A Model for a List-oriented Extension of the Lambda Calculus

Glenn Durfee

May 1997

CMU-CS-97-151

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Submitted in partial fulfillment of the requirements for the degree of
Master of Science (Honors Degree Program).
Department of Mathematics, Carnegie Mellon University.

Abstract

This work is intended to provide a semantics for a fragment of a programming language described by György Révész in [Rév88], for which no model was known. We begin with a brief presentation of the syntax of the lambda calculus and some relevant extensions. We then describe a class of complete lattices and use them as models for the lambda calculus. We then find specialized sublattices which we use as models for the extensions of the lambda calculus, thus achieving the original goal of finding a semantics for Révész's language.

Keywords: lambda calculus, applicative lists, denotational semantics, domain theory

Contents

1 A List-oriented Extension of the Lambda Calculus

In this section, three lambda calculi will be presented. The first is the standard untyped lambda calculus, and is presented to standardize notation and provide the reader with a review. The second is the lambda calculus with "applicative lists", an extension of the untyped lambda calculus and a substantial fragment of a programming language presented by Gryörgy Révész in [Rév88]. The third is another extension of the untyped lambda calculus — the lambda calculus with "explicit products", another invention of Révész, presented in [Rév95] — which will ease the transition from finding a model for the lambda calculus to finding a model for the Revesz's language in subsequent sections.

1.1 The Lambda Calculus

In this section we briefly review the definition of the lambda calculus and the axioms for convertability between terms, following the presentation in [Bar84].

Definition 1.1.1 *The set Λ of terms is defined inductively as follows:*

(i) *If x is a variable then $x \in \Lambda$;*

(ii) *If x is a variable and $P \in \Lambda$ then $(\lambda x.P) \in \Lambda$;*

(iii) *If $P, Q \in \Lambda$ then $P(Q) \in \Lambda$.*

To simplify notation, we omit the parentheses around lambda expressions when there is no ambiguity; e.g., $(\lambda x.P)$ is written $\lambda x.P$.

If a term $P \in \Lambda$ is an exact symbol-by-symbol copy of a term $Q \in \Lambda$, we write $P \equiv Q$. The relation $=$, intended to denote semantic equivalence, is axiomatized by the following rules: For all terms P, Q, R, S, and all variables x,

(R) $P = P$;

(S) $P = Q$ implies $Q = P$;

(T) $P = Q$ and $Q = R$ implies $P = R$;

(C) $P = Q$ and $R = S$ implies $P(R) = Q(S)$;

(ξ) $P = Q$ implies $\lambda x.P = \lambda x.Q$;

(α) $\lambda x.P = \lambda z.\{z/x\}P$, for any z which is neither free nor bound in P;[1]

(β) $(\lambda x.P)(Q) = \{Q/x\}P$.[2]

We will denote the resulting equational theory "$\lambda\beta$". This theory can be strengthened to a theory $\lambda\beta\eta$ by addition of the following rule:

(η) $\lambda x.P(x) = P$ when x is not free in P.

[1]Here the notation $\{z/x\}P$ is defined to mean "z replaces all free occurrences of x in P". We assume here that the reader is familiar with the notion of a "free occurrence" a variable and "replacement" of an expression for a variable; for an excellent discussion of the subject, see [Bar84, Appendix C]. Note that this presentation of the (α) rule is slightly different from the standard presentation, but in the presence of the (ξ) and (C) rules it is equivalent.

[2]Care must be taken when substituting Q for x in P so that free variables in Q do not become captured in P. This problem can be avoided by judicious use of the (α) rule on subterms of P. Henceforth we assume that free variables of Q will not become captured in $\{Q/x\}P$.

1.2 The Lambda Calculus with Applicative Lists

What follows is the syntax for a list-oriented extension of the Lambda Calculus, following [Rév88]. For the purposes of this paper, we will restrict it to the essential elements being studied, i.e., the lambda calculus along with nlist creation and manipulation primitives. We will denote this elementary programming language λAL.

Definition 1.2.1 *The set of terms Λ_{AL} of λAL is defined inductively as follows:*

(i) *The atoms* **head**, **tail**, **cons**, **nil**, **null**, **true**, **false**, *and* **cond** *are in* Λ_{AL}*;*

(ii) *If x is a variable then $x \in \Lambda_{\mathrm{AL}}$;*

(iii) *If x is a variable and $P \in \Lambda_{\mathrm{AL}}$ then $(\lambda x.P) \in \Lambda_{\mathrm{AL}}$;*

(iv) *If $P, Q \in \Lambda_{\mathrm{AL}}$ then $P(Q) \in \Lambda_{\mathrm{AL}}$.*

To simplify notation, we often use some syntactic sugar by writing composites of the list functions in standard list notation. That is, $\langle\,\rangle :=$ **nil** and for $n \geq 1$ and terms $E_1, \ldots, E_n$,

$$\langle E_1, \ldots, E_n \rangle := \mathbf{cons}(E_1)(\langle E_2, \ldots, E_n \rangle).$$

Convertability in λAL, denoted $=$, is axiomtized by the rules (R), (S), (T), (C), (ξ), (α), and (β) of the lambda calculus, plus:

(γ) (Applicative List Property) For all $n \geq 0$, variables x, and terms $M, E_1, \ldots, E_n \in \Lambda_{\mathrm{AL}}$,

(γ1) $\langle E_1, \ldots, E_n \rangle(M) = \langle E_1(M), \ldots, E_n(M) \rangle$,
(γ2) $\lambda x.\langle E_1, \ldots, E_n \rangle = \langle \lambda x.E_1, \ldots, \lambda x.E_n \rangle$;

(δ) For $n \geq 1$ and terms $M, N, E_1, \ldots, E_n$,

$$\begin{array}{ll}
\mathbf{head}(\langle\,\rangle) = \langle\,\rangle, & \mathbf{head}(\langle E_1, \ldots, E_n \rangle) = E_1, \\
\mathbf{tail}(\langle\,\rangle) = \langle\,\rangle, & \mathbf{tail}(\langle E_1, \ldots, E_n \rangle) = \langle E_2, \ldots, E_n \rangle, \\
\mathbf{null}(\langle\,\rangle) = \mathbf{true}, & \mathbf{null}(\langle E_1, \ldots, E_n \rangle) = \mathbf{false}, \\
\mathbf{cond}(\mathbf{true})(M)(N) = M, & \mathbf{cond}(\mathbf{false})(M)(N) = N.
\end{array}$$

These rules provide λAL with interesting functional properties, which are explored in [Rév88]. They are also reminiscient of Backus' FP construction in [Bac78].

It should be noted that if we assume the (η) rule as well, then (γ2) becomes superfluous. In fact, the models that we will be building for λAL in this work do satisfy the (η) rule, although it is not specifically required by the definition.

Although lists can be encoded in pure lambda calculus terms (for example, see [Rév88] and encodings for pairing in [Bar84]), it is not known whether there is an encoding that yields the applicative list property (the γ rule) while still satisfying (δ) rules. Given the unlikelihood of such an encoding, it is worthwhile to consider models in which the list manipulation primitives are represented directly.

This turns out to be a nontrivial problem, and it is easier to go by way of finding a model for a slightly simpler extension of the lambda calculus, described in the next section.

1.3 The Lambda Calculus with Explicit Products

It is quite common in programming language design to implement lists *via* pairing with a distinguished end-of-list element such as **nil**. In particular, one sees that given a pairing operator $(\cdot,\cdot)$ and element **nil**, one can encode the list $\langle E_1, \ldots, E_n\rangle$ as $(E_1, (E_2, (\cdots, (E_n, \mathbf{nil}))))$. Given projection functions for depairing, it is easy to imagine encodings for **head** and **tail**. Furthermore, if the pairing has the applicative property ($(F, G)(X) = (F(X), G(X))$ and $\mathbf{nil}(X) = \mathbf{nil}$) then the lists, which are encoded as cascading pairs, inherit the applicative property.

This is the motivation for studying the next system, the Lambda Calculus with Explicit Products. Although it is weaker than λAL, it retains the applicative property. We will find a model for this system, and with appropriate choices for the list atoms and an encoding for lists (which, as it turns out, is more complicated than the example above), we will be able to enrich this model to yield a model for λAL.

Definition 1.3.1 *The set of terms Λ_{EP} of λEP is defined inductively as follows:*

(i) *The atoms* **fst** *and* **snd** *are in Λ_{EP}.*

(ii) *If x is a variable then $x \in \Lambda_{\mathrm{EP}}$.*

(iii) *If x is a variable and $P \in \Lambda_{\mathrm{EP}}$ then $(\lambda x.P) \in \Lambda_{\mathrm{EP}}$.*

(iv) *If $P, Q \in \Lambda_{\mathrm{EP}}$ then $P(Q) \in \Lambda_{\mathrm{EP}}$.*

(v) *If $P, Q \in \Lambda_{\mathrm{EP}}$ then $(P, Q) \in \Lambda_{\mathrm{EP}}$.*

Convertability in λEP, denoted $=$, follows the rules (R), (S), (T), (C), (ξ), (α), and (β) of the lambda calculus, plus:

(γ) (Applicative Pair Property) For all $P, Q, R \in \Lambda_{\mathrm{EP}}$ and variables x,

(γ1) $(P, Q)(R) = (P(R), Q(R))$,

(γ2) $\lambda x.(P, Q) = (\lambda x.P, \lambda x.Q)$.

(δ) (Projections) For all $P, Q \in \Lambda_{\mathrm{EP}}$, $\mathbf{fst}((P, Q)) = P$, and $\mathbf{snd}((P, Q)) = Q$.

Notice that λEP is merely the untyped lambda calculus with pairing and the γ and δ rules. As with the comments in the last section, it is not known whether there is an encoding of pairing in the lambda calculus that satisfies these rules. Hence, the approach taken in this work is to build a model that represents pairing directly.

This pairing required above differs from the usual definition of pairing in that it need not be surjective. That is, a stronger version of pairing satisfies the following rule:

(SP) (Surjective Pairing) $P = (\mathbf{fst}(P), \mathbf{snd}(P))$ for all terms P.

The Lambda Calculus with Explicit Products does not require this condition, but it turns out that the models we shall build do indeed satisfy this property.

In the next section, a class of models for the untyped lambda calculus is introduced. Specific properties of these models are used to yield a model for λEP; these models are then further refined to achieve the original goal of finding a model for λAL.

2 Powerset Models of the Lambda Calculus

Over the years, a number of interesting models for the lambda calculus have been developed. A class of models with which it will be particularly convenient for us to work was developed and presented by Dana Scott in [Sco96], and is briefly presented again here.

2.1 Complete Lattices and Fixed Points

One method used to produce a model for the lambda calculus is to find an object D in a cartesian closed category such that $[D \to D]$ is a retract of D. More precisely, one finds mappings $\Phi\colon D \to [D \to D]$ and $\Psi\colon [D \to D] \to D$ such that $\Phi \circ \Psi = \mathrm{id}_{[D \to D]}$. However, by Cantor's Theorem, this is impossible when $[D \to D]$ is the full function space of D. This difficulty was overcome by Scott by restricting $[D \to D]$ to be the *continuous* maps from D to D, for a proper notion of continuity. It turns out that there are notions of continuity expressive enough to capture all λ-definable terms, and hence all computable functions.

We will work in the category of complete lattices. In this section we will quickly review a few basic definitions and theorems in order to set the stage for the model constructions that follow.

Definition 2.1.1 *A* **complete lattice** *is a partially ordered set* $(\mathbb{P}, \sqsubseteq)$ *such that every subset* $S \subseteq \mathbb{P}$ *has a least upper bound; that is, there exists a* $y \in \mathbb{P}$ *such that*

(i) $x \sqsubseteq y$ *for all* $x \in S$*, and*

(ii) $x \sqsubseteq z$ *for all* $x \in S$ *implies* $y \sqsubseteq z$.

We denote the unique least upper bound of S *by* $\bigsqcup S$*. The least upper bound of* $\{x, y\}$ *is denoted* $x \sqcup y$*. The greatest lower bound of* S *is* $\bigsqcup \big\{x \bigm| x \sqsubseteq s \text{ for all } s \in S\big\}$ *and is the unique* $y \in \mathbb{P}$ *such that*

(i) $y \sqsubseteq x$ *for all* $x \in S$*, and*

(ii) $z \sqsubseteq x$ *for all* $x \in S$ *implies* $z \sqsubseteq y$.

The greatest lower bound of $\{x, y\}$ *is denoted* $x \sqcap y$.

Definition 2.1.2 *Let* $(\mathbb{P}_1, \sqsubseteq_1)$ *and* $(\mathbb{P}_2, \sqsubseteq_2)$ *be complete lattices. A map* $f\colon \mathbb{P}_1 \to \mathbb{P}_2$ *is said to* **monotone** *if* $x \sqsubseteq_1 y \Rightarrow f(x) \sqsubseteq_2 f(y)$ *for all* $x, y \in \mathbb{P}_1$.

Theorem 2.1.3 (Tarski's Fixed Point Theorem) *Let* $(\mathbb{P}, \sqsubseteq)$ *be a complete lattice and let the map* $f\colon \mathbb{P} \to \mathbb{P}$ *be monotone. Denote by* $\mathcal{FP}(f)$ *the set of fixed points of* f*, i.e.,*

$$\mathcal{FP}(f) := \big\{x \in \mathbb{P} \bigm| x = f(x)\big\}$$

Then $\mathcal{FP}(f)$ *is nonempty and forms a complete lattice with respect to* $\sqsubseteq$.

Definition 2.1.4 *A subset* $S \subseteq \mathbb{P}$ *of a complete lattice is said to be* **directed** *if, for each* $x, y \in S$*, there is a* $z \in S$ *such that* $x \sqsubseteq z$ *and* $y \sqsubseteq z$.

Definition 2.1.5 *Let* $(\mathbb{P}_1, \sqsubseteq_1)$ *and* $(\mathbb{P}_2, \sqsubseteq_2)$ *be complete lattices. A map* $f\colon \mathbb{P}_1 \to \mathbb{P}_2$ *is said to* **continuous** *if, for every directed set* $S \subseteq \mathbb{P}_1$,

$$f(\bigsqcup S) = \bigsqcup \big\{f(s) \bigm| s \in S\big\}.$$

Theorem 2.1.6 *Every continuous function is monotone.*

Proof: Let $x \sqsubseteq y$. Then $\{x, y\}$ is directed so $f(y) = f(x \sqcup y) = f(x) \sqcup f(y)$, i.e., $f(x) \sqsubseteq f(y)$. ∎

Theorem 2.1.7 *If $f \colon \mathbb{P}_1 \to \mathbb{P}_2$ and $g \colon \mathbb{P}_2 \to \mathbb{P}_3$ are continuous then $g \circ f \colon \mathbb{P}_1 \to \mathbb{P}_3$ is continuous.*

Theorem 2.1.8 *If $f, g \colon \mathbb{P}_1 \to \mathbb{P}_2$ are continuous then $x \mapsto f(x) \sqcup g(x)$ and $x \mapsto f(x) \sqcap g(x)$ are continuous.*

Let $\mathbb{A}$ be any set. We denote by $\mathfrak{P}\mathbb{A}$ the power set of $\mathbb{A}$. It is a well-known result that $(\mathfrak{P}\mathbb{A}, \subseteq)$ forms a complete lattice, and that $\bigsqcup S = \bigcup S$ for all $S \subseteq \mathfrak{P}\mathbb{A}$. The lattice structure on $(\mathfrak{P}\mathbb{A})^n$ is determined by componentwise inclusion, and a map $f \colon (\mathfrak{P}\mathbb{A})^n \to \mathfrak{P}\mathbb{A}$ is said to be continuous when f is continuous in each argument. The lattice on $[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$ is determined pointwise; i.e., $f \sqsubseteq g$ if and only if $f(X) \subseteq g(X)$ for every $X \in \mathfrak{P}\mathbb{A}$.

We denote by $\mathfrak{F}X$ the collection of all finite subsets of X; that is,

$$\mathfrak{F}X := \{Y \in \mathfrak{P}X \mid Y \text{ is finite}\}.$$

Theorem 2.1.9 *A function $f \colon \mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}$ is continuous if and only if the following condition holds: for all $X \in \mathfrak{P}\mathbb{A}$,*

$$f(X) = \bigcup_{X_0 \in \mathfrak{F}X} f(X_0).$$

Proof:

($\Rightarrow$) Clearly $\mathfrak{F}X$ is directed. So if f is continuous then $f(X) = f(\bigcup \mathfrak{F}X) = \bigcup_{X_0 \in \mathfrak{F}X} f(X_0)$.

($\Leftarrow$) Suppose the condition holds. Let a directed $\mathcal{X} \subseteq \mathfrak{P}\mathbb{A}$ be given. Observe that for every $X \in \mathcal{X}$, $X \subseteq \bigcup \mathcal{X}$ so $\mathfrak{F}X \subseteq \mathfrak{F}(\bigcup \mathcal{X})$. Thus

$$f(X) = \bigcup_{X_0 \in \mathfrak{F}X} f(X_0) \subseteq \bigcup_{X_0 \in \mathfrak{F}(\bigcup \mathcal{X})} f(X_0) = f(\bigcup \mathcal{X}),$$

implying $\bigcup_{X \in \mathcal{X}} f(X) \subseteq f(\bigcup \mathcal{X})$. Now let $y \in f(\bigcup \mathcal{X}) = \bigcup_{X_0 \in \mathfrak{F}(\bigcup \mathcal{X})} f(X_0)$ be given. Choose $x_1, \ldots, x_n \in \bigcup \mathcal{X}$ such that $y \in f(\{x_1, \ldots, x_n\})$. Then choose $X_1, \ldots, X_n \in \mathcal{X}$ such that $x_i \in X_i$. Since $\mathcal{X}$ is directed we may choose $X_* \in \mathcal{X}$ such that $X_1, \ldots, X_n \subseteq X_*$. So

$$y \in f(\{x_1, \ldots, x_n\}) \subseteq \bigcup_{X_0 \in \mathfrak{F}X_*} f(X_0) = f(X_*) \subseteq \bigcup_{X \in \mathcal{X}} f(X).$$

So f is continuous.

∎

Notice that for monotone functions on a powerset, showing continuity reduces to showing that for each $y \in f(X)$, there are $x_1, \ldots, x_n \in X$ such that $y \in f(\{x_1, \ldots, x_n\})$. Intuitively, this means that each piece of information in the output of the computation of f on argument X is the result of running f on some finite portion of the input. (See [SHLG94] for a discussion and a proof of the Representation Theorem, which proves the equivalence of this and several other notions of computation.)

2.2 Sequential Algebras

In this section we make a few observations that will be useful when embedding the continuous function space $[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$ into $\mathfrak{P}\mathbb{A}$.

Definition 2.2.1 *Let $\mathbb{A}$ be a set. Then $\mathbb{A}^n := \overbrace{\mathbb{A} \times \cdots \times \mathbb{A}}^{n}$ is the n-fold Cartesian product of $\mathbb{A}$ with itself. By convention, $\mathbb{A}^0$ is the singleton set $\{0\}$, denoted $\mathbf{1}$. We define $\mathbb{A}^*$ as the disjoint union of all n-fold Cartesian products of $\mathbb{A}$; i.e.,*

$$\mathbb{A}^* := \mathbf{1} + \mathbb{A} + \mathbb{A}^2 + \mathbb{A}^3 + \cdots + \mathbb{A}^n + \cdots$$

Theorem 2.2.2 *Let $\mathbb{A}$ be an infinite set. Then there exists a bijection*

$$\varphi\colon\ \mathbb{A}^* \cong \mathbb{A}.$$

Proof: Immediate by cardinal arithmetic. ∎

For a given set $\mathbb{A}$ and φ as above, we denote $\varphi(x_1, \ldots, x_n)$ by $\langle x_1, \ldots, x_n \rangle$. Since φ is a bijection, we see that to every element a of $\mathbb{A}$ there is an associated $n \geq 0$ and $a_1, \ldots, a_n$ such that $a = \langle a_1, \ldots, a_n \rangle$, while to each $k \geq 0$ and $b_1, \ldots, b_k$ in $\mathbb{A}$ there is a b in $\mathbb{A}$ such that $b = \langle b_1, \ldots, b_k \rangle$. Hence, $\mathbb{A}$ is closed under formation of finite sequences (when the $\langle \cdots \rangle$ operator is applied to the sequence), while each member of $\mathbb{A}$ can itself be viewed as a finite sequence.

It turns out that this makes the following embedding of continuous functions convenient.

2.3 Representing Continuous Functions

Recall that we are looking for mappings $\Phi\colon\ \mathfrak{P}\mathbb{A} \to [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$, $\Psi\colon\ [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}] \to \mathfrak{P}\mathbb{A}$ such that $\Phi \circ \Psi = \mathrm{id}_{[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]}$. We begin with the definition for Ψ. The basic idea is that Ψ produces a "graph" of an argument $f\colon\ \mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}$, which, since f is continuous, is small enough to be contained in $\mathfrak{P}\mathbb{A}$.

Definition 2.3.1 *The map $\Psi\colon\ [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}] \to \mathfrak{P}\mathbb{A}$ is defined by*

$$\Psi(f) := \{0\} \cup \big\{ \langle x_1, \ldots, x_n, y \rangle \bigm| x_1, \ldots, x_n \in \mathbb{A} \text{ and } y \in f(\{x_1, \ldots, x_n\}) \big\}$$

for all $f \in [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$.

The intuition behind this definition is that Ψ records the behavior of f on arguments of *finite* cardinality, which, by the continuity of f, completely captures the behavior of f. The presence of $\{0\}$ in the definition may seem arbitrary; this is a technical consideration which will be justified later.

Definition 2.3.2 *The map $\Phi\colon\ \mathfrak{P}\mathbb{A} \to [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$ is defined by*

$$\Phi(F) := X \mapsto \big\{ y \in \mathbb{A} \bigm| \exists n \geq 0, x_1, \ldots, x_n \in X \text{ such that } \langle x_1, \ldots, x_n, y \rangle \in F \big\}$$

It is not obvious that this mapping is well-defined in the sense of giving a continuous function in $[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$. The following theorem proves that is well-defined and gives an additional result that will be needed later.

Theorem 2.3.3 *The map* App: $(\mathfrak{P}\mathbb{A})^2 \to \mathfrak{P}\mathbb{A}$ *defined by*

$$\mathrm{App}(F, X) = \Phi(F)(X)$$

is continuous in each argument. In particular, this implies that for any $F \in \mathfrak{P}\mathbb{A}$, $\Phi(F)$ *is continuous (hence* Φ *is well-defined.)*

Proof: It is immediate from the definition that $\mathrm{App}(F, X)$ is monotone in each argument. Fix $F, X \in \mathfrak{P}\mathbb{A}$ and let $y \in \Phi(F)(X)$ be given. We may choose $x_1, \ldots, x_n \in X$ such that $\langle x_1, \ldots, x_n, y\rangle \in F$. Define F_0 to be the singleton $\{\langle x_1, \ldots, x_n, y\rangle\}$ and X_0 to be the set $\{x_1, \ldots, x_n\}$. Then

$$y \in \Phi(\{\langle x_1, \ldots, x_n, y\rangle\})(\{x_1, \ldots, x_n\}) = \mathrm{App}(F_0, X_0)$$

which is in turn a subset of both $\mathrm{App}(F_0, X)$ and $\mathrm{App}(F, X_0)$ by monotonicity. Hence App is continuous in both arguments and Φ is well-defined. ∎

In order to use these mappings to give a continuous model of the lambda calculus we must confirm that they are continuous lattice operations. We have just shown that Φ is continuous.

Theorem 2.3.4 *The map* $\Psi\colon [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}] \to \mathfrak{P}\mathbb{A}$ *is continuous.*

Proof: Let $\{f_i \mid i \in I\}$ be a directed set in $[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$. We aim to show that $\Psi(\bigsqcup_{i\in I} f_i) = \bigcup_{i\in I} \Psi(f_i)$.

Let $z \in \bigcup_{i\in I} \Psi(f_i)$ be given. Choose $i \in I$ and $x_1, \ldots, x_n, y \in \mathfrak{P}\mathbb{A}$ such that $z = \langle x_1, \ldots, x_n, y\rangle \in \Psi(f_i)$. Then

$$y \in f_i(\{x_1, \ldots, x_n\}) \subseteq \left(\bigsqcup_{i\in I} f_i\right)(\{x_1, \ldots, x_n\}),$$

implying $z = \langle x_1, \ldots, x_n, y\rangle \in \Psi(\bigsqcup_{i\in I} f_i)$. Since z was arbitrary, $\bigcup_{i\in I} \Psi(f_i) \subseteq \Psi(\bigsqcup_{i\in I} f_i)$.

Now let $z \in \Psi(\bigsqcup_{i\in I} f_i)$ be given and choose $x_1, \ldots, x_n, y \in \mathfrak{P}\mathbb{A}$ such that $z = \langle x_1, \ldots, x_n, y\rangle \in \Psi(\bigsqcup f_i)$. Then $y \in (\bigsqcup_{i\in I} f_i)(\{x_1, \ldots, x_n\}) = \bigcup_{i\in I} f_i(\{x_1, \ldots, x_n\})$. So $y \in f_i(\{x_1, \ldots, x_n\})$ for some particular $i \in I$, implying $z = \langle x_1, \ldots, x_n, y\rangle \in \Psi(f_i) \subseteq \bigcup_{i\in I} \Psi(f_i)$. Since z was arbitrary, $\Psi(\bigsqcup_{i\in I} f_i) \subseteq \bigcup_{i\in I} \Psi(f_i)$. Hence

$$\bigcup_{i\in I} \Psi(f_i) = \Psi(\bigsqcup_{i\in I} f_i),$$

i.e., Ψ is continuous. ∎

Lastly, we wish to show that this is a retract.

Theorem 2.3.5

$$\Phi \circ \Psi = \mathrm{id}_{[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]}.$$

Proof: Let $f \in [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$ and $X \in \mathfrak{P}\mathbb{A}$ be given. Let $y \in f(X)$ be given. Since f is continuous, we may choose $x_1, \ldots, x_n \in X$ such that $y \in f(\{x_1, \ldots, x_n\})$. Then $\langle x_1, \ldots, x_n, y\rangle \in \Psi(f)$. But by definition of Φ, we then have $y \in \Phi(\Psi(f))(X)$. Since y was arbitrary, $f(X) \subseteq \Phi(\Psi(f))(X)$.

Now let $z \in \Phi(\Psi(f))(X)$ be given. Choose $x_1, \ldots, x_n \in X$ such that $\langle x_1, \ldots, x_n, y\rangle \in \Psi(f)$. But by definition of Ψ, we have $z \in f(\{x_1, \ldots, x_n\})$. Since $\{x_1, \ldots, x_n\} \subseteq X$ is finite, by continuity of f, $z \in f(X)$. Since z was arbitrary, $\Phi(\Psi(f))(X) \subseteq f(X)$.

So $\Phi(\Psi(f))(X) = f(X)$. Since X was arbitrary, $\Phi(\Psi(f)) = f$. Thus $\Phi \circ \Psi = \mathrm{id}_{[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]}$. ∎

For the rest of this work, $\mathbb{A}$ is assumed to be any infinite set with the a sequential algebra structure determined by $\langle \cdots \rangle$, and mappings Ψ and Φ are as above.

2.4 Modeling the Lambda Calculus

At this point, we have all that is required for a model for the lambda calculus.

Definition 2.4.1 *Let ρ be a map from variables to $\mathfrak{P}\mathbb{A}$. We define interpretation map $[\![\cdot]\!]_\rho \colon \Lambda \to \mathfrak{P}\mathbb{A}$ inductively as follows:*

(i) $[\![x]\!]_\rho = \rho(x)$,

(ii) $[\![P(Q)]\!]_\rho = \Phi([\![P]\!]_\rho)([\![Q]\!]_\rho)$,

(iii) $[\![(\lambda x.P)]\!]_\rho = \Psi(A \mapsto [\![P]\!]_{\rho(x:=A)})$.

It must be shown that rule (iii) is well-defined.

Theorem 2.4.2 *For all $P \in \Lambda$, the function $A \mapsto [\![P]\!]_{\rho(x:=A)}$ is continuous.*

Proof: We proceed by induction on P. If $P \equiv x$, then $A \mapsto [\![P]\!]_{\rho(x:=A)} = A \mapsto A = \mathrm{id}_{\mathfrak{P}\mathbb{A}}$, which is clearly continuous. If $P \equiv y$ for some $y \neq x$, then $A \mapsto [\![P]\!]_{\rho(x:=A)} = A \mapsto \rho(y)$, a constant map, which is also continuous.

Now suppose $P \equiv M(N)$ for some terms M, N for which the inductive hypothesis holds. Let $A \subseteq B \in \mathfrak{P}\mathbb{A}$ be given. By the inductive hypothesis, $[\![M]\!]_{\rho(x:=A)} \subseteq [\![M]\!]_{\rho(x:=B)}$ (similarly for N). So by the monotonicity of App in each argument, $[\![P]\!]_{\rho(x:=A)} \subseteq [\![P]\!]_{\rho(x:=B)}$. Now let $c \in [\![P]\!]_{\rho(x:=A)} = \mathrm{App}([\![M]\!]_{\rho(x:=A)}, [\![N]\!]_{\rho(x:=A)})$ be given. Since App is continuous in each argument we may choose a finite $F_0 \subseteq [\![M]\!]_{\rho(x:=A)}$ and a finite $X_0 \subseteq [\![N]\!]_{\rho(x:=A)}$ such that $c \in \mathrm{App}(F_0, X_0)$. Then by the inductive hypothesis we may find a finite A_0 such that $F_0 \subseteq [\![M]\!]_{\rho(x:=A_0)}$ and $X_0 \subseteq [\![M]\!]_{\rho(x:=A_0)}$. By the monotonicity of App this implies $c \in \mathrm{App}([\![M]\!]_{\rho(x:=A_0)}, [\![N]\!]_{\rho(x:=A_0)}) = [\![P]\!]_{\rho(x:=A_0)}$. Hence $A \mapsto [\![P]\!]_{\rho(x:=A)}$ is continuous.

Now suppose $P \equiv \lambda x.M$ for some term M. Then

$$[\![P]\!]_{\rho(x:=A)} = \Psi(B \mapsto [\![M]\!]_{\rho(x:=A)(x:=B)}) = \Psi(B \mapsto [\![M]\!]_{\rho(x:=B)})$$

which does not depend on A (so $A \mapsto [\![P]\!]_{\rho(x:=A)}$ is a constant, hence continuous, function).

Lastly, suppose $P \equiv \lambda y.M$ for some term M and some variable $y \not\equiv x$. Let $A \subseteq B \in \mathfrak{P}\mathbb{A}$ be given. By the inductive hypothesis, $[\![M]\!]_{\rho(x:=A)(y:=C)} \subseteq [\![M]\!]_{\rho(x:=B)(y:=C)}$ for all $C \in \mathfrak{P}\mathbb{A}$. This implies

$$\begin{aligned}
[\![P]\!]_{\rho(x:=A)} &= \Psi(C \mapsto [\![M]\!]_{\rho(x:=A)(y:=C)}) \\
&= \{0\} \cup \big\{\langle c_1, \dots, c_n, z\rangle \;\big|\; c_1, \dots, c_n \in \mathbb{A} \text{ and } z \in [\![M]\!]_{\rho(x:=A)(y:=\{c_1,\dots,c_n\})}\big\} \\
&\subseteq \{0\} \cup \big\{\langle c_1, \dots, c_n, z\rangle \;\big|\; c_1, \dots, c_n \in \mathbb{A} \text{ and } z \in [\![M]\!]_{\rho(x:=B)(y:=\{c_1,\dots,c_n\})}\big\} \\
&= \Psi(C \mapsto [\![M]\!]_{\rho(x:=B)(y:=C)}) \\
&= [\![P]\!]_{\rho(x:=B)}.
\end{aligned}$$

So $A \mapsto [\![P]\!]_{\rho(x:=A)}$ is monotone.

Now fix A and let $w \in [\![P]\!]_{\rho(x:=A)}$ be given. If $w = 0$ then $w \in [\![P]\!]_{\rho(x:=\emptyset)}$. Otherwise choose $c_1, \dots, c_n, z$ such that $w = \langle c_1, \dots, c_n, z\rangle$ and $z \in [\![M]\!]_{\rho(x:=A)(y:=\{c_1,\dots,c_n\})}$. By the inductive hypothesis we may find $A_0 \in \mathfrak{F}A$ such that $z \in [\![M]\!]_{\rho(x:=A_0)(y:=\{c_1,\dots,c_n\})}$. Thus

$$w = \langle c_1, \dots, c_n, z\rangle \in \Psi(C \mapsto [\![M]\!]_{\rho(x:=A_0)(y:=C)}) = [\![P]\!]_{\rho(x:=A_0)}.$$

So $A \mapsto [\![P]\!]_{\rho(x:=A)}$ is continuous. ∎

Theorem 2.4.3 *$(\mathfrak{P}\mathbb{A}, [\![\cdot]\!]_\rho)$ is a model for $\lambda\beta$.*

Proof: The axioms (R), (S), (T), and (C) follow immediately. We treat the remaining ones below.

(ξ) Let the terms P, Q be given and suppose $[\![P]\!]_\rho = [\![Q]\!]_\rho$ for all ρ. Then for all $A \in \mathfrak{P}\mathbb{A}$, $[\![P]\!]_{\rho(x:=A)} = [\![Q]\!]_{\rho(x:=A)}$. Thus

$$A \mapsto [\![P]\!]_{\rho(x:=A)} = A \mapsto [\![Q]\!]_{\rho(x:=A)},$$

implying

$$[\![\lambda x.P]\!]_\rho = \Psi(A \mapsto [\![P]\!]_{\rho(x:=A)}) = \Psi(A \mapsto [\![Q]\!]_{\rho(x:=A)}) = [\![\lambda x.Q]\!]_\rho$$

for all ρ.

(α) If z is neither bound nor free in P,

$$\begin{aligned}
[\![\lambda z.\{z/x\}P]\!]_\rho &= \Psi(A \mapsto [\![\{z/x\}P]\!]_{\rho(z:=A)}) \\
&= \Psi(A \mapsto [\![P]\!]_{\rho(x:=[\![z]\!]_{\rho(z:=A)})(z:=A)}) \\
&= \Psi(A \mapsto [\![P]\!]_{\rho(x:=A)(z:=A)}) \\
&= \Psi(A \mapsto [\![P]\!]_{\rho(x:=A)}) \\
&= [\![\lambda x.P]\!]_\rho
\end{aligned}$$

for all ρ.

(β) Observe

$$\begin{aligned}
[\![(\lambda x.P)(Q)]\!]_\rho &= \Phi([\![\lambda x.P]\!]_\rho)([\![Q]\!]_\rho) \\
&= \Phi(\Psi(A \mapsto [\![P]\!]_{\rho(x:=A)}))([\![Q]\!]_\rho) \\
&= (A \mapsto [\![P]\!]_{\rho(x:=A)})([\![Q]\!]_\rho) \\
&= [\![P]\!]_{\rho(x:=[\![Q]\!]_\rho)} \\
&= [\![\{Q/x\}P]\!]_\rho
\end{aligned}$$

for all ρ.

■

2.5 Generalized Application and Abstraction

In the last section we constructed an interpretation of application and abstraction for lambda *terms*. In this section, we generalize these notions to arbitrary continuous functions and elements of $\mathfrak{P}\mathbb{A}$.

The definition of App in Section 2.3 gives a natural interpretation of application between arbitrary elements of $\mathfrak{P}\mathbb{A}$.

Definition 2.5.1 *For $F, G \in \mathfrak{P}\mathbb{A}$,*

$$F(G) := \mathrm{App}(F, G) = \Phi(F)(G).$$

A generalized version of lambda abstraction is slightly trickier. We define it as follows.

Definition 2.5.2 *If the map $f\colon (\mathfrak{P}\mathbb{A})^n \to \mathfrak{P}\mathbb{A}$ is continuous then we define for $n \geq 1$,*

$$(\lambda X_1.f(X_1, X_2, \ldots, X_n)) := (X_2, \ldots, X_n) \mapsto \Psi(X_1 \mapsto f(X_1, X_2, \ldots, X_n)).$$

When $n = 1$ we say

$$(\lambda X_1.f(X_1)) := \Psi(f).$$

Some care must be taken to ensure that when an argument is "abstracted away" that we are left with a continuous function of the remaining arguments.

Theorem 2.5.3 *If $f\colon (\mathfrak{P}\mathbb{A})^n \to \mathfrak{P}\mathbb{A}$ is continuous and $n \geq 1$, then the map $g\colon (\mathfrak{P}\mathbb{A})^{n-1} \to \mathfrak{P}\mathbb{A}$ defined by*

$$g(X_2, \ldots, X_n) := (\lambda X_1.f(X_1, \ldots, X_n)).$$

is continuous in each argument.

Proof: We will show this for $n = 2$. The proof is similar (but considerably more tedious) for $n > 2$.

Let a continuous $f\colon (\mathfrak{P}\mathbb{A})^2 \to \mathfrak{P}\mathbb{A}$ be given and define

$$g(Y) := (\lambda X.f(X, Y)).$$

Let $z \in g(Y)$ be given. Obviously $0 \in g(Y) = \Psi(X \mapsto f(X, Y))$. So suppose $z \neq 0$. Choose $x_1, \ldots, x_n, y \in \mathbb{A}$ such that $\langle x_1, \ldots, x_n, y\rangle = z$. Then $y \in f(\{x_1, \ldots, x_n\}, Y)$. Since f is continuous in each argument, we may choose $Y_0 \in \mathfrak{F}Y$ such that $y \in f(\{x_1, \ldots, x_n\}, Y_0)$. So $\langle x_1, \ldots, x_n, y\rangle \in \Psi(X \mapsto f(X, Y_0)) = g(Y_0)$. Since y was arbitrary,

$$g(Y) \subseteq \bigcup_{Y_0 \in \mathfrak{F}Y} g(Y_0).$$

Now let $Y_0 \in \mathfrak{F}Y$ and $z \in g(Y_0) = \Psi(X \mapsto f(X, Y_0))$ be given. If $z = 0$ then we have immediately that $z = 0 \in g(Y)$. Otherwise, choose $x_1, \ldots, x_n, y \in \mathbb{A}$ such that $\langle x_1, \ldots, x_n, y\rangle = z$. Then So $y \in f(\{x_1, \ldots, x_n\}, Y_0)$. Since f is continuous and $Y_0 \subseteq Y$, $y \in f(\{x_1, \ldots, x_n\}, Y)$. Thus $z = \langle x_1, \ldots, x_n, y\rangle \in g(Y)$. Since z was arbitrary,

$$\bigcup_{Y_0 \in \mathfrak{F}Y} g(Y_0) \subseteq g(Y).$$

So g is continuous. ■

The use of the same abstraction and application notation as the syntax suggests that the conversion rules are satisfied. Obviously (R), (S), (T), and (C) are satisfied. If $f, g \in [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$ and $f = g$ then $\lambda X.f(X) = \Psi(f) = \Psi(g) = \lambda X.g(X)$, so (ξ) is satisfied. The (α) rule is purely a syntactic property, and is trivially satisfied under this notation — $\lambda X.f(X)$ and $\lambda Z.f(Z)$ are two ways of writing $\Psi(f)$, and hence are equal. The (β) rule is a consequence of the following: if $f \in [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$ and $Y \in \mathfrak{P}\mathbb{A}$ then

$$(\lambda X.f(X))(Y) = \Phi(\Psi(f))(Y) = f(Y).$$

In particular, all of these rules hold when $f = X \mapsto F(X)$ for some $F \in \mathfrak{P}\mathbb{A}$; for instance, the (β) rule then says that

$$(\lambda X.F(X))(Y) = F(Y).$$

Example 2.5.4 *Since $(X, Y) \mapsto X \cup Y$ is continuous, we may define*

$$\mathsf{U} := \lambda X.\lambda Y.(X \cup Y) \in \mathfrak{P}\mathbb{A}.$$

However, there is no known lambda term $\mathbf{U} \in \Lambda$ such that $[\![\mathbf{U}]\!]_\rho = \mathsf{U}$ for all ρ.

Example 2.5.5 *Notice that $\emptyset(X) = \emptyset$ and $\mathbb{A}(X) = \mathbb{A}$ for all $X \in \mathfrak{P}\mathbb{A}$.*

This next theorem says that App is completely additive in its first argument.

Theorem 2.5.6 *Let $\{F_i \mid i \in I\} \in \mathfrak{P}\mathbb{A}$ be given. Then*

$$\left(\bigcup_{i \in I} F_i\right)(X) = \bigcup_{i \in I} F_i(X)$$

for all $X \in \mathfrak{P}\mathbb{A}$.

Proof: Observe

$$\begin{aligned}
\left(\bigcup_{i \in I} F_i\right)(X) &= \left\{ y \in \mathbb{A} \mid \exists x_1, \ldots, x_n \in X \text{ such that } \langle x_1, \ldots, x_n, y\rangle \in \bigcup_{i \in I} F_i \right\} \\
&= \{ y \in \mathbb{A} \mid \exists i \in I, x_1, \ldots, x_n \in X \text{ such that } \langle x_1, \ldots, x_n, y\rangle \in F_i \} \\
&= \bigcup_{i \in I} \{ y \in C \mid \exists x_1, \ldots, x_n \in X \text{ such that } \langle x_1, \ldots, x_n, y\rangle \in F_i \} \\
&= \bigcup_{i \in I} F_i(X).
\end{aligned}$$

■

2.6 Combinators

This section defines a few combinators in the model and proves some useful results about them that we will need later.

Definition 2.6.1 *We define*

$$\begin{gathered}
F \circ G := \lambda X.F(G(X)), \\
\mathsf{I} := \lambda X.X, \\
\mathsf{Y} := \lambda F.(\lambda X.F(X(X)))(\lambda X.F(X(X))), \\
\mathsf{H} := \lambda F.\lambda X.F(X)
\end{gathered}$$

It is worth taking a moment and describing what some of these sets look like. For instance, $\mathsf{I} := \lambda X.X$ is seen to be

$$\mathsf{I} = \{0\} \cup \{ \langle x_1, \ldots, x_n, y\rangle \mid x_1, \ldots, x_n \in \mathbb{A} \text{ and } y \in \{x_1, \ldots, x_n\}\}$$

while for any $X \in \mathfrak{P}\mathbb{A}$,

$$\mathsf{U}(X) = \lambda Y.X \cup Y = \{0\} \cup \{ \langle x_1, \ldots, x_n, y\rangle \mid x_1, \ldots, x_n \in \mathbb{A} \text{ and } y \in \{x_1, \ldots, x_n\} \cup X\}.$$

Theorem 2.6.2 *For all $F \in \mathfrak{P}\mathbb{A}$, $F(\mathsf{Y}(F)) = \mathsf{Y}(F)$.*

Proof: This is a consequence of the fact that the model satisfies the (β) rule. We verify

$$\mathsf{Y}(F) = (\lambda X.F(X(X)))(\lambda X.F(X(X))) = F((\lambda X.F(X(X)))(\lambda X.F(X(X)))) = F(\mathsf{Y}(F)),$$

i.e., Y is a fixed-point combinator. ∎

Theorem 2.6.3 H *defined as above satisfies $\mathsf{H} = \mathsf{H} \circ \mathsf{H}$, and $\mathsf{H}(F) = \lambda X.F(X)$ is the maximum set G such that*

$$\forall X \in \mathfrak{P}\mathbb{A} \;\; G(X) = F(X).$$

Furthermore, the lattice of fixed points of H, ordered by inclusion, is isomorphic to the lattice $[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$ of continuous functions, ordered pointwise.

Proof: Notice that $\mathsf{H}(F) = (\lambda X.F(X)) = \Psi(\Phi(F))$. So for any $F \in \mathfrak{P}\mathbb{A}$,

$$\mathsf{H}(\mathsf{H}(F)) = \Psi(\Phi(\Psi(\Phi(F)))) = \Psi(\Phi(F)) = \mathsf{H}(F).$$

It follows then that

$$\mathsf{H} \circ \mathsf{H} = (\lambda F.\mathsf{H}(\mathsf{H}(F))) = (\lambda F.\mathsf{H}(F)) = \mathsf{H}.$$

Obviously $\mathsf{H}(F)(X) = F(X)$ for all $X \in \mathfrak{P}\mathbb{A}$. Now let a set $G \in \mathfrak{P}\mathbb{A}$ be given satisfying

$$\forall X \in \mathfrak{P}\mathbb{A} \;\; G(X) = F(X).$$

Claim: $G \subseteq \mathsf{H}(F)$. Let $g \in G$ be given. If $g = 0$, then $g \in \mathsf{H}(F) = \lambda X.F(X)$ trivially. Otherwise, we may choose $x_1, \dots, x_n, y \in \mathbb{A}$ such that $\langle x_1, \dots, x_n, y\rangle = g$. Then So $y \in G(\{x_1, \dots, x_n\}) = F(\{x_1, \dots, x_n\})$. So $g = \langle x_1, \dots, x_n, y\rangle \in (\lambda X.F(X)) = \mathsf{H}(F)$. So $G \subseteq \mathsf{H}(F)$. Hence $\mathsf{H}(F)$ is maximal.

Now we wish to show that the lattice of fixed points of H is isomorphic to $[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$. Notice that for $f \in [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$,

$$\mathsf{H}(\Psi(f)) = \Psi(\Phi(\Psi(f))) = \Psi(f).$$

So $\mathrm{Rng}\Psi \subseteq \mathcal{FP}(\mathsf{H})$. Now let $F \in \mathcal{FP}(\mathsf{H})$ be given and define $f := X \mapsto F(X)$. Then

$$\Psi(X \mapsto F(X)) = \Psi(\Phi(F)) = \mathsf{H}(F) = F.$$

So $\mathcal{FP}(\mathsf{H}) \subseteq \mathrm{Rng}\Psi$. Hence $\mathcal{FP}(\mathsf{H}) = \mathrm{Rng}\Psi$.

Now let $f, g \in [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$ be given and suppose $f \sqsubseteq g$ (i.e., for all $X \in \mathfrak{P}\mathbb{A}$, $f(X) \subseteq g(X)$.) Let $a \in \Psi(f)$ be given. If $a = 0$ then $a = 0 \in \Psi(g)$. Otherwise, choose $x_1, \dots, x_n, y \in \mathbb{A}$ such that $a = \langle x_1, \dots, x_n, y\rangle$. So $y \in f(\{x_1, \dots, x_n\})$. But $f \sqsubseteq g$, implying $y \in f(\{x_1, \dots, x_n\}) \subseteq g(\{x_1, \dots, x_n\})$. So $a \in \Psi(g)$. Thus $\Psi(f) \subseteq \Psi(g)$. So Ψ is monotonic.

Now let $f, g \in [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$ be given and suppose $\Psi(f) \subseteq \Psi(g)$. Let $X \in \mathfrak{P}\mathbb{A}$ and $y \in f(X)$ be given. Choose $x_1, \dots, x_n \in X$ such that $y \in f(\{x_1, \dots, x_n\})$. Then $\langle x_1, \dots, x_n, y\rangle \in \Psi(f) \subseteq \Psi(g)$. So $y \in \Psi(g)(\{x_1, \dots, x_n\}) \subseteq \Psi(g)(X)$. But $\Psi(g)(X) = g(X)$ so $y \in g(X)$. Hence $f \sqsubseteq g$ pointwise. So Ψ is order-preserving.

Since Ψ is order-preserving and monotonic onto $\mathcal{FP}(\mathsf{H})$, we have $\mathcal{FP}(\mathsf{H}) \cong [\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$. ∎

Notice that in this proof the 0 element was required in the definition of Ψ in order to guarantee the maximality of $\mathsf{H}(F)$.

The following theorem uses the power of the isomorphism constructed above to demonstrate an interesting set inclusion.

Theorem 2.6.4 $\mathsf{I} \subseteq \mathsf{H}$.

Proof: Let $F \in \mathfrak{P}\mathbb{A}$ be given. Clearly $\mathsf{H}(F)(X) = F(X)$ for all $X \in \mathfrak{P}\mathbb{A}$. By the first part of Theorem 2.6.3, this implies $\mathsf{I}(F) = F \subseteq \mathsf{H}(F)$. Since F was arbitrary, $\mathsf{I} \sqsubseteq \mathsf{H}$ as functions. By the second part of Theorem 2.6.3, this implies $\mathsf{I} \subseteq \mathsf{H}$ as sets in $\mathfrak{P}\mathbb{A}$. ■

2.7 Types as Closures

In the last section we found an operation H such that $\mathsf{I} \subseteq \mathsf{H} = \mathsf{H} \circ \mathsf{H}$, the lattice of fixed points of which was isomorphic to $[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$. It turns out that this is a specific instance of a more general notion of embedding mathematical types as sublattices of $\mathfrak{P}\mathbb{A}$.

Definition 2.7.1 *An element $C \in \mathfrak{P}\mathbb{A}$ is called a* **closure operation** *if $\mathsf{I} \subseteq C = C \circ C$.*

The range of a closure operation is thought of as the "type" that it represents. There are plenty of examples of closure operations — H is a closure operation representing the type $[\mathfrak{P}\mathbb{A} \to \mathfrak{P}\mathbb{A}]$, while I is a closure operation whose fixed point lattice is all of $\mathfrak{P}\mathbb{A}$ (and hence represents the "universal" type). In fact, a result from [Sco96] is that *any* algebraic lattice whose set of compact elements is bounded above by $|\mathbb{A}|$ can be represented as the fixed point lattice of a closure operation.

We will need a few specific types in order to build a model in the next section.

Definition 2.7.2 *Let C, D be closure operations. We define*

$$C \multimap D := \lambda F.(D \circ F \circ C).$$

It is easy to check that this is a closure operation if C and D are. Furthermore, we see that $F \in \mathcal{FP}(C \multimap D)$ precisely when $F(X) = D(F(C(X)))$ for all $X \in \mathfrak{P}\mathbb{A}$. The intuition behind this definition is that F makes distinctions only between elements of type C and outputs only elements of type D. For example, notice that $\mathsf{I} \multimap \mathsf{I} = \lambda F.\lambda X.F(X) = \mathsf{H}$, as we might expect. An important result is the following, proved in [Sco96].

Theorem 2.7.3 *If C and D are closure operations then*

$$\mathcal{FP}(C \multimap D) \cong [\mathcal{FP}(C) \to \mathcal{FP}(D)].$$

So function spaces between types are easy to embed in $\mathfrak{P}\mathbb{A}$. It will take a few more definitions to embed product spaces. For the following let $1 := \langle 0, 0\rangle \in \mathfrak{P}\mathbb{A}$.

Definition 2.7.4 *If $X, Y \in \mathfrak{P}\mathbb{A}$ then we define*

$$(X, Y) := \bigcup \{Z \in \mathfrak{P}\mathbb{A} \mid Z(\{0\}) = X \text{ and } Z(\{1\}) = Y\}.$$

It is easy to check that $X \mapsto (X, Y)$ and $Y \mapsto (X, Y)$ are continuous, and that $(X, Y)(\{0\}) = X$ and $(X, Y)(\{1\}) = Y$. In particular this means that $X \subseteq (X(\{0\}), X(\{1\}))$.

Definition 2.7.5 *Let C, D be closure operations. We define*

$$C \otimes D := \lambda X.(C(X(\{0\})), D(X(\{1\})))$$

Notice that $C \otimes D$ is a closure operation if C and D are. Fixed points of $C \otimes D$ look like pairs (X, Y) where $X \in \mathcal{FP}(C)$ and $Y \in \mathcal{FP}(D)$; i.e., $X(\{0\}) \in \mathcal{FP}(C)$ and $X(\{1\}) \in \mathcal{FP}(D)$ if and only if $X \in \mathcal{FP}(C \otimes D)$. This suggests the following theorem, also proved in [Sco96]:

Theorem 2.7.6 *If C and D are closure operations then*

$$\mathcal{FP}(C \otimes D) \cong \mathcal{FP}(C) \times \mathcal{FP}(D).$$

If C is a closure operation we often wish to define continuous functions $f\colon \mathcal{FP}(C) \to \mathfrak{P}\mathbb{A}$. We then run into a problem when trying to lambda abstract an argument from f in that we must ensure that f receives only arguments of type C. This motivates the following definition.

Definition 2.7.7

$$\lambda X{:}C.f(X) := \lambda X.f(C(X)).$$

Hence an argument X is "cast" to type C before being sent as an argument to f.

We finish up this section with the definition of a closure operation whose fixed point set *is the set of closure operations.*

Definition 2.7.8 *For $F \in \mathfrak{P}\mathbb{A}$ we define $F^{(0)} := \mathsf{I}$ and for $n > 0$, $F^{(n)} := F \circ F^{(n-1)}$. Then we define*

$$\mathsf{Clos} := \lambda F. \bigcup_{n=0}^{\infty} (\mathsf{I} \cup F)^{(n)}.$$

Theorem 2.7.9 *If $F \in \mathfrak{P}\mathbb{A}$ then $\mathsf{Clos}(F)$ is a closure operation. Furthermore, If $C \in \mathfrak{P}\mathbb{A}$ is a closure operation then $\mathsf{Clos}(C) = C$. Lastly, $\mathsf{Clos} = \mathsf{Clos} \circ \mathsf{Clos} = \mathsf{Clos}(\mathsf{Clos})$.*

Proof: First assume that $\mathsf{I} \subseteq F \in \mathfrak{P}\mathbb{A}$. We see that $\mathsf{Clos}(F) = \bigcup_{n=0}^{\infty} F^{(n)} = \mathsf{I} \cup \bigcup_{n=1}^{\infty} F^{(n)}$. So $\mathsf{I} \subseteq \mathsf{Clos}(F)$. We must check if $\mathsf{Clos}(F) \circ \mathsf{Clos}(F) = \mathsf{Clos}(F)$. Let $X \in \mathrm{Rng}(\mathsf{Clos}(F))$ be given, say $X = \mathsf{Clos}(F)(W)$. Since $\mathsf{I} \subseteq \mathsf{Clos}(F)$ we have $X \subseteq \mathsf{Clos}(F)(X)$. Now notice $\{F^{(n)}(W) \mid n \in \mathbb{N}\}$ is directed since $F^{(n)}(W) \subseteq F^{(n+1)}(W)$. Then

$$F(X) = F(\bigcup_{n=0}^{\infty} F^{(n)}(W)) = \bigcup_{n=0}^{\infty} F^{(n+1)}(W) \subseteq X.$$

So $F^{(n)}(X) \subseteq X$ for all n, implying $\mathsf{Clos}(F)(X) \subseteq X$. Hence $\mathsf{Clos}(F)(X) = X$. So $\mathsf{Clos}(F) \circ \mathsf{Clos}(F) = \mathsf{Clos}(F)$. Now if $\mathsf{I} \not\subseteq F$ then we simply observe that $\mathsf{Clos}(F) = \mathsf{Clos}(\mathsf{I} \cup F)$ to get the same result.

Now let C be any closure operation. Observe that $(\mathsf{I} \cup C)^{(n)} = C$ for all $n \geq 1$, so

$$\mathsf{Clos}(C) = \bigcup_{n=0}^{\infty} (\mathsf{I} \cup C)^{(n)} = \mathsf{I} \cup C = C.$$

Since $\mathsf{Clos}(F)$ is a closure operation for any F, it follows that $\mathsf{Clos}(\mathsf{Clos}(F)) = \mathsf{Clos}(F)$. Hence $\mathsf{Clos} \circ \mathsf{Clos} = \mathsf{Clos}$. Plus, it is immediate from the definition that $F \subseteq \mathsf{Clos}(F)$, so $\mathsf{I} \subseteq \mathsf{Clos}$ and hence Clos is itself a closure operation. So $\mathsf{Clos}(\mathsf{Clos}) = \mathsf{Clos}$. ∎

We now have all of the tools we need to build a model for the extensions of the lambda calculus. We have defined a model and have developed methods embedding continuous functions into the model. Furthermore, we have established techniques for embedding types into the model. These techniques will be employed in the following sections to find special types that yield a model for first λEP then λAL.

3 A Model for the Lambda Calculus with Explicit Products

In this section we shall present a nontrivial model first given by Scott that satisfies the rules listed in Section 1.3.

3.1 A Special Type: D

Let A be any closure operation in $\mathfrak{P}\mathbb{A}$. Define B by

$$\mathsf{B} := \mathsf{Y}(\lambda X.\mathsf{Clos}(X \otimes (\mathsf{A} \otimes X))).$$

Theorem 3.1.1

$$\mathsf{B} = \mathsf{B} \otimes (\mathsf{A} \otimes \mathsf{B}).$$

Proof: First notice that B is a fixed point of $\lambda X.\mathsf{Clos}(X \otimes (\mathsf{A} \otimes X))$, so $\mathsf{B} = \mathsf{Clos}(\mathsf{B} \otimes (\mathsf{A} \otimes \mathsf{B}))$. So $\mathsf{Clos}(\mathsf{B}) = \mathsf{B}$. Thus $\mathsf{A} \otimes \mathsf{B}$ is a closure operation, implying $\mathsf{Clos}(\mathsf{B} \otimes (\mathsf{A} \otimes \mathsf{B})) = \mathsf{B} \otimes (\mathsf{A} \otimes \mathsf{B})$. Hence $\mathsf{B} = \mathsf{B} \otimes (\mathsf{A} \otimes \mathsf{B})$ as desired. ∎

Now define

$$\mathsf{C} := \mathsf{A} \otimes \mathsf{B},$$

and observe $\mathsf{B} = \mathsf{B} \otimes \mathsf{C}$ by Theorem 3.1.1.

Theorem 3.1.2 (Scott) $\mathcal{FP}(\mathsf{C}) \times \mathcal{FP}(\mathsf{C}) \cong \mathcal{FP}(\mathsf{C})$.

This is implied by Theorem 3.1.4, proved below. To give a rough idea of how this isomorphism proceeds, we follow:

$$\begin{aligned}
\mathcal{FP}(\mathsf{C}) \times \mathcal{FP}(\mathsf{C}) &= \mathcal{FP}(\mathsf{A} \otimes \mathsf{B}) \times \mathcal{FP}(\mathsf{C}) \\
&\cong (\mathcal{FP}(\mathsf{A}) \times \mathcal{FP}(\mathsf{B})) \times \mathcal{FP}(\mathsf{C}) \\
&\cong \mathcal{FP}(\mathsf{A}) \times (\mathcal{FP}(\mathsf{B}) \times \mathcal{FP}(\mathsf{C})) \\
&\cong \mathcal{FP}(\mathsf{A}) \times (\mathcal{FP}(\mathsf{B} \otimes \mathsf{C})) \\
&= \mathcal{FP}(\mathsf{A}) \times \mathcal{FP}(\mathsf{B}) \\
&\cong \mathcal{FP}(\mathsf{A} \otimes \mathsf{B}) \\
&= \mathcal{FP}(\mathsf{C})
\end{aligned}$$

We will make this isomorphism more explicit using the following definitions.

Definition 3.1.3 *The maps*

$$\begin{aligned}
[\cdot,\cdot]_{\mathsf{C}} &: \mathcal{FP}(\mathsf{C}) \times \mathcal{FP}(\mathsf{C}) \to \mathcal{FP}(\mathsf{C}), \\
\pi_0^{\mathsf{C}} &: \mathcal{FP}(\mathsf{C}) \to \mathcal{FP}(\mathsf{C}), \\
\pi_1^{\mathsf{C}} &: \mathcal{FP}(\mathsf{C}) \to \mathcal{FP}(\mathsf{C})
\end{aligned}$$

are defined as follows:

$$\begin{aligned}
[F,G]_{\mathsf{C}} &:= (F(\{0\}), (F(\{1\}), G)), \\
\pi_0^{\mathsf{C}}(F) &:= (F(\{0\}), F(\{1\})(\{0\})), \\
\pi_1^{\mathsf{C}}(F) &:= F(\{1\})(\{1\}).
\end{aligned}$$

A quick check of the types of each of the components of F and G shows that these function are indeed well-defined. For instance, if $F \in \mathcal{FP}(\mathsf{C})$ then $F(\{0\}) \in \mathcal{FP}(\mathsf{A})$ and $F(\{1\}) \in \mathcal{FP}(\mathsf{B})$. So $F(\{1\})(\{0\}) \in \mathcal{FP}(\mathsf{B})$, implying $\pi_0^{\mathsf{C}}(F) = (F(\{0\}), F(\{1\})(\{0\})) \in \mathcal{FP}(\mathsf{A} \otimes \mathsf{B}) = \mathcal{FP}(\mathsf{C})$.

The notation used for these functions suggests that they are the pairing and projection functions that yield the isomorphism $\mathcal{FP}(\mathsf{C}) \times \mathcal{FP}(\mathsf{C}) \cong \mathcal{FP}(\mathsf{C})$. The following theorem confirms that this is the case.

Theorem 3.1.4 *For $F, G \in \mathcal{FP}(\mathsf{C})$, the following are satisfied:*

(i) $\pi_0^{\mathsf{C}}([F, G]_{\mathsf{C}}) = F$,

(ii) $\pi_1^{\mathsf{C}}([F, G]_{\mathsf{C}}) = G$,

(iii) $[\pi_0^{\mathsf{C}}(F), \pi_1^{\mathsf{C}}(F)]_{\mathsf{C}} = F$.

Proof:

(i)

$$\begin{aligned} \pi_0^{\mathsf{C}}([F, G]_{\mathsf{C}}) &= \pi_0^{\mathsf{C}}((F(\{0\}), (F(\{1\}), G))) \\ &= (F(\{0\}), F(\{1\})) \\ &= F \text{ (since } F \in \mathcal{FP}(\mathsf{C})); \end{aligned}$$

(ii)

$$\pi_1^{\mathsf{C}}([F, G]_{\mathsf{C}}) = \pi_1^{\mathsf{C}}((F(\{0\}), (F(\{1\}), G))) = G;$$

(iii)

$$\begin{aligned} [\pi_0^{\mathsf{C}}(F), \pi_1^{\mathsf{C}}(F)]_{\mathsf{C}} &= [(F(\{0\}), F(\{1\})(\{0\})), F(\{1\})(\{1\})]_{\mathsf{C}} \\ &= (F(\{0\}), (F(\{1\})(\{0\}), F(\{1\})(\{1\}))) \\ &= (F(\{0\}), F(\{1\})) \\ &= F. \end{aligned}$$

So these are the pairing and projection mappings that make $\mathcal{FP}(\mathsf{C}) \times \mathcal{FP}(\mathsf{C}) \cong \mathcal{FP}(\mathsf{C})$ explicit. ■

Now define

$$\mathsf{D} := \mathsf{Y}(\lambda X.\mathsf{Clos}(X \multimap \mathsf{C})).$$

Observe $\mathsf{D} = \mathsf{Clos}(\mathsf{D} \multimap \mathsf{C})$. So $\mathsf{Clos}(\mathsf{D}) = \mathsf{D}$, implying that

$$\mathsf{D} \multimap \mathsf{C} = \mathsf{Clos}(\mathsf{D} \multimap \mathsf{C}) = \mathsf{D}.$$

Because of the highly specialized construction of D, we find that it satisfies a number of interesting properties.

Theorem 3.1.5 (Scott) $\mathcal{FP}(\mathsf{D}) \times \mathcal{FP}(\mathsf{D}) \cong \mathcal{FP}(\mathsf{D})$.

As in Theorems 3.1.2 and 3.1.4, we start with a rough sketch of the isomorphism, then construct specific functions that yield the isomorphism. We expect the following to hold:

$$\begin{aligned}
\mathcal{FP}(\mathsf{D}) \times \mathcal{FP}(\mathsf{D}) &= \mathcal{FP}(\mathsf{D} \multimap \mathsf{C}) \times \mathcal{FP}(\mathsf{D} \multimap \mathsf{C}) \\
&\cong [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{C})] \times [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{C})] \\
&\cong [\mathcal{FP}(\mathsf{D}) \to (\mathcal{FP}(\mathsf{C}) \times \mathcal{FP}(\mathsf{C}))] \\
&\cong [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{C})] \\
&\cong \mathcal{FP}(\mathsf{D} \multimap \mathsf{C}) \\
&= \mathcal{FP}(\mathsf{D}).
\end{aligned}$$

Informally, this says that elements of type D can be viewed as pairs of elements from type D, and vice versa. In order to make this isomorphism explicit, we provide the pairing and unpairing operations as follows. The pair of elements F, G of type D is denoted $[F, G]_{\mathsf{D}}$ and is defined to be

$$[F, G]_{\mathsf{D}} := \lambda X{:}\mathsf{D}.[F(X), G(X)]_{\mathsf{C}}$$

while the projection functions $\pi_0^{\mathsf{D}}(F), \pi_1^{\mathsf{D}}(G)$ which yield the reverse direction of the isomorphism are defined to be

$$\pi_0^{\mathsf{D}}(F) := \lambda X{:}\mathsf{D}.\pi_0^{\mathsf{C}}(F(X)), \qquad \pi_1^{\mathsf{D}}(F) := \lambda X{:}\mathsf{D}.\pi_1^{\mathsf{C}}(F(X)).$$

Observe that when $F, G \in \mathcal{FP}(\mathsf{D}) = \mathcal{FP}(\mathsf{D} \multimap \mathsf{C})$ we have $F(X), G(X) \in \mathcal{FP}(\mathsf{C})$ for all $X \in \mathcal{FP}(\mathsf{D})$. So $[F(X), G(X)]_{\mathsf{C}} \in \mathcal{FP}(\mathsf{C})$, implying $[F, G]_{\mathsf{D}} = \lambda X{:}\mathsf{D}.[F(X), G(X)]_{\mathsf{C}} \in \mathcal{FP}(\mathsf{D} \multimap \mathsf{C}) = \mathcal{FP}(\mathsf{D})$. So $[\cdot, \cdot]_{\mathsf{D}}$ is well-defined. Similarly, typechecking π_0^{D} and π_1^{D} confirms that they are well-defined. We now verify that these do indeed behave as product and projection mappings.

Theorem 3.1.6 *For $F, G \in \mathcal{FP}(\mathsf{D})$, the following are satisfied:*

(i) $\pi_0^{\mathsf{D}}([F, G]_{\mathsf{D}}) = F$,

(ii) $\pi_1^{\mathsf{D}}([F, G]_{\mathsf{D}}) = G$,

(iii) $[\pi_0^{\mathsf{D}}(F), \pi_1^{\mathsf{D}}(F)]_{\mathsf{D}} = F$.

Proof:

(i)

$$\begin{aligned}
\pi_0^{\mathsf{D}}([F, G]_{\mathsf{D}}) &= \lambda X{:}\mathsf{D}.\pi_0^{\mathsf{C}}([F, G]_{\mathsf{D}}(X)) \\
&= \lambda X{:}\mathsf{D}.\pi_0^{\mathsf{C}}((\lambda Y{:}\mathsf{D}.[F(Y), G(Y)]_{\mathsf{C}})(X)) \\
&= \lambda X{:}\mathsf{D}.\pi_0^{\mathsf{C}}([F(X), G(X)]_{\mathsf{C}}) \\
&= \lambda X{:}\mathsf{D}.F(X) \\
&= F.
\end{aligned}$$

(ii) A similar argument as (i) is used to show $\pi_1^{\mathsf{D}}([F, G]_{\mathsf{D}}) = G$.

(iii)

$$\begin{aligned}
[\pi_0^{\mathsf{D}}(F), \pi_1^{\mathsf{D}}(F)]_{\mathsf{D}} &= \lambda X{:}\mathsf{D}.[\pi_0^{\mathsf{D}}(F)(X), \pi_1^{\mathsf{D}}(F)(X)]_{\mathsf{C}} \\
&= \lambda X{:}\mathsf{D}.[(\lambda Y{:}\mathsf{D}.\pi_0^{\mathsf{C}}(F(Y)))(X), (\lambda Y{:}\mathsf{D}.\pi_1^{\mathsf{C}}(F(Y)))(X)]_{\mathsf{C}} \\
&= \lambda X{:}\mathsf{D}.[\pi_0^{\mathsf{C}}(F(X)), \pi_1^{\mathsf{C}}(F(X))]_{\mathsf{C}} \\
&= \lambda X{:}\mathsf{D}.F(X) \\
&= F.
\end{aligned}$$

■

Theorem 3.1.7 (Scott) $[\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})] \cong \mathcal{FP}(\mathsf{D})$.

Functions that give this isomorphism are provided and proved below. The intuition behind this isomorphism is as follows:

$$\begin{aligned}
[\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})] &= [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D} \multimap \mathsf{C})] \\
&\cong [\mathcal{FP}(\mathsf{D}) \to [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{C})]] \\
&\cong [(\mathcal{FP}(\mathsf{D}) \times \mathcal{FP}(\mathsf{D})) \to \mathcal{FP}(\mathsf{C})] \\
&\cong [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{C})] \\
&\cong \mathcal{FP}(\mathsf{D} \multimap \mathsf{C}) \\
&= \mathcal{FP}(\mathsf{D}).
\end{aligned}$$

Reading from bottom to top, this equation says that every element of D can be viewed as a function from D to D. In particular, to make this direction of the isomorphism explicit, we define the map $\Phi_{\mathsf{D}}\colon \mathcal{FP}(\mathsf{D}) \to [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})]$ by

$$\Phi_{\mathsf{D}}(F) := X \mapsto \lambda Y{:}\mathsf{D}.F([X, Y]_{\mathsf{D}}),$$

while the reverse direction $\Psi_{\mathsf{D}}\colon [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})] \to \mathcal{FP}(\mathsf{D})$ is defined as

$$\Psi_{\mathsf{D}}(f) := \lambda X{:}\mathsf{D}.f(\pi_0^{\mathsf{D}}(X))(\pi_1^{\mathsf{D}}(X)).$$

Theorem 3.1.8 *Ψ_{D} and Φ_{D} are well-defined.*

Proof: Let $F \in \mathcal{FP}(\mathsf{D})$ be given. Claim: $\Phi_{\mathsf{D}}(F) \in [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})]$. Let $X \in \mathcal{FP}(\mathsf{D})$ be given. Let $G := \Phi_{\mathsf{D}}(F)(X)$. Then $G = \lambda Y{:}\mathsf{D}.F([X, Y]_{\mathsf{D}})$. So

$$\begin{aligned}
\mathsf{C} \circ G \circ \mathsf{D} &= \lambda Z.\mathsf{C}(G(\mathsf{D}(Z))) \\
&= \lambda Z.\mathsf{C}(\lambda Y.F([X, Y]_{\mathsf{D}})(\mathsf{D}(Z))) \\
&= \lambda Z.\mathsf{C}(F([X, \mathsf{D}(Z)]_{\mathsf{D}})) \\
&= \lambda Z.F([X, \mathsf{D}(Z)]_{\mathsf{D}}) \\
&= \lambda Z{:}\mathsf{D}.F([X, Z]_{\mathsf{D}}) \\
&= G.
\end{aligned}$$

So $G \in \mathcal{FP}(\mathsf{D} \multimap \mathsf{C}) = \mathcal{FP}(\mathsf{D})$. Since X was arbitrary, $\Phi_{\mathsf{D}}(F) \in [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})]$. So Φ_{D} is well-defined.

Now let $f \in [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})]$ be given and let $F := \Psi_{\mathsf{D}}(f)$. Observe

$$\begin{array}{rcl} \mathsf{C} \circ F \circ \mathsf{D} & = & \lambda Z.\mathsf{C}(F(\mathsf{D}(Z))) \\ & = & \lambda Z{:}\mathsf{D}.\mathsf{C}(\lambda X.f(\pi_0^{\mathsf{D}}(X))(\pi_1^{\mathsf{D}}(X))(Z)) \\ & = & \lambda Z{:}\mathsf{D}.\mathsf{C}(f(\pi_0^{\mathsf{D}}(Z))(\pi_1^{\mathsf{D}}(Z))). \end{array}$$

But for $Z \in \mathcal{FP}(\mathsf{D})$, $\pi_0^{\mathsf{D}}(Z) \in \mathcal{FP}(\mathsf{D})$ so $f(\pi_0^{\mathsf{D}}(Z)) \in \mathcal{FP}(\mathsf{D}) = \mathcal{FP}(\mathsf{D} \circ\!\!\to \mathsf{C})$. Since $\pi_1^{\mathsf{D}}(Z) \in D$, it follows that $f(\pi_0^{\mathsf{D}}(Z))(\pi_1^{\mathsf{D}}(Z)) \in \mathcal{FP}(\mathsf{C})$. So

$$\mathsf{C}(f(\pi_0^{\mathsf{D}}(Z))(\pi_1^{\mathsf{D}}(Z))) = f(\pi_0^{\mathsf{D}}(Z))(\pi_1^{\mathsf{D}}(Z)),$$

hence

$$\mathsf{C} \circ F \circ \mathsf{D} = \lambda Z{:}\mathsf{D}.f(\pi_0^{\mathsf{D}}(Z))(\pi_1^{\mathsf{D}}(Z)) = F.$$

So $F \in \mathcal{FP}(D \circ\!\!\to C) = \mathcal{FP}(\mathsf{D})$. Hence Ψ_{D} is well-defined. ■

Theorem 3.1.9 $\Phi_{\mathsf{D}} \circ \Psi_{\mathsf{D}} = \mathrm{id}_{[\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})]}$ and $\Psi_{\mathsf{D}} \circ \Phi_{\mathsf{D}} = \mathrm{id}_{\mathcal{FP}(\mathsf{D})}$.

Proof: Let $X \in \mathcal{FP}(\mathsf{D})$ be given. Observe

$$\begin{array}{rcl} \Psi_{\mathsf{D}}(\Phi_{\mathsf{D}}(F)) & = & \Psi_{\mathsf{D}}(X \mapsto \lambda Y{:}\mathsf{D}.F([X,Y]_{\mathsf{D}})) \\ & = & \lambda X{:}\mathsf{D}.(\lambda Y{:}\mathsf{D}.F([\pi_0^{\mathsf{D}}(X),Y]_{\mathsf{D}}))(\pi_1^{\mathsf{D}}(X)) \\ & = & \lambda X{:}\mathsf{D}.F([\pi_0^{\mathsf{D}}(X),\pi_1^{\mathsf{D}}(X)]_{\mathsf{D}}) \\ & = & \lambda X{:}\mathsf{D}.F(X) \\ & = & F. \end{array}$$

So $\Psi_{\mathsf{D}} \circ \Phi_{\mathsf{D}} = \mathrm{id}_{\mathcal{FP}(\mathsf{D})}$. Now let $f \in [\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})]$ be given. Notice

$$\begin{array}{rcl} \Phi_{\mathsf{D}}(\Psi_{\mathsf{D}}(f)) & = & \Phi_{\mathsf{D}}(\lambda X{:}\mathsf{D}.f(\pi_0^{\mathsf{D}}(X))(\pi_1^{\mathsf{D}}(X))) \\ & = & X \mapsto \lambda Y{:}\mathsf{D}.f(\pi_0^{\mathsf{D}}([X,Y]_{\mathsf{D}}))(\pi_1^{\mathsf{D}}([X,Y]_{\mathsf{D}})) \\ & = & X \mapsto \lambda Y{:}\mathsf{D}.f(X)(Y) \\ & = & X \mapsto f(X) \\ & = & f. \end{array}$$

Hence $\Phi_{\mathsf{D}} \circ \Psi_{\mathsf{D}} = \mathrm{id}_{[\mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})]}$. ■

Since we have an *isomorphism* between $\mathcal{FP}(\mathsf{D})$ and its function space, we actually have an *extensional* model. This will be formalized in Section 3.3.

3.2 Lattice structure of $\mathcal{FP}(\mathsf{D})$

Before proceeding further we should digress briefly to discuss the structure of $\mathcal{FP}(\mathsf{D})$. We have not yet confirmed that solution of the domain equations that yielded D is not trivial. That is, if it turns out that the D found above is equal to $\mathbb{A}$ then we will have $\mathsf{D}(X) = \mathbb{A}$ for all $X \in \mathfrak{P}\mathbb{A}$, i.e., $\mathcal{FP}(\mathsf{D})$ is a singleton. We confirm that this is not the case.

Theorem 3.2.1

$$|\mathcal{FP}(\mathsf{D})| \geq |\mathcal{FP}(\mathsf{A})|.$$

Specifically, there exists a closure operation $\mathsf{E} \in \mathcal{FP}(D \circ\!\!\rightarrow D)$ *such that*

$$\mathcal{FP}(\mathsf{A}) \cong \mathcal{FP}(\mathsf{E}) \subseteq \mathcal{FP}(\mathsf{D}).$$

Proof: Define

$$\mathsf{E} := \lambda F.\lambda X.(\mathsf{D}(F)(\mathbb{A})(\{0\}), \mathbb{A}).$$

Notice that $F \subseteq \mathsf{D}(F)$ so $F(\mathbb{A})(\{0\}) \subseteq \mathsf{D}(F)(\mathbb{A})(\{0\})$. Thus

$$F(X) \subseteq F(\mathbb{A}) \subseteq (F(\mathbb{A})(\{0\}), F(\mathbb{A})(\{1\})) \subseteq (\mathsf{D}(F)(\mathbb{A})(\{0\}), \mathbb{A}) = \mathsf{E}(F)(X).$$

for all X. So $F \subseteq \mathsf{E}(F)$, implying $\mathsf{I} \subseteq \mathsf{E}$.

For any $F \in \mathfrak{P}\mathbb{A}$ is easy to show that $\mathsf{E}(\mathsf{E}(F)) = \mathsf{E}(F)$ (massive use of the (β) rule). So $\mathsf{I} \subseteq \mathsf{E} = \mathsf{E} \circ \mathsf{E}$. Hence E is a closure operation. Furthermore from the definition we see that $\mathsf{E}(F) = \mathsf{E}(\mathsf{D}(F))$ for all F, while $\mathsf{E}(F) \in \mathcal{FP}(D \circ\!\!\rightarrow C) = \mathcal{FP}(D)$ so $\mathsf{D} \circ \mathsf{E} \circ \mathsf{D} = \mathsf{E}$. Thus $\mathsf{E} \in \mathcal{FP}(D \circ\!\!\rightarrow D)$.

Lastly, the combinators

$$\begin{array}{rcl} \lambda F.\mathsf{E}(F)(\mathbb{A})(\{0\}) & : & \mathsf{E} \circ\!\!\rightarrow \mathsf{A}, \\ \lambda X.\lambda Y.(\mathsf{A}(X), \mathbb{A}) & : & \mathsf{A} \circ\!\!\rightarrow \mathsf{E}. \end{array}$$

yield the desired isomorphism $\mathcal{FP}(\mathsf{E}) \cong \mathcal{FP}(\mathsf{A})$. ■

So by an appropriate choice of A, we can include in D whatever types we wish. For example, notice that if $\mathsf{A} = \mathsf{I}$ then the entire lattice structure of $\mathfrak{P}\mathbb{A}$ is available as a retract of $\mathcal{FP}(\mathsf{D})$.

Definition 3.2.2

$$\bot := \mathsf{D}(\emptyset), \qquad \top := \mathsf{D}(\mathbb{A}) = \mathbb{A}.$$

Theorem 3.2.3

$$[\bot, \bot]_{\mathsf{D}} = \bot, \qquad [\top, \top]_{\mathsf{D}} = \top.$$

Proof: By its construction, $[\cdot, \cdot]_{\mathsf{D}}$ is continuous (as are all of the other operations defined in this section). Since $\bot \subseteq \pi_0^{\mathsf{D}}(X)$ and $\bot \subseteq \pi_1^{\mathsf{D}}(X)$ for all $X \in \mathcal{FP}(\mathsf{D})$ we have that

$$[\bot, \bot]_{\mathsf{D}} \subseteq [\pi_0^{\mathsf{D}}(X), \pi_1^{\mathsf{D}}(X)]_{\mathsf{D}} = X$$

for all $X \in \mathcal{FP}(\mathsf{D})$. So $[\bot, \bot]_{\mathsf{D}}$ is a least element. But the least element is unique so $\bot = [\bot, \bot]_{\mathsf{D}}$. A symmetric argument is employed to show $[\top, \top]_{\mathsf{D}} = \top$. ■

For the remainder of this work we will assume that $|\mathcal{FP}(\mathsf{A})| \geq 2$, which enforces $\bot \neq \top$.

3.3 Modeling λEP

Definition 3.3.1 *Let ρ be a map from variables to $\mathcal{FP}(\mathsf{D})$. We define interpretation map*

$$[\![\cdot]\!]_\rho \colon \Lambda_{\mathrm{EP}} \to \mathcal{FP}(\mathsf{D})$$

inductively as follows:

(i) $[\![\mathbf{fst}]\!]_\rho = (\Lambda X{:}\mathsf{D}.\pi_0^{\mathsf{D}}(X)),\ [\![\mathbf{snd}]\!]_\rho = (\Lambda X{:}\mathsf{D}.\pi_1^{\mathsf{D}}(X))$,

(ii) $[\![x]\!]_\rho = \rho(x)$,

(iii) $[\![P(Q)]\!]_\rho = \Phi_{\mathsf{D}}([\![P]\!]_\rho)([\![Q]\!]_\rho)$,

(iv) $[\![(\lambda x.P)]\!]_\rho = \Psi_{\mathsf{D}}(a \mapsto [\![P]\!]_{\rho(x:=a)})$,

(v) $[\![(P,Q)]\!]_\rho = [[\![P]\!]_\rho, [\![Q]\!]_\rho]_{\mathsf{D}}$.

Theorem 3.3.2 $(\mathcal{FP}(\mathsf{D}), [\![\cdot]\!]_\rho)$ *is a model for $\lambda EP + (\eta) + (\mathrm{SP})$.*

Proof: The axioms (R), (S), (T), (C), (ξ), (α), and (β) follow just as in the proof of Theorem 2.4.3. So to get λEP we must verify (γ) and (δ):

$$\begin{aligned}
[\![(P,Q)(R)]\!]_\rho &= \Phi_{\mathsf{D}}([\![(P,Q)]\!]_\rho)([\![R]\!]_\rho)\\
&= (X \mapsto \lambda Y{:}\mathsf{D}.[\![(P,Q)]\!]_\rho([X,Y]_{\mathsf{D}}))([\![R]\!]_\rho)\\
&= \lambda Y{:}\mathsf{D}.[\![(P,Q)]\!]_\rho([[\![R]\!]_\rho,Y]_{\mathsf{D}})\\
&= \lambda Y{:}\mathsf{D}.[[\![P]\!]_\rho,[\![Q]\!]_\rho]_{\mathsf{D}}([[\![R]\!]_\rho,Y]_{\mathsf{D}})\\
&= \lambda Y{:}\mathsf{D}.(\lambda X{:}\mathsf{D}.[[\![P]\!]_\rho(X),[\![Q]\!]_\rho(X)]_{\mathsf{C}})([[\![R]\!]_\rho,Y]_{\mathsf{D}})\\
&= \lambda Y{:}\mathsf{D}.[[\![P]\!]_\rho([[\![R]\!]_\rho,Y]_{\mathsf{D}}),[\![Q]\!]_\rho([[\![R]\!]_\rho,Y]_{\mathsf{D}})]_{\mathsf{C}}\\
&= \lambda Y{:}\mathsf{D}.[\Phi_{\mathsf{D}}([\![P]\!]_\rho)([\![R]\!]_\rho)(Y),\Phi_{\mathsf{D}}([\![Q]\!]_\rho)([\![R]\!]_\rho)(Y)]_{\mathsf{C}}\\
&= \lambda Y{:}\mathsf{D}.[[\![P(R)]\!]_\rho(Y),[\![Q(R)]\!]_\rho(Y)]_{\mathsf{C}}\\
&= [[\![P(R)]\!]_\rho,[\![Q(R)]\!]_\rho]_{\mathsf{D}}\\
&= [\![(P(R),Q(R))]\!]_\rho,
\end{aligned}$$

and

$$\begin{aligned}
[\![\lambda x.(P,Q)]\!]_\rho &= \Psi_{\mathsf{D}}(a \mapsto [\![(P,Q)]\!]_{\rho(x:=a)})\\
&= \Psi_{\mathsf{D}}(a \mapsto [[\![P]\!]_{\rho(x:=a)},[\![Q]\!]_{\rho(x:=a)}]_{\mathsf{D}})\\
&= \lambda Y{:}\mathsf{D}.[[\![P]\!]_{\rho(x:=\pi_0^{\mathsf{D}}(Y))},[\![Q]\!]_{\rho(x:=\pi_0^{\mathsf{D}}(Y))}]_{\mathsf{D}}(\pi_1^{\mathsf{D}}(Y))\\
&= \lambda Y{:}\mathsf{D}.[[\![P]\!]_{\rho(x:=\pi_0^{\mathsf{D}}(Y))}(\pi_1^{\mathsf{D}}(Y)),[\![Q]\!]_{\rho(x:=\pi_0^{\mathsf{D}}(Y))}(\pi_1^{\mathsf{D}}(Y))]_{\mathsf{C}}\\
&= \lambda Y{:}\mathsf{D}.[\Psi_{\mathsf{D}}(a \mapsto [\![P]\!]_{\rho(x:=a)})(Y),\Psi_{\mathsf{D}}(a \mapsto [\![Q]\!]_{\rho(x:=a)})(Y)]_{\mathsf{C}}\\
&= [\Psi_{\mathsf{D}}(a \mapsto [\![P]\!]_{\rho(x:=a)}),\Psi_{\mathsf{D}}(a \mapsto [\![Q]\!]_{\rho(x:=a)})]_{\mathsf{D}}\\
&= [[\![\lambda x.P]\!]_\rho,[\![\lambda x.Q]\!]_\rho]_{\mathsf{D}}\\
&= [\![(\lambda x.P,\lambda x.Q)]\!]_\rho.
\end{aligned}$$

Also

$$\begin{aligned}
[\![\mathbf{fst}((P,Q))]\!]_\rho &= (\Lambda X{:}\mathsf{D}.\pi_0^{\mathsf{D}}(X))[[[\![P]\!]_\rho,[\![Q]\!]_\rho]_{\mathsf{D}}]_{\mathsf{D}}\\
&= \pi_0^{\mathsf{D}}([[\![P]\!]_\rho,[\![Q]\!]_\rho]_{\mathsf{D}})\\
&= [\![P]\!]_\rho
\end{aligned}$$

and

$$\begin{aligned}
[\![\mathbf{snd}((P,Q))]\!]_\rho &= (\Lambda X{:}\mathsf{D}.\pi_1^{\mathsf{D}}(X))[[[\![P]\!]_\rho,[\![Q]\!]_\rho]_{\mathsf{D}}]_{\mathsf{D}} \\
&= \pi_1^{\mathsf{D}}([[\![P]\!]_\rho,[\![Q]\!]_\rho]_{\mathsf{D}}) \\
&= [\![Q]\!]_\rho
\end{aligned}$$

We see that this model also satisfies (SP):

$$[\![(\mathbf{fst}(P),\mathbf{snd}(P))]\!]_\rho = [\pi_0^{\mathsf{D}}([\![P]\!]_\rho),\pi_1^{\mathsf{D}}([\![P]\!]_\rho)]_{\mathsf{D}} = [\![P]\!]_\rho.$$

Now we see that (η) is satisfied: for any term P and variable x that is not free in P,

$$\begin{aligned}
[\![(\lambda x.P(x))]\!]_\rho &= \Psi_{\mathsf{D}}(A \mapsto [\![P(x)]\!]_{\rho(x:=A)}) \\
&= \Psi_{\mathsf{D}}(A \mapsto \Phi_{\mathsf{D}}([\![P]\!]_{\rho(x:=A)})([\![x]\!]_{\rho(x:=A)})) \\
&= \Psi_{\mathsf{D}}(A \mapsto \Phi_{\mathsf{D}}([\![P]\!]_\rho)(A)) \\
&= \Psi_{\mathsf{D}}(\Phi_{\mathsf{D}}([\![P]\!]_\rho)) \\
&= [\![P]\!]_\rho
\end{aligned}$$

■

3.4 Generalized Application and Abstraction

Just as in Section 2.5 we can define lambda abstraction of arguments for arbitrary continuous functions $f\colon (\mathcal{FP}(\mathsf{D}))^n \to \mathcal{FP}(\mathsf{D})$ and application between arbitrary elements $F, G \in \mathcal{FP}(\mathsf{D})$.

Definition 3.4.1 *For $f\colon (\mathcal{FP}(\mathsf{D}))^n \to \mathcal{FP}(\mathsf{D})$, we define for $n > 1$,*

$$(\Lambda X_1.f(X_1,\dots,X_n)) := (X_2,\dots,X_n) \mapsto \Psi_{\mathsf{D}}(X_1 \mapsto f(X_1,\dots,X_n))$$

and

$$\Lambda X.f(X) := \Psi_{\mathsf{D}}(f).$$

For any $F, G \in \mathcal{FP}(\mathsf{D})$ we define

$$F[G]_{\mathsf{D}} := \Phi_{\mathsf{D}}(F)(G).$$

We will use these notations interchangeably in the sequel.

Again, it is easy to check that these generalized versions of lambda abstraction and application satisfy the axioms for λEP. (R), (S), (T), (C), (ξ), and (α) are trivial, while the (β) rule

$$(\Lambda X.f(X))[Y]_{\mathsf{D}} = f(Y)$$

and the the (γ) rule

$$[F,G]_{\mathsf{D}}[X]_{\mathsf{D}} = [F[X]_{\mathsf{D}}, G[X]_{\mathsf{D}}]_{\mathsf{D}}.$$

hold by Theorem 3.3.2.

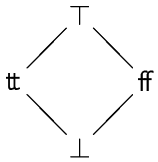
4 A Model for the Lambda Calculus with Applicative Lists

The model for λEP built on $\mathcal{FP}(\mathsf{D})$ in the last section turns out to have all of the elements needed for a model for λAL. In this section, we find interpretations for the atoms of λAL in $\mathcal{FP}(\mathsf{D})$, thus finding a model that satisfies the axioms from Section 1.2.

4.1 Booleans and Conditionals: true, false, and cond

To begin, we need to find interpretations of **true** and **false** in the model (denoted $\mathtt{tt}$ and $\mathtt{ff}$, respectively). The boolean type is traditionally represented as the four-element lattice $\mathbb{B}$ (shown in Figure 1), with the elements $\top$ and $\bot$ representing nonterminating computation or an error condition.

Figure 1: The lattice $\mathbb{B}$ of boolean values.



We wish to choose these elements so that there is continuous function

$$\mathsf{cond}\colon\ \mathbb{B} \times \mathcal{FP}(\mathsf{D}) \times \mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})$$

satisfying

$$\mathsf{cond}(x, y, z) = \begin{cases} \top & \text{if } x = \top, \\ y & \text{if } x = \mathtt{tt}, \\ z & \text{if } x = \mathtt{ff}, \\ \bot & \text{if } x = \bot. \end{cases} \tag{1}$$

In particular, it is crucial that $\mathtt{tt}$ and $\mathtt{ff}$ be incomparable; for example, if $\mathtt{ff} \subseteq \mathtt{tt}$, then any continuous candidate for cond satisfying the above would force

$$\top = \mathsf{cond}(\mathtt{ff}, \bot, \top) \subseteq \mathsf{cond}(\mathtt{tt}, \bot, \top) = \bot,$$

which is clearly a contradiction.

This next result gives us elements of the model and a closure operation corresponding to this type.

Theorem 4.1.1 *There exist elements* $\mathtt{tt}, \mathtt{ff}$ *of the model and a closure operation* bool *such that the lattice of fixed points of* bool *is* $\mathbb{B}$.

Proof: Following a suggestion of Scott, we define $\mathtt{tt} := [\top, \bot]_{\mathsf{D}}$ and $\mathtt{ff} := [\bot, \top]_{\mathsf{D}}$. Notice that

$$\begin{gathered} \bot = [\bot, \bot]_{\mathsf{D}} \subset [\top, \bot]_{\mathsf{D}} = \mathtt{tt} \subset [\top, \top]_{\mathsf{D}} = \top \\ \bot = [\bot, \bot]_{\mathsf{D}} \subset [\bot, \top]_{\mathsf{D}} = \mathtt{ff} \subset [\top, \top]_{\mathsf{D}} = \top \\ \mathtt{tt} = [\top, \bot]_{\mathsf{D}} \not\subseteq [\bot, \top]_{\mathsf{D}} = \mathtt{ff} \\ \mathtt{ff} = [\bot, \top]_{\mathsf{D}} \not\subseteq [\top, \bot]_{\mathsf{D}} = \mathtt{tt} \end{gathered}$$

as desired.

Now let $f\colon \mathcal{FP}(\mathsf{D}) \to \{\bot, \top\}$ be defined by $f(\bot) := \bot$ and $f(X) := \top$ for all $X \neq \bot$, and note that $X \subseteq f(X)$ for all X. Define $\mathsf{bool}\colon \mathcal{FP}(\mathsf{D}) \to \mathbb{B}$ by $\mathsf{bool}(X) := [f(\pi_0^{\mathsf{D}}(X)), f(\pi_1^{\mathsf{D}}(X))]_{\mathsf{D}}$. Since bool is the composition of continuous functions, it follows that bool is continuous. Furthermore, notice that $X = [\pi_0^{\mathsf{D}}(X), \pi_1^{\mathsf{D}}(X)]_{\mathsf{D}} \subseteq [f(\pi_0^{\mathsf{D}}(X)), f(\pi_1^{\mathsf{D}}(X))]_{\mathsf{D}} = \mathsf{bool}(X)$. Let $X = [U, V]_{\mathsf{D}}$ be in the range of bool. Then U and V are in the range of f; that is, $U, V \in \{\bot, \top\}$. So $X \in \mathbb{B}$. Lastly, it is easy to verify that $\{\bot, \mathsf{tt}, \mathsf{ff}, \top\}$ are fixed points of bool. So bool is a closure operation with fixed point lattice $\mathbb{B}$. ■

Now that we have introduced booleans, it is only a small step to get an "if-then" construct.

Theorem 4.1.2 *There exists a continuous mapping* $\mathsf{cond}\colon \mathbb{B} \times \mathcal{FP}(\mathsf{D}) \times \mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})$ *satisfying Equation 1.*

Proof: Let the function $\mathsf{bool}\colon \mathcal{FP}(\mathsf{D}) \to \mathbb{B}$ be as in Theorem 4.1.1. We define the operator

$$\mathsf{cond}\colon \mathbb{B} \times \mathcal{FP}(\mathsf{D}) \times \mathcal{FP}(\mathsf{D}) \to \mathcal{FP}(\mathsf{D})$$

by

$$\mathsf{cond}(X, Y, Z) := \big(\pi_0^{\mathsf{D}}(X) \cap (\pi_1^{\mathsf{D}}(X) \cup Y)\big) \cup \big(\pi_1^{\mathsf{D}}(X) \cap (\pi_0^{\mathsf{D}}(X) \cup Y)\big).$$

The continuity of cond follows from the fact that it is the composite of continuous functions. To see one case of Equation 1, observe

$$\begin{aligned} \mathsf{cond}(\mathsf{ff}, Y, Z) &= (\bot \cap (\top \cup Y)) \cup (\top \cap (\bot \cup Z)) \\ &= (\bot \cap \top) \cup (\top \cap Z) \\ &= \bot \cup Z \\ &= Z. \end{aligned}$$

The other cases are equally straightforward. ■

We have found interpretations of the booleans and the condition function in the model. We set $[\![\mathbf{true}]\!]_\rho := \mathsf{tt}$ and $[\![\mathbf{false}]\!]_\rho := \mathsf{ff}$ for all ρ, and

$$[\![\mathbf{cond}]\!]_\rho := \Lambda X.\Lambda Y.\Lambda Z.\mathsf{cond}(\mathsf{bool}(X), Y, Z)$$

for all ρ.

Theorem 4.1.3

$$\begin{aligned} [\![\mathbf{cond}(\mathbf{true})(P)(Q)]\!]_\rho &= [\![P]\!]_\rho, \\ [\![\mathbf{cond}(\mathbf{false})(P)(Q)]\!]_\rho &= [\![Q]\!]_\rho. \end{aligned}$$

Proof: Notice that

$$\begin{aligned} [\![\mathbf{cond}(\mathbf{true})(P)(Q)]\!]_\rho &= [\![\mathbf{cond}]\!]_\rho[[\![\mathbf{true}]\!]_\rho]_{\mathsf{D}}[[\![P]\!]_\rho]_{\mathsf{D}}[[\![Q]\!]_\rho]_{\mathsf{D}} \\ &= (\Lambda X.\Lambda Y.\Lambda Z.\mathsf{cond}(\mathsf{bool}(X), Y, Z))[\mathsf{tt}]_{\mathsf{D}}[[\![P]\!]_\rho]_{\mathsf{D}}[[\![Q]\!]_\rho]_{\mathsf{D}} \\ &= \mathsf{cond}(\mathsf{tt}, [\![P]\!]_\rho, [\![Q]\!]_\rho) \\ &= [\![P]\!]_\rho \end{aligned}$$

for all ρ. The proof for **false** is similar. ■

4.2 List Construction: nil and cons

We wish to find interpretations of **nil** and **cons** that facilitate the construction of the list manipulation functions. We define

$$\llbracket \mathbf{nil} \rrbracket_\rho := [\mathtt{tt}, \top]_{\mathsf{D}}$$

and

$$\llbracket \mathbf{cons} \rrbracket_\rho := \Lambda X.\Lambda L.[\mathtt{ff}, [X, L]_{\mathsf{D}}]_{\mathsf{D}}.$$

The choice of **nil** and **cons** here is crucial in order to get lists that satisfy the applicative property (the γ-rule listed in Section 1.2). We verify that this rule holds.

Theorem 4.2.1 *For all $n \geq 0$ and terms $M, E_1, \ldots, E_n$, the following equation holds:*

$$\llbracket \langle E_1, \ldots, E_n \rangle (M) \rrbracket_\rho = \llbracket \langle E_1(M), \ldots, E_n(M) \rangle \rrbracket_\rho \tag{2}$$

Proof: First note that for all $X \in \mathcal{FP}(\mathsf{D})$,

$$\mathtt{tt}[X]_{\mathsf{D}} = [\top, \bot]_{\mathsf{D}}[X]_{\mathsf{D}} = [\top[X]_{\mathsf{D}}, \bot[X]_{\mathsf{D}}]_{\mathsf{D}} = [\top, \bot]_{\mathsf{D}} = \mathtt{tt},$$

and

$$\mathtt{ff}[X]_{\mathsf{D}} = [\bot, \top]_{\mathsf{D}}[X]_{\mathsf{D}} = [\bot[X]_{\mathsf{D}}, \top[X]_{\mathsf{D}}]_{\mathsf{D}} = [\bot, \top]_{\mathsf{D}} = \mathtt{ff}.$$

We proceed by induction on n. If $n = 0$, then

$$\begin{aligned}
\llbracket \langle E_1, \ldots, E_n \rangle (M) \rrbracket_\rho &= \llbracket \mathbf{nil}(M) \rrbracket_\rho \\
&= \llbracket \mathbf{nil} \rrbracket [\llbracket M \rrbracket_\rho]_{\mathsf{D}} \\
&= [\mathtt{tt}, \top]_{\mathsf{D}} [\llbracket M \rrbracket_\rho]_{\mathsf{D}} \\
&= [\mathtt{tt}[\llbracket M \rrbracket]_{\mathsf{D}}, \top[\llbracket M \rrbracket_\rho]_{\mathsf{D}}]_{\mathsf{D}} \\
&= [\mathtt{tt}, \top]_{\mathsf{D}} \\
&= \llbracket \mathbf{nil} \rrbracket_\rho \\
&= \llbracket \langle E_1(M), \ldots, E_n(M) \rangle \rrbracket_\rho
\end{aligned}$$

for all ρ. Now assume $n \geq 1$. Then

$$\begin{aligned}
\llbracket \langle E_1, \ldots, E_n \rangle (M) \rrbracket_\rho &= \llbracket \mathbf{cons}(E_1)(\langle E_2, \ldots, E_n \rangle)(M) \rrbracket_\rho \\
&= [\mathtt{ff}, [\llbracket E_1 \rrbracket_\rho, \llbracket \langle E_2, \ldots, E_n \rangle \rrbracket_\rho]_{\mathsf{D}}]_{\mathsf{D}} [\llbracket M \rrbracket_\rho]_{\mathsf{D}} \\
&= [\mathtt{ff}[\llbracket M \rrbracket_\rho]_{\mathsf{D}}, [\llbracket E_1 \rrbracket_\rho [\llbracket M \rrbracket_\rho]_{\mathsf{D}}, \llbracket \langle E_2, \ldots, E_n \rangle \rrbracket_\rho [\llbracket M \rrbracket_\rho]_{\mathsf{D}}]_{\mathsf{D}}]_{\mathsf{D}} \\
&= [\mathtt{ff}, [\llbracket E_1(M) \rrbracket_\rho, \llbracket \langle E_2, \ldots, E_n \rangle (M) \rrbracket_\rho]_{\mathsf{D}}]_{\mathsf{D}} \\
&= [\mathtt{ff}, [\llbracket E_1(M) \rrbracket_\rho, \llbracket \langle E_2(M), \ldots, E_n(M) \rangle \rrbracket_\rho]_{\mathsf{D}}]_{\mathsf{D}} \\
&= \llbracket \mathbf{cons}(E_1(M))(\langle E_2(M), \ldots, E_n(M) \rangle) \rrbracket_\rho \\
&= \llbracket \langle E_1(M), \ldots, E_n(M) \rangle \rrbracket_\rho
\end{aligned}$$

for all ρ. ■

Hence, the applicative property is preserved in this model.

4.3 List Manipulation: head, tail, and null

We now finish up the proof by exhibiting the remaining atoms of λAL and verifying their properties.

Definition 4.3.1

$$[\![\mathbf{head}]\!]_\rho := \Lambda L.\mathsf{cond}(\mathsf{bool}(\pi_0^{\mathsf{D}}(L)), [\![\mathbf{nil}]\!]_\rho, \pi_0^{\mathsf{D}}(\pi_1^{\mathsf{D}}(L))),$$

$$[\![\mathbf{tail}]\!]_\rho := \Lambda L.\mathsf{cond}(\mathsf{bool}(\pi_0^{\mathsf{D}}(L)), [\![\mathbf{nil}]\!]_\rho, \pi_1^{\mathsf{D}}(\pi_1^{\mathsf{D}}(L))),$$

$$[\![\mathbf{null}]\!]_\rho := \Lambda L.\mathsf{bool}(\pi_0^{\mathsf{D}}(L)).$$

Theorem 4.3.2 *The following (δ) rules are satisfied:*

$$\begin{array}{ll} [\![\mathbf{head}(\langle\,\rangle)]\!]_\rho = [\![\langle\,\rangle]\!]_\rho, & [\![\mathbf{head}(\langle E_1,\dots,E_n\rangle)]\!]_\rho = [\![E_1]\!]_\rho, \\ [\![\mathbf{tail}(\langle\,\rangle)]\!]_\rho = [\![\langle\,\rangle]\!]_\rho, & [\![\mathbf{tail}(\langle E_1,\dots,E_n\rangle)]\!]_\rho = [\![\langle E_2,\dots,E_n\rangle]\!]_\rho, \\ [\![\mathbf{null}(\langle\,\rangle)]\!]_\rho = [\![\mathbf{true}]\!]_\rho, & [\![\mathbf{null}(\langle E_1,\dots,E_n\rangle)]\!]_\rho = [\![\mathbf{false}]\!]_\rho. \end{array}$$

Proof:

$$\begin{array}{rcl} [\![\mathbf{head}(\mathbf{nil})]\!]_\rho & = & [\![\mathbf{head}]\!]_\rho[[\![\mathbf{nil}]\!]_\rho]_{\mathsf{D}} \\ & = & [\![\mathbf{head}]\!]_\rho[[\mathtt{tt},\top]_{\mathsf{D}}]_{\mathsf{D}} \\ & = & (\Lambda L.\mathsf{cond}(\mathsf{bool}(\pi_0^{\mathsf{D}}(L)), [\![\mathbf{nil}]\!]_\rho, \pi_0^{\mathsf{D}}(\pi_1^{\mathsf{D}}(L))))[[\mathtt{tt},\top]_{\mathsf{D}}]_{\mathsf{D}} \\ & = & \mathsf{cond}(\mathtt{tt}, [\![\mathbf{nil}]\!]_\rho, \pi_0^{\mathsf{D}}(\top)) \\ & = & [\![\mathbf{nil}]\!]_\rho, \end{array}$$

$$\begin{array}{rcl} [\![\mathbf{head}(\mathbf{cons}(A)(L))]\!]_\rho & = & [\![\mathbf{head}]\!]_\rho[[\mathtt{ff},[[\![A]\!]_\rho,[\![L]\!]_\rho]_{\mathsf{D}}]_{\mathsf{D}}]_{\mathsf{D}} \\ & = & [\![A]\!]_\rho; \end{array}$$

$$\begin{array}{rcl} [\![\mathbf{tail}(\mathbf{nil})]\!]_\rho & = & [\![\mathbf{tail}]\!][[\mathtt{tt},\top]_{\mathsf{D}}]_{\mathsf{D}} \\ & = & [\![\mathbf{nil}]\!]_\rho, \end{array}$$

$$\begin{array}{rcl} [\![\mathbf{tail}(\mathbf{cons}(A)(L))]\!]_\rho & = & [\![\mathbf{tail}]\!]_\rho[[\mathtt{ff},[[\![A]\!]_\rho,[\![L]\!]_\rho]_{\mathsf{D}}]_{\mathsf{D}}]_{\mathsf{D}} \\ & = & [\![L]\!]_\rho; \end{array}$$

$$[\![\mathbf{null}(\mathbf{nil})]\!]_\rho = \mathsf{bool}(\pi_0^{\mathsf{D}}(\mathbf{nil})) = \mathsf{bool}(\mathtt{tt}) = \mathtt{tt} = [\![\mathbf{true}]\!]_\rho,$$

$$[\![\mathbf{null}(\mathbf{cons}(A)(L))]\!]_\rho = \mathsf{bool}(\pi_0^{\mathsf{D}}([\mathtt{ff},[[\![A]\!]_\rho,[\![L]\!]_\rho]_{\mathsf{D}}]_{\mathsf{D}})) = \mathsf{bool}(\mathtt{ff}) = \mathtt{ff} = [\![\mathbf{false}]\!]_\rho$$

for all ρ. ∎

4.4 Modeling λAL

Theorem 4.4.1 *$(\mathcal{FP}(\mathsf{D}), [\![\cdot]\!]_\rho)$ with the interpretations of* **head**, **tail**, **cons**, **nil**, **null**, **true**, **false**, *and* **cond** *defined as above yields a model for $\lambda AL + (\eta)$.*

Proof: That $(\mathcal{FP}(\mathsf{D}), [\![\cdot]\!]_\rho)$ satisfies (R), (S), (T), (C), (ξ), (α), (β), and (η) follows from Theorem 3.3.2. The (γ) rule follows from Theorem 4.2.1 and the (δ) rule is shown in Theorems 4.1.3 and 4.3.2. ∎

5 Conclusions and Future Research

We have accomplished the main goal of finding a model (a class of models, in fact) for λAL. Along the way we discussed models for $\lambda\beta$ and λEP, which are interesting in their own right.

The freedom of choice of A when building up D allows a rich family of types to be embedded in these models. It would be interesting to explore semantic models that take advantage of this feature to model more complicated languages. Also, the construction of these models embues them with specific properties that may be worth studying; for instance, the fact that $\mathcal{FP}(D) = \mathcal{FP}(D \multimap C)$ implies that the models for λEP and λAL are extensional. The model for λEP is interesting in that it satisfies the *surjective pairing* property, $[\pi_0^{\mathsf{D}}(F), \pi_1^{\mathsf{D}}(F)]_{\mathsf{D}} = F$, which is not required by the axioms for convertability in λEP.

The construction of lists for λAL based on the pairing of λEP can be mimicked for a wide variety of types, such as trees. Also, it might be worth considering models similar to this one for "folding" lists instead of applicative lists.

There remain several open questions. For example, is there an encoding of pairing in pure-lambda calculus terms that has projections and the applicative property? Note that [Klo80] shows that there is no encoding for *surjective* pairing in pure lambda calculus terms. If there is an encoding for pairing without the (SP)-rule, could encodings for the atoms of λAL be found so that the remaining (δ)-rules are satisfied?

6 Acknowledgements

I would like to thank everyone who assisted me in this effort, including Richard Statman, Peter Lee, Andrej Bauer, and Adam Klivans. I would also like to thank György Révész for his assistance in my research and for the introduction of the language with which this work is concerned. Most of all I would like to thank Dana Scott, my advisor for the past year, who is responsible for not only many of the results needed in this thesis but for the entire academic field in which this thesis rests.

References

[Bac78] J. W. Backus. Can programming be liberated from the von Neumann style? A functional style and its algebra of programs. *Communications of the ACM*, 21(8):613–641, August 1978.

[Bar84] H. P. Barendregt. *The Lambda Calculus.* North-Holland, New York, 1984.

[Cur93] P. Curien. *Categorical Combinators, Sequential Algorithms, and Functional Programming.* Birkhäuser, 1993.

[Klo80] J. W. Klop. *Combinatory Reduction Systems.* PhD thesis, Mathematisch Centrum, Amsterdam, 1980.

[Rév88] G. Révész. *Lambda-Calculus Combinators and Functional Programmming*, volume 4 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, Cambridge, 1988.

[Rév92] G. Révész. A list-oriented extension of the lambda-calculus satisfying the Church-Rosser theorem. *Theoretical Computer Science*, 93:75–89, 1992.

[Rév95] G. Révész. Categorical combinators with explicit products. *Fundamenta Informaticae*, 22:153–166, 1995.

[Sco96] D. Scott. A Theory of Domains. Carnegie Mellon University, 1996.

[SHLG94] V. Stoltenberg-Hansen, I. Lindström, and E. R. Griffor. *Mathematical Theory of Domains*, volume 22 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, 1994.