LNL-FPC: THE LINEAR/NON-LINEAR FIXPOINT CALCULUS [*]

BERT LINDENHOVIUS [a], MICHAEL MISLOVE [a], AND VLADIMIR ZAMDZHIEV [b]

[a] Tulane University, New Orleans, USA

[b] Université de Lorraine, CNRS, Inria, LORIA, F 54000 Nancy, France

ABSTRACT. We describe a type system with mixed linear and non-linear recursive types called LNL-FPC (the linear/non-linear fixpoint calculus). The type system supports linear typing, which enhances the safety properties of programs, but also supports non-linear typing as well, which makes the type system more convenient for programming. Just as in FPC, we show that LNL-FPC supports type-level recursion, which in turn induces term-level recursion. We also provide sound and computationally adequate categorical models for LNL-FPC that describe the categorical structure of the substructural operations of Intuitionistic Linear Logic at all non-linear types, including the recursive ones. In order to do so, we describe a new technique for solving recursive domain equations within cartesian categories by constructing the solutions over pre-embeddings. The type system also enjoys implicit weakening and contraction rules that we are able to model by identifying the canonical comonoid structure of all non-linear types. We also show that the requirements of our abstract model are reasonable by constructing a large class of concrete models that have found applications not only in classical functional programming, but also in emerging programming paradigms that incorporate linear types, such as quantum programming and circuit description programming languages.

1. INTRODUCTION

The *Fixpoint Calculus* (FPC) is a type system that has been extensively studied as a foundation for functional programming languages with recursive types. Originally proposed by Plotkin [Plo85], FPC was the focus of Fiore's celebrated PhD thesis [Fio94]. The seminal paper [FP94] gives a summary account of how axiomatic domain theory can be used to characterize sound and computationally adequate denotational models of FPC.

Girard's introduction of *linear logic* [Gir87] initiated a parallel line of research into the logics underpinning functional programming languages [Abr93], focusing on the analysis of intuitionistic logic in terms of how hypotheses are consumed as resources for proofs. Attempts to bring linear types into functional programming languages soon followed [Wad90]. Recently, there has been a concrete proposal for extending Haskell with linear types [BBN+17], and new languages are proposed that provide both linear and non-linear types with convenience for programming a design goal [TP11, Mor16]. One of the main benefits of linear types

---

is the enhanced safety properties of programs, which results from the more fine-grained control of resources, e.g., safe in-place updates of mutable data and safe access control to external resources such as files, sockets, etc. (see [BBN⁺17]). Linear types also appear in concurrent settings where session types can be used to model the π-calculus [CPT16]; in quantum programming languages where they are used to ensure error-freeness by enforcing compliance with the laws of quantum mechanics [PSV14]; and in circuit description programming languages where they are used to ensure that wires do not split and do not remain unconnected [RS18, LMZ18].

   In this paper we present a foundational treatment of mixed linear and non-linear recursive types. We formulate a denotational semantics that shows how solutions to a large class of domain equations may be interpreted in both a cartesian category and a linear one and that the solutions are strongly related.

Overview and summary of results. To illustrate our results, we present a type system called *LNL-FPC* (the linear/non-linear fixpoint calculus). The syntax of our language (§2) can roughly be understood as adding linear features to FPC, or alternatively, as adding recursive types to a linear/non-linear lambda calculus (such as DILL [Bar96] or LNL [Ben95]). More precisely, it is an extension with recursive types of the circuit-free fragment of Proto-Quipper-M [RS17] that is referred to as the CLNL calculus in [LMZ18]. The type system has implicit weakening and contraction rules: non-linear variables are automatically copied and discarded whenever necessary by the language (without requiring user input). The non-linear terms, types and contexts form *subsets* of the terms, types and contexts of LNL-FPC. There is no strict separation between linear and non-linear primitives in LNL-FPC. Instead, we view the non-linear primitives as having the additional property of being compatible with respect to the substructural operations of contraction and weakening (copying and discarding), and the remaining primitives are necessarily treated linearly by the type system. This leads to greater convenience for programming, because our type system reduces the need for applying promotion (lifting) and dereliction (forcing) operations, essentially only to linear function types. We also equip LNL-FPC with a call-by-value big-step operational semantics and show that type-level recursion induces term-level recursion (§3), thus recreating a well-known result from FPC.

   The primary difficulties in designing LNL-FPC are on the denotational side. Our categorical model (§5) is a **CPO**-enriched linear/non-linear (LNL) model [Ben95] that has suitable ω-colimits. It is given by a **CPO**-enriched symmetric monoidal adjunction

$$\mathbf{C} \xrightleftharpoons[G]{\overset{F}{\underset{\perp}{\longrightarrow}}} \mathbf{L}$$

. We show that the category $\mathbf{L}$ (representing the linear world), is **CPO**-algebraically compact in the sense of [FP94] and thus one can solve recursive domain equations in $\mathbf{L}$ by constructing limits and colimits over embedding-projection pairs using the famous limit-colimit coincidence theorem [SP82]. The same paper shows any mixed-variance **CPO**-functor $T : \mathbf{L}^{\mathrm{op}} \times \mathbf{L} \to \mathbf{L}$ may be seen as a *covariant* functor $T_e : \mathbf{L}_e \times \mathbf{L}_e \to \mathbf{L}_e$ on the subcategory $\mathbf{L}_e$ of embeddings and we use this as a basis for the interpretation of types.

   However, since we work in a mixed linear/non-linear setting, we also have to explain how to solve certain recursive domain equations involving mixed-variance functors within the category $\mathbf{C}$, which is a more challenging problem. We do so by *reflecting* the solutions from $\mathbf{L}_e$ via the left adjoint F into the subcategory of *pre-embeddings* $\mathbf{C}_{pe}$, where a pre-embedding $f \in \mathbf{C}$ is a morphism, s.t. Ff is an embedding in $\mathbf{L}_e$. Unlike the subcategory

of embeddings $\mathbf{C}_e$, the subcategory of pre-embeddings $\mathbf{C}_{pe}$ has sufficient structure for constructing (parameterised) initial algebras, which moreover satisfy important coherence properties with respect to the (parameterised) initial algebras constructed in $\mathbf{L}_e$ (§4).

Then, the (standard) interpretation of an (arbitrary) type $\Theta \vdash A$ is a covariant functor denoted $[\![\Theta \vdash A]\!] : \mathbf{L}_e^{|\Theta|} \to \mathbf{L}_e$. A non-linear type $\Theta \vdash P$ admits an *additional non-linear interpretation* as a covariant functor $(\!|\Theta \vdash P|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{C}_{pe}$ that is strongly related to its standard interpretation via a natural isomorphism

$$\alpha^{\Theta \vdash P} : [\![\Theta \vdash P]\!] \circ F_{pe}^{\times |\Theta|} \cong F_{pe} \circ (\!|\Theta \vdash P|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{L}_e,$$

where F_{pe} is the restriction of F to $\mathbf{C}_{pe}$. By exploiting this natural isomorphism and the strong coherence properties that it enjoys, we provide a coherent interpretation of the substructural operations of ILL at all *non-linear* types, including the recursive ones. This then allows us to characterise the canonical comonoid structure of non-linear (recursive) types and we prove our semantics sound (§6). We emphasize that even though our recursive type expressions allow use of $\multimap$, the reason we can interpret these types as *covariant* functors is because we have identified suitable well-behaved subcategories of $\mathbf{C}$ and $\mathbf{L}$ in our model.

We also show that the requirements of our abstract model are reasonable by constructing a large class of concrete models that have been used in different programming paradigms ranging from classical to quantum (§5.4). In addition, we present a computational adequacy result for a (non-empty) class of models that satisfy certain additional conditions (§7).

Our semantic contribution can be split into two main parts. The first one is the two-tier semantics (one linear and one non-linear), together with the coherence properties relating them, that show how to rigorously construct the comonoids we need for non-linear types in the linear category. However, in order to achieve this in the presence of recursive types, our second main contribution is fundamental. It shows how the solutions to recursive domain equations on the linear side, which are constructed over embeddings (and that are well-understood), can be reflected onto the cartesian side in a coherent way by constructing them over pre-embeddings (which is novel).

Our results presented here also carry over to two-judgement calculi, such as LNL [Ben95, BW96], and it is not hard to see that our two-tier semantics presented here allows one to extend such two-judgement calculi with recursive types on both the linear side and the non-linear side. More specifically, if we adopt this view, then our semantics allows for a full range of recursive types on the linear side and for all recursive types on the non-linear side that do not involve non-linear function space (but which do allow linear function space followed by the right-adjoint type constructor, making the composite type non-linear).

Publication History. This article is an extended version of the ICFP paper [LMZ19]. Compared to that version, extensive additions and improvements have been made, including:
- A more general notion of model where we no longer assume $\mathbf{C} = \mathbf{CPO}$.
- Adding an entire subsection devoted to showing that the formal approximation relations (needed for the adequacy proof) exist (§7.1).
- Improving the presentation of parameterised initial algebras (§4).
- Adding omitted proofs and lemmas throughout the entire paper.
- Adding an example program for the factorial function (Example 2.9) that illustrates the presence of non-linear types other than those of the form $!A$.

2. Syntax of LNL-FPC

We begin by describing the syntax of LNL-FPC. The type-level syntax is very similar to that of FPC [AF96] and the term-level syntax can be seen as an extension with recursive types of either the circuit-free fragment of Proto-Quipper-M [RS17], or the CLNL calculus in [LMZ18, LMZ20].

2.1. **LNL-FPC Types.** We use X, Y, Z to range over *type variables*, which are needed in order to form recursive types. We use Θ to range over *type contexts* (see Figure 1). A type context $\Theta = X_1, \ldots, X_n$ is simply a finite list of type variables. A type context Θ is *well-formed*, denoted $\vdash \Theta$, if it can be derived from the rules:

$$\frac{}{\vdash \cdot} \qquad \frac{\vdash \Theta}{\vdash \Theta, X} \, X \notin \Theta,$$

that is, if all the variables of Θ are distinct.

We use A, B, C to range over (arbitrary) types of our language (see Figure 1 for their grammar). A type A is well-formed in type context Θ, denoted $\Theta \vdash A$, if the judgement can be derived according to the following rules:

$$\frac{\vdash \Theta}{\Theta \vdash \Theta_i} \, 1 \leq i \leq |\Theta| \qquad \frac{\Theta \vdash A}{\Theta \vdash !A} \qquad \frac{\Theta \vdash A \qquad \Theta \vdash B}{\Theta \vdash A \star B} \, \star \in \{+, \otimes, \multimap\} \qquad \frac{\Theta, X \vdash A}{\Theta \vdash \mu X.A}$$

Note that if $\Theta \vdash A$ is derivable, then so is $\vdash \Theta$. A well-formed type A is *closed* if $\cdot \vdash A$.

The *non-linear types* form a subset of our types and we use P, R to range over them (see Figure 1 for their grammar). Of course, they are governed by the same formation rules as those for (arbitrary) types. A type that is not non-linear is called *linear*. Note that we do not provide a separate grammar for linear types. This is because we will mostly be working with arbitrary types (which we do not assume we may copy/discard/promote) and with non-linear types (which we may always copy/discard/promote). In particular, it is possible for a non-linear type to be treated as arbitrary, but not vice versa. Thus, when we write $\Theta \vdash A$, the type A may or may not be non-linear, but when we write $\Theta \vdash P$, then P can be only non-linear. In particular, we do not introduce specific notation for linear types.

Example 2.1. We list some important closed types that are definable in LNL-FPC. The *empty* type is $0 \equiv \mu X.X$, which is non-linear. Another non-linear type is the *unit* type, which is defined by $I \equiv !(0 \multimap 0)$. The type of natural numbers is $\mathtt{Nat} \equiv \mu X.I + X$, which is also non-linear. Given a closed type A, then lists of type A are defined by $\mathtt{List}\ A \equiv \mu X.I + A \otimes X$. The type $\mathtt{List}\ A$ is non-linear iff A is non-linear. *Lazy* datatypes can be defined by making use of the ! and the $\multimap$ connectives. For instance, streams of type A can be defined by $\mathtt{Stream}\ A \equiv \mu X.A \otimes !X$, which is a non-linear type iff A is non-linear. The ability to form recursive types using both $\multimap$ and ! is a powerful feature and it allows us to *derive* a term-level recursion operator (see §2.3). Notice that we do not allow non-linear function space $A \to B$, but by using both $\multimap$ and ! we may still define recursion for non-linear functions.

For any types A and B and type variable X, we denote with $A[B/X]$ the type where all free occurrences of X in A are replaced by B (which is defined in the standard way).

Lemma 2.2. *If $\Theta, X \vdash A$ and $\Theta \vdash B$, then $\Theta \vdash A[B/X]$. Moreover, if A and B are non-linear, then so is $A[B/X]$.*

Remark 2.3. Like FPC, our language allows *nested type recursion* (e.g. $\mu X.\mu Y.I + (X \otimes Y)$).

Type variables	X, Y, Z		
Term variables	x, y, z		
Types	A, B, C	$::=$	$X \mid A + B \mid A \otimes B \mid A \multimap B \mid {!}A \mid \mu X.A$
Non-linear types	P, R	$::=$	$X \mid P + R \mid P \otimes R \mid {!}A \mid \mu X.P$
Type contexts	Θ	$::=$	$X_1, X_2, \ldots, X_n$
Term contexts	Γ, Σ	$::=$	$x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n$
Non-linear term contexts	Φ	$::=$	$x_1 : P_1, x_2 : P_2, \ldots, x_n : P_n$
Terms	m, n, p	$::=$	$x \mid \mathtt{left}_{A,B}\, m \mid \mathtt{right}_{A,B}\, m$
			$\mid \mathtt{case}\ m\ \mathtt{of}\ \{\mathtt{left}\ x \to n\ \mathtt{right}\ y \to p\}$
			$\mid \langle m, n \rangle \mid \mathtt{let}\ \langle x, y \rangle = m\ \mathtt{in}\ n \mid \lambda x^A.m \mid mn$
			$\mid \mathtt{lift}\ m \mid \mathtt{force}\ m \mid \mathtt{fold}_{\mu X.A}\, m \mid \mathtt{unfold}\ m$
Values	v, w	$::=$	$x \mid \mathtt{left}_{A,B}\, v \mid \mathtt{right}_{A,B}\, v \mid \langle v, w \rangle \mid \lambda x^A.m$
			$\mid \mathtt{lift}\ m \mid \mathtt{fold}_{\mu X.A}\, v$

Figure 1: Syntax of the LNL-FPC Calculus.

2.2. **LNL-FPC Terms.** We use x, y, z to range over *term variables*. We use Γ, Σ to range over (arbitrary) *term contexts*. A term context is a list of term variables with types, written as $\Gamma = x_1 : A_1, \ldots, x_n : A_n$. A term context is *well-formed* in type context Θ, denoted $\Theta \vdash \Gamma$, if the judgement can be derived from the rules:

$$\frac{\vdash \Theta}{\Theta \vdash \cdot} \qquad \frac{\Theta \vdash \Gamma \qquad \Theta \vdash A}{\Theta \vdash \Gamma, x : A}\ x \notin \Gamma,$$

that is, if Γ is a list of distinct variables with well-formed types. A *non-linear term context* is a term context whose types are all non-linear. We use Φ to range over non-linear term contexts. Just as with types, we do not introduce specific notation for purely linear contexts.

The terms and values of our language are defined in Figure 1. A *term judgement* has the form $\Theta; \Gamma \vdash m : A$ and indicates that m is a well-formed term of type A in type context Θ and term context Γ. The formation rules are shown in Figure 2, under the condition that the Γ and Σ contexts do not have any variables in common (one can deduce $\Phi \cap \Gamma = \varnothing = \Phi \cap \Sigma$). The formation rules for $\mathtt{fold}$ and $\mathtt{unfold}$ are the same as in FPC (cf. [AF96]). Observe that if $\Theta; \Gamma \vdash m : A$, then also $\Theta \vdash \Gamma$ and $\Theta \vdash A$. Note that there is only one kind of term context and we do not have explicit notation to separate the linear variables from the non-linear ones. Thus, when we write $\Theta; \Gamma \vdash m : A$, then the context Γ could contain both linear and non-linear variables. However, when we write $\Theta; \Phi \vdash m : A$, then the context Φ contains only non-linear variables. The type system enforces that a linear variable is used exactly once, whereas a non-linear variable may be used any number of times, including zero.

Type assignment to terms is unique, but derivations of term judgements in LNL-FPC are in general not unique, because non-linear variables may be part of an arbitrary context Γ, that is, our type system allows for non-linear variables to be treated as if they were linear (but not vice versa). For example, if $\Theta \vdash P_1$ and $\Theta \vdash P_2$ are non-linear types, then:

$$\frac{\Theta; x : P_1 \vdash x : P_1 \qquad \Theta; y : P_2 \vdash y : P_2}{\Theta; x : P_1, y : P_2 \vdash \langle x, y \rangle : P_1 \otimes P_2} \qquad \frac{\Theta; x : P_1 \vdash x : P_1 \qquad \Theta; x : P_1, y : P_2 \vdash y : P_2}{\Theta; x : P_1, y : P_2 \vdash \langle x, y \rangle : P_1 \otimes P_2}$$

are two different derivations of the same judgement. In both of these derivations, the variable y is treated as if it were linear and it is propagated up into only one judgement. The variable x we treat non-linearly in one case (propagated up twice) and in other case we treat it as if

$$\frac{\Theta \vdash \Phi, x : A}{\Theta; \Phi, x : A \vdash x : A} \qquad \frac{\Theta; \Gamma \vdash m : A \quad \Theta \vdash B}{\Theta; \Gamma \vdash \texttt{left}_{A,B}\, m : A + B} \qquad \frac{\Theta; \Gamma \vdash m : B \quad \Theta \vdash A}{\Theta; \Gamma \vdash \texttt{right}_{A,B}\, m : A + B}$$

$$\frac{\Theta; \Phi, \Gamma \vdash m : A + B \quad \Theta; \Phi, \Sigma, x : A \vdash n : C \quad \Theta; \Phi, \Sigma, y : B \vdash p : C}{\Theta; \Phi, \Gamma, \Sigma \vdash \texttt{case } m \texttt{ of } \{\texttt{left } x \to n \mid \texttt{right } y \to p\} : C}$$

$$\frac{\Theta; \Phi, \Gamma \vdash m : A \quad \Theta; \Phi, \Sigma \vdash n : B}{\Theta; \Phi, \Gamma, \Sigma \vdash \langle m, n \rangle : A \otimes B} \qquad \frac{\Theta; \Phi, \Gamma \vdash m : A \otimes B \quad \Theta; \Phi, \Sigma, x : A, y : B \vdash n : C}{\Theta; \Phi, \Gamma, \Sigma \vdash \texttt{let } \langle x, y \rangle = m \texttt{ in } n : C}$$

$$\frac{\Theta; \Gamma, x : A \vdash m : B}{\Theta; \Gamma \vdash \lambda x^A.m : A \multimap B} \qquad \frac{\Theta; \Phi, \Gamma \vdash m : A \multimap B \quad \Theta; \Phi, \Sigma \vdash n : A}{\Theta; \Phi, \Gamma, \Sigma \vdash mn : B} \qquad \frac{\Theta; \Phi \vdash m : A}{\Theta; \Phi \vdash \texttt{lift } m : {!}A}$$

$$\frac{\Theta; \Gamma \vdash m : {!}A}{\Theta; \Gamma \vdash \texttt{force } m : A} \qquad \frac{\Theta; \Gamma \vdash m : A[\mu X.A/X] \quad \Theta, X \vdash A}{\Theta; \Gamma \vdash \texttt{fold}_{\mu X.A}\, m : \mu X.A} \qquad \frac{\Theta; \Gamma \vdash m : \mu X.A}{\Theta; \Gamma \vdash \texttt{unfold } m : A[\mu X.A/X]}$$

where $\Gamma \cap \Sigma = \varnothing$.

Figure 2: Formation rules for LNL-FPC terms.

it were linear (propagated up once). This is because non-linear variables are allowed to be part of arbitrary contexts (see formation rule for pairing).

This non-uniqueness is a result of the design choice to have only one kind of pairing, one kind of conditional branching, etc. (instead of having separate linear and non-linear ones), which we think results in a more convenient syntax for programming. We note that the interpretation of any two derivations of the same judgement are equal (see Theorem 6.12).

Example 2.4. The term $\cdot; \cdot \vdash \lambda x.\langle x, x \rangle : A \multimap A \otimes A$ is well-formed iff A is a non-linear type. Indeed, if A is a linear type, then the term $\cdot; x : A \vdash \langle x, x \rangle : A \otimes A$ is not well-formed, because the variable x is part of a linear context and contraction is not admissible.

A *term of closed type* is a term m, such that $\cdot\,; \Gamma \vdash m : A$ for some type A and context Γ. In such a situation we simply write $\Gamma \vdash m : A$. Naturally, we are primarily interested in these terms (observe that the term formation rules are invariant with respect to the type context). A *program* is a term p, such that $\cdot\,; \cdot \vdash p : A$ for some type A and we simply write $p : A$ to indicate this.

Example 2.5. We list some important programs. We define $* \equiv \texttt{lift } \lambda x^0.x : I$, which is the canonical value of unit type. The zero natural number is defined by the program $\texttt{zero} \equiv \texttt{fold}_{\texttt{Nat}} \texttt{ left}_{I,\texttt{Nat}} * : \texttt{Nat}$. The successor function can be defined by the program $\texttt{succ} \equiv \lambda n^{\texttt{Nat}}.\texttt{fold}_{\texttt{Nat}} \texttt{ right}_{I,\texttt{Nat}} n : \texttt{Nat} \multimap \texttt{Nat}$.

Given terms m, n and a variable x, we denote with $m[n/x]$ the term obtained from m by replacing all free occurrences of x with n (which is defined in the standard way).

Lemma 2.6 (Substitution). *If* $\Theta; \Phi, \Gamma, x : A \vdash m : B$ *and* $\Theta; \Phi, \Sigma \vdash n : A$ *and* $\Gamma \cap \Sigma = \varnothing$, *then* $\Theta; \Phi, \Gamma, \Sigma \vdash m[n/x] : B$.

We say a value $\Theta; \Gamma \vdash v : P$ is *non-linear* whenever the type P is non-linear.

Lemma 2.7. *If* $\Theta; \Gamma \vdash v : P$ *is a non-linear value, then* Γ *is also non-linear.*

2.3. **Term Recursion in LNL-FPC.** Recall that in FPC, general recursion on terms may be implemented using the `fold` and `unfold` terms. The same is also true for LNL-FPC. Moreover, the derived term for recursion has exactly the same syntax as the one in [LMZ18], where the authors showed how to extend a mixed linear/non-linear type system with recursion. In the next section we show it has the same operational behaviour as well. We also note that our recursion term is very similar to one of the recursive terms in [Epp03, Chapter 8].

Theorem 2.8. *The rule* $\quad \dfrac{\Theta; \Phi, z :! A \vdash m : A}{\Theta; \Phi \vdash \texttt{rec } z^{!A}.m : A} \quad$ *is derivable in LNL-FPC, where Φ is non-linear,*

$$\texttt{rec } z^{!A}.m \equiv (\texttt{unfold force } \alpha_m^z)\alpha_m^z \qquad \text{and}$$

$$\alpha_m^z \equiv \texttt{lift fold } \lambda x^{!\mu X.(!X \multimap A)}.(\lambda z^{!A}.m)(\texttt{lift }(\texttt{unfold force } x)x),$$

such that $X \notin \Theta$ and $x \notin \Phi$.

Proof. Since $\Theta; \Phi, z : !A \vdash m : A$, then $\Theta \vdash A$ (see §2.2) and since $X \notin \Theta$, then

$$\Theta, X \vdash !X \multimap A \quad \text{and} \quad \Theta \vdash x : !\mu X.(!X \multimap A) \tag{2.1}$$

We will first show that:

$$\Theta; x : !\mu X.(!X \multimap A) \vdash \texttt{lift }(\texttt{unfold force } x)x : !A \tag{2.2}$$

For brevity, we write $\mathcal{R} \equiv \mu X.(!X \multimap A)$. Then, we have:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\overline{\Theta \vdash x : !\mathcal{R}}^{(2.1)}}{\Theta; x : !\mathcal{R} \vdash x : !\mathcal{R}}}{\Theta; x : !\mathcal{R} \vdash \texttt{force } x : \mathcal{R}}}{\Theta; x : !\mathcal{R} \vdash \texttt{unfold force } x : (!\mathcal{R}) \multimap A} \qquad \cfrac{\overline{\Theta \vdash x : !\mathcal{R}}^{(2.1)}}{\Theta; x : !\mathcal{R} \vdash x : !\mathcal{R}}}{\Theta; x : !\mathcal{R} \vdash (\texttt{unfold force } x)x : A}}{\Theta; x : !\mathcal{R} \vdash \texttt{lift }(\texttt{unfold force } x)x : !A}$$

Next, we show that:

$$\cfrac{\Theta; \Phi, z : !A \vdash m : A}{\Theta; \Phi \vdash \alpha_m^z : !\mathcal{R}} \tag{2.3}$$

Indeed, we have:

$$\cfrac{\cfrac{\cfrac{\cfrac{\Theta; \Phi, z : !A \vdash m : A}{\Theta; \Phi \vdash (\lambda z^{!A}.m) : !A \multimap A} \quad \cfrac{}{\Theta; x : !\mathcal{R} \vdash \texttt{lift }(\texttt{unfold force } x)x : !A}^{(2.2)}}{\Theta; \Phi, x : !\mathcal{R} \vdash (\lambda z^{!A}.m)(\texttt{lift }(\texttt{unfold force } x)x) : A}}{\Theta; \Phi \vdash \lambda x^{!\mathcal{R}}.(\lambda z^{!A}.m)(\texttt{lift }(\texttt{unfold force } x)x) : (!\mathcal{R}) \multimap A} \quad \cfrac{}{\Theta, X \vdash !X \multimap A}^{(2.1)}}{\Theta; \Phi \vdash \texttt{fold } \lambda x^{!\mathcal{R}}.(\lambda z^{!A}.m)(\texttt{lift }(\texttt{unfold force } x)x) : \mathcal{R}}}{\Theta; \Phi \vdash \texttt{lift fold } \lambda x^{!\mathcal{R}}.(\lambda z^{!A}.m)(\texttt{lift }(\texttt{unfold force } x)x) : !\mathcal{R}}$$

Finally:

$$\cfrac{\cfrac{\cfrac{\cfrac{\Theta; \Phi, z : !A \vdash m : A}{\Theta; \Phi \vdash \alpha_m^z : !\mathcal{R}}^{(2.3)}}{\Theta; \Phi \vdash \texttt{force } \alpha_m^z : \mathcal{R}}}{\Theta; \Phi \vdash \texttt{unfold force } \alpha_m^z : (!\mathcal{R}) \multimap A} \quad \cfrac{\Theta; \Phi, z : !A \vdash m : A}{\Theta; \Phi \vdash \alpha_m^z : !\mathcal{R}}^{(2.3)}}{\Theta; \Phi \vdash (\texttt{unfold force } \alpha_m^z)\alpha_m^z : A}$$

$$\frac{}{x \Downarrow x} \qquad \frac{m \Downarrow v}{\texttt{left } m \Downarrow \texttt{left } v} \qquad \frac{m \Downarrow v}{\texttt{right } m \Downarrow \texttt{right } v}$$

$$\frac{m \Downarrow \texttt{left } v \quad n[v/x] \Downarrow w}{\texttt{case } m \texttt{ of } \{\texttt{left } x \to n \mid \texttt{right } y \to p\} \Downarrow w} \qquad \frac{m \Downarrow v \quad n \Downarrow w}{\langle m, n \rangle \Downarrow \langle v, w \rangle}$$

$$\frac{m \Downarrow \texttt{right } v \quad p[v/y] \Downarrow w}{\texttt{case } m \texttt{ of } \{\texttt{left } x \to n \mid \texttt{right } y \to p\} \Downarrow w} \qquad \frac{m \Downarrow \langle v, v' \rangle \quad n[v/x, v'/y] \Downarrow w}{\texttt{let } \langle x, y \rangle = m \texttt{ in } n \Downarrow w}$$

$$\frac{}{\lambda x.m \Downarrow \lambda x.m} \qquad \frac{m \Downarrow \lambda x.m' \quad n \Downarrow v \quad m'[v/x] \Downarrow w}{mn \Downarrow w}$$

$$\frac{}{\texttt{lift } m \Downarrow \texttt{lift } m} \qquad \frac{m \Downarrow \texttt{lift } m' \quad m' \Downarrow v}{\texttt{force } m \Downarrow v} \qquad \frac{m \Downarrow v}{\texttt{fold } m \Downarrow \texttt{fold } v} \qquad \frac{m \Downarrow \texttt{fold } v}{\texttt{unfold } m \Downarrow v}$$

Figure 3: Operational semantics of the LNL-FPC calculus.

and therefore:

$$\frac{\Theta; \Phi, z : !A \vdash m : A}{\Theta; \Phi \vdash \texttt{rec } z^{!A}.m : A} \qquad\qquad \square$$

We now consider an example that highlights some of the advantages of the ambivalence of our typing system.

Example 2.9. The factorial function on natural numbers may be defined by:

```
rec fact. λ n.
case unfold n of
left u -> succ zero
right n' -> mult(n, (force fact) n')
```

where `mult` is the multiplication function (which also can be easily defined). Note that copying and discarding of the non-linear variables is implicit. In more traditional linear typing systems, the variables for which contraction and weakening (copying and discarding) are admissible are those of type $!A$. But, notice that the variable `n` above is of type `Nat`, which is *not* of the form $!A$. We are able to implicitly copy it above, because we have extended the non-linear types to include more types than just those of the form $!A$. This means that `lift` and `force` are mostly needed for promotion and dereliction of terms of function types (as can be seen above).

3. Operational Semantics

The operational semantics of LNL-FPC is standard. We use a big-step call-by-value evaluation relation whose rules are shown in Figure 3.

As usual, the *values* are terms v such that $v \Downarrow v$ (see Figure 1). The evaluation relation $(- \Downarrow -)$ is, in fact, a *partial* function from terms to values. Of course, it is not a total function, because the language supports a general recursion operator, as we show next.

Theorem 3.1. *The evaluation rule* $\dfrac{m[\texttt{lift rec } z^{!A}.m \ / \ z] \Downarrow v}{\texttt{rec } z^{!A}.m \Downarrow v}$ *is derivable within LNL-FPC, where* $\texttt{rec } z^{!A}.m$ *is defined as in Theorem 2.8.*

Proof. First, we introduce the term
$$m' \equiv (\lambda z^{!A}.m)(\texttt{lift } (\texttt{unfold force } x)x),$$
so that
$$\alpha_m^z = \texttt{lift fold } \lambda x^{!\mu X.(!X \multimap A)}.m',$$
and
$$m'[\alpha_m^z \,/\, x] = (\lambda z^{!A}.m)(\texttt{lift } (\texttt{unfold force } \alpha_m^z)\alpha_m^z) = (\lambda z^{!A}.m)(\texttt{lift rec } z^{!A}.m).$$
Then
$$\frac{\overline{\lambda z^{!A}.m \Downarrow \lambda z^{!A}.m} \quad \overline{\texttt{lift rec } z^{!A}.m \Downarrow \texttt{lift rec } z^{!A}.m} \quad m[\texttt{lift rec } z^{!A}.m \,/\, z] \Downarrow v}{(\lambda z^{!A}.m)(\texttt{lift rec } z^{!A}.m) \Downarrow v}$$
hence
$$\frac{m[\texttt{lift rec } z^{!A}.m \,/\, z] \Downarrow v}{m'[\alpha_m^z \,/\, x] \Downarrow v} \tag{3.1}$$
Moreover,
$$\frac{\overline{\texttt{lift fold } \lambda x.m' \Downarrow \texttt{lift fold } \lambda x.m'} \quad \dfrac{\overline{\lambda x.m' \Downarrow \lambda x.m'}}{\texttt{fold } \lambda x.m' \Downarrow \texttt{fold } \lambda x.m'}}{\dfrac{\texttt{force lift fold } \lambda x.m' \Downarrow \texttt{fold } \lambda x.m'}{\texttt{unfold force lift fold } \lambda x.m' \Downarrow \lambda x.m'}}$$
hence we obtain
$$\frac{}{\texttt{unfold force } \alpha_m^z \Downarrow \lambda x.m'} \tag{3.2}$$
Then
$$\frac{\dfrac{}{\texttt{unfold force } \alpha_m^z \Downarrow \lambda x.m'} \,(3.2) \quad \dfrac{}{\alpha_m^z \Downarrow \alpha_m^z} \quad \dfrac{m[\texttt{lift rec } z^{!A}.m \,/\, z] \Downarrow v}{m'[\alpha_m^z \,/\, x] \Downarrow v} \,(3.1)}{(\texttt{unfold force } \alpha_m^z)\alpha_m^z \Downarrow v,}$$
and therefore:
$$\frac{m[\texttt{lift rec } z^{!A}.m \,/\, z] \Downarrow v}{\texttt{rec } z^{!A}.m \Downarrow v} \qquad \square$$

This theorem shows that our *derived* general recursion operator is exactly the same as the one in [LMZ18], where it was added as an axiom.

Notation 3.2. A term m is said to *terminate*, denoted by $m \Downarrow$, if there exists a value v, such that $m \Downarrow v$.

The simplest non-terminating program of type A is given by $\texttt{rec } z^{!A}. \texttt{ force } z : A$.

Example 3.3. The constant stream of zero natural numbers can be defined by
$$\texttt{const}_0 \equiv \texttt{rec } s^{!(\texttt{Stream Nat})}\texttt{fold}_{\texttt{Stream Nat}} \langle \texttt{zero}, s \rangle : \texttt{Stream Nat}.$$

Remark 3.4. Streams of type P should not be defined as $\mu X.P \otimes X$, because there are no *closed values* of this type. This is a consequence of Theorem 7.10 and the fact that $[\![\mu X.P \otimes X]\!] = 0$ (see §6).

Theorem 3.5 (Subject reduction). *If $\Theta; \Gamma \vdash m : A$ and $m \Downarrow v$, then $\Theta; \Gamma \vdash v : A$.*

Assumption 3.6. *Throughout the remainder of the paper, we implicitly assume that all types, contexts and terms are well-formed.*

4. ω-categories and (Parameterised) Initial Algebras

In this section we recall the theory of ω-categories introduced in [LS81] and develop new results of our own. In §4.1, we introduce some notation for operations on natural transformations that we use throughout the paper. In §4.2, we recall how initial algebras are constructed in ω-categories. In §4.3, we show how to construct parameterised initial algebras in ω-categories that we use to model recursive types that may potentially be defined by nested recursion. In §4.4, we present new results that show how (parameterised) initial algebras of functors acting on different categories may be related to one another, provided there exist suitable mediating functors between the two categories. The type-level semantics makes heavy use of this relationship in order to present *coherent* non-linear type interpretations that are strongly related to the standard type interpretations.

4.1. **Operations on natural transformations.** Given natural transformations τ and σ and a functor F, we denote: vertical composition by $\sigma \circ \tau$; horizontal composition by $\sigma * \tau$; whiskering by $F\tau$ and τF, whenever these operations are admissible.

If $\tau : T \Rightarrow H : \mathbf{C} \to \mathbf{C}$ is a natural transformation between two endofunctors, then we define $\tau^{*n} := \tau * \cdots * \tau : T^n \Rightarrow H^n : \mathbf{C} \to \mathbf{C}$ to be the n-fold horizontal composition of τ with itself (if $n = 0$, then τ^{*0} is the identity id : Id $\Rightarrow$ Id : $\mathbf{C} \to \mathbf{C}$).

If $\tau : T \Rightarrow T' : \mathbf{A} \to \mathbf{B}$ and $\sigma : H \Rightarrow H' : \mathbf{C} \to \mathbf{D}$ are natural transformations, we define a natural transformation $\tau \times \sigma : T \times H \Rightarrow T' \times H' : \mathbf{A} \times \mathbf{C} \to \mathbf{B} \times \mathbf{D}$ via the assignment $(\tau \times \sigma)_{(A,C)} := (\tau_A, \sigma_C)$.

If $\tau : T \Rightarrow T' : \mathbf{A} \to \mathbf{B}$ and $\sigma : H \Rightarrow H' : \mathbf{A} \to \mathbf{C}$ are natural transformations, we define a natural transformation $\langle \tau, \sigma \rangle : \langle T, H \rangle \Rightarrow \langle T', H' \rangle : \mathbf{A} \to \mathbf{B} \times \mathbf{C}$ by $\langle \tau, \sigma \rangle_A := (\tau_A, \sigma_A)$.

We denote with ω the poset of natural numbers when viewed as a category. A functor $D : \omega \to \mathbf{C}$ is then an ω-diagram. We let $[\mathbf{A}, \mathbf{B}]$ denote the functor category from $\mathbf{A}$ to $\mathbf{B}$. Given a functor $M : \mathbf{A} \to \mathbf{B}$, we define a functor $M \triangleright - : [\omega, \mathbf{A}] \to [\omega, \mathbf{B}]$ by:

$$M \triangleright D = M \circ D \qquad \text{and} \qquad M \triangleright \tau = M\tau.$$

So, the functor $M \triangleright -$ is just whiskering with M.

4.2. **Initial algebras in ω-categories.** We now recall some definitions and facts about ω-categories and ω-functors that are stated in [LS81]. A functor $F : \mathbf{A} \to \mathbf{C}$ is an ω-*functor* if F preserves all existing colimits of ω-diagrams. If, in addition, $\mathbf{A}$ has an initial object and F preserves it, then we say that F is a *strict* ω-functor. Of course, ω-functors are closed under composition and pairing, that is, if F and G are ω-functors, then so are $F \circ G$ and $\langle F, G \rangle$ whenever composition and pairing are admissible.

A category $\mathbf{C}$ is an ω-*category* if it has an initial object and all ω-colimits. We denote with $[\mathbf{A} \to_\omega \mathbf{C}]$ the full subcategory of $[\mathbf{A}, \mathbf{C}]$ consisting of ω-functors. If $\mathbf{C}$ is an ω-category, then so are $[\mathbf{A}, \mathbf{C}]$ and $[\mathbf{A} \to_\omega \mathbf{C}]$.

Definition 4.1 ([LS81]). Given an ω-category $\mathbf{C}$ with an initial object $\varnothing$, we define a functor $S : [\mathbf{C} \to_\omega \mathbf{C}] \to [\omega, \mathbf{C}]$ in the following way:

- Given an ω-functor $T : \mathbf{C} \to \mathbf{C}$, then $S(T)$ is the ω-diagram $\varnothing \xrightarrow{\iota} T\varnothing \xrightarrow{T\iota} T^2\varnothing \xrightarrow{T^2\iota} \cdots$. More specifically, $S(T)(n) := T^n\varnothing$ and $S(T)(n \le n+1) := T^n\iota$, where $\iota : \varnothing \to T\varnothing$ is the initial map.

- Given $\tau : T \Rightarrow H : \mathbf{C} \to \mathbf{C}$, then $S(\tau) : S(T) \Rightarrow S(H) : \omega \to \mathbf{C}$ is given by

$$S(\tau)_n := (\tau^{*n})_\varnothing : T^n \varnothing \to H^n \varnothing.$$

We also define a functor $Y := \mathrm{colim} \circ S : [\mathbf{C} \to_\omega \mathbf{C}] \to \mathbf{C}$, where $\mathrm{colim} : [\omega, \mathbf{C}] \to \mathbf{C}$ is the colimiting functor, which is the left-adjoint of the ω-ary diagonal functor $\Delta : \mathbf{C} \to [\omega, \mathbf{C}]$.

Theorem 4.2. [LS81] *Let $\mathbf{C}$ be an arbitrary ω-category. Then both $S : [\mathbf{C} \to_\omega \mathbf{C}] \to [\omega, \mathbf{C}]$ and $Y : [\mathbf{C} \to_\omega \mathbf{C}] \to \mathbf{C}$ are ω-functors.*

Therefore, for an ω-functor $T : \mathbf{C} \to \mathbf{C}$ on an ω-category $\mathbf{C}$, its initial sequence is given by $S(T)$ and the *carrier* of its initial algebra is given by $Y(T)$, thanks to a famous result in [Adá74]. To describe its initial algebra structure we need an additional definition.

Theorem 4.3 ([AMM18]). *Let $T : \mathbf{C} \to \mathbf{C}$ be an ω-endofunctor on an ω-category $\mathbf{C}$. We define a natural transformation $s^T : S(T) \Rightarrow T \circ S(T) : \omega \to \mathbf{C}$ by $(s^T)_n := T^n(\iota_{T\varnothing})$, where $\iota_{T\varnothing} : \varnothing \to T\varnothing$ is the initial morphism. Then one can define an isomorphism:*

$$y^T : T(Y(T)) \to Y(T)$$

$$y^T := \left(TY(T) = T\mathrm{colim}(S(T)) = \mathrm{colim}(TS(T)) \xrightarrow{(\mathrm{colim}(s^T))^{-1}} \mathrm{colim}(S(T)) = Y(T) \right)$$

and the pair $(Y(T), y^T)$ forms the initial T-algebra.

4.3. **Parameterised initial algebras.** Initial algebras can be used to model recursive types where the type recursion is done over a single type variable. In order to model recursive types defined by *nested recursion*, one has to allow recursive types to depend on several type variables. The interpretation of these more general recursive types requires a more general notion, namely *parameterised initial algebras*, which we introduce next.

Definition 4.4 (cf. [Fio94, §6.1]). Given categories $\mathbf{A}$ and $\mathbf{B}$ and a functor $T : \mathbf{A} \times \mathbf{B} \to \mathbf{B}$, a *parameterised initial algebra* for T is a pair $(T^\dagger, \phi^T)$, such that:

- $T^\dagger : \mathbf{A} \to \mathbf{B}$ is a functor;
- $\phi^T : T \circ \langle \mathrm{Id}, T^\dagger \rangle \Rightarrow T^\dagger : \mathbf{A} \to \mathbf{B}$ is a natural isomorphism;
- For every $A \in \mathrm{Ob}(\mathbf{A})$, the pair $(T^\dagger A, \phi_A^T)$ is an initial $T(A, -)$-algebra.

Remark 4.5. Notice that by trivialising the category $\mathbf{A}$, we recover the usual notion of initial algebra. Because of this, parameterised initial algebras are a more general notion.

Remark 4.6. The naturality of ϕ (4.1) in fact determines the action of $T^\dagger$ on morphisms. Indeed, for every $f : A_1 \to A_2$ in $\mathbf{A}$, $T^\dagger f$ is the unique $T(A_1, -)$-algebra morphism making (4.2) commute. To see this, notice that (4.1) and (4.2) are equivalent diagrams.

$$
\begin{array}{ccc}
T(A_1, T^\dagger A_1) & \xrightarrow{\phi_{A_1}^T} & T^\dagger A_1 \\
{\scriptstyle T(f, T^\dagger f)} \downarrow & & \downarrow {\scriptstyle T^\dagger f} \\
T(A_2, T^\dagger A_2) & \xrightarrow[\phi_{A_2}^T]{} & T^\dagger A_2
\end{array}
\tag{4.1}
$$

$$\begin{array}{ccc} T(A_1, T^\dagger A_1) & \xrightarrow{\quad\phi^T_{A_1}\quad} & T^\dagger A_1 \\ {\scriptstyle T(A_1, T^\dagger f)}\downarrow & & \downarrow{\scriptstyle T^\dagger f} \\ T(A_1, T^\dagger A_2) \xrightarrow[T(f, T^\dagger A_2)]{} T(A_2, T^\dagger A_2) & \xrightarrow[\phi^T_{A_2}]{} & T^\dagger A_2 \end{array} \qquad (4.2)$$

Next, we aim to show that the class of ω-functors on an ω-category is closed under formation of parameterised initial algebras (Theorem 4.12). In order to do so, we have to establish a few additional lemmas.

Lemma 4.7. *Let* $\mathbf{B}$ *be an* ω-*category and let* $T : \mathbf{A} \times \mathbf{B} \to \mathbf{B}$ *be an* ω-*functor. The mapping*

$$S(T(A, -)) \xRightarrow{\quad s^{T(A,-)}\quad} T(A, -) \circ S(T(A, -)) : \omega \to \mathbf{B}$$

is natural in A. *More specifically, the following diagram:*

$$\begin{array}{ccc} S(T(A, -)) & \xRightarrow{\quad s^{T(A,-)}\quad} & T(A, -) \circ S(T(A, -)) \\ {\scriptstyle S(T(f, -))}\big\Vert & & \big\Vert{\scriptstyle T(f, -) * S(T(f, -))} \\ S(T(B, -)) & \xRightarrow[\quad s^{T(B,-)}\quad]{} & T(B, -) \circ S(T(B, -)) \end{array}$$

commutes for any $f : A \to B$ *in* $\mathbf{A}$.

Proof. In Appendix A.1.　　　　　　　　　　　　　　　　　　　　　　　　$\square$

Notation 4.8. Given an ω-diagram $D : \omega \to \mathbf{A}$, we denote objects $D(n)$ by D_n, the colimit of D (if it exists) by D_ω and its colimiting cocone morphisms by $d_n : D_n \to D_\omega$.

We proceed with two simple lemmas that show that the colim $: [\omega, \mathbf{C}] \to \mathbf{C}$ functor is quite well-behaved on ω-categories and ω-functors, as one would expect.

Lemma 4.9. *Let* $T : \mathbf{A} \to \mathbf{B}$ *be an* ω-*functor between* ω-*categories* $\mathbf{A}$ *and* $\mathbf{B}$. *Assume further* $D, D' : \omega \to \mathbf{A}$ *are* ω-*diagrams and* $\tau : D \Rightarrow D'$ *a natural transformation. Then the following diagram commutes:*

$$\begin{array}{ccc} \mathrm{colim}(T \circ D) & =\!\!=\!\!=\!\!= & T(\mathrm{colim}(D)) \\ {\scriptstyle \mathrm{colim}(T\tau)}\downarrow & & \downarrow{\scriptstyle T(\mathrm{colim}(\tau))} \\ \mathrm{colim}(T \circ D') & =\!\!=\!\!=\!\!= & T(\mathrm{colim}(D')) \end{array}$$

Proof. In Appendix A.2.　　　　　　　　　　　　　　　　　　　　　　　　$\square$

Lemma 4.10. *Let* $T, H : \mathbf{A} \to \mathbf{B}$ *be* ω-*functors between* ω-*categories* $\mathbf{A}$ *and* $\mathbf{B}$, *and let* $\tau : T \Rightarrow H$ *be a natural transformation. Given an* ω-*diagram* $D : \omega \to \mathbf{A}$, *the following diagram commutes:*

$$\begin{array}{ccc}
\mathrm{colim}(T \circ D) & =\!=\!= & T(\mathrm{colim}(D)) \\
\Big\downarrow {\scriptstyle \mathrm{colim}(\tau D)} & & \Big\downarrow {\scriptstyle \tau_{\mathrm{colim}(D)}} \\
\mathrm{colim}(H \circ D) & =\!=\!= & H(\mathrm{colim}(D))
\end{array}$$

Proof. In Appendix A.3.   □

We now recall an important lemma that shows that ω-functors on ω-categories are closed under currying.

Lemma 4.11 ([LS81])**.** *Given an ω-category $\mathbf{C}$ and an ω-functor $T : \mathbf{A} \times \mathbf{B} \to \mathbf{C}$, we define a functor $\lambda B.T(-, B) : \mathbf{A} \to [\mathbf{B} \to_\omega \mathbf{C}]$ by:*

$$\lambda B.T(A, B) := T(A, -) : \mathbf{B} \to \mathbf{C}, \qquad\qquad\qquad \textit{for } A \in \mathrm{Ob}(\mathbf{A});$$
$$\lambda B.T(f, B) := T(f, -) : T(A_1, -) \Rightarrow T(A_2, -) : \mathbf{B} \to \mathbf{C}, \qquad \textit{for } f : A_1 \to A_2.$$

Then $\lambda B.T(-, B) : \mathbf{A} \to [\mathbf{B} \to_\omega \mathbf{C}]$ is an ω-functor.

The main result from this subsection follows.

Theorem 4.12. *Let $\mathbf{B}$ be an ω-category and let $T : \mathbf{A} \times \mathbf{B} \to \mathbf{B}$ be an ω-functor. Then:*
(1) *The functor $T^\dagger := Y \circ \lambda B.T(-, B) : \mathbf{A} \to \mathbf{B}$ is an ω-functor;*
(2) *There exists a natural isomorphism $\phi^T : T \circ \langle Id, T^\dagger \rangle \Rightarrow T^\dagger : \mathbf{A} \to \mathbf{B}$ given by:*

$$\phi_A^T := \left(T(A, T^\dagger A) = T(A, Y(T(A, -))) \xrightarrow{y^{T(A,-)}} Y(T(A, -)) = T^\dagger A \right);$$

(3) *The pair $(T^\dagger, \phi^T)$ is a parameterised initial algebra of T.*

Proof.
    (1) Because $T^\dagger$ is the composition of two ω-functors.
    (2) Every component of ϕ^T is an isomorphism by Theorem 4.3. It remains to show naturality. So let $f : A \to B$ be a morphism in $\mathbf{A}$. For brevity we write T_A, T_B, T_f for $T(A, -), T(B, -), T(f, -)$, respectively. By definition

$$(\phi_A^T)^{-1} = \left(T^\dagger A = \mathrm{colim}(S(T_A)) \xrightarrow{\mathrm{colim}(s^{T(A,-)})} \mathrm{colim}(T_A S(T_A)) = T_A(\mathrm{colim}(S(T_A))) = T(A, T^\dagger A) \right)$$

and naturality of ϕ^T comes down to the commutativity of the outer square of the diagram:

$$\begin{array}{ccccccc}
T^\dagger A =\!=\!= \mathrm{colim}(S(T_A)) & \xrightarrow{\mathrm{colim}(s^{T_A})} & \mathrm{colim}(T_A \circ S(T_A)) =\!= & T_A\mathrm{colim}(S(T_A)) = T(A, T^\dagger A) \\
\Big\downarrow{\scriptstyle T^\dagger f} \quad \Big\downarrow{\scriptstyle \mathrm{colim}(S(T_f))} & \Big\downarrow{\scriptstyle \mathrm{colim}(T_f * S(T_f))} & \Big\downarrow{\scriptstyle T_f \mathrm{colim}(S(T_f))} & \Big\downarrow{\scriptstyle T(f, T^\dagger f)} \\
\quad (1) & (2) & (3) & (4) \\
T^\dagger B =\!=\!= \mathrm{colim}(S(T_B)) & \xrightarrow[\mathrm{colim}(s^{T_B})]{} & \mathrm{colim}(T_B \circ S(T_B)) =\!= & T_B\mathrm{colim}(S(T_B)) = T(B, T^\dagger B)
\end{array}$$

This is indeed the case, since squares (1) and (4) commute by definition of $T^\dagger$, and diagram (2) commutes by the functoriality of colim and Lemma 4.7. By definition of horizontal

composition, we have $T_f * S(T_f) = T_f S(T_B) \circ T_A S(T_f)$. Since colim is a functor, it follows from the latter equality that the commutativity of square (3) comes down to the commutativity of the outer square of the following diagram:

$$\mathrm{colim}(T_A \circ S(T_A)) = T_A(\mathrm{colim}(S(T_A)))$$

$$\mathrm{colim}(T_A S(T_f)) \qquad\qquad T_A \mathrm{colim}(S(T_f))$$

$$\mathrm{colim}(T_A S(T_B)) = T_A(\mathrm{colim}(S(T_B)))$$

$$\mathrm{colim}(T_f S(T_B)) \qquad\qquad T_f \mathrm{colim}(S(T_B))$$

$$\mathrm{colim}(T_B \circ S(T_B)) = T_B(\mathrm{colim}(S(T_B)))$$

Here the upper square commutes by Lemma 4.9 and the lower one by Lemma 4.10.

(3) For every $A \in \mathrm{Ob}(\mathbf{A})$, we have $(T^\dagger A, \phi_A^T) = (Y(T_A), y^{T_A})$, which is the initial $T(A, -)$-algebra by Theorem 4.3. $\qquad \square$

Remark 4.13. In the situation of the above theorem, we sometimes simply write ϕ instead of ϕ^T whenever T is clear from the context. Moreover, in the special case when $\mathbf{A} = \mathbf{B}^n$, we see that both $T : \mathbf{B}^{n+1} \to \mathbf{B}$ and $T^\dagger : \mathbf{B}^n \to \mathbf{B}$ are ω-functors. In fact, in our semantic treatment, the interpretation of a type $\Theta \vdash A$ is given by an ω-functor that is of the form $H : \mathbf{B}^{|\Theta|} \to \mathbf{B}$.

We conclude the subsection by showing an important proposition for proving the type substitution lemma.

Proposition 4.14. *Let* $\mathbf{C}$ *be an* ω-*category,* $T : \mathbf{B} \times \mathbf{C} \to \mathbf{C}$ *an* ω-*functor and* $H : \mathbf{A} \to \mathbf{B}$ *an* ω-*functor. Then* $(T \circ (H \times \mathrm{Id}))^\dagger = T^\dagger \circ H : \mathbf{A} \to \mathbf{C}$.

Proof. For any object $A \in \mathrm{Ob}(\mathbf{A})$ and for any morphism $f \in \mathrm{Mor}(\mathbf{A})$, we have:

$$
\begin{aligned}
(T \circ (H \times \mathrm{Id}))^\dagger(A) &= Y(T(HA, -)) \\
&= (Y \circ \lambda B.T(-, B))(HA) \\
&= T^\dagger(HA) \\
&= (T^\dagger \circ H)(A)
\end{aligned}
$$

$$
\begin{aligned}
(T \circ (H \times \mathrm{Id}))^\dagger(f) &= Y(T(Hf, -)) \\
&= (Y \circ \lambda B.T(-, B))(Hf) \\
&= T^\dagger(Hf) \\
&= (T^\dagger \circ H)(f). \qquad \square
\end{aligned}
$$

4.4. **Coherence properties of (parameterised) initial algebras.** We model (arbitrary) recursive types within a linear category that is algebraically compact in a strong sense and that allows us to model recursive types involving the $\multimap$ connective. The non-linear recursive types form a subset of our types for which we have to, in addition, provide categorical structure that allows for them to be copied, discarded and promoted (the substructural rules of intuitionistic linear logic). We do this by providing a non-linear interpretation of these types within a cartesian category. The two interpretations live in different categories, but they are strongly related to each other via suitable mediating functors and a natural isomorphism. In order to show this, we first explain how parameterised initial algebras can be related to each other in such a strong sense. The current subsection is devoted to this.

Notation 4.15. For a functor $F : \mathbf{A} \times \mathbf{B} \to \mathbf{B}$, we define a functor

$$F^* := S \circ \lambda B.F(-, B) : \mathbf{A} \to [\omega, \mathbf{B}],$$

so that $F^\dagger = \mathrm{colim} \circ F^*$.

Assumption 4.16. *Throughout the remainder of the section, we assume we are given the following data. Let $\mathbf{A}$ and $\mathbf{C}$ be categories and let $\mathbf{B}$ and $\mathbf{D}$ be ω-categories with initial objects $\varnothing$ and 0, respectively. Let $\alpha : T \circ (N \times M) \Rightarrow M \circ H$ be a natural isomorphism, as in:*

$$
\begin{array}{ccc}
\mathbf{A} \times \mathbf{B} & \xrightarrow{\;\;N \times M\;\;} & \mathbf{C} \times \mathbf{D} \\
{\scriptstyle H}\Big\downarrow & \quad\alpha\;\Swarrow & \Big\downarrow{\scriptstyle T} \\
\mathbf{B} & \xrightarrow[\;\;M\;\;]{} & \mathbf{D}
\end{array}
$$

where H and T are ω-functors and where M is a strict ω-functor with $z : 0 \to M\varnothing$ the required (unique) isomorphism. The functor N need not be an ω-functor.

Lemma 4.17. *The assignment*

$$\alpha^* : T^* \circ N \Rightarrow (M \triangleright -) \circ H^* : \mathbf{A} \to [\omega, \mathbf{D}] \quad \text{given by}$$

$$(\alpha_A^*)_0 := \left(0 \xrightarrow{z} M\varnothing \right)$$

$$(\alpha_A^*)_{n+1} := \left(T(NA, -)^{n+1}0 \xrightarrow{T(NA, (\alpha_A^*)_n)} T(NA, MH(A, -)^n\varnothing) \xrightarrow{\alpha_{A, H(A, -)^n\varnothing}} MH(A, -)^{n+1}\varnothing \right).$$

defines a natural isomorphism.

Proof. In Appendix A.4.   □

The next theorem shows how to extend the action of $(-)^\dagger$ to natural transformations.

Theorem 4.18. *The natural isomorphism α induces a natural isomorphism*

$$\alpha^\dagger : T^\dagger \circ N \Rightarrow M \circ H^\dagger : \mathbf{A} \to \mathbf{D} \quad \text{defined by}$$

$$\alpha_A^\dagger := \left(T^\dagger NA = \mathrm{colim}(T^*NA) \xrightarrow{\mathrm{colim}(\alpha_A^*)} \mathrm{colim}(MH^*A) = M\mathrm{colim}(H^*A) = MH^\dagger A \right).$$

Proof. By Lemma 4.17, α_A^* is an isomorphism and thus so is $\alpha_A^\dagger$. Naturality follows from:

$$
\begin{array}{ccccccccc}
T^\dagger NA & = & \mathrm{colim}(T^*NA) & \xrightarrow{\mathrm{colim}(\alpha_A^*)} & \mathrm{colim}(MH^*A) & = & M\mathrm{colim}(H^*A) & = & MH^\dagger A \\
\Big\downarrow {\scriptstyle T^\dagger Nf} & & \Big\downarrow {\scriptstyle \mathrm{colim}(T^*Nf)} & & \Big\downarrow {\scriptstyle \mathrm{colim}(MH^*f)} & \Big\downarrow {\scriptstyle M\mathrm{colim}(H^*f)} & & & \Big\downarrow {\scriptstyle MH^\dagger f} \\
& & (1) & & (2) & (3) & & (4) & \\
T^\dagger NB & = & \mathrm{colim}(T^*NB) & \xrightarrow[\mathrm{colim}(\alpha_B^*)]{} & \mathrm{colim}(MH^*B) & = & M\mathrm{colim}(H^*B) & = & MH^\dagger B
\end{array}
$$

where (1) and (4) commute by definition, (2) commutes by naturality of α^* and functoriality of colim, and (3) commutes because M preserves ω-colimits (more specifically, Lemma 4.9). $\qquad\square$

Remark 4.19. The above construction generalises the operation $(-)^\dagger$ from [Fio94, Corollary 7.3.13] to the context of ω-categories.

Corollary 4.20. *If $\mathbf{A} = \mathbf{B}^n$, $\mathbf{C} = \mathbf{D}^n$, $N = M^{\times n}$ and $\alpha : T \circ M^{\times n+1} \Rightarrow M \circ H : \mathbf{B}^{n+1} \to \mathbf{D}$ is a natural isomorphism, then so is $\alpha^\dagger : T^\dagger \circ M^{\times n} \Rightarrow M \circ H^\dagger : \mathbf{B}^n \to \mathbf{D}$.*

By reading off the proof of Theorem 4.18, we obtain another corollary that we use in our adequacy proof.

Corollary 4.21. *In the special case where $\alpha = id$ and $z = id$, then $\alpha^\dagger = id$ and thus $T^\dagger \circ N = M \circ H^\dagger$.*

We need two lemmas that establish some properties of the $(-)^*$ and $(-)^\dagger$ operations.

Lemma 4.22. *The operation $(-)^*$ defined in Lemma 4.17 enjoys the following properties:*
(1) *If $\beta : H \Rightarrow H' : \mathbf{A} \times \mathbf{B} \to \mathbf{B}$ is a natural isomorphism, then for any $A \in \mathbf{A}$:*
$$
(M\beta \circ \alpha)_A^* = M\beta_A^* \circ \alpha_A^*.
$$
(2) *Given functors $P : \mathbf{E} \to \mathbf{C}$ and $Q : \mathbf{E} \to \mathbf{A}$ and given a natural isomorphism $\gamma : P \Rightarrow N \circ Q : \mathbf{E} \to \mathbf{C}$, then for any $E \in \mathbf{E}$:*
$$
(\alpha(Q \times Id) \circ T(\gamma \times M))_E^* = \alpha_{QE}^* \circ T^*\gamma_E.
$$

Proof. In Appendix A.5. $\qquad\square$

Lemma 4.23. *The operation $(-)^\dagger$ enjoys the following properties:*
(1) *If $\beta : H \Rightarrow H' : \mathbf{A} \times \mathbf{B} \to \mathbf{B}$ is a natural isomorphism, then $(M\beta \circ \alpha)^\dagger = M\beta^\dagger \circ \alpha^\dagger$.*
(2) *Given functors $P : \mathbf{E} \to \mathbf{C}$ and $Q : \mathbf{E} \to \mathbf{A}$ and given a natural isomorphism $\gamma : P \Rightarrow N \circ Q : \mathbf{E} \to \mathbf{C}$, then $(\alpha(Q \times Id) \circ T(\gamma \times M))^\dagger = \alpha^\dagger Q \circ T^\dagger \gamma$.*

Proof. (1) We have:
$$
\begin{aligned}
(M\beta \circ \alpha)_A^\dagger &= \mathrm{colim}((M\beta \circ \alpha)_A^*) && \text{(Definition)} \\
&= \mathrm{colim}(M\beta_A^* \circ \alpha_A^*) && \text{(Lemma 4.22)} \\
&= \mathrm{colim}(M\beta_A^*) \circ \mathrm{colim}(\alpha_A^*) && \text{(Functoriality)} \\
&= M\mathrm{colim}(\beta_A^*) \circ \mathrm{colim}(\alpha_A^*) && \text{(Lemma 4.9)} \\
&= M\beta_A^\dagger \circ \alpha_A^\dagger && \text{(Definition)} \\
&= (M\beta^\dagger \circ \alpha^\dagger)_A
\end{aligned}
$$

(2) We have:

$$
\begin{aligned}
(\alpha(Q \times \mathrm{Id}) \circ T(\gamma \times M))_E^\dagger &= \mathrm{colim}((\alpha(Q \times \mathrm{Id}) \circ T(\gamma \times M))_E^*) && \text{(Definition)} \\
&= \mathrm{colim}(\alpha_{QE}^* \circ T^* \gamma_E) && \text{(Lemma 4.22)} \\
&= \mathrm{colim}(\alpha_{QE}^*) \circ \mathrm{colim}(T^* \gamma_E) && \text{(Functoriality)} \\
&= \alpha_{QE}^\dagger \circ T^\dagger \gamma_E && \text{(Definition)} \\
&= (\alpha^\dagger Q \circ T^\dagger \gamma)_E && \square
\end{aligned}
$$

Remark 4.24. As a special case, if one takes N and M to be identity functors, then given a natural isomorphism $\alpha : T \Rightarrow H : \mathbf{A} \times \mathbf{B} \to \mathbf{B}$, it follows $\alpha^\dagger : T^\dagger \Rightarrow H^\dagger : \mathbf{A} \to \mathbf{B}$ is also a natural isomorphism and Lemma 4.23 (1) shows the operation $(-)^\dagger$ is functorial.

We need one more lemma before we may prove the main coherence property.

Lemma 4.25. *In the special case where categories* $\mathbf{A}$, $\mathbf{C}$ *and the functor* N *are trivial in Assumption 4.16, the following diagram of natural transformations:*

commutes, where we regard H *and* T *as functors* $H : \mathbf{B} \to \mathbf{B}$ *and* $T : \mathbf{D} \to \mathbf{D}$ *and we regard* α *as a natural isomorphism* $\alpha : T \circ M \Rightarrow M \circ H$.

Proof. In Appendix A.6. $\square$

The main result from this section follows. It shows how parameterised initial algebras may be related in the situation of Assumption 4.16. We will later use this theorem in order to show our semantics is sound.

Theorem 4.26. *The following diagram of natural isomorphisms:*

commutes.

Proof. Fix an object $A \in \mathbf{A}$. We have to show:

$$\begin{array}{ccc}
T'Y(T') & \xrightarrow{\;\;T'\alpha_A^\dagger\;\;} & T'MY(H') \\
\downarrow{\scriptstyle y^{T'}} & & \downarrow{\scriptstyle \alpha_{A,Y(H')}} \\
& & MH'Y(H') \\
& & \downarrow{\scriptstyle My^{H'}} \\
Y(T') & \xrightarrow[\;\;\alpha_A^\dagger\;\;]{} & MY(H')
\end{array}$$

where $T' := T(NA, -) : \mathbf{D} \to \mathbf{D}$ and $H' := H(A, -) : \mathbf{B} \to \mathbf{B}$. Note that we then have a natural isomorphism $\alpha' : T' \circ M \Rightarrow M \circ H'$ given by $\alpha' := \alpha_{A,-}$. Using Lemma 4.25, we get:

$$\mathrm{colim}(Ms^{H'}) \circ \mathrm{colim}(\alpha_A^*) = \mathrm{colim}(\alpha' * S(H')) \circ \mathrm{colim}(T'\alpha_A^*) \circ \mathrm{colim}(s^{T'}) \tag{4.3}$$

Using Lemma 4.10 and the fact that M and T preserve ω-colimits, we get:

$$M\mathrm{colim}(s^{H'}) \circ \mathrm{colim}(\alpha_A^*) = \alpha_{A,Y(H')} \circ T'\mathrm{colim}(\alpha_A^*) \circ \mathrm{colim}(s^{T'}) \tag{4.4}$$

Since $\mathrm{colim}(s^{H'})$ and $\mathrm{colim}(s^{T'})$ are isomorphisms, the above is equivalent to:

$$\mathrm{colim}(\alpha_A^*) \circ \mathrm{colim}(s^{T'})^{-1} = M\mathrm{colim}(s^{H'})^{-1} \circ \alpha_{A,Y(H')} \circ T'\mathrm{colim}(\alpha_A^*) \tag{4.5}$$

Finally, by definition of $\alpha^\dagger$ and y, we get: $\alpha_A^\dagger \circ y^{T'} = My^{H'} \circ \alpha_{A,Y(H')} \circ T'\alpha_A^\dagger$. $\qquad\square$

By combining this theorem together with Corollary 4.21, we obtain another corollary that is important for the adequacy proof.

Corollary 4.27. *In the special case where $\alpha = id$ and $z = id$, the 2-categorical diagram*

$$\begin{array}{ccc}
\mathbf{A} & \xrightarrow{\;\;N\;\;} & \mathbf{C} \\
{\scriptstyle H \circ \langle Id, H^\dagger\rangle}\left(\;\Downarrow{\scriptstyle \phi^H}\;\right){\scriptstyle H^\dagger} & & {\scriptstyle T \circ \langle Id, T^\dagger\rangle}\left(\;\Downarrow{\scriptstyle \phi^T}\;\right){\scriptstyle T^\dagger} \\
\mathbf{B} & \xrightarrow[\;\;M\;\;]{} & \mathbf{D}
\end{array}$$

commutes, i.e. $\phi^T N = M\phi^H$.

We also see that ϕ is natural in the choice of functor.

Corollary 4.28. *In the special case where $N = M = Id$, then $\alpha^\dagger \circ \phi^T = \phi^H \circ \alpha\langle Id, \alpha^\dagger\rangle$, i.e., ϕ^X is natural in X.*

5. CATEGORICAL MODEL

In this section we introduce our categorical model for LNL-FPC and describe its categorical properties. A **CPO**-LNL model is a **CPO**-enriched model of intuitionistic linear logic (also known as linear/non-linear model [Ben95, BW96]) that has suitable ω-colimits.

5.1. **CPO-categories and Algebraic Compactness.** A *cpo* (complete partial order) is a poset that has suprema of all increasing chains. If, in addition, the cpo has a least element, then we say it is *pointed*. A monotone function between two cpo's is *Scott continuous* if it preserves all suprema of increasing chains. If, in addition, the two cpo's are pointed and the function preserves the least element, then we say the function is *strict*.

We denote with **CPO** the category of cpo's and Scott continuous functions and with **CPO**$_{\perp!}$ the category of pointed cpo's and strict Scott continuous functions. Both categories are complete and cocomplete, **CPO** is cartesian closed and **CPO**$_{\perp!}$ is symmetric monoidal closed when equipped with the smash product and the strict function space [AJ94].

Our categorical model makes heavy use of *enriched* category theory. In particular, we use **CPO**-enriched and **CPO**$_{\perp!}$-enriched categories. We only provide a brief introduction to **CPO**-categories and we suggest the reader consults [Fio94, Chapter 2] for a more detailed introduction.

A **CPO**-category **C** is a category **C** whose homsets have the additional structure of a cpo and composition of morphisms is a Scott continuous function (in both arguments). A **CPO**-functor $F : \mathbf{C} \to \mathbf{D}$ between **CPO**-categories is a functor whose action on hom-cpo's $F_{X,Y} : \mathbf{C}(X,Y) \to \mathbf{D}(FX, FY)$ is Scott continuous for each pair of objects $X, Y \in \mathbf{C}$. A **CPO**-adjunction $\Phi : F \dashv G : \mathbf{D} \to \mathbf{C}$ is given by **CPO**-categories **C** and **D**, **CPO**-functors $F : \mathbf{C} \to \mathbf{D}$ and $G : \mathbf{D} \to \mathbf{C}$, together with a natural isomorphism $\Phi : \mathbf{C}(-, G-) \cong \mathbf{D}(F-, -) : \mathbf{C}^{\mathrm{op}} \times \mathbf{D} \to \mathbf{CPO}$. A **CPO**-symmetric monoidal closed category $\mathbf{C}(I, \otimes, \multimap)$ is a **CPO**-category **C** together with **CPO**-functors $\otimes : \mathbf{C} \times \mathbf{C} \to \mathbf{C}$ and $\multimap: \mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to \mathbf{C}$ that form a symmetric monoidal closed category in the usual sense (one can then conclude Currying is a **CPO**-adjunction). Similarly, **CPO**$_{\perp!}$-enriched versions of these notions may be defined by requiring the hom-cpo's to be pointed and Scott continuity to be strict.

Definition 5.1. In a **CPO**-category **C**, a morphism $e : A \to B$ is called an *embedding* if there is a morphism $p : B \to A$, called a *projection*, such that $p \circ e = \mathrm{id}_A$ and $e \circ p \leq \mathrm{id}_B$. The pair (e, p) is said to form an *embedding-projection (e-p) pair*.

It is well-known that one component of an e-p pair uniquely determines the other, so we shall write $e^\bullet$ to indicate the projection counterpart of an embedding e. Every isomorphism is an embedding and embeddings are closed under composition with $(e_1 \circ e_2)^\bullet = e_2^\bullet \circ e_1^\bullet$, which allows us to make the next definition.

Definition 5.2. Given a **CPO**-category **C**, its *subcategory of embeddings*, denoted $\mathbf{C}_e$, is the full-on-objects subcategory of **C** whose morphisms are exactly the embeddings of **C**.

By duality, one can also define $\mathbf{C}_p$ to be the subcategory of projections. The notion of an e-p pair is of fundamental importance for establishing the famous limit-colimit coincidence theorem [SP82]. Before we can state it, we need to define a few additional notions.

Definition 5.3. In a **CPO**-category **C**, an ω-*diagram over embeddings* is an ω-diagram $D : \omega \to \mathbf{C}$, such that all $D(i \leq j)$ are embeddings. Given such a diagram, we define $D^\bullet : \omega^{\mathrm{op}} \to \mathbf{C}$, to be the ω^{op}-diagram given by $D^\bullet(n) := D(n)$ and $D^\bullet(i \leq j) := D(i \leq j)^\bullet$. Given a cocone $\mu : D \to A$ in **C**, we define a cone $\mu^\bullet : A \to D^\bullet$ in **C** by $(\mu^\bullet)_i := (\mu_i)^\bullet$. We shall say that **C** has all ω-colimits over embeddings if every ω-diagram over embeddings has a colimit in **C**.

By combining several results from [SP82], one can show the next theorem.

Theorem 5.4 (Limit-colimit coincidence)**.** *Let* **C** *be a* **CPO***-category with all ω-colimits over embeddings. Let $D : \omega \to$ **C** *be an ω-diagram over embeddings and let $\mu : D \to A$ be a cocone of D. The following are equivalent:*

(1) *μ is a colimiting cocone of D in* **C***,*
(2) *μ is a colimiting cocone of D in* **C**$_e$*,*
(3) *Each μ_i is an embedding, $(\mu_i \circ \mu_i^\bullet)_i$ is an increasing chain and $\bigvee_i \mu_i \circ \mu_i^\bullet = id_A$,*
(4) *$\mu^\bullet$ is a limiting cone of $D^\bullet$ in* **C***,*
(5) *$\mu^\bullet$ is a limiting cone of $D^\bullet$ in* **C**$_p$*.*

Proof. In Appendix A.7. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $\square$

Remark 5.5. This theorem applies to both categories **CPO** and **CPO**$_{\perp!}$. However, when constructing an initial algebra for an endofunctor $T :$ **CPO** $\to$ **CPO**, the theorem is not useful, because the initial map $\iota : \varnothing \to T\varnothing$ is not an embedding (unless $T\varnothing = \varnothing$) and therefore the initial sequence of T is not necessarily computed over embeddings. Also, observe that **CPO**$_e$ has ω-colimits, but has no initial object and thus it is not a good setting for constructing initial algebras.

On the other hand, this theorem is very useful for constructing initial (and final) (co)algebras for many categories like **CPO**$_{\perp!}$ that have some additional structure, which we explain next.

Definition 5.6 ([Fio94, Definition 7.1.1])**.** In a **CPO**-category, an *e-initial* object is an initial object such that every morphism with it as source is an embedding. The dual notion is called a *p-terminal* object. An object that is both e-initial and p-terminal is called an *ep-zero* object.

The category **CPO** has an initial object given by $\varnothing$, but it does not have an e-initial object. The category **CPO**$_{\perp!}$ has an ep-zero object that is the one-element cpo $\{\perp\}$.

An endofunctor $T : $ **C** $\to$ **C** is *algebraically compact* if T has an initial T-algebra (Ω, τ), such that (Ω, τ^{-1}) is a final T-coalgebra. In this situation we say (Ω, τ) is a *free T-algebra*. A **CPO**-category **C** is **CPO***-algebraically compact* if every **CPO**-endofunctor $T : $ **C** $\to$ **C** has a free T-algebra. **CPO**-algebraic compactness also implies the existence of *parameterised free algebras* [Fio94].

Theorem 5.7 ([Fio94, Corollary 7.2.4])**.** *A* **CPO***-category with ep-zero and colimits of ω-chains over embeddings is* **CPO***-algebraically compact.*

Therefore, categories that enjoy this structure are an ideal setting for solving recursive domain equations. In particular, algebraic compactness is crucial for interpreting isorecursive types involving $\multimap$. We will see that the linear category of our model is indeed such a category and that we can very easily interpret (arbitrary) types within it. Non-linear types, however, need to admit an additional non-linear interpretation within a cartesian category that does not enjoy such a strong property. To resolve this problem, our type interpretations will be based on the following theorem, which combines results of [SP82].

Theorem 5.8. *Let* **A**, **B** *and* **C** *be* **CPO***-categories where* **A** *and* **B** *have ω-colimits over embeddings. If $T : $ **A**$^{\mathrm{op}} \times$ **B** $\to$ **C** *is a* **CPO***-functor, then the* covariant *functor*

$$T_e : \mathbf{A}_e \times \mathbf{B}_e \to \mathbf{C}_e \quad \text{given by} \quad T_e(A, B) = T(A, B) \quad \text{and} \quad T_e(e_1, e_2) = T((e_1^\bullet)^{\mathrm{op}}, e_2)$$

is an ω-functor.

Proof. This follows from [SP82, Theorem 3], because both categories **A** and **B** have locally determined ω-colimits of embeddings thanks to Theorem 5.4. □

Therefore, this theorem allows us to consider mixed-variance **CPO**-functors on **CPO**-categories as covariant ones on subcategories of embeddings. By trivialising the category **A**, we get:

Corollary 5.9. *Let* $T : \mathbf{B} \to \mathbf{C}$ *be a* **CPO**-*functor between* **CPO**-*categories* **B** *and* **C** *where* **B** *has* ω-*colimits over embeddings. Then* T *restricts to an* ω-*functor* $T_e : \mathbf{B}_e \to \mathbf{C}_e$.

5.2. **Models of Intuitionistic Linear Logic.** Our type system has both linear and non-linear features, so naturally, it should be interpreted within a model of Intuitionistic Linear Logic, also known as a linear/non-linear (LNL) model (see [BW96, Ben95] for more details).

Definition 5.10. A *model of Intuitionistic Linear Logic* is given by the following data: a cartesian monoidal category with finite coproducts $(\mathbf{C}, \times, 1, \amalg, \varnothing)$; a symmetric monoidal closed category with finite coproducts $(\mathbf{L}, \otimes, \multimap, I, +, 0)$; and a symmetric monoidal adjunction

$$\mathbf{C} \underset{G}{\overset{F}{\underset{\longleftarrow}{\overset{\longrightarrow}{\perp}}}} \mathbf{L}$$

. We also adopt the following notation. The comonad-endofunctor is $! := F \circ G : \mathbf{L} \to \mathbf{L}$; the unit and counit of the adjunction are $\eta : \mathrm{Id} \Rightarrow G \circ F : \mathbf{C} \to \mathbf{C}$ and $\epsilon : \, ! \Rightarrow \mathrm{Id} : \mathbf{L} \to \mathbf{L}$, respectively.

Moreover, in this situation, the left-adjoint F is both cocontinuous and strong symmetric monoidal. Thus, there exist isomorphisms $I \xrightarrow{u} F(1)$ and $0 \xrightarrow{z} F(\varnothing)$ in **L**, together with natural isomorphisms

$$\otimes \circ (F \times F) \overset{m}{\Longrightarrow} F \circ \times : \mathbf{C} \times \mathbf{C} \to \mathbf{L} \qquad \text{and} \qquad + \circ (F \times F) \overset{c}{\Longrightarrow} F \circ \amalg : \mathbf{C} \times \mathbf{C} \to \mathbf{L}$$

that satisfy some coherence conditions, which we omit here. Observe, that the structure of the natural isomorphisms m and c is very similar. In fact, most of our propositions involving these natural isomorphisms can be proven in a uniform way, so we introduce some notation for brevity.

Notation 5.11. We write $\odot$ for either the $\otimes$ or $+$ bifunctor; $\boxdot$ for the bifunctor $\times$ or $\amalg$ respectively; β for the natural isomorphisms m or c, respectively. Then, we have a natural isomorphism:

$$\odot \circ (F \times F) \overset{\beta}{\Longrightarrow} F \circ \boxdot : \mathbf{C} \times \mathbf{C} \to \mathbf{L}.$$

5.3. **Models of LNL-FPC.** Before we may define our categorical model, we first introduce the notion of a *pre-embedding*, which is novel (to the best of our knowledge). Pre-embeddings are of fundamental importance for defining the non-linear interpretations of non-linear types.

Definition 5.12. Let $T : \mathbf{A} \to \mathbf{B}$ be a **CPO**-functor between **CPO**-categories **A** and **B**. Then a morphism f in **A** is called a *pre-embedding with respect to* T if $T(f)$ is an embedding in **B**.

We can now describe our categorical model by combining results from the previous two subsections. Our categorical model is a model of Intuitionistic Linear Logic with some additional structure for solving recursive domain equations. Because of that, we shall use the same notation for it as introduced in the previous subsection.

Definition 5.13. A **CPO**-*LNL model* is given by the following data:

1. A **CPO**-symmetric monoidal closed category $(\mathbf{L}, \otimes, \multimap, I)$ with finite **CPO**-coproducts $(\mathbf{L}, +, 0)$;
2. A **CPO**-cartesian monoidal category $(\mathbf{C}, \times, 1)$ with finite **CPO**-coproducts $(\mathbf{C}, \mathrm{II}, \varnothing)$;
3. A **CPO**-symmetric monoidal adjunction $\mathbf{C} \; \underset{G}{\overset{F}{\underset{\longleftarrow}{\overset{\longrightarrow}{\perp}}}} \; \mathbf{L}$, such that:
4. $\mathbf{L}$ has an e-initial object 0 and all ω-colimits over embeddings; $\mathbf{C}$ has all ω-colimits over pre-embeddings, w.r.t F, and the product functor $(- \times -) : \mathbf{C} \times \mathbf{C} \to \mathbf{C}$ preserves them.

Assumption 5.14. *Throughout the rest of the paper we assume we are working with an arbitrary, but fixed, **CPO**-LNL model as in Definition 5.13.*

While not immediately obvious, $\mathbf{L}$ is **CPO**-algebraically compact. We reason as follows.

Theorem 5.15. *In every **CPO**-LNL model:*

(1) *The initial object $0 \in \mathbf{L}$ is an ep-zero object;*
(2) *Each zero morphism $\perp_{A,B}$ is least in $\mathbf{L}(A, B)$;*
(3) $\mathbf{L}$ *is $\mathbf{CPO}_{\perp!}$-enriched;*
(4) $\mathbf{L}$ *is **CPO**-algebraically compact.*

Proof. (1) For every object $A \in \mathbf{L}$, define a morphism $\perp_A := I \xrightarrow{e_I^\bullet} 0 \xrightarrow{e_A} A$, where $e_X : 0 \to X$ is unique and $e_X^\bullet : X \to 0$ is its projection counterpart. Then, for every $h : A \to B$, we have $h \circ \perp_A = \perp_B$. This means the category $\mathbf{L}$ is *weakly pointed* in the sense of [Bra97, Definition 9]. One can now define morphisms

$$\perp_{A,B} := A \xrightarrow{\lambda_A^{-1}} I \otimes A \xrightarrow{\mathbf{uncurry}(\perp_{A \multimap B})} B$$

and prove that:

$$g \circ \perp_{A,B} = \perp_{A,C} \qquad \text{for } g : B \to C \qquad \text{[Bra97, Proposition 11]}$$
$$\perp_{B,C} \circ f = \perp_{A,C} \qquad \text{for } f : A \to B \qquad \text{[Bra97, Proposition 12]}$$

In the presence of an initial object, this then implies that 0 is a zero object; $\perp_A = \perp_{I,A}$ and that the morphisms $\perp_{A,B}$ are zero morphisms [LMZ18, Lemma 4.8]. Therefore 0 is an ep-zero, because $(e_X, e_X^\bullet)$ forms an e-p pair for every object X and also $\perp_{A,B} = A \xrightarrow{e_A^\bullet} 0 \xrightarrow{e_B} B$.

(2) For any $f : A \to B$, we have $\perp_{A,B} = \perp_{B,B} \circ f = e_B \circ e_B^\bullet \circ f \le \mathrm{id}_B \circ f = f$.

(3) is now immediate from (2) and (4) follows by Theorem 5.7. $\qquad\qquad\square$

Every embedding in $\mathbf{C}$ is a pre-embedding (because **CPO**-functors preserve embeddings) and pre-embeddings are closed under composition, so one can form the full-on-objects subcategory of $\mathbf{C}$ of all pre-embeddings. We define such a subcategory within our model.

Definition 5.16. Let $\mathbf{C}_{pe}$ be the full-on-objects subcategory of $\mathbf{C}$ of pre-embeddings with respect to the functor F.

The importance of this subcategory is that it allows us to *reflect* solutions of certain recursive domain equations from the category $\mathbf{L}_e$ into the category $\mathbf{C}_{pe}$. As we have previously explained, the construction of initial algebras of endofunctors on $\mathbf{C}$ is not done over embeddings, in general (e.g. if $\mathbf{C} = \mathbf{CPO}$), because the initial map $\iota : \varnothing \to X$ usually is not an embedding. Observe, however, that $\iota : \varnothing \to X$ always is a pre-embedding with respect

to F. In fact, we shall see that the initial algebras for *all* non-linear type interpretations are constructed over initial sequences consisting of pre-embeddings.

    We proceed with a simple lemma.

Lemma 5.17. *Let* $\mathbf{A}$ *be a* $\mathbf{CPO}$*-category that has* ω*-colimits over embeddings. Then* $\mathbf{A}_e$ *has all* ω*-colimits, and the inclusion functor* $\mathbf{A}_e \hookrightarrow \mathbf{A}$ *is an* ω*-functor that reflects* ω*-colimits.*

Proof. This follows immediately by Theorem 5.4 (1) – (2).   □

    Our next theorem shows $\mathbf{C}_{pe}$ is a suitable setting for constructing initial algebras.

Theorem 5.18. *In every* $\mathbf{CPO}$*-LNL model:*

(1) $\mathbf{L}_e$ *is an* ω*-category, and the subcategory inclusion* $\mathbf{L}_e \hookrightarrow \mathbf{L}$ *is a strict* ω*-functor that also reflects* ω*-colimits.*

(2) $\mathbf{C}_{pe}$ *is an* ω*-category and the subcategory inclusion* $\mathbf{C}_{pe} \hookrightarrow \mathbf{C}$ *is a strict* ω*-functor that also reflects* ω*-colimits.*

(3) *The subcategory inclusion* $\mathbf{C}_e \hookrightarrow \mathbf{C}_{pe}$ *preserves and reflects* ω*-colimits (*$\mathbf{C}_e$ *does not necessarily have an initial object).*

Proof. (1) follows from Lemma 5.17 and the fact that $\mathbf{L}$ has an e-initial object 0.

    (2) Let $A : \mathbf{C}_{pe} \hookrightarrow \mathbf{C}$ be the inclusion, and let $D : \omega \to \mathbf{C}_{pe}$ be an ω-diagram. Let $\mu : D \to X$ be a cocone in $\mathbf{C}_{pe}$ and assume that it is colimiting in $\mathbf{C}$. We aim to show that μ is also colimiting in $\mathbf{C}_{pe}$, since this would imply that the inclusion A reflects ω-colimits. So let $\nu : D \to Y$ be another cocone in $\mathbf{C}_{pe}$, then ν is a cocone in $\mathbf{C}$, hence there is a unique morphism $f : X \to Y$ in $\mathbf{C}$ such that $f \circ \mu = \nu$. Now, both $F\mu : F \circ D \to FX$ and $F\nu : F \circ D \to FY$ are cocones of $F \circ D$ in $\mathbf{L}_e$. Since μ is colimiting in $\mathbf{C}$, and F (as a left adjoint) preserves colimits, it follows that $F\mu$ is the colimiting cocone of $F \circ D$ in $\mathbf{L}$. Hence there is a unique $e : FX \to FY$ in $\mathbf{L}$ such that $e \circ F\mu = F\nu$. Since $f \circ \mu = \nu$, we obtain $Ff \circ F\mu = F\nu$, so $Ff = e$. By (1), the inclusion $\mathbf{L}_e \hookrightarrow \mathbf{L}$ reflects ω-colimits, hence e must be a morphism in $\mathbf{L}_e$, hence an embedding. We conclude that Ff is an embedding, so f is a pre-embedding, which shows that μ is indeed colimiting in $\mathbf{C}_{pe}$.

    Next, we have to show that $\mathbf{C}_{pe}$ has all ω-colimits, so we have to show that the colimit of D exists. First, since $\mathbf{C}$ has ω-colimits over pre-embeddings, the colimiting cocone $\mu : D \to X$ of D in $\mathbf{C}$ exists. We have to show that μ is actually a cocone in $\mathbf{C}_{pe}$. Since F is a left adjoint, it preserves colimits, hence $F\mu : F \circ D \to FX$ is colimiting in $\mathbf{L}$. Now $F \circ D : \omega \to \mathbf{L}_e$ is an ω-diagram, hence by (1), it follows that $F\mu$ is also the colimiting cocone of $F \circ D$ in $\mathbf{L}_e$, meaning that $F\mu$ consists of embeddings. Hence μ consists of pre-embeddings, so is a cocone in $\mathbf{C}_{pe}$. Since we already showed that the inclusion A reflects ω-colimits, it follows that μ is colimiting in $\mathbf{C}_{pe}$, so the colimit of D exists in $\mathbf{C}_{pe}$, and is clearly preserved by A. So the inclusion A is an ω-functor.

    Let $\varnothing$ be the initial object of $\mathbf{C}$, and let X be some other object of $\mathbf{C}$. Let $i : \varnothing \to X$ be the unique initial map in $\mathbf{C}$. As a left adjoint, F preserves initial objects, hence $F\varnothing \in \mathbf{L}$ is initial, and $Fi : F\varnothing \to FX$ is unique. Since $F\varnothing \cong 0$ is e-initial, it follows that Fi is an embedding. Hence i is a pre-embedding, so a morphism in $\mathbf{C}_{pe}$. It follows that also $\mathbf{C}_{pe}$ is an ω-category, and since we showed that $\varnothing$ is also initial in $\mathbf{C}_{pe}$, it follows that A is strict.

    (3) We aim to show that the inclusion $B : \mathbf{C}_e \hookrightarrow \mathbf{C}_{pe}$ preserves and reflects ω-colimits. Note that $A \circ B : \mathbf{C}_e \hookrightarrow \mathbf{C}$ is the inclusion into $\mathbf{C}$. It follows from Lemma 5.17 that $\mathbf{C}_e$ has all ω-colimits, and the inclusion $A \circ B : \mathbf{C}_e \hookrightarrow \mathbf{C}$ preserves and reflects ω-colimits. Let $D : \omega \to \mathbf{C}_e$ be a diagram, and let $\mu : D \to X$ be a cocone in $\mathbf{C}_e$ such that $B\mu$ is colimiting in $\mathbf{C}_{pe}$. Since A preserves ω-colimits, it follows that $(A \circ B)\mu$ is colimiting in $\mathbf{C}$. Since

$A \circ B$ reflects ω-colimits, it follows that μ is colimiting in $\mathbf{C}_e$ so B indeed reflects ω-colimits. In order to show that B also preserves ω-colimits, let μ be the colimiting cocone of D in $\mathbf{C}_e$. Since $A \circ B$ preserves ω-colimits, it follows that $A(B\mu) = (A \circ B)\mu$ is colimiting in $\mathbf{C}$. Since A reflects ω-colimits by (2), it follows that $B\mu$ is colimiting, hence B indeed preserves ω-colimits. $\qquad\square$

By definition, the functor $F : \mathbf{C} \to \mathbf{L}$ restricts to a functor $F_{pe} : \mathbf{C}_{pe} \to \mathbf{L}_e$. By Corollary 5.9 the functor $G : \mathbf{L} \to \mathbf{C}$ restricts to an ω-functor $G_e : \mathbf{L}_e \to \mathbf{C}_e$. Let $G_{pe} : \mathbf{L}_e \to \mathbf{C}_{pe}$ be the functor given by $G_{pe} \coloneqq \left(\mathbf{L}_e \xrightarrow{G_e} \mathbf{C}_e \hookrightarrow \mathbf{C}_{pe} \right)$. The next theorem shows our model has suitable mediating functors for relating initial algebras.

Theorem 5.19. *In any* **CPO***-LNL model, $F_{pe} : \mathbf{C}_{pe} \to \mathbf{L}_e$ is a strict ω-functor. In addition, $G_{pe} : \mathbf{L}_e \to \mathbf{C}_{pe}$ is an ω-functor.*

Proof. Let $D : \omega \to \mathbf{C}_{pe}$ be an ω-diagram and let $\mu : D \to A$ be its colimit in $\mathbf{C}_{pe}$. Since F_{pe} is simply a restriction of F, then its action on D and μ is simply $F \circ D$ and $F\mu$, respectively. By Theorem 5.18 (1), μ is also a colimit of D in $\mathbf{C}$ and since $F : \mathbf{C} \to \mathbf{L}$ is a left adjoint, it follows $F\mu$ is a colimit of $F \circ D$ in $\mathbf{L}$. However, $F \circ D$ is a diagram over embeddings (in $\mathbf{L}$) and $F\mu$ is its colimit over embeddings (in $\mathbf{L}$) and therefore by Theorem 5.18 (1), $F\mu$ is a colimit of $F \circ D$ in $\mathbf{L}_e$. Thus, F_{pe} is an ω-functor and strictness follows because $z : 0 \to F\varnothing$ is an isomorphism. By Corollary 5.9 and by Theorem 5.18 (3), we see that G_{pe} is the composition of two ω-functors and is therefore an ω-functor itself. $\qquad\square$

By Notation 5.11, $\odot \circ (F \times F) \xRightarrow{\beta} F \circ \boxdot : \mathbf{C} \times \mathbf{C} \to \mathbf{L}$ is a natural isomorphism and by Corollary 5.9, we see that $\odot : \mathbf{L} \times \mathbf{L} \to \mathbf{L}$ restricts to an ω-functor $\odot_e : \mathbf{L}_e \times \mathbf{L}_e \to \mathbf{L}_e$. The final theorem of this subsection, together with the previous two, show that $\mathbf{C}_{pe}$ is a good setting for the non-linear interpretation of non-linear types.

Theorem 5.20. *In any* **CPO***-LNL model, the bifunctor $\boxdot : \mathbf{C} \times \mathbf{C} \to \mathbf{C}$ restricts to an ω-bifunctor $\boxdot_{pe} : \mathbf{C}_{pe} \times \mathbf{C}_{pe} \to \mathbf{C}_{pe}$. Moreover, the natural isomorphism β restricts to a natural isomorphism $\odot_e \circ (F_{pe} \times F_{pe}) \xRightarrow{\beta^{pe}} F_{pe} \circ \boxdot_{pe} : \mathbf{C}_{pe} \times \mathbf{C}_{pe} \to \mathbf{L}_e$.*

Proof. Let $f : X_1 \to Y_1$ and $g : X_2 \to Y_2$ be morphisms in $\mathbf{C}_{pe}$, so Ff and Fg are embeddings. Naturality of β gives us :

$$\beta_{(Y_1, Y_2)} \circ (Ff \odot Fg) = F(f \boxdot g) \circ \beta_{(X_1, X_2)},$$

hence

$$F(f \boxdot g) = \beta_{(Y_1, Y_2)} \circ (Ff \odot Fg) \circ \beta_{(X_1, X_2)}^{-1}.$$

As a natural isomorphism, each component of β is an embedding. Since $\odot : \mathbf{C} \times \mathbf{C} \to \mathbf{C}$ is a **CPO**-functor, it preserves embeddings, hence $Ff \odot Fg$ is an embedding. We conclude that $F(f \boxdot g)$ is an embedding, so $\boxdot$ restricts to a functor $\boxdot_{pe} : \mathbf{C}_{pe} \times \mathbf{C}_{pe} \to \mathbf{C}_{pe}$.

Let $D : \omega \to \mathbf{C}_{pe} \times \mathbf{C}_{pe}$ be an ω-diagram and let $\mu : D \to (A, B)$ be its colimiting cocone in $\mathbf{C}_{pe} \times \mathbf{C}_{pe}$. Since $\boxdot_{pe}$ is simply a restriction of $\boxdot$, then its action on D and μ is simply $\boxdot \circ D$ and $\boxdot\mu$, respectively. By Theorem 5.18 (2) and the fact that colimits are calculated pointwise in product categories, it follows μ is also a colimit of D in $\mathbf{C} \times \mathbf{C}$. Since $\boxdot : \mathbf{C} \times \mathbf{C} \to \mathbf{C}$ preserves ω-colimits over pre-embeddings ($\times$ by assumption and $\amalg$ is a colimit), it follows $\boxdot\mu$ is a colimit of $\boxdot \circ D$ in $\mathbf{C}$. Therefore by Theorem 5.18 (2), it may be reflected, so that $\boxdot\mu$ is a colimit of $\boxdot \circ D$ in $\mathbf{C}_{pe}$. Thus, $\boxdot_{pe}$ is an ω-functor. Finally, every

isomorphism is an embedding and therefore every component of β is an embedding (in $\mathbf{L}$), so that indeed β restricts to a natural isomorphism

$$\odot_e \circ (F_{pe} \times F_{pe}) \overset{\beta^{pe}}{\Longrightarrow} F_{pe} \circ \boxdot_{pe} : \mathbf{C}_{pe} \times \mathbf{C}_{pe} \to \mathbf{L}_e. \qquad \square$$

5.4. **Concrete CPO-LNL Models.** We now describe some concrete **CPO**-LNL models.

Theorem 5.21. *The adjunction* $\mathbf{CPO} \underset{U}{\overset{(-)_\bot}{\rightleftarrows}} \mathbf{CPO}_{\bot!}$ *, where the left adjoint is given by (domain-theoretic) lifting and the right adjoint U is the forgetful functor, is a* **CPO**-*LNL model.*

We will also show this model is computationally adequate in §7. This concrete model actually arises from a more general class of **CPO**-LNL models, as we shall now explain.

Let $\mathbf{M}$ be a small $\mathbf{CPO}_{\bot!}$-symmetric monoidal category and let $\widehat{\mathbf{M}}$ be the category of $\mathbf{CPO}_{\bot!}$-presheafs, that is, the category whose objects are the $\mathbf{CPO}_{\bot!}$-enriched functors $T : \mathbf{M}^{\mathrm{op}} \to \mathbf{CPO}_{\bot!}$ and whose morphisms are natural transformations between them (the notion of a $\mathbf{CPO}_{\bot!}$-natural transformation coincides with the ordinary notion). Then, by the enriched Yoneda Lemma, the category $\widehat{\mathbf{M}}$ is complete and cocomplete and it has a $\mathbf{CPO}_{\bot!}$-symmetric monoidal closed structure when equipped with the Day convolution [Day74]. In this situation there exists an adjunction $\mathbf{CPO}_{\bot!} \underset{\widehat{\mathbf{M}}(I,-)}{\overset{-\odot I}{\rightleftarrows}} \widehat{\mathbf{M}}$, where the left adjoint is given by taking the $\mathbf{CPO}_{\bot!}$-copower with the tensor unit I and the right adjoint is the representable functor (see [Bor94, §6]).

Theorem 5.22. *Composing the two adjunctions* $\mathbf{CPO} \underset{U}{\overset{(-)_\bot}{\rightleftarrows}} \mathbf{CPO}_{\bot!} \underset{\widehat{\mathbf{M}}(I,-)}{\overset{-\odot I}{\rightleftarrows}} \widehat{\mathbf{M}}$

yields a **CPO**-*LNL model.*

Observe that any $\mathbf{CPO}_{\bot!}$-functor necessarily preserves least morphisms and therefore also ep-zero objects. Thus, if $\mathbf{M} = \mathbf{1}$, then $\widehat{\mathbf{M}} \simeq \mathbf{1}$ and we recover the trivial (or completely degenerate) **CPO**-LNL model. If $\mathbf{M} = \mathbf{1}_\bot$ (the category with a unique object and two morphisms id and $\bot$, where $\bot$ is least), then $\widehat{\mathbf{M}} \cong \mathbf{CPO}_{\bot!}$ and we recover the **CPO**-LNL model from Theorem 5.21. Choosing $\mathbf{M}$ to be a suitable category of quantum computation, we recover a model of Proto-Quipper-M, a quantum programming language [RS17]. Choosing $\mathbf{M}$ to be a suitable category of string diagrams, we recover a model of ECLNL, a programming language for string diagrams [LMZ18]. Similar presheaf models have been used to model Ewire, a circuit description language [RS18], and the quantum lambda calculus [MSS13]. All of these languages have mixed linear/non-linear features and therefore our results presented here could help with design decisions on introducing recursive (or inductive) datatypes.

$$\llbracket \Theta \vdash A \rrbracket : \mathbf{L_e}^{|\Theta|} \to \mathbf{L}_e$$

$$\llbracket \Theta \vdash \Theta_i \rrbracket := \Pi_i$$

$$\llbracket \Theta \vdash !A \rrbracket := !_e \circ \llbracket \Theta \vdash A \rrbracket$$

$$\llbracket \Theta \vdash A + B \rrbracket := +_e \circ \langle \llbracket \Theta \vdash A \rrbracket, \llbracket \Theta \vdash B \rrbracket \rangle$$

$$\llbracket \Theta \vdash A \otimes B \rrbracket := \otimes_e \circ \langle \llbracket \Theta \vdash A \rrbracket, \llbracket \Theta \vdash B \rrbracket \rangle$$

$$\llbracket \Theta \vdash A \multimap B \rrbracket := \multimap_e \circ \langle \llbracket \Theta \vdash A \rrbracket, \llbracket \Theta \vdash B \rrbracket \rangle$$

$$\llbracket \Theta \vdash \mu X.A \rrbracket := \llbracket \Theta, X \vdash A \rrbracket^{\dagger}$$

$$(\!|\Theta \vdash P|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{C}_{pe}$$

$$(\!|\Theta \vdash \Theta_i|\!) := \Pi_i$$

$$(\!|\Theta \vdash !A|\!) := G_{pe} \circ \llbracket \Theta \vdash A \rrbracket \circ F_{pe}^{\times|\Theta|}$$

$$(\!|\Theta \vdash P + Q|\!) := \amalg_{pe} \circ \langle (\!|\Theta \vdash P|\!), (\!|\Theta \vdash Q|\!) \rangle$$

$$(\!|\Theta \vdash P \otimes Q|\!) := \times_{pe} \circ \langle (\!|\Theta \vdash P|\!), (\!|\Theta \vdash Q|\!) \rangle$$

$$(\!|\Theta \vdash \mu X.P|\!) := (\!|\Theta, X \vdash P|\!)^{\dagger}$$

Figure 4: Standard interpretation of types (left) and non-linear interpretation of non-linear types (right).

6. Denotational Semantics

We now describe the denotational semantics of LNL-FPC. Every type $\Theta \vdash A$ admits a (standard) interpretation as an ω-functor $\llbracket \Theta \vdash A \rrbracket : \mathbf{L}_e^{|\Theta|} \to \mathbf{L}_e$. Every *non-linear* type $\Theta \vdash P$ admits an *additional* non-linear interpretation as an ω-functor $(\!|\Theta \vdash P|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{C}_{pe}$ that is related to its standard interpretation via a natural isomorphism (see Figure 5)

$$\alpha^{\Theta \vdash P} : \llbracket \Theta \vdash P \rrbracket \circ F_{pe}^{\times|\Theta|} \Rightarrow F_{pe} \circ (\!|\Theta \vdash P|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{L}_e.$$

It is precisely the existence of this natural isomorphism that allows us to define the substructural operations (copying, discarding and promotion) on non-linear types.

As in FPC, the interpretation of a term $\Theta; \Gamma \vdash m : A$ is a *family* of morphisms of $\mathbf{L}$

$$\llbracket \Theta; \Gamma \vdash m : A \rrbracket = \left\{ \llbracket \Theta; \Gamma \vdash m : A \rrbracket_{\vec{Z}} : \llbracket \Theta \vdash \Gamma \rrbracket (F^{\times|\Theta|}\vec{Z}) \to \llbracket \Theta \vdash A \rrbracket (F^{\times|\Theta|}\vec{Z}) \mid \vec{Z} \in \mathrm{Ob}(\mathbf{C}^{|\Theta|}) \right\}$$

indexed by objects $\vec{Z}$ of $\mathbf{C}^{|\Theta|}$. In the special case when $\Theta = \cdot$ the interpretation can be seen as a morphism of $\mathbf{L}$. In order to show the semantics is sound, we also provide a non-linear interpretation of non-linear values that is compatible with the substructural operations. We remark that types are interpreted as functors on $\mathbf{L}_e$, which is a subcategory of $\mathbf{L}$, the category in which terms are interpreted. Doing so is crucial for defining the semantics.

6.1. **Interpretation of LNL-FPC Types.** The (standard) interpretation of a well-formed type $\Theta \vdash A$ is a functor $\llbracket \Theta \vdash A \rrbracket : \mathbf{L}_e^{|\Theta|} \to \mathbf{L}_e$ defined by induction on the derivation of $\Theta \vdash A$ (Figure 4, left). The non-linear interpretation of a non-linear type $\Theta \vdash P$ is a functor $(\!|\Theta \vdash P|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{C}_{pe}$ defined by induction on the derivation of $\Theta \vdash P$ (Figure 4, right). The non-linear interpretation of non-linear types depends on the standard interpretation, so the latter needs to be defined first, as we have done. Both assignments are ω-functors defined on ω-categories, as we show next, and so the assignments are well-defined, i.e., the parameterised initial algebras we need to interpret recursive types exist.

Theorem 6.1. *For any well-formed types $\Theta \vdash A$ and $\Theta \vdash P$, where P is non-linear:*

(1) *The assignment $\llbracket \Theta \vdash A \rrbracket : \mathbf{L}_e^{|\Theta|} \to \mathbf{L}_e$ is an ω-functor.*
(2) *The assignment $(\!|\Theta \vdash P|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{C}_{pe}$ is an ω-functor.*

$$\begin{array}{ccc}
\mathbf{C}_{pe}^{|\Theta|} & \xrightarrow{\;\;F_{pe}^{\times|\Theta|}\;\;} & \mathbf{L}_e^{|\Theta|} \\[2mm]
{\scriptstyle(\!|\Theta \vdash P|\!)}\Big\downarrow & \quad \alpha\!\!\diagdown\!\!\Swarrow & \Big\downarrow{\scriptstyle[\![\Theta \vdash P]\!]} \\[2mm]
\mathbf{C}_{pe} & \xrightarrow[\;\;F_{pe}\;\;]{} & \mathbf{L}_e
\end{array}$$

Figure 5: Relationship between standard and non-linear interpretations of non-linear types.

Proof. Both statements are proved by induction one after the other. The functor Π_i is obviously an ω-functor. The functors $!_e, +_e, \otimes_e, \multimap_e$ are ω-functors because of Theorem 5.8. The functors $G_{pe}, F_{pe}, \times_{pe}, \amalg_{pe}$ are ω-functors (Theorems 5.19 and 5.20). Moreover ω-functors are closed under composition and tupling. Finally, the μ-case follows from Theorem 4.12. $\qquad\square$

The next theorem shows that the standard and non-linear interpretations of non-linear types are strongly related via a natural isomorphism (see Figure 5).

Theorem 6.2. *For any non-linear type $\Theta \vdash P$, there exists a natural isomorphism*

$$\alpha^{\Theta\vdash P} : [\![\Theta \vdash P]\!] \circ F_{pe}^{\times|\Theta|} \Rightarrow F_{pe} \circ (\!|\Theta \vdash P|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{L}_e \qquad \textit{inductively defined by}$$

$$\alpha^{\Theta\vdash\Theta_i} := id$$

$$\alpha^{\Theta\vdash !A} := id$$

$$\alpha^{\Theta\vdash P\odot Q} := \beta^{pe}\langle(\!|\Theta \vdash P|\!),(\!|\Theta \vdash Q|\!)\rangle \circ \odot_e\langle\alpha^{\Theta\vdash P},\alpha^{\Theta\vdash Q}\rangle \qquad \textit{(see Theorem 5.20)}$$

$$\alpha^{\Theta\vdash \mu X.P} := (\alpha^{\Theta,X\vdash P})^\dagger \qquad\qquad\qquad\qquad\qquad \textit{(see Theorem 4.18)}.$$

Proof. In Appendix A.8. $\qquad\square$

In FPC, one has to prove a substitution lemma that shows the interpretation of types respects type substitution. In LNL-FPC, we have to prove three such lemmas: one for the standard interpretation $[\![\Theta \vdash A]\!]$ of (arbitrary) types, one for the non-linear interpretation $(\!|\Theta \vdash P|\!)$ of non-linear types, and one for the natural isomorphism $\alpha^{\Theta\vdash P}$ that relates them. In order to do this, one first has to show a permutation and a contraction lemma.

Lemma 6.3 (Permutation)**.** *Given types $\Theta, X, Y, \Theta' \vdash A$ and $\Theta, X, Y, \Theta' \vdash P$, with P non-linear:*
(1) $[\![\Theta, Y, X, \Theta' \vdash A]\!] = [\![\Theta, X, Y, \Theta' \vdash A]\!] \circ \mathrm{swap}_{m,m'}$
(2) $(\!|\Theta, Y, X, \Theta' \vdash P|\!) = (\!|\Theta, X, Y, \Theta' \vdash P|\!) \circ \mathrm{swap}_{m,m'}$
(3) $\alpha^{\Theta,Y,X,\Theta'\vdash P} = \alpha^{\Theta,X,Y,\Theta'\vdash P}\mathrm{swap}_{m,m'}$
where $|\Theta| = m$, $|\Theta'| = m'$ and $\mathrm{swap}_{m,m'} = \langle\Pi_1, \ldots, \Pi_m, \Pi_{m+2}, \Pi_{m+1}, \Pi_{m+3}, \ldots, \Pi_{m+m'+2}\rangle$.

Proof. Essentially the same as [Fio94, Lemma C.0.1]. $\qquad\square$

Lemma 6.4 (Contraction)**.** *Given types $\Theta, \Theta' \vdash A$ and $\Theta, \Theta' \vdash P$, with P non-linear:*
(1) $[\![\Theta, X, \Theta' \vdash A]\!] = [\![\Theta, \Theta' \vdash A]\!] \circ \mathrm{drop}_{m,m'}$
(2) $(\!|\Theta, X, \Theta' \vdash P|\!) = (\!|\Theta, \Theta' \vdash P|\!) \circ \mathrm{drop}_{m,m'}$
(3) $\alpha^{\Theta,X,\Theta'\vdash P} = \alpha^{\Theta,\Theta'\vdash P}\mathrm{drop}_{m,m'}$
where $X \notin \Theta \cup \Theta'$, $|\Theta| = m$, $|\Theta'| = m'$ and $\mathrm{drop}_{m,m'} = \langle\Pi_1, \ldots, \Pi_m, \Pi_{m+2}, \ldots, \Pi_{m+m'+1}\rangle$.

Proof. Essentially the same as [Fio94, Lemma C.0.2]. $\qquad\square$

Figure 6: The commuting diagram of natural isomorphisms for Lemma 6.5 (3).

In all models of FPC, the substitution lemma (and also the permutation and contraction lemmas) hold up to isomorphism. However, in models where the equality of Proposition 4.14 holds, this can be strengthened to an equality (cf. [Fio94, pp. 181]). Indeed, in LNL-FPC the same is true for the standard substitution lemma, but the non-linear substitution lemma holds only up to isomorphism, which is induced by the isomorphism α.

Lemma 6.5 (Substitution). *Given types $\Theta, X \vdash A$ and $\Theta \vdash B$ and $\Theta, X \vdash P$ and $\Theta \vdash R$ with P and R non-linear:*

(1) $\llbracket \Theta \vdash A[B/X] \rrbracket = \llbracket \Theta, X \vdash A \rrbracket \circ \langle Id, \llbracket \Theta \vdash B \rrbracket \rangle$

(2) $(\! | \Theta \vdash P[R/X] | \!) \cong (\! | \Theta, X \vdash P | \!) \circ \langle Id, (\! | \Theta \vdash R | \!) \rangle$, *where the natural isomorphism, denoted $\gamma^{\Theta \vdash P[R/X]}$ is given by:*

$$\gamma^{\Theta \vdash \Theta_i[R/X]} = id$$
$$\gamma^{\Theta \vdash X[R/X]} = id$$
$$\gamma^{\Theta \vdash !A[R/X]} = G_{pe} \llbracket \Theta, X \vdash A \rrbracket \langle F_{pe}^{\times |\Theta|}, \alpha^{\Theta \vdash R} \rangle$$
$$\gamma^{\Theta \vdash (P \odot Q)[R/X]} = \boxdot_{pe} \langle \gamma^{\Theta \vdash P[R/X]}, \gamma^{\Theta \vdash Q[R/X]} \rangle$$
$$\gamma^{\Theta \vdash \mu Y.P[R/X]} = (\gamma^{\Theta, Y \vdash P[R/X]})^{\dagger}$$

(3) $\alpha^{\Theta \vdash P[R/X]} = (F_{pe} \gamma^{\Theta \vdash P[R/X]})^{-1} \circ \alpha^{\Theta, X \vdash P} \langle Id, (\! | \Theta \vdash R | \!) \rangle \circ \llbracket \Theta, X \vdash P \rrbracket \langle F_{pe}^{\times |\Theta|}, \alpha^{\Theta \vdash R} \rangle$, *(see Figure 6).*

Proof. In Appendix A.9.   □

The most important type isomorphism in FPC is the folding and unfolding of recursive types. Denotationally, this is modelled using the properties of parameterised initial algebras and the substitution lemma. The same is true in LNL-FPC, but, in addition to the standard folding and unfolding of recursive types, we also have a non-linear interpretation for folding and unfolding of non-linear recursive types.

Definition 6.6. Given types $\Theta \vdash \mu X.A$ and $\Theta \vdash \mu X.P$, with P non-linear, let

$$\mathrm{fold}^{\Theta \vdash \mu X.A} : [\![\Theta \vdash A[\mu X.A/X]]\!] \Rightarrow [\![\Theta \vdash \mu X.A]\!] : \mathbf{L}_e^{|\Theta|} \to \mathbf{L}_e$$

$$\mathit{fold}^{\Theta \vdash \mu X.P} : (\!|\Theta \vdash P[\mu X.P/X]|\!) \Rightarrow (\!|\Theta \vdash \mu X.P|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{C}_{pe}$$

be the natural isomorphisms given by

$$\mathrm{fold}^{\Theta \vdash \mu X.A} := \left([\![\Theta \vdash A[\mu X.A/X]]\!] = [\![\Theta, X \vdash A]\!] \circ \langle \mathrm{Id}, [\![\Theta \vdash \mu X.A]\!] \rangle \overset{\phi}{\Rightarrow} [\![\Theta \vdash \mu X.A]\!] \right)$$

$$\mathit{fold}^{\Theta \vdash \mu X.P} := \left((\!|\Theta \vdash P[\mu X.P/X]|\!) \overset{\gamma}{\Rightarrow} (\!|\Theta, X \vdash P|\!) \circ \langle \mathrm{Id}, (\!|\Theta \vdash \mu X.P|\!) \rangle \overset{\phi}{\Rightarrow} (\!|\Theta \vdash \mu X.P|\!) \right).$$

We denote their inverses by $\mathrm{unfold}^{\Theta \vdash \mu X.A}$ and $\mathit{unfold}^{\Theta \vdash \mu X.P}$, respectively.

We now show that given a non-linear recursive type, the standard and non-linear interpretations of its folding and unfolding are also strongly related. We shall use this theorem later in order to prove that values $\Theta; \Gamma \vdash \mathtt{fold}\ v : P$ with P non-linear admit a non-linear interpretation.

Theorem 6.7. *Let $\Theta \vdash \mu X.P$ be a non-linear type. Then the diagram of natural isomorphisms*

$$
\begin{array}{ccc}
[\![\Theta \vdash P[\mu X.P/X]]\!] \circ F_{pe}^{\times|\Theta|} & \overset{\alpha}{\Longrightarrow} & F_{pe} \circ (\!|\Theta \vdash P[\mu X.P/X]|\!) \\
\Big\Downarrow {\scriptstyle \mathrm{fold} F_{pe}^{\times|\Theta|}} & & \Big\Downarrow {\scriptstyle F_{pe}\mathit{fold}} \\
[\![\Theta \vdash \mu X.P]\!] \circ F_{pe}^{\times|\Theta|} & \underset{\alpha}{\Longrightarrow} & F_{pe} \circ (\!|\Theta \vdash \mu X.P|\!)
\end{array}
$$

commutes.

Proof.

$$F_{pe}\mathit{fold}^{\Theta \vdash \mu X.P} \circ \alpha^{\Theta \vdash P[\mu X.P/X]}$$

$$= F_{pe}\phi \circ F_{pe}\gamma^{\Theta \vdash P[\mu X.P/X]} \circ \alpha^{\Theta \vdash P[\mu X.P/X]} \qquad\qquad \text{(Definition)}$$

$$= F_{pe}\phi \circ \alpha^{\Theta, X \vdash P}\langle \mathrm{Id}, (\!|\Theta \vdash \mu X.P|\!) \rangle \circ [\![\Theta, X \vdash P]\!]\langle F_{pe}^{\times|\Theta|}, \alpha^{\Theta \vdash \mu X.P} \rangle \quad \text{(Lemma 6.5 (3))}$$

$$= F_{pe}\phi \circ \alpha^{\Theta, X \vdash P}\langle \mathrm{Id}, (\!|\Theta, X \vdash P|\!)^\dagger \rangle \circ [\![\Theta, X \vdash P]\!]\langle F_{pe}^{\times|\Theta|}, (\alpha^{\Theta, X \vdash P})^\dagger \rangle \quad \text{(Definition)}$$

$$= (\alpha^{\Theta, X \vdash P})^\dagger \circ \phi F_{pe}^{\times|\Theta|} \qquad\qquad\qquad \text{(Theorem 4.26)}$$

$$= \alpha^{\Theta \vdash \mu X.P} \circ \mathrm{fold}^{\Theta \vdash \mu X.P} F_{pe}^{\times|\Theta|} \qquad\qquad\qquad \text{(Definition)}\ \square$$

6.2. **Interpretation of LNL-FPC Term Contexts.** The interpretation of term contexts is straightforward. The interpretation of an (arbitrary) term context $\Theta \vdash \Gamma$ is an ω-functor $[\![\Theta \vdash \Gamma]\!] : \mathbf{L}_e^{|\Theta|} \to \mathbf{L}_e$ (see Figure 7, left). A non-linear term context $\Theta \vdash \Phi$ admits an additional non-linear interpretation as an ω-functor $(\!|\Theta \vdash \Phi|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{C}_{pe}$ (see Figure 7, right). Just like with the interpretation of types, the two are strongly related.

$$\llbracket \Theta \vdash \Gamma \rrbracket : \mathbf{L}_e^{|\Theta|} \to \mathbf{L}_e \qquad\qquad (\!|\Theta \vdash \Phi|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{C}_{pe}$$

$$\llbracket \Theta \vdash \cdot \rrbracket := K_I \qquad\qquad (\!|\Theta \vdash \cdot|\!) := K_1$$

$$\llbracket \Theta \vdash \Gamma, x : A \rrbracket := \otimes_e \circ \langle \llbracket \Theta \vdash \Gamma \rrbracket, \llbracket \Theta \vdash A \rrbracket \rangle \qquad (\!|\Theta \vdash \Phi, x : P|\!) := \times_{pe} \circ \langle (\!|\Theta \vdash \Phi|\!), (\!|\Theta \vdash P|\!) \rangle$$

Figure 7: Interpretation of term contexts (left); non-linear interpretation of non-linear term contexts (right). K_X is the constant X functor.

Theorem 6.8. *For any non-linear term context $\Theta \vdash \Phi$, there exists a natural isomorphism*

$$\alpha^{\Theta \vdash \Phi} : \llbracket \Theta \vdash \Phi \rrbracket \circ F_{pe}^{\times |\Theta|} \Rightarrow F_{pe} \circ (\!|\Theta \vdash \Phi|\!) : \mathbf{C}_{pe}^{|\Theta|} \to \mathbf{L}_e$$

that is given by:

$$\alpha^{\Theta \vdash \cdot} := k_u \qquad \text{(the constant u natural transformation)}$$

$$\alpha^{\Theta \vdash \Phi, x : P} := m^{pe} \langle (\!|\Theta \vdash \Phi|\!), (\!|\Theta \vdash P|\!) \rangle \circ \otimes_e \langle \alpha^{\Theta \vdash \Phi}, \alpha^{\Theta \vdash P} \rangle \qquad \text{(see Theorem 5.20).}$$

Proof. Straightforward verification. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $\square$

6.3. **Interpretation of LNL-FPC Terms.** We introduce some notation for brevity that we use throughout the rest of the section. Given a type context Θ, we use $\vec{Z}$ to range over $\mathrm{Ob}(\mathbf{C}_{pe}^{|\Theta|}) = \mathrm{Ob}(\mathbf{C}^{|\Theta|})$. We write $\overrightarrow{FZ}$ for $F_{pe}^{\times |\Theta|} \vec{Z} = F^{\times |\Theta|} \vec{Z}$. We use Υ to range over both types and term contexts and we use Ψ to range over non-linear types and non-linear term contexts. Then, let

$$\llbracket \Upsilon \rrbracket_{\vec{Z}}^{\Theta} := \llbracket \Theta \vdash \Upsilon \rrbracket \left(\overrightarrow{FZ} \right) \in \mathrm{Ob}(\mathbf{L}), \qquad (\!|\Psi|\!)_{\vec{Z}}^{\Theta} := (\!|\Theta \vdash \Psi|\!) \left(\vec{Z} \right) \in \mathrm{Ob}(\mathbf{C}).$$

With this notation in place, from the previous two subsections we have isomorphisms (in $\mathbf{L}$)

$$\mathrm{unfold}_{\overrightarrow{FZ}}^{\Theta \vdash \mu X.A} : \llbracket \mu X.A \rrbracket_{\vec{Z}}^{\Theta} \cong \llbracket A[\mu X.A/X] \rrbracket_{\vec{Z}}^{\Theta} : \mathrm{fold}_{\overrightarrow{FZ}}^{\Theta \vdash \mu X.A}, \qquad \alpha_{\vec{Z}}^{\Theta \vdash \Psi} : \llbracket \Psi \rrbracket_{\vec{Z}}^{\Theta} \cong F(\!|\Psi|\!)_{\vec{Z}}^{\Theta}.$$

Remark 6.9. The isomorphisms above are natural in $\mathbf{C}_{pe}^{|\Theta|}$, which we saw is crucial for defining the type interpretations. However, as in FPC, this naturality is irrelevant for the term interpretations – we only need the fact that each component is an isomorphism in $\mathbf{L}$. Thus, there is no danger in working within $\mathbf{C}$ and $\mathbf{L}$, instead of $\mathbf{C}_{pe}$ and $\mathbf{L}_e$.

Next, we show how to interpret the substructural rules of Intuitionistic Linear Logic in a **CPO**-LNL model. The isomorphism α plays a fundamental part.

Definition 6.10. We define discarding ($\diamond$), copying ($\triangle$) and promotion ($\square$) morphisms:

$$\diamond_{\vec{Z}}^{\Theta \vdash \Psi} := \llbracket \Psi \rrbracket_{\vec{Z}}^{\Theta} \xrightarrow{\alpha} F(\!|\Psi|\!)_{\vec{Z}}^{\Theta} \xrightarrow{F1} F1 \xrightarrow{u^{-1}} I$$

$$\triangle_{\vec{Z}}^{\Theta \vdash \Psi} := \llbracket \Psi \rrbracket_{\vec{Z}}^{\Theta} \xrightarrow{\alpha} F(\!|\Psi|\!)_{\vec{Z}}^{\Theta} \xrightarrow{F\langle \mathrm{id}, \mathrm{id} \rangle} F\left((\!|\Psi|\!)_{\vec{Z}}^{\Theta} \times (\!|\Psi|\!)_{\vec{Z}}^{\Theta} \right) \xrightarrow{m^{-1}} F(\!|\Psi|\!)_{\vec{Z}}^{\Theta} \otimes F(\!|\Psi|\!)_{\vec{Z}}^{\Theta} \xrightarrow{\alpha^{-1} \otimes \alpha^{-1}} \llbracket \Psi \rrbracket_{\vec{Z}}^{\Theta} \otimes \llbracket \Psi \rrbracket_{\vec{Z}}^{\Theta}$$

$$\square_{\vec{Z}}^{\Theta \vdash \Psi} := \llbracket \Psi \rrbracket_{\vec{Z}}^{\Theta} \xrightarrow{\alpha} F(\!|\Psi|\!)_{\vec{Z}}^{\Theta} \xrightarrow{F\eta} !F(\!|\Psi|\!)_{\vec{Z}}^{\Theta} \xrightarrow{!\alpha^{-1}} !\llbracket \Psi \rrbracket_{\vec{Z}}^{\Theta}$$

where, for brevity, we have written α instead of $\alpha_{\vec{Z}}^{\Theta \vdash \Psi}$.

Proposition 6.11. *The triple $\left(\llbracket \Psi \rrbracket_{\vec{Z}}^{\Theta}, \triangle_{\vec{Z}}^{\Theta \vdash \Psi}, \diamond_{\vec{Z}}^{\Theta \vdash \Psi} \right)$ forms a cocommutative comonoid in $\mathbf{L}$.*

Proof. By the axioms of monoidal adjunctions and the cartesian structure of $\mathbf{C}$. $\qquad$ $\square$

$$[\![\Phi, x : A \vdash x : A]\!] := [\![\Phi]\!] \otimes [\![A]\!] \xrightarrow{\diamond \otimes \mathrm{id}} I \otimes [\![A]\!] \xrightarrow{\cong} [\![A]\!]$$

$$[\![\Gamma \vdash \mathtt{left}_{A,B}\, m : A + B]\!] := [\![\Gamma]\!] \xrightarrow{[\![m]\!]} [\![A]\!] \xrightarrow{\mathrm{left}} [\![A]\!] + [\![B]\!] = [\![A + B]\!]$$

$$[\![\Gamma \vdash \mathtt{right}_{A,B}\, m : A + B]\!] := [\![\Gamma]\!] \xrightarrow{[\![m]\!]} [\![B]\!] \xrightarrow{\mathrm{right}} [\![A]\!] + [\![B]\!] = [\![A + B]\!]$$

$$[\![\Phi, \Gamma, \Sigma \vdash \mathtt{case}\; m\; \mathtt{of}\; \{\mathtt{left}\; x \to n \mid \mathtt{right}\; y \to p\} : C]\!] := [\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Sigma]\!] \xrightarrow{\triangle \otimes \mathrm{id}}$$

$$[\![\Phi]\!] \otimes [\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Sigma]\!] \xrightarrow{\cong} [\![\Phi]\!] \otimes [\![\Sigma]\!] \otimes [\![\Phi]\!] \otimes [\![\Gamma]\!] \xrightarrow{\mathrm{id} \otimes [\![m]\!]} [\![\Phi]\!] \otimes [\![\Sigma]\!] \otimes [\![A + B]\!] \xrightarrow{\cong}$$

$$([\![\Phi]\!] \otimes [\![\Sigma]\!] \otimes [\![A]\!]) + ([\![\Phi]\!] \otimes [\![\Sigma]\!] \otimes [\![B]\!]) \xrightarrow{[[\![n]\!], [\![p]\!]]} [\![C]\!]$$

$$[\![\Phi, \Gamma, \Sigma \vdash \langle m, n \rangle : A \otimes B]\!] := [\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Sigma]\!] \xrightarrow{\triangle \otimes \mathrm{id}} [\![\Phi]\!] \otimes [\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Sigma]\!] \xrightarrow{\cong}$$

$$[\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Phi]\!] \otimes [\![\Sigma]\!] \xrightarrow{[\![m]\!] \otimes [\![n]\!]} [\![A]\!] \otimes [\![B]\!] = [\![A \otimes B]\!]$$

$$[\![\Phi, \Gamma, \Sigma \vdash \mathtt{let}\; \langle x, y \rangle = m\; \mathtt{in}\; n : C]\!] := [\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Sigma]\!] \xrightarrow{\triangle \otimes \mathrm{id}} [\![\Phi]\!] \otimes [\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Sigma]\!] \xrightarrow{\cong}$$

$$[\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Phi]\!] \otimes [\![\Sigma]\!] \xrightarrow{[\![m]\!] \otimes \mathrm{id}} [\![A \otimes B]\!] \otimes [\![\Phi]\!] \otimes [\![\Sigma]\!] \xrightarrow{\cong} [\![\Phi]\!] \otimes [\![\Sigma]\!] \otimes [\![A]\!] \otimes [\![B]\!] \xrightarrow{[\![n]\!]} [\![C]\!]$$

$$[\![\Gamma \vdash \lambda x^A . m : A \multimap B]\!] := [\![\Gamma]\!] \xrightarrow{\mathbf{curry}([\![m]\!])} ([\![A]\!] \multimap [\![B]\!]) = [\![A \multimap B]\!]$$

$$[\![\Phi, \Gamma, \Sigma \vdash mn : B]\!] := [\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Sigma]\!] \xrightarrow{\triangle \otimes \mathrm{id}} [\![\Phi]\!] \otimes [\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Sigma]\!] \xrightarrow{\cong}$$

$$[\![\Phi]\!] \otimes [\![\Gamma]\!] \otimes [\![\Phi]\!] \otimes [\![\Sigma]\!] \xrightarrow{[\![m]\!] \otimes [\![n]\!]} ([\![A]\!] \multimap [\![B]\!]) \otimes [\![A]\!] \xrightarrow{\mathrm{eval}} [\![B]\!]$$

$$[\![\Phi \vdash \mathtt{lift}\; m : !A]\!] := [\![\Phi]\!] \xrightarrow{\Box} ![\![\Phi]\!] \xrightarrow{![\![m]\!]} ![\![A]\!] = [\![!A]\!]$$

$$[\![\Gamma \vdash \mathtt{force}\; m : A]\!] := [\![\Gamma]\!] \xrightarrow{[\![m]\!]} ![\![A]\!] \xrightarrow{\epsilon} [\![A]\!]$$

$$[\![\Gamma \vdash \mathtt{fold}_{\mu X.A}\, m : \mu X.A]\!] := [\![\Gamma]\!] \xrightarrow{[\![m]\!]} [\![A[\mu X.A/X]]\!] \xrightarrow{\mathrm{fold}} [\![\mu X.A]\!]$$

$$[\![\Gamma \vdash \mathtt{unfold}\; m : A[\mu X.A/X]]\!] := [\![\Gamma]\!] \xrightarrow{[\![m]\!]} [\![\mu X.A]\!] \xrightarrow{\mathrm{unfold}} [\![A[\mu X.A/X]]\!]$$

where, for brevity, we write $[\![\Upsilon]\!]$ for $[\![\Upsilon]\!]_{\vec{Z}}^{\Theta}$ for all types or term contexts Υ and where we omit the subscripts and type superscripts of some morphisms (which can be easily inferred).

Figure 8: Interpretation of LNL-FPC terms.

We can now define the interpretation of LNL-FPC terms. A term $\Theta; \Gamma \vdash m : A$ is interpreted as a family morphisms of $\mathbf{L}$ indexed by $\vec{Z}$. That is

$$[\![\Theta; \Gamma \vdash m : A]\!] := \left\{ [\![\Gamma \vdash m : A]\!]_{\vec{Z}}^{\Theta} : [\![\Gamma]\!]_{\vec{Z}}^{\Theta} \to [\![A]\!]_{\vec{Z}}^{\Theta} \mid \vec{Z} \in \mathrm{Ob}(\mathbf{C}^{|\Theta|}) \right\},$$

where $[\![\Gamma \vdash m : A]\!]_{\vec{Z}}^{\Theta}$ is defined by induction on the derivation of $\Theta; \Gamma \vdash m : A$ in Figure 8 (notice the remark). Just as in FPC, the interpretation of terms is invariant with respect to both Θ and $\vec{Z}$, so omitting the subscript and superscript annotations (as we have done) should not lead to confusion, but on the other hand it improves readability. For brevity, we also informally write $[\![m]\!]$ instead of $[\![\Gamma \vdash m : A]\!]_{\vec{Z}}^{\Theta}$, which can be inferred from context.

The careful reader might have noticed that the above notation is not completely precise, because terms do not have unique derivations. Technically, the definition should be read as

providing an interpretation for a derivation D of $\Theta; \Gamma \vdash m : A$. However, the next theorem justifies our notation.

Theorem 6.12. *Let D_1 and D_2 be two derivations of a judgement $\Theta; \Gamma \vdash m : A$. Then $[\![D_1]\!]_{\vec{Z}}^{\Theta} = [\![D_2]\!]_{\vec{Z}}^{\Theta}$.*

Proof. As previously explained, two derivations may only differ in whether some non-linear variables are propagated up into both premises or just one. Variables that are propagated up but not used are always discarded. The proof follows by induction using Proposition 6.11. $\square$

In order to prove our semantics sound, we have to first formulate a substitution lemma for terms. This lemma, in turn, requires that the interpretation of *non-linear values* behave well with respect to the substructural morphisms of Definition 6.10. To show this, we reason as follows.

Definition 6.13. Given a non-linear term context $\Theta \vdash \Phi$ and a non-linear type $\Theta \vdash P$, a morphism $f : [\![\Phi]\!]_{\vec{Z}}^{\Theta} \to [\![P]\!]_{\vec{Z}}^{\Theta}$ in $\mathbf{L}$ is called *non-linear*, whenever there exists a morphism

$$f' : (\!|\Phi|\!)_{\vec{Z}}^{\Theta} \to (\!|P|\!)_{\vec{Z}}^{\Theta} \text{ in } \mathbf{C}, \text{ with } f = \left([\![\Phi]\!]_{\vec{Z}}^{\Theta} \xrightarrow{\alpha_{\vec{Z}}^{\Theta \vdash \Phi}} F(\!|\Phi|\!)_{\vec{Z}}^{\Theta} \xrightarrow{Ff'} F(\!|P|\!)_{\vec{Z}}^{\Theta} \xrightarrow{\left(\alpha_{\vec{Z}}^{\Theta \vdash P}\right)^{-1}} [\![P]\!]_{\vec{Z}}^{\Theta} \right).$$

Proposition 6.14. *If $f : [\![\Phi]\!]_{\vec{Z}}^{\Theta} \to [\![P]\!]_{\vec{Z}}^{\Theta}$ is non-linear, then:*

$$\diamond_{\vec{Z}}^{\Theta \vdash P} \circ f = \diamond_{\vec{Z}}^{\Theta \vdash \Phi};$$
$$\triangle_{\vec{Z}}^{\Theta \vdash P} \circ f = (f \otimes f) \circ \triangle_{\vec{Z}}^{\Theta \vdash \Phi};$$
$$\square_{\vec{Z}}^{\Theta \vdash P} \circ f = !f \circ \square_{\vec{Z}}^{\Theta \vdash \Phi}.$$

In particular, non-linear morphisms are comonoid homomorphisms (with respect to Proposition 6.11).

Recall that by Lemma 2.7, non-linear values necessarily have a non-linear term context.

Proposition 6.15. *Let $\Theta; \Phi \vdash v : P$ be a non-linear value. Then $[\![\Phi \vdash v : P]\!]_{\vec{Z}}^{\Theta}$ is non-linear.*

Proof. We define a non-linear interpretation

$$(\!|\Phi \vdash v : P|\!)_{\vec{Z}}^{\Theta} : (\!|\Phi|\!)_{\vec{Z}}^{\Theta} \to (\!|P|\!)_{\vec{Z}}^{\Theta}$$

in $\mathbf{C}$ by induction on the derivation of $\Theta; \Phi \vdash v : P$ as follows:

$$(\!|\Phi, x : P \vdash x : P|\!) := (\!|\Phi|\!) \times (\!|P|\!) \xrightarrow{\pi_2} (\!|P|\!)$$

$$(\!|\Phi \vdash \texttt{left}_{P,R} v : P + R|\!) := (\!|\Phi|\!) \xrightarrow{(\!|v|\!)} (\!|P|\!) \xrightarrow{\text{inl}} (\!|P|\!) \amalg (\!|R|\!) = (\!|P + R|\!)$$

$$(\!|\Phi \vdash \texttt{right}_{P,R} v : P + R|\!) := (\!|\Phi|\!) \xrightarrow{(\!|v|\!)} (\!|R|\!) \xrightarrow{\text{inr}} (\!|P|\!) \amalg (\!|R|\!) = (\!|P + R|\!)$$

$$(\!|\Phi \vdash \langle v, w \rangle : P \otimes R|\!) := (\!|\Phi|\!) \xrightarrow{\langle (\!|v|\!), (\!|w|\!) \rangle} (\!|P|\!) \times (\!|R|\!) = (\!|P \otimes R|\!)$$

$$(\!|\Phi \vdash \texttt{lift } m : !A|\!) := (\!|\Phi|\!) \xrightarrow{\eta} GF(\!|\Phi|\!) \xrightarrow{G\alpha^{-1}} G[\![\Phi]\!] \xrightarrow{G[\![m]\!]} G[\![A]\!] = (\!|!A|\!)$$

$$(\!|\Phi \vdash \texttt{fold}_{\mu X.P} v : \mu X.P|\!) := (\!|\Phi|\!) \xrightarrow{(\!|v|\!)} (\!|P[\mu X.P/X]|\!) \xrightarrow{fold} (\!|\mu X.P|\!),$$

where π_2 is the second projection in $\mathbf{C}$; inl and inr are the coproduct injections in $\mathbf{C}$; $\langle f, g \rangle$ is the unique map induced by the product in $\mathbf{C}$ and *fold* is defined in Definition 6.6. The

definition is invariant with respect to Θ and $\vec{Z}$, so again, we omit them from the subscript and superscript to improve readability.

To complete the proof, one has to show

$$\llbracket \Phi \vdash v : P \rrbracket^{\Theta}_{\vec{Z}} = \left(\llbracket \Phi \rrbracket^{\Theta}_{\vec{Z}} \xrightarrow{\alpha^{\Theta \vdash \Phi}_{\vec{Z}}} F(\!|\Phi|\!)^{\Theta}_{\vec{Z}} \xrightarrow{F(\!|\Phi \vdash v : P|\!)^{\Theta}_{\vec{Z}}} F(\!|P|\!)^{\Theta}_{\vec{Z}} \xrightarrow{\left(\alpha^{\Theta \vdash P}_{\vec{Z}}\right)^{-1}} \llbracket P \rrbracket^{\Theta}_{\vec{Z}} \right),$$

which follows by simple verification using the available categorical data. The `fold` case is the most complicated (in general), but because of the preparatory work we have done, it may be established simply by using Theorem 6.7. $\qquad\square$

Next, as usual, we also have to formulate a substitution lemma for terms. Before we do so, recall Lemma 2.6.

Lemma 6.16 (Substitution). *Let* $\Theta; \Phi, \Gamma, x : A \vdash m : B$ *be a term and* $\Theta; \Phi, \Sigma \vdash v : A$ *a value with* $\Gamma \cap \Sigma = \varnothing$. *Then*

$$\llbracket m[v/x] \rrbracket = \left(\llbracket \Phi \rrbracket \otimes \llbracket \Gamma \rrbracket \otimes \llbracket \Sigma \rrbracket \xrightarrow{\cong \circ (\triangle \otimes id)} \llbracket \Phi \rrbracket \otimes \llbracket \Gamma \rrbracket \otimes \llbracket \Phi \rrbracket \otimes \llbracket \Sigma \rrbracket \xrightarrow{id \otimes \llbracket v \rrbracket} \llbracket \Phi \rrbracket \otimes \llbracket \Gamma \rrbracket \otimes \llbracket A \rrbracket \xrightarrow{\llbracket m \rrbracket} \llbracket B \rrbracket \right)$$

where we have omitted the obvious subscript and superscript annotations, for brevity.

Proof. By induction using Lemma 2.6 and Propositions 6.14 - 6.15. $\qquad\square$

With this in place, we may now show our semantics is sound.

Theorem 6.17 (Soundness). *If* $\Theta; \Gamma \vdash m : A$ *and* $m \Downarrow v$, *then* $\llbracket \Gamma \vdash m : A \rrbracket^{\Theta}_{\vec{Z}} = \llbracket \Gamma \vdash v : A \rrbracket^{\Theta}_{\vec{Z}}$.

Proof. Straightforward induction. $\qquad\square$

6.4. **Notation for closed types.** In the special case when $\Theta = \cdot$, the notation for terms and term contexts can be simplified, as we showed in §2.2. The same is also true for the denotational semantics and we will use this simplified notation for the proof of adequacy in the next section. We denote with $*$ the unique object of the terminal category $\mathbf{1}$.

Let $\llbracket \Upsilon \rrbracket := \llbracket \cdot \vdash \Upsilon \rrbracket(*) \in \mathrm{Ob}(\mathbf{L})$. Let $(\!|\Psi|\!) := (\!|\cdot \vdash \Psi|\!)(*) \in \mathrm{Ob}(\mathbf{C})$. For $\chi \in \{\alpha, \diamond, \triangle, \square\}$, we write $\chi^{\Psi} := \chi^{\cdot \vdash \Psi}_{*}$. Then, we have isomorphisms $\alpha^{\Psi} : \llbracket \Psi \rrbracket \cong F(\!|\Psi|\!)$, and substructural morphisms $\diamond^{\Psi} : \llbracket \Psi \rrbracket \to I$, $\triangle^{\Psi} : \llbracket \Psi \rrbracket \to \llbracket \Psi \rrbracket \otimes \llbracket \Psi \rrbracket$ and $\square^{\Psi} : \llbracket \Psi \rrbracket \to !\llbracket \Psi \rrbracket$.

Writing $\mathrm{fold}^{\mu X.A}$ for $\mathrm{fold}^{\cdot \vdash \mu X.A}_{*}$ and $\mathrm{unfold}^{\mu X.A}$ for its inverse, we get an isomorphism $\mathrm{unfold}^{\mu X.A} : \llbracket \mu X.A \rrbracket \cong \llbracket A[\mu X.A/X] \rrbracket : \mathrm{fold}^{\mu X.A}$.

For the interpretation of terms, we write $\llbracket \Gamma \vdash m : A \rrbracket := \llbracket \Gamma \vdash m : A \rrbracket^{\cdot}_{*} : \llbracket \Gamma \rrbracket \to \llbracket A \rrbracket$, which is a morphism in $\mathbf{L}$ and its definition can be simply read off from Figure 8, disregarding the remark at the bottom.

7. Computational Adequacy

In this section we show that computational adequacy holds at non-linear types for a class of **CPO**-LNL models that satisfy some additional conditions (Theorem 7.10). In the process, we also provide sufficient conditions for program termination at *any* type (Theorem 7.9). The main difficulty is showing that the formal approximation relations exist (Lemma 7.6), the proof of which is presented in §7.1.

We begin by specifying the additional properties of **CPO**-LNL models that we use in the proof of adequacy.

Definition 7.1. A *computationally adequate* **CPO**-LNL model is a **CPO**-LNL model where:
(1) $\mathrm{id}_I \neq \perp_{I,I}$ (or equivalently $I \not\cong 0$).
(2) For any two pairs of parallel morphisms $f_1, g_1 : I \to A$ and $f_2, g_2 : I \to B$, such that $f_1 \neq \perp \neq g_1$ and $f_2 \neq \perp \neq g_2 : f_1 \otimes f_2 \leq g_1 \otimes g_2 \Rightarrow f_1 \leq g_1$ and $f_2 \leq g_2$.

For instance, the **CPO**-LNL model of Theorem 5.21 is computationally adequate.

Remark 7.2. Condition (1) is clearly necessary for adequacy. If $I \cong 0$, then $\mathbf{L} \simeq \mathbf{1}$, and since $\mathbf{L}$ is completely degenerate, then clearly adequacy cannot hold. Condition (2) is a strong assumption that requires the tensor product of $\mathbf{L}$ to behave like the smash product of domain theory. This is a sufficient condition, but we do not know if it is necessary. However, it appears to be unavoidable when using the standard proof strategy based on formal approximation relations, just as we do (see Remark 7.22). In principle, there could be a different proof strategy that does not make this assumption, but we do not know of such a strategy that works in our setting.

Assumption 7.3. *Throughout the rest of the section we consider an arbitrary, but fixed, computationally adequate* **CPO**-*LNL model.*

We begin by showing that non-linear values correspond to non-zero morphisms.

Lemma 7.4. *Let $\cdot \vdash v : P$ be a non-linear value. Then $[\![v]\!] \neq \perp$.*

Proof. Assume for contradiction $[\![v]\!] = \perp$. Therefore $\diamond^P \circ [\![v]\!] = \perp_{I,I}$. Since $\cdot \vdash v : P$ is a non-linear value, it may be discarded so that $\diamond^P \circ [\![v]\!] = \diamond^{\cdot} = \mathrm{id}_I$ (Propositions 6.14-6.15). But then $\perp_{I,I} = \mathrm{id}_I$, which contradicts Definition 7.1. $\square$

Notation 7.5. Given morphisms $f_1 : I \to A_1, \ldots, f_n : I \to A_n$, we write $\langle\!\langle f_1, \ldots, f_n \rangle\!\rangle$ for the morphism

$$\langle\!\langle f_1, \ldots, f_n \rangle\!\rangle := \left(I \xrightarrow{\cong} I \otimes \cdots \otimes I \xrightarrow{f_1 \otimes \cdots \otimes f_n} A_1 \otimes \cdots \otimes A_n \right).$$

For example, $\langle\!\langle f_1, f_2 \rangle\!\rangle = (f_1 \otimes f_2) \circ \lambda_I^{-1}$.

Let $\mathrm{Values}(A) := \{v \mid v \text{ is a value and } \cdot \vdash v : A\}$ and $\mathrm{Programs}(A) := \{p \mid \cdot \vdash p : A\}$ for any closed type $\cdot \vdash A$. We prove adequacy using the standard method based on *formal approximation relations*, a notion first devised in [Plo85].

Lemma 7.6. *For each closed type $\cdot \vdash A$, there exist* formal approximation relations:

$$\trianglelefteq_A \subseteq (\mathbf{L}(I, [\![A]\!]) - \{\perp\}) \times \mathrm{Values}(A)$$
$$\sqsubseteq_A \subseteq \mathbf{L}(I, [\![A]\!]) \times \mathrm{Programs}(A)$$

with the properties:
(A1.1): $f \trianglelefteq_{A+B} \texttt{left } v$ *iff* $\exists f'.\ f = \mathrm{left} \circ f'$ *and* $f' \trianglelefteq_A v$;
(A1.2): $f \trianglelefteq_{A+B} \texttt{right } v$ *iff* $\exists f'.\ f = \mathrm{right} \circ f'$ *and* $f' \trianglelefteq_B v$;
(A2): $f \trianglelefteq_{A \otimes B} \langle v, w \rangle$ *iff* $\exists f', f''.\ f = \langle\!\langle f', f'' \rangle\!\rangle$ *and* $f' \trianglelefteq_A v$ *and* $f'' \trianglelefteq_B w$;
(A3): $f \trianglelefteq_{A \multimap B} \lambda x^A.\ p$ *iff* $\forall f' \in \mathbf{L}(I, [\![A]\!]), \forall v \in \mathrm{Values}(A).\ f' \trianglelefteq_A v \Rightarrow \mathrm{eval} \circ \langle\!\langle f, f' \rangle\!\rangle \sqsubseteq_B p[v/x]$;
(A4): $f \trianglelefteq_{!A} \texttt{lift } p$ *iff* $\exists g \in \mathbf{C}(1, G[\![A]\!]).\ f = I \xrightarrow{u} F(1) \xrightarrow{F(g)} ![\![A]\!]$ *and* $\epsilon \circ f \sqsubseteq_A p$;
(A5): $f \trianglelefteq_{\mu X.A} \texttt{fold } v$ *iff* $\mathrm{unfold}^{\mu X.A} \circ f \trianglelefteq_{A[\mu X.A/X]} v$;
(B): $f \sqsubseteq_A p$ *iff* $f \neq \perp \Rightarrow \exists v \in \mathrm{Values}(A).\ p \Downarrow v$ *and* $f \trianglelefteq_A v$.

Proof. Because of (A5), the relations are defined by recursion and proving their existence requires considerable effort. The full proof is presented in §7.1. □

Lemma 7.7. *If $f \trianglelefteq_P v$, where P is a non-linear type, then f is a non-linear morphism.*

Proof. By induction on $\cdot \vdash v : P$, essentially the same as Proposition 6.15. □

The next proposition is fundamental for the proof of adequacy.

Proposition 7.8. *Let $\Gamma \vdash m : A$, where $\Gamma = x_1 : A_1, \ldots, x_n : A_n$. Assume further we are given v_i and f_i, such that $f_i \trianglelefteq_{A_i} v_i$. Then $[\![m]\!] \circ \langle\!\langle f_1, \ldots, f_n \rangle\!\rangle \sqsubseteq_A m[\vec{v} \ / \ \vec{x}]$.*

Proof. By induction on $\Gamma \vdash m : A$, using Lemma 7.7 where necessary. □

The next theorem establishes sufficient conditions for termination at *any* type.

Theorem 7.9 (Termination). *Let $\cdot \vdash p : A$ be a well-typed program. If $[\![p]\!] \neq \bot$, then $p \Downarrow$.*

Proof. This is a special case of the previous proposition when $\Gamma = \cdot$. We get $[\![\cdot \vdash p : A]\!] \sqsubseteq_A p$, and thus $p \Downarrow$ by definition of $\sqsubseteq_A$. □

Theorem 7.10 (Adequacy). *Let $\cdot \vdash p : P$ be a well-typed program, with P non-linear. Then*

$$p \Downarrow \ \ iff \ \ [\![p]\!] \neq \bot \ .$$

Proof. ($\Rightarrow$) By soundness and Lemma 7.4; ($\Leftarrow$) by Theorem 7.9. □

This formulation of adequacy immediately implies another one that some readers might be more familiar with. We shall say that a closed type $\cdot \vdash P$ is *ground* if it is formed without any usage of $!$ or $\multimap$ (note that such a type is necessarily non-linear)[1]. By using soundness, adequacy, Lemma 7.4 and injectivity of $[\![-]\!]$ on values of ground type, we get:

Corollary 7.11. *Let $\cdot \vdash p : P$ and $\cdot \vdash v : P$ be a term and a value of ground type P. Then*

$$p \Downarrow v \ iff \ [\![p]\!] = [\![v]\!].$$

7.1. **Existence of the formal approximation relations.** Our proof strategy is very similar to the one in [Fio94, Chapter 9] and we now provide an overview. We define a category where the objects are pairs $(X, \trianglelefteq)$ of objects X of **L** and binary relations $\trianglelefteq \subseteq (\mathbf{L}(I, X) - \{\bot\}) \times \text{Values}(A)$ together with a suitable notion of morphism that respects these relations (Definitions 7.12 – 7.14). We show that the constructed category has sufficient structure to solve recursive domain equations (Theorem 7.17) and that the functors we use to interpret types can be lifted to this category (Theorem 7.21), while respecting the obvious forgetful functor. Then, we provide a non-standard interpretation of types that contains not only the standard interpretation of types, but also the logical relations needed for the adequacy proof (Proposition 7.26). We also show that this *augmented interpretation* is appropriate by showing that (parameterised) initial algebras are computed in the same way as for the standard interpretation (Corollary 7.27) and that it also respects substitution (Lemma 7.30). This then allows us to prove the existence of the required relations.

Definition 7.12. Given a closed type $\cdot \vdash A$, an object $X \in \text{Ob}(\mathbf{L})$ and given a relation $\trianglelefteq \subseteq (\mathbf{L}(I, X) - \{\bot\}) \times \text{Values}(A)$, we define another relation $\sqsubseteq \subseteq \mathbf{L}(I, X) \times \text{Programs}(A)$ by $g \sqsubseteq p$ iff $g \neq \bot \Rightarrow \exists v \in \text{Values}(A). \ p \Downarrow v \wedge g \trianglelefteq v$.

---

[1]To make ground types inhabited, we also have to hardcode the unit type I in the language (see Example 2.1, where we defined $I \equiv !(0 \multimap 0)$), so that I has a unique value. This is trivial and causes no problems.

Definition 7.13. For any closed type $\cdot \vdash A$, we define a **Poset**-category $\mathbf{Rel}(A)$ with:
- Objects $(X, \trianglelefteq)$, where $X \in \mathrm{Ob}(\mathbf{L})$ and $\trianglelefteq \subseteq (\mathbf{L}(I, X) - \{\bot\}) \times \mathrm{Values}(A)$ is a relation;
- Morphisms $f : (X, \trianglelefteq_X) \to (Y, \trianglelefteq_Y)$ are exactly the morphisms $f : X \to Y$ in $\mathbf{L}$, such that:

$$g \trianglelefteq_X v \Rightarrow f \circ g \sqsubseteq_Y v;$$

- Composition and identities as in $\mathbf{L}$;
- Order inherited from $\mathbf{L}$, i.e. $f \leq g$ (in $\mathbf{Rel}(A)$) iff $f \leq g$ (in $\mathbf{L}$).

Unfortunately, the category $\mathbf{Rel}(A)$ is not $\mathbf{CPO}$-enriched (in general) and therefore it is unsuitable for our purposes. Because of this, we consider a full subcategory that has the desired structure.

Definition 7.14. Let $\mathbf{R}(A)$ be the full subcategory of $\mathbf{Rel}(A)$ consisting of objects $(X, \trianglelefteq)$, such that for any $v \in \mathrm{Values}(A)$, the predicate $(- \trianglelefteq v)$ is closed under suprema of ascending chains in $\mathbf{L}(I, X) - \{\bot\}$. More specifically, if $\{g_i\}_{i=1}^{\omega}$ is an increasing sequence and $g_i \trianglelefteq v$ for every $i \in \omega$, then $\bigvee g_i \trianglelefteq v$.

Next, we list some useful properties of the categories $\mathbf{R}(A)$.

Lemma 7.15. *Let $\cdot \vdash A$ be a closed type. In the category $\mathbf{R}(A)$:*
(1) *For any object $(X, \trianglelefteq)$ and $v \in \mathrm{Values}(A)$ and $g \neq \bot$, it follows $g \trianglelefteq v$ iff $g \sqsubseteq v$.*
(2) *For any morphism $f : (X, \trianglelefteq_X) \to (Y, \trianglelefteq_Y)$, if $g \sqsubseteq_X p$ then $f \circ g \sqsubseteq_Y p$.*
(3) *For any object $(X, \trianglelefteq)$ and $p \in \mathrm{Programs}(A)$, the predicate $(- \sqsubseteq p)$ is closed under taking suprema of increasing sequences in $\mathbf{L}(I, X)$.*

Proof. **(1)** Because $v \Downarrow v$ and then by definition.

(2) Assume $f \circ g \neq \bot$. Then $g \neq \bot$ and therefore $\exists v.\ p \Downarrow v \ \wedge\ g \trianglelefteq_X v$. Since $f : (X, \trianglelefteq_X) \to (Y, \trianglelefteq_Y)$ is a morphism, it follows $f \circ g \sqsubseteq_Y v$ and therefore $f \circ g \trianglelefteq_Y v$. This now implies $f \circ g \sqsubseteq_Y p$.

(3) Let $(f_i)_{i \in \omega}$ be an increasing sequence in $\mathbf{L}(I, X)$, such that for each $i \in \omega$ we have $f_i \sqsubseteq p$. Assume $\bigvee_i f_i \neq \bot$. Then, there exists k, such that for $l \geq k : f_l \neq \bot$ and $f_l \sqsubseteq p$. Therefore, $\exists v \in \mathrm{Values}(A)$, such that $p \Downarrow v$ and $f_l \trianglelefteq v$, for every $l \geq k$. Since $(- \trianglelefteq v)$ is closed under taking suprema of increasing sequences, we conclude $\bigvee_{i \geq k} f_i \trianglelefteq v$. But then also $\bigvee_{i \geq k} f_i \sqsubseteq p$, by definition. To finish the proof, observe that $\bigvee_{i \geq 0} f_i = \bigvee_{i \geq k} f_i$. $\qquad \square$

There is an obvious forgetful functor $U^A : \mathbf{R}(A) \to \mathbf{L}$, defined by:

$$U^A(X, \trianglelefteq) := X \qquad \text{and} \qquad U^A(f) := f \tag{7.1}$$

The next lemma establishes some crucial properties of the categories $\mathbf{R}(A)$.

Lemma 7.16. *For any closed type $\cdot \vdash A$:*
(1) *The category $\mathbf{R}(A)$ is $\mathbf{CPO}$-enriched.*
(2) *$\mathbf{R}(A)$ has an ep-zero object $(0, \varnothing)$.*
(3) *$\mathbf{R}(A)$ is $\mathbf{CPO}_{\bot!}$-enriched.*
(4) *$U^A : \mathbf{R}(A) \to \mathbf{L}$ is a $\mathbf{CPO}_{\bot!}$-functor.*
(5) *$\mathbf{R}(A)$ has all ω-colimits of embeddings.*
(6) *$U_e^A : \mathbf{R}(A)_e \to \mathbf{L}_e$ is a strict ω-functor (cf. Theorem 5.8).*

Proof. **(1)** It is obvious that $\mathbf{R}(A)$ is **Poset**-enriched. We will show that suprema of ascending chains in $\mathbf{R}(A)$ exist and coincide with those in $\mathbf{L}$. Assume that $\{f_i\}_{i=1}^{\omega}$ is an increasing sequence, where $f_i : (X, \trianglelefteq_X) \to (Y, \trianglelefteq_Y)$. Assume g and v are arbitrary, such that $g \trianglelefteq_X v$.

We have to show $(\bigvee_{i\geq 0} f_i) \circ g \sqsubseteq_Y v$. Towards that end, assume $(\bigvee_{i\geq 0} f_i) \circ g \neq\perp$. Therefore there exists $k \in \omega$, such that for every $l \geq k : f_l \circ g \neq\perp$. By definition, $f_l \circ g \sqsubseteq_Y v$ and since $f_l \circ g \neq\perp$ and $v \Downarrow v$, it follows $f_l \circ g \trianglelefteq v$. Since $(- \trianglelefteq_Y v)$ is closed under suprema, it follows $\bigvee_{i\geq k} f_i \circ g = (\bigvee_{i\geq k} f_i) \circ g = (\bigvee_{i\geq 0} f_i) \circ g \trianglelefteq_Y v$ and therefore it follows $(\bigvee_{i\geq 0} f_i) \circ g \sqsubseteq_Y v$.

Finally, observe that composition in $\mathbf{R}(A)$ is a Scott-continuous operation, because it coincides with composition in $\mathbf{L}$, which is itself a Scott-continuous operation.

(2) Let $(X, \trianglelefteq_X)$ be an arbitrary object of $\mathbf{R}(A)$. $(0, \varnothing)$ is initial, because the unique $\mathbf{L}$-morphism $\perp_{0,X} \colon 0 \to X$ is also a $\mathbf{R}(A)$-morphism (the additional required condition is trivially satisfied for the empty relation). Moreover, $(0, \varnothing)$ is terminal, because the unique $\mathbf{L}$-morphism $\perp_{X,0} \colon X \to 0$ is also a $\mathbf{R}(A)$-morphism (if $g \trianglelefteq_X v$ then $\perp_{X,0} \circ g = \perp_{I,0} \sqsubseteq_\varnothing v$, where $\sqsubseteq_\varnothing$ is constructed via Definition 7.12 when $\trianglelefteq = \varnothing$). Finally, $\perp_{X,0} \circ \perp_{0,X} = \perp_{0,0} = \mathrm{id}_0$ and $\perp_{0,X} \circ \perp_{X,0} = \perp_{X,X} \leq \mathrm{id}_X$ because the order in $\mathbf{R}(A)$ is inherited from $\mathbf{L}$. Thus, $(0, \varnothing)$ is an ep-zero.

(3) Combine (1), (2) and the fact that $\perp_{X,Y}$ is least in its hom-cpo.

(4) True by construction (the order and suprema in $\mathbf{R}(A)$ coincide with those in $\mathbf{L}$).

(5) Let $D = ((X_k, \trianglelefteq_k), e_k)_{k\in\omega}$ be an ω-diagram, s.t. each $e_k : (X_k, \trianglelefteq_k) \to (X_{k+1}, \trianglelefteq_{k+1})$ is an embedding. Let $\mu : UD \to X_\omega$ be the colimiting cocone in $\mathbf{L}$ of the ω-diagram $UD = (X_k, e_k)_{k\in\omega}$. Define a relation $\trianglelefteq_\omega \subseteq (\mathbf{L}(I, X_\omega) - \{\perp\}) \times \mathrm{Values}(A)$ by:

$$f \trianglelefteq_\omega v \text{ iff } \forall k \in \omega.\ \mu_k^\bullet \circ f \sqsubseteq_k v,$$

where $\mu_k : X_k \to X_\omega$ is the colimiting embedding and $\mu_k^\bullet : X_\omega \to X_k$ is its corresponding projection. We will show that $(X_\omega, \trianglelefteq_\omega)$ is the colimit of D in $\mathbf{R}(A)$ with colimiting cocone given by μ.

We have to show $(- \trianglelefteq_\omega v)$ is closed under suprema of ascending chains, for arbitrary v. Let $(f_i)_{i\in\omega}$ be an increasing sequence, s.t. $f_i \trianglelefteq_\omega v$. Then $\forall k.\forall i.\ \mu_k^\bullet \circ f_i \sqsubseteq_k v$. But $(- \sqsubseteq_k v)$ is closed under suprema of increasing sequences, thus $\forall k.\ \bigvee_i \mu_k^\bullet \circ f_i = \mu_k^\bullet \circ \bigvee_i f_i \sqsubseteq_k v$. Then, by definition $\bigvee_i f_i \trianglelefteq_\omega v$.

Next, we show that $\mu_k : (X_k, \trianglelefteq_k) \to (X_\omega, \trianglelefteq_\omega)$ is a morphism of $\mathbf{R}(A)$. Towards that end, assume that $g \trianglelefteq_k v$. We have to show $\mu_k \circ g \sqsubseteq_\omega v$. To do this, assume further $\mu_k \circ g \neq\perp$. Now, $\mu_k \circ g \sqsubseteq_\omega v$ is equivalent to $\mu_k \circ g \trianglelefteq_\omega v$, which is equivalent to $\forall l.\ \mu_l^\bullet \circ \mu_k \circ g \sqsubseteq_l v$. For $l \leq k$, we have

$$\mu_l^\bullet \circ \mu_k \circ g = e_{l,k}^\bullet \circ \mu_k^\bullet \circ \mu_k \circ g = e_{l,k}^\bullet \circ g$$

and since $e_{l,k}^\bullet : (X_k, \trianglelefteq_k) \to (X_l, \trianglelefteq_l)$ is a morphism, then it follows $e_{l,k}^\bullet \circ g \sqsubseteq_l v$. For $l \geq k$, we have:

$$\mu_l^\bullet \circ \mu_k \circ g = \mu_l^\bullet \circ \mu_l \circ e_{k,l} \circ g = e_{k,l} \circ g$$

and since $e_{k,l} : (X_k, \trianglelefteq_k) \to (X_l, \trianglelefteq_l)$ is a morphism, then it follows $e_{k,l} \circ g \sqsubseteq_l v$. Combining both cases, we conclude that $\mu_k : (X_k, \trianglelefteq_k) \to (X_\omega, \trianglelefteq_\omega)$ is a morphism of $\mathbf{R}(A)$.

Next, observe that $\mu_k^\bullet : (X_\omega, \trianglelefteq_\omega) \to (X_k, \trianglelefteq_k)$ is a morphism of $\mathbf{R}(A)$, by definition of $\trianglelefteq_\omega$.

Therefore, $\mu : D \to (X_\omega, \trianglelefteq_\omega)$ is a cocone over embeddings in $\mathbf{R}(A)$. Clearly, $(\mu_i^\bullet \circ \mu_i)_{i\in\omega}$ forms an increasing sequence of morphisms in $\mathbf{R}(A)$ and $\bigvee_i \mu_i^\bullet \circ \mu_i = \mathrm{id}$, because the order and suprema in $\mathbf{R}(A)$ coincide with those of $\mathbf{L}$. By the limit-colimit coincidence theorem [SP82], it follows $\mu : D \to (X_\omega, \trianglelefteq_\omega)$ is a colimiting cocone, as required.

(6) Combining (4), (5) and Theorem 5.8 shows U_e^A is an ω-functor. Strictness is obvious. $\qquad\square$

Note that by the above lemma, the categories $\mathbf{R}(A)_e$ are ω-categories. Even better, we can show they are suitable categories for interpreting recursive types.

Theorem 7.17. *Let $\cdot \vdash A$ be a closed type. Then $\mathbf{R}(A)$ is $\mathbf{CPO}$-algebraically compact.*

Proof. Combine Theorem 5.7 with Lemma 7.16 (1) (2) (5). ∎

Next, we show that the functors we use to interpret type expressions may be defined on the categories $\mathbf{R}(A)$ by extending their action to relations as well.

Definition 7.18. Let $\cdot \vdash A$ and $\cdot \vdash B$ be two closed types. We define functors as follows:
(1) $!^A : \mathbf{R}(A) \to \mathbf{R}(!A)$ given by:

$$!^A(X, \trianglelefteq) := (!X, !\trianglelefteq) \qquad \text{and} \qquad !^A f := !f,$$

where the relation $(! \trianglelefteq) \subseteq (\mathbf{L}(I, !X) - \{\bot\}) \times \text{Values}(!A)$ is given by:

$$g \ (! \trianglelefteq) \ \texttt{lift} \ p \quad \text{iff} \quad \exists h \in \mathbf{C}(1, GX). \ g = I \xrightarrow{u} F(1) \xrightarrow{F(h)} !X \text{ and } \epsilon \circ g \sqsubseteq p$$

(2) $\otimes^{A,B} : \mathbf{R}(A) \times \mathbf{R}(B) \to \mathbf{R}(A \otimes B)$ given by :

$$(X, \trianglelefteq_X) \otimes^{A,B} (Y, \trianglelefteq_Y) := (X \otimes Y, \trianglelefteq_X \otimes \trianglelefteq_Y) \qquad \text{and} \qquad f \otimes^{A,B} h := f \otimes h,$$

where the relation $(\trianglelefteq_X \otimes \trianglelefteq_Y) \subseteq (\mathbf{L}(I, X \otimes Y) - \{\bot\}) \times \text{Values}(A \otimes B)$ is given by:

$$g \ (\trianglelefteq_X \otimes \trianglelefteq_Y) \ \langle v, w \rangle \quad \text{iff} \quad \exists g', g''. \ g = (g' \otimes g'') \circ \lambda_I^{-1} \ \wedge \ g' \trianglelefteq_X v \ \wedge \ g'' \trianglelefteq_Y w$$

(3) $\multimap^{A,B} : \mathbf{R}(A)^{\mathrm{op}} \times \mathbf{R}(B) \to \mathbf{R}(A \multimap B)$ given by :

$$(X, \trianglelefteq_X) \multimap^{A,B} (Y, \trianglelefteq_Y) := (X \multimap Y, \trianglelefteq_X \multimap \trianglelefteq_Y) \qquad \text{and} \qquad f^{\mathrm{op}} \multimap^{A,B} h := f^{\mathrm{op}} \multimap h,$$

where the relation $(\trianglelefteq_X \multimap \trianglelefteq_Y) \subseteq (\mathbf{L}(I, X \multimap Y) - \{\bot\}) \times \text{Values}(A \multimap B)$ is given by:

$$g \ (\trianglelefteq_X \multimap \trianglelefteq_Y) \ \lambda x.p \quad \text{iff} \quad \forall g' \in \mathbf{L}(I, X), \forall v \in \text{Values}(A) : g' \trianglelefteq_X v \Rightarrow \text{eval} \circ \langle\!\langle g, g' \rangle\!\rangle \sqsubseteq_Y p[v/x]$$

(4) $+^{A,B} : \mathbf{R}(A) \times \mathbf{R}(B) \to \mathbf{R}(A + B)$ given by :

$$(X, \trianglelefteq_X) +^{A,B} (Y, \trianglelefteq_Y) := (X + Y, \trianglelefteq_X + \trianglelefteq_Y) \qquad \text{and} \qquad f +^{A,B} h := f + h,$$

where the relation $(\trianglelefteq_X + \trianglelefteq_Y) \subseteq (\mathbf{L}(I, X + Y) - \{\bot\}) \times \text{Values}(A + B)$ is given by:

$$g \ (\trianglelefteq_X + \trianglelefteq_Y) \ \texttt{left} \ v \quad \text{iff} \quad \exists g'. \ g = \text{left} \circ g' \ \wedge \ g' \trianglelefteq_X v$$

and

$$g \ (\trianglelefteq_X + \trianglelefteq_Y) \ \texttt{right} \ v \quad \text{iff} \quad \exists g'. \ g = \text{right} \circ g' \ \wedge \ g' \trianglelefteq_Y v$$

Before we prove our next theorem, we need a couple of lemmas that allow us to establish some order-reflection properties of our models that will be used in its proof.

Lemma 7.19. *The coproduct injections in $\mathbf{L}$ are split monomorphisms.*

Proof. Let $A \xrightarrow{\text{left}} A + B \xleftarrow{\text{right}} B$ be two coproduct injections. Since the category $\mathbf{L}$ is pointed:

$$[\text{id}_A, \bot_{B,A}] \circ \text{left} = \text{id}_A \qquad \text{and} \qquad [\bot_{A,B}, \text{id}_B] \circ \text{right} = \text{id}_B. \qquad \qquad \square$$

Lemma 7.20 (Order-reflection via adjunction)**.** *Let $f : X \to GA$ and $h : X \to GA$ be two parallel morphisms in $\mathbf{C}$, where $A \in Ob(\mathbf{L})$ and $X \in Ob(\mathbf{C})$. Then $F(f) \leq F(h)$ iff $f \leq h$.*

Proof. The right-to-left direction follows because F is a **CPO**-functor. For the other direction:

$$F(f) \leq F(h) \Rightarrow G\epsilon \circ GF(f) \circ \eta \leq G\epsilon \circ GF(h) \circ \eta \quad (G \text{ and } (- \circ -) \text{ are Scott-continuous})$$

$$\Leftrightarrow G\epsilon \circ \eta \circ f \leq G\epsilon \circ \eta \circ h \qquad\qquad\qquad\qquad\qquad (\text{Naturality of } \eta)$$

$$\Leftrightarrow f \leq h \qquad\qquad\qquad\qquad\qquad\qquad\qquad (\text{counit-unit equation}) \;\square$$

Theorem 7.21. *All functors from Definition 7.18 are well-defined and are moreover* **CPO**-*functors. In addition, the following diagrams:*

$$
\begin{array}{ccc}
\mathbf{R}(A)_e \times \mathbf{R}(B)_e & \xrightarrow{\;\star_e^{A,B}\;} & \mathbf{R}(A \star B)_e \\
{\scriptstyle U_e^A \times U_e^B}\downarrow & & \downarrow{\scriptstyle U_e^{A\star B}} \\
\mathbf{L}_e \times \mathbf{L}_e & \xrightarrow[\;\star_e\;]{} & \mathbf{L}_e
\end{array}
\qquad\qquad
\begin{array}{ccc}
\mathbf{R}(A)_e & \xrightarrow{\;!_e^A\;} & \mathbf{R}(!A)_e \\
{\scriptstyle U_e^A}\downarrow & & \downarrow{\scriptstyle U_e^{!A}} \\
\mathbf{L}_e & \xrightarrow[\;!_e\;]{} & \mathbf{L}_e
\end{array}
$$

commute, where $\star \in \{+, \otimes, \multimap\}$.

Proof. **Case ! :**

We first show $!^A$ is a well-defined functor. We begin by showing that $(- \; (! \trianglelefteq) \; \mathtt{lift} \; p)$ is closed under suprema of increasing sequences. Let $(g_i)_{i \in \omega}$ be an increasing sequence, such that $g_i \; (! \trianglelefteq) \; \mathtt{lift} \; p$. Then there exist $h_i \in \mathbf{C}(1, GX)$, such that $g_i = F(h_i) \circ u$ and $\epsilon \circ g_i \sqsubseteq p$. By Lemma 7.15 (3) it follows $\epsilon \circ \bigvee_i g_i = \bigvee_i \epsilon \circ g_i \sqsubseteq p$. Since u is an isomorphism and $(g_i)_{i \in \omega}$ forms an increasing sequence, then $(Fh_i)_{i \in \omega}$ also forms an increasing sequence. The functor F reflects the order of this sequence (Lemma 7.20) and therefore $(h_i)_{i \in \omega}$ also forms an increasing sequence. Moreover, $\bigvee_i g_i = \bigvee_i F(h_i) \circ u = (\bigvee_i F(h_i)) \circ u = F(\bigvee_i h_i) \circ u$ and it follows by definition that $\bigvee_i g_i \; (! \trianglelefteq) \; \mathtt{lift} \; p$. Therefore $!^A$ is well-defined on objects.

Next, we show $!^A$ is well-defined on morphisms. Let $f : (X, \trianglelefteq_X) \to (Y, \trianglelefteq_Y)$ be a morphism in $\mathbf{R}(A)$. Assume that $g \; (! \trianglelefteq_X) \; \mathtt{lift} \; p$ and that $!f \circ g \neq \bot$. We have to show $!f \circ g \; (! \trianglelefteq_Y) \; \mathtt{lift} \; p$. Then $\exists h. \; g = Fh \circ u$ and $\epsilon \circ g \sqsubseteq_X p$. We have

$$!f \circ g = FGf \circ Fh \circ u = F(Gf \circ h) \circ u$$

and also

$$\epsilon \circ !f \circ g = f \circ \epsilon \circ g \sqsubseteq_Y p$$

because of Lemma 7.15 (2). Therefore, by definition $!f \circ g \; (! \trianglelefteq_Y) \; \mathtt{lift} \; p$. Therefore $!^A$ is well-defined on morphisms.

Finally, it immediately follows by definition of $!^A$ that it preserves identities and composition, that it is a **CPO**-functor and that the required diagram commutes.

Case $\star = \otimes$:

We first show $\otimes^{A,B}$ is a well-defined functor. We show $(- \; (\trianglelefteq_X \otimes \trianglelefteq_Y) \; \langle v, w \rangle)$ is closed under suprema of increasing sequences. Let $(g_i)_{i \in \omega}$ be an increasing sequence, such that $g_i \; (\trianglelefteq_X \otimes \trianglelefteq_Y) \; \langle v, w \rangle$. Then there exist g_i', g_i'', such that $g_i = (g_i' \otimes g_i'') \circ \lambda_I^{-1}$ and such that $g_i' \trianglelefteq_X v$ and $g_i'' \trianglelefteq_Y w$. Since λ_I^{-1} is an isomorphism, $(g_i' \otimes g_i'')_{i \in \omega}$ is an increasing sequence and by Definition 7.1(2) it follows that both $(g_i')_{i \in \omega}$ and $(g_i'')_{i \in \omega}$ are increasing sequences. Therefore $\bigvee_i g_i' \trianglelefteq_X v$ and $\bigvee_i g_i'' \trianglelefteq_Y w$. Also:

$$\bigvee_i g_i = \bigvee_i (g_i' \otimes g_i'') \circ \lambda_I^{-1} = ((\bigvee_i g_i') \otimes (\bigvee_i g_i'')) \circ \lambda_I^{-1}$$

so that by definition $\bigvee_i g_i \; (\trianglelefteq_X \otimes \trianglelefteq_Y) \; \langle v, w \rangle$. Therefore $\otimes^{A,B}$ is well-defined on objects.

Next, we show $\otimes^{A,B}$ is well-defined on morphisms. Let

$$f_1 : (X_1, \trianglelefteq_{X_1}) \to (Y_1, \trianglelefteq_{Y_1})$$
$$f_2 : (X_2, \trianglelefteq_{X_2}) \to (Y_2, \trianglelefteq_{Y_2})$$

be two morphisms in $\mathbf{R}(A)$ and $\mathbf{R}(B)$ respectively. Assume that $g \ (\trianglelefteq_{X_1} \otimes \trianglelefteq_{X_2}) \ \langle v, w \rangle$ and that $(f_1 \otimes f_2) \circ g \neq \bot$. We have to show $(f_1 \otimes f_2) \circ g \ (\trianglelefteq_{Y_1} \otimes \trianglelefteq_{Y_2}) \ \langle v, w \rangle$. Then there exist g', g'', such that $g = (g' \otimes g'') \circ \lambda_I^{-1}$ and such that $g' \trianglelefteq_{X_1} v$ and $g'' \trianglelefteq_{X_2} w$. Therefore, $f_1 \circ g' \sqsubseteq_{Y_1} v$ and $f_2 \circ g'' \sqsubseteq_{Y_2} w$. It's easy to see that $f_1 \neq \bot \neq g'$ and $f_2 \neq \bot \neq g''$. Therefore, $f_1 \circ g' \trianglelefteq_{Y_1} v$ and $f_2 \circ g'' \trianglelefteq_{Y_2} w$. Also:

$$(f_1 \otimes f_2) \circ g = (f_1 \otimes f_2) \circ (g' \otimes g'') \circ \lambda_I^{-1} = ((f_1 \circ g') \otimes (f_2 \circ g'')) \circ \lambda_I^{-1}$$

By definition, $(f_1 \otimes f_2) \circ g \ (\trianglelefteq_{Y_1} \otimes \trianglelefteq_{Y_2}) \ \langle v, w \rangle$. Therefore, $\otimes^{A,B}$ is well-defined on morphisms.

Finally, it immediately follows by definition of $\otimes^{A,B}$ that it preserves identities and composition, that it is a **CPO**-functor and that the required diagram commutes.

Case $\star = \multimap$:

We first show $\multimap^{A,B}$ is a well-defined functor. We show $(- \ (\trianglelefteq_X \multimap \trianglelefteq_Y) \ \lambda x.p)$ is closed under suprema of increasing sequences. Let $(g_i)_{i \in \omega}$ be an increasing sequence, such that $g_i \ (\trianglelefteq_X \multimap \trianglelefteq_Y) \ \lambda x.p$. Let g' and v be arbitrary, such that $g' \trianglelefteq_X v$. It now follows that $\mathrm{eval} \circ \langle\!\langle g_i, g' \rangle\!\rangle \sqsubseteq_Y p[v/x]$. From Lemma 7.15 we get:

$$\mathrm{eval} \circ \langle\!\langle \bigvee_i g_i, g' \rangle\!\rangle = \bigvee_i \mathrm{eval} \circ \langle\!\langle g_i, g' \rangle\!\rangle \sqsubseteq_Y p[v/x]$$

so that by definition $\bigvee_i g_i \ (\trianglelefteq_X \multimap \trianglelefteq_Y) \ \lambda x.p$. Therefore $\multimap^{A,B}$ is well-defined on objects.

Next, we show $\multimap^{A,B}$ is well-defined on morphisms. Let

$$f_1^{\mathrm{op}} : (X_1, \trianglelefteq_{X_1}) \to (Y_1, \trianglelefteq_{Y_1})$$
$$f_2 : (X_2, \trianglelefteq_{X_2}) \to (Y_2, \trianglelefteq_{Y_2})$$

be two morphisms in $\mathbf{R}(A)^{\mathrm{op}}$ and $\mathbf{R}(B)$ respectively. Assume that $g \ (\trianglelefteq_{X_1} \multimap \trianglelefteq_{X_2}) \ \lambda x.p$ and that $(f_1^{\mathrm{op}} \multimap f_2) \circ g \neq \bot$. We have to show $(f_1^{\mathrm{op}} \multimap f_2) \circ g \ (\trianglelefteq_{Y_1} \multimap \trianglelefteq_{Y_2}) \ \lambda x.p$. Let g' and v be arbitrary such that $g' \trianglelefteq_{Y_1} v$. Since

$$f_1 : (Y_1, \trianglelefteq_{Y_1}) \to (X_1, \trianglelefteq_{X_1})$$

it follows $f_1 \circ g' \sqsubseteq_{X_1} v$.

If $f_1 \circ g' = \bot$, then it trivially follows that $\mathrm{eval} \circ \langle\!\langle g, f_1 \circ g' \rangle\!\rangle \sqsubseteq_{X_2} p[v/x]$.

If $f_1 \circ g' \neq \bot$, then $f_1 \circ g' \trianglelefteq_{X_1} v$ and therefore $\mathrm{eval} \circ \langle\!\langle g, f_1 \circ g' \rangle\!\rangle \sqsubseteq_{X_2} p[v/x]$.

Therefore, in both cases, $\mathrm{eval} \circ \langle\!\langle g, f_1 \circ g' \rangle\!\rangle \sqsubseteq_{X_2} p[v/x]$. This then implies:

$$f_2 \circ \mathrm{eval} \circ \langle\!\langle g, f_1 \circ g' \rangle\!\rangle \sqsubseteq_{Y_2} p[v/x],$$

because $f_2 : (X_2, \trianglelefteq_{X_2}) \to (Y_2, \trianglelefteq_{Y_2})$. And therefore:

$$\begin{aligned}
&\text{eval} \circ \langle\!\langle (f_1^{\mathrm{op}} \multimap f_2) \circ g, g' \rangle\!\rangle = \\
&= \text{eval} \circ (((f_1^{\mathrm{op}} \multimap f_2) \circ g) \otimes g') \circ \lambda_I^{-1} \\
&= \text{eval} \circ ((f_1^{\mathrm{op}} \multimap f_2) \otimes \text{id}) \circ (g \otimes g') \circ \lambda_I^{-1} \\
&= \text{eval} \circ ((\text{id}^{\mathrm{op}} \multimap f_2) \otimes \text{id}) \circ ((f_1^{\mathrm{op}} \multimap \text{id}) \otimes \text{id}) \circ (g \otimes g') \circ \lambda_I^{-1} \\
&= f_2 \circ \text{eval} \circ ((f_1^{\mathrm{op}} \multimap \text{id}) \otimes \text{id}) \circ (g \otimes g') \circ \lambda_I^{-1} \qquad \text{(Naturality of eval)} \\
&= f_2 \circ \text{eval} \circ (\text{id} \otimes f_1) \circ (g \otimes g') \circ \lambda_I^{-1} \qquad \text{(Parameterised adjunction [ML98, pp. 102])} \\
&= f_2 \circ \text{eval} \circ \langle\!\langle g, f_1 \circ g' \rangle\!\rangle \\
&\sqsubseteq_{Y_2} p[v/x]
\end{aligned}$$

By definition, $(f_1^{\mathrm{op}} \multimap f_2) \circ g \ (\trianglelefteq_{Y_1} \multimap \trianglelefteq_{Y_2}) \ \lambda x.p$. Therefore, $\multimap^{A,B}$ is well-defined on morphisms.

Finally, it immediately follows by definition of $\multimap^{A,B}$ that it preserves identities and composition, that it is a **CPO**-functor and that the required diagram commutes.

Case $\star = +$:

We first show $+^{A,B}$ is a well-defined functor. We show $(- \ (\trianglelefteq_X + \trianglelefteq_Y) \ \mathtt{left} \ v)$ is closed under suprema of increasing sequences. Let $(g_i)_{i \in \omega}$ be an increasing sequence, such that $g_i \ (\trianglelefteq_X + \trianglelefteq_Y) \ \mathtt{left} \ v$. Then there exist g_i', such that $g_i = \text{left} \circ g_i' \wedge g_i' \trianglelefteq_X v$. Since left is a split monomorphism (Lemma 7.19), it follows $(g_i')_{i \in \omega}$ is an increasing sequence and therefore $\bigvee_i g_i' \trianglelefteq_X v$. Since $\bigvee_i g_i = \text{left} \circ \bigvee_i g_i'$ then by definition $\bigvee_i g_i \ (\trianglelefteq_X + \trianglelefteq_Y) \ \mathtt{left} \ v$. Similarly, it follows that $(- \ (\trianglelefteq_X + \trianglelefteq_Y) \ \mathtt{right} \ v)$ is also closed under suprema of increasing sequences. Therefore $+^{A,B}$ is well-defined on objects.

Next, we show $+^{A,B}$ is well-defined on morphisms. Let

$$\begin{aligned}
f_1 &: (X_1, \trianglelefteq_{X_1}) \to (Y_1, \trianglelefteq_{Y_1}) \\
f_2 &: (X_2, \trianglelefteq_{X_2}) \to (Y_2, \trianglelefteq_{Y_2})
\end{aligned}$$

be two morphisms in $\mathbf{R}(A)$ and $\mathbf{R}(B)$ respectively. Assume that $g \ (\trianglelefteq_{X_1} + \trianglelefteq_{X_2}) \ v$ and that $(f_1 + f_2) \circ g \neq \bot$. We have to show $(f_1 + f_2) \circ g \ (\trianglelefteq_{Y_1} + \trianglelefteq_{Y_2}) \ v$. There are two cases for v: either $v = \mathtt{left} \ w$ or $v = \mathtt{right} \ w$. We show the first one and the other one follows by complete analogy. Thus, assume $v = \mathtt{left} \ w$. Then $\exists g'. \ g = \text{left} \circ g' \wedge g' \trianglelefteq_{X_1} w$. Therefore, $f_1 \circ g' \sqsubseteq_{Y_1} w$. Since $\text{left} \circ f_1 \circ g' = (f_1 + f_2) \circ \text{left} \circ g' = (f_1 + f_2) \circ g \neq \bot$ it follows $f_1 \circ g' \neq \bot$ and therefore $f_1 \circ g' \trianglelefteq_{Y_1} w$. By definition, $(f_1 + f_2) \circ g \ (\trianglelefteq_{Y_1} + \trianglelefteq_{Y_2}) \ \mathtt{left} \ w$. Therefore, $+^{A,B}$ is well-defined on morphisms.

Finally, it immediately follows by definition of $+^{A,B}$ that it preserves identities and composition, that it is a **CPO**-functor and that the required diagram commutes. $\qquad \square$

Remark 7.22. In the proof above, condition (2) of Definition 7.1 is used to show $\otimes^{A,B}$ is well-defined. This is the only place where this assumption is used in the paper.

So far, we have defined operations (and functors) that show how to construct appropriate logical relations for types of the form $!A, A + B, A \otimes B$ and $A \multimap B$. Next, we show how to fold and unfold logical relations via the functors $\mathbb{I}$ and $\mathbb{E}$ below. $\mathbb{I}$ should be thought of as the *introduction* (fold) and $\mathbb{E}$ as the *elimination* (unfold) for logical relations involving recursive types.

Lemma 7.23. *For every closed type $\cdot \vdash \mu X.A$, there exists an isomorphism of categories*

$$\mathbb{I}^{\mu X.A} : \mathbf{R}(A[\mu X.A/X]) \cong \mathbf{R}(\mu X.A) : \mathbb{E}^{\mu X.A}$$

given by:

$$\mathbb{I}^{\mu X.A} : \mathbf{R}(A[\mu X.A/X]) \to \mathbf{R}(\mu X.A) \qquad \mathbb{E}^{\mu X.A} : \mathbf{R}(\mu X.A) \to \mathbf{R}(A[\mu X.A/X])$$

$$\mathbb{I}^{\mu X.A}(Y, \trianglelefteq) = (Y, \mathbb{I}^{\mu X.A} \trianglelefteq) \qquad\qquad \mathbb{E}^{\mu X.A}(Y, \trianglelefteq) = (Y, \mathbb{E}^{\mu X.A} \trianglelefteq)$$

$$\mathbb{I}^{\mu X.A}(f) = f \qquad\qquad\qquad\qquad\qquad \mathbb{E}^{\mu X.A}(f) = f$$

where the folded relation $(\mathbb{I}^{\mu X.A} \trianglelefteq)$ is defined by:

$$(\mathbb{I}^{\mu X.A} \trianglelefteq) \subseteq (\mathbf{L}(I, Y) - \{\bot\}) \times \textit{Values}(\mu X.A)$$

$$g \; (\mathbb{I}^{\mu X.A} \trianglelefteq) \; \texttt{fold } v \quad \Longleftrightarrow \quad g \trianglelefteq v$$

and the unfolded relation $(\mathbb{E}^{\mu X.A} \trianglelefteq)$ is defined by:

$$(\mathbb{E}^{\mu X.A} \trianglelefteq) \subseteq (\mathbf{L}(I, Y) - \{\bot\}) \times \textit{Values}(A[\mu X.A/X])$$

$$g \; (\mathbb{E}^{\mu X.A} \trianglelefteq) \; v \quad \Longleftrightarrow \quad g \trianglelefteq \texttt{fold } v$$

Moreover, the functors $\mathbb{I}^{\mu X.A}$ and $\mathbb{E}^{\mu X.A}$ are $\mathbf{CPO}_{\bot!}$-functors.

Proof. We first show that the functor $\mathbb{I}^{\mu X.A}$ is well-defined on objects. This is equivalent to showing $(- \; (\mathbb{I}^{\mu X.A} \trianglelefteq) \; \texttt{fold } v)$ is closed under suprema of increasing sequences in $\mathbf{L}(I, Y) - \{\bot\}$. This, by definition, is equivalent to showing that $(- \; \trianglelefteq \; v)$ is also closed under the same suprema, which is true by assumption.

Next, we show the functor $\mathbb{I}^{\mu X.A}$ is well-defined on morphisms. Let $f : (Y, \trianglelefteq_Y) \to (Z, \trianglelefteq_Z)$ be a morphism in $\mathbf{R}(A[\mu X.A/X])$. Assume $g \; (\mathbb{I}^{\mu X.A} \trianglelefteq_Y) \; \texttt{fold } v$ and assume $f \circ g \neq \bot$. From the former it follows $g \trianglelefteq_Y v$ and therefore $f \circ g \sqsubseteq_Z v$. Since $f \circ g \neq \bot$ it follows $f \circ g \trianglelefteq_Z v$ and by definition $f \circ g \; (\mathbb{I}^{\mu X.A} \trianglelefteq_Z) \; \texttt{fold } v$. Therefore, $\mathbb{I}^{\mu X.A}$ is well-defined on morphisms.

Next, we show that the functor $\mathbb{E}^{\mu X.A}$ is well-defined on objects. This is equivalent to showing $(- \; (\mathbb{E}^{\mu X.A} \trianglelefteq) \; v)$ is closed under suprema of increasing sequences in $\mathbf{L}(I, Y) - \{\bot\}$. This, by definition, is equivalent to showing that $(- \; \trianglelefteq \; \texttt{fold } v)$ is also closed under the same suprema, which is true by assumption.

Next, we show $\mathbb{E}^{\mu X.A}$ is well-defined on morphisms. Let $f : (Y, \trianglelefteq_Y) \to (Z, \trianglelefteq_Z)$ be a morphism in $\mathbf{R}(\mu X.A)$. Assume $g \; (\mathbb{E}^{\mu X.A} \trianglelefteq_Y) \; v$ and assume $f \circ g \neq \bot$. From the former it follows $g \trianglelefteq_Y \texttt{fold } v$ and therefore $f \circ g \sqsubseteq_Z \texttt{fold } v$. Since $f \circ g \neq \bot$ it follows $f \circ g \trianglelefteq_Z \texttt{fold } v$ and by definition it follows $f \circ g \; (\mathbb{E}^{\mu X.A} \trianglelefteq_Z) \; v$. Therefore, $\mathbb{E}^{\mu X.A}$ is well-defined on morphisms.

Obviously, both functors preserve identities and composition.

Next, we show that for any relation $\trianglelefteq$, we have that $\mathbb{I}^{\mu X.A}(\mathbb{E}^{\mu X.A} \trianglelefteq) = \trianglelefteq$. This is true, because:

$$g \; (\mathbb{I}^{\mu X.A}(\mathbb{E}^{\mu X.A} \trianglelefteq)) \; \texttt{fold } v \quad \Longleftrightarrow \quad g \; (\mathbb{E}^{\mu X.A} \trianglelefteq) \; v \quad \Longleftrightarrow \quad g \trianglelefteq \texttt{fold } v.$$

Also, we show that for any relation $\trianglelefteq$, we have that $\mathbb{E}^{\mu X.A}(\mathbb{I}^{\mu X.A} \trianglelefteq) = \trianglelefteq$. This is true, because:

$$g \; (\mathbb{E}^{\mu X.A}(\mathbb{I}^{\mu X.A} \trianglelefteq)) \; v \quad \Longleftrightarrow \quad g \; (\mathbb{I}^{\mu X.A} \trianglelefteq) \; \texttt{fold } v \quad \Longleftrightarrow \quad g \trianglelefteq v.$$

Therefore $\mathbb{I}^{\mu X.A} \circ \mathbb{E}^{\mu X.A} = \mathrm{Id}_{\mathbf{R}(\mu X.A)}$ and $\mathbb{E}^{\mu X.A} \circ \mathbb{I}^{\mu X.A} = \mathrm{Id}_{\mathbf{R}(A[\mu X.A/X])}$, which shows the two functors are isomorphisms. Since both functors are identity on morphisms, it follows that they are both $\mathbf{CPO}_{\bot!}$-functors. $\qquad\square$

This finishes the categorical development of the categories $\mathbf{R}(A)$. Next, we define a *non-standard type interpretation* on the categories $\mathbf{R}(A)_e$. When we compose this interpretation with the forgetful functor, we get exactly the standard type interpretation (Proposition 7.26). Since the new interpretation carries additional information (the logical relations needed for the proof of adequacy), we shall refer to it as the *augmented interpretation*.

Notation 7.24. In what follows, given a type context $\Theta = X_1, \ldots, X_n$ and closed types $\cdot \vdash C_i$ $(1 \le i \le n)$, we write $\vec{C}$ for $C_1, \ldots, C_n$ and $[\vec{C}/\Theta]$ for $[C_1/X_1, \ldots, C_n/X_n]$.

Definition 7.25. Given a well-formed type $\Theta \vdash A$ and closed types $\vec{C}$ as in Notation 7.24, we define the *augmented interpretation* to be the functor

$$\|\Theta \vdash A\|^{\vec{C}} : \mathbf{R}(C_1)_e \times \cdots \times \mathbf{R}(C_n)_e \to \mathbf{R}(A[\vec{C}/\Theta])_e$$

defined inductively by:

$$\|\Theta \vdash \Theta_i\|^{\vec{C}} := \Pi_i$$

$$\|\Theta \vdash !A\|^{\vec{C}} := !_e^{A[\vec{C}/\Theta]} \circ \|\Theta \vdash A\|^{\vec{C}}$$

$$\|\Theta \vdash A \star B\|^{\vec{C}} := \star_e^{A[\vec{C}/\Theta], B[\vec{C}/\Theta]} \circ \langle \|\Theta \vdash A\|^{\vec{C}}, \|\Theta \vdash B\|^{\vec{C}} \rangle \qquad (\text{where } \star \in \{+, \otimes, \multimap\})$$

$$\|\Theta \vdash \mu X.A\|^{\vec{C}} := \left(\mathbb{I}_e^{\mu X.A[\vec{C}/\Theta]} \circ \|\Theta, X \vdash A\|^{\vec{C}, \mu X.A[\vec{C}/\Theta]} \right)^{\dagger}$$

Proposition 7.26. *Every functor* $\|\Theta \vdash A\|^{\vec{C}}$ *is an ω-functor and the following diagram:*

$$
\begin{array}{ccc}
\mathbf{R}(C_1)_e \times \cdots \times \mathbf{R}(C_n)_e & \xrightarrow{\|\Theta \vdash A\|^{\vec{C}}} & \mathbf{R}(A[\vec{C}/\Theta])_e \\
\Big\downarrow{\scriptstyle U_e^{C_1} \times \cdots \times U_e^{C_n}} & & \Big\downarrow{\scriptstyle U_e^{A[\vec{C}/\Theta]}} \\
\mathbf{L}_e \times \cdots \times \mathbf{L}_e & \xrightarrow[{[\![\Theta \vdash A]\!]}]{} & \mathbf{L}_e
\end{array}
$$

commutes.

Proof. By induction on $\Theta \vdash A$.

    Case $\Theta \vdash \Theta_i$. Obvious.

    Case $\Theta \vdash !A$.

By induction $\|\Theta \vdash A\|^{\vec{C}}$ is an ω-functor. ω-functors are closed under composition and $!_e^{A[\vec{C}/\Theta]}$ is an ω-functor because of Theorem 5.8 and Theorem 7.21. Therefore $\|\Theta \vdash !A\|^{\vec{C}}$ is an ω-functor. Moreover:

$$U_e^{!A[\vec{C}/\Theta]} \circ \|\Theta \vdash !A\|^{\vec{C}} = U_e^{!A[\vec{C}/\Theta]} \circ !_e^{A[\vec{C}/\Theta]} \circ \|\Theta \vdash A\|^{\vec{C}} \qquad \text{(Definition)}$$

$$= !_e \circ U_e^{A[\vec{C}/\Theta]} \circ \|\Theta \vdash A\|^{\vec{C}} \qquad \text{(Theorem 7.21)}$$

$$= !_e \circ [\![\Theta \vdash A]\!] \circ (U_e^{C_1} \times \cdots \times U_e^{C_n}) \qquad \text{(IH)}$$

$$= [\![\Theta \vdash !A]\!] \circ (U_e^{C_1} \times \cdots \times U_e^{C_n}), \qquad \text{(Definition)}$$

which shows the required diagram commutes.

Case $\Theta \vdash A \star B$.

By induction $\|\Theta \vdash A\|^{\vec{C}}$ and $\|\Theta \vdash B\|^{\vec{C}}$ are ω-functors. ω-functors are closed under composition and pairing (§4.2) and $\star_e^{A[\vec{C}/\Theta], B[\vec{C}/\Theta]}$ is an ω-functor because of Theorem 5.8 and Theorem 7.21. Therefore $\|\Theta \vdash A \star B\|^{\vec{C}}$ is an ω-functor. Moreover:

$$U_e^{A \star B[\vec{C}/\Theta]} \circ \|\Theta \vdash A \star B\|^{\vec{C}} =$$

$$= U_e^{A \star B[\vec{C}/\Theta]} \circ \star_e^{A[\vec{C}/\Theta], B[\vec{C}/\Theta]} \circ \langle \|\Theta \vdash A\|^{\vec{C}}, \|\Theta \vdash B\|^{\vec{C}} \rangle \qquad \text{(Definition)}$$

$$= \star_e \circ (U_e^{A[\vec{C}/\Theta]} \times U_e^{B[\vec{C}/\Theta]}) \circ \langle \|\Theta \vdash A\|^{\vec{C}}, \|\Theta \vdash B\|^{\vec{C}} \rangle \qquad \text{(Theorem 7.21)}$$

$$= \star_e \circ \langle U_e^{A[\vec{C}/\Theta]} \circ \|\Theta \vdash A\|^{\vec{C}}, U_e^{B[\vec{C}/\Theta]} \circ \|\Theta \vdash B\|^{\vec{C}} \rangle$$

$$= \star_e \circ \langle [\![\Theta \vdash A]\!] \circ (U_e^{C_1} \times \cdots \times U_e^{C_n}), [\![\Theta \vdash B]\!] \circ (U_e^{C_1} \times \cdots \times U_e^{C_n}) \rangle \qquad \text{(IH)}$$

$$= \star_e \circ \langle [\![\Theta \vdash A]\!], [\![\Theta \vdash B]\!] \rangle \circ (U_e^{C_1} \times \cdots \times U_e^{C_n})$$

$$= [\![\Theta \vdash A \star B]\!] \circ (U_e^{C_1} \times \cdots \times U_e^{C_n}), \qquad \text{(Definition)}$$

which shows the required diagram commutes.

Case $\Theta \vdash \mu X.A$.

By induction $\|\Theta, X \vdash A\|^{\vec{C}, \mu X.A[\vec{C}/\Theta]}$ is an ω-functor. ω-functors are closed under composition and $\mathbb{I}_e^{\mu X.A[\vec{C}/\Theta]}$ is an ω-functor because of Theorem 5.8 and Theorem 7.21. Therefore $\|\Theta \vdash \mu X.A\|^{\vec{C}}$ is an ω-functor, because of Theorem 4.12.

Next, observe that for any closed type $\cdot \vdash \mu Y.B$, we have:

$$U^{\mu Y.B} \circ \mathbb{I}^{\mu Y.B} = U^{B[\mu Y.B/Y]} \qquad (7.2)$$

which follows immediately from the definition of $\mathbb{I}$. Then:

$$U_e^{\mu X.A[\vec{C}/\Theta]} \circ \mathbb{I}_e^{\mu X.A[\vec{C}/\Theta]} \circ \|\Theta, X \vdash A\|^{\vec{C}, \mu X.A[\vec{C}/\Theta]} =$$

$$= U_e^{A[\vec{C}/\Theta][\mu X.A[\vec{C}/\Theta]/X]} \circ \|\Theta, X \vdash A\|^{\vec{C}, \mu X.A[\vec{C}/\Theta]} \qquad (7.2)$$

$$= U_e^{A[\vec{C}/\Theta, \mu X.A[\vec{C}/\Theta]/X]} \circ \|\Theta, X \vdash A\|^{\vec{C}, \mu X.A[\vec{C}/\Theta]} \qquad \text{(Type identity)}$$

$$= [\![\Theta, X \vdash A]\!] \circ (U_e^{C_1} \times \cdots \times U_e^{C_n} \times U_e^{\mu X.A[\vec{C}/\Theta]}), \qquad \text{(IH)}$$

which shows that all of the conditions of Assumption 4.16 are satisfied. Therefore, by Corollary 4.21 :

$$U_e^{\mu X.A[\vec{C}/\Theta]} \circ \left(\mathbb{I}_e^{\mu X.A[\vec{C}/\Theta]} \circ \|\Theta, X \vdash A\|^{\vec{C}, \mu X.A[\vec{C}/\Theta]} \right)^{\dagger} = [\![\Theta, X \vdash A]\!]^{\dagger} \circ (U_e^{C_1} \times \cdots \times U_e^{C_n})$$

and then by definition:

$$U_e^{\mu X.A[\vec{C}/\Theta]} \circ \|\Theta \vdash \mu X.A\|^{\vec{C}} = [\![\Theta \vdash \mu X.A]\!] \circ (U_e^{C_1} \times \cdots \times U_e^{C_n}),$$

which shows the required diagram commutes. $\qquad \square$

By considering the proof of the above proposition for type $\Theta \vdash \mu X.A$, we get an important corollary which shows that for any recursive type, the parameterised initial algebra for the augmented interpretation is exactly the same as the one for the standard interpretation.

Corollary 7.27. *The following (2-categorical) diagram:*

$$\mathbb{I}_e^{\mu X.A[\vec{C}/\Theta]} \circ \|\Theta, X \vdash A\|^{\vec{C}, \mu X.A[\vec{C}/\Theta]} \circ \langle Id, \|\Theta \vdash \mu X.A\|^{\vec{C}} \rangle$$

$$\mathbf{R}(C_1)_e \times \cdots \times \mathbf{R}(C_n)_e \qquad \Downarrow \phi \qquad \mathbf{R}(A[\vec{C}/\Theta])_e$$

$$\|\Theta \vdash \mu X.A\|^{\vec{C}}$$

$$U_e^{C_1} \times \cdots \times U_e^{C_n} \qquad\qquad\qquad U_e^{A[\vec{C}/\Theta]}$$

$$[\![\Theta, X \vdash A]\!] \circ \langle Id, [\![\Theta \vdash \mu X.A]\!] \rangle$$

$$\mathbf{L}_e \times \cdots \times \mathbf{L}_e \qquad \Downarrow \phi \qquad \mathbf{L}_e$$

$$[\![\Theta \vdash \mu X.A]\!]$$

commutes (see Theorem 4.12 for ϕ).

Proof. In Proposition 7.26, we have shown

$$U_e^{\mu X.A[\vec{C}/\Theta]} \circ \left(\mathbb{I}_e^{\mu X.A[\vec{C}/\Theta]} \circ \|\Theta, X \vdash A\|^{\vec{C}, \mu X.A[\vec{C}/\Theta]} \right) =$$

$$= [\![\Theta, X \vdash A]\!] \circ (U_e^{C_1} \times \cdots \times U_e^{C_n} \times U_e^{\mu X.A[\vec{C}/\Theta]}),$$

which satisfies the conditions of Assumption 4.16 (with M and N given by the (products of) forgetful functors). Therefore, by Corollary 4.27 we complete the proof. $\qquad\square$

Next, we prove a substitution lemma for the augmented interpretations which shows that they behave as expected. Its proof depends on a permutation and a contraction lemma.

Lemma 7.28 (Permutation). *For any well-formed type $\Theta, X, Y, \Theta' \vdash A$ and closed types $C_1, \ldots, C_{m+m'+2}$:*

$$\|\Theta, Y, X, \Theta' \vdash A\|^{\vec{C}, C_{m+2}, C_{m+1}, \vec{C}'} = \|\Theta, X, Y, \Theta' \vdash A\|^{\vec{C}, C_{m+1}, C_{m+2}, \vec{C}'} \circ \mathrm{swap}_{m,m'}$$

where $|\Theta| = m$, $|\Theta'| = m'$, $\mathrm{swap}_{m,m'} = \langle \Pi_1, \ldots, \Pi_m, \Pi_{m+2}, \Pi_{m+1}, \Pi_{m+3}, \ldots, \Pi_{m+m'+2} \rangle$ and where $\vec{C} = C_1, \ldots, C_m$ and $\vec{C}' = C_{m+3}, \ldots, C_{m+m'+2}$.

Proof. Straightforward induction, essentially the same as Lemma 6.3 (1). $\qquad\square$

Lemma 7.29 (Contraction). *For any well-formed type $\Theta, \Theta' \vdash A$, such that $X \notin \Theta \cup \Theta'$, and closed types $C_1, \ldots, C_{m+m'+1}$:*

$$\|\Theta, X, \Theta' \vdash A\|^{\vec{C}, C_{m+1}, \vec{C}'} = \|\Theta, \Theta' \vdash A\|^{\vec{C}, \vec{C}'} \circ \mathrm{drop}_{m,m'}$$

where $|\Theta| = m$, $|\Theta'| = m'$, $\mathrm{drop}_{m,m'} = \langle \Pi_1, \ldots, \Pi_m, \Pi_{m+2}, \ldots, \Pi_{m+m'+1} \rangle$ and where we have $\vec{C} = C_1, \ldots, C_m$ and $\vec{C}' = C_{m+2}, \ldots, C_{m+m'+1}$.

Proof. Straightforward induction, essentially the same as Lemma 6.4 (1). $\qquad\square$

Lemma 7.30 (Substitution). *For any well-formed types $\Theta, X \vdash A$ and $\Theta \vdash B$ and closed types $C_1, \ldots, C_m$:*

$$\|\Theta \vdash A[B/X]\|^{\vec{C}} = \|\Theta, X \vdash A\|^{\vec{C}, B[\vec{C}/\Theta]} \circ \langle Id, \|\Theta \vdash B\|^{\vec{C}} \rangle.$$

Proof. Straightforward induction, essentially the same as Lemma 6.5 (1). $\qquad\square$

We may now finally prove the existence of the formal approximation relations as promised in Lemma 7.6. First, we introduce some additional notation for convenience when dealing with closed types and their augmented interpretations. Recall that the augmented interpretation of a closed type $\cdot \vdash A$ is a functor $\|\cdot \vdash A\|^{\cdot} : \mathbf{1} \to \mathbf{R}(A)$. Therefore let $\|A\| := \|\cdot \vdash A\|^{\cdot}(*)$, where $*$ is the unique object of $\mathbf{1}$. Then $\|A\|$ is an object of $\mathbf{R}(A)$. For the standard interpretation of types we use the notation from §6.4.

Proof of Lemma 7.6.
First, observe that by Proposition 7.26, $\|A\| = (\llbracket A \rrbracket, \trianglelefteq_A)$ for some

$$\trianglelefteq_A \subseteq (\mathbf{L}(I, \llbracket A \rrbracket) - \{\bot\}) \times \mathrm{Values}(A).$$

We will show that these relations $\trianglelefteq_A$ have the required properties. Next, recall that by Theorem 5.8, $T_e(A, B) = T(A, B)$, for any (bi)functor T and objects A, B. Then:

Property (B) holds by construction (cf. Definitions 7.12, 7.13 and 7.14).

Properties (A1.1), (A1.2), (A2) and (A3) hold, because:

$$\|A \star B\| = \|A\| \star^{A,B} \|B\| = (\llbracket A \star B \rrbracket, \trianglelefteq_A \star^{A,B} \trianglelefteq_B) \qquad (\star \in \{+, \otimes, \multimap\})$$

and then by Definition 7.18.

Property (A4) holds, because: $\|!A\| = !^A(\|A\|) = (\llbracket !A \rrbracket, ! \trianglelefteq_A)$ and then by Definition 7.18.

To prove property (A5), we reason as follows. By using Corollary 7.27 and Lemma 7.30 in the special case where $\Theta = \cdot$, we get an isomorphism (in the category $\mathbf{R}(\mu X.A)$)

$$\phi_*^{-1} : \|\mu X.A\| \cong \mathbb{I}^{\mu X.A} \left(\|X \vdash A\|^{\mu X.A} (\|\mu X.A\|) \right) = \mathbb{I}^{\mu X.A} \left(\|A[\mu X.A/X]\| \right) : \phi_*,$$

where $*$ is the unique object of the category $\mathbf{1}$. But by Corollary 7.27 and the notation from §6.4, it follows $\phi_*^{-1} = \mathrm{unfold}^{\mu X.A}$ and so we have an isomorphism in $\mathbf{R}(\mu X.A)$

$$\begin{aligned}
\mathrm{unfold}^{\mu X.A} : (\llbracket \mu X.A \rrbracket, \trianglelefteq_{\mu X.A}) &= \|\mu X.A\| \\
&\cong \mathbb{I}^{\mu X.A} \left(\|A[\mu X.A/X]\| \right) \qquad\qquad (7.3) \\
&= (\llbracket A[\mu X.A/X] \rrbracket, \mathbb{I}^{\mu X.A} \trianglelefteq_{A[\mu X.A/X]}) : \mathrm{fold}^{\mu X.A}
\end{aligned}$$

We may now easily complete the proof. For the left-to-right direction of (A5):

$$\begin{aligned}
&f \trianglelefteq_{\mu X.A} \texttt{fold}\ v \\
\Rightarrow\quad & \mathrm{unfold}^{\mu X.A} \circ f\ (\mathbb{I}^{\mu X.A} \trianglelefteq_{A[\mu X.A/X]})\ \texttt{fold}\ v \qquad ((7.3) \text{ and Lemma 7.15 (1)}) \\
\Rightarrow\quad & \mathrm{unfold}^{\mu X.A} \circ f\ \trianglelefteq_{A[\mu X.A/X]}\ v \qquad\qquad\qquad\qquad (\text{Lemma 7.23})
\end{aligned}$$

And for the right-to-left direction of (A5):

$$\begin{aligned}
&\mathrm{unfold}^{\mu X.A} \circ f\ \trianglelefteq_{A[\mu X.A/X]}\ v \\
\Rightarrow\quad & \mathrm{unfold}^{\mu X.A} \circ f\ (\mathbb{I}^{\mu X.A} \trianglelefteq_{A[\mu X.A/X]})\ \texttt{fold}\ v \qquad\qquad (\text{Lemma 7.23}) \\
\Rightarrow\quad & f = \mathrm{fold}^{\mu X.A} \circ \mathrm{unfold}^{\mu X.A} \circ f \trianglelefteq_{\mu X.A} \texttt{fold}\ v \qquad ((7.3) \text{ and Lemma 7.15 (1)}) \quad \square
\end{aligned}$$

8. Related Work

LNL-FPC can be seen as an extension with recursive types of the circuit-free fragment of Proto-Quipper-M [RS17], which corresponds to the CLNL calculus in [LMZ18]. The present paper adds a syntactic, operational and denotational treatment of recursive types to these type systems.

Our work is also closely related to the LNL calculus [Ben95] (also known as the adjoint calculus [BW96]). The LNL type assignment system has two kinds of term judgements – a linear one and a non-linear one, which are interpreted in a monoidal category and a cartesian category, respectively. However, LNL-FPC differs syntactically, because it has only one kind of term judgement, which we believe results in a more convenient syntax for programming.

Eppendahl [Epp03] extends LNL with recursive types only for the linear judgements. In that work, recursive types are treated linearly and are interpreted in the linear category only. In LNL-FPC, we also show how to add non-linear recursive types. We are able to treat them non-linearly, because we present new techniques for solving recursive domain equations within **CPO**-enriched cartesian categories that we use for their interpretation. Moreover, in LNL-FPC, both term-level recursion and type-level recursion are handled elegantly and agnostically with respect to the linearity of the type/term in question, which we think is a point in favor of our syntax. In contrast, term recursion in LNL and the adjoint calculus seems to be less elegant [BW96, pp. 11]. In an LNL-style syntax, it is also natural to add type recursion over the non-linear types, including non-linear function space, but it is an open problem how to interpret this in the cartesian category. Indeed, even if we assume $(\mathbf{C}, \times, [- \to -], 1)$ is cartesian closed, then the methods presented here do not explain how to compute fixpoints involving the $[- \to -]$ bifunctor, because it is unclear how to see it as a covariant functor on $\mathbf{C}$ while still being able to form initial algebras. In LNL-FPC, we do not assume our cartesian category is closed and so we do not expose any syntax for the internal-hom of the cartesian category and we have shown how to interpret the remaining non-linear recursive types (in both categories, see Figure 4).

Some mixed linear/non-linear type systems, such as the adjoint calculus [BW96], also support additive conjunction. We did not include additive conjunction in our type system, because we do not think it is interesting from a programming perspective. However, it appears it could be included without causing problems – on the denotational side we merely have to assume that our linear category has **CPO**-products, which is true in all of our concrete models.

The Lily language [BPR00] is a polymorphic linear/non-linear lambda calculus where recursive types may be encoded using polymorphism. The non-linear types in Lily are those of the form $!A$, whereas in LNL-FPC the non-linear types are also closed under sums, pairs and formation of recursive types, which is a major focus of our paper and that we believe improves the usability of the language. A version of Lily where the function space $A \multimap B$ is *strict*, rather than linear, was considered in [RS04] and the language was equipped with a denotational model based on synthetic domain theory. The authors prove two computational adequacy results (one for call-by-name and one for call-by-value evaluation) that hold at all types of the form $!A$. Computational adequacy for Lily at types $!A$ was established in [MBR08], again using a model based on synthetic domain theory. These are the only adequacy results, that we know of, for languages with a similar feature set to LNL-FPC. Our adequacy result, however, covers a larger range of types (all non-linear types, rather than

only those of the form $!A$) and our categorical model is considerably simpler and easier to understand. LNL-FPC does not support polymorphism, however.

LNL-FPC is also related to Barber's Dual Intuitionistic Linear Logic (DILL) [Bar96], which, like LNL-FPC, also has one kind of term judgement, but it enforces a strict separation between linear and non-linear contexts. As a result, for a lambda abstraction $\lambda x^A.p$ in DILL, the variable x is always treated as a linear variable, whereas LNL-FPC allows it to be treated as non-linear, provided A is, resulting in increased convenience for programming.

Other type systems like $\lambda_\to^q$ [BBN$^+$17], $^a\lambda_{ms}$ [TP11] and Quill [Mor16] support linear or affine types where the substructural operations are implicit, whereas in LNL-FPC only the weakening and contraction rules are implicit. However, none of them have so far been equipped with a categorical semantics (which is the main contribution of this paper).

Fiore and Plotkin model recursive types within FPC using categories that are algebraically compact in an enriched sense [Fio94, FP94]. Any category that is algebraically compact in such a sense and is also cartesian closed is necessarily degenerate. In a mixed linear/non-linear setting, we also have to solve recursive domain equations within the cartesian closed category, and because of this we cannot model recursive types using these techniques.

Another approach to modeling recursive types involves *bilimit compact categories* [Lev04] where type expressions are modelled as functors $T : \mathbf{A}^{op} \times \mathbf{B}^{op} \times \mathbf{A} \times \mathbf{B} \to \mathbf{B}$ for which one can solve recursive domain equations by using domain-theoretic methods [Pit96]. We do not know if these techniques would also work in a mixed linear/non-linear setting like ours, but we think that our presentation is simpler and more compact, because we simply interpret type expressions as functors $T : \mathbf{A}_e \times \mathbf{B}_e \to \mathbf{B}_e$. In addition, our categorical models can be easily simplified for languages that have only inductive datatypes (recursive types allowing only $\otimes$ and $+$). In such a case, **CPO**-enrichment from the model may be removed and our semantics can be adapted quite easily.

In [PPRZ20a, PPRZ20b], the authors present a denotational semantics for the first-order quantum programming language QPL extended with inductive datatypes (i.e., datatypes formed using $\otimes$ and $+$, but not $\multimap$ and $!$). This type system is not equipped with $\multimap$ and $!$ types and in order to model copying of classical information, the authors use methods based on the work reported here (and in [LMZ19]) that shows how to construct the required comonoids.

9. Conclusion and Future Work

We introduced a new language called LNL-FPC, which can be seen as extending FPC with a mixed linear/non-linear typing discipline (§2). An interesting feature of our type system is that it has implicit weakening and contraction rules – the users of the language do not have to explicitly specify when to copy and discard non-linear variables, while still enjoying linear type checking for the linear variables.

We presented a big-step call-by-value operational semantics for the language and showed that a previously known canonical fixpoint operator on terms [LMZ18] may be derived via the isorecursive type structure, thereby recreating an important result from FPC (§3).

In preparation for the denotational semantics of the language, we first had to establish some new coherence properties between (parameterised) initial algebras defined in potentially different categories and related by appropriate mediating functors (§4).

We then described our categorical model, which is a **CPO**-enriched model of Intuitionistic Linear Logic with suitable ω-colimits (§5). A crucial insight of our approach and of our

model, is that non-linear types that correspond to mixed-variance functors can be seen as *covariant* functors both on the linear category of embeddings, but also on the non-linear category of *pre-embeddings*. We also presented an entire class of concrete models that have found applications not only in classical functional programming, but also in programming languages for certain emerging fields (quantum computation and diagrammatic languages).

We proved our denotational semantics sound (§6) and cemented our results by demonstrating a computational adequacy result (§7). In doing so, we provided two kinds of denotational interpretations for non-linear types, terms and contexts – a standard interpretation within the linear category, and one within the cartesian category – that we showed are very strongly related to one another via a coherent natural isomorphism. This isomorphism, in turn, allowed us to elegantly interpret the substructural operations of Intuitionistic Linear Logic and in particular we precisely described the canonical comonoid structure of the non-linear (recursive) types. This, in turn, required developing new techniques for solving recursive domain equations within a cartesian category, which is perhaps the main contribution of this paper. More specifically, we showed that the solution of many (mixed-variance) recursive domain equations on the linear side (constructed over embeddings) can be reflected onto the cartesian side (by constructing them over pre-embeddings) and this result can be of interest even without our sound and adequate semantics.

As part of future work we would like to investigate what additional categorical structure (if any) is required in order to design and model a type system similar to ours, but where the substructural rules for promotion and dereliction are also implicit. In practical terms, this would result in a programming language that is more convenient for use, while still retaining the benefits of linear type checking. In theoretical terms, the categorical treatment could lead to a better understanding of the underlying design principles.

Another line of future work is to consider a more abstract model of LNL-FPC where the **CPO**-enrichment of our model is axiomatised away. This is a challenging problem, but if resolved, this could help to discover additional models of LNL-FPC.

Acknowledgments

References

[Abr93]   Samson Abramsky. Computational interpretations of linear logic. *Theor. Comput. Sci.*, 111(1&2):3–57, 1993.

[Adá74]   Jiří Adámek. Free algebras and automata realizations in the language of categories. *Commentationes Mathematicae Universitatis Carolinae*, 15(4):589–602, 1974.

[AF96]   M. Abadi and M. P. Fiore. Syntactic considerations on recursive types. In *Proceedings 11th Annual IEEE Symposium on Logic in Computer Science*, pages 242–252, July 1996.

[AJ94]   S. Abramsky and A. Jung. Handbook of logic in computer science (vol. 3). chapter Domain Theory, pages 1–168. Oxford University Press, Oxford, UK, 1994.

[AMM18]   Jiří Adámek, Stefan Milius, and Lawrence S Moss. Fixed points of functors. *Journal of Logical and Algebraic Methods in Programming*, 95:41–81, 2018.

[Bar96]   Andrew Barber. Dual intuitionistic linear logic, 1996. Technical Report, University of Edinburgh.

[BBN+17]   Jean-Philippe Bernardy, Mathieu Boespflug, Ryan R. Newton, Simon Peyton Jones, and Arnaud Spiwack. Linear haskell: Practical linearity in a higher-order polymorphic language. *Proc. ACM Program. Lang.*, 2(POPL):5:1–5:29, December 2017.

[Ben95]   P.N. Benton. A mixed linear and non-linear logic: Proofs, terms and models. In *Computer Science Logic: 8th Workshop, CSL '94, Selected Papaers*, 1995.

[Bor94]   F. Borceux. *Handbook of Categorical Algebra 2: Categories and Structures*. Cambridge University Press, 1994.

[BPR00]   Gavin M. Bierman, Andrew M. Pitts, and Claudio V. Russo. Operational properties of lily, a polymorphic linear lambda calculus with recursion. *Electr. Notes Theor. Comput. Sci.*, 41(3):70–88, 2000.

[Bra97]   T. Braüner. A general adequacy result for a linear funcitonal language. *Theoretical Computer Science*, 177:27–58, 1997.

[BW96]   P. N. Benton and P. Wadler. Linear logic, monads and the lambda calculus. In *LICS 1996*, 1996.

[CPT16]   L. Caires, F. Pfenning, and B. Toninho. Linear logic propositions as session types. *Mathematical Structures in Computer Science*, 26(3):367–423, 2016.

[Day74]   B. Day. On closed categories of functors ii. In *Category Seminar: Proceedings Sydney Category Theory Seminar 1972/1973*, pages 20–54, 1974.

[Epp03]   Adam Eppendahl. *Categories and types for axiomatic domain theory*. PhD thesis, Queen Mary University of London, UK, 2003.

[Fio94]   M. P. Fiore. *Axiomatic domain theory in categories of partial maps*. PhD thesis, University of Edinburgh, UK, 1994.

[FP94]   Marcelo Fiore and Gordon Plotkin. An axiomatization of computationally adequate domain theoretic models of FPC. In *LICS*, pages 92–102. IEEE Computer Society, 1994.

[Gir87]   J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1 – 101, 1987.

[Lev04]   Paul Blain Levy. *Call-By-Push-Value: A Functional/Imperative Synthesis*. Springer, 2004.

[LMZ18]   Bert Lindenhovius, Michael Mislove, and Vladimir Zamdzhiev. Enriching a linear/non-linear lambda calculus: A programming language for string diagrams. In Anuj Dawar and Erich Grädel, editors, *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, pages 659–668. ACM, 2018.

[LMZ19]   Bert Lindenhovius, Michael Mislove, and Vladimir Zamdzhiev. Mixed linear and non-linear recursive types. *Proc. ACM Program. Lang.*, 3(ICFP):111:1–111:29, August 2019.

[LMZ20]   Bert Lindenhovius, Michael Mislove, and Vladimir Zamdzhiev. Semantics for a lambda calculus for string diagrams, 2020. Preprint.

[LS81]   Daniel J Lehmann and Michael B Smyth. Algebraic specification of data types: A synthetic approach. *Mathematical Systems Theory*, 1981.

[MBR08]   Rasmus Ejlers Møgelberg, Lars Birkedal, and Giuseppe Rosolini. Synthetic domain theory and models of linear abadi & plotkin logic. *Ann. Pure Appl. Logic*, 155(2):115–133, 2008.

[ML98]   Saunders Mac Lane. *Categories for the Working Mathematician (2nd ed.)*. Springer, 1998.

[Mor16]   J. Garrett Morris. The best of both worlds: Linear functional programming without compromise. *SIGPLAN Not.*, 51(9):448–461, September 2016.

[MSS13]   Octavio Malherbe, Philip Scott, and Peter Selinger. *Presheaf Models of Quantum Computation: An Outline*, pages 178–194. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[Pit96]   Andrew M. Pitts. Relational properties of domains. *Inf. Comput.*, 127(2):66–90, 1996.

[Plo85]   G. D. Plotkin. Lectures on predomains and partial functions. notes for a course given at csli stanford university., 1985.

[PPRZ20a] Romain Péchoux, Simon Perdrix, Mathys Rennela, and Vladimir Zamdzhiev. Quantum programming with inductive datatypes, 2020. Preprint.

[PPRZ20b] Romain Péchoux, Simon Perdrix, Mathys Rennela, and Vladimir Zamdzhiev. Quantum programming with inductive datatypes: Causality and affine type theory. In *Foundations of Software Science and Computation Structures - 23rd International Conference, FOSSACS 2020*, volume 12077 of *Lecture Notes in Computer Science*, pages 562–581. Springer, 2020.

[PSV14]   Michele Pagani, Peter Selinger, and Benoît Valiron. Applying quantitative semantics to higher-order quantum computing. In Suresh Jagannathan and Peter Sewell, editors, *The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*, pages 647–658. ACM, 2014.

[RS04]   Giuseppe Rosolini and Alex Simpson. Using synthetic domain theory to prove operational properties of a polymorphic programming language based on strictness, 2004. Manuscript.

[RS17]   Francisco Rios and Peter Selinger. A categorical model for a quantum circuit description language. In *QPL 2017*, volume 266 of *EPTCS*, 2017.

[RS18]   Mathys Rennela and Sam Staton. Classical control and quantum circuits in enriched category theory. *Electr. Notes Theor. Comput. Sci.*, 336:257–279, 2018.

[SP82]   M.B. Smyth and G.D. Plotkin. The category-theoretic solution of recursive domain equations. *Siam J. Comput.*, 1982.

[TP11]   Jesse A. Tov and Riccardo Pucella. Practical affine types. In *POPL '11*, 2011.

[Wad90]   Philip Wadler. Linear types can change the world! In *PROGRAMMING CONCEPTS AND METHODS*. North, 1990.

Appendix A. Omitted Proofs

A.1. **Proof of Lemma 4.7.**

Proof. For brevity we write T_A, T_B, T_f for $T(A, -), T(B, -), T(f, -)$ respectively.

First, observe that $(T_f^{*n})_X = T(f, T(f, \cdots T(f, X) \cdots))$.

Therefore, $(S(T_f))_n = (T_f^{*n})_\varnothing = T(f, T(f, \cdots T(f, \varnothing) \cdots))$.

Next, observe that $T_A^n(g) = T(A, T(A, \cdots T(A, g) \cdots))$.

Next, we have

$$(T_f * S(T_f))_n = ((T_f * S(T_B)) \circ (T_A * S(T_f)))_n = T(f, T_B^n(\varnothing)) \circ T(A, (T_f^{*n})_\varnothing)$$
$$= T(f, (T_f^{*n})_\varnothing) = (T_f^{*n+1})_\varnothing.$$

And therefore

$$\begin{aligned}
((T_f * S(T_f)) \circ s^{T_A})_n &= (T_f^{*n+1})_\varnothing \circ T_A^n(\iota_{T_A\varnothing}) \\
&= T(f, T(f, \cdots T(f, T(f, \varnothing)) \cdots)) \circ T(A, T(A, \cdots T(A, \iota_{T_A\varnothing}) \cdots)) \\
&= T(f, T(f, \cdots T(f, T(f, \varnothing) \circ \iota_{T_A\varnothing}) \cdots)) \\
&= T(f, T(f, \cdots T(f, \iota_{T_B\varnothing}) \cdots)) \\
&= T_B^n(\iota_{T_B\varnothing}) \circ (T_f^{*n})_\varnothing \\
&= (s^{T_B} \circ S(T_f))_n \qquad \qquad \qquad \square
\end{aligned}$$

A.2. **Proof of Lemma 4.9.**

Proof. Composing $\tau_n : D_n \to D_n'$ with $d_n' : D_n' \to D_\omega'$ yields a morphism $d_n' \circ \tau_n : D_n \to D_\omega'$ for each $n \in \omega$, which forms a cocone of the diagram D. Since D_ω is the colimit of D, then $\mathrm{colim}(\tau) : D_\omega \to D_\omega'$ is by definition the unique map such that

$$\mathrm{colim}(\tau) \circ d_n = d_n' \circ \tau_n,$$

hence by functoriality of T, we obtain

$$T\mathrm{colim}(\tau) \circ Td_n = Td_n' \circ T\tau_n \qquad \qquad (A.1)$$

for each $n \in \omega$. Now, $T \circ D : \omega \to \mathbf{B}$ is an ω-diagram and since T preserves the colimit of D, it follows TD_ω is the colimit of $T \circ D$ with colimiting morphisms $Td_n : TD_n \to TD_\omega$ for each $n \in \omega$. Similarly, TD_ω' is the colimit of $T \circ D'$ with colimiting morphisms $Td_n' : TD_n' \to TD_\omega'$. Then $Td_n' \circ T\tau_n : TD_n \to TD_\omega'$ forms a cocone of the diagram $T \circ D$, and by definition $\mathrm{colim}(T\tau) : TD_\omega \to TD_\omega'$ is the unique morphism such that

$$\mathrm{colim}(T\tau) \circ Td_n = Td_n' \circ T\tau_n.$$

It follows now from (A.1) that $T\mathrm{colim}(\tau) = \mathrm{colim}(T\tau)$. $\qquad \square$

A.3. **Proof of Lemma 4.10.**

Proof. By Notation 4.8, $(D_\omega, d_n)_{n\in\omega}$ is the colimiting cocone of D. By definition of ω-functors, it follows that $(TD_\omega, Td_n)_{n\in\omega}$ and $(HD_\omega, Hd_n)_{n\in\omega}$ are the colimiting cocones of $T \circ D$ and $H \circ D$, respectively. Then $(HD_\omega, Hd_n \circ \tau_{D_n})_{n\in\omega}$ is a cocone of TD, and by definition $\mathrm{colim}(\tau D) : TD_\omega \to HD_\omega$ is the unique morphism such that

$$\mathrm{colim}(\tau D) \circ Td_n = Hd_n \circ \tau_{D_n}$$

for each $n \in \omega$. On the other hand, by naturality of $\tau : T \Rightarrow H$, we have

$$\tau_{D_\omega} \circ Td_n = Hd_n \circ \tau_{D_n},$$

for each $n \in \omega$, hence we must have $\mathrm{colim}(\tau D) = \tau_{D_\omega}$. $\qquad\square$

A.4. **Proof of Lemma 4.17.**

Proof. First, we have to show that for each $A \in \mathbf{A}$, we have that

$$\alpha_A^* : S(T(NA, -)) \Rightarrow M \circ S(H(A, -) : \omega \to \mathbf{D}$$

is natural in ω, i.e., we have to show that

$$MS(H(A, -))(n \le n+1) \circ (\alpha_A^*)_n = (\alpha_A^*)_{n+1} \circ S(T(NA, -)(n \le n+1)) \qquad (A.2)$$

for each $n \in \omega$. Let $\iota_1 : 0 \to T(NA, 0)$ and $\iota_2 : \varnothing \to H(A, \varnothing)$ be the indicated unique morphisms. By definition, equation (A.2) is then equivalent to:

$$MH(A, -)^n \iota_2 \circ (\alpha_A^*)_n = (\alpha_A^*)_{n+1} \circ T(NA, -)^n \iota_1. \qquad (A.3)$$

When $n = 0$, the equality becomes $M\iota_2 \circ z = (\alpha_A^*)_1 \circ \iota_1$, which is true by initiality. We proceed by induction, so assume that (A.3) holds for n. Expanding the equality for $n+1$, by definition of α_A^*, we get following diagram:

$$
\begin{array}{ccc}
T(NA, -)^{n+1}0 & \xrightarrow{\quad T(NA, -)^{n+1}\iota_1 \quad} & T(NA, -)^{n+2}0 \\[2mm]
\downarrow{\scriptstyle T(NA, (\alpha_A^*)_n)} & (1) & \downarrow{\scriptstyle T(NA, (\alpha_A^*)_{n+1})} \\[2mm]
T(NA, MH(A, -)^n\varnothing) & \xrightarrow{\;T(NA, MH(A, -)^n\iota_2)\;} & T(NA, MH(A, -)^{n+1}\varnothing) \\[2mm]
\downarrow{\scriptstyle \alpha_{A, H(A, -)^n\varnothing}} & (2) & \downarrow{\scriptstyle \alpha_{A, H(A, -)^{n+1}\varnothing}} \\[2mm]
MH(A, -)^{n+1}\varnothing & \xrightarrow[\quad MH(A, -)^{n+1}\iota_2 \quad]{} & MH(A, -)^{n+2}\varnothing
\end{array}
$$

Here the outer square is just equation (A.3) for $n+1$. Subdiagram (2) commutes by naturality of α, whereas (1) commutes because it is $T(NA, -)$ applied to equation (A.3), which is the induction step. We conclude that $(\alpha_A^*)_n$ is indeed natural in n.

Next, we have to show that α_A^* is also natural in A. Hence let $f : A_1 \to A_2$ be a morphism in $\mathbf{A}$. Then we need to show that

$$MS(H(f, -)) \circ \alpha_{A_1}^* = \alpha_{A_2}^* \circ S(T(Nf, -)).$$

This equality holds if and only it holds for all its components n, i.e., if

$$MH(f, -)^n \varnothing \circ (\alpha_{A_1}^*)_n = (\alpha_{A_2}^*)_n \circ T(Nf, -)^n 0 \tag{A.4}$$

holds for each $n \in \omega$. For $n = 0$, it reduces to $\text{id} \circ z = z \circ \text{id}$, which indeed holds. We proceed by induction. Assume that (A.4) holds for n. Consider the following diagram:

$$
\begin{array}{ccccc}
T(NA_1, -)^{n+1}0 & \xrightarrow{T(NA_1, (\alpha_{A_1}^*)_n)} & T(NA_1, MH(A_1, -)^n\varnothing) & \xrightarrow{\alpha_{A_1, H(A_1,-)^n\varnothing}} & MH(A_1, -)^{n+1}\varnothing \\
\Big\downarrow{\scriptstyle T(NA_1, T(Nf, -)^n 0)} & (1) & \Big\downarrow{\scriptstyle T(NA_1, MH(f, -)^n\varnothing)} & & \\
T(NA_1, T(NA_2, -)^n 0) & \xrightarrow{T(NA_1, (\alpha_{A_2}^*)_n)} & T(NA_1, MH(A_2, -)^n\varnothing) & (3) & \Big\downarrow{\scriptstyle MH(f, -)^{n+1}\varnothing} \\
\Big\downarrow{\scriptstyle T(Nf, T(NA_2, -)^n 0)} & (2) & \Big\downarrow{\scriptstyle T(Nf, MH(A_2, -)^n\varnothing)} & & \\
T(NA_2, -)^{n+1}0 & \xrightarrow[T(NA_2, (\alpha_{A_2}^*)_n)]{} & T(NA_2, MH(A_2, -)^n\varnothing) & \xrightarrow[\alpha_{A_2, H(A_2,-)^n\varnothing}]{} & MH(A_2, -)^{n+1}\varnothing
\end{array}
$$

Here the outer square is just (A.4) for $n + 1$. Subdiagram (1) commutes because it is $TN(A_1, -)$ acting on equation (A.4) for n, the induction step. Subdiagram (2) because T is a bifunctor. Subdiagram (3) commutes by naturality of α. We conclude that (A.4) also holds for $n + 1$ and therefore α^* is a natural transformation.

Finally, a simple induction argument shows that every component α_A^* is an isomorphism and therefore α^* is a natural isomorphism. $\qquad\square$

A.5. **Proof of Lemma 4.22.**

Proof. For (1), notice β can be seen as a natural transformation $\beta : H \circ (\text{Id} \times \text{Id}) \Rightarrow \text{Id} \circ H'$, hence β^* becomes a natural transformation $\beta^* : H^* \circ \text{Id} \Rightarrow (\text{Id} \triangleright -) \circ (H')^*$, defined by

$$(\beta_A^*)_0 := \left(\varnothing \xrightarrow{\text{id}_\varnothing} \varnothing \right)$$

$$(\beta_A^*)_{n+1} := \left(H(A, -)^{n+1}\varnothing \xrightarrow{H(A, (\beta_A^*)_n)} H(A, H'(A, -)^n\varnothing) \xrightarrow{\beta_{A, H'(A,-)^n\varnothing}} H'(A, -)^{n+1}\varnothing \right).$$

Furthermore, we have a natural transformation $M\beta \circ \alpha : T \circ (N \times M) \Rightarrow M \circ H'$, hence $(M\beta \circ \alpha)^* : T^* \circ N \Rightarrow (M \rhd -) \circ (H')^*$ is a natural transformation defined by

$$((M\beta \circ \alpha)_A^*)_0 := \left(0 \xrightarrow{z} M\varnothing\right)$$

$$((M\beta \circ \alpha)_A^*)_{n+1} :=$$
$$\left(T(NA,-)^{n+1}0 \xrightarrow{T(NA,(\delta_A^*)_n)} T(NA, MH'(A,-)^n\varnothing) \xrightarrow{\delta_{A,H'(A,-)^n\varnothing}} MH'(A,-)^{n+1}\varnothing\right).$$

where $\delta = M\beta \circ \alpha$. We have to show that we have $(M\beta \circ \alpha)_A^* = M\beta_A^* \circ \alpha_A^*$. Base case:

$$((M\beta \circ \alpha)_A^*)_0 = z = M(\mathrm{id}_\varnothing) \circ z = M(\beta_A^*)_0 \circ (\alpha_A^*)_0.$$

Assume as an induction hypothesis that $((M\beta \circ \alpha)_A^*)_n = (M\beta_A^* \circ \alpha_A^*)_n$, and consider the following diagram.

Here (1) commutes, because it is $T(NA,-)$ applied to the induction hypothesis, (2) commutes by definition of α_A^*, (3) commutes by naturality of α, and (4) commutes by definition of β_A^*. The outer right column of the diagram equals by definition $((M\beta \circ \alpha)_A^*)_{n+1}$. It follows that

$$((M\beta \circ \alpha)_A^*)_{n+1} = (M\beta_A^* \circ \alpha_A^*)_{n+1},$$

which concludes the proof of the first statement.

For the second statement, base case:

$$((\alpha(Q \times \mathrm{Id}) \circ T(\gamma \times M))_E^*)_0 = z \qquad\qquad\qquad (\text{Definition of } (-)^*)$$
$$= z \circ \mathrm{id}_0$$
$$= (\alpha_{QE}^*)_0 \circ (T^*\gamma_E)_0 \qquad (\text{Definition of } (-)^* \text{ and } T^*)$$

For the induction step, assume that $((\alpha(Q \times \mathrm{Id}) \circ T(\gamma \times M))_E^*)_n = (\alpha_{QE}^*)_n \circ (T^*\gamma_E)_n$. Using the induction hypothesis, then $((\alpha(Q \times \mathrm{Id}) \circ T(\gamma \times M))_E^*)_{n+1}$ is given by the composition of

the right and top sides of the diagram:

$$
\begin{array}{ccccc}
T(PE,-)^{n+1}0 & \xrightarrow{T(PE,(T^*\gamma_E)_n)} & T(PE,T(NQE,-)^n0) & \xrightarrow{T(PE,(\alpha^*_{QE})_n)} & T(PE,MH(QE,-)^n\varnothing) \\
\end{array}
$$

Triangle (1) commutes by definition of T^* and the Godement product:

$$(T^*\gamma_E)_{n+1} = T(\gamma_E,-)^{*(n+1)} = T(\gamma_E,-) * T(\gamma_E,-)^{*n} = T(\gamma_E,T(NQE,-)^n0) \circ T(PE,(T^*\gamma_E)_n).$$

Square (2) commutes by bifunctoriality of T and (3) commutes by definition of α^*. $\square$

A.6. Proof of Lemma 4.25.

Proof. The two compositions are natural transformations of ω-diagrams in $\mathbf{D}$, so we prove this by induction on $n \in \omega$. We first note that if $\mathbf{A}$ is taken to be trivial, then $T^* = S(T)$ and $H^* = S(H)$. Recall that $S(T)(n) = T^n0$, and $S(H)(n) = H^n\varnothing$. Let $\iota_{T0} : 0 \to T0$ and $\iota_{H\varnothing} : \varnothing \to H\varnothing$ be the unique initial maps. Then we have $(s^T)_n = T^n(\iota_{T0})$ and $(s^H)_n = H^n(\iota_{H\varnothing})$. Then $\alpha^* : T^* \Rightarrow MH^* : \omega \to \mathbf{D}$ is a natural transformation given by:

$$\alpha^*_0 := \left(0 \xrightarrow{z} M\varnothing\right)$$

$$\alpha^*_{n+1} := \left(T^{n+1}0 \xrightarrow{T\alpha^*_n} TMH^n\varnothing \xrightarrow{\alpha_{H^n\varnothing}} MH^{n+1}\varnothing\right).$$

In order to check that the diagram of natural transformations commutes, we first consider the 0-component as a basis step. Then we have to show that

$$(Ms^H \circ \alpha^*)_0 = (\alpha * S(H) \circ T\alpha^* \circ s^T)_0.$$

Since the object $S(T)(0)$ that is the source of both sides of the last equality is the initial object 0, it follows that the last equality indeed holds. Next, as an induction hypothesis, we assume that the n-component of the diagram commutes, hence

$$(Ms^H \circ \alpha^*)_n = (\alpha * S(H) \circ T\alpha^* \circ s^T)_n, \tag{A.5}$$

and we aim to show that the $(n+1)$-component commutes as well. Note that

$$(\alpha * S(H) \circ T\alpha^* \circ s^T)_n = (\alpha * S(H))_n \circ T\alpha^*_n \circ s^T_n = \alpha_{S(H)(n)} \circ T\alpha^*_n \circ s^T_n$$

$$= \alpha_{H^n\varnothing} \circ T\alpha^*_n \circ s^T_n = \alpha^*_{n+1} \circ s^T_n,$$

hence (A.5) becomes

$$Ms^H_n \circ \alpha^*_n = \alpha^*_{n+1} \circ s^T_n. \tag{A.6}$$

Consider now the following diagram:

$$
\begin{array}{ccccc}
& & \xrightarrow{\quad\alpha^*_{n+1}\quad} & & \\
& & (1) & & \\
T^{n+1}0 & \xrightarrow{\ T\alpha^*_n\ } & TMH^n\varnothing & \xrightarrow{\ \alpha_{H^n\varnothing}\ } & MH^{n+1}\varnothing \\
{\scriptstyle s^T_{n+1}=Ts^T_n}\Big\downarrow & (2) & {\scriptstyle TMs^H_n}\Big\downarrow & (3) & \Big\downarrow{\scriptstyle Ms^H_{n+1}=MHs^H_n} \\
T^{n+2}0 & \xrightarrow[\ T\alpha^*_{n+1}\ ]{} & TMH^{n+1}\varnothing & \xrightarrow[\ \alpha_{S(H)(n)}=\alpha_{H^{n+1}\varnothing}\ ]{} & MH^{n+2}\varnothing
\end{array}
$$

Here the outside of the diagram is exactly the identity we aim to show, namely the $(n+1)$-component. (1) commutes by definition of α^*_{n+1}, (2) commutes, since this is T applied to equation (A.6), and finally (3) commutes since α is natural. □

A.7. Proof of Theorem 5.4.

Proof. Using [SP82, Theorem 2], we get the following implications:

$$(4) \iff (3) \iff (1) \Rightarrow (2) \iff (5).$$

To finish the proof we will show $(2) \Rightarrow (3)$, which is stated in a slightly weaker form in the same paper (but its proof is the same).

Assume μ is a colimiting cocone of D in $\mathbf{C}_e$. Let $\nu : D \to B$ be a colimiting cocone of D in $\mathbf{C}$. Since $(1) \Rightarrow (3)$ and $(1) \Rightarrow (2)$ it follows that:

1. Each ν_i is an embedding, $(\nu_i \circ \nu_i^\bullet)_i$ is an increasing sequence and $\bigvee_i \nu_i \circ \nu_i^\bullet = \mathrm{id}_B$;
2. ν is a colimiting cocone of D in $\mathbf{C}_e$.

Therefore, μ and ν are isomorphic as cocones and therefore there exists an isomorphism $h : B \to A$, such that $\mu_i = h \circ \nu_i$. Therefore μ_i is an embedding (for every i) and for $i \leq j$, we have:

$$\mu_i \circ \mu_i^\bullet = h \circ \nu_i \circ \nu_i^\bullet \circ h^\bullet \leq h \circ \nu_j \circ \nu_j^\bullet \circ h^\bullet = \mu_j \circ \mu_j^\bullet$$

so that $(\mu_i \circ \mu_i^\bullet)_i$ forms an increasing sequence and:

$$\bigvee_i \mu_i \circ \mu_i^\bullet = \bigvee_i h \circ \nu_i \circ \nu_i^\bullet \circ h^\bullet = h \circ \left(\bigvee_i \nu_i \circ \nu_i^\bullet\right) \circ h^\bullet = h \circ \mathrm{id}_B \circ h^{-1} = \mathrm{id}_A. \qquad \square$$

A.8. Proof of Theorem 6.2.

Proof. By induction on the derivation of $\Theta \vdash P$. Let $|\Theta| = n$. The case $\Theta \vdash \Theta_i$ is given by:

$$\llbracket \Theta \vdash \Theta_i \rrbracket \circ F^{\times n}_{pe} = \Pi_i \circ F^{\times n}_{pe} = F_{pe} \circ \Pi_i = F_{pe} \circ (\!|\Theta \vdash \Theta_i|\!).$$

So that $\alpha^{\Theta \vdash \Theta_i} = \mathrm{id}$, as required. The case $\Theta \vdash !A$ is given by:

$$\llbracket \Theta \vdash !A \rrbracket \circ F^{\times n}_{pe} = !_e \circ \llbracket \Theta \vdash A \rrbracket \circ F^{\times n}_{pe} = F_{pe} \circ G_{pe} \circ \llbracket \Theta \vdash A \rrbracket \circ F^{\times n}_{pe} = F_{pe} \circ (\!|\Theta \vdash !A|\!)$$

So that $\alpha^{\Theta \vdash !A} = \mathrm{id}$, as required.

The case $\Theta \vdash P \odot Q$ is given by:

$$
\begin{aligned}
[\![\Theta \vdash P \odot Q]\!] \circ F_{pe}^{\times n} &= \odot_e \circ \langle [\![\Theta \vdash P]\!], [\![\Theta \vdash Q]\!] \rangle \circ F_{pe}^{\times n} && \text{(Definition)} \\
&= \odot_e \circ \langle [\![\Theta \vdash P]\!] \circ F_{pe}^{\times n}, [\![\Theta \vdash Q]\!] \circ F_{pe}^{\times n} \rangle \\
&\cong \odot_e \circ \langle F_{pe} \circ (\!|\Theta \vdash P|\!), F_{pe} \circ (\!|\Theta \vdash Q|\!) \rangle && \text{(IH)} \\
&= \odot_e \circ (F_{pe} \times F_{pe}) \circ \langle (\!|\Theta \vdash P|\!), (\!|\Theta \vdash Q|\!) \rangle \\
&\cong F_{pe} \circ \boxdot_{pe} \circ \langle (\!|\Theta \vdash P|\!), (\!|\Theta \vdash Q|\!) \rangle && \text{(Theorem 5.20)} \\
&= F_{pe} \circ (\!|\Theta \vdash P \odot Q|\!) && \text{(Definition)}
\end{aligned}
$$

Reading off the morphism, we get $\alpha^{\Theta \vdash P \odot Q} = \beta^{pe} \langle (\!|\Theta \vdash P|\!), (\!|\Theta \vdash Q|\!) \rangle \circ \odot_e \langle \alpha^{\Theta \vdash P}, \alpha^{\Theta \vdash Q} \rangle$, as required.

The case $\Theta \vdash \mu X.P$ is given by:

$$
\begin{aligned}
[\![\Theta \vdash \mu X.P]\!] \circ F_{pe}^{\times n} &= [\![\Theta, X \vdash P]\!]^{\dagger} \circ F_{pe}^{\times n} \\
&\cong F_{pe} \circ (\!|\Theta, X \vdash P|\!)^{\dagger} && \text{(Theorem 4.18)} \\
&= F_{pe} \circ (\!|\Theta \vdash \mu X.P|\!)
\end{aligned}
$$

Reading off the morphism, we get $\alpha^{\Theta \vdash \mu X.P} = (\alpha^{\Theta, X \vdash P})^{\dagger}$, as required. $\qquad\square$

A.9. Proof of Lemma 6.5.

Proof. **For (1):** Essentially the same as [Fio94, Lemma C.0.3].

For (2): By induction on the derivation of $\Theta, X \vdash P$. Let $|\Theta| = n$.

Case $\Theta, X \vdash \Theta_i$:

$$(\!|\Theta \vdash \Theta_i[R/X]|\!) = (\!|\Theta \vdash \Theta_i|\!) = \Pi_i = \Pi_i \circ \langle \Pi_1, \ldots, \Pi_n, (\!|\Theta \vdash R|\!) \rangle = (\!|\Theta, X \vdash \Theta_i|\!) \circ \langle \text{Id}, (\!|\Theta \vdash R|\!) \rangle.$$

so that $\gamma^{\Theta \vdash \Theta_i[R/X]} = \text{id}$, as required.

Case $\Theta, X \vdash X$:

$$(\!|\Theta \vdash X[R/X]|\!) = (\!|\Theta \vdash R|\!) = \Pi_{n+1} \circ \langle \text{Id}, (\!|\Theta \vdash R|\!) \rangle = (\!|\Theta, X \vdash X|\!) \circ \langle \text{Id}, (\!|\Theta \vdash R|\!) \rangle$$

so that $\gamma^{\Theta \vdash X[R/X]} = \text{id}$, as required.

Case $\Theta, X \vdash\, !A$:

$$
\begin{aligned}
(\!|\Theta \vdash\, !A[R/X]|\!) &= (\!|\Theta \vdash\, !(A[R/X])|\!) \\
&= G_{pe} \circ [\![\Theta \vdash A[R/X]]\!] \circ F_{pe}^{\times n} \\
&= G_{pe} \circ [\![\Theta, X \vdash A]\!] \circ \langle \text{Id}, [\![\Theta \vdash R]\!] \rangle \circ F_{pe}^{\times n} && \text{(Lemma 6.5 (1))} \\
&= G_{pe} \circ [\![\Theta, X \vdash A]\!] \circ \langle F_{pe}^{\times n}, [\![\Theta \vdash R]\!] \circ F_{pe}^{\times n} \rangle \\
&\cong G_{pe} \circ [\![\Theta, X \vdash A]\!] \circ \langle F_{pe}^{\times n}, F_{pe} \circ (\!|\Theta \vdash R|\!) \rangle && \text{(Theorem 6.2)} \\
&= G_{pe} \circ [\![\Theta, X \vdash A]\!] \circ F_{pe}^{\times(n+1)} \circ \langle \text{Id}, (\!|\Theta \vdash R|\!) \rangle \\
&= (\!|\Theta, X \vdash\, !A|\!) \circ \langle \text{Id}, (\!|\Theta \vdash R|\!) \rangle
\end{aligned}
$$

so that $\gamma^{\Theta \vdash\, !A[R/X]} = G_{pe}[\![\Theta, X \vdash A]\!] \langle F_{pe}^{\times n}, \alpha^{\Theta \vdash R} \rangle$.

Case $\Theta, X \vdash P \odot Q$:

$$
\begin{aligned}
(\!|\Theta \vdash (P \odot Q)[R/X]|\!) &= (\!|\Theta \vdash P[R/X] \odot Q[R/X]|\!) \\
&= \boxdot_{pe} \circ \langle (\!|\Theta \vdash P[R/X]|\!), (\!|\Theta \vdash Q[R/X]|\!) \rangle \\
&\cong \boxdot_{pe} \circ \langle (\!|\Theta, X \vdash P|\!) \circ \langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \rangle, (\!|\Theta, X \vdash Q|\!) \circ \langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \rangle \rangle \quad \text{(IH)} \\
&= \boxdot_{pe} \circ \langle (\!|\Theta, X \vdash P|\!), (\!|\Theta, X \vdash Q|\!) \rangle \circ \langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \rangle \\
&= (\!|\Theta, X \vdash P \odot Q|\!) \circ \langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \rangle
\end{aligned}
$$

so that $\gamma^{\Theta \vdash (P \odot Q)[R/X]} = \boxdot_{pe} \langle \gamma^{\Theta \vdash P[R/X]}, \gamma^{\Theta \vdash Q[R/X]} \rangle$.

Case $\Theta, X \vdash \mu Y.P$:

$$
\begin{aligned}
&(\!|\Theta \vdash \mu Y.P[R/X]|\!) \\
&= (\!|\Theta, Y \vdash P[R/X]|\!)^{\dagger} \\
&\cong \left((\!|\Theta, Y, X \vdash P|\!) \circ \langle \mathrm{Id}, (\!|\Theta, Y \vdash R|\!) \rangle \right)^{\dagger} &&\text{(IH and Remark 4.24)} \\
&= \left((\!|\Theta, X, Y \vdash P|\!) \circ swap_{n,0} \circ \langle \mathrm{Id}, (\!|\Theta, Y \vdash R|\!) \rangle \right)^{\dagger} &&\text{(Lemma 6.3 (2))} \\
&= \left((\!|\Theta, X, Y \vdash P|\!) \circ \langle \mathrm{Id}, (\!|\Theta, Y \vdash R|\!), \Pi_{n+1} \rangle \right)^{\dagger} \\
&= \left((\!|\Theta, X, Y \vdash P|\!) \circ \langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \circ drop_{n,0}, \Pi_{n+1} \rangle \right)^{\dagger} &&\text{(Lemma 6.4 (2))} \\
&= \left((\!|\Theta, X, Y \vdash P|\!) \circ (\langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \rangle \times \mathrm{Id}) \right)^{\dagger} \\
&= (\!|\Theta, X, Y \vdash P|\!)^{\dagger} \circ \langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \rangle &&\text{(Proposition 4.14)} \\
&= (\!|\Theta, X \vdash \mu Y.P|\!) \circ \langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \rangle
\end{aligned}
$$

so that $\gamma^{\Theta \vdash \mu Y.P[R/X]} = (\gamma^{\Theta, Y \vdash P[R/X]})^{\dagger}$.

For (3): Let $|\Theta| = n$ and let $\vec{Z} := (Z_1, \ldots, Z_n) \in Ob(\mathbf{C}_{pe}^{\times n})$ be an arbitrary object. We shall prove the equivalent statement

$$
F_{pe} \gamma^{\Theta \vdash P[R/X]} \circ \alpha^{\Theta \vdash P[R/X]} = \alpha^{\Theta, X \vdash P} \langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \rangle \circ [\![\Theta, X \vdash P]\!] \langle F_{pe}^{\times n}, \alpha^{\Theta \vdash R} \rangle.
$$

We do so by induction on the derivation of $\Theta, X \vdash P$.

Case $\Theta, X \vdash \Theta_i$:

$$
\begin{aligned}
(F_{pe} \gamma^{\Theta \vdash \Theta_i[R/X]} \circ \alpha^{\Theta \vdash \Theta_i[R/X]})_{\vec{Z}} &= \mathrm{id}_{F(Z_i)} \\
&= (\alpha^{\Theta, X \vdash \Theta_i} \langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \rangle \circ [\![\Theta, X \vdash \Theta_i]\!] \langle F_{pe}^{\times n}, \alpha^{\Theta \vdash R} \rangle)_{\vec{Z}}
\end{aligned}
$$

Case $\Theta, X \vdash X$:

$$
\begin{aligned}
(F_{pe} \gamma^{\Theta \vdash X[R/X]} \circ \alpha^{\Theta \vdash X[R/X]})_{\vec{Z}} &= \alpha_{\vec{Z}}^{\Theta \vdash R} \\
&= (\alpha^{\Theta, X \vdash X} \langle \mathrm{Id}, (\!|\Theta \vdash R|\!) \rangle \circ [\![\Theta, X \vdash X]\!] \langle F_{pe}^{\times n}, \alpha^{\Theta \vdash R} \rangle)_{\vec{Z}}
\end{aligned}
$$

Case $\Theta, X \vdash !A$:

$$
\begin{aligned}
F_{pe}\gamma^{\Theta\vdash !A[R/X]} \circ \alpha^{\Theta\vdash !A[R/X]} &= F_{pe}G_{pe}[\![\Theta, X \vdash A]\!]\langle F_{pe}^{\times n}, \alpha^{\Theta\vdash R}\rangle \circ \mathrm{id} \\
&= !_e[\![\Theta, X \vdash A]\!]\langle F_{pe}^{\times n}, \alpha^{\Theta\vdash R}\rangle \\
&= [\![\Theta, X \vdash !A]\!]\langle F_{pe}^{\times n}, \alpha^{\Theta\vdash R}\rangle \\
&= \mathrm{id} \circ [\![\Theta, X \vdash !A]\!]\langle F_{pe}^{\times n}, \alpha^{\Theta\vdash R}\rangle \\
&= \alpha^{\Theta, X\vdash !A}\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \circ [\![\Theta, X \vdash !A]\!]\langle F_{pe}^{\times n}, \alpha^{\Theta\vdash R}\rangle.
\end{aligned}
$$

Case $\Theta, X \vdash P \odot Q$:

Let $|\Theta| = n$ and let $\vec{Z} := (Z_1, \ldots, Z_n) \in Ob(\mathbf{C}_{pe}^{\times n})$ be an arbitrary object. Then the $\vec{Z}$ component of $\gamma^{\Theta\vdash P[R/X]}$ is a morphism

$$
\gamma_{\vec{Z}}^{\Theta\vdash P[R/X]} : (\!|\Theta \vdash P[R/X]|\!)\vec{Z} \to (\!|\Theta, X \vdash P|\!)\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle\vec{Z}
$$

hence we obtain from Theorem 5.20:

$$
\begin{aligned}
&F_{pe}\big(\gamma_{\vec{Z}}^{\Theta\vdash P[R/X]} \,\Box_{pe}\, \gamma_{\vec{Z}}^{\Theta\vdash Q[R/X]}\big) \circ \beta_{(\!|\Theta\vdash P[R/X]|\!)\vec{Z},(\!|\Theta\vdash Q[R/X]|\!)\vec{Z}}^{pe} \\
&= \beta_{(\!|\Theta,X\vdash P|\!)\langle \mathrm{Id},(\!|\Theta\vdash R|\!)\rangle\vec{Z},(\!|\Theta,X\vdash Q|\!)\langle \mathrm{Id},(\!|\Theta\vdash R|\!)\rangle\vec{Z}}^{pe} \circ \big(F_{pe}\gamma_{\vec{Z}}^{\Theta\vdash P[R/X]} \odot_e F_{pe}\gamma_{\vec{Z}}^{\Theta\vdash Q[R/X]}\big),
\end{aligned}
$$

hence we find

$$
\begin{aligned}
&F_{pe}\,\Box\,\langle\gamma^{\Theta\vdash P[R/X]}, \gamma^{\Theta\vdash Q[R/X]}\rangle \circ \beta^{pe}\langle(\!|\Theta \vdash P[R/X]|\!), (\!|\Theta \vdash Q[R/X]|\!)\rangle \\
&= \beta^{pe}\langle(\!|\Theta, X \vdash P|\!) \circ \langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle, (\!|\Theta, X \vdash Q|\!) \circ \langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle\rangle\circ \\
&\qquad \odot_e \langle F_{pe}\gamma^{\Theta\vdash P[R/X]}, F_{pe}\gamma^{\Theta\vdash Q[R/X]}\rangle \\
&= \beta^{pe}\langle(\!|\Theta, X \vdash P|\!), (\!|\Theta, X \vdash Q|\!)\rangle\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \circ \odot_e\langle F_{pe}\gamma^{\Theta\vdash P[R/X]}, F_{pe}\gamma^{\Theta\vdash Q[R/X]}\rangle, \qquad (A.7)
\end{aligned}
$$

Then, surpressing superscripts to improve readability:

$$
\begin{aligned}
&F_{pe}\gamma \circ \alpha \\
&= F_{pe}\,\Box_{pe}\,\langle\gamma, \gamma\rangle \circ \beta^{pe}\langle(\!|\Theta \vdash P[R/X]|\!), (\!|\Theta \vdash Q[R/X]|\!)\rangle \circ \odot_e\langle\alpha, \alpha\rangle \\
&= \beta^{pe}\langle(\!|\Theta, X \vdash P|\!), (\!|\Theta, X \vdash Q|\!)\rangle\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \circ \odot_e\langle F_{pe}\gamma, F_{pe}\gamma\rangle \circ \odot_e\langle\alpha, \alpha\rangle \\
&= \beta^{pe}\langle(\!|\Theta, X \vdash P|\!), (\!|\Theta, X \vdash Q|\!)\rangle\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \circ \odot_e\langle F_{pe}\gamma \circ \alpha, F_{pe}\gamma \circ \alpha\rangle \\
&= \beta^{pe}\langle(\!|\Theta, X \vdash P|\!), (\!|\Theta, X \vdash Q|\!)\rangle\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle\circ \\
&\qquad \odot_e\big\langle\alpha\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \circ [\![\Theta, X \vdash P]\!]\langle F_{pe}^{\times n}, \alpha\rangle, \alpha\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \circ [\![\Theta, X \vdash Q]\!]\langle F_{pe}^{\times n}, \alpha\rangle\big\rangle \\
&= \beta^{pe}\langle(\!|\Theta, X \vdash P|\!), (\!|\Theta, X \vdash Q|\!)\rangle\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle\circ \\
&\qquad \odot_e\big\langle\alpha\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle, \alpha\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle\big\rangle \circ \odot_e\big\langle[\![\Theta, X \vdash P]\!]\langle F_{pe}^{\times n}, \alpha\rangle, [\![\Theta, X \vdash Q]\!]\langle F_{pe}^{\times n}, \alpha\rangle\big\rangle \\
&= \beta^{pe}\langle(\!|\Theta, X \vdash P|\!), (\!|\Theta, X \vdash Q|\!)\rangle\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \circ \odot_e\langle\alpha, \alpha\rangle\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle\circ \\
&\qquad \odot_e\langle[\![\Theta, X \vdash P]\!], [\![\Theta, X \vdash Q]\!]\rangle\langle F_{pe}^{\times n}, \alpha\rangle \\
&= (\beta^{pe}\langle(\!|\Theta, X \vdash P|\!), (\!|\Theta, X \vdash Q|\!)\rangle \circ \odot_e\langle\alpha, \alpha\rangle)\,\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle\circ \\
&\qquad \odot_e\langle[\![\Theta, X \vdash P]\!], [\![\Theta, X \vdash Q]\!]\rangle\langle F_{pe}^{\times n}, \alpha\rangle \\
&= \alpha\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \circ [\![\Theta, X \vdash P \odot Q]\!]\langle F_{pe}^{\times n}, \alpha\rangle
\end{aligned}
$$

where the first and the last equalities are by definition, the second equality is (A.7), the third and the fifth equalities are by functoriality of $\odot$, the fourth equality is the induction

hypothesis, and the penultimate equality follows from $(\rho \circ \sigma)K = \rho K \circ \sigma K$ for any composable functor K and natural transformations ρ and σ.

Case $\Theta, X \vdash \mu Y.P$:

$$F_{pe}\gamma^{\Theta \vdash \mu Y.P[R/X]} \circ \alpha^{\Theta \vdash \mu Y.P[R/X]}$$

$$= F_{pe}(\gamma^{\Theta,Y \vdash P[R/X]})^\dagger \circ (\alpha^{\Theta,Y \vdash P[R/X]})^\dagger \qquad \text{(Definition)}$$

$$= (F_{pe}\gamma^{\Theta,Y \vdash P[R/X]} \circ \alpha^{\Theta,Y \vdash P[R/X]})^\dagger \qquad \text{(Lemma 4.23 (1))}$$

$$= (\alpha^{\Theta,Y,X \vdash P}\langle \mathrm{Id}, (\!|\Theta, Y \vdash R|\!)\rangle \circ [\![\Theta, Y, X \vdash P]\!]\langle F_{pe}^{\times n+1}, \alpha^{\Theta,Y \vdash R}\rangle)^\dagger \qquad \text{(IH)}$$

$$= (\alpha^{\Theta,X,Y \vdash P} swap_{n,0}\langle \mathrm{Id}, (\!|\Theta, Y \vdash R|\!)\rangle \circ$$
$$\qquad [\![\Theta, X, Y \vdash P]\!] swap_{n,0}\langle F_{pe}^{\times n+1}, \alpha^{\Theta,Y \vdash R}\rangle)^\dagger \qquad \text{(Lemma 6.3)}$$

$$= (\alpha^{\Theta,X,Y \vdash P}\langle \Pi_1, \ldots, \Pi_n, (\!|\Theta, Y \vdash R|\!), \Pi_{n+1}\rangle \circ$$
$$\qquad [\![\Theta, X, Y \vdash P]\!]\langle F_{pe}\Pi_1, \ldots, F_{pe}\Pi_n, \alpha^{\Theta,Y \vdash R}, F_{pe}\Pi_{n+1}\rangle)^\dagger$$

$$= (\alpha^{\Theta,X,Y \vdash P}(\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \times \mathrm{Id}) \circ [\![\Theta, X, Y \vdash P]\!](\langle F_{pe}^{\times n}, \alpha^{\Theta \vdash R}\rangle \times F_{pe}))^\dagger \qquad \text{(Lemma 6.4)}$$

$$= (\alpha^{\Theta,X,Y \vdash P})^\dagger\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \circ [\![\Theta, X, Y \vdash P]\!]^\dagger\langle F_{pe}^{\times n}, \alpha^{\Theta \vdash R}\rangle \qquad \text{(Lemma 4.23 (2))}$$

$$= \alpha^{\Theta,X \vdash \mu Y.P}\langle \mathrm{Id}, (\!|\Theta \vdash R|\!)\rangle \circ [\![\Theta, X \vdash \mu Y.P]\!]\langle F_{pe}^{\times n}, \alpha^{\Theta \vdash R}\rangle \qquad \text{(Definition)} \quad \square$$