designated subset. In Rel the terminal object is $(1 \times 1, 1, 1)$, the total relation on two singletons.

Products are not much more complex. In Pred the product of (P, A) and (Q, B) is $(P \times Q, A \times B)$. And in Rel the product of (R, A_1, A_2) and (S, B_1, B_2) is $(R \times S, A_1 \times B_1, A_2 \times B_2)$, where $((a_1, b_1), (a_2, b_2)) \in R \times S$ if and only if $(a_1, a_2) \in R$ and $(b_1, b_2) \in S$.

The last part of the cartesian closed structure is the function space construction. In Pred, the function space $[(P, A) \to (Q, B)]$ is $[R, A \to B]$ where R is the subset of functions $A \to B$ mapping P into Q. In Rel The function space $[(R, A_1, A_2) \to (S, B_1, B_2)]$ is $(R \to S, A_1 \to B_1, A_2 \to B_2)$, where $f_1[R \to S]f_2$ if and only if $\forall a_1 \in A_1, a_2 \in A_2.\ a_1 R A_2 \Rightarrow f_1(a_1) S f_2(a_2)$.

So, both Pred and Rel are not only cartesian closed categories, but the forgetful functors p and q preserve all of that structure.

2.2 Sums

These properties extend to sums (coproducts). In Pred $(P, A) + (Q, B) = (P + Q, A + B)$ where $c \in P + Q$ if and only if $c \in P$ or $c \in Q$.

Sums in Rel are constructed similarly. $(R, A_1, A_2) + (S, B_1, B_2) = (R + S, A_1 + B_1, A_2 + B_2)$, where $x[R + S]y$ if and only if $(x = \mathrm{inl}\, a \wedge y = \mathrm{inl}(b) \wedge aRb) \vee (x = \mathrm{inr}\, a \wedge y = \mathrm{inr}(b) \wedge aSb)$.

3 Fibrations

We start with an interpretation of predicate logic such as $p : \mathsf{Pred} \to \mathsf{Set}$. This has structure beyond being a homomorphism of cartesian closed categories. It is a fibration equipped with structure that supports an interpretation of predicate logic. See Jacobs [7] for a comprehensive introduction to fibrations and the interpretation of logic in them.

Given a functor $p : E \to C$, and an object A of C, then $p^{-1}(A)$ forms a subcategory in which the morphisms are those mapping onto the identity of A. This is called the *fibre* of p over A, written E_A. In the context of fibrations, the morphisms of E_A are called *vertical* maps. In the context of logic, the object A represents a context in the sense of a structure giving variable:type bindings, and the fibre over it represents propositions definable in that context ordered by consequence.

For any functor $p : E \to C$, each object of E is an object of a unique fibre. Hence the objects of E are the disjoint union of the objects of the fibres, $\mathrm{ob}\, E = \coprod_{A \in \mathrm{ob}\, C} E_A$. But for a general functor p the fibres over distinct objects A and B may be entirely unrelated, even when there is a morphism $f : A \to B$. That is not the case in $p : \mathsf{Pred} \to \mathsf{Set}$, or in fibrations more generally.

In $p : \mathsf{Pred} \to \mathsf{Set}$, pulling back along $f : A \to B$ induces a functor $f^* : \mathsf{Pred}_B \to \mathsf{Pred}_A$ and rewriting the defining property of the pullback of p along f in terms of the functor p we get the definition of a cartesian map.

Definition 3.1 If $p : E \to C$ is a functor, then $F : e_1 \to e_2$ is said to be *cartesian with respect to* p iff for any $G : e_0 \to e_2$ and any factorisation $p(G) = p(F) \circ h$, then there is a unique map $H : e_0 \to e_1$ such that $G = F \circ H$ and $p(H) = h$.

Definition 3.2 A functor $p : E \to C$ is a *fibration* if for any $f : c_1 \to p(e_2)$ there is a cartesian map $F : e_1 \to e_2$ such that $p(F) = f$ (and hence $p(e_1) = c_1$). This map is called a *cartesian lifting* of f.

The next lemma collects properties of $p : \mathsf{Pred} \to \mathsf{Set}$, and follows immediately from the definitions.

Lemma 3.3 *For* $p : \mathsf{Pred} \to \mathsf{Set}$*:*

(i) $f : (P, A) \to (Q, B)$ *is cartesian if and only if* $P = f^{-1}(Q)$ *(i.e. the implicit commutative square is a pullback)*

(ii) p *is a fibration*

(iii) *the fibre of* p *over* A *is the lattice of subsets of* A*.*

It follows easily from the definition that cartesian liftings are unique up to unique isomorphism. Specifically, if $F : e_1 \to e_2$ and $F' : e_1' \to e_2$ are both cartesian liftings of $f : c_1 = p(e_1) = p(e_1') \to e_2$, then there is a unique map $i : e_1 \to e_1'$ such that $F = i; F'$ and $p(i)$ is the identity on c_1. The map i is necessarily an isomorphism.

Moreover, suppose we are given $f : A \to B$ and for each object e_2 of E_B a chosen cartesian lifting of f, $F_e : f^*(e) \to e$. Then f^* (defined on objects) extends uniquely to a functor $f^* : E_A \to E_B$. This does not mean that if we are given chosen cartesian liftings, then we can use these to derive a functor $p^* : C^{\mathrm{op}} \to \mathsf{Cat}$. We may not be able to choose our cartesian liftings so that the composite of the liftings along morphisms g and f are always the liftings along the composite gf. Instead we get a *pseudo-functor*, in which the functor $p^*(gf)$ is naturally isomorphic to $(p^*g) \circ (p^*f)$. The components of these natural isomorphisms are vertical, they are maps in the relevant fibres.

However, we will be working mainly with fibrations where the fibres are partial orders. In this case, the only isomorphisms in a fibre are identities, cartesian liftings are thus unique, and we do get a functor.

Note that in any fibration $p : E \to C$, any morphism $f : e_1 \to e_2$ in E factors as a vertical map followed by a cartesian (the cartesian is the lifting of $p(f)$, and the vertical is obtained by factoring f through the cartesian over the triangle $\mathrm{id}; p(f) = p(f)$). This factorisation is unique up to isomorphism, so when fibres are partially ordered it is unique. That applies in particular to $p : \mathsf{Pred} \to \mathsf{Set}$ and to $q : \mathsf{Rel} \to \mathsf{Set} \times \mathsf{Set}$.

A *morphism of fibrations* (or *cartesian functor*) is given by a commutative square in Cat as on the left below in which the topmost functor preserves cartesian maps. Since any Φ over ϕ preserves vertical maps, a cartesian functor also preserves the factorisation of morphisms into vertical and cartesian. In any square of functors as on the left below, Φ induces functors between the fibres: $\Phi_A : E_A \to E'_{\phi(A)}$. Since a morphism of fibrations preserves cartesian maps, those functors are natural with respect to the reindexing functors f^*, at least up to isomorphism, as on the right:

$$\begin{tikzcd}
E \arrow[r, "\Phi"] \arrow[d, "p"'] & E' \arrow[d, "p'"] \\
C \arrow[r, "\phi"] & C'
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
E_B \arrow[r, "\Phi_B"] \arrow[d, "f^*"] \arrow[dr, phantom, "\rotatebox{45}{$\Leftrightarrow$}"] & E_{\phi B} \arrow[d, "(\phi f)^*"] \\
E_A \arrow[r, "\Phi_A"] & E_{\phi A}
\end{tikzcd}$$

The exact formal expression of this naturality for general fibrations requires detailed coherence conditions for pseudo-natural transformations, while the formal definition in the fibrational approach is simple. In our case, we are dealing with partial orders and might reasonably adopt this alternative approach.

The interpretation of logical relations requires extending type structure from the base category C to the global category E. In order to do that we need structure defined by adjunctions. Formalisation of that requires 2-cells between functors.

Definition 3.4 The 2-category $\mathsf{Cat}^{\downarrow}$ has:

objects (0-cells): an object is a functor $p : E \to C$

morphisms (1-cells): a morphism from $p : E \to C$ to $p' : E' \to C'$ is a pair of functors $\Phi : E \to E'$ and $\phi : C \to C'$ such that $p'\Phi = \phi p$.

2-cells: a 2-cell from (Φ, ϕ) to (Ψ, ψ) is a pair of natural transformations $X : \Phi \Rightarrow \Psi$ and $\chi : \phi \Rightarrow \psi$ such that $p'X = \chi p$.

$$\begin{tikzcd}
E \arrow[r, bend left, "\Phi"] \arrow[r, bend right, "\Psi"'] \arrow[r, phantom, "\Downarrow X"] \arrow[d, "p"'] & E' \arrow[d, "p'"] \\
C \arrow[r, bend right, "\psi"'] & C'
\end{tikzcd}
=
\begin{tikzcd}
E \arrow[r, bend left, "\Phi"] \arrow[d, "p"'] & E' \arrow[d, "p'"] \\
C \arrow[r, bend left, "\phi"] \arrow[r, bend right, "\psi"'] \arrow[r, phantom, "\Downarrow \chi"] & C'
\end{tikzcd}$$

The 2-category Fib is the subcategory of $\mathsf{Cat}^{\downarrow}$ in which objects are fibrations, morphisms are cartesian functors and 2-cells are as in $\mathsf{Cat}^{\downarrow}$ (it is a locally full sub-2-category).

Suppose $f : C \to C'$ and $p' : E' \to C'$ is a fibration. Then a simple check shows that the pullback of p' along f in Cat is also a fibration, and that the pullback square is a morphism of fibrations.

$$\begin{tikzcd}
f^*E' \arrow[r, "F"] \arrow[d, "f^*p'"'] \arrow[dr, phantom, "\lrcorner", very near start] & E' \arrow[d, "p'"] \\
C \arrow[r, "f"] & C'
\end{tikzcd}$$

Note that if A is an object of C, then the fibre of f^*p over A is (isomorphic to) the fibre of p' over $f(A)$. A particular example of this is Rel, which is the pullback of Pred along the product map $\mathsf{Set} \times \mathsf{Set} \to \mathsf{Set}$.

$$\begin{tikzcd}
\mathsf{Rel} \arrow[r] \arrow[d, "q"'] \arrow[dr, phantom, "\lrcorner", very near start] & \mathsf{Pred} \arrow[d, "p"] \\
\mathsf{Set} \times \mathsf{Set} \arrow[r, "(-)\times(-)"] & \mathsf{Set}
\end{tikzcd}$$

We shall be particularly interested in two kinds of fibrations, semantic and type-theoretic/logical.

Our paradigmatic example of a semantic fibration is Pred. In a semantic fibration, the underlying category C is a category of (semantic) mathematical objects, such as Set. The fibre of E over an object A of C is (derived from) a class of maps into A in C. In the case of Pred, these are the inclusions of subsets. Reindexing is given by pullback, and hence cartesian maps correspond to pullback squares. If we need to interpret logical formulae, then we use the standard trick in which an n-ary predicate on $A_1, \ldots, A_n$ is interpreted as an element of the fibre over $A_1 \times \ldots \times A_n$. In the case of Pred this says that an n-ary predicate on $A_1, \ldots, A_n$ is interpreted as a subset of, and hence a unary predicate on $A_1 \times \ldots \times A_n$.

In a logical/type-theoretic fibration, the objects of C consist of contexts, usually a list of variable/type bindings: $x_1 : A_1, \ldots, x_n : A_n$. Objects of the fibre over a context consist of types (if modelling type theory) or predicates (if logic) in the given context. An example would be: $x : A, y : A, z : B \vdash x = y$. Morphisms in C are substitutions. So a morphism from $y_1 : B_1, \ldots, y_m : B_m$ to $x_1 : A_1, \ldots, x_n : A_n$ is given an n-tuple of terms in context $y_1 : B_1, \ldots, y_m : B_m$, $y_1 : B_1, \ldots, y_m : B_m \vdash t_i : A_i$, representing the substitution $(x_i \mapsto t_i)_{i=1\ldots n}$. Reindexing, and hence the cartesian maps of the fibration, corresponds to performing these substitutions in the types or in the predicates. If contexts admit the laws of contraction, weakening and exchange, then the base category has finite products. If (in addition) the type theory itself has finite product types, then the context $x_1 : A_1, \ldots, x_n : A_n$ can be replaced by $x : A_1 \times \ldots \times A_n$, and so can be taken to consist only of a single type.

4 Adjunctions

Much type-theoretic structure is defined categorically through adjunctions. This includes products, coproducts and function spaces, though not monadic structure.

The standard definition of an adjunction between a functor $F : C \to D$ and a functor $G : D \to C$ (written $F \dashv G : C \to D$) is an isomorphism $D(Fa, b) \cong C(a, Gb)$ natural in a and b. But this is equivalent to a formulation that allows us to define an adjunction between morphisms in an arbitrary 2-category:

Definition 4.1 An *adjunction $F \dashv G : C \to D$ in a 2-category* is a pair of 2-cells $\eta : 1_C \to GF$ and $\epsilon : FG \to 1_D$ satisfying

$$\begin{tikzcd} C \arrow[r, "F"] \arrow[d, "\mathrm{id}"'] \arrow[dr, phantom, "\eta\Rightarrow" near start] & D \arrow[d, "\mathrm{id}"] \arrow[dl, "G"'] \arrow[dl, phantom, "\Rightarrow\epsilon" near end, shift left=4] \\ C \arrow[r, "F"'] & D \end{tikzcd} = \begin{tikzcd} C \arrow[r, "F"] \arrow[d, "\mathrm{id}"'] & D \arrow[d, "\mathrm{id}"] \\ C \arrow[r, "F"'] & D \end{tikzcd} \qquad \begin{tikzcd} D \arrow[r, "G"] \arrow[d, "\mathrm{id}"'] \arrow[dr, phantom, "\epsilon\Leftarrow" near start] & C \arrow[d, "\mathrm{id}"] \arrow[dl, "F"'] \arrow[dl, phantom, "\Leftarrow\eta" near end, shift left=4] \\ D \arrow[r, "G"'] & C \end{tikzcd} = \begin{tikzcd} D \arrow[r, "G"] \arrow[d, "\mathrm{id}"'] & C \arrow[d, "\mathrm{id}"] \\ D \arrow[r, "G"'] & C \end{tikzcd}$$

This gives us the definition of an adjunction in $\mathsf{Cat}^{\downarrow}$ and an adjunction in Fib. Unpacking these definitions we see that in both cases an adjunction between

$$\begin{tikzcd} E \arrow[r, "\Phi"] \arrow[d, "p"'] & E' \arrow[d, "p'"] \\ C \arrow[r, "\phi"'] & C' \end{tikzcd} \quad \text{and} \quad \begin{tikzcd} E' \arrow[r, "\Psi"] \arrow[d, "p"'] & E \arrow[d, "p'"] \\ C' \arrow[r, "\psi"'] & C \end{tikzcd}$$

amounts to an adjunction in the base $\phi \dashv \psi$ and an adjunction between the global categories $\Phi \dashv \Psi$ which live over each other in the appropriate sense. Specifically, this means that an adjunction $(\Phi, \phi) \dashv (\Psi, \psi)$ in $\mathsf{Cat}^{\downarrow}$ is exactly a map (p, p') of adjunctions as in Mac Lane [11] IV.7.

We will also use a third characterisation of adjunctions.

Lemma 4.2 *Suppose $F \dashv G : C \to D$, and let $\epsilon : FGd \to d$ be the adjoint of* $\mathrm{id} : Gd \to Gd$ *(the counit of the adjunction). Then for any $f : Fc \to d$, $\hat{f}$, the adjoint of f, is the unique map $\hat{f} : c \to Gd$ such that $f = F(\hat{f}); \epsilon$.*

This tells us that if we consider the comma category $(F \downarrow d)$, whose objects are morphisms in D $Fc \to d$, and whose morphisms are maps f in C, such that $F(f)$ makes the obvious triangle commute, then $\epsilon : FGd \to d$ is terminal in $(F \downarrow d)$. A converse is also true. Given F, and for each object d of D a choice of terminal object $\epsilon_d : F(c_d) \to d$ in $(F \downarrow d)$, then there is a unique functor $G : D \to C$ such that $Gd = c_d$, $F \dashv G$ and ϵ is the counit of the adjunction.

A dual result links the other direction of the adjunction and the unit (the adjoint of $\mathrm{id} : Fa \to Fa$): for any $g : a \to Gb$, $\hat{g}$, the adjoint of g, is the unique map $\hat{g} : Fa \to b$ such that $g = \eta; G(\hat{g})$.

5 Adjunctions between Fibrations

Each map in a fibration factors as a vertical part (something that lives in a single fibre), followed by a cartesian part (something that represents a straightforward translation between fibres). In this section we reproduce Hermida's result that adjunctions between fibrations factor in a similar way, and extend it slightly to morphisms of categories.

5.1 Change of base

The equivalent of a cartesian map in a fibration, is a cartesian functor that arises through change of base. The first result says that an adjunction in the base extends to an adjunction between a fibration and its change of base.

Lemma 5.1 (Hermida, [4,5]) *Suppose $p : C \to A$ is a fibration, and $f \dashv g : B \to A$ is an adjunction, then there is a fibred adjunction*

$$\begin{tikzcd}
f^*C \arrow[r, bend left, "p^*f"] \arrow[d, "f^*p"'] \arrow[r, phantom, "\perp"] & C \arrow[l, bend left, "\hat{g}"] \arrow[d, "p"] \\
B \arrow[r, bend left, "f"] \arrow[r, phantom, "\perp"] & A \arrow[l, bend left, "g"]
\end{tikzcd}$$

Proof. Let ϵ be the counit of $f \dashv g$ and c an object of C. Then define $\hat{g}(c) = (g(c), \epsilon^*c)$, where $\overline{\epsilon} : \epsilon^*c \to c$ is a (chosen) cartesian lifting of $\epsilon_{pc} : fgpc \to pc$. □

Looking closely at the proof, we have only used the fibration structure to justify the existence of $\overline{\epsilon}$. So the same proof gives a result for arbitrary categories:

Corollary 5.2 *Suppose $q : C \to A$, and $f \dashv g : B \to A$ is an adjunction, and for all $c \in C$, the counit of $f \dashv g$, $\epsilon : gfpc \to pc$ has a cartesian lifting to c, then there is an adjunction $p^*f \dashv \hat{g} : p^*C \to C$ over the original adjunction, as in section 4, giving an adjunction in $\mathsf{Cat}^{\downarrow}$.*

Taking the opposite of all categories we get a corresponding result for left adjoints:

Corollary 5.3 *Suppose $q : C \to A$, and $f : B \to A$ has a left adjoint g. Suppose moreover that the unit of the adjunction has a cocartesian lifting to C, then $p^*f : p^*C \to C$ has a left adjoint $\hat{g}$, and the adjunction $\hat{g} \vdash p^*f$ lies over the original adjunction $g \vdash f$.*

5.2 Factorisation of adjunctions

These results can now be extended to arbitrary adjunctions. We show that under some lifting conditions, an adjunction can be composed of an adjunction over a fixed based (the vertical part) and an adjunction obtained from change of base as in Lemma 5.2 (the cartesian, horizontal part). The following is also a vanilla extension of Hermida, [4,5]. We do not require the functors p and q to be fibrations.

Proposition 5.4 *Suppose $\overline{f} : D \to C$ over $f : B \to A$, that f has a right (resp. left) adjoint g, and that both the unit, η and counit ϵ of that adjunction have cartesian liftings to arbitrary objects of C and D Let $\overline{f}$ factor through the pullback f^*C as follows:*

$$\begin{tikzcd}
D \arrow[r, bend left, "\overline{f}"] \arrow[d, "q"'] \arrow[r, phantom, "\perp"] & C \arrow[l, bend left, dashed, "\overline{g}"] \arrow[d, "p"] \\
B \arrow[r, bend left, "f"] \arrow[r, phantom, "\perp"] & A \arrow[l, bend left, "g"]
\end{tikzcd}
\quad = \quad
\begin{tikzcd}
D \arrow[r, bend left, "\tilde{f}"] \arrow[r, phantom, "\perp"] \arrow[rr, bend left=40, "\overline{f}"] \arrow[dr, "q"'] & f^*C \arrow[l, bend left, dashed, "\tilde{g}"] \arrow[r, bend left, "p^*f"] \arrow[r, phantom, "\perp"] \arrow[d, "f^*p"] & C \arrow[l, bend left, dashed, "\hat{g}"] \arrow[d, "p"] \\
& B \arrow[r, bend left, "f"] \arrow[r, phantom, "\perp"] & A \arrow[l, bend left, "g"]
\end{tikzcd}$$

Then the following are equivalent:

(i) *$\overline{f}$ has a right (resp. left) adjoint $\overline{g}$ over the adjunction $f \dashv g$*

(ii) *$\tilde{f}$ has a right (resp. left) adjoint $\tilde{g}$ over the identity on B.*

Proof. For right adjoint (i)→(ii): Suppose $(b, c) \in \mathrm{ob}(f^*C)$. Let $\tilde{\eta} : \eta^*c \to \overline{g}c$ be a (chosen) cartesian lifting of the unit $\eta : b \to gfb = gpc = q\overline{g}c$. Define $\tilde{g}(b, c) = \eta^*c$. As before, $\tilde{g}$ extends canonically to a functor over the identity on B.

To establish the adjunction, suppose $\beta = (\beta^1, \beta^2) : \tilde{f}d = (qd, \overline{f}d) \to (b, c)$ in f^*C. We have $f\beta^1 = p\beta^2 : fqd = p\overline{f}d \to fb = pc$. Using the adjunction in the base, the right adjoint of this is $(gf\beta^1)\eta : qd \to gfb$. But η

is natural, so $(gf\beta^1)\eta = \eta\beta^1$. Turning now to the adjunction $\overline{f} \dashv \overline{g}$, the right adjoint $\widehat{\beta} : d \to \overline{g}c$ of $\beta^2 : \overline{f}d \to c$ lives over $\eta\beta^1$. Since $\tilde{\eta}$ is cartesian over η, there is a unique $\tilde{\beta} : d \to \eta^* c = \tilde{g}(b,c)$, lying over β^1, such that $\tilde{\eta}\tilde{\beta} = \widehat{\beta}$. This construction is natural in both variables, and hence gives an adjunction lying over the identity adjunction on B as required.

For left adjoint: apply the result to the duals of all categories. □

This decomposition of adjoints says that right adjoints are obtained in two parts. The first part is a cartesian lifting. In terms of standard predicate structure, this amounts to substituting terms into the predicate given as argument. The second part is an adjunction between two fibrations over the same base. This part depends on the existence of particular logical structure. For example, we will see that conjunction is used to give products, disjunction is used for sums (using the dual result for left adjoints), and implication and first order quantification produce function spaces.

5.3 Examples

In this section we look out how the construction above helps define structure on $\mathsf{Pred} \to \mathsf{Set}$.

Example 5.5 [Products] Product is right adjoint to the diagonal:

$$\begin{tikzcd} \mathsf{Set} \arrow[r, bend left, "\Delta"] \arrow[r, phantom, "\perp"] & \mathsf{Set} \times \mathsf{Set} \arrow[l, bend left, "\times"] \end{tikzcd}$$

Consider the pullback of $\mathsf{Pred} \times \mathsf{Pred}$ along Δ:

$$\begin{tikzcd}[column sep=large] \mathsf{Pred} \arrow[rr, bend left=40, "\Delta"] \arrow[r, bend left, "\tilde{\Delta}"] \arrow[r, phantom, "\perp"] \arrow[dr, "p"'] & \Delta^*(\mathsf{Pred} \times \mathsf{Pred}) \arrow[l, bend left, dashed, "\wedge"] \arrow[r, bend left, "(p\times p)^*\Delta"] \arrow[r, phantom, "\perp"] \arrow[d, "\Delta^*(p\times p)"] & \mathsf{Pred} \times \mathsf{Pred} \arrow[l, bend left, dashed, "\hat{g}"] \arrow[d, "p\times p"] \\ & \mathsf{Set} \arrow[r, bend left, "\Delta"] \arrow[r, phantom, "\perp"] & \mathsf{Set} \times \mathsf{Set} \arrow[l, bend left, "\times"] \end{tikzcd}$$

On Pred, Δ maps a predicate P on set A to the pair of predicates (P, P) on the pair of sets (A, A). The fibre of $\Delta^*(\mathsf{Pred} \times \mathsf{Pred})$ over the set A is $\mathsf{Pred}(A) \times \mathsf{Pred}(A)$, so Δ factors through $\Delta^*(\mathsf{Pred} \times \mathsf{Pred})$ by $\tilde{\Delta}$, where $\tilde{\Delta}$ maps P to (P, P) in the fibre of $\Delta^*(\mathsf{Pred} \times \mathsf{Pred})$ over A. $(p \times p)^*\Delta$ now maps this to (P, P) in the fibre of $\mathsf{Pred} \times \mathsf{Pred}$ over $A \times A$.

The adjoint of $(p \times p)^*\Delta$, as per Lemma 5.1, is given by lifting along the counit of the adjunction in the base. That counit is the product of the projections $\epsilon : (a_0 \times a_1, a_0 \times a_1) \to (a_0, a_1)$: $\lambda(x, y).\ (\pi_0(x), \pi_1(y))$. Hence, given predicates P on A and Q on B, $\hat{g}$ maps (P, Q) over (A, B) to $(\pi_0^* P, \pi_1^* Q)$ over $A \times B$.

Putting this in more logical terms: if $a : A \vdash P(a) : \mathsf{Pred}$ and $b : B \vdash Q(b) : \mathsf{Pred}$, then the result of applying $\hat{g}$ to this pair of predicates is the pair got by substituting $a \mapsto \pi_0(x)$ into $P(a)$, and $b \mapsto \pi_1(x)$ into $Q(b)$: $x : A \times B \vdash (P(\pi_0 x), Q(\pi_1 x)) : \mathsf{Pred} \times \mathsf{Pred}$.

The point of this operation is to get two predicates on the same type. This is obscured if we write something like: $a : A, b : B \vdash (P(a), Q(b)) : \mathsf{Pred} \times \mathsf{Pred}$.

The adjoint of $\tilde{\Delta}$ is $\wedge$, which takes a pair (P_1, P_2) of predicates on A to their intersection, $P_1 \wedge P_2$. From a logical perspective, this is conjunction.

Putting this together, we get that the right adjoint to $\Delta : \mathsf{Pred} \to \Delta \times \Delta$ takes a pair of predicates (P, Q) over (A, B) (written as $(P(a), Q(b))$ to the predicate $P(a) \wedge Q(b)$ on $A \times B$. This is what we expect. It would be easy to calculate this directly, but the point is to see this as coming from our general framework for adjunctions.

Example 5.6 [Function spaces] For function spaces we have:

$$\begin{tikzcd}
\mathsf{Pred} \arrow[r, bend left, "-\wedge P"] \arrow[r, phantom, "\perp"] \arrow[rr, bend left=40, "-\times P"] \arrow[dr, "p"'] & (-\times A)^*\mathsf{Pred} \arrow[l, bend left, dashed, "\tilde{g}"] \arrow[r, bend left, "p^*(-\times A)"] \arrow[r, phantom, "\perp"] \arrow[d, "(-\times A)^* p"] & \mathsf{Pred} \arrow[l, bend left, dashed, "\hat{g}"] \arrow[d, "p"] \\
 & \mathsf{Set} \arrow[r, bend left, "-\times A"] \arrow[r, phantom, "\perp"] & \mathsf{Set} \arrow[l, bend left, "A\to -"]
\end{tikzcd}$$

Here, P is a predicate on A, and $-\times P$ is the functor just calculated, sending a predicate Q on B to $Q(b)\wedge P(a)$ on $B\times A$. As before, we compute the right adjoint in stages.

Given (Q, B) in $\mathsf{Pred}(B)$, the counit in the base is evaluation: $\mathrm{ev}_B : (A \to B)\times A \to B$.

$\hat{g}(Q, B)$ is then $\mathrm{ev}^*_B(Q, B)$, the pullback of (Q, B) along ev_B. Reading this from a logical perspective, this amounts to a substitution $b \mapsto f(a)$ in $Q(b)$. We write the result informally as $f : A \to B, a : A \vdash Q(f(a))$ Pred, or more formally as $x : (A \to B) \times A \vdash Q((\pi_0 x)(\pi_1 x))$ Pred.

Now, the fibre of $(-\times A)^*\mathsf{Pred}$ over a set C is $\mathsf{Pred}(C\times A)$, and $-\times P$ factors through this via $-\wedge P$. Specifically, if R is a predicate on C, then $R\wedge P$ is $R(c)\wedge P(a)$, a predicate on $C\times A$.

The right adjoint to this, $\tilde{g}$, takes a predicate $S(c, a)$ to $\forall a \in A. P(a) \to S(c, a)$. This exists in Pred, but if we are working in a fibration encoding a more general logic, it will be sufficient for that logic to have implication and first-order quantification.

Putting this with the previous adjoint, we get that Q is sent to a predicate on $A \to B$ which holds of f iff $\forall a \in A, P(a) \to Q(f(a))$, as expected.

Example 5.7 [Coproducts] The theory for coproducts is similar to that for products. The overall structure of the diagram below is identical to that for products, but we have left adjoints rather than right. In particular, Δ, $\tilde{\Delta}$ and $\Delta^*(\mathsf{Pred}\times\mathsf{Pred})$, etc are exactly as before. The left adjoint to $\tilde{\Delta}$ is disjunction in the fibres. To get a left adjoint to $(p\times p)^*\Delta$ we use a cocartesian lifting along the unit in the base. The unit in the base is $\lambda(a, b).(\mathrm{inl}\, a, \mathrm{inr}\, b) : (A, B) \to (A + B, A + B)$. The cocartesian lifting of this takes a pair of predicates P on A and Q on B to the predicates `case x of {inl a -> P(a);  inr b -> False}` and `case x of {inl a -> False; inr b -> Q(b)}`, both predicates on $A+B$. This is a weak form of existential quantification, significantly weaker than having cocartesian liftings along arbitrary maps. Cocartesian liftings along product projections correspond to standard existential quantification.

$$\begin{tikzcd}
\mathsf{Pred} \arrow[r, bend left, "\tilde{\Delta}"] \arrow[r, phantom, "\top"] \arrow[rr, bend left=40, "\Delta"] \arrow[dr, "p"'] & \Delta^*(\mathsf{Pred}\times\mathsf{Pred}) \arrow[l, bend left, dashed, "\vee"] \arrow[r, bend left, "(p\times p)^*\Delta"] \arrow[r, phantom, "\top"] \arrow[d, "\Delta^*(p\times p)"] & \mathsf{Pred}\times\mathsf{Pred} \arrow[l, bend left, dashed, "\hat{g}"] \arrow[d, "p\times p"] \\
 & \mathsf{Set} \arrow[r, bend left, "\Delta"] \arrow[r, phantom, "\top"] & \mathsf{Set}\times\mathsf{Set} \arrow[l, bend left, "+"]
\end{tikzcd}$$

6 Change of base

6.1 Results

This section is about the implication of the results in section 5 for situations obtained by change of base. We want to transfer structure from a primary fibration to another obtained from it by pullback along a suitable change of base. The key example is pullback of Pred along $\Pi : \mathsf{Set}^n \to \mathsf{Set}$, which expresses the fact that an n-ary relation can be regarded as a unary predicate on the product of the underlying sets.

First, we observe that pullback preserves cartesianness, even when the categories involved are not fibrations.

Lemma 6.1 *Suppose the diagram below is a pullback of categories, and that $\overline{\alpha} : P_0 \to P_1$ is cartesian in P over $F\alpha$ where $\alpha : B_0 \to B_1$. Then $(\alpha, \overline{\alpha})$ is cartesian in F^*P over α.*

$$\begin{tikzcd}
F^*P \arrow[r, "p^*F"] \arrow[d, "F^*p"] & P \arrow[d, "p"] \\
B \arrow[r, "F"] & A
\end{tikzcd}$$

For our purposes we consider the following situation in the base. We have a commutative square, as below, in which the horizontal arrows F and G have right adjoints F^r and G^r. It is not necessarily the case that $KG^r = F^rH$, though there is a canonical natural transformation between them, the *mate* of the identity on $FK = HG$. We will consider an adjunction of categories over $F \dashv F^r$, pulled back along H and K.

Lemma 6.2 *Suppose we are given a commutative square as above, and a category P over A. This produces the cube below in which the front face and the left and right sides are pullbacks. The functor $\overline{G}$ is obtained as the obvious pullback factorisation. The functors F and G have right adjoints F^r and G^r.*

$$\begin{tikzcd}
K^*F^*P \arrow[rr, dashed, "\overline{G}"] \arrow[dr] \arrow[dd, "K^*F^*p"'] & & H^*P \arrow[ll, dashed, bend left] \arrow[ll, phantom, "\perp"] \arrow[dr] \arrow[dd] & \\
& F^*P \arrow[rr] \arrow[dd] & & P \arrow[dd, "p"] \\
D \arrow[rr, "G"] \arrow[dr, "K"'] & & C \arrow[ll, dashed, bend left, "G^r"] \arrow[ll, phantom, "\perp"] \arrow[dr, "H"] & \\
& B \arrow[rr, "F"] & & A \arrow[ll, dashed, bend left, "F^r"] \arrow[ll, phantom, "\perp"]
\end{tikzcd}$$

Suppose ϵ is the counit of $G \dashv G^r$, and that $H\epsilon$ has cartesian liftings to P. Then $\overline{G}$ has a right adjoint over G^r.

Proof. Since pullbacks compose, the composite of the front and left faces is also a pullback. Hence so is the composite of the back and right faces. This in turn implies that the back face is a pullback. We now apply lemma 6.1, to show that ϵ has cartesian liftings to H^*P, and lemma 5.2 to yield the result. □

Lemma 6.3 *Suppose $G \dashv G^r$ is an adjunction in Cat/B, then pulling back along a functor $K : D \to B$ gives an adjunction over D.*

Proof. Pullback along K gives a 2-functor $K^* : \mathsf{Cat}/B \to \mathsf{Cat}/D$. K^* therefore preserves unit and counit of the adjunction in Cat/B, and the fact that they satisfy the triangle identities. It therefore preserves the adjunction. □

The following proposition assembles these results together to give an account of changing the base of an adjunction between p and q by puling back along functors H and K as in the diagram below. The input data here consists of the front and bottom faces of the cube

$$\begin{tikzcd}
K^*Q \arrow[rr, dashed, "\overline{G}"] \arrow[dr] \arrow[dd, "K^*q"'] & & H^*P \arrow[dr] \arrow[dd, "H^*p"] & \\
& Q \arrow[rr, "\overline{F}"] \arrow[dd, "q"] & & P \arrow[ll, dashed, bend left, "\overline{F}^r"] \arrow[ll, phantom, "\perp"] \arrow[dd, "p"] \\
D \arrow[rr, "G"] \arrow[dr, "K"'] & & C \arrow[ll, dashed, bend left, "G^r"] \arrow[ll, phantom, "\perp"] \arrow[dr, "H"] & \\
& B \arrow[rr, "F"] & & A \arrow[ll, dashed, bend left, "F^r"] \arrow[ll, phantom, "\perp"]
\end{tikzcd}$$

so that (q, p) is a map of adjunctions and $HG = FK$. $\overline{G}$ is the map $K^*Q \to H^*P$ obtained by factorisation through pullback.

Proposition 6.4 *Given a cube as above, suppose the front face is a map of adjunctions obtained by factorisation through the pullback as in proposition 5.4, and that $HG = FK$, Suppose further that the counit*

$\epsilon : GG^r \to 1_C$ of $G \dashv G^r$ has cartesian liftings to H^*P (e.g. if $H\epsilon$ has cartesian liftings to P). Then $\overline{G} : K^*Q \to H^*p$ is the left adjoint in an adjunction living over $G \dashv G^r$, in which the right adjoint is also constructed by factorisation through the pullback as in 5.4.

Proof. This follows immediately from lemmas 6.2 and 6.3. □

6.2 Examples

Example 6.5 [products] We consider the binary product of n-ary relations.

$$\begin{tikzcd}
\mathsf{Rel}_n \arrow[rr, dashed, "\Delta"] \arrow[dr] \arrow[dd] & & \mathsf{Rel}_n \times \mathsf{Rel}_n \arrow[dr] \arrow[dd] & \\
& \mathsf{Pred} \arrow[rr, "\Delta"] \arrow[rr, phantom, "\perp"'] \arrow[dd] & & \mathsf{Pred} \times \mathsf{Pred} \arrow[ll, dashed, bend left] \arrow[dd] \\
\mathsf{Set}^n \arrow[rr, "\Delta"] \arrow[rr, phantom, "\perp"'] \arrow[dr, "\Pi"'] & & \mathsf{Set}^n \times \mathsf{Set}^n \arrow[ll, dashed, bend left, "\times"] \arrow[dr, "\Pi\times\Pi"] & \\
& \mathsf{Set} \arrow[rr, "\Delta"] \arrow[rr, phantom, "\perp"'] & & \mathsf{Set} \times \mathsf{Set} \arrow[ll, dashed, bend left, "\times"]
\end{tikzcd}$$

The front face of the cube is the construction of the binary product of logical predicates that we saw in example 5.5.

Relations are constructed by pullback from Pred along the product functor $\Pi : \mathsf{Set}^n \to \mathsf{Set}$. This leads us to the bottom face of the cube, which commutes. It is worth spelling this out.

Given an n-tuple of sets: $(A_i)_{i\in\{1\ldots n\}}$, taking this along the rear and right hand sides gives $(\Pi\times\Pi)\Delta(A_i) = (\Pi A_i, \Pi A_i)$ And taking it along the left and front gives $\Delta\Pi(A_i) = (\Pi A_i, \Pi A_i)$. These are equal, not just canonically isomorphic, and so we can apply the theory developed above immediately to show that the product of n-ary relations is constructed as we expect.

Example 6.6 [function spaces] To derive n-ary function spaces from unary ones, the diagram is:

$$\begin{tikzcd}
\mathsf{Rel}_n \arrow[rr, dashed] \arrow[dr] \arrow[dd] & & \mathsf{Rel}_n \arrow[dr] \arrow[dd] & \\
& \mathsf{Pred} \arrow[rr] \arrow[rr, phantom, "\perp"'] \arrow[dd] & & \mathsf{Pred} \arrow[ll, dashed, bend left] \arrow[dd] \\
\mathsf{Set}^n \arrow[rr, "(-\times A_i)"] \arrow[rr, phantom, "\perp"'] \arrow[dr, "\Pi"'] & & \mathsf{Set}^n \arrow[ll, dashed, bend left, "(A_i\to -)"] \arrow[dr, "\Pi"] & \\
& \mathsf{Set} \arrow[rr, "-\times\Pi A_i"] \arrow[rr, phantom, "\perp"'] & & \mathsf{Set} \arrow[ll, dashed, bend left, "(\Pi A_i)\to -"]
\end{tikzcd}$$

However the bottom face of this cube does not commute. Consider an object (B_i) of Set^n. Going left and down round the square we get $\Pi(B_i \times A_I)$, while going down and left we get $(\Pi B_i) \times (\Pi A_i)$. These are naturally isomorphic but not equal. Consequently we cannot simply apply the results above.

There are at least two ways out of this problem. The more principled is to redo the theory so that it will cope with commutation up to natural isomorphism. The technically simpler alternative is to fix the square so that it commutes. Replace the front left vertical edge of the cube by its product with the indiscrete category on $\mathrm{ob}\,\mathsf{Set}^n$. The objects of $\mathrm{ob}\,\mathsf{Set}^n \times \mathsf{Set}$ are pairs $((B_i), C)$ where (B_i) is an n-tuple of sets and C is a set. Morphisms $((B_i), C) \to (B'_i), C')$ are functions $C \to C'$. This gives us a structure equivalent to the original, and with the same formal properties. But we can tinker with the two functors that form edges of the bottom square. We take the first to be given by $(B_i) \mapsto ((B_i), \Pi B_i)$, and the second to be given by the natural extension to morphisms of:

$F((B_i), C) = C \times \Pi A_i$ if $C \neq \Pi B_i$

$F((B_i), C) = \Pi(A_i \times B_i)$ if $C = \Pi B_i$

The square now commutes and our theory applies. But since everything is equivalent to the original, the logical description of the function space still works.

Example 6.7 [coproducts] As before, the story for coproducts is the same as the story for products. It makes use of the same diagrams, but the theory is applied to the opposite categories, yielding left adjoints rather than right. In particular the bottom face of the cube is obtained from the bottom face of the diagram for products by applying op to all of the categories involved. It therefore commutes exactly.

7 Monads

Monads provide another approach to defining structure. Given a category B representing types, a category P representing a predicate logic for B, and a monad T on B, then there is no fully general method of extending T to a monad on P. But there are ways that work under some circumstances.

7.1 Initial cocartesian lifting of a monad

In [8,9,10] such an extension of a monad from the base B to the total category P of a fibration (logic) is called a *lifting*. Katsumata's method involves embedding into a given continuation or codensity monad, which relies both on closed structure and a somewhat arbitrary parameter. Kammar's method seeks an *initial lift* for the given monad. We can indeed provide such an initial lift under very minimal conditions on p:

Theorem 7.1 (Initial lifting of a monad) *Given a monad $(T : B \to B, \eta, \mu)$, whenever $p : E \to B$ admits cocartesian liftings of η, the category* $\mathsf{Lift}(T)$ *of monad liftings admits an initial object.*

Proof. Define the lifted monad $(\overline{T} : E \to E, \overline{\eta}, \overline{\mu})$ as follows: for an object $X \in E$, let $\overline{\eta}_X : X \to \overline{T}X$ be a cocartesian lifting of $\eta_{pX} : pX \to TpX$ at X. This defines both $\overline{T}$ and $\overline{\eta}$. Using both unit equations, we have $\mu \circ (\eta \bullet \eta) = \eta$ and since $\overline{\eta} \bullet \overline{\eta}$ is cocartesian, there is a unique transformation $\overline{\mu} : \overline{TT} \to \overline{T}$ over μ such that $\overline{\mu} \circ (\overline{\eta} \bullet \overline{\eta}) = \overline{\eta}$. Cocartesianness also proves the monad equations for these data, as well as the initiality. □

In general the lifted monad functor $\overline{T}$ will not be fibred even when p is a fibration, but it is cofibred.

The initial lifting of [8, Theorem 4.8] is a special case of this, under the assumption that the fibres of the functor p are small complete preorders. When p admits cartesian as well as cocartesian liftings, we have for $\eta_X : X \to TX$ the adjunction $\Sigma_{\eta_X} \dashv \eta^*_X : E_{TX} \to E_X$. Under the assumption that the fibres are small complete preorders, the adjoint functor theorem tells us that the left adjoint (= cocartesian lift) is computed as $\Sigma_{\eta_X}(y) = \bigwedge\{x \in E_{TX} | y \leq \eta^*_X x\}$. The condition $y \leq \eta^*_X x$ is equivalent, for a faithful functor p, to the existence of a morphism $y \to x$ over η_X. This latter is precisely the condition *the unit respects* y in the construction of the initial lifting of T in p. 254 of [8]. We are ignoring here the presence of algebraic operations in T, which we deal with separately in a follow up article.

It is worth noticing that the 'formula' for initial lift of a monad T, $\overline{T} = \Sigma_\eta(id)$ as presented in our theorem, is used in [1] for the interpretation of the $\diamond$-modality of their FIX-logic but without reference to it being a monad, let alone an initial lift.

7.2 Image factorisation and monads on monics

If P is presented as a subcategory of monics in C (e.g. Pred), then T is already defined on the objects of P, and we can take $T(P \rightarrowtail A)$ to be defined from $TP \to TA$. If T preserves monics in our subcategory, then this is fine. If not we can hope to use someimage factorisation $TP \twoheadrightarrow T'P \rightarrowtail TA$, as in [3]. This method of lifting has been recast in fibrational language, using cocartesian liftings and comprehension, in place of image factorisation in [2].

We use two particular monads as running examples: the list monad, L, and the continuation monad, C. The list monad is an example of algebraic structure, and the continuation monad is not.

7.2.1 Monads on Set *and unary predicates*

Let T be a monad on Set. Then T preserves monics.

Lemma 7.2 (i) *Let F : $\mathsf{Set} \to \mathsf{Set}$ be a functor, and $i : A \rightarrowtail B$ a monic, where $A \neq \emptyset$, then Fi is also monic.*

(ii) *Let M : $\mathsf{Set} \to \mathsf{Set}$ be a monad, and $i : A \rightarrowtail B$ any monic, then Mi is also monic.*

(iii) *Let M : $\mathsf{Set} \to \mathsf{Set}$ be a monad, then M extends to a functor* $\mathsf{Pred} \to \mathsf{Pred}$ *over* Set.

Proof. (i) i has a retraction which is preserved by F. (ii) If A is non-empty, then this follows from the previous remark. If A is empty, then there are two cases. If $M\emptyset = \emptyset$, then $Mi : \emptyset = M\emptyset = MA \to MB$ is automatically monic. If $M\emptyset \neq \emptyset$, then let r be any map $B \to M\emptyset$. MB is the free M-algebra on B, and therefore there is a unique M-algebra homomorphism $r^* : MB \to M\emptyset$ extending this. Mi is also an M-algebra homomorphism and hence so is the composite $r^*(Mi)$. Since $M\emptyset$ is the initial M-algebra, it must be the identity, and hence Mi is monic. (iii) Immediate. □

Example 7.3 The list monad: LB is the set of lists whose elements come from B, and so is a subset of LA.

The direct image powerset functor $P : \mathsf{Set} \to \mathsf{Set}$, where $P(f) = \{b | \exists a.f(a) = b\}$, also has the property that $PB \subseteq PA$ if $B \subseteq A$.

The continuation monad is defined by $CA = (A \to K) \to K$, where K is the set of continuations. $C\emptyset$ is isomorphic to K and contains the constant functionals. These are included in any type CA. If $B \subseteq A$, then $CB = (B \to K) \to K$, and if $j : B \to A$ is the inclusion of B in A, then Cj sends CB to

$$\{\Phi : (A \to K) \to K \mid g \restriction B = g' \restriction B \to \Phi g = \Phi g'\}.$$

But there is a difference from the point of view of fibrations: L and P are fibred functors, while C is not. From the logical perspective, this means that L works well with substitution, but C does not.

Lemma 7.4 *C is not a fibred functor.*

Proof. Let $A = \{0, 1\}$, $B = \{a, b, c\}$ and $f : B \to A$ be defined by: $fa = 0$, $fb = fc = 1$. Let $P = \{1\} \subset A$. Then $f^*P = \{b, c\}$. Given $\Psi : CB$, $(Cf)\Psi = \lambda g : A \to K.\ \Psi(g \circ f)$. Consider the functional $\Theta : CB$, defined by

$$\begin{aligned} \Theta h &= hb \quad \text{if } hb = hc \\ &= ha \quad \text{if } hb \neq hc \end{aligned}$$

Now, for any $g : A \to K$, $((Cf)\Theta)g = \Theta(g \circ f)$, but $(g \circ f)b = g1 = (g \circ f)c$, so $((Cf)\Theta)g = g1$. It follows that $(Cf)\Theta$ is in CP, and hence Θ is in $(Cf)^*(CP)$. However, the result of applying Θ to h can clearly depend on ha. Θ is therefore not in $C(f^*P)$. It follows that C is not a fibred functor. □

7.2.2 Monads on Set *and binary relations*

There is also a difference when it comes to binary, and more generally, n-ary relations. Suppose $R \rightarrowtail A \times B$, then $MR \rightarrowtail M(A \times B)$, and there is a canonical map $M(A \times B) \to MA \times MB$, but this is not necessarily monic, and hence the induced $MR \to MA \times MB$ is also not necessarily monic.

Lemma 7.5 *The map $(C\pi_A, C\pi_B) : C(A \times B) \to CA \times CB$ is not necessarily monic.*

Proof. $CX = (X \to K) \to K$. Consider the example where $K = 2$, $A = B = 2$. Then a simple cardinality argument shows that no map $C(A \times B) \to CA \times CB$ can be monic. $|C(A \times B)| = 2^{2^4} = 2^{16}$, while $|CA \times CB| = 2^{2^2} * 2^{2^2} = 2^8$. □

Some monads, such as the list monad, do have the property that this map is monic, and hence that they preserve binary relations.

Lemma 7.6 *If $i : R \rightarrowtail A \times B$, then $LR \rightarrowtail L(A \times B) \to LA \times LB$ is monic.*

Proof. Let $[(a_0, b_0), \ldots (a_n, b_n)]$ and $[(a'_0, b'_0), \ldots (a'_m, b'_m)]$ be two typical elements of LR. Then these map to $([a_0, \ldots, a_n], [b_0, \ldots, b_n])$ and $([a'_0, \ldots, a'_m], [b'_0, \ldots, b'_m])$ in $LA \times LB$. These are equal if and only if $[a_0, \ldots, a_n] = [a'_0, \ldots, a'_m]$ and $[b_0, \ldots, b_n] = [b'_0, \ldots, b'_m]$. But this implies $m = n$ and that $a_i = a'_i$ and $b_i = b'_i$ for all $i = 0 \ldots n$. So $[(a_0, b_0), \ldots (a_n, b_n)] = [(a'_0, b'_0), \ldots (a'_m, b'_m)]$. □

It may appear that this is because the list monad is algebraic, but not all algebraic monads have normal forms that can be used like this.

Example 7.7 A *Mal'cev operation* is a ternary operation $m(x, y, z)$ such that $m(x, x, z) = z$ and $m(x, z, z) = x$. An important example is $m(x, y, z) = xy^{-1}z$ in the theory of groups. Let M be the monad corresponding to an algebra freely generated by a single Mal'cev operation. Then $M(A \times B) \to MA \times MB$ is not necessarily monic. Consider $A = \{a, b\}$ and $B = \{0, 1\}$. Then $m((a, 0), (a, 1), (b, 1))$ maps to $m(a, a, b) = b$ in MA and to $m(0, 1, 1) = 0$ in MB, but $m((a, 0), (a, 1), (b, 1))$ is not equal to $(b, 0)$ in $M(A \times B)$. This can be shown by exhibiting any algebra with a Mal'cev operation and four elements u, v, w, x such that $m(u, v, w) \neq x$.

As a consequence, if M is the monad corresponding to the theory of a free Mal'cev operator, then $M4$ does not map injectively into $M2 \times M2$.

But factorisation will always work in Set. If M is any monad, then $MA \times MB$ has a natural M-algebra structure, and $M(A \times B) \to MA \times MB$ is an M-algebra homomorphism. Just as with standard algebraic varieties, sub-algebras are closed under arbitrary intersecion, and therefore we can take MR to be the intersection of all the sub-algebras containing the image of R under η.

References

[1] Crole, R. L. and A. M. Pitts, *New foundations for fixpoint computations: Fix-hyperdoctrines and the fix-logic*, Information and computation **98** (1992), pp. 171–210.

[2] Ghani, N., P. Johann and C. Fumex, *Fibrational induction rules for initial algebras*, in: *International Workshop on Computer Science Logic*, Springer, 2010, pp. 336–350.

[3] Goubault-Larrecq, J., S. Lasota and D. Nowak, *Logical relations for monadic types*, Mathematical Structures in Computer Science **18** (2008), pp. 1169–1217.

[4] Hermida, C., "Fibrations, logical predicates and related topics," Ph.D. thesis, PhD thesis, University of Edinburgh, 1993. Tech. Report ECS-LFCS-93-277. Also available as Aarhus Univ. DAIMI Tech. Report PB-462 (1993).

[5] Hermida, C., *Some properties of fib as a fibred 2-category*, Journal of Pure and Applied Algebra **134** (1999), pp. 83–109.

[6] Hermida, C., U. S. Reddy and E. P. Robinson, *Logical relations and parametricity - A reynolds programme for category theory and programming languages*, Electr. Notes Theor. Comput. Sci. **303** (2014), pp. 149–180.
URL https://doi.org/10.1016/j.entcs.2014.02.008

[7] Jacobs, B., "Categorical logic and type theory," Studies in logic and the foundations of mathematics **141**, Elsevier, 1999.

[8] Kammar, O. and D. McDermott, *Factorisation systems for logical relations and monadic lifting in type-and-effect system semantics*, Electronic Notes in Theoretical Computer Science **341** (2018), pp. 239–260.

[9] Katsumata, S., *A semantic formulation of tt-lifting and logical predicates for computational metalanguage*, in: C. L. Ong, editor, *Computer Science Logic, 19th International Workshop, CSL 2005, 14th Annual Conference of the EACSL, Oxford, UK, August 22-25, 2005, Proceedings*, Lecture Notes in Computer Science **3634** (2005), pp. 87–102.
URL https://doi.org/10.1007/11538363_8

[10] Katsumata, S. and T. Sato, *Codensity liftings of monads*, in: L. S. Moss and P. Soboci'nski, editors, *6th Conference on Algebra and Coalgebra in Computer Science, CALCO 2015, June 24-26, 2015, Nijmegen, The Netherlands*, LIPIcs **35** (2015), pp. 156–170.
URL https://doi.org/10.4230/LIPIcs.CALCO.2015.156

[11] Mac Lane, S., "Categories for the working mathematician," Number 5 in Graduate Texts in Math, Springer-Verlag, 1971.

[12] Milne, R., "The formal semantics of computer languages and their implementations," Oxford University Computing Laboratory, Programming Research Group, 1974.

[13] Plotkin, G., "Lambda-definability and logical relations," Edinburgh University, 1973.

[14] Plotkin, G. and M. Abadi, *A logic for parametric polymorphism*, in: *International Conference on Typed Lambda Calculi and Applications*, Springer, 1993, pp. 361–375.

[15] Plotkin, G. D., *Lambda-definability in the full type hierarchy*, To HB Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism (1980), pp. 363–373.