MIT/LCS/TM-201

WHAT IS A MODEL OF THE LAMBDA CALCULUS?

EXPANDED VERSION

Albert R. Meyer

July 1981

WHAT IS A MODEL OF THE LAMBDA CALCULUS? EXPANDED VERSION

by

Albert R. Meyer
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

JULY 1, 1981

Abstract. An elementary, purely algebraic definition of model for the untyped lambda calculus is given. This definition is shown to be equivalent to the natural semantic definition based on environments. These definitions of model are consistent with, and yield a completeness theorem for, the standard axioms for lambda convertibility. A simple construction of models for lambda calculus is reviewed. The algebraic formulation clarifies the relation between combinators and lambda terms.

Keywords: lambda calculus, combinatory algebra, combinator, model, Bohm tree, β-conversion, extensional

This work was supported in part by The National Science Foundation, Grant Nos. MCS 7719754 and MCS 8010707, and by a grant to the M.I.T. Laboratory for Computer Science by the IBM Corporation.

1. Introduction. Lambda notation provides a convenient means for writing expressions which denote functions. As an informal example, consider the polynomial expression x^2+7x-1. One can construct an expression $\lambda x.x^2+7x-1$ called a lambda *abstraction* denoting the polynomial function whose values are given by the polynomial expression. Thus, $\lambda x.x^2+7x-1$ could be read as "the function of x whose value is x^2+7x-1," and the defining equation $p(x) = x^2+7x-1$ for the polynomial p could as well be written $p = \lambda x.x^2+7x-1$. The value of p at the argument 3, for example, is obtainable by *applying* the expression $\lambda x.x^2+7x-1$ to 3, which entails substituting 3 for x to obtain $3^2+7\cdot3-1$ and evaluating the result to obtain 29. This process of substitution and evaluation reflects the computational behavior of many modern programming languages -- which explains in part the recent interest in the lambda calculus among computer scientists (*cf.* [Landin 64,65; Stoy 77]).

Some of the power of the lambda calculus is suggested by the way functions of several arguments can be handled. The addition function of two variables, for example, whose value is the sum of the values of the variables, could be denoted $\lambda x.\lambda y.x+y$. More accurately, the value of $\lambda x.\lambda y.x+y$ is a *functional* which, applied say to the argument 2, yields the add-two function of one variable: $\lambda y.2+y$. The add-two function can in turn be applied to the argument 4 to yield the sum 6. Thus a function of two variables can be regarded as a functional of one variable whose value is a function of one variable, in this case an "add a constant" function. Thus, in studying calculations with lambda notations there is no loss of generality in restricting attention to functions -- or more precisely functionals -- of one argument, and we shall do so in what follows.

A more intriguing example suggesting the importance and special character of the lambda calculus is the "triple composition" functional T. For any function f of one argument and positive integer n, let $f^{(n)}$ denote the composition $f \circ f \circ \cdots \circ f$ of f with itself n times. The functional T can be defined by the equation $T(f) = f^{(3)}$ or equivalently by $T = \lambda f.(\lambda x.f(f(f(x))))$. Thus, T applied to the cubic polynomial $\lambda z.z^3$ would yield the 27^{th} degree polynomial $\lambda z.z^{27}$. By the same reasoning, T applied to T equals the "compose 27 times" functional because T(T) applied to f equals $(T(T))(f) = (T \circ T \circ T)(f) = T(T(T(f))) = T(T(f^{(3)})) = T(f^{(3)} \circ f^{(3)} \circ f^{(3)}) = T(f^{(9)}) = f^{(27)}$.

But although it makes good intuitive sense to define the value of T(T) in this way, there are obvious logical difficulties. Applying a function to itself violates the rules of ordinary set theory which forbid a function from being in its own domain. The violation can quickly lead to contradiction. For example, let P be the "paradoxical" functional such that P(f) is zero if f(f) is not the integer zero, and P(f) is the integer one otherwise. So by definition P(f) $\neq$ f(f) for all f; substituting P for f immediately yields the contradiction P(P) $\neq$ P(P).

The problem we consider is how to give values to such expressions involving functionals which may be applied to themselves. The intuitive sense of examples like T(T) must be preserved while avoiding contradictions from examples like P(P). Such a domain of values would be a *model* for the lambda calculus.

Lambda calculus has been the subject of research by logicians for roughly fifty years, although the *model theory* of the lambda calculus is a development primarily of the past decade -- largely carried out following the lead of Dana Scott. While a host of models and methods for model construction are now available, the clear statement of just what *in general* a model of the lambda calculus may be seems not to be well known. The purpose of this paper, which is largely tutorial, is to review briefly why there is an apparent difficulty in defining the notion of model for the lambda calculus, and then to show how this difficulty is overcome.

This question of what a model of the lambda calculus is has bothered me for some time. A similar concern is expressed in [Hindley and Longo, 1980] who comment that, "...there seems to be, firstly an assumption that the definition is too obvious to need stating, and secondly a disagreement about what the definition should be." Reading through the literature describing the various model constructions ([Plotkin 72, Wadsworth 76, Scott 76, Stoy 77, Plotkin 78, Engeler 79, Scott 80a,c]), I felt as though I kept asking, "What is a group?" and kept being told "Permutations on n letters are a group," or "$\mathbb{Z}_k$ is a group," but was never told that a group is simply an algebraic structure with a binary operation satisfying the well known conditions. It turns out that there *is* a comparably simple definition of model for the lambda calculus which we state next. The remainder of this paper provides a justification for the claim that the following definition is appropriate.

Definition. A *combinatory algebra* is a structure $\langle \mathbf{D}, \cdot \rangle$ where $\cdot$ is a binary operation on $\mathbf{D}$, such that there are elements $\mathbf{K}, \mathbf{S} \in \mathbf{D}$ satisfying

(1.1) $(K \cdot d_0) \cdot d_1 = d_0$, and

(1.2) $((S \cdot d_0) \cdot d_1) \cdot d_2 = (d_0 \cdot d_2) \cdot (d_1 \cdot d_2)$

for all $d_0, d_1, d_2 \in D$.

A *combinatory model* of the lambda calculus is a structure $\langle D, \cdot, \epsilon \rangle$ where $\langle D, \cdot \rangle$ is a combinatory algebra and $\epsilon \in D$ satisfies

(1.3a) $(\epsilon \cdot d_0) \cdot d_1 = d_0 \cdot d_1$,

(1.3b) if $\forall d \in D(d_0 \cdot d = d_1 \cdot d)$ then $\epsilon \cdot d_0 = \epsilon \cdot d_1$,

for all $d_0, d_1 \in D$, and

(1.3c) $\epsilon \cdot \epsilon = \epsilon$.[1]

Combinatory models serve for what is known as the β-lambda calculus. For the other main variant, known as η-lambda calculus, we simply require that the element ϵ be a left identity on D, i.e., $\epsilon \cdot d = d$ for all $d \in D$. Equivalently, we can simplify condition (1.3) to (1.4).

Definition. An *extensional combinatory model* of the lambda calculus is a combinatory algebra $\langle D, \cdot \rangle$ such that

(1.4) if $\forall d \in D(d_0 \cdot d = d_1 \cdot d)$ then $d_0 = d_1$

for all $d_0, d_1 \in D$.

The above definitions are not specially new. The definition of combinatory algebra is due to Curry. Condition (1.4) is known as *extensionality*[2], and the fact that extensional combinatory algebras serve as models for the η-calculus has been observed often [Barendregt 77, Hindley and Longo 80, Scott 80a]. A variant of the definition of combinatory model above is mentioned along the way in [Scott 80b]. Other slightly more complex but still simple, purely algebraic formulations appear in [Obtulowicz 77, Obtulowicz and Wiweger 78, Volken 78, Aczel 80, Barendregt and Longo 80, Barendregt 81]. [Barendregt 81, CH.5§4, Cooperstock 81] survey many of these.

Nevertheless, it still seems worthwhile to emphasize again here that the general definition of lambda calculus model can be formulated in this elementary way without any of the algebraic baggage -- very useful for other purposes -- of lattices, continuity, or categories, and also without any of the syntactic baggage of lambda calculus terms. Although the results described are known in one form or another to a number of researchers, I have not seen the story told in quite so elementary a way as attempted below.[3]

To keep this paper self-contained, we review in the next section the basic definitions of the syntactic properties of what is known as the *untyped* lambda calculus. It will turn out that most of the standard syntactic results about reductions, normal forms, and Church-Rosser properties will not be needed in our development. The main syntactic notion required is merely that of a *lambda theory*, namely, a system of equations between lambda terms closed under the standard inference rules. (See [Hindley, Lercher, and Seldin 72] for a more complete treatment of the syntactic theory and [Barendregt 81] as a comprehensive reference.)

Section 3 introduces *environment models*. We develop enough of their properties to explain the view that environment models are the natural, most general formulation of what might be meant by mathematical models for the untyped lambda calculus. In particular, the axioms and rules of inference of lambda calculus are sound when interpreted by environment models; every environment model thus yields an associated lambda theory. The consistency of lambda calculus -- a purely syntactic notion commonly proved by syntactic (Church-Rosser) properties -- is shown to follow from the existence of nontrivial environment models. The central result is a completeness theorem demonstrating that every lambda theory is the theory associated with some environment model.

The drawback of environment models is that they define purely algebraic conditions by induction on the syntactic structure of lambda terms. In Section 4 we demonstrate the equivalence between the combinatory models defined above and environment models, thereby revealing how to formulate the algebraic conditions needed for models without reference to syntax.

As an easy application of the notion of combinatory model, we will see in Section 5 that the construction of a lambda calculus model in [Engeler 79] follows for simpler, more general reasons than was demonstrated there. This completes the main part of the story.

In Sections 6 and 7 we indulge in an algebraic excursion in which several structures akin to combinatory algebras are defined and compared. In Section 6 *lambda models* are introduced; they provide a technically useful variation of combinatory models. In Section 7 the connection between lambda terms and combinatory terms given in Sections 4 and 6 leads to the formulation of *lambda algebras*, which are *not* equivalent either to combinatory algebras or lambda models, but retain the best features of both. In the final section we cite some additional results connecting the model theory and proof theory of lambda calculus.

2. Syntax and Lambda Theories. We let x, y, z denote variables chosen from some fixed infinite set of variables, **d** denote a constant, and u, v, w denote *lambda terms* defined inductively as follows. A lambda term is either a variable, a constant, an *application* of the form (uv), or an *abstraction* of the form (λx u). For readability the notation λx.u is usually used for abstractions, and parentheses are omitted in applications with association to the left being understood. Thus, uvw abbreviates ((uv)w). Occurrences of variables in terms are said to be *bound* or *free* following the usual rules as though λx was a quantifier such as $\exists$x. Finally, $\lambda x_1 x_2 ... x_n.u$ abbreviates $\lambda x_1.\lambda x_2....\lambda x_n.u$. For every set C of constants, we let $\Lambda(C)$ denote the lambda terms whose constants are chosen solely from C; so $\Lambda(\emptyset)$ denotes the constant-free or *pure* terms.

Let u[v/x] denote the result of substituting the term v for *free* occurrences of x in u subject to the usual provisos about renaming bound variables in u to avoid capture of free variables in v. Two basic axiom schemes (α) and (β), and an optional third axiom scheme (η), reflect the intuition behind abstraction and application.

(α) $(\lambda x\ u) = (\lambda y\ u[y/x])$ for y not free in u,

(β) $((\lambda x\ u)v) = u[v/x]$,

(η) $(\lambda y\ (uy)) = u$ for y not free in u.

With these axiom schemes we take the usual inference rules for a congruence relation, namely three rules of inference, (transitivity and symmetry) $u = v,\ u = v' \vdash v = v'$,

(congruence) $u = u',\ v = v' \vdash (uv) = (u'v')$,

(ξ) $u = v \vdash (\lambda x\ u) = (\lambda x\ v)$.

We would also insist on the additional axiom

(reflexivity) $u = u$,

except that it happens to follow already from (β) and (transitivity and symmetry) as the reader may check.

Terms which are provably equal by these rules from instances of (α), (β) (and (η)), are said to (η-)*convert* to one another. Note that convertibility is an equivalence relation on terms -- transitivity and symmetry follow immediately from the corresponding inference rule once reflexivity is proved.

Axioms and inference rules lead directly to the notion of a theory.

Definition. A *lambda theory* $\mathcal{T}$ over a set C of constants is a set of equations between terms in $\Lambda(C)$ containing all instances of (α) and (β), and closed under the rules (transitivity and symmetry), (congruence), and (ξ). The notation $\vdash_{\mathcal{T}} u = v$ means that the equation "$u = v$" is in $\mathcal{T}$. The theory is *extensional* if it also contains all instances of (η).

Clearly, if u converts to v, then $\vdash_{\mathcal{T}} u = v$ for all lambda theories $\mathcal{T}$.

As with convertibility, equality in any lambda theory $\mathcal{T}$ defines an equivalence relation on lambda terms. The $\mathcal{T}$-equivalence class of u is denoted $[[u]]_{\mathcal{T}}$. Namely,

$$[[u]]_{\mathcal{T}} = \{v \mid \vdash_{\mathcal{T}} u = v\}.$$

Because the axioms and inference rules are given by schemes in which any terms may be substituted for u and v, it follows that simultaneous substitution preserves equations. That is, if $\vdash_{\mathcal{T}} u_i = v_i$ for $i=0,\ldots,n$, then $\vdash_{\mathcal{T}} u_0[u_1/x_1,\ldots u_n/x_n] = v_0[v_1/x_1,\ldots v_n/x_n]$,

where $u_0[u_1/x_1,\ldots u_n/x_n]$ denotes the result of *simultaneously* substituting $u_1,\ldots u_n$ for free occurrences of $x_1,\ldots,x_n$ in u_0. (We require of course that the variables $x_1\ldots,x_n$ be distinct and, as in the case of (β), that bound variables of u_0 are renamed to avoid capture of free variables in $u_1,\ldots u_n$.)

This is all we need in the way of syntactic notions about lambda calculus.

3. Values of Terms and Environment Models. The simplest first notion of lambda calculus model might be any set **D** of *values* together with a mapping from any lambda term u to a value $[[u]] \in \mathbf{D}$ such that convertible terms are assigned the same value. Clearly it is a minimal requirement of any notion of model that convertible terms receive the same value in the model. This ought not be the sole requirement of models, however, because it does not guarantee the condition that the value of a term determines its behavior with respect to the values of other terms. Specifically, we expect the inference rules to be *sound*, which leads to the following

Definition. A(n *extensional*) *term model* of the lambda calculus over a set C of constants consists of a set **D** whose elements are called *values*, and a mapping $[[\cdot]]$ from Λ(C) *onto* **D** such that

$[[(\lambda x\ u)]] = [[(\lambda y\ u[y/x])]]$ for y not free in u,

$[[((\lambda x\ u)v)]] = [[\ u[v/x]\ ]]$,

($[[(\lambda y\ (uy))]] = [[u]]$ for y not free in u,) and

if $[[u]] = [[v]]$ and $[[u']] = [[v']]$, then
$[[(uu')]] = [[(vv')]]$ and $[[(\lambda x\ u)]] = [[(\lambda x\ v)]]$.

Clearly, term models are a trivial reformulation of lambda theories. Namely, if $\mathcal{T}$ is a lambda theory, then mapping a term to its $\mathcal{T}$-equivalence class yields a term model. Conversely, if $[[\cdot]]$ is the mapping of a term model, then the set of equations "u = v" such that $[[u]] = [[v]]$ is the lambda theory which yields the term model.

So although the notion of term model is simple and natural *given* the axioms and rules for lambda calculus, it remains an essentially syntactic notion which hardly serves to justify belief or interest in the axioms. That is, term models fail to capture the intuitive idea of lambda terms as descriptions of functions. (It is like saying that a model of group theory is any assignment of truth values to formulas such that provably equivalent formulas about groups receive the same truth value. The central model-theoretic notion which justifies the rules of proof, namely the notion of how an algebraic structure satisfies a formula, has been left out.)

What sort of structure allows interpretation of lambda terms? It would be easiest if we could appeal to the standard inductive definition of the value of a term over an ordinary algebraic structure -- for example, a structure $\mathcal{C} = \langle \mathbf{D}, \cdot \rangle$ where $\cdot$ is a binary operation on the set **D**. It will be helpful to review how values are determined in this standard case.

We define the set of *$\mathcal{C}$-terms* to be constructed from constants in **D**, variables, parentheses, and a symbol for the binary operation $\cdot$ on **D**. Actually for reasons which will appear below, it will be convenient to omit the symbol for $\cdot$ and write (uv) instead of (u·v), so that $\mathcal{C}$-terms become the special case of lambda terms in which lambda abstractions do not occur.

Each $\mathcal{C}$-term has a value in **D** which is determined as soon as an assignment of values to the free variables in the term is given. That is, the term is thought of as defining a function on **D** of as many arguments as there are free variables. It turns out to be simpler technically to regard terms as defining functions of *all* the variables, even though the value will actually depend only on the values of the free variables in the term. In the context of lambda calculus, assignments of values to variables are usually referred to as *environments.*

Formally, an environment ρ is any map from the set of all variables into **D**. The *valuation mapping* $\mathcal{V}_{\mathcal{C}}$ defines for each $\mathcal{C}$-term u, a function $\mathcal{V}_{\mathcal{C}}[\![u]\!]$ from environments to **D**. The value of u in the environment ρ is written as $\mathcal{V}[\![u]\!]\rho$. (We omit the subscript $\mathcal{C}$ whenever it is clear from context.)

The value of a term consisting of a single constant is simply the value of the constant.

(3.1) $\mathscr{V}[d]\rho = \mathbf{d}$ for $\mathbf{d} \in \mathbf{D}$.

The value of a term consisting of a single variable is the value assigned the variable by an environment.

(3.2) $\mathscr{V}[x]\rho = \rho(x)$.

Finally, the value in $\mathscr{C}$ of a term of the form (uv) is simply the result of applying the binary operation $\cdot$ to the values of u and v.

(3.3) $\mathscr{V}_{\mathscr{C}}[(uv)]\rho = (\mathscr{V}_{\mathscr{C}}[u]\rho)\cdot(\mathscr{V}_{\mathscr{C}}[v]\rho)$.

Definition. An equation u = v between $\mathscr{C}$-terms is defined to be *valid* in $\mathscr{C}$, written $\mathscr{C} \models u = v$, iff the values of u and v are the same in all environments. Namely,

$\mathscr{C} \models u = v$ iff $\mathscr{V}_{\mathscr{C}}[u] = \mathscr{V}_{\mathscr{C}}[v]$.

The difficulty in extending these familiar definitions to lambda terms is that lambda abstractions are meant to denote functions, so their values would not, like ordinary terms, be expected to be *elements* of the structure but rather to be *functions* on it. Of course, once we allow functions as values, a lambda term might be applied to a lambda term whose value was determined to be a function, so we must then admit that the value of a term could also be a *functional* on functions on the structure. Things get even messier if we think of applying a term to itself, for, as we noted in the introduction, this violates the rules of set theory which forbid a function from being in its own domain.

The way out of this potential paradox is quite straightforward (and familiar in recursion theory). Namely, we regard each element over the structure as denoting a function on the structure (much as an integer denotes a partial recursive function via a Godel numbering). So we first require of a model that it consist of a nonempty set **D** whose elements will be the values of terms, together with a map Φ from **D** *onto* a set $\mathbf{D} \to \mathbf{D}$ of certain functions from **D** to **D**. We will also want to represent each function in $\mathbf{D} \to \mathbf{D}$ as an element of **D**, so we require an inverse map Ψ from $\mathbf{D} \to \mathbf{D}$ into **D** (much like a mapping from a recursive function to its least Godel number). That is,

$$D \underset{\Psi}{\overset{\Phi}{\rightleftarrows}} D \to D$$

(3.4) $f = \Phi(\Psi(f))$ for all $f \in \mathbf{D} \to \mathbf{D}$.

We shall call the structure $\mathscr{E} = \langle \mathbf{D},\Phi,\Psi \rangle$ a *functional domain*. Note that since Φ maps $\mathbf{D}$ *onto* $\mathbf{D} \to \mathbf{D}$, it follows from Cantor's cardinality theorems that $\mathbf{D} \to \mathbf{D}$ cannot equal the set of all functions from $\mathbf{D}$ to $\mathbf{D}$ except in the trivial case that $\mathbf{D}$ has exactly one element.

Now the intended interpretation of an application (uv) is that u denotes a function applied to the argument v. So the value over a functional domain $\mathscr{E}$ of the application is gotten by interpreting the value of u as a function and applying that function to the value of v.

(3.5) $\mathscr{V}_{\mathscr{E}}[(uv)]\rho = f(\mathscr{V}_{\mathscr{E}}[v]\rho)$ where $f = \Phi(\mathscr{V}_{\mathscr{E}}[u]\rho)$.

Finally, we must assign values to lambda abstractions of the form $\lambda x.u$. The intended interpretation here is that u is an expression which can be evaluated for any given value of x, and $\lambda x.u$ denotes the function f whose value $f(\mathbf{d})$ is obtained from the evaluation of u when x is assigned the value $\mathbf{d} \in \mathbf{D}$. However, since we want values of terms to be elements rather than functions, we define the value of the abstraction to be the element $\Psi(f) \in \mathbf{D}$ which represents the function f. To describe the assignment of $\mathbf{d}$ to x, let $\rho\{\mathbf{d}/x\}$ denote an environment which agrees with ρ at all variables other than x and which assigns x the value $\mathbf{d}$.

(3.6) $\mathscr{V}[\lambda x.u]\rho = \Psi(f)$,
where $f{:}\mathbf{D}\to\mathbf{D}$ is the function such that $f(\mathbf{d}) = \mathscr{V}[u](\rho\{\mathbf{d}/x\})$.

The only possible catch in clause (3.6) is that the function f may not be in the set $\mathbf{D} \to \mathbf{D}$, in which case $\Psi(f)$ is undefined. We take the denial of this possibility as our fundamental definition of model.

Definition. An *environment model*[4] of the lambda calculus is any functional domain such that if values are assigned to lambda terms according to (3.1), (3.2), (3.5), and (3.6) above, the functions $f = \lambda \mathbf{d}\in\mathbf{D}.\mathscr{V}[u](\rho\{\mathbf{d}/x\})$ arising in (3.6) are all in $\mathbf{D} \to \mathbf{D}$.

An equation u = v between lambda terms is defined to be *valid* in an environment model $\mathscr{E}$, written $\mathscr{E} \models u = v$, iff the values of u and v are the same in all environments. Namely,

$$\mathscr{E} \models u = v \text{ iff } \mathscr{V}_{\mathscr{E}}[u] = \mathscr{V}_{\mathscr{E}}[v].$$

To illustrate this definition, consider the term $\lambda x_1 x_2 . x_1$ which intuitively denotes the first projection function of two variables. Let $\mathbf{p}_1$ be the value of $\lambda x_1 x_2 . x_1$ in some environment ρ. For $\mathbf{d}_1, \mathbf{d}_2 \in \mathbf{D}$, let $\mathbf{d}_1 \mathbf{d}_2$ abbreviate $(\Phi(\mathbf{d}_1))(\mathbf{d}_2)$, and let $\mathbf{d}_1 \mathbf{d}_2 ... \mathbf{d}_n$ be read as associated to the left, i.e., as $(...((\mathbf{d}_1 \mathbf{d}_2)\mathbf{d}_3)...\mathbf{d}_n)$. Then we expect $\mathbf{p}_1$ to have the property that $\mathbf{p}_1 \mathbf{d}_1 \mathbf{d}_2 = \mathbf{d}_1$ for all $\mathbf{d}_1, \mathbf{d}_2 \in \mathbf{D}$.

To verify this, observe that $\mathbf{p}_1 \mathbf{d}_1 = (\mathscr{V}[\lambda x_2 . x_1])(\rho\{\mathbf{d}_1/x_1\})$ by (3.6). Then $\mathbf{p}_1 \mathbf{d}_1 \mathbf{d}_2 = \mathscr{V}[x_1]((\rho\{\mathbf{d}_1/x_1\})\{\mathbf{d}_2/x_2\})$ by (3.6) again, and the righthand side of this equation equals $\mathbf{d}_1$ by (3.2).

A more general technical justification of the reasonableness of the definition of environment model comes from the fact that the axioms and rules of lambda calculus already follow from the definition. To show this, we begin by observing, following [Wadsworth 76], that our use of environments properly reflects the properties of substitution in formulas. First, the value of a term depends only on the values of its free variables.

Free Variable Lemma. $\mathscr{V}[u]\rho = \mathscr{V}[u](\rho\{\mathbf{d}/y\})$ for y not free in u.

More generally, we have the

Substitution Lemma. $\mathscr{V}[\ u[v/x]\ ]\rho = \mathscr{V}[u](\rho\{\mathbf{d}/x\})$ for $\mathbf{d} = \mathscr{V}[v]\rho$.

Both lemmas are proved by routine induction on the structure of lambda terms. We obtain directly from the Substitution Lemma the fundamental

Soundness Theorem. The equations valid in an environment model form a lambda theory. In particular, if u converts to v, then u = v is valid in all environment models.

Proof. By (3.5) and (3.6), $\mathcal{V}[\![(uv)]\!]$ and $\mathcal{V}[\![(\lambda x\ u)]\!]$ are determined solely by $\mathcal{V}[\![u]\!]$ and $\mathcal{V}[\![v]\!]$, so (congruence) and (ξ) preserve validity.

To verify that (α) is valid, let $\mathcal{V}[\![(\lambda x\ u)]\!]\rho = \Psi(f)$ as in (3.6), so that $f(\mathbf{d}) = \mathcal{V}[\![u]\!](\rho\{\mathbf{d}/x\})$. Let y be a variable distinct from x and such that y is not free in u. Let $\mathcal{V}[\![(\lambda y\ u[y/x])]\!]\rho = \Psi(g)$, so $g(\mathbf{d}) = \mathcal{V}[\![\ u[y/x]\ ]\!](\rho\{\mathbf{d}/y\})$. Then $\lambda x.u = \lambda y.u[y/x]$ will be valid providing $f = g$.

By the Substitution Lemma, $g(\mathbf{d}) = \mathcal{V}[\![u]\!]((\rho\{\mathbf{d}/y\})\{\mathbf{d}'/x\})$ where $\mathbf{d}' = \mathcal{V}[\![y]\!](\rho\{\mathbf{d}/y\})$. By (3.2), $\mathbf{d}' = \mathbf{d}$. Also, $(\rho\{\mathbf{d}/y\})\{\mathbf{d}'/x\} = (\rho\{\mathbf{d}'/x\})\{\mathbf{d}/y\}$ by definition since $y \neq x$, so $g(\mathbf{d}) = \mathcal{V}[\![u]\!]((\rho\{\mathbf{d}/x\})\{\mathbf{d}/y\})$. By the Substitution Lemma again, $g(\mathbf{d}) = \mathcal{V}[\![\ u[x/y]\ ]\!](\rho\{\mathbf{d}/x\})$, but since y is not free in u, $u[x/y] = u$, so $g(\mathbf{d}) = f(\mathbf{d})$.

Verification of (β) follows even more easily from the Substitution Lemma.

Finally, if u converts to v, then the equation $u = v$ is in every lambda theory, and hence is in the lambda theory of equations valid in any particular environment model. ∎

A simple application of the notion of environment model is a model-theoretic proof of the syntactic consistency of lambda calculus, viz., nonconvertiblity between some pair of terms.

Lemma. The equation $\lambda x_1 \ldots x_n.x_i = \lambda x_1 \ldots x_n.x_j$, where $1 \leq i < j \leq n$, is *not* valid in any environment model with more than one element.

Proof. For $1 \leq k \leq n$ and any environment ρ, let $\mathbf{p}_k = \mathcal{V}[\![\lambda x_1 \ldots x_n.x_k]\!]\rho$. From the definition of $\mathcal{V}$, it follows as in the example above that $\mathbf{p}_k \mathbf{d}_1 \ldots \mathbf{d}_n = \mathbf{d}_k$ for all $\mathbf{d}_1, \ldots, \mathbf{d}_n \in \mathbf{D}$. If $\mathbf{D}$ has more than one element, there exist $\mathbf{d}_1, \ldots, \mathbf{d}_n \in \mathbf{D}$ such that $\mathbf{d}_i \neq \mathbf{d}_j$, so that $\mathbf{p}_i \mathbf{d}_1 \ldots \mathbf{d}_n = \mathbf{d}_i \neq \mathbf{d}_j = \mathbf{p}_j \mathbf{d}_1 \ldots \mathbf{d}_n$ and therefore $\mathbf{p}_i \neq \mathbf{p}_j$. ∎

An immediate consequence of this lemma is that every *nontrivial* model, i.e., model with more than one element, is infinite. Assuming, as we show in Section 5, that nontrivial environment models exist, the preceding lemma and the Soundness Theorem immediately imply the

Consistency Theorem. For $1 \leq i < j \leq n$ the term $\lambda x_1 \ldots x_n.x_i$ does not convert to $\lambda x_1 \ldots x_n.x_j$.

The preceding definitions can be trivially modified to deal with η-conversion. The intuitive content of the η-rule is that an element may be identified with the function it specifies, so that distinct elements must specify distinct functions. This amounts simply to the

Definition. An *extensional environment model* is an environment model for which the map $\Phi: \mathbf{D} \to (\mathbf{D} \to \mathbf{D})$ is one-to-one.

We then can easily show the

η-Soundness Theorem. The equations valid in an extensional environment model form an extensional lambda theory. In particular, if u η-converts to v, then u = v is valid in all extensional environment models.

The Consistency Theorem similarly extends to extensional environment models and η-conversion.

A converse to the Soundness Theorem provides the most important technical support for the argument that environment models correctly capture the intuitive meaning of lambda calculus as embodied in the convertibility rules. The axioms and rules of lambda calculus provide a *complete* logical system for proving equations about environment models.

Completeness Theorem. Every lambda theory consists of precisely the equations valid in some environment model. That is, for every lambda theory $\mathscr{T}$, there is an environment model $\mathscr{E}$ such that

$$\vdash_{\mathscr{T}} u = v \text{ iff } \mathscr{E} \models u = v.$$

In particular, u = v is valid in all environment models iff u converts to v.

The required environment model is given as follows.

Definition. Let $\mathscr{T}$ be a lambda theory over a set C of constants. The functional domain *associated* with $\mathscr{T}$ is $\langle \mathbf{D}, \Phi, \Psi \rangle$ where

$\mathbf{D} = \{ [[u]]_{\mathcal{T}} \mid u \in \Lambda(C)\}$,

$(\Phi([[u]]_{\mathcal{T}}))([[v]]_{\mathcal{T}}) = [[(uv)]]_{\mathcal{T}}$, and

$\Psi(\Phi([[u]]_{\mathcal{T}})) = [[\lambda x.ux]]_{\mathcal{T}}$ for x not free in u.

Note that the (congruence) rule implies that Φ in the above definition is well defined. To see that Ψ is well defined, note that if $\Phi([[u]]_{\mathcal{T}}) = \Phi([[v]]_{\mathcal{T}})$, then evaluating at argument $[[x]]_{\mathcal{T}}$ for x not free in u,v yields $[[ux]]_{\mathcal{T}} = [[vx]]_{\mathcal{T}}$, so ($\xi$) implies $[[\lambda x.ux]]_{\mathcal{T}} = [[\lambda x.vx]]_{\mathcal{T}}$, that is $\Psi(\Phi([[u]]_{\mathcal{T}})) = \Psi(\Phi([[v]]_{\mathcal{T}}))$ by definition. Finally, (β) immediately implies that Φ is a left inverse of Ψ, so $\langle \mathbf{D},\Phi,\Psi\rangle$ is indeed a functional domain.

Proof of the Completeness Theorem. Let $\mathcal{T}$ be a lambda theory and $\mathcal{E}$ its associated functional domain. For any environment ρ, let $u[\rho]$ abbreviate $[[\ u[u_1/x_1,\ldots u_n/x_n]\ ]]_{\mathcal{T}}$ where $x_1,\ldots,x_n$ are the free variables of u and $\rho(x_i) = [[u_i]]_{\mathcal{T}}$. Note that $u[\rho]$ is well defined since simultaneous substitution preserves equations.

We claim that for all $u \in \Lambda(C)$ and environments ρ, if $\mathcal{V}_{\mathcal{E}}[u]\rho$ is defined by (3.1-2), (3.5-6), then

$$\mathcal{V}_{\mathcal{E}}[u]\rho = [[\ u[\rho]\ ]]_{\mathcal{T}}.$$ [5]

In particular, $\mathcal{E}$ is an environment model.

The claim follows by induction on the definition of a lambda term u. We consider only the most difficult case when u is of the form $\lambda x.v$ where x is free in v. In this case,

$$\mathcal{V}_{\mathcal{E}}[u]\rho = \Psi(\lambda d \in \mathbf{D}.\mathcal{V}_{\mathcal{E}}[v](\rho\{d/x\}))$$

by (3.6), providing the argument of Ψ is in the range of Φ. So it suffices to prove that $\lambda d \in \mathbf{D}.\mathcal{V}_{\mathcal{E}}[v](\rho\{d/x\}) = \Phi((\lambda x.v)[\rho])$.

But for any $[[w]]_{\mathcal{T}} \in \mathbf{D}$,

$$(\lambda d \in \mathbf{D}.\mathcal{V}_{\mathcal{E}}[v](\rho\{d/x\}))([[w]]_{\mathcal{T}})$$

$$= \mathcal{V}_{\mathcal{E}}[v](\rho\{[[w]]_{\mathcal{T}}/x\})$$

$$= [[\ v[u_1/x_1,\dots,u_n/x_n,w/x]\ ]]_{\mathcal{T}} \quad \text{by induction hypothesis.}$$

Now let y, z be a new variables not free in $v,w,u_1,\dots u_n$, and let $u'_i = u_i[z/x]$. Then by the definition of simultaneous substitution,

$$[[\ v[u_1/x_1,\dots,u_n/x_n,w/x]\ ]]_{\mathcal{T}} = [[\ v[u'_1/x_1,\dots,u'_n/x_n][w/x][x/z]\ ]]_{\mathcal{T}}$$

$$= [[\ (\ (\lambda x.(v[u'_1/x_1,\dots,u'_n/x_n]))\ w\ )[x/z]\ ]]_{\mathcal{T}} \quad \text{by } (\beta)$$

$$= [[\ (\ (\lambda x.(v[u'_1/x_1,\dots,u'_n/x_n]))[x/z]\ w\ )\ ]]_{\mathcal{T}} \quad \text{since z is not free in w}$$

$$= [[\ (\ (\lambda y.(v[u'_1/x_1,\dots,u'_n/x_n][y/x]))[x/z]\ w\ )\ ]]_{\mathcal{T}} \quad \text{renaming } \lambda x \text{ to } \lambda y \text{ to avoid capture of x}$$

$$= [[\ (\ (\lambda y.(v[y/x][u'_1/x_1,\dots,u'_n/x_n]))[x/z]\ w\ )\ ]]_{\mathcal{T}} \quad \text{since y is not free in } u'_i$$

$$= [[\ (\ \lambda y.(v[y/x][u_1/x_1,\dots,u_n/x_n])\ w\ )\ ]]_{\mathcal{T}} \quad \text{by definition of } u'_i \text{ since z is not free in } v[y/x]$$

$$= [[\ (\ (\lambda y.(v[y/x]))[u_1/x_1,\dots,u_n/x_n]\ w\ )\ ]]_{\mathcal{T}} \quad \text{since y is not free in } u_i$$

$$= [[\ (\ (\lambda x.v)[u_1/x_1,\dots,u_n/x_n]\ w\ )\ ]]_{\mathcal{T}} \quad \text{renaming } \lambda y \text{ to } \lambda x \text{ by } (\alpha)$$

$$= (\Phi([[\ (\lambda x.v)[\rho]\ ]]_{\mathcal{T}}))([[w]]_{\mathcal{T}}) \text{ by definition of } [\rho] \text{ and } \Phi.$$

Therefore, $\lambda d \in \mathbf{D}.\mathcal{V}_{\mathcal{E}}[v](\rho\{d/x\}) = \Phi((\lambda x.v)[\rho])$, and the claim is proved.

Now if $\mathcal{E} \models u = v$, then in the particular environment ρ_0 such that $\rho_0(x) = [[x]]_{\mathcal{T}}$ for all variables x, the terms u and v have the same value. By the above claim, the value of u is $[[u]]_{\mathcal{T}}$ and that of v is $[[v]]_{\mathcal{T}}$, so $[[u]]_{\mathcal{T}} = [[v]]_{\mathcal{T}}$. That is, $\vdash_{\mathcal{T}} u = v$.

Conversely, if $\vdash_{\mathcal{T}} u = v$, then $\vdash_{\mathcal{T}} u[\rho] = v[\rho]$ for all ρ since simultaneous substitution preserves equations. The above claim immediately implies that $\mathcal{S} \models u = v$. ∎

4. **Combinatory Models.**

The notion of environment model may best reflect the intuitively correct way to assign values to terms of the lambda calculus, but it is mathematically a bit awkward. The condition that all the functions f arising in (3.6) be in $\mathbf{D} \to \mathbf{D}$ obviously defines some kind of closure condition on this set of functions, but the formulation of the condition is so entangled with the syntax of lambda terms that it is hard to visualize what models look like, and it can be awkward to verify that particular functional domains are indeed models.

Is there some way to define the closure conditions implicit in (3.6) without reference to the syntactic machinery of lambda terms? Not surprisingly, a solution lies in considering combinators, which were originally devised to short-circuit the syntactic complexities of variables in terms. *Combinators* are simply variable free terms over combinatory algebras.

The key property which motivates the rather odd definition of combinatory algebra given in the introduction is revealed by considering the more natural notion of combinatory completeness defined below.

Definition. Let $\cdot$ be a binary operation on a set $\mathbf{D}$. The structure $\mathcal{C} = \langle \mathbf{D}, \cdot \rangle$ is *combinatorially complete* iff for every $\mathcal{C}$-term u and every finite sequence $x_1,\ldots,x_n$ of variables, there is a $\mathcal{C}$-term u' not containing $x_1,\ldots,x_n$ and such that $\mathcal{C} \models u = u'x_1 \ldots x_n$.

Definition. Let I abbreviate the combinator SKK. For every $\mathcal{C}$-term u and variable x define a new $\mathcal{C}$-term $\langle x \rangle u$ as follows:

$\langle x \rangle u = Ku$ if x is does not occur in u,
$\langle x \rangle x = I$,
$\langle x \rangle (uv) = S(\langle x \rangle u)(\langle x \rangle v)$ if x does occur in u or v.

Combinatory Completeness Lemma (Curry). A structure $\mathcal{C} = \langle \mathbf{D}, \cdot \rangle$ is combinatorially complete iff it is a combinatory algebra.

Proof.(←) Let $\mathcal{C}$ be a combinatory algebra and K,S ∈ D satisfy (1.1) and (1.2). It follows directly from the definitions that x does not occur in ⟨x⟩u, and that $\mathcal{C} \models u = (\langle x \rangle u)x$. Let u′ be $\langle x_1 \rangle \ldots \langle x_n \rangle u$. Then $\mathcal{C} \models u = u' x_1 \ldots x_n$ as required.

(→) Left to the reader. ∎

We now show that the simple definition of combinatory models given at the outset provides the desired algebraic characterization of environment models.

Definition. Let $\mathcal{C} = \langle \mathbf{D}, \cdot, \epsilon \rangle$ be a combinatory model. Let Φ map elements of **D** into the functions from **D** to **D** defined by left multiplication. That is, let $(\Phi(\mathbf{d}_0))(\mathbf{d}) = \mathbf{d}_0 \cdot \mathbf{d}$. The *functional domain associated with* $\mathcal{C}$ is $\langle \mathbf{D}, \Phi, \Psi \rangle$ where Ψ is given by the rule $\Psi(\Phi(\mathbf{d})) = \epsilon \cdot \mathbf{d}$.

Note that (1.3b) implies that Ψ is well defined, and (1.3a) implies that Ψ is a right inverse of Φ, so that ⟨D,Φ,Ψ⟩ is indeed a functional domain.

Definition. Let $\mathcal{E} = \langle \mathbf{D}, \Phi, \Psi \rangle$ be an environment model. Define a binary operation · on **D** by the rule $\mathbf{d}_0 \cdot \mathbf{d}_1 = (\Phi(\mathbf{d}_0))(\mathbf{d}_1)$, and let $\epsilon = \mathcal{V}_{\mathcal{E}}[\lambda xy.xy]\rho$. The *algebra associated with* $\mathcal{E}$ is ⟨D,·,ϵ⟩.

Note that by the Free Variable Lemma, the value of ϵ does not depend on the environment ρ.

Combinatory Model Theorem. (i) The functional domain $\mathcal{E}$ associated with a combinatory model $\mathcal{C}$ is an environment model which assigns the same values to $\mathcal{C}$-terms. That is,

$$\mathcal{V}_{\mathcal{C}}[u] = \mathcal{V}_{\mathcal{E}}[u]$$

for all $\mathscr{C}$-terms u.

(ii) The algebra associated with an environment model is a combinatory model.

(iii) The associations between combinatory models and environment models defined above are inverses of each other. That is, if $\mathscr{E}$ is the environment model associated with a combinatory model $\mathscr{C}$, then $\mathscr{C}$ is the combinatory model associated with $\mathscr{E}$, and *vice versa.*

Proof. (i) Let $\bar{x}_n$ abbreviate the sequence of distinct variables $x_1,\ldots,x_n$. Let $\bar{\mathbf{d}}_n$ abbreviate the sequence $\mathbf{d}_1,\ldots,\mathbf{d}_n$ of (not necessarily distinct) elements in **D**, and let $\rho\{\bar{\mathbf{d}}_n/\bar{x}_n\}$ abbreviate $(\ldots(\rho\{\mathbf{d}_1/x_1\})\ldots\{\mathbf{d}_n/x_n\})$. We claim that for every term $u \in \Lambda(\mathbf{D})$, for every environment ρ, and all $\bar{x}_n$, there is an element $\mathbf{d}_{u\rho\bar{x}_n} \in \mathbf{D}$ such that

$$\mathscr{V}_{\mathscr{E}}[u](\rho\{\bar{\mathbf{d}}_n/\bar{x}_n\}) = \mathbf{d}_{u\rho\bar{x}_n}\bar{\mathbf{d}}_n$$

for all $\bar{\mathbf{d}}_n \in \mathbf{D}$.

This claim follows by induction on the definition of lambda terms. We give the details only for the most difficult case that u is of the form $\lambda x_{n+1}.v$ where x_{n+1} is distinct from $\bar{x}_n$. We have,

$$\mathscr{V}_{\mathscr{E}}[\lambda x_{n+1}.v](\rho\{\bar{\mathbf{d}}_n/\bar{x}_n\})$$

$$= \Psi(\ \lambda\mathbf{d}_{n+1}\in\mathbf{D}.\mathscr{V}_{\mathscr{E}}[v]((\rho\{\bar{\mathbf{d}}_n/\bar{x}_n\})\{\mathbf{d}_{n+1}/x_{n+1}\})\ ) \text{ by (3.6)}$$

$$= \Psi(\ \lambda\mathbf{d}_{n+1}\in\mathbf{D}.\mathscr{V}_{\mathscr{E}}[v](\rho\{\bar{\mathbf{d}}_{n+1}/\bar{x}_{n+1}\})\ )$$

$$= \Psi(\ \lambda\mathbf{d}_{n+1}\in\mathbf{D}.\mathbf{d}_{v\rho\bar{x}_{n+1}}\bar{\mathbf{d}}_{n+1}\ ) \text{ by induction hypothesis}$$

$$= (\epsilon(\mathbf{d}_{v\rho\bar{x}_{n+1}}\bar{\mathbf{d}}_n)) \text{ by definition of } \Psi.$$

By combinatory completeness, there is a $\mathbf{d} \in \mathbf{D}$ such that

$$\mathscr{C} \models (\epsilon(\mathbf{d}_{v\rho\bar{x}_{n+1}}\bar{x}_n)) = \mathbf{d}\bar{x}_n,$$

so we define $\mathbf{d}_{u\rho\bar{x}_n}$ to be $\mathbf{d}$.

The claim immediately implies that $\lambda\mathbf{d}\in\mathbf{D}.\mathscr{V}_{\mathscr{E}}[u](\rho\{\mathbf{d}/x\}) = \Phi(\mathbf{d}_{u\rho x}) \in \mathbf{D} \to \mathbf{D}$, so that the functional domain $\mathscr{E}$ is an environment model.

A trivial induction on the definition of $\mathcal{C}$-terms establishes that $\mathcal{V}_{\mathcal{E}}$ and $\mathcal{V}_{\mathcal{C}}$ coincide on $\mathcal{C}$-terms.

(ii) Let $\mathcal{E} = \langle \mathbf{D},\Phi,\Psi\rangle$ be an environment model and $\mathcal{C}$ its associated algebra. Choose K, S $\in$ **D** to be $\mathcal{V}_{\mathcal{E}}[\lambda xy.x]$ and $\mathcal{V}_{\mathcal{E}}[\lambda xyz.xz(yz)]$, respectively. Then (1.1), (1.2), (1.3a,c) follow directly from the Soundness Theorem, (β), and the definitions. To verify (1.3b), note that

$$\begin{aligned}
\epsilon\cdot\mathbf{d}_0 &= (\Phi(\mathcal{V}_{\mathcal{E}}[\lambda xy.xy]))(\mathbf{d}_0) && \text{by definition of } \Phi \text{ and } \epsilon \\
&= (\lambda \mathbf{d}\in\mathbf{D}.\mathcal{V}_{\mathcal{E}}[\lambda y.xy](\rho\{\mathbf{d}/x\}))(\mathbf{d}_0) && \text{by (3.4) and (3.6)} \\
&= \mathcal{V}_{\mathcal{E}}[\lambda y.xy](\rho\{\mathbf{d}_0/x\}) \\
&= \Psi(\lambda \mathbf{d}\in\mathbf{D}.\mathcal{V}_{\mathcal{E}}[(xy)]((\rho\{\mathbf{d}_0/x\})\{\mathbf{d}/y\})) && \text{by (3.6)} \\
&= \Psi(\lambda \mathbf{d}\in\mathbf{D}.(\Phi(\mathbf{d}_0))(\mathbf{d})) && \text{by (3.2), (3.4), (3.5)} \\
&= \Psi(\Phi(\mathbf{d}_0)).
\end{aligned}$$

Hence, if $\mathbf{d}_0\cdot\mathbf{d} = \mathbf{d}_1\cdot\mathbf{d}$ for all $\mathbf{d}$, then $\Phi(\mathbf{d}_0) = \Phi(\mathbf{d}_1)$ by definition of Φ, so $\epsilon\cdot\mathbf{d}_0 = \Psi(\Phi(\mathbf{d}_0)) = \Psi(\Phi(\mathbf{d}_1)) = \epsilon\cdot\mathbf{d}_1$. This proves (1.3b) holds in $\mathcal{C}$.

(iii) Let $\mathcal{E} = \langle \mathbf{D},\Phi,\Psi\rangle$ be an environment model, $\mathcal{C} = \langle \mathbf{D},\cdot,\epsilon\rangle$ the associated combinatory model, $\mathcal{E}' = \langle \mathbf{D},\Phi',\Psi'\rangle$ the environment model associated with $\mathcal{C}$, and $\mathcal{C}'$ the combinatory model associated with $\mathcal{E}'$. Clearly, $\Phi = \Phi'$. But $\Psi'(\Phi(\mathbf{d}_0)) = \epsilon\cdot\mathbf{d}_0$ by definition of Ψ', and $\epsilon\cdot\mathbf{d}_0 = \Psi(\Phi(\mathbf{d}_0))$ by the proof of (ii) above, so $\Psi = \Psi'$. Hence, $\mathcal{E} = \mathcal{E}'$. The proof that $\mathcal{C} = \mathcal{C}'$ follows similarly. ∎

To state the relation between the values of lambda terms in an environment model and $\mathcal{C}$-terms in the associated combinatory model, we establish for lambda terms a result corresponding to combinatorial completeness. Let K_λ abbreviate $\lambda xy.x$, S_λ abbreviate $\lambda xyz.xz(yz)$, and I_λ abbreviate $((S_\lambda K_\lambda)K_\lambda)$. Define *combinatory lambda terms* inductively to be lambda terms which are either constants, variables, K_λ, S_λ, or applications of combinatory lambda terms. In other words, combinatory lambda terms are precisely those lambda terms in which the only abstractions occurring are in K_λ and S_λ.

Combinatory Lambda Term Lemma. For every $u \in \Lambda(C)$ there is a combinatory lambda term $u^{(\lambda)} \in \Lambda(C)$ such that u converts to $u^{(\lambda)}$.

Proof. For any *combinatory* lambda term $u \in \Lambda(C)$ and variable x, define $\langle\langle x\rangle\rangle u \in \Lambda(C)$ as follows:

$$\langle\langle x\rangle\rangle u = (K_\lambda u) \quad \text{if } x \text{ is not free in } u,$$
$$\langle\langle x\rangle\rangle x = I_\lambda,$$
$$\langle\langle x\rangle\rangle(uv) = ((S_\lambda \langle\langle x\rangle\rangle u)\langle\langle x\rangle\rangle v) \quad \text{if } x \text{ is free in } u \text{ or } v.$$

It follows by simple calculation from the definitions that $\langle\langle x\rangle\rangle u$ is a combinatory lambda term which converts to $\lambda x.u$.

Let $u^{(\lambda)}$ to be the combinatory lambda term obtained from an *arbitrary* lambda term u by replacing all occurrences of λx by $\langle\langle x\rangle\rangle$. Inductively, we may define $u^{(\lambda)}$ as follows:

$$\mathbf{d}^{(\lambda)} = \mathbf{d}, \quad x^{(\lambda)} = x, \quad (uv)^{(\lambda)} = (u^{(\lambda)} v^{(\lambda)}), \quad (\lambda x.u)^{(\lambda)} = \langle\langle x\rangle\rangle(u^{(\lambda)}). \quad \blacksquare$$

Definition. Let $\mathscr{C} = \langle \mathbf{D}, \cdot, \mathbf{e}\rangle$ be a combinatory model. For any $u \in \Lambda(\mathbf{D})$, define $u^{(\mathscr{C})}$ be the $\mathscr{C}$-term obtained by replacing all occurrences of K_λ and S_λ in $u^{(\lambda)}$ by the constant values in $\mathbf{D}$ of K_λ and S_λ in the environment model associated with $\mathscr{C}$.

Note that by the Free Variable Lemma, the values of the closed terms K_λ and S_λ are determined by the environment model alone and not by any particular choice of environment. So $u^{(\mathscr{C})}$ is well defined.

Combinatory Model Theorem. (iv) Let $\mathscr{E}$ be an environment model and $\mathscr{C}$ its associated combinatory model. Then for all lambda terms u,

$$\mathscr{V}_{\mathscr{E}}[u] = \mathscr{V}_{\mathscr{E}}[u^{(\mathscr{C})}].$$

Proof. $\mathscr{V}_{\mathscr{E}}[u] = \mathscr{V}_{\mathscr{E}}[u^{(\lambda)}]$ by Soundness since u converts to $u^{(\lambda)}$. The proof now follows immediately by induction on the number of occurrences of free variables, constants, and terms K_λ and S_λ in $u^{(\lambda)}$. $\blacksquare$

The Combinatory Model Theorem demonstrates that combinatory models and environment models are merely notational variants of the same class of mathematical structures.

The significance of the these results is that we can now straightforwardly interpret arbitrary lambda terms and equations between them as though they were standard terms and equations over an ordinary algebraic structure defined by first order axioms.

5. An Elementary Model Construction.

We now present a simple construction of a class of combinatory models using only elementary properties of sets.

Let A be any nonempty set, and let B be the least set containing A and all ordered pairs consisting of a finite subset $\beta \subseteq B$ and an element $b \in B$. Such an ordered pair is denoted $(\beta \to b)$. Assume that elements of A are distinguishable from ordered pairs.

Let $\mathbf{D}_A = 2^B = \{\mathbf{d} \mid \mathbf{d} \subseteq B\}$, and define the binary operation $\cdot$ on $\mathbf{D}_A$ by the rule

(5.1) $\mathbf{d}_1 \cdot \mathbf{d}_2 = \{b \in B \mid (\beta \to b) \in \mathbf{d}_1 \text{ for some } \beta \subseteq \mathbf{d}_2\}$.

Model Existence Theorem [Plotkin 72].[6] The structure $\langle \mathbf{D}_A, \cdot, \epsilon \rangle$ is a combinatory model where

(5.2) $\epsilon = \{ (\alpha \to (\beta \to b)) \mid \alpha, \beta$ finite subsets of B and $b \in \alpha \cdot \beta\}$.[7]

Proof. Choose

(5.3) $K = \{ (\alpha \to (\beta \to b)) \mid b \in \alpha\}$, and

(5.4) $S = \{ (\alpha \to (\beta \to (\gamma \to b))) \mid b \in \alpha \cdot \gamma \cdot (\beta \cdot \gamma)\}$

The proof that ϵ, K, and S given by (5.2-4) satisfy (1.1), (1.2), (1.3a,c) is a direct consequence of the definitions and is omitted. To verify (1.3b), note that $A \cap \epsilon\mathbf{d}_0 = \varnothing$ and that $b \in \mathbf{d}_0 \cdot \beta$ iff $(\beta \to b) \in \epsilon\mathbf{d}_0$ by definition. Hence, if $\epsilon\mathbf{d}_0 - \epsilon\mathbf{d}_1 \neq \varnothing$, then $(\beta \to b) \in \epsilon\mathbf{d}_0 - \epsilon\mathbf{d}_1$ for some $(\beta \to b)$, so $b \in \mathbf{d}_0 \cdot \beta - \mathbf{d}_1 \cdot \beta$. ∎

The model Existence Theorem validates the assumption made in proving the Consistency Theorem in Section 3 that nontrivial models exist. Indeed, $\mathbf{D}_A$ has uncountably many elements. There are also countably infinite models, e.g., the recursively enumerable elements of $\mathbf{D}_A$ form a countable submodel. (The Lambda Algebra Theorem in Section 7 provides another mechanism for constructing a nontrivial countable model from any nontrivial combinatory model.)

Now let $\times$ be any binary operation on the set A. Let functions f_n and f from A to $\mathbf{D}_A$ be defined as follows:

$$(5.5)\quad f_0(a) = \{a\},$$
$$f_{n+1}(a) = f_n(a) \cup \{ (\{a'\}\to b) \mid a' \in A,\ b \in f_n(a\times a')\},$$
$$f(a) = \bigcup_{n\geq 0} f_n(a).$$

Embedding Theorem [Engeler 79]. The function f given in (5.5) isomorphically embeds the structure $\langle A,\times\rangle$ into $\langle \mathbf{D}_A,\cdot\rangle$.

Proof. Note that $f(a) \cap A = \{a\}$, so f is injective. The verification that f is a homomorphism is a routine calculation which we omit (*cf.* [Engeler 79]). ∎

To illustrate the significance of the Embedding Theorem, we can now make sense of the examples involving integers, polynomials, and triple composition given in the Introduction. For example, to obtain the piecewise integer polynomials as part of a combinatory model, let A be the least set containing the integers $\mathbb{Z}$ and distinct new elements **add**, $\mathbf{add}_n$, **mult**, $\mathbf{mult}_n$, **cond**, $\mathbf{cond}_a$, $\mathbf{cond}_{a,a'}$ for $n \in \mathbb{Z}$, $a,a' \in A$. Define a binary operation $\times$ on A by the rules

$$\mathbf{add}\times n = \mathbf{add}_n,\quad \mathbf{add}_n\times m = n+m,$$
$$\mathbf{mult}\times n = \mathbf{mult}_n,\quad \mathbf{mult}_n\times m = nm,$$
$$\mathbf{cond}\times a = \mathbf{cond}_a,\quad \mathbf{cond}_a\times a' = \mathbf{cond}_{a,a'},$$
$$\mathbf{cond}_{a,a'}\times a'' = a' \text{ if } a \in \mathbb{N}, \text{ otherwise } a'',$$

for $n,m \in \mathbb{Z}$, and $a,a',a'' \in A$; the operation $\times$ may be defined arbitrarily on arguments not specified above.

Now embedding $\langle A,\times\rangle$ into $\mathbf{D}_A$ yields (an isomorphic copy of) $\mathbb{Z}$ along with addition, multiplication, and conditionals. The triple composition functional T of the Introduction also appears in $\mathbf{D}_A$, since $T = \lambda fx.f(f(fx))$ is defined by a pure lambda expression and so can be interpreted in $\mathbf{D}_A$ without even appealing to the Embedding Theorem. The reader is invited to consider why the difficulties surrounding the paradoxical functional P no longer threaten.

For construction of extensional models, see [Scott 80a,§5] who sketches an elementary construction of an embedding theorem into extensional models based on a modification of $\mathbf{D}_A$. [Scott 80c] provides another construction of extensional embeddings based on a universal embedding property for the $\mathbf{P}\omega$ model. See [Wadsworth 77] for a detailed treatment of Scott's original construction of extensional models from continuous lattices.

6. **Another Algebraic Axiomatization: Lambda Models.** The definition of combinatory model connects nicely with the definition of environment model, but suffers the small technical disadvantage that the elements K, S are not identified uniquely. For example, $\{(\{b\}\to(\emptyset\to b)) \mid b \in B\}$ is another element in $\mathbf{D}_A$ distinct from the K of (5.3) which satisfies (1.1). In order to maintain the correspondence between the algebraic structure of the model and the values of lambda terms, it is important to make the appropriate choice of K and S, namely as the values of K_λ and S_λ in the associated environment model. It is an amusing exercise to describe these values in a purely algebraic way.

Let $\mathcal{C}$ be a combinatory model, $\mathcal{E}$ its associated environment model, and begin by choosing *any* K, S satisfying (1.1) and (1.2). Let

$$(6.1)\quad B = S(KS)K$$

be the "composition" combinator. It is easy to verify that $Bxyz = x(yz)$ is valid in any combinatory algebra. Let

$$(6.2)\quad \epsilon_2 = (B\cdot\epsilon)\cdot(B\cdot\epsilon).$$

We now have

$$(6.3)\quad \mathcal{V}_{\mathcal{E}}[K_\lambda] = \epsilon_2\cdot K,$$

because

$$\mathcal{V}_{\mathcal{E}}[\lambda xy.x]\rho = \Psi(\lambda d.\mathcal{V}_{\mathcal{E}}[\lambda y.x]\rho^d_x) = \Psi(\lambda d.\Psi(\lambda e.d)) = \Psi(\lambda d.\Psi(\Phi(K\cdot d))) =$$
$$= \Psi(\lambda d.\epsilon\cdot(K\cdot d)) = \Psi(\Phi((B\cdot\epsilon)\cdot K)) = \epsilon\cdot((B\cdot\epsilon)\cdot K) = ((B\cdot\epsilon)\cdot(B\cdot\epsilon))\cdot K.$$

Note that as predicted by the Free Variable Lemma, the value of the closed term K_λ is determined by the environment model $\mathcal{E}$ alone and *not* by any particular choice of ρ, K, or S.

Letting

$$(6.4)\quad \epsilon_3 = (B\cdot\epsilon)\cdot(B\cdot\epsilon_2),$$

a similar calculation shows that

$$(6.5)\quad \mathcal{V}_{\mathcal{E}}[S_\lambda] = \epsilon_3\cdot S.$$

Definition. ([Scott 80b], [Barendregt 81]) A *lambda model* is an algebra $\langle D,\cdot,K,S\rangle$ such that

K, S $\in$ **D** satisfy (1.1), (1.2),

$\langle D,\cdot,\epsilon\rangle$ is a combinatory model where $\epsilon = S(KI)$,

(6.6) $K = \epsilon_2\cdot K$, and

(6.7) $S = \epsilon_3\cdot S$,

where ϵ_2, ϵ_3 are given by (6.2) and (6.4).

Because the righthand sides of (6.6) and (6.7) are the values of K_λ and S_λ in the environment model associated with any combinatory model $\langle D,\cdot,\epsilon\rangle$, they are uniquely determined independently of the particular choice of K and S.[8] Conversely, the values of K_λ and S_λ determine ϵ because ϵ is the value of $\lambda xy.xy$, and $\lambda xy.xy$ converts to $S_\lambda(K_\lambda I_\lambda)$. Thus we have established the

Lambda Model Theorem. Any combinatory model $\langle D,\cdot,\epsilon\rangle$ uniquely determines a lambda model $\langle D,\cdot,\epsilon_2\cdot K,\epsilon_3\cdot S\rangle$, independently of the choice of K and S $\in$ **D** satisfying (1.1-2).

Conversely, if $\langle D,\cdot,K,S\rangle$ is a lambda model, then $\langle D,\cdot,S(KI)\rangle$ is a combinatory model. Moreover, these two correspondences are inverses of each other.

The axioms for lambda models are a bit more elaborate than for combinatory models, but lambda models have the advantage that the standard algebraic notion of a *substructure* relates nicely to certain syntactic properties of lambda terms. For example, the *interior* of a lambda calculus model is normally defined as the values of the pure, i.e. constant free, *closed* lambda terms [Barendregt 76]. In terms of lambda models, the interior now has a familiar algebraic definition as the minimum subalgebra of a lambda model; this follows immediately from the Combinatory Lambda Term Lemma, (6.3), and (6.5).

It might seem that we are now ready to develop a nice theory of models using the usual algebraic notions of substructures, morphisms, etc. One serious technical impediment remains, however. Neither the class of combinatory models nor the class of lambda models is closed under the operation of taking substructures or of applying morphisms with respect to the binary operation $\cdot$!

The difficulty springs from the fact that the first order axioms (1.3b) and (1.4) are not equations. *Equationally* axiomatized structures are guaranteed to be closed under taking substructures and morphisms, but first order axiomatizable structures are not in general [*cf.* Monk 76,§24].

In fact the combinatory algebra which is the interior of the extensional term model is not extensional [Plotkin 74], i.e. it does not satisfy (1.4), and is not even expandable by any choice of K, S into a (not necessarily extensional) lambda model [Barendregt, letter to Meyer, Oct. 1980]. This implies among other things that there is no purely equational definition of lambda models since equationally defined classes of structures are closed under taking substructures.

Nevertheless, there is an equationally definable class of structures called lambda algebras which serve so well for interpreting lambda terms that it is tempting to identify them as the proper algebraic embodiment of lambda calculus [*cf.* Lambek 80]. We consider these next.

7. **Lambda Algebras.** The mapping from u to $u^{(\lambda)}$ given in Section 4 suggests an obvious way to interpret lambda terms within arbitrary combinatory *algebras* -- by replacing K_λ in $u^{(\lambda)}$ by the constant $K \in \mathbf{D}$ and similarly for S_λ. If the combinatory algebra is a combinatory *model*, then the content of the Combinatory Model Theorem (iv) is that this idea indeed works. However, in an arbitrary algebra replacing K_λ by some K satisfying (1.1) may cause problems because there may be no K which behaves completely like K_λ with respect to convertibility. For example, since K_λ converts to

$(K_\lambda)^{(\lambda)} = \langle\langle x\rangle\rangle(\langle\langle y\rangle\rangle x) = S_\lambda(K_\lambda K_\lambda)I_\lambda$, their interpretations in the combinatory algebra should agree, but there is no guarantee that the algebra will contain K,S such that K has the same value as $\langle x\rangle(\langle y\rangle x) = S(KK)I$. In general, the problem is to guarantee the existence of K, S in the algebra such that the analogue of rule (ξ) holds, namely, if $u = v$, then $\langle x\rangle u = \langle x\rangle v$. To do this, some slightly weaker conditions than those for a combinatory model suffice.

Definition.(Curry, [*cf.* Barendregt 81,CH.7]) A *Lambda Algebra* is a structure $\langle \mathbf{D},\cdot,K,S\rangle$ where $\langle \mathbf{D},\cdot\rangle$ is a combinatory algebra, $K,S \in \mathbf{D}$ satisfy (1.1), (1.2), and

(7.1) $K = \langle x\rangle(\langle y\rangle(Kxy))$,

(7.2) $S = \langle x\rangle(\langle y\rangle(\langle z\rangle(Sxyz)))$,

(7.3) $\langle x\rangle(\langle y\rangle(S(Kx)(Ky))) = \langle x\rangle(\langle y\rangle(K(xy)))$,

(7.4) $\langle x\rangle(\langle y\rangle(S(S(KK)x)y)) = \langle x\rangle(\langle y\rangle(\langle z\rangle(xz)))$,

(7.5) $\langle x\rangle(\langle y\rangle(\langle z\rangle(S(S(S(KS)x)y)z) = \langle x\rangle(\langle y\rangle(\langle z\rangle(S(Sxz)(Syz))$.

Note that (7.1-5) denote equations between constants. For example, (7.1) in less abbreviated form reads

$$K = S(S(KS)(S(KK)(S(KK)I)))(KI)$$

which would be even longer if we had expanded the combinator I as SKK and put in full parenthesization. The reader will appreciate the utility of the abbreviations.[9] Even with the abbreviations, (7.1-5) are hardly memorable having been chosen solely for the purpose of carrying out the proofs below.

In the following lemmas we develop some of the elementary properties of the transform $\langle x\rangle$ on combinatory terms. These properties will imply that combinatory models can be obtained from lambda algebras simply by extending lambda algebras with indeterminates -- just as the ring of integer multivariate polynomials is obtained from the ring of integers. This result then yields a mathematically robust characterization of lambda algebras as substructures and homomorphic images of lambda models.

Let $\mathcal{C} = \langle D,\cdot\rangle$ be a combinatory algebra, and $K,S \in D$ be any elements satisfying (1.1-2).

Let X be a set of variables and $\mathcal{C}[X]$ be the free combinatory algebra generated by X over the constants in $\mathcal{C}$. That is, $\mathcal{C}[X]$ is the free word algebra of $\mathcal{C}$-terms with variables only from X, modulo the congruence relation on $\mathcal{C}$-terms generated by the equations between constant terms valid in $\mathcal{C}$ and all substitution instances of (1.1-2).

Formally, let u,v,w range over $\mathcal{C}$-terms, and define the *proof system of β-Combinatory Logic* for $\mathcal{C}$ to have axioms

$u = v$ such that u,v are variable free terms and $\mathcal{C} \models u = v$,

$Kuv = u$, and

$Suvw = uw(vw)$,

and inference rules: (transitivity and symmetry) and (congruence). (We would also insist on the axiom (reflexivity) except that it follows already from $Kuv = u$ and (transitivity and symmetry).)

Write $\mathcal{C}\text{-}CL_\beta \vdash u = v$ iff the equation $u = v$ is provable in this system, and let

$$[[u]] = \{v \mid \mathcal{C}\text{-}CL_\beta \vdash u = v\},$$

$$D[X] = \{\, [[u]] \mid u \in \Lambda(D) \text{ and all variables in } u \text{ are in } X\}.$$

Then

$$\mathcal{C}[X] = \langle D[X],\cdot\rangle \text{ where } [[u]]\cdot[[v]] = [[(uv)]].$$

(7.6)*Lemma.* $\mathcal{C}[X]$ is a combinatory algebra and the mapping taking $\mathbf{d} \in D$ to $[[\mathbf{d}]]$ isomorphically embeds $\mathcal{C}$ into $\mathcal{C}[X]$. Moreover, if u,v are $\mathcal{C}$-terms all of whose variables are in the set X, then

$$\mathcal{C}[X] \models u = v \quad \text{iff} \quad \mathcal{C}\text{-}CL_\beta \vdash u = v.$$

Proof. The construction of $\mathcal{C}[X]$ from $\mathcal{C}$ is the standard one for constructing a "polynomial" algebra from any equationally defined algebra. ∎

Lemma (7.6) justifies identifying **d** $\in$ **D** with the element [[**d**]] of **D**[X] which we shall continue to do. Note that because (7.1-5) denote equations between variable free $\mathcal{C}$-terms, it now follows immediately that $\mathcal{A}$[X] satisfies whichever of (7.1-5) that $\mathcal{C}$ satisfies. In particular, if $\mathcal{C}$ is a lambda algebra, then so is $\mathcal{A}$[X].

(7.7)*Lemma.* For any $\mathcal{C}$-term u and variables x,y,

(i) x does not occur in $\langle x\rangle u$,

(ii) if y does not occur in u, then $\langle x\rangle u = \langle y\rangle(u[y/x])$.

Proof. By induction on the definition of $\langle x\rangle$. ∎

(7.8)*Lemma.* For all $\mathcal{C}$-terms u,v and distinct variables x,y, if x does not occur in v, then

$$(\langle x\rangle u)[v/y] = \langle x\rangle(u[v/y]).$$

Proof. By induction on the definition of $\langle x\rangle$. The cases that u = x or x does not occur in u are trivial.

Suppose $u = (u_1 u_2)$ and x occurs in u. Then

$$(\langle x\rangle u)[v/y] = (S(\langle x\rangle u_1)(\langle x\rangle u_2))[v/y] \quad \text{by definition of } \langle x\rangle$$

$$= S(\ (\langle x\rangle u_1)[v/y]\ )(\ (\langle x\rangle u_2)[v/y]\ ) \quad \text{by definition of } [v/y]$$

$$= S(\ \langle x\rangle(u_1[v/y])\ )(\ \langle x\rangle(u_2[v/y])\ ) \quad \text{by induction hypothesis}$$

$$= \langle x\rangle(\ u_1[v/y]\ u_2[v/y]\ ) \quad \text{by definition of } \langle x\rangle$$

$$= \langle x\rangle(u[v/y]) \quad \text{by definition of } [v/y].$$ ∎

(7.9)*Lemma.* For any $\mathcal{C}$-terms u,v and variable x, $\mathcal{C} \models ((\langle x\rangle u)v) = u[v/x]$.

Proof. As already observed in the Combinatory Completeness Lemma, an induction on the definition of $\langle x\rangle$ implies $\mathcal{C} \models (\langle x\rangle u)x = u$. Since substitution preserves validity of equations, $\mathcal{C} \models ((\langle x\rangle u)x)[v/x] = u[v/x]$, but by (7.7(i)) x does not occur in $\langle x\rangle u$, so $((\langle x\rangle u)x)[v/x] = ((\langle x\rangle u)v)$. ■

(7.10)*Lemma.* If K,S satisfy (7.1-2), then for all distinct variables x,y,z, and $\mathcal{C}$-terms u

(i) $\mathcal{C} \models \langle y\rangle(Kxy) = Kx$,

(ii) $\mathcal{C} \models \langle z\rangle(Sxyz) = Sxy$,

(iii) $\mathcal{C} \models \langle y\rangle(\ (\langle x\rangle u)y\ ) = \langle x\rangle u$ if y does not occur in $\langle x\rangle u$.

Proof.(i) $\mathcal{C} \models \langle y\rangle(\ Kxy\ ) = (\langle x\rangle(\langle y\rangle(\ Kxy\ )))\ x$ by (7.9)

$= Kx$ by (7.1).

(ii) $\mathcal{C} \models \langle z\rangle(\ Sxyz\ ) = (\langle x\rangle(\langle y\rangle(\langle z\rangle(\ Sxyz\ ))))\ x\ y$ by (7.9) twice

$= Sxy$ by (7.2).

(iii) By definition $\langle x\rangle u$ is always of the form Kv or Svw. In the first case,

$\langle x\rangle u = Kv = (Kx)[v/x]$, but

$\mathcal{C} \models (Kx)[v/x] = (\langle y\rangle(Kxy))[v/x]$ by (i),

$= \langle y\rangle((Kxy)[v/x])$ by (7.8) providing y does not occur in v

$= \langle y\rangle(Kvy) = \langle y\rangle(\ (\langle x\rangle u)y\ )$.

The case u = Svw follows similarly from (ii). ■

(7.11)*Lemma.* If K,S satisfy (7.3), then for all $\mathcal{C}$-terms u,v and variables x,

$\mathcal{C} \models \langle x\rangle(uv) = S(\langle x\rangle u)(\langle x\rangle v)$.

Proof. If x occurs in (uv), then the equation is identically true, so assume x does *not* occur in (uv). Let y,z be distinct variables not equal to x and not occurring in (uv). Then

$\langle x\rangle(uv) = K(uv)$ by definition of $\langle x\rangle$

$= K(uy)[v/y]$ since y does not occur in u. But

$\mathscr{C} \models K(uy)[v/y]$

$= (\langle y\rangle(K(uy))\ v)$ by (7.9)

$= (\ \langle y\rangle(\ (K(xy))[u/x]\ )\ v)$

$= (\ (\langle y\rangle(K(xy)))[u/x]\ v)$ by (7.8) since y does not occur in u

$= (\langle x\rangle(\langle y\rangle(\ K(xy)\ )))\ u\ v$ by (7.9)

$= (\langle x\rangle(\langle y\rangle(\ S(Kx)(Ky)\ )))\ u\ v$ by (7.3)

$= (\ (\langle y\rangle(\ S(Kx)(Ky)\ ))[u/x]\ v)$ by (7.9)

$= (\ \langle y\rangle(\ (S(Kx)(Ky))[u/x]\ )\ v)$ by (7.8) since y does not occur in u

$= (\ \langle y\rangle(\ S(Ku)(Ky)\ )\ v)$ by substitution

$= (\ S(Ku)(Ky)\ )[v/y]$ by (7.9)

$= S(Ku)(Kv)$ since y does not occur in u

$= S(\langle x\rangle u)(\langle x\rangle v)$ by definition of $\langle x\rangle$. ∎

(7.12)*Lemma.* Let $\mathscr{C}$ be a lambda algebra and u,v be $\mathscr{C}$-terms with variables only in X. If $\mathscr{C}[X] \models u = v$, then $\mathscr{C}[X] \models \langle x\rangle u = \langle x\rangle v$.

Proof. By (7.6) validity is the same as provability for equations between $\mathscr{C}$-terms u,v all of whose variables are in X. We proceed by induction on the length of the proof that $\mathscr{C}\text{-}CL_\beta\ u = v$.

If the proof is of length one, i.e., u = v is an axiom, then if u,v are variable free terms, the result is immediate. If $u = Ku_1u_2$ and $v = u_1$, then, noting that (7.11) holds for $\mathscr{C}[X]$ because by (7.6) $\mathscr{C}[X]$ is a combinatory algebra satisfying the same variable free equations as $\mathscr{C}$, we have $\langle x\rangle u = \langle x\rangle v$ because

$\mathcal{A}[X] \models \langle x \rangle (K u_1 u_2)$

$= S(\langle x \rangle (K u_1))(\langle x \rangle u_2)$ by (7.11)

$= S(S(KK)(\langle x \rangle u_1))(\langle x \rangle u_2)$ by (7.11)

$= (S(S(KK)(\langle x \rangle u_1))y)[\langle x \rangle u_2/y]$ where y is chosen not to occur in $\langle x \rangle u_1$

$= (\langle y \rangle (S(S(KK)(\langle x \rangle u_1))y)) \langle x \rangle u_2$ by (7.9)

$= (\langle y \rangle ((S(S(KK)x)y)[\langle x \rangle u_1/x])) \langle x \rangle u_2$

$= ((\langle y \rangle ((S(S(KK)x)y)))[\langle x \rangle u_1/x]) \langle x \rangle u_2$ by (7.8) since y is not in $\langle x \rangle u_1$

$= ((\langle x \rangle (\langle y \rangle ((S(S(KK)x)y)))) \langle x \rangle u_1 \langle x \rangle u_2$ by (7.9)

$= ((\langle x \rangle (\langle y \rangle (\langle z \rangle (xz)))) \langle x \rangle u_1 \langle x \rangle u_2$ by (7.4)

$= \langle z \rangle ((\langle x \rangle u_1)z)$ by (7.8-9) where z is chosen not to occur in $\langle x \rangle u_1$

$= \langle x \rangle u_1$ by (7.10(iii)).

The case that u = v is the axiom $Su_1u_2u_3 = u_1u_3(u_2u_3)$ follows similarly using (7.5). So (7.12) holds for the axioms of $\mathcal{C}$-CL_β.

If the last inference rule in the proof of u = v was (transitivity and symmetry), then (7.12) follows immediately by induction. If the last rule was (congruence), then $u = (u_1u_2)$, $v = (v_1v_2)$, and $\mathcal{A}[X] \models u_1 = v_1$, $u_2 = v_2$. Hence,

$\mathcal{A}[X] \models \langle x \rangle u = S(\langle x \rangle u_1)(\langle x \rangle u_2)$ by (7.11)

$= S(\langle x \rangle v_1)(\langle x \rangle v_2)$ by induction since substitution preserves validity

$= \langle x \rangle v$ by (7.11) ∎

A point of possible confusion about (7.12) is that it does not hold in $\mathcal{C}$ as opposed to $\mathcal{A}[X]$. That is, it may be that the equation u = v is valid in $\mathcal{C}$, but the equation $\langle x \rangle u = \langle x \rangle v$ is not. The source of the confusion is that while $\mathcal{A}[X] \models u = v$ implies $\mathcal{A} \models u = v$, the converse fails. (This frequently happens in classical algebras. For example, $x = x^2$ is valid in the ring $\mathbb{Z}_2$, but not in the polynomial ring $\mathbb{Z}_2[x]$.) The key property of $\mathscr{A}[X]$ required in the proof of (7.12) is the equivalence of validity and provability given by (7.6) which holds only for $\mathscr{C}$-terms all of whose variables are in X.

Lambda Algebra Theorem. If $\mathscr{C} = \langle \mathbf{D},\cdot,K,S\rangle$ is a lambda algebra and X is an infinite set of variables, then $\mathscr{C}_\lambda[X] = \langle \mathbf{D}[X],\cdot,K,S\rangle$ is a lambda model.

Proof. Let $\epsilon = \langle x\rangle(\langle y\rangle(\ xy\ ))$. We first observe that $\langle \mathbf{D}[X],\cdot,\epsilon\rangle$ is a combinatory model. To see this, note that by (7.8-9), $\epsilon\mathbf{d} = \langle y\rangle(\mathbf{dy})$ for all $\mathbf{d} \in \mathbf{D}[X]$. (1.3a) follows directly by another application of (7.9), and (1.3c) follows by (7.10(iii)).

To verify (1.3b), suppose $[[u]]\mathbf{d} = [[v]]\mathbf{d}$ for all $\mathbf{d}$ in $\mathbf{D}[X]$. Let $y \in X$ be a variable not in u,v; there is such a y since X is infinite. Then letting $\mathbf{d}$ be $[[y]]$, we have $[[uy]] = [[vy]]$, and so by (7.6) and (7.12), $[[\langle y\rangle(uy)]] = [[\langle y\rangle(vy)]]$. But by (7.8-9), $[[(\epsilon u)]] = [[\langle y\rangle(uy)]]$ and likewise with v in place of u, so $\epsilon[[u]] = \epsilon[[v]]$.

So $\langle \mathbf{D}[X],\cdot,\epsilon\rangle$ is a combinatory model. By the Lambda Model Theorem, $\langle \mathbf{D}[X],\cdot,\epsilon_2K,\epsilon_3S\rangle$ is a lambda model. But

$\mathscr{A}[X] \vDash \epsilon_2K = \epsilon((B\epsilon)K)$ by (6.1-2)

$= \langle x\rangle(B\epsilon Kx)$ by (7.9)

$= \langle x\rangle(\epsilon(Kx))$ by (6.1) and (7.12)

$= \langle x\rangle(\langle y\rangle(\ Kxy\ ))$ by (7.9) and (7.12)

$= K$ by (7.1),

and a similar calculation using (7.2) shows that $\mathscr{A}[X] \vDash \epsilon_3S = S$, so $\mathscr{C}_\lambda[X]$ is this lambda model. ∎

So given a lambda algebra $\mathscr{C}$, we can always extend it with at most a countable number of indeterminates to obtain a lambda model. Conversely, every lambda model is a lambda algebra; (7.1-5) follow from the Combinatory Model Theorem (iv) since each is of the form $u^{(\mathscr{C})} = v^{(\mathscr{C})}$ for convertible lambda terms u,v.

Corollary. (i)(Barendregt) The lambda algebras are precisely the class of all substructures of lambda models.

(ii) The lambda algebras are precisely the class of all homomorphic images of lambda models.

Proof. Applying homomorphisms and taking substructures preserves equations, so homomorphic images and substructures of models are algebras. Conversely, every lambda algebra $\mathscr{C}$ is an image and a substructure of $\mathscr{C}_\lambda[X]$. ■

Thus, we learn the unexpected facts that the class of homomorphic images and the class of substructures of lambda models coincide and are finitely axiomatizable by equations, namely the axioms for lambda algebras.

Extensional combinatory algebras and extensional combinatory models coincide. To characterize their substructures axiomatically, just add the axiom $I = \langle x\rangle(\langle y\rangle(xy))$ to (7.1-5). The resulting class of algebras are called *Curry algebras* (*cf.* [Lambek 80]).[10]

8. Further Directions. The development above reveals how to treat lambda calculus as a theory of equations for a class of ordinary algebraic structures which can be described alternatively as combinatory models, lambda models, or the polynomial algebras over lambda algebras.

Algebraic definitions and arguments can often offer more simplicity and greater appeal than syntactic ones, particularly if one can avoid the notorious pitfalls of substitution in the presence of bound variables. Having in principle eliminated the need for syntactic notions in defining which structures are models, the general question arises of how much more of the highly developed syntactic-computational "proof theory" of lambda calculus can be usefully understood from an algebraic "model theory" viewpoint. There has already been valuable interaction between the two viewpoints. One important example is worth sketching.

A lambda term has a *head normal form* if it converts to a term of the form $\lambda x_1...x_n.(yu)$ for some $n \geq 0$; "$\lambda x_1...x_n.y$" is called the *head* of the term and is unique up to renaming bound variables (for η-calculus there is a slightly more complicated kind of uniqueness property). A lambda term is *unsolvable* if it does not have a head normal form. By repeatedly converting the solvable subterms of any term u into head normal form and replacing unsolvable subterms by a new constant Ω, one obtains in the limit a unique, possibly infinite, term called the *Bohm tree* of u. The Bohm tree can be regarded as the trace of the possibly infinite computation needed to evaluate the term. Following earlier work in [Hyland 76, Wadsworth 76, Plotkin 78, Barendregt and Longo 80], Longo has recently observed that the value in $\mathbf{D}_A$ of a closed term u is set theoretically included in the value of a closed term v iff the Bohm tree of u *approximates* that of v, namely the Bohm tree of u is obtainable from the Bohm tree of v by replacing some of the subterms of the tree of v by Ω. In particular, an equation between lambda terms is valid in $\mathbf{D}_A$ iff the terms have the same Bohm tree. This provides an elegant connection between the syntactic-computational behavior of lambda terms and their meaning in a mathematically elementary model.[11]

As [Scott 80b] has emphasized, the *un*typed lambda calculus considered above can be viewed as the special case of the typed lambda calculus in which there is only *one* type. Most applications of lambda calculus in the study of programming languages and computability require the richer structure of multiple types. I hope to provide an elementary treatment of this generalization in a sequel tentatively titled "What is a solution of a domain equation?"

Acknowledgments. I thank Dana Scott for encouragement, technical suggestions, and explanation of his results, David McAllester whose excellent comments led to the present formulation of the Completeness Theorem, H. Barendregt and G. Longo for pointing out solutions to nearly all the questions I raised, Kim Bruce for patient close reading of several drafts, and Gerard Berry, Dan Cooperstock, T. Fehlmann, Joe Halpern, Roger Hindley, Anne Mahoney, A. Obtulowicz, Rohit Parikh, Gordon Plotkin, David Park, Vaughan Pratt, Charles Rackoff, Rick Statman, Robert Streett, Paul Weiss, and A. Wiweger for their comments and interest.

Notes.

1. If $\langle \mathbf{D}, \cdot \rangle$ is a combinatory algebra and ϵ satisfies (1.3a,b), then $\epsilon \cdot \epsilon$ satisfies (1.3a,b, *and* c), i.e., $\langle \mathbf{D}, \cdot, \epsilon \cdot \epsilon \rangle$ is a combinatory model, as is easily verified. So in a sense (1.3c) is a redundant, normalizing condition. The reason for requiring it is revealed in the Combinatory Model Theorem (iii) in Section 4.

2. [Scott 80a,80b] argues that "extensionality" should be reserved to refer to the weaker condition (ξ) given in Section 1, but I prefer to follow the more familiar usage.

3. [Cooperstock 81] is a study similar to this one in which Barendregt's structures, environment models, Aczel structures, Obtulowicz structures, and variations of combinatory models and Scott models (*cf.* Note 8) are compared. Cooperstock also presents a thoughtful discussion of the sense in which all the structures provide equivalent mechanisms for interpreting lambda terms.

[Barendregt 81] gives a comprehensive treatment of combinatory algebras, lambda models and lambda algebras (*cf.* Sections 6 and 7).

4. The technical setup here is very close to that of [Wadsworth 76], except that I have dropped any requirement of a lattice structure on **D** as well as the requirement that the maps Φ and Ψ be (continuous) isomorphisms. In fact [Obtulowicz 77, Obtulowicz and Wiweger 78] give essentially this definition which they credit as implicit in [Wadsworth 76]. Precisely the definition of environment model is also given in [Cooperstock 81] where it is credited as jointly proposed by Cooperstock and C. Rackoff based on the preceding earlier references.

[Barendregt 77] defines valuations over a more general class of structures resembling functional domains using essentially the same rules (3.1-2),(3.5-6), but valuations over these more general structures suffer the flaw that the (ξ) rule is not sound. More recently [Berry 80] has offered a definition which is a combination of Barendregt's notion and term models.

The pathologies of Barendregt's structures are lucidly analyzed by [Hindley and Longo 78] who essentially identify the structures as combinatory algebras and lambda algebras (considered in Section 7). Hindley and Longo also arrive at a definition equivalent to environment models (which they call λ-structures) by adding to Barendregt's formulation the requirement that the (ξ) rule be sound. They note by the way that their formulation was obtained independently of Barendregt's, and I note that my definitions were formulated independently of all the papers subsequent to [Wadsworth 76].

5. Alert readers may remember that our formal definitions require that constants in lambda terms given values in $\mathscr{E}$ must be elements of **D**, so we must identify constants $c \in C$ with the corresponding constants $[[c]]_{\mathscr{T}} \in \mathbf{D}$.

6. The construction is taken directly from [Engeler 79, *cf.* Fehlmann 79]. It is a notational variant of one of several models first described in [Plotkin 72]. These constructions are nearly the same as the better known **P**ω construction [Scott 76].

Indeed, Longo [personal communication 1981] has shown that $\mathbf{D}_A$ and **P**ω have the same pure lambda theory and each is isomorphically embeddable in the other. However, they define different set theoretic *inclusions* among the values of the pure closed terms, and their binary operations behave differently, i.e., they are not even isomorphic as combinatory algebras.

7. The choice of ϵ is not unique. For example, let

$$\epsilon^+ = \epsilon \cup A \cup \{(\beta \to a) \mid \beta \subseteq B \text{ and } a \in A\}.$$

Then $\langle \mathbf{D}_A, \cdot, \epsilon \rangle$ and $\langle \mathbf{D}_A, \cdot, \epsilon^+ \rangle$ are distinct expansions of the combinatory algebra $\langle \mathbf{D}_A, \cdot \rangle$ to combinatory models. Longo [personal communication 1981] has even shown that they have distinct pure lambda theories; in fact ϵ^+ yields an interesting model which, in contrast to the $\mathbf{D}_A$ model with ϵ or the **P**ω model, does not give all unsolvable terms the same value (*cf.* Section 8).

8. In general, ϵ_n is chosen to be $\mathscr{V}_{\mathscr{B}}[\lambda x_0 \ldots x_n . (x_0 \ldots x_n)]$. Continuing with a purely algebraic approach, we could define following [Scott 80b]

$$\epsilon_1 = \epsilon \text{ and } \epsilon_{n+1} = (B \cdot \epsilon) \cdot (B \cdot \epsilon_n) \text{ for } n > 0.$$

It is easy to verify that in any combinatory model $\langle \mathbf{D}, \cdot, \epsilon \rangle$,

(N.1) $\epsilon_n \mathbf{d}_0 \ldots \mathbf{d}_n = \mathbf{d}_0 \mathbf{d}_1 \ldots \mathbf{d}_n$,

(N.2) if $\forall e_1, \ldots, e_n \in \mathbf{D}.\ \mathbf{d}_0 e_1 \ldots e_n = \mathbf{d}_1 e_1 \ldots e_n$ then $\epsilon_n \mathbf{d}_0 = \epsilon_n \mathbf{d}_1$, and

(N.3) $\epsilon(\epsilon_n \mathbf{d}_0) = \epsilon_n(\epsilon \mathbf{d}_0) = \epsilon \mathbf{d}_0$

for all $\mathbf{d}_0, \ldots, \mathbf{d}_n \in \mathbf{D}$, $n > 0$.

The reader might enjoy deriving an algebraic proof solely from (N.1-3) that in any combinatory model $\langle \mathbf{D}, \cdot, \epsilon \rangle$, there is exactly one pair of elements K and S satisfying (1.1-2), (6.6-7).

These equations suggest another axiomatization of models proposed by Scott [*cf.* Volken 78, Barendregt 81, Thm.5.4.9].

Definition. Let $\mathcal{S} = \langle \mathbf{D},\cdot,\mathbf{F}\rangle$ be a structure where $\cdot$ is a binary operation on $\mathbf{D}$ and $\mathbf{F} \subseteq \mathbf{D}$. Let $\mathbf{F}_0 = \mathbf{D}$ and $\mathbf{F}_{n+1} = \{\mathbf{d}_0 \in \mathbf{F} \mid \mathbf{d}_0 \cdot \mathbf{d}_1 \in \mathbf{F}_n \text{ for all } \mathbf{d}_1 \in \mathbf{D}\}$. $\mathcal{S}$ is a *Scott Model* if, for all n>0 and any $\mathcal{C}$-term u over **D** such that x_0 is not free in u, $\mathcal{S}$ satisfies

$$\exists! x_0 \in \mathbf{F}_n \ \forall x_1 \ldots x_n \in \mathbf{D}[x_0 \ldots x_n = u].$$

It is easy to see that if $\mathcal{S} = \langle \mathbf{D},\cdot,\mathbf{F}\rangle$ is a Scott model, then $\langle \mathbf{D},\cdot,\epsilon\rangle$ is a combinatory model, where ϵ is the unique element of $\mathbf{F}_2$ such that (1.3b) is valid in $\mathcal{S}$. Conversely, if $\langle \mathbf{D},\cdot,\epsilon\rangle$ is a combinatory model, then $\langle \mathbf{D},\cdot,\mathbf{F}\rangle$ is a Scott model where $\mathbf{F} = \{\epsilon \cdot \mathbf{d} \mid \mathbf{d} \in \mathbf{D}\} = \epsilon \cdot \mathbf{D}$. In fact, $\mathbf{F}_n = \epsilon_n \cdot \mathbf{D}$.

9. Our definition of the transformation $\langle x\rangle$ on $\mathcal{C}$-terms was chosen for ease in proofs rather than efficiency, and consequently the length of $\langle x\rangle u$ has been allowed to grow exponentially in the length of u. There exist transforms with the same properties as $\langle x\rangle$ which increase the length only linearly.

10. [Barendregt and Koymans 80] show that not all combinatory algebras can be *expanded* by choice of K,S into lambda algebras. The interior of the combinatory word algebra based on K,S-terms is an example of such a combinatory algebra. They also show, as noted at the end of Section 6, that not all lambda algebras are lambda models.

11. It also provides a simple model theoretic characterization of the syntactic concept of *normal form*, as pointed out by Longo. Namely, $\mathbf{d} \in \mathbf{D}_A$ is the value of a pure closed lambda term in normal form iff **d** is maximal under set inclusion in the interior of $\mathbf{D}_A$ and contains only finitely many elements of the interior.

References.

Aczel, P. (1980) Frege structures and the notions of proposition, truth and set, *The Kleene Symposium*, Barwise *et. al.*, ed., Studies in Logic 101, North Holland, New York, 31-60.

Barendregt, H.P. (1977) The type free lambda calculus, *Handbook of Mathematical Logic*, J. Barwise, ed., North Holland, New York, 1091-1132.

Barendregt, H.P. (1981) *The Lambda Calculus Its Syntax And Semantics*, Studies in Logic, 103, North Holland, New York, xiv+615pp.

Barendregt, H. and Koymans, K. (1980) Comparing some classes of lambda-calculus models, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, Seldin,J.P. and Hindley, J.R., ed., Academic Press, New York, 287-302.

Barendregt, H.P. and Longo, G. (1980) Equality of λ-terms in the model T^ω, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, Seldin,J.P. and Hindley, J.R., ed., Academic Press, New York, 303-338.

Berry, G. (1980) On the definition of lambda-calculus models, *INRIA Rapports de Recherche*, No. 46, 12pp.

Cooperstock, D. (1981) Alternative axiomatizations of models of the lambda-calculus, Technical Report 151/81, Dept. of Computer Science, Univ. of Toronto, 72pp.

Engeler, E. (1979) Algebras and combinators, Berichte des Instituts fur Informatik, Nr. 32, ETH Zurich, 12pp.

Fehlmann, T. (1979) Computation with infinite objects, *manuscript*, 47pp.

Hyland, M. (1976) A syntactic characterization of the equality in some models for the lambda calculus, *J. London Math. Soc.*, 361-370.

Hindley, R., Lercher, B. and Seldin, J. (1972) *Introduction to Combinatory Logic*, London Math. Soc. Lecture Note Series **7**, Cambridge Univ. Press, 170pp.

Hindley, R., and Longo, G. (1980) Lambda-calculus models and extensionality, *Zeit. f. Math. Logik u. Grund. der Math.* **26**, 289-310.

Lambek, J. (1980) From λ-calculus to cartesian closed categories, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, Seldin,J.P. and Hindley, J.R., ed., Academic Press, New York, 375-402.

Landin, P.J. (1964) The mechanical evaluation of expressions, *Computer J.* **6**, 308-20.

Landin, P.J. (1965) A correspondence between ALGOL 60 and Church's lambda-notation: part I and II, *Communications ACM* **8**, 89-101, 158-65.

Obtulowicz, A. (1977) Functorial semantics of the type free λ-$\beta\eta$ calculus, *Fundamentals of Computation Theory*, Lecture Notes in Computer Science **56**, Springer-Verlag, New York, 302-307.

Obtulowicz, A. and Wiweger, A. (1978) Categorical, functorial and algebraic aspects of the type free lambda calculus, Preprint Nr 164, Institute of Mathematics, Polish Academy of Sciences, Sniadeckich 8, skr.poczt. 137, 00-950 Warszaw Poland, 38pp.

Plotkin, G.D. (1972) A set-theoretical definition of application, Memorandum MIP-R-95, School of Artificial Intelligence, Univ. of Edinburgh, 32pp.

Plotkin, G.D. (1974) The lambda calculus is ω-incomplete, *J. Symbolic Logic*, **39**, 313-317.

Plotkin. G.D. (1978) $\mathbf{T}^{\omega}$ as a universal domain, *Journal of Computer and System Sciences*, **17**, 2, 209-236.

Scott, D.S., (1976) Data types as lattices, *SIAM J. on Computing*, **5**, 522-587.

Scott, D.S., (1980a) Lambda calculus: some models, some philosophy, *The Kleene Symposium*, Barwise *et. al.*, ed., Studies in Logic 101, North Holland, New York, 381-421.

Scott, D.S., (1980b) Relating theories of the λ-calculus, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, Seldin,J.P. and Hindley, J.R., ed., Academic Press, New York, 403-450.

Scott, D.S. (1980c) Lectures on a mathematical theory of computation, *manuscript*, Univ. of Oxford, 135pp.

Stoy, J. E. (1977) *Denotational Semantics: The Scott-Strachey Approach to Programming Language Theory*, M.I.T. Press, Cambridge, Mass., xxx+414pp.

Volken, H. (1978) Formale Stetigkeit un Modelle des Lambda Kalkuls, Ph. D. thesis, E.T.H., Zurich.

Wadsworth, C. (1976) The relation between computational and denotational properties for Scott's D_{∞}-models of the lambda-calculus, *SIAM J. Comput.*, **5**,3, 488-521.