January 1992

Statman's 1-Section Theorem

Jon G. Riecke
University of Pennsylvania

Recommended Citation

Jon G. Riecke, "Statman's 1-Section Theorem", . January 1992.

University of Pennsylvania Department of Computer and Information Science Technical Report No.MS-CIS-92-03.

Statman's 1-Section Theorem

Abstract

Statman's 1-Section Theorem [17] is an important but little-known result in the model theory of the simply-typed λ-calculus. The λ-Section Theorem states a necessary and sufficient condition on models of the simply-typed λ-calculus for determining whether βη-equational reasoning is *complete* for proving equations that hold in a model. We review the statement of the theorem, give a detailed proof, and discuss its significance.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No.MS-CIS-92-03.

Statmans's 1-Section Theorem

MS-CIS-92-03
LOGIC & COMPUTATION 46

Jon G. Riecke

University of Pennsylvania
School of Engineering and Applied Science
Computer and Information Science Department
Philadelphia, PA 19104-6389

January 1992

Statman's 1-Section Theorem

Jon G. Riecke*
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104

January 17, 1992

Abstract

Statman's 1-Section Theorem [17] is an important but little-known result in the model theory of the simply-typed λ-calculus. The 1-Section Theorem states a necessary and sufficient condition on models of the simply-typed λ-calculus for determining whether $\beta\eta$-equational reasoning is *complete* for proving equations that hold in a model. We review the statement of the theorem, give a detailed proof, and discuss its significance.

1 Introduction

The theory of the simply-typed λ-calculus forms the foundation of call-by-name functional languages. The simply-typed λ-calculus comes equipped with an equational theory—the (β) and (η) axioms together with the usual rules of equality—and an independently-characterizable class of models. The equations (β) and (η) are **sound** for proving facts in models, *viz.*, an equation derivable from the axioms is valid in all models. A more general fact encompassing soundness is **completeness**: an equation between simply-typed λ-terms is provable via $\beta\eta$-reasoning iff the equation holds in all models. An arbitrary model of the simply-typed λ-calculus may, of course, satisfy more equations than those provable from (β) and (η). Here we shall discuss a simple necessary and sufficient criterion for determining whether the equational theory of a *single* model, or more generally a class of models, is captured completely by $\beta\eta$-equality. The criterion, due to Richard Statman, is crystallized in the *1-Section Theorem.* Part of our purpose here will be to state the theorem and present a rigorous proof: although it is cited in [17] and follows from results in [16], a complete proof has never appeared in the literature. Statman's criterion may be easily applied to a host of models, and we will demonstrate its applicability to show the completeness of $\beta\eta$-reasoning for some of the more familiar models of the simply-typed λ-calculus.

It is not hard to find a model that satisfies exactly the equations provable from (β) and (η). One can construct such a model $\mathcal{B}$ out of $\beta\eta$-equivalence classes of open terms; the (β) and (η) axioms are crucial in verifying that $\mathcal{B}$ is a model. Note that the non-soundness direction of the completeness theorem above follows as a corollary: if an equation is *not* provable from (β) and (η), there is a model, namely $\mathcal{B}$, that denies the equation. But this "term model" has little independent

*This research was partially supported by an ONR grant number N00014–88–K–0557, NSF grant number CCR–8912778, NRL grant number N00014–91–J–2022, and NOSC grant number 19–920123–31.

interest beyond proving the completeness theorem. Another model, definable without reference to the (β) and (η) axioms, is the full set-theoretic model $\mathcal{S}$ over $\mathbf{N}$, defined

$$\begin{array}{rcl} \mathcal{S}^{\iota} & = & \mathbf{N} \\ \mathcal{S}^{\tau\to\nu} & = & [\mathcal{S}^{\tau} \to \mathcal{S}^{\nu}] \end{array}$$

where $[A \to B]$ is the set of all total functions from A to B. From a mathematical point of view, the model $\mathcal{S}$ is important precisely because it contains all the higher-order functions one may encounter in mathematical statements and proofs. Of course, not all the functions in $\mathcal{S}$ are definable in the simply-typed λ-calculus, since the set of simply-typed λ-terms is countable and the set of elements in $\mathcal{S}$ is uncountable. For those functions that *are* definable, $\beta\eta$-equality is complete:

Theorem 1.1 (Friedman [4]) *$M =_{\beta\eta} N$ iff $\mathcal{S} \models M = N$.*

Thus, if we wish to prove a fact about λ-definable functions in $\mathcal{S}$, we may substitute $\beta\eta$-reasoning (which is decidable, *cf.* [1]) for denotational reasoning.

$\mathcal{S}$ is but one example of a model for which $\beta\eta$-reasoning is complete; standard denotational models of functional languages provide more examples. In denotational models, we usually use *posets* instead of sets and *continuous* functions rather than set-theoretic functions. Continuity allows an easy interpretation of recursion present in most programming languages [5]. The model $\mathcal{N}$ built out of all continuous functions over the base Scott domain $\mathbf{N}_{\perp}$ is a familiar example of a denotational model for the language PCF [9, 15]. More formally,

$$\begin{array}{rcl} \mathcal{N}^{\iota} & = & \mathbf{N}_{\perp} \\ \mathcal{N}^{\tau\to\nu} & = & [\mathcal{N}^{\tau} \to_c \mathcal{N}^{\nu}] \end{array}$$

where $\mathbf{N}_{\perp}$ is the poset of natural numbers ordered discretely with an element $\perp$ ordered below every element of $\mathbf{N}$, and $[\mathcal{N}^{\tau} \to_c \mathcal{N}^{\nu}]$ is the Scott domain of continuous functions from $\mathcal{N}^{\tau}$ to $\mathcal{N}^{\nu}$ ordered pointwise [5]. Then

Theorem 1.2 (Plotkin [10]) *$M =_{\beta\eta} N$ iff $\mathcal{N} \models M = N$.*

Classes of models can also be complete for $\beta\eta$-equational reasoning:

Theorem 1.3 (Plotkin [8]) *$M =_{\beta\eta} N$ iff in all models $\mathcal{M}$ with finite base type, $\mathcal{M} \models M = N$.*

In particular, showing that $M = N$ holds in all models with finite base type can be established by showing that $M =_{\beta\eta} N$.

The original proofs of Theorems 1.1 and 1.2, and 1.3 proceed quite differently: the proofs of the first two construct *logical relations* between the term model and the model in question, while the proof of the third relies on certain combinatorial facts about λ-terms. In fact, the combinatorial arguments may be adapted to prove Theorems 1.1 and 1.2. (The logical relation argument has important uses in other contexts, *cf.* [11, 12].) The combinatorial argument is essentially captured by Statman's 1-Section Theorem, which states that if a certain algebra can be faithfully embedded in the first-order part of a class of models—in a sense to be made more precise in Section 3—then $\beta\eta$-equational reasoning is complete for proving all equations in the class of models. Section 4 describes a detailed proof of the 1-Section Theorem, showing how the combinatorial structure of the required embedded algebra can be used to deduce completeness.

Variables	$x_i^\sigma : \sigma$, where $i \in \mathbb{N}$	Constants	$c^\sigma : \sigma$, where $c \in \Sigma$
λ-abstraction	$\dfrac{M : \nu}{(\lambda x_i^\tau . M) : (\tau \rightarrow \nu)}$	Application	$\dfrac{M : (\tau \rightarrow \nu) \qquad N : \tau}{(M\ N) : \nu}$

Table 1: Syntactic formation rules of the simply-typed λ-calculus.

The importance of the 1-Section Theorem lies not in its statement but in its applicability. In Section 5, we show how the 1-Section Theorem can be used to prove Theorems 1.1, 1.2, and 1.3. We also show how these, and other similar theorems, can be used in reasoning about functional programming languages. Finally, Section 6 concludes the paper with a discussion of some open problems.

2 Review of the simply-typed λ-calculus

We first briefly review the syntax, semantics, and equational theory of the simply-typed λ-calculus. The reader familiar with the simply-typed λ-calculus may care to skim this section in order to understand the notation we use.

2.1 Syntax

Each term in the simply-typed λ-calculus comes with a *simple type*. Simple types are defined inductively to be the base type ι, usually taken to be the type of natural numbers, and $(\tau \rightarrow \nu)$, the type of functions from τ to ν, where τ and ν themselves are types. For readability, we often drop parentheses from types with the understanding that $\rightarrow$ associates to the right, *e.g.*, $(\iota \rightarrow (\iota \rightarrow \iota))$ is abbreviated $(\iota \rightarrow \iota \rightarrow \iota)$. This convention implies that any simple type σ can be written uniquely in the form $(\sigma_1 \rightarrow \sigma_2 \rightarrow \ldots \sigma_n \rightarrow \iota)$ for some $n \geq 0$.

The set of simply-typed terms is parameterized by a **signature**, which is just a set of typed constants. The set of simply-typed terms over the signature Σ, together with their types, is defined in Table 1. We adopt many of the standard notational conventions of the λ-calculus from [1]. For instance, the usual definitions of free and bound variables are used and $FV(M)$ denotes the set of free variables of M. Terms are identified up to renaming of bound variables, and are denoted by the letters M, N, P, Q, S, and T. Parentheses may be dropped from applications under the assumption that application associates to the left, *i.e.*, $(M\ N\ P)$ is short for $((M\ N)\ P)$. We will also drop types from variables whenever the types are unimportant or can be deduced from the context, and use the letters u, v, w, x, y, and z to denote variables. Finally, syntactic substitution is written $M[x := N]$, where the substitution renames the bound variables of M to avoid capturing the free variables of N.

2.2 Semantics via environment models

Although there are other equivalent definitions of models, here we assign meaning to terms using environment models [4, 6, 7]. Environment models have two components, the first of which is a type frame:

Definition 2.1 A **type frame** is a tuple

$$(\{\mathcal{M}^\sigma : \sigma \text{ a type}\}, \{\mathsf{Ap}^{\tau,\nu} : \tau, \nu \text{ types}\}),$$

where each $\mathcal{M}^\sigma$ is a nonempty set and $\mathsf{Ap}^{\tau,\nu} : \mathcal{M}^{\tau\to\nu} \times \mathcal{M}^\tau \to \mathcal{M}^\nu$. The components of a type frame must also obey the **extensionality property**: for any $f, g \in \mathcal{M}^{\tau\to\nu}$, $f = g$ iff for all d, $\mathsf{Ap}^{\tau,\nu}(f,d) = \mathsf{Ap}^{\tau,\nu}(g,d)$.

Intuitively, $\mathsf{Ap}^{\tau,\nu}$ is an abstract "application" function for applying elements in the set $\mathcal{M}^{\tau\to\nu}$ to elements in the set $\mathcal{M}^\tau$. The extensionality property states that the set $\mathcal{M}^{\tau\to\nu}$ can be regarded as a set of total functions from $\mathcal{M}^\tau$ to $\mathcal{M}^\nu$; the most familiar type frames are constructed out of total functions.

The second component of an environment model is a meaning function $\mathcal{M}[\![\cdot]\!]$ that assigns elements of a type frame to terms. Since there is no way to assign a meaning to an open term *a priori*, an **environment** is used to assign meaning to free variables.

Definition 2.2 Let $\mathcal{M}$ be a type frame. An $\mathcal{M}$**-environment** ρ is a map from variables to elements of $\mathcal{M}$ that respects types, *i.e.*, $\rho(x^\sigma) \in \mathcal{M}^\sigma$.

We use the notation $\rho[x^\tau \mapsto d]$ for a new environment that maps x^τ to d and every other variable y to $\rho(y)$.

Definition 2.3 An **environment model** over a signature Σ is a type frame $\mathcal{M}$ with a meaning function $\mathcal{M}[\![\cdot]\!]$ defined inductively on the structure of terms as follows:

$$\begin{array}{rcl}
\mathcal{M}[\![x^\sigma]\!]\rho &=& \rho(x^\sigma) \\
\mathcal{M}[\![c^\sigma]\!]\rho &=& \mathcal{I}(c^\sigma) \\
\mathcal{M}[\![(M\ N)]\!]\rho &=& \mathsf{Ap}(\mathcal{M}[\![M]\!]\rho, \mathcal{M}[\![N]\!]\rho) \\
\mathcal{M}[\![\lambda x^\tau.\, M]\!]\rho &=& f, \quad \text{where } \mathsf{Ap}(f,d) = \mathcal{M}[\![M]\!]\rho[x^\tau \mapsto d]
\end{array}$$

where $\mathcal{I} : \Sigma \to \mathcal{M}$ is a constant interpretation function that respects types, *i.e.*, for all $c^\sigma \in \Sigma$, $\mathcal{I}(c^\sigma) \in \mathcal{M}^\sigma$.

Equations are interpreted in the obvious way in environment models: for any environment model $\mathcal{M}$, we write $\mathcal{M} \models M = N$ iff for all environments ρ, $\mathcal{M}[\![M]\!]\rho = \mathcal{M}[\![N]\!]\rho$.

Not every type frame is an environment model, since the definition requires the *existence* of an appropriate meaning for each λ-abstraction. Some standard examples of environment models were given in the introduction. Another example is the type frame consisting of all set-theoretic functions over a base set X, defined by

$$\begin{array}{rcl}
\mathcal{X}^\iota &=& X \\
\mathcal{X}^{\tau\to\nu} &=& [\mathcal{X}^\tau \to \mathcal{X}^\nu] \\
\mathsf{Ap}(f,d) &=& f(d).
\end{array}$$

There are other ways of defining environment models other than by explicit constructions. For example, we may construct a direct product out of a class of models.

Definition 2.4 Let $\{\mathcal{M}_0, \mathcal{M}_1, \mathcal{M}_2, \ldots\}$ be a countable (possibly finite) class of models over a signature Σ. Then the **direct product of** $\{\mathcal{M}_i\}$ is a tuple $(\{\mathcal{M}^\sigma\}, \{\mathsf{Ap}^{\sigma,\tau}\})$ with

$$\begin{array}{rcl} \mathcal{M}^\sigma & = & \{\langle a_0, a_1, a_2, \ldots\rangle : a_i \in \mathcal{M}_i^\sigma\} \\ \mathsf{Ap}(\langle f_0, f_1, \ldots\rangle, \langle a_0, a_1, \ldots\rangle) & = & \langle \mathsf{Ap}_0(f_0, a_0), \mathsf{Ap}_1(f_1, a_1), \ldots\rangle \end{array}$$

We could, of course, generalize the definition to uncountable direct products, but we will only need countable direct products here. It is important to note that this construction always yields a model.

Proposition 2.5 *If $\mathcal{M}$ is the countable direct product of a class of models $\{\mathcal{M}_0, \mathcal{M}_1, \mathcal{M}_2, \ldots\}$ over a signature Σ, then $\mathcal{M}$ is a model over the same signature in which*

$$\mathcal{M} \models M = N \quad \textit{iff for all } i,\ \mathcal{M}_i \models M = N.$$

Proof: First we need to verify that $\mathcal{M}$ is indeed a type frame, and so we need to show that application in the structure is extensional. To that end, consider any $f = \langle f_0, f_1, f_2, \ldots\rangle$ and $g = \langle g_0, g_1, g_2, \ldots\rangle$ in the set $\mathcal{M}^{\tau\to\nu}$. If $f = g$, then it is easy to see that $\mathsf{Ap}(f,d) = \mathsf{Ap}(g,d)$ for any $d \in \mathcal{M}^\tau$. If $f \neq g$, then for some i, $f_i \neq g_i$. Since $\mathcal{M}_i$ is a type frame, there exists an element $d_i \in \mathcal{M}_i^\tau$ such that $\mathsf{Ap}_i(f_i, d_i) \neq \mathsf{Ap}_i(g_i, d_i)$. For all $j \neq i$, pick any $d_j \in \mathcal{M}_j^\tau$, and let $d = \langle d_0, d_1, \ldots, d_{i-1}, d_i, d_{i+1}, \ldots\rangle$. Then

$$\begin{array}{rcl} \mathsf{Ap}(f,d) & = & \langle \mathsf{Ap}_0(f_0, d_0), \mathsf{Ap}_1(f_1, d_1), \ldots\rangle \\ & \neq & \langle \mathsf{Ap}_0(g_0, d_0), \mathsf{Ap}_1(g_1, d_1), \ldots\rangle \;=\; \mathsf{Ap}(g,d). \end{array}$$

Therefore, $\mathcal{M}$ obeys the extensionality property.

To see that $\mathcal{M}$ is a model, define

$$\mathcal{M}[\![M]\!]\rho = \langle \mathcal{M}_0[\![M]\!]\rho_0, \mathcal{M}_1[\![M]\!]\rho_1, \mathcal{M}_2[\![M]\!]\rho_2, \ldots\rangle$$

where $\rho_i(x) = d_i$ if $\rho(x) = \langle d_0, d_1, d_2, \ldots\rangle$. We claim that this matches the inductive definition of the meaning function in Definition 2.2; the proof is a straightforward induction on terms. Finally, we must show that $\mathcal{M} \models M = N$ iff for all i, $\mathcal{M}_i \models M = N$, which follows easily from the definition of $\models$. ■

2.3 $\beta\eta$-equational theory

Reasoning about equalities of λ-terms can also be done purely syntactically. The equational theory of the simply-typed λ-calculus appears in Table 2. All equational theories include the axiom (*refl*) and the rules (*symm*), which axiomatize $=$ as an equivalence relation; the rules (*cong*) and (ξ) similarly allow substitution of equals for equals. The only other axioms of the theory are (β) and (η), which can be justified by examining the intended class of models defined above. We write $M =_{\beta\eta} N$ when M and N are provably equivalent using the axioms and rules of Table 2.

The equational axioms of (β) and (η) may be directed into a rewrite system. Table 2 also defines the rewriting relation $\to_{\beta\eta}$. We write $M \twoheadrightarrow_{\beta\eta} N$ if M rewrites to N in 0 or more steps, and say that a term M is in **normal form** if $M \not\to_{\beta\eta} N$ for any N [1]. An important fact about normal forms is summarized by the following proposition:

Proposition 2.6 *Suppose M and N are terms of the same type in $\beta\eta$-normal form, and $M \neq N$. Then $M \neq_{\beta\eta} N$.*

Equational System	(β) $((\lambda x.\, M)\, N) = M[x := N]$ (η) $(\lambda x.\, M\ x) = M$, if $x \notin FV(M)$ (*refl*) $M = M$ (*symm*) $\dfrac{M = N}{N = M}$ (*trans*) $\dfrac{M = N \qquad N = P}{M = P}$ (*cong*) $\dfrac{M = N \qquad P = Q}{(M\ P) = (N\ Q)}$ (ξ) $\dfrac{M = N}{(\lambda x.\, M) = (\lambda x.\, N)}$
Rewrite System	$((\lambda x.\, M)\, N) \to_{\beta\eta} M[x := N]$ $(\lambda x.\, M\ x) \to_{\beta\eta} M$, if $x \notin FV(M)$ $\dfrac{M \to_{\beta\eta} N}{(M\ P) \to_{\beta\eta} (N\ P)}$ $\dfrac{M \to_{\beta\eta} N}{(P\ M) \to_{\beta\eta} (P\ N)}$ $\dfrac{M \to_{\beta\eta} N}{(\lambda x.\, M) \to_{\beta\eta} (\lambda x.\, N)}$

Table 2: Equational and rewrite systems of the simply-typed λ-calculus. Each term appearing in these rules must be a well-formed term of the simply-typed λ-calculus.

The proposition follows easily from the Church-Rosser Theorem for the simply-typed λ-calculus [1, 7, 18].

For the proofs in this paper, we will use **extended $\beta\eta$-normal forms** instead of $\beta\eta$-normal forms.

Definition 2.7 A term M of type $(\sigma_1 \to \ldots \to \sigma_n \to \iota)$ is in **extended $\beta\eta$-normal form** if M has the form

$$\lambda x_1^{\sigma_1}. \ldots \lambda x_n^{\sigma_n}. u\,(M_1\, x_1 \ldots x_n) \ldots (M_k\, x_1 \ldots x_n)$$

where $k, n \geq 0$, u is either a variable or constant, and each M_i is in extended $\beta\eta$-normal form.

Extended $\beta\eta$-normal forms are called Φ-normal forms in [16]. *Closed* extended $\beta\eta$-normal forms are easier to induct upon than $\beta\eta$-normal forms, because the constituent terms $M_1, \ldots, M_k$ are also closed extended $\beta\eta$-normal forms. It is also appropriate to call these terms "normal forms" due to the following proposition:

Proposition 2.8 *For any M, there exists a unique extended $\beta\eta$-normal form N such that $M =_{\beta\eta} N$.*

Proof: By the Strong-Normalization Theorem for $\to_{\beta\eta}$ in the simply-typed λ-calculus, there exists a term M' in $\beta\eta$-normal form such that $M \twoheadrightarrow_{\beta\eta} M'$. Note that M' must have the form $\lambda x_1. \ldots \lambda x_m. u\; M_1 \ldots M_l$, where each M_j is in $\beta\eta$-normal form and u is a variable or constant. If M has type $(\sigma_1 \to \ldots \to \sigma_m \to \ldots \to \sigma_n \to \iota)$, first let

$$M'' = \lambda x_1. \ldots \lambda x_n. u\; M_1 \ldots\; M_l\; x_{m+1} \ldots x_n.$$

Each $M_1, \ldots, M_l, x_{m+1}, \ldots, x_n$ may be turned into a closed term by λ-abstracting over all the variables $x_1, \ldots, x_n$, resulting in the term

$$M''' = \lambda x_1. \ldots \lambda x_n. u\,((\lambda \vec{x}. M_1)\, \vec{x}) \ldots\; ((\lambda \vec{x}. M_l)\, \vec{x}) \ldots\; ((\lambda \vec{x}. x_{m+1})\, \vec{x}) \ldots\; ((\lambda \vec{x}. x_n)\, \vec{x}).$$

where $M =_{\beta\eta} M'''$. The terms $(\lambda \vec{x}. M_i)$ and $(\lambda \vec{x}. x_j)$ may be then be turned into extended $\beta\eta$-normal forms recursively; the process eventually terminates at a extended $\beta\eta$-normal form that is $\beta\eta$-equivalent to M.

To prove uniqueness, suppose M and N are in extended $\beta\eta$-normal form and $M =_{\beta\eta} N$; we will show that $M = N$ by induction on M. In the basis, $M = \lambda x_1. \ldots \lambda x_n. u$ where u is a variable or constant of type ι. Now because N is in extended $\beta\eta$-normal form,

$$N = \lambda x_1. \ldots \lambda x_n. v\,(N_1\, x_1 \ldots x_n) \ldots (N_k\, x_1 \ldots x_n).$$

But note that M and N must have the same $\beta\eta$-normal form by Proposition 2.6; thus, $v = u$ and hence since u is of base type, $k = 0$. Thus, $M = N$. In the induction case, suppose

$$M = \lambda x_1. \ldots \lambda x_n. u\,(M_1\, x_1 \ldots x_n) \ldots (M_k\, x_1 \ldots x_n)$$

where each M_j is in extended $\beta\eta$-normal form. Since M and N have the same $\beta\eta$-normal form by Proposition 2.6, N must have the form $\lambda x_1. \ldots \lambda x_n. u\,(N_1\, x_1 \ldots x_n) \ldots (N_k\, x_1 \ldots x_n)$ where $M_j =_{\beta\eta} N_j$. Since M_j and N_j are in extended $\beta\eta$-normal form, by induction $M_j = N_j$. Thus, $M = N$ as desired. ■

3 Statement of the Theorem

Suppose two closed terms M and N are not equivalent in a model $\mathcal{M}$; then by the extensionality property, one may find arguments in the model driving $\mathcal{M}[\![M]\!]$ and $\mathcal{M}[\![N]\!]$ to different base type elements in the model. Thus, in order for $\beta\eta$-reasoning to be complete for a model, there must be "enough elements" in the model to distinguish all terms that are not $\beta\eta$-equivalent. Informally, the 1-Section Theorem states a condition on the combinatorial structure of a model $\mathcal{M}$ that guarantees that $\mathcal{M}$ has enough elements. This condition is stated in the form of whether a certain algebra can be faithfully embedded in $\mathcal{M}$.

Recall that an **algebra** $\mathcal{A} = (A, f_0, f_1, \ldots)$ over an algebraic signature $\{F_0, F_1, \ldots\}$ is a tuple comprised of a carrier set A together with functions

$$f_i : \underbrace{A \to \ldots \to A}_{n_i} \to A$$

where $n_i \geq 0$ is the arity of F_i. Here we have taken the liberty of *currying* the function symbols in anticipation of incorporating them into the λ-calculus. One familiar example is algebra $(\mathbf{N}, \mathbf{0}, +)$ over the signature $\{0, +\}$, where $\mathbf{0}$ is the number zero and $+$ is the curried addition function. An **algebraic equation** is an equation involving algebraic terms with variables, and an algebra **satisfies** an algebraic equation $t_1 = t_2$, written $\mathcal{A} \models t_1 = t_2$, if for any instantiation of the variables by elements of the carrier of $\mathcal{A}$, the equation holds in $\mathcal{A}$. For instance, $(\mathbf{N}, \mathbf{0}, +) \models (+\ x\ y) = (+\ y\ x)$.

Given a model of the simply-typed λ-calculus, we may extend it to model of an algebraic signature. A model $\mathcal{M}'$ is an **extension** of $\mathcal{M}$ if $\mathcal{M}$ and $\mathcal{M}'$ are based on the same type frame, and $\mathcal{M}'$ extends the interpretation $\mathcal{I}$ of constants in $\mathcal{M}$ to a new interpretation $\mathcal{I}' \supseteq \mathcal{I}$. When an extension preserves the equalities of an algebra, the algebra is faithfully embedded in the model.

Definition 3.1 Let $\mathcal{A}$ be an algebra over the function symbols F_i of arity $n_i \geq 0$. Suppose $\mathcal{M}$ is an environment model. Then $\mathcal{A}$ **is faithfully embedded in** $\mathcal{M}$ if there exists an extension $\mathcal{M}'$ of $\mathcal{M}$ such that $\mathcal{M}'$ gives meaning to all the function symbols F_i, and for any algebraic terms t_1 and t_2 (possibly involving variables), $\mathcal{M}' \models t_1 = t_2$ iff $\mathcal{A} \models t_1 = t_2$.

We are now ready to give the statement of the 1-Section Theorem. Let $\mathcal{T}$ be the free closed term algebra on a single binary constant F and a single nullary constant C. In computer science terminology, the carrier set of $\mathcal{T}$ can be described by the context-free grammar

$$T ::= C \mid (F\ T\ T)$$

where $\mathcal{T} \models T_0 = T_1$ iff T_0 and T_1 are syntactically equivalent. The name $\mathcal{T}$ stands for "tree algebra", since the elements in the algebra denote binary trees.

Statman's 1-Section Theorem 3.2 *Let $\mathcal{C}$ be a class of models over the empty signature. Then $\beta\eta$-equality completely axiomatizes the valid equations of $\mathcal{C}$ iff $\mathcal{T}$ can be faithfully embedded in some countable direct product of models in $\mathcal{C}$.*

The name of the 1-Section Theorem comes from the fact that an algebra is embeddable in the first-order part—the 1-section—of a model.

4 Proof of the Theorem

One direction of the 1-Section Theorem follows fairly straightforwardly from the fact that there are only a countable number of equations at any given type.

Proof of Theorem 3.2, (⇒): Suppose $=_{\beta\eta}$ is complete for the valid equations in the class of models $\mathcal{C}$ (over the empty signature). Let $E_0, E_1, E_2, \ldots$ be an enumeration of equations of the form

$$\lambda g^{\iota\to\iota\to\iota}.\, \lambda x^{\iota}.\, t_i = \lambda g^{\iota\to\iota\to\iota}.\, \lambda x^{\iota}.\, t'_i$$

where t_i and t'_i are syntactically different terms in the grammar

$$t ::= x \mid (g\ t\ t).$$

Note that the terms in each equation E_i are in $\beta\eta$-normal form, and hence by Proposition 2.6 each equation E_i is *not* provable by $\beta\eta$-equational reasoning. Since $=_{\beta\eta}$ is complete for $\mathcal{C}$, for every equation E_i there exists a model $\mathcal{A}_i \in \mathcal{C}$ such that $\mathcal{M}_i \not\models E_i$. In particular, there exist $f_i \in \mathcal{M}_i^{\iota\to\iota\to\iota}$ and $c_i \in \mathcal{M}_i^{\iota}$ such that

$$\mathsf{Ap}_i\,(\mathsf{Ap}_i\,(\mathcal{M}_i[\![\lambda g^{\iota\to\iota\to\iota}.\, \lambda x^{\iota}.\, t_i]\!], f_i), c_i) \neq \mathsf{Ap}_i\,(\mathsf{Ap}_i\,(\mathcal{M}_i[\![\lambda g^{\iota\to\iota\to\iota}.\, \lambda x^{\iota}.\, t'_i]\!], f_i), c_i).$$

Define $\mathcal{M}$ to be the countable direct product of the models $A_0, A_1, A_2, \ldots$ and set

$$f = \langle f_0, f_1, f_2, \ldots\rangle \quad\text{and}\quad c = \langle c_0, c_1, c_2, \ldots\rangle.$$

It is not hard to check that $\mathcal{M} \not\models E_i$ for every $i \geq 0$. Let $\mathcal{M}'$ be the extension of $\mathcal{M}$ such that $\mathcal{M}'[\![F^{\iota\to\iota\to\iota}]\!] = f$ and $\mathcal{A}'[\![C^{\iota}]\!] = c$. By the construction, for every pair of distinct terms T_0 and T_1 in the grammar T above, $\mathcal{M}'[\![T_0]\!] \neq \mathcal{A}'[\![T_1]\!]$. Thus, the algebra $\mathcal{T}$ is faithfully embedded in some countable direct product (namely $\mathcal{M}$) of the elements of $\mathcal{C}$ as desired. ■

The proof of the (⇐) direction of Theorem 3.2 is more difficult but more interesting. Given a countable direct product $\mathcal{M}$ of models into which the tree algebra $\mathcal{T}$ can be faithfully embedded and an equation $M = N$ which is *not* provable by $\beta\eta$-reasoning, we wish to show that the model $\mathcal{M}$ denies the equation $M = N$. The essential idea is to show that M and N can be transformed into closed terms of type $((\iota\to\iota\to\iota)\to\iota\to\iota)$ that are not $\beta\eta$-equivalent:

Lemma 4.1 *Suppose M and N are closed terms over the empty signature of type σ and $M \neq_{\beta\eta} N$. Then there exists a closed term P (over the empty signature) of type $(\sigma\to(\iota\to\iota\to\iota)\to\iota\to\iota)$ such that $(P\ M) \neq_{\beta\eta} (P\ N)$.*

The main combinatorial arguments lie in the proof of this lemma, which we shall explicate shortly. The interesting direction of the 1-Section Theorem is then relatively easy to deduce from Lemma 4.1.

Proof of Theorem 3.2, (⇐): Suppose the tree algebra $\mathcal{T}$ can be faithfully embedded in the countable direct product $\mathcal{M}$, and suppose M and N are terms of type σ such that $M \neq_{\beta\eta} N$. Let $\{x_0, x_1, \ldots, x_n\}$ be the set of free variables appearing in M and N, and let $M' = (\lambda x_0.\, \ldots \lambda x_n.\, M)$ and similarly let $N' = (\lambda x_0.\, \ldots \lambda x_n.\, N)$. We will show that $\mathcal{M} \not\models M' = N'$.

By Lemma 4.1, choose a closed term P of type $(\sigma \to (\iota \to \iota \to \iota) \to \iota \to \iota)$ such that $(P\ M') \neq_{\beta\eta} (P\ N')$. Let M'' be the $\beta\eta$-normal form of $(P\ M')$, and similarly let N'' be the $\beta\eta$-normal form of $(P\ N')$. Since M'' and N'' are closed,

$$\begin{aligned} M'' &= \lambda g^{\iota\to\iota\to\iota}.\,\lambda x^{\iota}.\,M_0 \\ N'' &= \lambda g^{\iota\to\iota\to\iota}.\,\lambda x^{\iota}.\,N_0 \end{aligned}$$

where M_0 and N_0 are $\beta\eta$-normal forms of base type involving only the free variables g and x. Since $M'' \neq_{\beta\eta} N''$, it must be the case that $M_0 \not\equiv N_0$. We claim that M_0 and N_0 are terms in the syntax t above. The proof of the claim is a little induction on the size of the term M_0 (and similarly for N_0). In the basis, M_0 has size 1 and hence M_0 must be x. In the inductive case, M_0 has size greater than 1 and must therefore be an application $(P_0\ P_1 \ldots P_l)$, where P_0 is a λ-abstraction or variable; it cannot be a λ-abstraction because M_0 has type ι. Since M_0 is in $\beta\eta$-normal form, P_0 must be a variable, and since the only free variables in M_0 are g and x, $M_0 = (g\ M_0'\ M_0'')$. By induction, M_0' and M_0'' must be in the syntax t, so M_0 is.

Let $f \in \mathcal{M}^{\iota\to\iota\to\iota}$ and $c \in \mathcal{M}^{\iota}$ be the elements in $\mathcal{M}$ that we use to embed the algebra $\mathcal{T}$. Then $\mathsf{Ap}(\mathsf{Ap}(\mathcal{M}[\![M'']\!], f), c)$ and $\mathsf{Ap}(\mathsf{Ap}(\mathcal{M}[\![N'']\!], f), c)$ must be different elements in the model, since M_0 and N_0 are different terms in the grammar t. Thus, $\mathcal{M} \not\models M'' = N''$, from whence it follows that $\mathcal{M} \not\models M' = N'$. ■

In order to complete the proof of the 1-Section Theorem, we are left with proving Lemma 4.1. In outline, we first prove a restricted version of Lemma 4.1, where the terms M and N to be distinguished only take arguments of first-order type, *i.e.*, those with type $(\iota \to \ldots \to \iota)$. We then show, for the more general case when M and N's arguments are *not* of first-order type, how to reduce the problem to terms that take arguments of only first-order type. In proving the two lemmas, we will always assume that the terms are in extended $\beta\eta$-normal form, which we may assume without loss of generality by Proposition 2.8.

We begin by establishing the first claim.

Lemma 4.2 (Statman [16]) *Suppose $M \not\equiv N$ are closed extended $\beta\eta$-normal forms of type*

$$\sigma = (\sigma_1 \to \ldots \to \sigma_n \to \iota)$$

where each σ_i is a first-order type. Then there is a closed L of type $(\sigma \to (\iota \to \iota \to \iota) \to \iota \to \iota)$ where $(L\ M) \neq_{\beta\eta} (L\ N)$.

Proof: Given $M \not\equiv N$ in extended $\beta\eta$-normal form, our goal is to find an appropriate L. In fact, the definition of L, which is done in two stages, will only depend on the types of M and N. First, pick any variables $g^{\iota\to\iota\to\iota}$ and x^{ι}, and define

$$\begin{aligned} \mathbf{0} &= x \\ (\boldsymbol{n} + \mathbf{1}) &= (f\ x\ \boldsymbol{n}) \end{aligned}$$

Suppose the type $\sigma_i = (\iota \to \ldots \to \iota)$ with $(k+1)$ occurrences of ι; then define

$$P_i = \begin{cases} \boldsymbol{i} & \text{if } k = 0 \\ \lambda y_1^{\iota}.\ \ldots \lambda y_k^{\iota}.\, g\ \boldsymbol{i}\ (g\ y_1\ (g\ y_2\ (\ldots(g\ y_{k-1}\ y_k)\ldots))) & \text{otherwise} \end{cases}$$

and finally set $L = \lambda w^{\sigma} .\, \lambda g^{\iota\to\iota\to\iota} .\, \lambda x^{\iota} .\, w\, P_1 \ldots P_n$.

We claim that $(L\ M) \neq_{\beta\eta} (L\ N)$. The proof goes by induction on the extended $\beta\eta$-normal forms of M and N. In the basis, $M = (\lambda x_1^{\sigma_1} .\, \ldots \lambda x_n^{\sigma_n} .\, x_i^{\iota})$ and $N = (\lambda x_1^{\sigma_1} .\, \ldots \lambda x_n^{\sigma_n} .\, x_j^{\iota})$ where $i \neq j$. Then calculating,

$$\begin{aligned}(L\ M) &=_{\beta\eta} \lambda g .\, \lambda x .\, M\ P_1 \ldots\ P_n \\ &=_{\beta\eta} \lambda g .\, \lambda x .\, P_i \\ &=_{\beta\eta} \lambda g .\, \lambda x .\, \boldsymbol{i}\end{aligned}$$

and similarly, $(L\ M) =_{\beta\eta} \lambda g .\, \lambda x .\, \boldsymbol{j}$. Since both $\boldsymbol{i}$ and $\boldsymbol{j}$ are terms in $\beta\eta$-normal form and $\boldsymbol{i} \neq \boldsymbol{j}$, $\boldsymbol{i} \neq_{\beta\eta} \boldsymbol{j}$ by Proposition 2.6. Thus, $(L\ M) \neq_{\beta\eta} (L\ N)$ as desired.

In the induction case, there are three cases (up to symmetry) depending on the form of M and N:

1. For some $k \geq 1$,

$$\begin{aligned}M &= \lambda x_1^{\sigma_1} .\, \ldots \lambda x_n^{\sigma_n} .\, x_i^{\iota} \\ N &= \lambda x_1^{\sigma_1} .\, \ldots \lambda x_n^{\sigma_n} .\, x_j\, (N_1\ x_1 \ldots x_n)\ \ldots\ (N_k\ x_1 \ldots x_n)\end{aligned}$$

2. For some $k, l \geq 1$ and $i \neq j$,

$$\begin{aligned}M &= \lambda x_1^{\sigma_1} .\, \ldots \lambda x_n^{\sigma_n} .\, x_i\, (M_1\ x_1 \ldots x_n)\ \ldots\ (M_k\ x_1 \ldots x_n) \\ N &= \lambda x_1^{\sigma_1} .\, \ldots \lambda x_n^{\sigma_n} .\, x_j\, (N_1\ x_1 \ldots x_n)\ \ldots\ (N_l\ x_1 \ldots x_n)\end{aligned}$$

3. For some $k \geq 1$

$$\begin{aligned}M &= \lambda x_1^{\sigma_1} .\, \ldots \lambda x_n^{\sigma_n} .\, x_i\, (M_1\ x_1 \ldots x_n)\ \ldots\ (M_k\ x_1 \ldots x_n) \\ N &= \lambda x_1^{\sigma_1} .\, \ldots \lambda x_n^{\sigma_n} .\, x_i\, (N_1\ x_1 \ldots x_n)\ \ldots\ (N_k\ x_1 \ldots x_n)\end{aligned}$$

The first two cases can be argued similarly to the basis; the only difficult case is the third case. By our hypothesis on the types of the terms M and N, the variable $x_i^{\sigma_i}$ has a first-order type. Thus, $(M_j\ x_1 \ldots x_n)$ must have type ι and hence M_j has type σ. Likewise, N_j has type σ. Thus, since $M_j \neq N_j$, it follows from the induction hypothesis that $(L\ M_j) \neq_{\beta\eta} (L\ N_j)$. Performing some calculation,

$$\begin{aligned}(L\ M_j) &=_{\beta\eta} \lambda g .\, \lambda x .\, M_j\ P_1 \ldots\ P_n \\ (L\ N_j) &=_{\beta\eta} \lambda g .\, \lambda x .\, N_j\ P_1 \ldots\ P_n\end{aligned}$$

It therefore follows that $(M_j\ P_1 \ldots\ P_n) \neq_{\beta\eta} (N_j\ P_1 \ldots\ P_n)$. Thus, since

$$\begin{aligned}(L\ M) &=_{\beta\eta} \lambda g .\, \lambda x .\, M\ P_1 \ldots P_n \\ &=_{\beta\eta} \lambda g .\, \lambda x .\, g\ \boldsymbol{i}\ (g\ (M_1\ \vec{P})\ (g\ (M_2\ \vec{P})\ (\ldots (g\ (M_{k-1}\ \vec{P})\ (M_k\ \vec{P}))\ldots))) \\ (L\ N) &=_{\beta\eta} \lambda g .\, \lambda x .\, N\ P_1 \ldots P_n \\ &=_{\beta\eta} \lambda g .\, \lambda x .\, g\ \boldsymbol{i}\ (g\ (N_1\ \vec{P})\ (g\ (N_2\ \vec{P})\ (\ldots (g\ (N_{k-1}\ \vec{P})\ (N_k\ \vec{P}))\ldots)))\end{aligned}$$

it must be the case that $(L\ M) \neq_{\beta\eta} (L\ N)$. This completes the induction case and hence the proof of the lemma. ■

Now our goal is to reduce the original problem to the statement of Lemma 4.2. Let X be the set of first-order variables and let Λ_X be the set of simply-typed λ-terms which contain no constants and whose only free variables are in X. The second lemma for proving Lemma 4.1 states that we may apply M and N to terms in this set to arrive at inequivalent terms:

Lemma 4.3 (Statman [16]) *Suppose M, N are closed extended $\beta\eta$-normal forms of type $(\sigma_1 \rightarrow \ldots \sigma_n \rightarrow \iota)$, and $M \neq N$. Then there exist terms $V_i \in \Lambda_X$ where*

$$(M\ V_1 \ldots V_n) \neq_{\beta\eta} (N\ V_1 \ldots V_n).$$

Proof: By induction on the extended $\beta\eta$-normal form structure of both M and N. In the basis, suppose $M = \lambda x_1^{\sigma_1}. \ldots \lambda x_n^{\sigma_n}. x_i^{\iota}$ and $N = \lambda x_1^{\sigma_1}. \ldots \lambda x_n^{\sigma_n}. x_j^{\iota}$ where by hypothesis, it must be the case that $i \neq j$. Pick distinct variables z_0^{ι} and z_1^{ι}, and let

$$V_p = \begin{cases} z_0^{\iota} & \text{if } p = i \\ z_1^{\iota} & \text{if } p = j \\ \lambda \vec{w}.\, z_0^{\iota} & \text{otherwise} \end{cases}$$

for any $1 \leq p \leq n$. Then it is easy to see that $(M\ V_1 \ldots V_n) =_{\beta\eta} z_0^{\iota}$ and $(N\ V_1 \ldots V_n) =_{\beta\eta} z_1^{\iota}$. Since both z_0^{ι} and z_1^{ι} are in $\beta\eta$-normal form and are distinct, $(M\ V_1 \ldots V_n) \neq_{\beta\eta} (N\ V_1 \ldots V_n)$.

In the induction case, there are a number of cases depending on the form of M and N. The only difficult case is when for some $k \geq 1$,

$$\begin{array}{rcl} M & = & \lambda x_1^{\sigma_1}. \ldots \lambda x_n^{\sigma_n}. x_i\, (M_1\ x_1 \ldots x_n)\ \ldots\ (M_k\ x_1 \ldots x_n) \\ N & = & \lambda x_1^{\sigma_1}. \ldots \lambda x_n^{\sigma_n}. x_i\, (N_1\ x_1 \ldots x_n)\ \ldots\ (N_k\ x_1 \ldots x_n) \end{array}$$

for which we will need the induction hypothesis. Since $M \neq N$, it must be the case that $M_j \neq N_j$ for some $1 \leq j \leq k$. By induction, there exist terms $U_1, \ldots, U_n, \ldots, U_m$ such that $(M_j\ U_1 \ldots U_m) \neq_{\beta\eta} (N_j\ U_1 \ldots U_m)$, and both $(M_j\ U_1 \ldots U_m)$ and $(N_j\ U_1 \ldots U_m)$ are of type ι. Choose fresh variables $h^{\iota \rightarrow \iota \rightarrow \iota}$ and $y_1, \ldots, y_k$, *i.e.*, variables not appearing free in any of the terms $U_1, \ldots, U_m$. For any $1 \leq p \leq n$, define

$$V_p = \begin{cases} \lambda y_1. \ldots \lambda y_k. h\, (y_j\ U_{n+1} \ldots U_m)\, (U_i\ y_1 \ldots y_k) & \text{if } p = i \\ U_p & \text{otherwise} \end{cases}$$

The reader should convince himself that V_i has the appropriate type.

We just need to verify that $(M\ V_1 \ldots V_n) \neq_{\beta\eta} (N\ V_1 \ldots V_n)$. First, we do a little calculation:

$$\begin{array}{rcl} (M\ V_1 \ldots V_n) & =_{\beta\eta} & V_i\, (M_1\ V_1 \ldots V_n) \ldots (M_k\ V_1 \ldots V_n) \\ & =_{\beta\eta} & (\lambda y_1. \ldots \lambda y_k. h\, (y_j\ U_{n+1} \ldots U_m)\, (U_i\ y_1 \ldots y_k))\, (M_1\ V_1 \ldots V_n) \ldots (M_k\ V_1 \ldots V_n) \\ & =_{\beta\eta} & h\, (M_j\ V_1 \ldots V_n\ U_{n+1} \ldots U_m)\, (U_i\, (M_1\ V_1 \ldots V_n) \ldots (M_k\ V_1 \ldots V_n)) \end{array}$$

Similarly,

$$(N\ V_1 \ldots V_n) =_{\beta\eta} h\, (N_j\ V_1 \ldots V_n\ U_{n+1} \ldots U_m)\, (U_i\, (N_1\ V_1 \ldots V_n) \ldots (N_k\ V_1 \ldots V_n))$$

By way of contradiction, assume that $(M\ V_1 \ldots V_n) =_{\beta\eta} (N\ V_1 \ldots V_n)$. By Proposition 2.6, $(M\ V_1 \ldots V_n)$ and $(N\ V_1 \ldots V_n)$ have the same $\beta\eta$-normal form. It follows that

$$(M_j\ V_1 \ldots V_n\ U_{n+1} \ldots U_m) =_{\beta\eta} (N_j\ V_1 \ldots V_n\ U_{n+1} \ldots U_m).$$

Let $H = \lambda u^\iota.\, \lambda v^\iota.\, v$; then by λ-abstracting over h in the two above terms and applying the result to H, we obtain

$$(M_j\ V_1 \ldots V_n\ U_{n+1} \ldots U_m)[f := H] =_{\beta\eta} (N_j\ V_1 \ldots V_n\ U_{n+1} \ldots U_m)[f := H]. \tag{1}$$

But because we chose h to be fresh with respect to the terms $U_1, \ldots, U_m$, h only occurs in the term V_i. Calculating,

$$\begin{aligned} V_i[f := H] &=_{\beta\eta} \lambda y_1. \ldots \lambda y_k.\, H\ (y_j\ U_{n+1} \ldots U_m)\ (U_i\ y_1 \ldots y_k) \\ &=_{\beta\eta} \lambda y_1. \ldots \lambda y_k.\, (\lambda u.\, \lambda v.\, v)\ (y_j\ U_{n+1} \ldots U_m)\ (U_i\ y_1 \ldots y_k) \\ &=_{\beta\eta} \lambda y_1. \ldots \lambda y_k.\, U_i\ y_1 \ldots y_k \\ &=_{\beta\eta} U_i \end{aligned}$$

Thus,

$$\begin{aligned} (M_j\ V_1 \ldots V_n\ U_{n+1} \ldots U_m)[f := H] &=_{\beta\eta} (M_j\ U_1 \ldots U_n\ U_{n+1} \ldots U_m) \\ (N_j\ V_1 \ldots V_n\ U_{n+1} \ldots U_m)[f := H] &=_{\beta\eta} (N_j\ U_1 \ldots U_n\ U_{n+1} \ldots U_m). \end{aligned}$$

It follows from Equation (1) that

$$(M_j\ U_1 \ldots U_n\ U_{n+1} \ldots U_m) =_{\beta\eta} (N_j\ U_1 \ldots U_n\ U_{n+1} \ldots U_m).$$

This contradicts our original choice of $U_1, \ldots, U_m$, so $(M\ V_1 \ldots V_n) \neq_{\beta\eta} (N\ V_1 \ldots V_n)$. This completes the induction case and hence the proof. ■

Proof of Lemma 4.1: Suppose M and N have type $\sigma = (\sigma_1 \to \ldots \to \sigma_n \to \iota)$ and $M \neq_{\beta\eta} N$. By Proposition 2.8, we may assume without loss of generality that M and N are in extended $\beta\eta$-normal form and that $M \neq N$. Thus, by Lemma 4.3, there exist terms $V_i \in \Lambda_X$ such that $(M\ V_1 \ldots V_n) \neq_{\beta\eta} (N\ V_1 \ldots V_n)$. Let $x_1, \ldots, x_m$ be all the free variables appearing in $V_1, \ldots, V_n$ (which are necessarily of first-order type). Then

$$\lambda x_1. \ldots \lambda x_m.\, M\ V_1 \ldots V_n \neq_{\beta\eta} \lambda x_1. \ldots \lambda x_m.\, N\ V_1 \ldots V_n.$$

These two terms have type $\tau = (\tau_1 \to \ldots \to \tau_m \to \iota)$, where each τ_i is a first-order type. Thus, by Lemma 4.2, one can choose a term L of type $(\tau \to (\iota \to \iota \to \iota) \to \iota \to \iota)$ such that

$$(L\ (\lambda x_1. \ldots \lambda x_m.\, M\ V_1 \ldots V_n)) \neq_{\beta\eta} (L\ (\lambda x_1. \ldots \lambda x_m.\, N\ V_1 \ldots V_n)).$$

Thus, let $P = \lambda x^\sigma.\, L\ (\lambda x_1. \ldots \lambda x_m.\, x\ V_1 \ldots V_n)$; then $(P\ M) \neq_{\beta\eta} (P\ N)$. ■

This completes the proof of the 1-Section Theorem.

5 Corollaries of the 1-Section Theorem

The significance of the 1-Section Theorem lies in its corollaries. Here we give some examples of models that satisfy the criterion of the 1-Section Theorem, and then briefly discuss applications of the theorem in the context of simply-typed call-by-name functional languages.

5.1 Models

In most familiar models of the simply-typed λ-calculus, it is easy to check that Statman's 1-Section condition holds. For instance, recall the set-theoretic model $\mathcal{S}$ defined in the introduction:

$$\begin{aligned} \mathcal{S}^{\iota} &= \mathbb{N} \\ \mathcal{S}^{\tau\to\nu} &= [\mathcal{S}^{\tau} \to \mathcal{S}^{\nu}] \end{aligned}$$

Since $\mathcal{S}^{\iota\to\iota\to\iota}$ contains all set-theoretic functions on the integers, it contains the function p, where $p(x)(y) = 2^x 3^y$ for any natural numbers x and y. The function p is a pairing function; choosing this function for the representation of F and any natural number as the representation of C, we may faithfully embed $\mathcal{T}$ into $\mathcal{S}$. Thus, by the 1-Section Theorem, $\beta\eta$-equality completely axiomatizes the valid equations in $\mathcal{S}$, proving Theorem 1.1.

The proof of Theorem 1.1 is much easier than the proof contained in [4], and may be easily adapted to other situations as well. For instance, we may easily prove the following theorem:

Theorem 5.1 (Berger & Schwichtenberg [2]) *Suppose a model $\mathcal{M}$ provides meaning for all primitive recursive functions over the base type $\mathbb{N}$. Then $\beta\eta$-reasoning is complete for proving equations in $\mathcal{M}$.*

This follows immediately from the fact that in any model $\mathcal{M}$ with representations for all the primitive recursive functions, the function p defined above is representable. Berger and Schwichtenberg's proof technique (which may be of independent interest) is completely different and much more complex, relying upon an "inverse" to an evaluation function.

Even in models based on posets rather than sets, Statman's 1-Section condition is easy to verify. Consider, for instance, the continuous model $\mathcal{N}$ defined in the introduction. Let the continuous function $p' \in [\mathbb{N}_\perp \to_c [\mathbb{N}_\perp \to_c \mathbb{N}_\perp]]$ be defined by

$$p'(x)(y) = \begin{cases} \perp & \text{if } x = \perp \text{ or } y = \perp \\ 2^x 3^y & \text{otherwise} \end{cases}$$

Together with any natural number $n \in \mathbb{N}$, p' and n can be used to faithfully embed the algebra $\mathcal{T}$, and so Theorem 1.2 follows immediately. Note that the same proof works for the model composed of all **monotone** functions as well.

The 1-Section Theorem applies to classes of models as well as single models. For instance, Theorem 1.3, which states that $\beta\eta$-reasoning is complete for the class $\mathcal{F}$ of models with finite base type, follows as a corollary. To see this, note that for each distinct pair of terms t_i and t'_i in the syntax t given above, there is a finite model $\mathcal{M}_i$ which distinguishes $M_i = \lambda g.\, \lambda x.\, t_i$ and $N_i = \lambda g.\, \lambda x.\, t'_i$; the full set-theoretic model $\mathcal{X}$ over a finite base set X with $|X|$ = the number of subterms of t_i and t'_i is one choice for $\mathcal{M}_i$. Let f_i and c_i be the elements of $\mathcal{M}_i$ used to distinguish M_i and N_i; then $f = \langle f_0, f_i, \ldots\rangle$ and $c = \langle c_0, c_1, \ldots\rangle$ serve for embedding the algebra $\mathcal{T}$ into the countable direct product of $\{\mathcal{M}_0, \mathcal{M}_1, \ldots\}$. Thus, by the 1-Section Theorem, $\beta\eta$-reasoning is complete for $\mathcal{F}$.

5.2 Implications for Programming Languages

The 1-Section Theorem may also be applied to reasoning about programming languages based on the simply-typed λ-calculus. Here we give some brief examples that show that $\beta\eta$-equality is *complete* for reasoning about fragments of certain call-by-name languages.

One example of a language based on the simply-typed λ-calculus is the language PCF [15, 9]. PCF without booleans includes constants for numerals, successor, predecessor, conditional, and fixpoint operators for recursion, where the conditional operator checks its first argument to see if it is 0 and branches. A precise definition of the language and its interpreter may be found in [12]; importantly, applications are evaluated **call-by-name**.

To the programmer, two pieces of PCF code are "equivalent" if they can be used interchangeably. For closed terms of base type in PCF, two terms are interchangeable iff both diverge or both produce the same numeral when evaluated. This notion of equality tells us nothing about open terms or terms of higher-type, but we may extend the equality to general terms in the following natural manner:

Definition 5.2 Two PCF terms M and N are **observationally congruent** (written $M \equiv_{PCF} N$) iff for any context $C[\cdot]$ (a term with a "hole") such that $C[M]$ and $C[N]$ are closed terms of base type, $C[M]$ and $C[N]$ either both diverge or both yield the same numeral.

For instance, the two PCF terms $(\lambda x.\, \mathsf{pred}\ (\mathsf{succ}\ x))$ and $(\lambda x.\, x)$ are observationally congruent, since both are, in some sense, the identity function on the integers.

PCF observational congruence is too complex to be axiomatized by any reasonable system. All partial recursive functions are representable in PCF in such a way that two partial recursive functions are equivalent iff their representations are equivalent. Thus, since equivalence of partial recursive functions cannot be axiomatized in an r.e. proof system, neither can observational congruence [15, 20]. But if we are willing to settle for proving observational congruences among **pure terms**—those not involving the constants of PCF—we can obtain a complete proof system. The following remark is due to Albert Meyer:

Theorem 5.3 *For two pure simply-typed terms M and N, $M =_{\beta\eta} N$ iff $M \equiv_{PCF} N$.*

The theorem follows from the fact that one can define a strict pairing function p'' in PCF by the term $(\lambda x^{\iota}.\, \lambda y^{\iota}.\, (2^x 3^y))$, where exponentiation and multiplication are defined from successor, predecessor, and recursion in the usual way. (Full recursion is, of course, unnecessary for defining a pairing function, *e.g.*, a representation for addition and multiplication suffices [3].) Thus, the model constructed out of observational congruence classes of terms in PCF (which can be verified to be a model) satisfies the conditions of the 1-Section Theorem, and hence observational congruence and $\beta\eta$-equality of pure terms coincides.

Theorem 1.2 also implies a similar fact for reasoning about a *parallel* version of PCF. Consider the usual PCF language augmented with a "parallel or" constant por of type $(\iota \to \iota \to \iota)$; por returns 0 if either of its arguments reduces to 0, 1 if both reduce to numerals not equal to 0, and diverges otherwise. We call the extended language *Parallel PCF*, or PPCF for short. We may extend the model $\mathcal{N}$ of continuous functions defined above to a model $\mathcal{N}'$ of PPCF by interpreting the constants in the right way, and in this model, denotational equality coincides with observational congruence [9, 14, 19]:

Theorem 5.4 *For any PPCF terms M and N, $\mathcal{N}' \models M = N$ iff $M \equiv_{PPCF} N$.*

Thus, it follows from Theorem 1.2 that

Theorem 5.5 *For two pure simply-typed terms M and N, $M =_{\beta\eta} N$ iff $M \equiv_{PPCF} N$.*

Coupled with Theorem 5.3, Theorem 5.5 implies that among pure terms, sequential and parallel observational congruence theories coincide.

The practical implications of Theorems 5.3 and 5.5 should be clear: for reasoning about pure terms, $\beta\eta$-equality proves all operationally valid equations. In more philosophical terms, $\beta\eta$-reasoning is the core of any reasoning system for these two languages. One may then extend the reasoning systems to include equations about successors and predecessors, or to more complex systems involving induction principles for reasoning about recursion. Such systems will necessarily be incomplete, but at the very least, the core of the reasoning system will be complete.

6 Conclusion

We have demonstrated the use of Statman's 1-Section Theorem in proving that $\beta\eta$-equality is complete for proving the valid equations in a model. Essentially, Statman's 1-Section Theorem isolates a single combinatorial argument for proving the completeness of $\beta\eta$-reasoning in a model of the simply-typed λ-calculus. The combinatorial fact needed, that a binary tree algebra can be faithfully embedded in the particular model, is quite easy to check, and thus Statman's 1-Section Theorem simplifies the proofs of many completeness results.

The power of Statman's 1-Section Theorem is unquestionable, but one may well wonder whether a simpler version of the theorem is possible. For instance, the tree algebra might be unnecessary in the statement of the theorem; the unary algebra $\mathcal{U}$, whose elements are in the grammar

$$U ::= C \mid (F\ U)$$

and $U_1 =_{\mathcal{U}} U_2$ iff U_1 and U_2 are syntactically equivalent, may suffice. In other words, we could imagine that it is sufficient to embed the algebra $\mathcal{U}$ in a model in order for $\beta\eta$-reasoning to be complete. This conjecture, however, is false:

Theorem 6.1 (Subrahmanyam) *There is a model $\mathcal{M}$ in which the algebra $\mathcal{U}$ can be faithfully embedded but whose valid equations are not completely axiomatized by $=_{\beta\eta}$.*

Proof: Consider the model $\mathcal{M}$ over the signature $F^{\iota\to\iota}, C^{\iota}$ built in the following manner:

1. Let $\mathcal{L}^{\sigma}$ be the set of $\beta\eta$-equivalence classes of closed terms of type σ, *viz.*,

$$\mathcal{L}^{\sigma} = \{[M] : M \text{ is closed of type } \sigma\},$$

   where $[M] = \{N : N \text{ is closed and } M =_{\beta\eta} N\}$.

2. Define the binary relations $\simeq^{\sigma}$ on elements of $\mathcal{L}^{\sigma}$ by induction on types as follows:
   - $[M] \simeq^{\iota} [N]$ iff $[M] = [N]$;
   - $[M] \simeq^{\tau\to\nu} [N]$ iff for all $[P] \simeq^{\tau} [Q]$, $[M\ P] \simeq^{\nu} [N\ Q]$.

3. Let $\mathcal{M}^{\sigma}$ be the set of $\simeq^{\sigma}$ equivalence classes of terms; abusing notation, we write $[M]$ for these equivalence classes. Finally, define application of these elements by $\mathsf{Ap}([M],[N]) = [M\ N]$.

It is easy to check that $\mathcal{M}$ is a type frame (extensionality holds by construction) and a model. Note that in $\mathcal{M}$, each element in $\mathcal{M}^{\iota\to\iota\to\iota}$ has one of forms

- $[\lambda x.\, \lambda y.\, F\,(F\,(\ldots(F\,C)\ldots))]$,
- $[\lambda x.\, \lambda y.\, F\,(F\,(\ldots(F\,x)\ldots))]$, or
- $[\lambda x.\, \lambda y.\, F\,(F\,(\ldots(F\,y)\ldots))]$,

where the number of leading F's is ≥ 0. Now consider the terms

$$\begin{aligned} P &= \lambda f.\, \lambda x.\, \lambda y.\, f\,(f\,x\,y)\,(f\,x\,y) \\ Q &= \lambda f.\, \lambda x.\, \lambda y.\, f\,(f\,x\,y)\,(f\,y\,y) \end{aligned}$$

where f has type $(\iota \rightarrow \iota \rightarrow \iota)$ and x, y have type ι. A simple case analysis shows that P and Q are equivalent in the model. Nevertheless, $P \neq_{\beta\eta} Q$. ■

Other extensions to the 1-Section Theorem look much more promising. For example, we may consider adding first-order algebraic theories to the simply-typed λ-calculus, and ask when (β), (η), and the equations of the algebraic theory completely axiomatize the valid equations of the model. For instance, we could axiomatize the equations of the algebra $(\mathbf{N}, \mathbf{0}, +)$ by

$$\begin{aligned} &0 + x = x \\ &x + y = y + x \\ &x + (y + z) = (x + y) + z \end{aligned}$$

Preliminary results with Ramesh Subrahmanyam [13] give a version of the 1-Section Theorem for algebraic theories. Extensions to when simply-typed λ-calculus has a *lazy* or *call-by-value* semantics are also being investigated.

Acknowledgements

I thank Albert Meyer for initially pointing me to Statman's 1-Section Theorem and convincing me of its importance, and also for comments on a draft of this paper. I also thank Andre Scedrov for his encouragement and pointing out the work of Berger and Schwichtenberg, Ramesh Subrahmanyam for pointing out Theorem 6.1, and Harry Mairson for comments on a draft of this paper.

References

[1] Henk P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*, volume 103 of *Studies in Logic*. North-Holland, 1981. Revised Edition, 1984.

[2] U. Berger and H. Schwichtenberg. An inverse of the evaluation functional for typed λ-calculus. In *Proceedings, Sixth Annual IEEE Symposium on Logic in Computer Science*, pages 203–211, 1991.

[3] Herbert B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, 1972.

[4] Harvey Friedman. Equality between functionals. In Rohit Parikh, editor, *Logic Colloquium '73*, volume 453 of *Lect. Notes in Math.*, pages 22–37. Springer-Verlag, 1975.

[5] Carl A. Gunter and Dana S. Scott. Semantic domains. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 633–674. Elsevier, 1990.

[6] Albert R. Meyer. What is a model of the lambda calculus? *Information and Control*, 52:87–122, 1982.

[7] John C. Mitchell. Type systems for programming languages. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 365–458. Elsevier, 1990.

[8] Gordon D. Plotkin. λ-definability and logical relations. Technical Report SAI-RM-4, University of Edinburgh, School of Artificial Intelligence, 1973.

[9] Gordon D. Plotkin. LCF considered as a programming language. *Theoretical Computer Sci.*, 5:223–257, 1977.

[10] Gordon D. Plotkin. Notes on completeness of the full continuous type hierarchy. Unpublished manuscript, Massachusetts Institute of Technology, November 1982.

[11] Jon G. Riecke. Fully abstract translations between functional languages (preliminary report). In *Conference Record of the Eighteenth Annual ACM Symposium on Principles of Programming Languages*, pages 245–254. ACM, 1991.

[12] Jon G. Riecke. *The Logic and Expressibility of Simply-Typed Call-by-Value and Lazy Languages.* PhD thesis, Massachusetts Institute of Technology, 1991.

[13] Jon G. Riecke and Ramesh Subrahmanyam. Reasoning for simply-typed languages in the presence of first-order equations. Extended abstract submitted to LICS '92, December 1991.

[14] V.Yu. Sazonov. Expressibility of functions in D. Scott's LCF language. *Algebra i Logika*, 15:308–330, 1976. Russian.

[15] Dana Scott. A type theoretical alternative to CUCH, ISWIM, OWHY. Unpublished manuscript, Oxford University, 1969.

[16] Richard Statman. Completeness, invariance and lambda-definability. *J. Symbolic Logic*, 47:17–26, 1982.

[17] Richard Statman. Equality between functionals revisited. In L.A. Harrington, *et al.*, editor, *Harvey Friedman's Research on the Foundations of Mathematics*, volume 117 of *Studies in Logic*, pages 331–338. North-Holland, 1985.

[18] Richard Statman. Logical relations in the typed λ-calculus. *Information and Control*, 65:86–97, 1985.

[19] Allen Stoughton. Message to types@theory.lcs.mit.edu electronic mail forum, July 1989.

[20] Christopher P. Wadsworth. The relation between computational and denotational properties for Scott's D_∞ models. *SIAM J. Computing*, 5(3):488–521, 1976.