On Fixpoint Objects and Gluing Constructions

Roy L. Crole

email: rlc3@mcs.le.ac.uk
http://www.mcs.le.ac.uk/~rlc3

April 21, 1997

Abstract

This article[1] has two parts:

In the first part, we present some general results about fixpoint objects. The minimal categorical structure required to model soundly the equational type theory which combines higher order recursion and computation types (introduced by [4]) is shown to be precisely a let-category possessing a fixpoint object. Functional completeness for such categories is developed. We also prove that categories with fixpoint operators do not necessarily have a fixpoint object.

In the second part, we extend Freyd's gluing construction for cartesian closed categories to cartesian closed let-categories, and observe that this extension does not obviously apply to categories possessing fixpoint objects. We solve this problem by giving a new gluing construction for a limited class of categories with fixpoint objects; this is the main result of the paper. We use this category-theoretic construction to prove a type-theoretic conservative extension result.

A version of this paper is due to appear in the *Journal of Applied Categorical Structures.*

[1]AMS classification 18 C 10

1 Introduction

We begin by indicating what material readers should be acquainted with. We shall assume that the reader is fully conversant with basic category theory; see, for example, [13], [12], [1] or [2]. In particular, the Yoneda lemma will play a key role in the proof of our main result. This paper also assumes familiarity with the basic philosophy of categorical type theory, namely that there are precise mathematical correspondences between certain kinds of type theories and logics, and appropriate categorical structures. For an introduction to categorical type theory, see for example [12], [2] or [15]. For a deeper treatment of the topic, see [7]. It is well known that the equational theory corresponding to a cartesian closed category is a theory with a unit type, binary product types and function types. That this is well known stems in part from the fact that such a theory amounts to (a version of) the λ-calculus, which has played a prominent role in Computer Science in recent decades. However, many other such correspondences are proving to be useful in the Computer Science community; this paper concentrates on the correspondence between appropriate equational theories, and let-categories with fixpoint objects. Moggi introduced equational theories for let-categories in [14]. The equational theories corresponding to categories with fixpoint objects were introduced in [4]. We shall give a brief summary of the definitions of these categorical structures, and the corresponding equational type theories, in Section 2.

The aims of this paper are twofold. The first goal is to present some miscellaneous results about fixpoint objects. In particular, we are aiming to give some feel for the associated category theory/type theory correspondence. We show that the minimal requirements on a category for it to provide a sound model of the type theory of fixpoints are precisely that the category possess (a strong monad and) a fixpoint object. Continuing this theme, we show how to define polynomial categories over categories with fixpoint objects; our results provide sufficient machinery for the above correspondence to be presented á lá Lambek and Scott. We give an indication of the nature of the recursion which is captured by a fixpoint object by giving an example of a category where (certain kinds of) fixed points abound which can not be characterised using the universal properties of a fixpoint object.

The second goal is to use our categorical machinery to prove a conservative extension result for a type theory. The crux of the proof of this conservative extension appears in the proof of Theorem 6.7, which may be regarded as the most important result of this paper. Note that the techniques used in the proof of Theorem 6.7 are not directly connected with many of the earlier miscellaneous results—those that are used will be clearly indicated.

The layout of this article is as follows:

- In Sections 2 to 4, we give some general results about fixpoint objects. Section 2 provides some background material for the remainder of the paper. Sections 3 and 4 are independent of each other.

- In the second part, Sections 5 to 7, we build up some categorical machinery to prove a certain conservative extension result.

In section 2 we give a summary of the correspondence between let-theories and let-categories, and fix-theories and fix-categories; in particular we derive from first principles the minimal categorical structure required for the sound modelling of a fix-theory. In Section 3 we extend the basic notion of a polynomial category to the situation where the base category is a cartesian closed let-category with a fixpoint object. We prove that in such a set-up, the fixpoint object of the base category embeds to a fixpoint object for the polynomial category. In Section 4 we show that a category which has fixpoints of a certain class of endomorphisms, and for which we might expect the category to be equipped with a fixpoint object, does in fact not have one. In Section 5, we prove an abstract version of the usual gluing lemma for cartesian closed categories, but here for cartesian closed let-categories. In Section 6 we give a new gluing construction for a restricted class of FIX-categories.[2] Finally, in Section 7, we show how certain type theories generated from an algebraic theory give rise to freely generated categories. We use the categorical machinery developed in Sections 5 and 6 to prove type-theoretic conservative extension results.

2 Categorical Background

The material in this section, apart from Proposition 2.2, provides a summary of those concepts central to the remainder of the paper.

Notation and Conventions

We shall assume that all categories in this article are locally small: all of our constructions are closed under local smallness, and dealing with hom *sets* makes for a smooth presentation. We shall use the phrase "morphisms $A \to B$" to mean "morphisms with source A and target B." If $F: \mathcal{C} \to \mathcal{D}$ is a functor and $\mathcal{E}$ a category, we write $F^*: \mathcal{E}^{\mathcal{D}} \to \mathcal{E}^{\mathcal{C}}$ for the functor sending $G: \mathcal{D} \to \mathcal{E}$ to $G \circ F$, and define $F_*: \mathcal{C}^{\mathcal{E}} \to \mathcal{D}^{\mathcal{E}}$ similarly.

Let-Categories and Let-Theories

We recall [3], [4] that a *let-category* $(\mathcal{C}, T, \eta, \mathit{lift})$ is simply a category $\mathcal{C}$ with finite products which is equipped with an indexed Kleisli triple, that is to say:

- For each object A in $\mathcal{C}$, there is an object TA,
- for each object A in $\mathcal{C}$, there is a morphism $\eta_A: A \to TA$, and

[2] See page 21.

$$\frac{g \circ (f \times id_B) : A \times B \longrightarrow A' \times B \longrightarrow TC}{\mathit{lift}(g) \circ (f \times id_{TB}) : A \times TB \longrightarrow A' \times TB \longrightarrow TC} \quad \texttt{LiftS}$$

$$\begin{tikzcd}
& A \times TB \arrow[dr, "\mathit{lift}(f)"] & \\
A \times B \arrow[ur, "id \times \eta_B"] \arrow[rr, "f"'] & & TC
\end{tikzcd} \quad \texttt{LiftB}$$

$$\frac{\eta_B \circ \pi_2 : A \times B \longrightarrow B \longrightarrow TB}{\pi_2 : A \times TB \longrightarrow TB} \quad \texttt{LiftH}$$

$$\frac{\mathit{lift}(g) \circ \langle \pi_1, f \rangle : A \times B \longrightarrow A \times TC \longrightarrow TD}{\mathit{lift}(g) \circ \langle \pi_1', \mathit{lift}(f) \rangle : A \times TB \longrightarrow A \times TC \longrightarrow TD} \quad \texttt{LiftA}$$

Figure 1: The Equations of a Let-Category

- for each morphism $f: A \times B \to TC$, there is a morphism $\mathit{lift}(f): A \times TB \to TC$,

which satisfy certain equations which are given in Figure 1; we shall make use of the notation

$$\frac{f: A \times B \longrightarrow TC}{g: A \times TB \longrightarrow TC}$$

to indicate that $g = \mathit{lift}(f)$.

Recall also that let-categories are simply categories with finite products which are equipped with strong monads—for the latter see [9], [14] and [3]. This is made precise in the following lemma:

LEMMA 2.1 If $f: B \to TC$ in $\mathcal{C}$ then we write

$$\mathit{let}(f) \stackrel{\text{def}}{=} \mathit{lift}(f \circ \pi_B) \circ \pi_{TB}^{-1}: TB \to TC$$

where $\pi_B: 1 \times B \to B$ and $\pi_{TB}: 1 \times TB \to TB$. Given a let-category $\mathcal{C}$, the corresponding strong monad (T, η, μ, τ) is given by a functor $T: \mathcal{C} \to \mathcal{C}$ sending $f: A \to B$ to $\mathit{let}(\eta_B \circ f): TA \to TB$, a unit η whose components are the η_A, a multiplication μ with components $\mu_A \stackrel{\text{def}}{=} \mathit{let}(id_{TA})$ and a strength

$$\tau : id_{\mathcal{C}} \times T \longrightarrow (T \circ \times) : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$$

with components $\tau_{(A,B)} \stackrel{\text{def}}{=} \mathit{lift}(\eta_{A \times B})$. Conversely, given a category $\mathcal{C}$ with finite products and strong monad, the let-category structure is as expected, with

$$\mathit{lift}(f) \stackrel{\text{def}}{=} \mu_C \circ Tf \circ \tau_{(A,B)}$$

if $f: A \times B \to TC$ in $\mathcal{C}$.

PROOF Routine calculations. $\square$

We shall now briefly recall the equational type theory corresponding to let-categories. Judgements will take the form

$$[x_1: \alpha_1, \ldots, x_n: \alpha_n] \vdash M: \alpha$$

where each α_i and α are types, $[x_1: \alpha_1, \ldots, x_n: \alpha_n]$ is an environment (which for us is a *list* of (variable,type) pairs) and M is a (raw) term. We will usually denote environments by Γ. Such a judgement asserts that in the environment Γ the term M is assigned the type α. We call these judgements *typing assertions*. Judgements will also take the form

$$\Gamma \vdash M = M': \alpha$$

and these assert a formal equality between the two terms M and M'.

A *theory* [2] page 127, *Th*, consists (as usual) of a collection of rules for deriving typing assertions and equations between terms. A *let-theory* contains typing assertions of the form

$$\frac{\Gamma \vdash M: \alpha}{\Gamma \vdash \mathsf{Val}(M): T\alpha} \qquad \frac{\Gamma \vdash E: T\alpha \qquad \Gamma, x: \alpha \vdash F(x): T\beta}{\Gamma \vdash \mathsf{Let}\,(E, F): T\beta}$$

and the let-theory equations are

$$\begin{aligned} \Gamma \vdash \mathsf{Let}\,(\mathsf{Val}(M), F) &= F(M) && (1)\\ \Gamma \vdash \mathsf{Let}\,(E, u.\mathsf{Val}(u)) &= E && (2)\\ \Gamma \vdash \mathsf{Let}\,(\mathsf{Let}\,(E, F), G) &= \mathsf{Let}\,(E, u.\mathsf{Let}\,(F(u), G)) && (3) \end{aligned}$$

As usual, we write $[\![-]\!]: Th \to \mathcal{C}$ for the operation giving a structure in a category $\mathcal{C}$ to a type theory *Th*. We can interpret a let-theory in a let-category by defining

$$\begin{aligned} [\![T\alpha]\!] &\stackrel{\text{def}}{=} T([\![\alpha]\!])\\ [\![\Gamma \vdash \mathsf{Val}(M): T\alpha]\!] &\stackrel{\text{def}}{=} \eta_{[\![\alpha]\!]}([\![\Gamma \vdash M: \alpha]\!])\\ [\![\Gamma \vdash \mathsf{Let}\,(E, F): T\beta]\!] &\stackrel{\text{def}}{=} \mathit{lift}([\![\Gamma, x: \alpha \vdash F(x): T\beta]\!])\langle id, [\![\Gamma \vdash E: T\alpha]\!]\rangle \end{aligned}$$

and the let-theory equations are modelled by the above structure in a let-category.

Fix-Categories and Fix-Theories

We recall the notion of a fixpoint object in a suitably structured category. Given a category $\mathcal{C}$ with finite products, the $\mathcal{C}$-indexed category $\mathcal{C}^{(-)}$ is specified by:

- The objects of $\mathcal{C}^C$ are those of $\mathcal{C}$,

- $\mathcal{C}^C(A,B) \stackrel{\text{def}}{=} \mathcal{C}(C \times A, B)$ where composition of g and h in $\mathcal{C}^C$ is $h\langle \pi, g\rangle$ in $\mathcal{C}$,
- $\mathcal{C}^k(A) \stackrel{\text{def}}{=} A$ and $\mathcal{C}^k(g) \stackrel{\text{def}}{=} g(k \times id)$,

where $k: C' \to C$ in $\mathcal{C}$ and $A \xrightarrow{g} B \xrightarrow{h} D$ in $\mathcal{C}^C$. Note that given a let-category $\mathcal{C}$ there is an indexed endofunctor $T^{(-)}$ on $\mathcal{C}^{(-)}$ where $T^C(A) \stackrel{\text{def}}{=} TA$ and $T^C(g) \stackrel{\text{def}}{=} \mathit{lift}(\eta_B g)$. With this we make the following definition: in a let-category $\mathcal{C}$, a *fixpoint object* (FPO) is specified by the following data

- An initial $T^{(-)}$ algebra whose structure map in the fibre $\mathcal{C}^C$ is the morphism $C \times T\Omega \xrightarrow{\pi} T\Omega \xrightarrow{\sigma} \Omega$ in $\mathcal{C}$. Thus given a morphism $f: TA \to A$ in $\mathcal{C}^C$, there is a unique morphism $it(f): \Omega \to A$ in $\mathcal{C}^C$ for which the following diagram commutes (in $\mathcal{C}$):

$$\begin{tikzcd}
C \times T\Omega \arrow[r, "id \times \sigma"] \arrow[d, "\langle \pi_C, \mathit{lift}(\eta_A \circ it(f))\rangle"'] & C \times \Omega \arrow[d, "it(f)"] \\
C \times TA \arrow[r, "f"'] & A
\end{tikzcd}$$

- A global element $\omega: 1 \to T\Omega$ which gives rise to an equaliser diagram of the form

$$1 \xrightarrow{\omega} T\Omega \underset{id}{\overset{\eta_\Omega \sigma}{\rightrightarrows}} T\Omega.$$

We shall call a let-category equipped with a FPO a *fix-category*.

A *fix-theory* [4] consists of typing assertions of the form

$$\frac{}{\Gamma \vdash \omega: T\mathit{fix}} \qquad \frac{\Gamma \vdash E: T\mathit{fix}}{\Gamma \vdash \sigma(E): \mathit{fix}} \qquad \frac{\Gamma, x: T\alpha \vdash F(x): \alpha \quad \Gamma \vdash N: \mathit{fix}}{\Gamma \vdash \mathsf{It}_\alpha(F, N): \alpha}$$

and equations of the form

$$\begin{aligned}
\Gamma \vdash \omega &= \mathsf{Val}(\sigma(\omega)) && (4)\\
\Gamma \vdash E = \mathsf{Val}(\sigma(E)) &\supset \Gamma \vdash E = \omega && (5)\\
\Gamma, e \vdash G(\sigma(e)) = F(\mathsf{Let}\,(e, n.\mathsf{Val}(G(n)))) &\supset \Gamma \vdash G(N) = \mathsf{It}_\alpha(F, N) && (6)\\
\Gamma \vdash \mathsf{It}_\alpha(F, \sigma(E)) &= F(\mathsf{Let}\,(E, n.\mathsf{Val}(\mathsf{It}_\alpha(F, n)))) && (7)
\end{aligned}$$

It is instructive to show that, subject to the usual paradigm of categorical type theory that substitution is interpreted by composition and rule formation by natural transformations [2], the minimal structural requirements of a category $\mathcal{C}$ for the sound modelling of a fix-theory are that $\mathcal{C}$ be a let-category with FPO. This is our first proposition:

PROPOSITION 2.2 In order to model a fix-theory in a category $\mathcal{C}$, the minimal requirements of $\mathcal{C}$ are that it is a let-category which is endowed with a FPO.

PROOF We shall need an object Ω of $\mathcal{C}$ to model *fix*, that is $[\![\mathit{fix}]\!] \stackrel{\text{def}}{=} \Omega$. To interpret the first typing assertion, we shall need a global element of $T\Omega$. So we have

$$[\![\Gamma \vdash \omega : T\mathit{fix}]\!] \stackrel{\text{def}}{=} \omega \circ ! : [\![\Gamma]\!] \to 1 \to T\Omega,$$

where $[\![\Gamma]\!]$ is the usual finite product of the interpretations of the types in Γ. To interpret the second typing assertion, with the usual assumptions about the way we shall model substitution, we shall need a natural transformation

$$\theta : \mathcal{C}(-, T\Omega) \to \mathcal{C}(-, \Omega).$$

Using the Yoneda Lemma, we can deduce that the effect of the components θ_C arise by post composition with a morphism $\sigma : T\Omega \to \Omega$. Thus we can set

$$[\![\Gamma \vdash \sigma(E) : \mathit{fix}]\!] \stackrel{\text{def}}{=} \theta_{[\![\Gamma]\!]}(e) = \sigma \circ e,$$

where, by structural induction we already have $[\![\Gamma \vdash E : T\mathit{fix}]\!] = e$. To interpret the third typing assertion, we shall need a natural transformation with components

$$\theta_C : \mathcal{C}(C \times TA, A) \times \mathcal{C}(C, \Omega) \longrightarrow \mathcal{C}(C, A).$$

Applying naturality to $\langle id, n \rangle : C \to C \times \Omega$, we get

$$\theta_C(f, n) = \theta_{C \times \Omega}(f \circ (\pi_1 \times id), \pi_2) \circ \langle id, n \rangle$$

where $\pi_1 : C \times \Omega \to C$ and $\pi_2 : C \times \Omega \to \Omega$. Thus it is enough to model the third typing assertion if we have a natural transformation which has components

$$\phi_C : \mathcal{C}(C \times TA, A) \longrightarrow \mathcal{C}(C \times \Omega, A).$$

Writing $it(f) \stackrel{\text{def}}{=} \phi_C(f)$, we may define

$$[\![\Gamma \vdash \mathsf{It}(F, N) : \alpha]\!] \stackrel{\text{def}}{=} it(f) \circ \langle id, n \rangle,$$

where we already have

$$\begin{aligned} [\![\Gamma, x : T\alpha \vdash F(x) : \alpha]\!] &= f \\ [\![\Gamma \vdash N : \mathit{fix}]\!] &= n. \end{aligned}$$

Now we consider the equations of a fix-theory. It is easy to see that for the sound modelling of (4) and (5) it is necessary and sufficient that the triple (ω, σ, Ω) satisfies the equaliser requirement of a FPO.

Note that using the usual logical rules of equational type theory, rule (6) is equivalent to

$$\Gamma, e \vdash G(\sigma(e)) = F(\mathsf{Let}\,(e, n.\mathsf{Val}(G(n)))) \quad \supset \quad \Gamma, n \vdash G(n) = \mathsf{It}_\alpha(F, n) \tag{8}$$

In order to model (8) we see it is necessary that[3]

$$\frac{g(id \times \sigma) = f\langle \pi_1, \mathit{lift}(\eta g)\rangle}{g = it(f)}\ (*)$$

where $f: C \times TA \to A$, $g: C \times \Omega \to A$ and $\pi_1: C \times T\Omega \to C$; sufficiency of the uniqueness requirement of a FPO is immediate.

Finally, we look at the structure needed to model (7). Put

$$\pi_2: C \times T\Omega \to T\Omega \qquad \pi_3: C \times \Omega \to C \qquad \pi_4: C \times \Omega \to \Omega.$$

Working through the details we shall need

$$it(f)(id \times \sigma)\langle id, e\rangle \quad = \quad f\langle id, \mathit{lift}(it(\eta(f(\pi_3 \times id)))\langle id, \pi_4\rangle)\langle id, e\rangle\rangle \tag{9}$$

where $e: C \to T\Omega$. However, we can appeal to $(*)$ to deduce that $it((f(\pi_3 \times id))) = it(f)(\pi_3 \times id)$; thus (9) reduces to

$$it(f)(id \times \sigma)\langle id, e\rangle \quad = \quad f\langle id, \mathit{lift}(\eta \circ it(f))\langle id, e\rangle\rangle. \tag{10}$$

The universal property of a FPO is certainly sufficient to ensure that (10) holds. To see that it is also necessary, note that

$$\begin{array}{rcl} f\langle \pi_1, \mathit{lift}(\eta \circ it(f))\rangle & = & f(\pi_1 \times id)\langle id, \mathit{lift}(\eta \circ it(f)(\pi_1 \times id))\langle id, \pi_2\rangle\rangle \\ & = & f(\pi_1 \times id)\langle id, \mathit{lift}(\eta \circ it((f(\pi_1 \times id))))\langle id, \pi_2\rangle\rangle \\ \text{using an instance of } (10) & = & it((f(\pi_1 \times id)))(id \times \sigma)\langle id, \pi_2\rangle \\ & = & it(f)(\pi_1 \times id)(id \times \sigma)\langle id, \pi_2\rangle \\ & = & it(f)(id \times \sigma). \end{array}$$

Finally note that the universal property of a FPO ensures Ω and σ are determined up to isomorphism. □

A few remarks are in order. The basic definition of FPO requires the structure of a let-category. However, an equivalent, but much simpler definition, can be given in the presence of cartesian closure. For convenience, from now on we shall always deal with let-categories $\mathcal{C}$ which are also cartesian closed; such a category will be referred to as a *let-ccc*. Recall [4] that a fixed point object (FPO) (Ω, σ, ω) in a let-ccc amounts to specifying an initial T-algebra for the *functor* T, say $\sigma: T\Omega \to \Omega$, together with a global element ω of $T\Omega$ which is the equaliser of $\eta_\Omega \circ \sigma$ and $id_{T\Omega}$. (See [4] for the proof that this definition is equivalent to that given on page 6.)

Let us look at an example, which will play a key role in Section 6. It is well known that $\omega\mathcal{CPO}$, the category of ωcpos (not necessarily with bottom) and Scott continuous functions, is a let-ccc $(\omega\mathcal{CPO}, (-)_\perp, \iota, \mathit{lift}(-))$ where

[3] In the remainder of this paper, we omit most composition symbols $\circ$, writing uv for $u \circ v$, and often omit sub-scripts from natural transformations (e.g. $\eta: A \to TA$).

- $D_\perp \stackrel{\mathrm{def}}{=} \{[d] \mid d \in D\} \cup \{\perp\}$ for an ωcpo D is the lifted ωcpo,
- $\iota_D\colon D \to D_\perp$ is defined by $d \mapsto [d]$, and
- if $f\colon D \times D' \to D''_\perp$ is Scott continuous, then $\mathit{lift}(f)\colon D \times D'_\perp \to D''_\perp$ is defined by $(d, \perp) \mapsto \perp$ and $(d, d') \mapsto f(d, d')$. We shall write $f_\perp$ for $\mathit{let}(f)$ in $\omega\mathcal{CPO}$.

Moreover, $\omega\mathcal{CPO}$ has a FPO $(\sigma, \mathbb{N}^\infty, [\infty])$ where $\mathbb{N}^\infty$ is the vertical natural numbers with top element ∞, and $\sigma\colon \mathbb{N}^\infty_\perp \to \mathbb{N}^\infty$ is the function defined by $\sigma(\perp) = 0$, $\sigma([n]) = n + 1$ and $\sigma([\infty]) = \infty$.

Freely Generated Categories

In Section 7 we shall make use of freely generated categories; here we clarify our definition of this concept. Suppose that *struc* refers to a categorical structure (for example cartesian closure, possessing a strong monad and so on). Then a *morphism* of struc-categories is a functor which preserves the structure. If the structure is a property of the category, this notion is standard. We clarify our definition as applied to strong monads (let structures) which are, of course, not categorical properties but chosen structures. A *morphism* of let-categories $(\mathcal{C}, T) \to (\mathcal{D}, S)$ is a monad morphism (F, i) between the underlying monads for which the functor $F\colon \mathcal{C} \to \mathcal{D}$ preserves finite products, and each $i\colon F(TC) \cong S(FC)$ is a natural *iso*morphism. Thus to be given a let-category morphism $(F, i)\colon (\mathcal{C}, T) \to (\mathcal{D}, S)$ means for each object A in $\mathcal{C}$ we are given an isomorphism $i_A\colon FTA \cong SFA$ which is compatible with the remaining structure. For example, suppose we are given a morphism $f\colon A \times B \to TC$ in $\mathcal{C}$. Then there is a morphism

$$g\colon FA \times FB \cong F(A \times B) \xrightarrow{Ff} FTC \xrightarrow{i_C} SFC$$

and a morphism

$$h\colon FA \times FTB \cong F(A \times TB) \xrightarrow{F(\mathit{lift}(f))} FTC$$

for which, by definition, the following diagram commutes:

$$\begin{tikzcd}
FA \times SFB \arrow[r, "\mathit{lift}(g)"] & SFC \\
FA \times FTB \arrow[u, "id \times i_B"] \arrow[r, "h"'] & FTC \arrow[u, "i_C"']
\end{tikzcd}$$

The (categorical) isomorphisms are, of course, the canonical ones.

Let $\mathcal{C}$ be a category with finite products. Then $\mathcal{FC}$ is the *freely generated* struc-category on $\mathcal{C}$ if there is a functor $I\colon \mathcal{C} \to \mathcal{FC}$ such that

(i) I preserves finite products,

(ii) if $\mathcal{D}$ is any struc-category, and $F: \mathcal{C} \to \mathcal{D}$ preserves finite products, then there is a morphism $\overline{F}: \mathcal{FC} \to \mathcal{D}$ of struc-categories for which there is a natural isomorphism $\phi: \overline{F} \circ I \cong F$, and

(iii) if also $\phi': \overline{\overline{F}} \circ I \cong F$ where $\overline{\overline{F}}: \mathcal{FC} \to \mathcal{D}$ also satisfy (ii), then there is a natural isomorphism $\psi: \overline{F} \cong \overline{\overline{F}}$ for which $\psi_I = \phi'^{-1} \circ \phi$.

3 Functional Completeness

In this section we shall prove that the polynomial category of a let-ccc is functionally complete. We refer to page 59 of [12] for a similar discussion concerning cartesian closed categories. We also show that if a let-ccc has a FPO, this embeds to a FPO in the associated polynomial category. These results are presented for their own inherent interest, and because the category theory/type theory correspondence can be presented using the notion of polynomial categories; in particular, our results show that the approach of Lambek and Scott [12] (page 77 onwards) to such a correspondence applies in our setting.

We begin with the definition of let-ccc polynomial categories. Note that the definition of these categories is simple, but should be treated with care[4] and so we spell out the details. Let $\mathcal{C}$ be a let-ccc and let E be an object of $\mathcal{C}$. The *polynomial let-ccc* $\mathcal{C}[X]$ *in variable* X *for* E is specified by the following data:

- The objects of $\mathcal{C}[X]$ are just the objects of $\mathcal{C}$.

- We set $mor\ \mathcal{C}[X] \stackrel{\text{def}}{=} \langle PM \rangle / \sim$ where

  (i) $PM \stackrel{\text{def}}{=} mor\,\mathcal{C} \cup \{X\}$,
  (ii) $\langle PM \rangle$ is generated from PM via the (formal) rules

$$\frac{f: A \to B \in PM}{f: A \to B \in \langle PM \rangle} \qquad \frac{u: A \to B \in \langle PM \rangle \quad v: B \to C \in \langle PM \rangle}{v \circ u: A \to C \in \langle PM \rangle}$$

$$\frac{u: A \to B \in \langle PM \rangle \quad v: A \to C \in \langle PM \rangle}{\langle u, v \rangle: A \to B \times C \in \langle PM \rangle} \qquad \frac{u: A \times B \to C \in \langle PM \rangle}{\lambda(u): A \to (B \Rightarrow C) \in \langle PM \rangle}$$

$$\frac{u: A \times B \to TC \in \langle PM \rangle}{\mathit{lift}(u): A \times TB \to TC \in \langle PM \rangle}$$

  (iii) $\sim$ is the equivalence relation generated by
  1. If $f = g$ in $\mathcal{C}$, then $f \sim g$.

[4] Thanks to Martin Hyland for making me realise just how careful one needs to be!

2. $u \circ id \sim u \sim id \circ u$.
3. If $u \sim u'$ and $v \sim v'$, then $v \circ u \sim v' \circ u'$.
4. $(w \circ v) \circ u \sim w \circ (v \circ u)$.
5. Conditions forcing (formal) cartesian closure of $\mathcal{C}[X]$; see [12], page 57.
6. If $u \sim u'$ then $\mathit{lift}(u) \sim \mathit{lift}(u')$.
7. Conditions forcing $\mathcal{C}[X]$ to be a let-category; for example, this amounts to requiring
$$\mathit{lift}(v \circ (u \times id)) \sim \mathit{lift}(v) \circ (u \times id),$$
corresponding to `LiftS` (see page 4); similar conditions hold for the other let structure equations. Here, of course, $u \times id \stackrel{\text{def}}{=} \langle u \circ \pi, \pi' \rangle$.

- If $u \in \langle PM \rangle$ we write $[u]$ for the equivalence class of u in $mor\, \mathcal{C}[X]$. The source of[5] $[X]$ is defined to be 1 and the target E; then the source-target predicate in $\mathcal{C}[X]$ is the expected one. The identity on A is $[id_A]$, and composition in $\mathcal{C}[X]$ is given by $[u] \cdot [v] \stackrel{\text{def}}{=} [u \circ v]$. Note that (1) to (4) ensure that $\mathcal{C}[X]$ is indeed a category and that there is an obvious functorial embedding $\iota : \mathcal{C} \hookrightarrow \mathcal{C}[X]$.

Proposition 3.1 For the above data, $\mathcal{C}[X]$ is a let-ccc with projections $[\pi] : A \times B \to A$, pairing $\langle [u], [v] \rangle \stackrel{\text{def}}{=} [\langle u, v \rangle]$, and the let-structure given by $A \mapsto TA$ on objects, units $[\eta_A] : A \to TA$, and $\mathit{lift}([u]) \stackrel{\text{def}}{=} [\mathit{lift}(u)]$ for a morphism $[u] : A \times B \to TC$ in $\mathcal{C}[X]$.

Proof The cc-stucture is standard [12]. We check (for example) that `LiftS` holds:
$$\begin{aligned} \mathit{lift}([u]) \cdot ([v] \times id_{TB}) &= [\mathit{lift}(u)] \cdot \langle [v] \cdot [\pi], [\pi'] \rangle \\ &= [\mathit{lift}(u)] \cdot [\langle v \circ \pi, \pi' \rangle] \\ &= \mathit{lift}([u] \cdot ([v] \times id_B)). \end{aligned}$$
The other details are routine computations. □

Proposition 3.2 The polynomial category is characterized by the following universal property: Suppose that $\iota : \mathcal{C} \hookrightarrow \mathcal{C}[X]$ is the embedding described above. Let $\mathcal{D}$ be a let-ccc, $F : \mathcal{C} \to \mathcal{D}$ a morphism of let-cccs and $d : F1 \to FE$ any morphism in $\mathcal{D}$. Then there is a unique morphism of let-cccs $\overline{F} : \mathcal{C}[X] \to \mathcal{D}$ such that

$$\begin{tikzcd} \mathcal{C} \arrow[r, "\iota"] \arrow[d, "F"'] & \mathcal{C}[X] \arrow[dl, "\overline{F}"] \\ \mathcal{D} & \end{tikzcd}$$

commutes with $\overline{F}([X]) = d$.

[5]Note that $\mathcal{C}[X]$ denotes the polynomial category, but that $[X]$ denotes the equivalence class of the symbol X in $\langle PM \rangle / \sim$.

PROOF The proof details are essentially routine, where $\overline{F}$ is F on the objects of $\mathcal{C}[X]$, and defined recursively on the morphisms of $\mathcal{C}[X]$ through the rules specifying $mor\,\mathcal{C}[X]$. $\overline{F}([X]) \stackrel{\text{def}}{=} d$ is of course a base clause for the recursive definition. □

REMARK 3.3 We shall write just u rather than $[u]$ for morphisms in the polynomial category, and write uv for composition rather than $[u]\cdot[v]$.

Now we prove a version of functional completeness for let-cccs:

THEOREM 3.4 Let $\mathcal{C}$ be a let-ccc and $\mathcal{C}[X]$ its polynomial category. For every morphism $u\colon B \to C$ in $\mathcal{C}[X]$ there is a unique morphism

$$\Lambda(u)\colon E \times B \to C$$

in $\mathcal{C}$ such that

$$\Lambda(u)\langle X!, id\rangle = u$$

where $!\colon B \to 1$. In particular, each global element of C in $\mathcal{C}[X]$ is of the form fX for some unique morphism $f\colon E \to C$ in $\mathcal{C}$.

PROOF As there is a proof of this result for cccs in [12], page 59, we just give details of the proof for the let structure. Take morphisms

$$\begin{array}{rrcl} & u & : & B \times B' \to TC, \\ \text{and hence} & \mathit{lift}(u) & : & B \times TB' \to TC, \\ & l & : & (E \times B) \times B' \to E \times (B \times B'), \\ & m & : & E \times (B \times TB') \to (E \times B) \times TB', \end{array}$$

where l and m are the obvious isomorphisms. By induction, we shall assume that there is a morphism

$$\Lambda(u)\colon E \times (B \times B') \to TC$$

that satisfies functional completeness. Then we define

$$\Lambda(\mathit{lift}(u)) \stackrel{\text{def}}{=} \mathit{lift}(\Lambda(u)l)m,$$

and refer the reader to [12], page 51, for the definition of Λ on the ccc structure. We have to check that this is well defined, i.e. if $u \sim v$ in $\mathcal{C}[X]$ then $\Lambda(u) \sim \Lambda(v)$ in $\mathcal{C}[X]$. This amounts to showing that Λ respects each of conditions (1) to (7) of Definition 3; most of these are easy to verify, but we check in detail one of the let-structure conditions of (7), namely $\mathit{lift}(v\langle u\pi, \pi'\rangle) \sim \mathit{lift}(v)\langle u\pi, \pi'\rangle$. We have

$$\begin{array}{rcl} \Lambda(\mathit{lift}(v\langle u\pi, \pi'\rangle)) & \sim & \mathit{lift}(\Lambda(v\langle u\pi, \pi'\rangle)l)m \\ \text{by definition is} & \sim & \mathit{lift}(\Lambda(v)\langle \pi, \langle \Lambda(u)\langle \pi, \pi\pi'\rangle, \pi'\pi'\rangle\rangle\langle \pi\pi, \langle \pi'\pi, \pi'\rangle\rangle)m \\ \text{which is} & \sim & \mathit{lift}(\Lambda(v)\langle \pi\pi, \langle \Lambda(u)\pi, \pi'\rangle\rangle)m \\ \text{for appropriate } l' & \sim & \mathit{lift}(\Lambda(v)l'(\langle \pi, \Lambda(u)\rangle \times id))m \\ \text{from the let structure} & \sim & \mathit{lift}(\Lambda(v)l')(\langle \pi, \Lambda(u)\rangle \times id)m \\ & \sim & \mathit{lift}(\Lambda(v)l')m'\langle \pi, \langle \Lambda(u)\langle \pi, \pi\pi'\rangle, \pi'\pi'\rangle\rangle \\ & \sim & \Lambda(\mathit{lift}(v))\langle \pi, \Lambda(\langle u\pi, \pi'\rangle)\rangle \\ & \sim & \Lambda(\mathit{lift}(v)\langle u\pi, \pi'\rangle) \end{array}$$

which is what we had to show. Now that we know Λ is well defined we prove the result itself. Indeed,

$$\begin{array}{rcl}
\Lambda(\mathit{lift}(u))\langle X!, id\rangle & = & \mathit{lift}(\Lambda(u)l)\circ m\circ\langle X!, id\rangle \\
 & = & \mathit{lift}(\Lambda(u)l)\circ(\langle X, id\rangle\times id) \\
\text{by }\texttt{LiftS}\text{ for the let-ccc }\mathcal{C}[X] & = & \mathit{lift}((\Lambda(u)l)\circ(\langle X!, id\rangle\times id)) \\
 & = & \mathit{lift}(\Lambda(u)\circ\langle X!, id\rangle) \\
\text{and by induction} & = & \mathit{lift}(u).
\end{array}$$

It remains to prove uniqueness. Suppose that there is $g\colon E\times B\to C$ for which $g\langle X!, id_B\rangle = u$. We have to show that $g=\Lambda(u)$. This follows from the following computation:

$$\begin{array}{rcl}
\Lambda(u) & = & \Lambda(g\langle X!, id\rangle) \\
 & = & \Lambda(g)\langle\pi, \Lambda(\langle X!, id\rangle)\rangle \\
 & = & g\pi'\langle\pi, \Lambda(\langle X!, id\rangle)\rangle \\
 & = & g\Lambda(\langle X!, id\rangle) \\
 & = & g\langle\Lambda(X!), \Lambda(id)\rangle \\
 & = & g\langle\pi, \pi'\rangle \\
 & = & g\circ id_{E\times B} \\
 & = & g.
\end{array}$$

(Again we emphasise that the definition of Λ on the ccc structure of $\mathcal{C}[X]$ can be found on page 51 of [12].) $\square$

PROPOSITION 3.5 Let $\mathcal{C}$ be a let-ccc with FPO Ω. Then $\mathcal{C}[X]$ also has a FPO, given by $\Omega = i(\Omega)$, where $i:\mathcal{C}\hookrightarrow\mathcal{C}[X]$ is the canonical inclusion functor.

PROOF Consider the diagram in $\mathcal{C}[X]$ given in Figure 2, where $\sigma\colon T\Omega\to\Omega$ is the embedding of the FPO of $\mathcal{C}$ in $\mathcal{C}[X]$, f is any morphism $TB\to B$ in $\mathcal{C}[X]$, and where $k\stackrel{\text{def}}{=}\langle\pi_E, \mathit{lift}(\eta\circ it(\Lambda(f)))\rangle$.

We set $h\stackrel{\text{def}}{=} it(\Lambda(f))\langle X!, id\rangle$, where the morphism $\Lambda(f)$ is defined by appealing to Theorem 3.4. In order to prove that $h\sigma = flet(\eta h)$ we inspect the diagram above. It is clear all regions commute except for $k\langle X!, id\rangle = \langle X!, id\rangle let(\eta h)$. To prove this it is enough to show that $\mathit{lift}(\eta\circ it(\Lambda(f)))\langle X!, id\rangle = let(\eta\circ it(\Lambda(f))\langle X!, id\rangle)$. We appeal to Theorem 3.4 and note

$$\begin{array}{rcl}
\Lambda(let(\eta\circ it(\Lambda(f))\langle X!, id\rangle)) & = & \mathit{lift}(\Lambda(\eta\circ it(\Lambda(f)))\langle\pi, \Lambda(\langle X!, id\rangle)\rangle) \\
 & = & \mathit{lift}(\eta\pi'\langle\pi, it(\Lambda(f))\pi'\rangle\langle\pi, \langle\pi, \pi'\rangle\rangle) \\
 & = & \mathit{lift}(\eta\circ it(\Lambda(f))\langle\pi, \pi'\rangle).
\end{array}$$

It remains to prove uniqueness. Suppose also that $g\sigma = flet(\eta g)$. Note that if $\Lambda(g) = it(\Lambda(f))$ then $h = it(\Lambda(f))\langle X!, id\rangle = \Lambda(g)\langle X!, id\rangle = g$; so it is sufficient to

$$
\begin{tikzcd}
T\Omega \arrow[rrr, "\sigma"] \arrow[dr, "\langle X!, id\rangle"] \arrow[ddd, dotted, "let(\eta h)"'] & & & \Omega \arrow[dl, "\langle X!, id\rangle"'] \arrow[dd, dotted, "h"] \\
& E \times T\Omega \arrow[r, "id \times \sigma"] \arrow[d, "k"] & E \times \Omega \arrow[d, "it(\Lambda(f))"] & \\
& E \times TB \arrow[r, "\Lambda(f)"] & B \arrow[r, shift left] \arrow[d, shift right, "="'] & B \arrow[l, shift left, "="] \\
TB \arrow[ur, "\langle X!, id\rangle"] \arrow[rr, "f"'] & & B \arrow[u, shift right] &
\end{tikzcd}
$$

Figure 2: A Diagram Used in the Proof of Proposition 3.5

prove the former. For this, if we can show $\Lambda(g)(id \times \sigma) = \Lambda(f)\langle \pi_E, lift(\eta\Lambda(g))\rangle$ we are done using the universal property of the FPO Ω in $\mathcal{C}$. As $\Lambda(g\sigma) = \Lambda(g)(id \times \sigma)$ it is sufficient to prove $\Lambda(f)\langle \pi_E, lift(\eta\Lambda(g))\rangle\langle X!, id\rangle = g\sigma$. We have

$$
\begin{aligned}
\Lambda(f)\langle \pi_E, lift(\eta\Lambda(g))\rangle\langle X!, id\rangle &= \Lambda(f)\langle X!, lift(\eta\Lambda(g))\langle X!, id\rangle\rangle \\
\text{using uniqueness of } \Lambda(-) &= \Lambda(f)\langle X!, let(\eta g)\rangle \\
&= f\,let(\eta g)
\end{aligned}
$$

and so we are done. Finally, the equaliser condition is simple to verify. □

4 A Category with Fixpoints, without a Fixpoint Object

A *domain* [16] is usually defined to be a poset with some form of cocompleteness property, that is, joins of certain kinds are required to exist. Categories of domains arise naturally as suitable arenas in which to give semantics to programming languages, but that does not concern us here [6]. One reason for this is that the existence of fixed point operators in such categories can be used to give semantics to recursive program declarations. Turning to more theoretical issues, we might ask if it is always the case that a concrete category of domains has a FPO whenever the morphisms between lifted domains may be guaranteed to have fixpoints. This is not so, and is the central result of this section. We present it because it shows clearly that the properties captured by abstract FPOs are quite closely tied to the usual concrete models of recursion.

Let $\mathcal{O}n\mathcal{CPO}$ be the category which has

- objects posets possessing suprema of *all* chains and
- morphisms monotone set functions.

It is easy to see that this category is indeed a let-ccc when we regard the lifting functor as giving rise to a strong monad. It is the case that all morphisms $f: D_\perp \to D_\perp$ have a least fixpoint. For given any such morphism f, define for any ordinal α the element $f^\alpha(\perp)$ by

$$\begin{array}{rcl} f^0(\perp) & = & \perp \\ f^{\alpha+1}(\perp) & = & f(f^\alpha(\perp)) \\ f^\lambda(\perp) & = & \bigvee_{\alpha<\lambda} f^\alpha(\perp) \quad \text{where } \lambda \text{ is a limit ordinal.} \end{array}$$

This is a good definition. The only thing that is not trivial is the existence of $f^\lambda(\perp)$: the set $\{f^\alpha(\perp) \mid \alpha < \lambda\}$ must be a chain. But a simple transfinite induction shows that $\alpha \leq \alpha' < \lambda$ implies $f^\alpha(\perp) \leq f^{\alpha'}(\perp)$. Next note that there is some ordinal β for which $f^\beta(\perp)$ is a least fixpoint of f. Suppose this were not the case. By appealing to Hartog's Lemma, we can find a least ordinal (say γ) whose cardinality is strictly greater than the cardinality of the domain of f, say $|D_\perp|$. Using the hypothesis, it must be the case that the cardinality of $\{f^\alpha(\perp) \mid \alpha \leq \gamma\}$ is strictly greater than $|D_\perp|$, a contradiction.

THEOREM 4.1 The let-ccc $\mathcal{O}n\mathcal{CPO}$ does not possess a fixpoint object for the lifting monad.

PROOF Let us suppose that the fixpoint object Ω exists in $\mathcal{O}n\mathcal{CPO}$. We write $i: \Omega \hookrightarrow \Omega_\perp,\ \ n \mapsto [n]$ for the (monic) component of the unit of the lifting monad at Ω. From the equaliser condition of the fixpoint object, $i\sigma: \Omega_\perp \to \Omega_\perp$ has a unique fixpoint (which cannot be bottom), say $[\top] \in \Omega_\perp$. From remarks above there is an ordinal β for which $(i\sigma)^\beta(\perp) = [\top]$; note that $\beta > 1$.

We write σ^1 for $\sigma(\perp)$, $\sigma^{\alpha+1}$ for $\sigma([\sigma^\alpha])$ and σ^λ for $\bigvee_{\alpha<\lambda} \sigma^\alpha$; note that each supremum exists.

The ordinals $\omega + 2$ and $\omega + 1$ are objects of $\mathcal{O}n\mathcal{CPO}$; we take $\omega \stackrel{\text{def}}{=} \{1, 2, \ldots\}$. Define a morphism $f: (\omega + 2)_\perp \to \omega + 2$ by setting

$$f(x) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } x = \perp \\ n+1 & \text{if } x = [n] \\ \omega & \text{if } x = [\omega] \\ \omega+1 & \text{if } x = [\omega+1] \end{cases}$$

and a morphism $f_r: (\omega + 1)_\perp \to \omega + 1$ to be f with restricted (co)domain.

Consider the following diagram in which j is inclusion:

$$\begin{tikzcd}
\Omega_\perp \arrow[r, "\sigma"] \arrow[d, "\tilde{f}_{r\perp}"'] \arrow[dr, phantom, "(1)"] & \Omega \arrow[d, "\tilde{f}_r"] \\
(\omega+1)_\perp \arrow[r, "f_r"] \arrow[d, "j_\perp"'] \arrow[dr, phantom, "(2)"] & \omega+1 \arrow[d, "j"] \\
(\omega+2)_\perp \arrow[r, "f"] & \omega+2
\end{tikzcd}$$

(1) commutes using the universal property of a FPO, (2) trivially. Similarly, there is $\tilde{f}: \Omega \to \omega + 2$ making

$$\begin{tikzcd}
\Omega_\perp \arrow[r, "\sigma"] \arrow[d, "\tilde{f}_\perp"'] \arrow[dr, phantom, "(3)"] & \Omega \arrow[d, "\tilde{f}"] \\
(\omega+2)_\perp \arrow[r, "f"] & \omega+2
\end{tikzcd}$$

commute. From uniqueness, $\tilde{f} = j\tilde{f}_r$. We shall now consider the cases when β is strictly less than, or greater than, ω. In the latter case we obtain a contradiction by defining a morphism different from $\tilde{f}$ which makes (3) commute.

(*Case* $1 < \beta < \omega$): By chasing (1) we see that for any finite ordinal k, $\tilde{f}_r(\sigma^k) = k$. From the definition of β it is easy to see that $\sigma^\beta = \top$. Chasing (1) at $[\top] \in \Omega_\perp$ we get

$$\beta = \tilde{f}_r(\top) = \tilde{f}_r\sigma([\top]) = f_r\tilde{f}_{r\perp}([\top]) = f_r([\tilde{f}_r(\top)]) = \beta + 1$$

a contradiction. Hence we must have

(*Case* $\omega \le \beta$): Certainly $\sigma^k \le \sigma^\omega$ for any finite ordinal k, and so $k = \tilde{f}_r(\sigma^k) \le \tilde{f}_r(\sigma^\omega)$. Hence $\tilde{f}_r(\sigma^\omega) = \omega$. As $\sigma^\beta = \top$ we have $\omega = \tilde{f}_r(\sigma^\omega) \le \tilde{f}_r(\top)$ and so $\tilde{f}_r(\top) = \omega$.

Define a function $F: \Omega \to \omega + 2$ by

$$F(n) \stackrel{\text{def}}{=} \begin{cases} \omega + 1 & \text{if } \top \le n \\ \tilde{f}(n) & \text{otherwise} \end{cases}$$

Note that $F(\top) = \omega + 1$ and $\tilde{f}(\top) = j\tilde{f}_r(\top) = \omega$, that is $F \ne \tilde{f}$. It is the case that F is monotone:

1. On elements of Ω not greater than $\top$, F is monotone for $\tilde{f}$ is.
2. If $\top \le n \le n'$ in Ω then $F(n) = F(n')$.

3. If $n \leq n'$ in Ω where n' is greater than $\top$ but n is not, then $F(n) \leq \omega + 1 = F(n')$.

Finally, F makes the square (3) commute.

1. Clearly $\sigma^1 \leq \top$ and σ^1 cannot be $\top$. So $F\sigma(\perp) = \tilde{f}(\sigma^1) = 1 = fF_\perp(\perp)$.
2. Suppose $[n] \in \Omega_\perp$ and n is greater than $\top$. Then $\top = \sigma([\top]) \leq \sigma([n])$. So

$$F\sigma([n]) = \omega + 1 = f([F(n)]) = fF_\perp([n]).$$

3. Suppose $[n] \in \Omega_\perp$ and n is not greater than $\top$. Then $\sigma([n])$ is not greater than $\top$, for if so, as σ is an isomorphism, $[\top] = \sigma^{-1}(\top) \leq [n]$ and this is not so. Hence,

$$F\sigma([n]) = \tilde{f}(\sigma([n])) = f(\tilde{f}_\perp([n])) = f([\tilde{f}(n)]) = f([F(n)]) = fF_\perp([n]).$$

By the uniqueness criterion, we have $F = \tilde{f}$. This is not so. Hence there can be no fixpoint object in $\mathcal{O}n\mathcal{CPO}$, as claimed. $\square$

5 Gluing for Cartesian Closed Let-Categories

Our aim now is to investigate to what extent the gluing construction of Freyd applies in the setting of categories with FPOs. This is interesting in itself, but in particular we are aiming to use such methods to prove a conservative extension result in Section 7. The technique of gluing originated with Freyd, who used it to present a neat method for proving the existence and disjunction properties of certain intuitionistic type theories. His proof made use of a certain topos $\mathcal{E}$ manufactured from two other toposes; $\mathcal{E}$ is the so-called glued topos and for an account of this see [12], page 237. Other information about gluing (sometimes called sconing) can be found in [8], [11], [10], [5], page 190 and [2], page 185.

As a first step towards our goal, we show that the gluing construction for cccs can be extended to let-cccs:

THEOREM 5.1 Let $\Gamma\colon \mathcal{D} \to \mathcal{C}$ be a functor where $\mathcal{D}$ is a let-ccc and $\mathcal{C}$ is a ccc. Suppose also that $\mathcal{C}$ has pullbacks and that Γ preserves finite products. Write $\mathcal{GL}(\Gamma)$ for the comma category $(\mathcal{C} \downarrow \Gamma)$ and note that there is an obvious functor $\pi_2\colon \mathcal{GL}(\Gamma) \to \mathcal{D}$. Then it is the case that

1. $\mathcal{GL}(\Gamma)$ is a let-ccc, and
2. π_2 is a morphism of let-cccs.

Proof It is a well known result (see, for example, [10] or [2], page 189) that $\mathcal{GL}(\Gamma)$ is a ccc . We shall denote a typical morphism in $\mathcal{GL}(\Gamma)$ by

$$(a, s): (A, f, D) \to (B, g, D').$$

Recall that to be a let-ccc is not a property of the category, but a chosen structure. By the statement that $\mathcal{GL}(\Gamma)$ is a let-ccc, we mean that there is a canonical choice of this structure. We make the following definitions:

1. $T(A, f, D) \stackrel{\text{def}}{=} (A, \Gamma(\eta) \circ f, TD)$.
2. $\eta_{(A,f,D)} \stackrel{\text{def}}{=} (id_A, \eta_D): (A, f, D) \longrightarrow T(A, f, D)$.
3. $$\frac{(a, s): (A, f, D) \times (B, g, D') \longrightarrow T(C, h, D'')}{\mathit{lift}(a, s) \stackrel{\text{def}}{=} (a, \mathit{lift}(s)): (A, f, D) \times T(B, g, D') \longrightarrow T(C, h, D'')}$$

Some readers may prefer to see the definition of the strong monad; the multiplication is given by

$$\mu_{(A,f,D)} \stackrel{\text{def}}{=} (id_A, \mu_D)$$

and the strength is given by

$$\tau_{((A,f,D),(B,g,D'))} \stackrel{\text{def}}{=} (id_{A\times B}, \tau_{(D,D')}).$$

Definitions (1) and (2) clearly make sense. We note that definition (3) is good:

$$\begin{tikzcd}
A \times B \arrow[r, "f \times g"] \arrow[d, "a"'] & \Gamma D \times \Gamma D' \arrow[r, shift left, "\cong"'] & \Gamma(D \times D') \arrow[l, shift left] \arrow[r, "\Gamma(id \times \eta)"] \arrow[d, "\Gamma(s)"] & \Gamma(D \times TD') \arrow[d, "\Gamma(\mathit{lift}(s))"] \\
C \arrow[r, "h"] & \Gamma D'' \arrow[r, "\Gamma(\eta)"] & \Gamma TD'' \arrow[r, shift left] & \Gamma TD'' \arrow[l, shift left, "="]
\end{tikzcd}$$

We have to verify that the equations of a let-category (see page 4) hold. We just check one of them: Take morphisms

$$\begin{aligned}
(a, s)&: (C, h, D'') \times (A, f, D) \longrightarrow T(B, g, D')\\
(b, t)&: (C, h, D'') \times (B, g, D') \longrightarrow T(E, k, D''')\\
(\pi, \pi')&: (C, h, D'') \times (A, f, D) \longrightarrow (C, h, D'').
\end{aligned}$$

Consider the commutative diagram in Figure 3. From this, it is easy to see that

$$\begin{aligned}
\mathit{lift}(\mathit{lift}(b, t) \circ \langle(\pi, \pi'), (a, s)\rangle) &= \mathit{lift}(\mathit{lift}(b, t) \circ (\langle\pi, a\rangle, \langle\pi', s\rangle))\\
\text{and using the diagram} &= \mathit{lift}(b, t) \circ (\langle\pi, a\rangle, \langle\pi', \mathit{lift}(s)\rangle)\\
&= \mathit{lift}(b, t) \circ \langle(\pi, \pi'), (a, \mathit{lift}(s))\rangle.
\end{aligned}$$

This is exactly `LiftA` for the glued category $\mathcal{GL}(\Gamma)$. It is immediate that π_2 is a morphism of let-cccs. ☐

$$
\begin{tikzcd}
C \times A \arrow[rr, "\langle \pi, a \rangle"] \arrow[d, "h"'] \arrow[dr, "h \times f"] & & C \times B \arrow[r, "b"] \arrow[d, "h \times (\Gamma(\eta) \circ g)"] & E \arrow[ddd, "\Gamma(\eta) \circ k"'] \\
\Gamma D'' \times \Gamma T D \arrow[dd, shift right, "\cong"] & \Gamma D'' \times \Gamma D \arrow[l, "id \times \Gamma(\eta)"] \arrow[d, shift right, "\cong"] & \Gamma D'' \times \Gamma T D' \arrow[dd, shift right, "\cong"] & \\
& \Gamma(D'' \times D) \arrow[u, shift right] \arrow[dl, "\Gamma(id \times \eta)"'] \arrow[dr, "\Gamma(\langle \pi', s \rangle)"] & & \\
\Gamma(D'' \times T D) \arrow[uu, shift right] \arrow[rr, "\Gamma(\langle \pi', \mathit{lift}(s) \rangle)"'] \arrow[d, equal] & & \Gamma(D'' \times T D') \arrow[uu, shift right] \arrow[r, "\Gamma(\mathit{lift}(t))"'] & \Gamma T D''' \arrow[d, equal] \\
\Gamma(D'' \times T D) \arrow[rrr, "\Gamma(\mathit{lift}(\mathit{lift}(t) \langle \pi', s \rangle))"] & & & \Gamma T D'''
\end{tikzcd}
$$

Figure 3: A Diagram Used in the Proof of Theorem 5.1

We use the theorem on gluing to prove the following corollary: please see Remark 7.3 for comments on the use of this result. We refer to the proof of Corollary 5.2 in the proof of Corollary 6.8.

COROLLARY 5.2 Let $\mathcal{C}$ be a locally small category, and $\mathcal{FC}$ the freely generated let-ccc. Then the canonical functor $I: \mathcal{C} \to \mathcal{FC}$ is full and faithful.

PROOF Let Γ be the composite

$$(I^{op})^* \circ Y: \mathcal{FC} \longrightarrow \mathcal{S}et^{\mathcal{FC}^{op}} \longrightarrow \mathcal{S}et^{\mathcal{C}^{op}}$$

where Y is the Yoneda embedding. Thus, by definition, if D is an object of $\mathcal{FC}$, then ΓD is the object of $\mathcal{S}et^{\mathcal{C}^{op}}$ given by

$$\mathcal{C}^{op} \xrightarrow{I^{op}} \mathcal{FC}^{op} \xrightarrow{YD} \mathcal{S}et.$$

One can see that Γ preserves finite products, because both Y and I do. We may define a functor

$$J: \mathcal{C} \longrightarrow \mathcal{GL}(\Gamma)$$

by setting $JC \stackrel{\text{def}}{=} (YC, \gamma^C, IC)$ and $Jc \stackrel{\text{def}}{=} (Yc, Ic)$, where the component of the natural transformation γ^C at an object C' of $\mathcal{C}$ is a function

$$\gamma^C_{C'}: \mathcal{C}(C', C) \longrightarrow \mathcal{FC}(IC', IC)$$

given by $c \mapsto Ic$. Consider the diagram

$$\begin{tikzcd}
 & \mathcal{FC} \arrow[r, equal] \arrow[d, "\overline{J}"] & \mathcal{FC} \arrow[dd, "id_{\mathcal{FC}}"] \\
\mathcal{C} \arrow[ur, "I"] \arrow[r, "J"] \arrow[dr, "I"'] & \mathcal{GL}(\Gamma) \arrow[d, "\pi_2"] & \\
 & \mathcal{FC} \arrow[r, equal] & \mathcal{FC}
\end{tikzcd}$$

The functor J preserves products because both I and Y do. Thus the functor $\overline{J}$ exists because Theorem 5.1 says that $\mathcal{GL}(\Gamma)$ is a let-ccc. The diagram does indeed commute: The upper triangle commutes up to (canonical) isomorphism using property (ii) of $\mathcal{FC}$. The lower triangle commutes up to strict equality for trivial reasons. The rectangle commutes up to (canonical) isomorphism from property (iii) of $\mathcal{FC}$ because π_2 is a morphism of let-cccs.

It follows that $\overline{J}$ is faithful for trivial reasons. Thus I will be full and faithful if we can show that J is, which we now do.

That Y is faithful clearly implies that J is too. It remains to show that J is full which takes a little effort. Suppose that $(\phi, h): JC \to JC'$ in $\mathcal{GL}(\Gamma)$. We need to find $c: C \to C'$ in $\mathcal{C}$ such that $Jc \stackrel{\text{def}}{=} (Yc, Ic) = (\phi, h)$. Now, $\phi: YC \to YC'$, and it is standard (from the proof that Y is full) that $\phi = Y(\phi_C(id_C))$ (see page 11 of [12]). Set $c \stackrel{\text{def}}{=} \phi_C(id_C)$. We are done if $h = Ic$. In fact this follows from the fact that (ϕ, h) is a morphism in $\mathcal{GL}(\Gamma)$, that is, the diagram

$$\begin{tikzcd} YC \arrow[r, "\phi"] \arrow[d, "\gamma^C"'] & YC' \arrow[d, "\gamma^{C'}"] \\ \Gamma IC \arrow[r, "\Gamma h"'] & \Gamma IC' \end{tikzcd}$$

commutes. One needs to take components of the diagram at C, and trace the diagram for $id_C \in (YC)C = \mathcal{C}(C, C)$, which yields $h = I(\phi_C(id_C)) = Ic$. $\square$

6 Gluing for FIX-Categories

As remarked at the start of Section 5, our aim is to consider the gluing construction for categories with FPO. However, we might also consider categories with other structure, such as coproducts and natural numbers objects (NNOs). Indeed, in Section 7 it will be crucial that we can deal with such additional structure. We shall refer to cartesian closed categories with finite coproducts, NNO and FPO as *FIX-categories*. The author is of the opinion that this name is not ideal. However, as it was used in the reference [4], which is the original source for the current work, we shall stick with it.

We would like to be able to prove a variant of the gluing construction for the case of a finite product preserving functor $\Gamma: \mathcal{D} \to \mathcal{C}$ where $\mathcal{C}$ and $\mathcal{D}$ are FIX-categories. However, close inspection shows that it is not at all clear how to define a FPO in $(\mathcal{C} \downarrow \Gamma)$, that is, there is certainly no obvious candidate for a FPO. In fact the author suspects that the comma category is not in general a FIX-category, though no counter example has been found. We shall provide a limited solution to this problem by giving a new gluing style construction in the case that $\mathcal{C}$ is a functor category with target $\omega\mathcal{CPO}$. This construction is most interesting in itself, but moreover will be put to good use in Section 7 where it provides a key step in the proof of a theorem.

For this section we proceed as follows. The first few propositions are devoted to showing that $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$ is a FIX-category. We can then prove Theorem 6.7, a gluing construction, which is our main result. Finally, we prove a corollary which will be used crucially in Section 7.

PROPOSITION 6.1 Let $\mathcal{C}$ be locally small. Then there is a NNO $(\hat{N}, \hat{s}, \hat{0})$ in the functor category $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$.

PROOF A standard result for the topos $\mathcal{S}et^{\mathcal{C}^{op}}$. The same proof works here, namely, the NNO is defined pointwise. $\square$

PROPOSITION 6.2 Let $\mathcal{C}$ be locally small. Then the functor category $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$ has finite products (for $\omega\mathcal{CPO}$ does) and is in fact a let-category $(\perp, \eta, \mathit{lift})$ with a structure inherited from the lifting let-structure of $\omega\mathcal{CPO}$—see page 8.

PROOF $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$ is a let-category where

- if $F: \mathcal{C}^{op} \to \omega\mathcal{CPO}$ then $\perp F$ sends an object C of $\mathcal{C}$ to $(FC)_\perp$ and sends a morphism $c: C' \to C$ in $\mathcal{C}$ to $(Fc)_\perp$,
- $\eta_F: F \to \perp F$ is given by $(\eta_F)_C \stackrel{\text{def}}{=} \iota_{FC}: FC \to (FC)_\perp$, and
- if $\alpha: F \times F' \to F''$, then $\mathit{lift}(\alpha): F \times \perp F' \to F''$ is given by $\mathit{lift}(\alpha)_C \stackrel{\text{def}}{=} \mathit{lift}(\alpha_C)$.

$\square$

PROPOSITION 6.3 There is a FPO $(\hat{\Omega}, \hat{\sigma}, \hat{\omega})$ in $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$ for the let structure $(\perp, \eta, \mathit{lift})$.

PROOF We define $\hat{\Omega}: \mathcal{C}^{op} \to \omega\mathcal{CPO}$ by sending an object C of $\mathcal{C}$ to the vertical natural numbers $\mathbb{N}^\infty$, and any morphism of $\mathcal{C}^{op}$ to $id_{\mathbb{N}^\infty}$. The natural transformations $\hat{\sigma}$ and $\hat{\omega}$ are defined pointwise. $\square$

LEMMA 6.4 Let $dis: \mathcal{S}et \to \omega\mathcal{CPO}$ map sets to discrete ωcpos, and write $Y: \mathcal{C}^{op} \to \mathcal{S}et^{\mathcal{C}^{op}}$ for the Yoneda embedding. Write $H \stackrel{\text{def}}{=} dis_* \circ Y: \mathcal{C}^{op} \to \omega\mathcal{CPO}^{\mathcal{C}^{op}}$.

Then $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$ is an $\omega\mathcal{CPO}$ enriched category and moreover there is an isomorphism

$$FC \cong \omega\mathcal{CPO}^{\mathcal{C}^{op}}(H_C, F)$$

in $\omega\mathcal{CPO}$ which is natural in C and F.

PROOF This is essentially the Yoneda lemma for enriched categories, but we give "bare hand" details.

Given objects (functors) F and F' of the functor category $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$, we can regard the set of natural transformations $\omega\mathcal{CPO}^{\mathcal{C}^{op}}(F, F')$ as a poset with the pointwise order, that is if $\alpha, \alpha': F \to F'$ then $\alpha \leq \alpha'$ just in case for all objects C of $\omega\mathcal{CPO}$, $\alpha_C \leq \alpha'_C$ as Scott continuous functions (more precisely $\alpha_C \leq \alpha'_C$ in the exponential ωcpo $FC \Rightarrow F'C$).

It is easy to verify that $\omega\mathcal{CPO}^{\mathcal{C}^{op}}(F, F')$ is an ωcpo: For let $(\alpha^i \mid i \in \omega)$ be an ω-chain in $\omega\mathcal{CPO}^{\mathcal{C}^{op}}(F, F')$. We need to see that the join of the chain exists. Clearly, if there is a natural transformation $\beta: F \to F'$ whose component at an object A of

$\mathcal{C}$ is $\beta_A \stackrel{\text{def}}{=} \bigsqcup_{i\in\omega}\alpha^i_A$ (and the latter join certainly exists) then $\bigsqcup_{i\in\omega}\alpha^i$ exists and is β. So let $a: A' \to A$ in $\mathcal{C}$ and take any $\xi \in FA$; we need to see that

$$\begin{tikzcd}
FA \arrow[r, "\beta_A"] \arrow[d, "Fa"'] & F'A \arrow[d, "F'a"] \\
FA' \arrow[r, "\beta_A"'] & F'A'
\end{tikzcd}$$

commutes. We have

$$\begin{array}{rcl}
F'a \circ \beta_A(\xi) & = & F'a((\bigsqcup_{i\in\omega}\alpha^i_A)\xi) \\
& = & F'a(\bigsqcup_{i\in\omega}\alpha^i_A(\xi)) \\
& = & \bigsqcup_{i\in\omega}(F'a(\alpha^i_A(\xi))) \\
& = & \bigsqcup_{i\in\omega}(\alpha^i_{A'}(Fa(\xi))) \\
& = & \beta_{A'} \circ (Fa)(\xi).
\end{array}$$

Thus $\omega\mathcal{CPO}(F, F')$ is an ωcpo and the functor category is certainly $\omega\mathcal{CPO}$-enriched. The required isomorphism in $\omega\mathcal{CPO}$ follows by using the principle of the proof of the Yoneda lemma for $\mathcal{S}et$. □

PROPOSITION 6.5 The category $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$ is a cartesian closed category with finite coproducts.

PROOF Of course $\omega\mathcal{CPO}$ has finite products implying that $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$ does too from standard results. Let us look at cartesian closure. In fact the proof is essentially identical to that for the presheaf category $\mathcal{S}et^{\mathcal{C}^{op}}$.

Let $F, F': \mathcal{C}^{op} \to \omega\mathcal{CPO}$ be objects of the functor category. We define a functor $F \Rightarrow F': \mathcal{C}^{op} \to \omega\mathcal{CPO}$ as follows: on object C of $\mathcal{C}^{op}$ set

$$(F \Rightarrow F')C \stackrel{\text{def}}{=} \omega\mathcal{CPO}^{\mathcal{C}^{op}}(H_C \times F, F')$$

qua object of $\omega\mathcal{CPO}$ via the enrichment of Lemma 6.4. Note that if the exponential $F \Rightarrow F'$ exists then it *must* have the above action on objects C by considering the bijection of Lemma 6.4. On morphisms $c: C' \to C$ we define

$$(F \Rightarrow F')c \stackrel{\text{def}}{=} \omega\mathcal{CPO}^{\mathcal{C}^{op}}(H_c \times id_F, id_{F'}).$$

It is routine to verify that $(F \Rightarrow F')c$ is a Scott continuous function.

We define a natural transformation $ev: (F \Rightarrow F') \times F \longrightarrow F'$ by giving a Scott continuous function $ev_C: (F \Rightarrow F')C \times FC \to F'C$ defined by $ev_C(\Phi, x) \stackrel{\text{def}}{=}$

$\Phi_C(id_C, x)$. Finally, if $\alpha: F \times F' \to F''$ then $\lambda(\alpha): F \to (F' \Rightarrow F'')$ is defined by giving for each object C of $\mathcal{C}$ a Scott continuous function

$$\lambda(\alpha)_C: FC \to (F' \Rightarrow F'')(C)$$

which is in turn defined by giving for each $x \in FC$ and C'' in $\mathcal{C}$ a Scott continuous function

$$\lambda(\alpha)_C(x)_{C'} : \mathcal{C}(C', C) \times F'C' \to F''C'$$

given by $\lambda(\alpha)_C(x)_{C'}(c, x') \stackrel{\mathrm{def}}{=} \alpha_{C'}(F(c)(x), x')$. The detailed verification of continuity is routine. Once this is done the result follows from the standard proof that $\mathcal{S}et^{\mathcal{C}^{op}}$ is cartesian closed. Finite coproducts are defined pointwise and we omit the details. $\square$

Corollary 6.6 $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$ is a FIX-category.

Proof Follows from Propositions 6.1, 6.2, 6.3 and 6.5. $\square$

We can now state and prove the most important result of this paper, which is a gluing style construction closely related to logical relations techniques. This theorem will be used to prove Corollary 6.8, and the corollary will be applied in Section 7.

Theorem 6.7 Let $\mathcal{D}$ be a FIX-category and let $I: \mathcal{C} \to \mathcal{D}$ preserve finite products. Write

$$\Gamma \stackrel{\mathrm{def}}{=} (I^{op})^* \circ H: \mathcal{D} \longrightarrow \omega\mathcal{CPO}^{\mathcal{D}^{op}} \longrightarrow \omega\mathcal{CPO}^{\mathcal{C}^{op}}.$$

We define a category $\mathcal{G}l(\Gamma)$ as follows:

- The objects of $\mathcal{G}l(\Gamma)$ are triples $(F, \lhd, D)$ where $F: \mathcal{C}^{op} \to \omega\mathcal{CPO}$ is a functor, D is an object of $\mathcal{D}$, and $\lhd$ is an operation which assigns to each object C of $\mathcal{C}$ a subobject, denoted by $\lhd_C$, of $FC \times \Gamma DC$.
- A morphism $(\alpha, d): (F, \lhd, D) \to (F', \lhd', D')$ is given by a natural transformation $\alpha: F \to F'$ and a morphism $d: D \to D'$ in $\mathcal{D}$ for which if $x \lhd_C u$ then $\alpha_C(x) \lhd'_C d \circ u$, where of course $x \in FC$ and $u \in \Gamma DC \stackrel{\mathrm{def}}{=} \mathcal{D}(IC, D)$.

Then $\mathcal{G}l(\Gamma)$ is a FIX-category (for a particular strong monad) and the obvious functor $\pi_2: \mathcal{G}l(\Gamma) \to \mathcal{D}$ is a morphism of FIX-categories.

Proof That $\mathcal{G}l(\Gamma)$ is indeed a category is routine. As a general principle, the structure of $\mathcal{G}l(\Gamma)$ is specified by a "logical relations" procedure on the subobjects $\lhd_C$. We sketch most of the constructions, giving full details for exponentials and the FPO; Corollary 6.6 is used implicitly.

(*Terminal Object*): The terminal object is $(1_{\omega\mathcal{CPO}^{\mathcal{C}^{op}}}, \lhd^1, 1_{\mathcal{D}})$ where for an object C of $\mathcal{C}$ we set $* \lhd^1_C\ !_{IC}$. Here, $* \in 1_{\omega\mathcal{CPO}^{\mathcal{C}^{op}}}(C) = \{*\}$, and $!_{IC}: IC \to 1_{\mathcal{D}}$ is the unique map to the terminal object of $\mathcal{D}$.

(*Binary Products*): We set

$$(F, \lhd, D) \times (F', \lhd', D') \stackrel{\text{def}}{=} (F \times F', \lhd \times \lhd', D \times D')$$

where $(x, x')(\lhd \times \lhd')_C u$ just in case $x \lhd_C \pi u$ and $x' \lhd'_C \pi' u$ where of course $\pi\colon D \times D' \to D$ and $\pi'\colon D \times D' \to D'$ in $\mathcal{D}$. It is easy to see that $(\lhd \times \lhd')_C$ is indeed a subobject. The projections in $\mathcal{G}l(\Gamma)$ are of course given by pairing of projections in $\omega\mathcal{CPO}^{\mathcal{C}^{op}}$ and $\mathcal{D}$, such as:

$$\pi_{(F,\lhd,D)} \stackrel{\text{def}}{=} (\pi_F, \pi_D) : (F \times F', \lhd \times \lhd', D \times D') \longrightarrow (F, \lhd, D).$$

(*Binary Coproducts*): Write $i\colon A \to A + A'$ and $j\colon A' \to A + A'$ for coproduct insertions.

We set

$$(F, \lhd, D) + (F', \lhd', D') \stackrel{\text{def}}{=} (F + F', \lhd + \lhd', D + D')$$

where if $x \in FC$ then $i(x)(\lhd + \lhd')_C u$ just in case $u\colon IC \to D + D'$ factors through $i\colon D \to D + D'$ in $\mathcal{D}$ and if $x' \in F'C$ then $j(x')(\lhd + \lhd')_C u$ just in case u factors through $j\colon D' \to D + D'$ in $\mathcal{D}$. (Note that every $\xi \in FC + F'C$ is of the form $i(x)$ or $j(x')$).

(*Exponentials*): Set

$$(F, \lhd, D) \Rightarrow (F', \lhd', D') \stackrel{\text{def}}{=} (F \Rightarrow F', \lhd \Rightarrow \lhd', D \Rightarrow D')$$

where $\Phi(\lhd \Rightarrow \lhd')_C w$ just in case for all $x \lhd_C u$ we have

$$ev_C(\Phi, x) \lhd'_C ev \circ \langle w, u \rangle$$

(ev_C was defined in the proof of Proposition 6.5). Let us check that $(\lhd \Rightarrow \lhd')_C$ is a subobject. Take a chain $(\Phi^i(\lhd \Rightarrow \lhd')_C w \mid i \in \omega)$ and let $x \lhd_C u$. Then, using continuity of Φ^i_C,

$$ev_C(\bigsqcup \Phi^i, x) = (\bigsqcup \Phi^i)_C(id_C, x) = \bigsqcup ev_C(\Phi^i, x) \lhd'_C ev \circ \langle w, u \rangle$$

as required. The evaluation morphisms in $\mathcal{G}l(\Gamma)$ are given pointwise, as is the operation of currying. So for example if (α, d) is a morphism in $\mathcal{G}l(\Gamma)$ then we set $\lambda((\alpha, d)) \stackrel{\text{def}}{=} (\lambda(\alpha), \lambda(d))$: we check that this is a good definition. Suppose that $x \lhd_C u$ and $x' \lhd'_C u'$. Then

$$ev_C(\lambda(\alpha)_C(x), x') = \alpha_C(x, x') \lhd'_C d \circ \langle u, u' \rangle = ev \circ \langle \lambda(d) \circ u, u' \rangle$$

and so $\lambda(\alpha)_C(x)(\lhd \Rightarrow \lhd')_C \lambda(d) \circ u$ whenever $x \lhd_C u$.

(*Strong Monad*):

- $T(F, \lhd, D) \stackrel{\text{def}}{=} (\bot F, \lhd^T, TD)$ where $y \lhd^T_C v$ just in case if $y = [x] \in (FC)_\bot$ then v factors through η_D (by u say) and $x \lhd_C u$.

- $\eta_{(F,\lhd,d)} \stackrel{\text{def}}{=} (\eta_F, \eta_D)$.
- $\mathit{lift}(\alpha, d) \stackrel{\text{def}}{=} (\mathit{lift}(\alpha), \mathit{lift}(d))$, where, say,

$$(\alpha, d) : (F, \lhd, D) \times T(F', \lhd', D') \longrightarrow T(F'', \lhd'', D'').$$

It is clear that we have indeed defined a strong monad provided that the definitions make sense. Let us verify that $\mathit{lift}(\alpha, d)$ is well defined. Suppose that

$$(x, y')(\lhd \times \lhd'^T)_C t$$

and further that $\mathit{lift}(\alpha)_C(x, y') = [x''] \in (F''C)_\perp$. Then it remains to prove that $\mathit{lift}(d) \circ t$ factors through $\eta_{D''}$ (say by some $\xi: IC \to D''$) for which $x'' \lhd''_C \xi$. Now we must have $y' = [x'] \in (F'C)_\perp$ and so $[x'] \lhd'^T_C \pi' t$, implying $\pi' t = \eta_{D'} u'$ with $x' \lhd'_C u'$. But also $x \lhd_C \pi t$ and thus as (α, d) is a morphism in $\mathcal{G}l(\Gamma)$ we have

$$[x''] = \alpha_C(x, x') \lhd''^T_C d \circ \langle \pi t, u' \rangle = \eta_{D''} \circ \xi$$

for some ξ, where $x'' \lhd''_C \xi$. Finally

$$\mathit{lift}(d) \circ t = \mathit{lift}(d) \circ (id \times \eta_{D'}) \circ \langle \pi t, u' \rangle = d \circ \langle \pi t, u' \rangle,$$

using `LiftB`.

(*FPO*): The FPO is $(\hat{\Omega}, \lhd, \Omega)$ where $\lhd_C$ is a subobject of $\mathbb{N}^\infty \times \mathcal{D}(IC, \Omega)$ defined by

- $\infty \lhd_C u$ just in case $n \lhd^\Omega_C u$ for all $n \in \mathbb{N}$, and
- $n \lhd^\Omega_C u$ if there is some $u': IC \to T\Omega$ for which $u = (\sigma \circ \eta_\Omega)^n \circ \sigma \circ u'$.

The structure map is $(\hat{\sigma}, \sigma)$ and the global element is $(\hat{\omega}, \omega)$.

(*NNO*): The NNO is $(\hat{N}, \lhd, N)$ where $\lhd_C$ is a subobject of $\mathbb{N} \times \mathcal{D}(IC, N)$ given by $n \lhd u$ iff $u = s^n \circ 0$. The successor and zero morphisms are the expected ones.

It is easy to see that π_2 is a FIX-category morphism. $\square$

COROLLARY 6.8 Let $\mathcal{C}$ be a locally small category, and $\mathcal{FC}$ the freely generated FIX-category. Then the canonical functor $I: \mathcal{C} \to \mathcal{FC}$ is full and faithful.

PROOF We appeal to Theorem 6.7 where $\mathcal{D}$ is taken to be $\mathcal{FC}$. We define a functor $J: \mathcal{C} \to \mathcal{G}l(\Gamma)$: on objects C of $\mathcal{C}$ define JC by $(HC, \lhd^C, IC)$ where the subobject

$$\lhd^C_{C'} \hookrightarrow \mathcal{C}(C', C) \times \mathcal{D}(IC', IC)$$

is defined by requiring $c \lhd^C_{C'} Ic$ for each morphism $c: C' \to C$ in $\mathcal{C}$. Each $\lhd^C_{C'}$ is trivially a subobject, for

$$\mathcal{C}(C', C) \times \mathcal{D}(IC', IC)$$

is a discrete ωcpo. On morphisms c of $\mathcal{C}$ we set $Jc \stackrel{\text{def}}{=} (Hc, Ic)$. Note that J is faithful for the Yoneda embedding H is faithful. For fullness, let $(\alpha, d): JC \to JC'$. Hence $\alpha: HC \to HC'$ and so $\alpha = Hc$ for some $c: C \to C'$ in $\mathcal{C}$. Now certainly $id_C \lhd_C^C id_{IC}$ and so

$$\alpha_C(id_C) = \mathcal{C}(C, c)(id_C) = c \lhd_C^{C'} d \circ id_{IC} = d$$

implying $d = I(c)$; therefore $J(c) = (\alpha, d)$, that is J is full.

The remainder of the proof follows by a diagram chasing argument identical to that of Corollary 5.2. $\square$

Remark 6.9 In proving that $I: \mathcal{C} \to \mathcal{FC}$ is a full embedding, the crucial part of the proof is the construction of a category $\mathcal{G}$ (say) and a functor $J: \mathcal{C} \to \mathcal{G}$ which has particular factorisation properties. Theorem 5.1 is certainly a variant of Freyd's original gluing construction. As we noted, the technique of Theorem 5.1 does not obviously carry over to categories with fixpoint objects. The new construction given in Theorem 6.7 works, of course, for let-cccs and in the author's opinion furnishes a simpler construction of an appropriate glued category $\mathcal{G}$ than that of Theorem 5.1. The pointwise construction of exponentials in $\mathcal{G}l(\Gamma)$ is much easier than the construction of exponentials in $\mathcal{GL}(\Gamma)$.

Remark 6.10 It is interesting to note the following: consider an instance of Γ, say the global sections $\Gamma \stackrel{\text{def}}{=} \mathcal{D}(1, -): \mathcal{D} \to \mathcal{S}et$, in Theorem 5.1. Take objects (X, f, D) and (X', f', D') in $\mathcal{GL}(\Gamma)$, and recall that

$$(X, f, D) \Rightarrow (X', f', D') \stackrel{\text{def}}{=} (P, p_2, D \Rightarrow D')$$

arises from the pullback

$$\begin{tikzcd}
(X \Rightarrow X') \times_{X \Rightarrow \Gamma D'} \Gamma(D \Rightarrow D') \arrow[r, "p_2"] \arrow[d, "p_1"] & \Gamma(D \Rightarrow D') \arrow[d, "\lambda(\Gamma ev \circ \cong \circ (id \times f))"] \\
X \Rightarrow X' \arrow[r, "id \Rightarrow f'"'] & X \Rightarrow \Gamma D'
\end{tikzcd}$$

Take any $\theta \in X \Rightarrow X'$ and $\lambda(\phi) \in \Gamma(D \Rightarrow D')$ such that $(\theta, \lambda(\phi)) \in P$, and tracing the diagram we get

$$\begin{tikzcd}
(\theta, \lambda(\phi)) \arrow[r, mapsto] \arrow[d, mapsto] & \lambda(\phi) \arrow[d, mapsto] \\
\theta \arrow[r, mapsto] & (x \mapsto f'(\theta(x)) = (x \mapsto \phi \circ f(x))
\end{tikzcd}$$

Now consider an analogue of Theorem 6.7 in which $\mathcal{C} \stackrel{\text{def}}{=} \mathbf{1}$, the one point category, $I \stackrel{\text{def}}{=} 1_{\mathcal{D}}: \mathbf{1} \to \mathcal{D}$ is the (obvious) constant functor, and we replace $\omega\mathcal{CPO}$ by $\mathcal{S}et$ and H by Y. Identifying $\mathcal{S}et^{\mathbf{1}}$ with $\mathcal{S}et$, we have a category $\mathcal{G}l(\Gamma)$. Now consider (X, f, D) and (X', f', D') as objects of $\mathcal{G}l(\Gamma)$ with exponential

$$(X \Rightarrow X', f \Rightarrow f', D \Rightarrow D').$$

Certainly we have a subobject

$$(f \Rightarrow f') \quad \hookrightarrow \quad (X \Rightarrow X') \times \Gamma(D \Rightarrow D'),$$

and it is easy to see that in fact $(f \Rightarrow f') = P$, that is, the pullback construction of exponentials in $\mathcal{GL}(\Gamma)$ gives rise to exactly the logical relations style construction of exponentials in $\mathcal{G}l(\Gamma)$.

7 A Conservative Extension via Full Embedding

We again assume that the reader is familiar with certain equational type theories. For details see [2] and [3].

Here, a FIX-theory[6] $Th = (Sg, Ax)$ is specified by giving a *signature* Sg of *ground types* γ and *function symbols* $f: \alpha_1, \ldots, \alpha_n \to \alpha_{n+1}$, where the higher *types* are given by

$$\alpha \quad ::= \quad \gamma \mid unit \mid fix \mid nat \mid \alpha \times \alpha \mid \alpha + \alpha \mid \alpha \Rightarrow \alpha \mid T\alpha,$$

and *fix* is the fixpoint type. Ax is a collection of *equations-in-context* $\Gamma \vdash M = M': \alpha$ giving the *axioms* of the theory. The *theorems* are generated by the usual β and η rules associated with the given types.

We aim to prove Theorem 7.1: very informally, we might say that "given an algebraic theory, the FIX-theory generated from it gives a more expressive syntax for programming, but does not increase the computational power of the original theory." Note that for us, an algebraic theory is the type-theoretic counterpart of a category with finite products.

THEOREM 7.1 Let $Th = (Ax, Sg)$ be an algebraic theory where Sg is a signature of types and function symbols giving rise to well-formed terms-in-context (*proved terms*) of the form $[\vec{x}: \vec{\gamma}] \vdash_{Sg} M: \gamma$ where $[\vec{x}: \vec{\gamma}]$ is an environment of the form $[x_1: \gamma_1, \ldots, x_n: \gamma_n]$, M is a raw term, and Ax is a collection of equations-in-context of the form $\Gamma \vdash M = M': \gamma$. Write $Th' = (Ax', Sg')$ to indicate the FIX-theory which has the ground types the types of Sg, and axioms those of Ax. Then

(i) If $[\vec{x}: \vec{\gamma}] \vdash_{Th'} E: \gamma$ where γ_i and γ are ground types, then there is a raw term M of Sg such that

$$[\vec{x}: \vec{\gamma}] \vdash_{Sg} M: \gamma \qquad \text{and} \qquad [\vec{x}: \vec{\gamma}] \vdash_{Th'} E = M: \gamma.$$

[6] A FIX-theory was called a $\text{FIX}_{=}$-theory in [4].

(ii) If also $[\vec{x}{:}\,\vec{\gamma}] \vdash_{Th'} E = M'{:}\,\gamma$ where $[\vec{x}{:}\,\vec{\gamma}] \vdash_{Sg} M'{:}\,\gamma$ then

$$[\vec{x}{:}\,\vec{\gamma}] \vdash_{Th} M = M'{:}\,\gamma.$$

The key step in proving Theorem 7.1 is showing the following category-theoretic result:

PROPOSITION 7.2 Let Th and Th' be as in Theorem 7.1. Let us write $Cl(Th)$ and $Cl(Th')$ for their respective classifying categories as an algebraic theory and a FIX-theory. Then $Cl(Th')$ is the freely generated FIX-category of $Cl(Th)$ (via a functor $I{:}\, Cl(Th) \to Cl(Th')$ to be given).

PROOF This kind of result is prevalent in categorical type theory and the proof techniques routine. We sketch the details of the proof.

Objects of $Cl(Th)$ are finite lists $\vec{\gamma}$ of types of Sg and a morphism

$$[([\vec{x}{:}\,\vec{\gamma}] \mid M_1)_{Th}, \ldots, ([\vec{x}{:}\,\vec{\gamma}] \mid M_n)_{Th}]{:}\, \vec{\gamma} \longrightarrow \vec{\gamma}'$$

is a list of equivalence classes of pairs $([\vec{x}{:}\,\vec{\gamma}], M)$ (where $[\vec{x}{:}\,\vec{\gamma}] \vdash_{Sg} M_i{:}\,\gamma_i')$ using the equivalence relation of provable equality in Th.

An object of $Cl(Th')$ is a type σ of Sg', and a morphism $(x{:}\,\sigma \mid M)_{Th'}{:}\,\sigma \to \tau$ is given by a similar equivalence class up to provable equality in Th'.

We define a finite product preserving functor $I{:}\, Cl(Th) \to Cl(Th')$ by $I(\vec{\gamma}) \stackrel{\text{def}}{=} \Pi_1^n \gamma_i$ and

$$I([([\vec{x}{:}\,\vec{\gamma}] \mid M_1)_{Th}, \ldots, ([\vec{x}{:}\,\vec{\gamma}] \mid M_n)_{Th}]) \stackrel{\text{def}}{=} (z{:}\,\Pi_1^n \gamma_i \mid \langle \widehat{M_1}, \ldots, \widehat{M_n} \rangle)_{Th'}$$

where $\widehat{M_i} \stackrel{\text{def}}{=} M_i[\mathsf{Proj}_i/x_i]$ and $\mathsf{Proj}_i(z)$ is the obvious raw term of the FIX-theory such that $z{:}\,\Pi_1^n \gamma_i \vdash_{Sg} \mathsf{Proj}_i(z){:}\,\gamma_i$, and the sub-scripts Th and Th' indicate in which theory we are taking provable equality.

Let $F{:}\, Cl(Th) \to \mathcal{D}$ where $\mathcal{D}$ is a FIX-category. The definition of the functor $\overline{F}{:}\, Cl(Th') \to \mathcal{D}$ is by structural induction. We just give the cases involving T-types and the *fix* type.

- $\overline{F}(T\sigma) \stackrel{\text{def}}{=} T(\overline{F}\sigma)$,
- $\overline{F}(x{:}\,\sigma \mid \mathsf{Val}(M)) \stackrel{\text{def}}{=} \eta_{\overline{F}\tau} \circ \overline{F}(x{:}\,\sigma \mid M)$, (where $x{:}\,\sigma \vdash_{Sg} M{:}\,\tau$, say)
- $$\begin{aligned}&\overline{F}(x{:}\,\sigma \mid \mathsf{Let}\,(E, y.F)) \stackrel{\text{def}}{=} \\ &\qquad \mathit{lift}(\overline{F}(z{:}\,\sigma \times \tau \mid F[\mathsf{Fst}(z), \mathsf{Snd}(z)/x, y])) \circ \langle id_{\overline{F}\sigma}, \overline{F}(x{:}\,\sigma \mid E)\rangle,\end{aligned}$$
- $\overline{F}(\mathit{fix}) \stackrel{\text{def}}{=} \Omega$ for Ω in $\mathcal{D}$,

- $\overline{F}(x\colon unit \mid \omega) \stackrel{\text{def}}{=} \omega$ in $\mathcal{D}$,
- $\overline{F}(x\colon \sigma \mid \mathsf{Inc}(E)) \stackrel{\text{def}}{=} \sigma \circ \overline{F}(x\colon \sigma \mid E)$,
-

$$\begin{aligned}&\overline{F}(x\colon \sigma \mid \mathsf{It}(y.F, N)) \stackrel{\text{def}}{=} \\ &\qquad (\overline{F}(z\colon \sigma \times T\tau \mid F[\mathsf{Fst}(z), \mathsf{Snd}(z)/x, y])) \circ \langle id_{\overline{F}\sigma}, \overline{F}(x\colon \sigma \mid N)\rangle.\end{aligned}$$

It is immediate from its definition that $\overline{F}$ is a FIX-category morphism.

We define a natural isomorphism $\phi\colon \overline{F}I \to F\colon Cl(Th) \to \mathcal{D}$ by observing that at each object $\vec{\gamma}$ of $Cl(Th)$ we have

$$\overline{F}I(\vec{\gamma}) = \overline{F}(\Pi_1^n \gamma_i) \stackrel{\text{def}}{=} \Pi_1^n F([\gamma_i]) \cong F(\vec{\gamma})$$

in $\mathcal{D}$. One can prove by a lengthy structural induction over the morphisms of $Cl(Th)$ that the above isomorphisms give rise to a *natural* isomorphism ϕ. $\square$

Now we can prove Theorem 7.1:

PROOF It follows from Corollary 6.8 that $I\colon Cl(Th) \to Cl(Th')$ is a full embedding, and this proves the theorem: we expand on the details for the existence part of the theorem. Suppose that $x_1\colon \gamma_1, \ldots, x_n\colon \gamma_n \vdash_{Sg'} E\colon \gamma$. We shall write

$$\widehat{E} \stackrel{\text{def}}{=} E[\mathsf{Proj}_1(z)/x_1, \ldots, \mathsf{Proj}_j(z)/x_j, \ldots, \mathsf{Proj}_n(z)/x_n]$$

and

$$\Pi\vec{\gamma} \stackrel{\text{def}}{=} (\ldots(\gamma_1 \times \gamma_2) \times \ldots) \times \gamma_n.$$

Then we certainly have $e \stackrel{\text{def}}{=} (z\colon \Pi\vec{\gamma} \mid \widehat{E})_{Th'}\colon I\vec{\gamma} \to I[\gamma]$ in $\mathcal{E}'$. Using the fullness of I, there is a morphism $([x_1\colon \gamma_1, \ldots, x_n\colon \gamma_n] \mid M)_{Th}\colon \vec{\gamma} \to [\gamma]$ which is taken to e by I. But this implies that $z\colon \Pi\vec{\gamma} \vdash_{Th'} \widehat{M} = \widehat{E}\colon \gamma$ (using an obvious notation for $\widehat{M}$) and so $x_1\colon \gamma_1, \ldots, x_n\colon \gamma_n \vdash_{Th'} M = E\colon \gamma$. The uniqueness part of the theorem follows from I's faithfulness in a similar fashion. $\square$

REMARK 7.3 The original (category-theoretic) proof that "λ-calculus is conservative over algebra" uses techniques similar to those found in Theorem 5.1 and Corollary 5.2. We have included these two results for their intrinsic interest, and because they amount to the obvious extension of the original proof technique. However, for the purposes of proving the results in Section 7, the methods of Section 6 subsume those of Section 5.

8 Conclusions and Acknowledgements

We have presented some miscellaneous results about fixpoint objects. We have also extended some standard ideas of category theory, such as polynomial categories and gluing (for cccs), to cccs equipped with a strong monad. We noted that the usual gluing construction does not obviously apply to FIX-categories, and solved the problem by giving a new, but similar, construction for a limited class of functors. This yields a corresponding type-theoretic conservative extension result for FIX-theories.

The precise relationship between the underlying construction of $\mathcal{G}l(\Gamma)$, and the logical-relations structure of $\mathcal{GL}(\Gamma)$ needs to be understood from a more abstract point of view. The form of the construction of the latter category also needs to be compared with other applications of gluing, such as the existence and disjunction properties of intuitionistic logic.

I would like to thank Martin Hyland and Andrew Pitts for comments on material in Sections 2 to 5. Thanks to the U.K. Engineering and Physical Sciences Research Council (formally SERC) for funding in the form of a Research Fellowship. Two referees supplied a number of astute comments on the original version of this paper, which I have acted upon. I believe that their reports have significantly improved the original presentation, and also resulted in non-trivial changes and additions to a couple of proofs. I am most grateful to them for their advice. Last, but not least, the commutative diagrams were drawn using Paul Taylor's Commutative Diagrams package.

References

[1] M. Barr and C. Wells.: 1990, Category Theory for Computing Science, *International Series in Computer Science, Prentice Hall.*

[2] Crole R.L.: 1993, Categories for Types, *Cambridge Mathematical Textbooks, Cambridge University Press.*

[3] Crole R. L.:1991, Programming Metalogics with a Fixpoint Type, *PhD thesis, Computer Laboratory, University of Cambridge.*

[4] Crole R. L. and Pitts A. M.:1992, New foundations for fixpoint computations: FIX-hyperdoctrines and the FIX-logic, *Information and Computation,* **Vol. no. 98** pp. 171–210.

[5] Freyd P. J. and Scedrov A.: 1990, Categories, Allegories, *North Holland.*

[6] Semantics of Programming Languages: Structures and Techniques: 1992, *Foundations of Computing, MIT Press.*

[7] Jacobs B.:1991, Categorical Type Theory. *Ph.D. thesis, University of Nijmegen.*

[8] Johnstone P. T.:1977, *Topos Theory*, **Academic Press**.

[9] Kock A.:1970, Monads on symmetric monoidal closed categories, *Archiv der Mathematik,* **Vol. no. 21**, pp. 1–10.

[10] Lafont Y.:1988, Logiques, Catégories et Machines, *PhD thesis, Univ. Paris VII.*

[11] Lambek J. and Scott P. J.:1980, Intuitionist type theory and the free topos, *Journal of Pure and Applied Algebra,* **Vol. no. 19**, pp. 215–257.

[12] Lambek J. and Scott P. J.:1986, Introduction to Higher Order Categorical Logic, *Cambridge Studies in Advanced Mathematics, Cambridge University Press.*

[13] Mac Lane S.:1971, Categories for the Working Mathematician, *Graduate Texts in Mathematics, Springer-Verlag*, **Vol. no. 5**.

[14] Moggi E.:1989, Notions of computation and monads, *Theoretical Computer Science*, **Vol. no. 93**, pp. 55–92.

[15] Pitts A. M.:1995, Categorical Logic. To appear in the Handbook of Logic in Computer Science, Abramsky S., Gabbay, D. M. and Maibaum T. S. E. (eds), Oxford University Press. *Technical Report No. 367, University of Cambridge Computer Laboratory,*

[16] Plotkin G.D.:1981, Post-graduate lecture notes in advanced domain theory (incorporating the "Pisa Notes"), *Dept. of Computer Science, Univ. of Edinburgh.*