To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism (ed. by J. P. Seldin and J. R. Hindley), Academic Press (1980), pp. 403-450.

RELATING THEORIES OF THE λ-CALCULUS

Dana S. Scott

Merton College
Oxford

Dedicated to Professor H. B. Curry on the occasion of his 80th Birthday

Mathematical theories arise for many different reasons, sometimes in connection with specific applications and often owing to accidental inspiration. From time to time we ought to ask ourselves concerning our theories where *should* they have come from; usually the answer will have little to do with the exact historical development. The λ-calculus is, I feel, a case in point. In Scott (1980), in the Kleene Festschrift, I made up a story of where the theory of type-free λ-calculus *could* have come from. Any number of people who heard my lecture and read the manuscript were cross with me. They said "But it didn't develop that way! And besides we doubt it ever would have." But this reaction misses the point of my story. I shall not, however, repeat the earlier story here, for the point of the present paper is different. For those people who do not like to discuss philosophy - even Philosophy of Mathematics - my remarks here can be taken as a suggestion of how to group diverse models of λ-calculus rather uniformly under a general scheme. The scheme is by now rather well known and not at all original with me. What I hope can be regarded as a useful contribution is my putting of the ideas in a certain order. As I consider the order to be a natural one, I feel there is philosophical significance to my activity; but I should not want to force this view on anyone.

1. THEORIES OF FUNCTIONS.

Everyone agrees that λ-calculus is a theory of functions. But we must ask: "What kind of a theory?" And also: "Have we got the best theory?" Personally, I think we should also inquire: "How does it relate to other theories?" I certainly find many discussions far too silent on this last issue.

Well, what other theories are there? Certainly *set theory* comes to mind at once, and no set theory would be worth its salt if it did not provide a theory of functions. Let us not try to catalogue the various known theories here but look at a theory in the style of Zermelo - and we do not have even to be too specific, since in any case such a theory is very standard. What is "unsatisfactory" about Zermelo's theory is the limitization-of-size view of sets: any *one* set A is extremely small compared to the size of V, the class or universe of *all* sets. Thus, functions $f : A \to B$ mapping one set A into another set B tell us very little about operations on all sets, maps on V into V. We therefore have an urge to "improve" our set theory by constructing a *class theory*. Sets are elements $A \in V$; while classes are subcollections $B \subseteq V$. As V is (by the usual assumptions) so highly closed under so many operations, we have no difficulty in construing certain classes as maps $F : V \to V$. For example for all $X \in V$ we could have $F(X) = \{X\}$ or $F(X) = A \times X$ (where A is a fixed set).

The passage from sets to classes is a familiar and useful move in the formalization of the theory: many things can be done generally for classes and then specialized to sets. And having a notation for functions defined on all sets is in many cases a great advantage. But wait. What about operations on classes? What should we say about them? Given any two classes A and B, we can form their union, $A \cup B$. The operation,

$\cup : V \times V \to V$, of union of *sets* does not directly apply to *classes* even though there is a connection. Do we also want a theory of class operations? Do we have to go to hyperclasses (classes of classes)? Is there any end to this expansion?

[*An Aside*: The story of Scott (1980) was meant to suggest one answer - the one known to Plotkin (1972). Namely, we consider only "continuous" class operations. These are objects F such that F(X) is defined for every class $X \subseteq V$ and F(X) is a class, too. Moreover F should satisfy:

(1) $X \subseteq Y$ always implies $F(X) \subseteq F(Y)$;

(2) Whenever $A \subseteq F(X)$ and A is a set, then $A \subseteq F(B)$ for some set $B \subseteq X$.

We do not have time to discuss the justification of the word "continuous" here; suffice it to say that conditions (1) and (2) are not as strict as they at first might seem. Every *ordinary* map $f : V \to V$ determines a continuous class operator by the definition:

$$F(X) = \{f(x) \mid x \in X\}.$$

Furthermore, F determines f, for we have:

$$y = f(x) \text{ iff } \{y\} = F(\{x\}),$$

for all $x,y \in V$. In a suitable sense, then, nothing has been lost; but what has been gained?

The reply is that continuous class operators can be *identified* with classes. We could write, for instance:

$$F = \{(A,B) \mid A,B \in V \wedge A \subseteq F(B)\},$$

where, say:

$$(A,B) = \{\{A\}, \{A,B\}\}.$$

More in harmony with Scott (1980) would be:

$$F = \{(a,B) \mid a,B \in V \wedge a \in F(B)\}.$$

Either trick reduces operator theory to class theory - in the continuous case. And the same trick could be carried over to other kinds of set theory (e.g. Quine's). What we know is that operator theory gives a model for λ-calculus; it is a quite elementary model, too.

Nice as this connection is, it is not the topic of the present paper: we do not want to make λ-calculus depend on set theory, since then we have still to explain where set theory comes from. But the connection should be borne in mind.]

Perhaps set theory brings in too many extraneous issues. V, after all, is a massive object closed under all manner of strange operations. What we are probably seeking is a "purer" view of functions: a theory of functions in themselves, not a theory of functions derived from sets. What, then, is a pure theory of functions? Answer: category theory.

General category theory is a *very* pure theory: it is the milk-and-water theory of functions under *composition*. This composition operation is associative and possesses neutral elements (compositions of zero terms). That is about all you can say about it except to stress that it is also a rather bland theory of *types*. Every function f has a (unique) *domain* and *codomain*, and we write:

$$f : \mathrm{dom}\ f \to \mathrm{cod}\ f$$

Every possible domain is a codomain (and conversely), because if A is such, then

$$\mathrm{dom}\ 1_A = A = \mathrm{cod}\ 1_A,$$

where 1_A is the neutral element of type A.

[If we want to be especially parsimonious in entities, we can even write $1_A = A$, because each of 1_A and A uniquely determines the other.]

The point of distinguishing domains and codomains is not only do they specify the type of f, but a composition $g \circ f$ is

defined if, and only if, $\mathrm{dom}\ g = \mathrm{cod}\ f$. And then $\mathrm{dom}(g \circ f) = \mathrm{dom}\ f$ and $\mathrm{cod}\ (g \circ f) = \mathrm{cod}\ (g)$. We usually write this as a "rule of inference":

$$\frac{f : A \to B \qquad g : B \to C}{g \circ f : A \to C}$$

with the understanding that the typing of $f \circ g$ can only be obtained by such an application of the rule. The types, then, are invoked just to type functions, and the only theory involved is that of the "transition" of types under composition.

Sets (and set-theoretical mappings) do of course form a category; category theory is *meant* to be more general than set theory. We should construe the function entities here as triples of sets (A,f,B) where

$$f \subseteq A \times B \wedge \forall x \in A\ \exists ! y \in B.\ (x,y) \in f.$$

The definition of composition is obvious. Sets, in this way, give us only one special example of a category.

I beg forgiveness of the reader for boring him. All of this is well known to the moderately awake undergraduate in mathematics. Indeed, that is the point: there is plenty of evidence now that category theory is a natural *and* useful theory of functions. I do not have to rehearse the examples as they can be found in any number of books (e.g. Mac Lane [1971]). There is a rather important logical point to stress, however - important for anyone who has thought about λ-calculus models. Category theory is very *extensional*. We assume as axioms the *equations*:

$$1_A \circ f = f \circ 1_B = f \quad \text{and}$$

$$h \circ (g \circ f) = (h \circ g) \circ f,$$

provided $A = \mathrm{dom}\ f$ and $B = \mathrm{cod}\ f$ and the double compositions are defined. These are *functional* equations, and they say that two functions defined in *different* ways are in fact *identical*.

Furthermore, identical things can everywhere replace one another.

This point about extensionality may not seem exciting or important, but the logician should remember that, in certain *intensional* theories of functions, "obvious" definitions will *not* provide categories. We shall return to this point later.

But is category theory the long-sought answer? No, no, not at all. Category theory *pure* provides nothing explicitly aside from identity functions - and they occur only *if* we have some possible domain. We do get compositions *if* we have the necessary terms. Thus, as it stands, category theory has no existential import. (It was not meant to.) Set theory has "too much" existential import. (It was meant to.) What we seek is the middle way - and an argument that the middle way is natural and general.

There is no need to build up unnecessary suspense: the middle way is the theory of the (so called) *cartesian closed categories*. Fortunately Lambek has written extensively about the theory, and I can refer the reader to his papers for further details; I also am happy to acknowledge his writings as helping me understand what is going on. If we remark that his paper in *this* volume is called "From λ-calculus to cartesian closed categories", we might say that my present paper ought to be called "From cartesian closed categories to λ-calculus." I am trying to find out where λ-calculus *should* come from, and the fact that the notion of a cartesian closed category (c.c.c) is a late developing one (Eilenberg & Kelly (1966)), is not relevant to the argument: I shall try to explain in my own words in the next section why we should look to it *first*.

2. A THEORY OF TYPES

I say "a theory", because there are many possible theories; indeed pure category theory is one of the theories. Its weakness lies in the fact that we are given no construction principles, no way of making new types from old. From the point of view of logic what should we expect? What more do we want to say beyond relations between types which hold when a mapping statement $f : A \to B$ obtains.

An immediate question that must come to anyone's mind concerns the *arity* of functions. The usual way of reading a mapping statement is to take it as a statement about *one-place* functions, and the $\circ$ of composition is the composition of one-place functions. This seems very restricted.

People have suggested generalizing categories to *multi-place* functions with concomitant compositions (cf. e.g. the book Szabo (1978)), but it does not seem the neatest solution. Much easier is to assume that the category has cartesian products - and more specifically particular representatives of the product domains are chosen. As a special case we will know what the cartesian power A^n is for each $n=0,1,2,\ldots$, and n-ary functions are then maps $f : A^n \to B$. Not much of a surprise.

We have to take care, however. In the first place, a given category may not have cartesian products (it fails to have enough types). Even if it does, the maps allowed may be too restricted - for logical purposes. Take the category of groups and homomorphisms, for example. The required products exist. A map $f : A^2 \to B$ in this category has to be a group homomorphism, naturally. Suppose two maps $u,v : A \to B$ were given. Intuitively we think in terms of elements and that we are mapping $x \mapsto u(x)$ and $y \mapsto v(y)$. The pointwise group product of the maps, namely, $x,y \mapsto u(x) \cdot v(y)$ is a very nice map $g : A^2 \to B$ in the ordinary sense - but unless the group B is abelian, g is not a homomorphism. It is a "logical" map but not an "algebraic" map. Pure category theory applies to many algebraic situations (as everyone knows that is why it is a good theory), but not all categories are "logical" even if they have products. In the example of groups, what was "missing" was the group multiplication $\mu : B^2 \to B$ as a map in the category. (Inverse is missing as well, since it reverses order.) There is an interesting theory of algebraic theories that address the question of the proper categorial construction of categories of algebras, but I do not think we should invoke that theory here.

The precise description of products is as follows. We assume our category has a special domain $\mathbb{1}$ (*the* empty product, so $A^0 = B^0 = \mathbb{1}$), and for each domain A a special map $0_A : A \to \mathbb{1}$. (The domain $\mathbb{1}$, intuitively, has just one "element".) Moreover the rule about maps is that 0_A is unique; that is whenever $f : A \to \mathbb{1}$, then we have the equation

$$f = 0_A.$$

Concerning *binary* products, we have for any two domains A and B a special choice of a domain $A \times B$ (and so $A^{n+1} = A^n \times A$), and special maps

$$p_{AB} : A \times B \to A$$
$$q_{AB} : A \times B \to B$$

But the mere existence of "projections" does not characterize $A \times B$ as a product. We have to assume that there is a chosen pairing operation $\langle f,g \rangle$ on maps such that types are assigned by the rule:

$$\frac{f : C \to A \qquad g : C \to B}{\langle f,g \rangle : C \to A \times B}$$

Moreover, provided f and g are as above and $h : C \to A \times B$ we have to assume

$$p_{AB} \circ \langle f,g \rangle = f$$

$$q_{AB} \circ \langle f,g \rangle = g$$

$$\langle p_{AB} \circ h,\ q_{AB} \circ h \rangle = h.$$

that is to say, there is an explicit one-one correspondence between the pairs of maps f,g and the maps h into the product. This all now makes $A \times B$ well behaved within the category.

So much for a theory of tuples and multiary maps. But we still want a *theory* of functions: a category allows us to talk of *selected* functions, while we would want various equations to have a force relating to *arbitrary* functions. The answer to this desire is *function spaces* as explicit domains in the category. Given A and B we want to form $(A \to B)$ as a domain in its own right; if so, there are many maps that have to be set down to make the function space behave. (And here we must definitely leave the category of groups.)

In the first place there has to be an *evaluation map* $\varepsilon_{BC} : (B \to C) \times B \to C$ with the intuitive interpretation that it is the map $f,x \mapsto f(x)$. In the second place there has to be a map for shifting around variables; more precisely, suppose $h : A \times B \to C$ is a map with two arguments. In an evaluation $h(x,y)$, we can think of holding x constant and regarding $h(x,y)$ as a function of y. We need a name for this function – and for the correspondence with possible values of x. We write

$$\Lambda_{ABC}\, h : A \to (B \to C)$$

so that the function we were thinking of – given x – was $(\Lambda_{ABC}\, h)(x)$. But all this function-value notation is not categorical notation; what we have to say is that there is a one-one correspondence via Λ between maps $h : A \times B \to C$ and maps $k : A \to (B \to C)$. This comes down to these two equations:

$$\varepsilon \circ \langle (\Lambda h) \circ p,\ q \rangle = h, \quad \text{and}$$

$$\Lambda(\varepsilon \circ \langle k \circ p,\ q \rangle) = k.$$

(It is necessary to have subscripts here: Λ_{ABC}, p_{AB}, q_{AB}, and ε_{BC}; but we leave them off when there is no ambiguity.)

The notation is now wholly categorical (and mostly unreadable). Category theorists put the whole thing (that is, the definition of a c.c.c., which is what we have just given) into the language of functors, which has a lot of sense. But if you have never seen any abstract category theory before, it is really rather too abstract. The idea of a c.c.c. as a system of types is, I think, reasonably simple. *Each c.c.c. represents a theory of functions.* The maps in the category are certain special functions that are used to express the relations between the types (the domains of the category). In order to be able to deal with multiary functions, we assume we can form (and analyze) products. In order to be able to work with transformations of arbitrary functions ("arbitrary" within the theory), we assume we can form function spaces: this is where "higher" types enter the theory, as in the sequence of domains:

$$A,\ (A \to A),\ ((A \to A) \to A),\ (((A \to A) \to A) \to A), \ldots$$

To be able really to view these domains as function spaces, certain operations, ε and Λ, with characteristic equations have to be laid down.

If a c.c.c. is a theory of functions (and we include here higher-type functions), then the theory of c.c.c's is the *theory of types* of the title of this section. It is only one such theory. "Bigger" theories could be obtained by demanding more types: for example we could axiomatize *coproducts* (disjoint sums), $\mathbb{0}$, and $A + B$. We could demand *infinite* products and coproducts. We could throw in a type Ω of "propositions" so that higher types like $(A^n \to \Omega)$ correspond to n-ary *predicates*. This gets into *topos theory* (as in Johnstone (1977) or Goldblatt (1979) - just to name two recent texts). But the bigger the theory, the more involved, and full definitions at this point would not help this discussion very much.

We could also look for "smaller" theories. Some examples - of a rather highly formal nature - can be found in Szabo (1978) with an indication of the algebraic interest of these other type theories. However, a c.c.c. is rather more "logical" and good as a middle ground; further Lambek has explained the logical interest; *there is a perfect correspondence between c.c.c.'s and (extensional) typed λ-calculi.* The reader can turn to Lambek's paper for details and references.

Roughly put, when we formally define the typed λ-language (with types in the given c.c.c.), then if τ is a typed λ-expression with free variables of types $A_0, A_1, \ldots, A_{n-1}$, we can define the "meaning" of τ as a map

$$[\![\tau]\!] : A_0 \times A_1 \times \ldots \times A_{n-1} \to B,$$

where B is the type of τ. For example if u is a variable of type $(B \to C)$ and v a variable of type B, then

$$[\![u(v)]\!] : (B \to C) \times B \to C,$$

and in fact $[\![u(v)]\!] = \varepsilon_{BC}$. Also if x is the variable in τ of type A_{n-1}, then

$$[\![\lambda x . \tau]\!] : A_0 \times \ldots \times A_{n-2} \to (A_{n-1} \to B),$$

and in fact $[\![\lambda x . \tau]\!] = \Lambda[\![\tau]\!]$. (Warning: for other of the variables that are not the last mentioned, it is not so easy to write down the answer: some permutations of the products have to be introduced.)

The two characteristic equations for ε and Λ in the axioms for a c.c.c. have very familiar translations:

$$(\lambda y . h(x,y))(y') = h(x,y') \text{ and}$$

$$\lambda y . k(x)(y) = k(x),$$

where the type of x is A, the type of y and of y' is B. (That is to say, the $[\![\cdot]\!]$-meaning of the two sides of the equation is the same map in the category.) Of course this all has to be defined more rigorously, but I hope I have conveyed the main part of the idea of Lambek's correspondence. A typed λ-calculus (with pairs, products, and function spaces) is just another notation for a c.c.c.

No, we have to be more specific than that. Take a c.c.c. How does it correspond to a theory (of functions)? The domains of the category are the *types* of the theory, and they are structured by the $\mathbb{1}$, $(A \times B)$, $(A \to B)$ operations on types. Things like $\circ$, $0, p, q$, $\langle \cdot, \cdot \rangle$, ε, Λ stand for *logical constants* (or operators) – with type subscripts as needed. Maps $f : A \to B$ of the category stand for the *non-logical constants* of the theory. The equations $f = g$ between maps are the *assertions* of the theory. The logical axioms are those special equations common to all c.c.c.'s – the other equations are those that just happen to work out in the category. From this point of view the theory has *no free variables:* all assertions are written with constant terms. Equations with free variables can be construed as functional equations (by a heavy use of Λ).

Conversely, a more conventional typed λ-calculus is an equational theory with both the familiar logical axioms as well as with non-logical axioms as desired. The equations can involve free variables. Aside from the usual deduction rules for equality, we *must* employ the extensionality rule

$$\frac{\tau = \sigma}{\lambda x \,.\, \tau \;=\; \lambda x \,.\sigma}$$

A category is formed from the types (which are *given* as closed under $\mathbb{1}$, $(A \times B)$, $(A \to B)$). The terms all have unambiguous types, and they are divided into equivalence classes by the theory. As the maps of the category we take the equivalence classes $[\lambda x \,.\, \tau]$ where the term τ of type B has at most the variable x of type A free, and we write $[\lambda x \,.\, \tau] : A \to B$. Of course

$$1_A = [\lambda x \,.\, x], \text{ and}$$

$$[\lambda y \,.\, \sigma] \circ [\lambda x \,.\, \tau] = [\lambda x \,.\, \sigma(\tau/y)]$$

where τ is substituted for y in σ. We must verify that a category is obtained – using the laws of λ-calculus. And we must see that if we go back again to a λ-calculus from the category we have essentially the same theory.

A c.c.c. (or typed λ-calculus – with non-logical axioms) is a satisfactory (extensional) theory of functions because all we have built into the theory is the idea of the product and the function space. The axioms set down are just those needed to make this structuring explicit.

The reason that category theory is a convenient way to formalize this definition is that starting from the especially elementary concept of maps under composition, we can see that we have done *nothing more* than close up under products and function spaces. λ-calculus, then, becomes mostly a notational device for setting down our functional equations. At least for typed λ-calculus, we can see in this way that it is harmless.

The typed λ-calculus is even more harmless than these last remarks suggested. By the well-known Yoneda embedding, one can prove that an arbitrary (small) category has a full and faithful embedding into a c.c.c. This means that starting with a given category and its maps, there is a precise sense in which it is consistent to close up under products and function spaces. No new maps are added to the given category; no new equations between the given maps are imposed by the adjunction of higher types. One can say even more than this about relative consistency, but the remark is best deferred to Section 4, where references to the proof are provided.

A final remark must be added to this section to clear up a possible confusion between *theories* and *models*.

Up to this point we have been talking about *theories*. In many systems of logic models can be described by theories: every model has a "diagram" involving constants for all the "elements" of the model and taking as axioms all statements in the language "true" about the model. It depends on the nature of the logic how hard it is to show that every "consistent" theory has a model.

In the case of a c.c.c., a domain A could be said to have an "element" if there is a map $a : \mathbb{1} \to A$. The question is: *are there enough elements*? Suppose $f, g : A \to B$ are two maps. If $a : \mathbb{1} \to A$, then $f \circ a : \mathbb{1} \to B$; so in a certain sense maps in the category behave as functions on elements. (This is not an original suggestion but is one well known in category theory.) It is natural to ask whether, if $f \circ a = g \circ a$ for all $a : \mathbb{1} \to A$, then $f = g$. If this is true in a c.c.c., then it is said to have "enough" elements or to be *concrete*. In case it is concrete, domains can be identified with sets, maps with functions, products $A \times B$ in the category with the corresponding cartesian product of the sets (ask: which $c : \mathbb{1} \to A \times B$?), and function spaces $(A \to B)$ with spaces of actual functions (because there is a one-one correspondence between maps $f : A \to B$ and elements $e : \mathbb{1} \to (A \to B)$).

For a theory in the form of a c.c.c., to ask whether it has a (non-trivial) model is to ask whether it can be expanded to a concrete c.c.c. by the adjunction of elements (and other maps and additional equations, but no new domains) which is non-trivial in the sense of not making all domains isomorphic to $\mathbb{1}$.

An answer - though perhaps a rather formal one - is supplied by the method of adjunction of indeterminates $x : \mathbb{1} \to A$ presented in Lambek's paper (this volume). We just have to adjoin infinitely many for each domain, one after the other. Each polynomial involves only finitely many indeterminates. But the results stated by Lambek (esp. Corollary to Theorem 2) show us at once that this expanded category is concrete. The idea is really just like the idea of having "free algebras" for any equational theory. (In λ-calculus an algebraic equation that is regarded as universally quantified, say $x + y = y + x$, is

replaced by the functional equation

$$\lambda x \lambda y \,.\, x + y = \lambda x \lambda y \,.\, y + x.)$$

More thoughts on concretness will be brought out in Section 4.

That is the (easy) passage from theories to (certain) models. But remember, a theory is *not* a model: the maps in a given c.c.c. are not concrete maps, they are just the *definable* maps in the language of the theory, and the equations between them are the "theorems" of the theory. It is no surprise that a given theory may not have enough *definable* elements: we may need to expand the stock of elements in order to have a model. For a c.c.c. we find we can. So far, so good; and this is the (known) story of typed λ-calculus.

3. "TYPE-FREE" DOMAINS

In the paper of Lambek, the *analogy* between typed and type-free is illustrated (in the obvious way), but no real *connection* or *relation* is established. This we shall now do, and the relationship will be deepened in the next section.

In the first place, we shall only consider the λ-calculus (or λK-calculus) and not the λη-calculus; the latter can be regarded as a special case. What is needed is a notion of domain appropriate to the interpretation of the "type-free" calculus.

In a category, a *retraction* between two domains A and B is a pair of maps $i : A \to B$ and $j : B \to A$ where $j \circ i = 1_A$. Regard A as the "smaller" domain; it is injected into B, and B is surjectively mapped onto A. The notion shares qualities, then, of A being both a *subspace* of B and at the same time a *quotient*. But the injection and surjection have to be related.

Now, suppose that in a cartesian closed category a domain U satisfies the condition that the function space $(U \to U)$ is a retract of the domain U itself. (This is always so for $U = \mathbb{1}$, but we seek non-trivial examples.) Let the retraction maps be $i : (U \to U) \to U$ and $j : U \to (U \to U)$. Then U (as it sits in its category) gives us an interpretation of the type-free calculus, which we now explain.

Let the type-free terms be constructed in the usual way from variables $x,y,z,\ldots$ by means of application and λ-abstraction. Think of all variables as being of type U and define a translation τ^* from untyped terms to typed terms so that

$$x^* = x,$$
$$(\tau(\sigma))^* = j(\tau^*)(\sigma^*),$$
$$(\lambda x . \tau)^* = i(\lambda x . \tau^*).$$

We intend this in such a way that τ^* is always of type U. The type-free theory (determined by the category, the domain U, and by the choice of i and j) has as its assertions exactly those equations $\tau = \sigma$ where $\tau^* = \sigma^*$ in the category. The theory satisfies (α), (β)-conversion, all the rules of equality, and the rule (ξ): from $\tau = \sigma$ to deduce $\lambda x . \tau = \lambda x . \sigma$. This much is surely obvious to anyone reading Lambek's paper.

What I would like to point out here is the *converse*: given any type-free theory, there is a c.c.c. and a domain U (with a suitable retraction pair i,j) so that the above interpretation gives *exactly the same* type-free theory. Consequently, nothing is lost in considering type-free theories just as *special parts* of typed theories. I do not find this result mentioned by Lambek.

The proof is elementary. Let the domains for the category be the λ-terms A, without free variables, for which we can prove in the theory:

$$A = \lambda x . A(A(x)).$$

The maps $f : A \to B$ are terms f without free variables for which we can prove

$$f = \lambda x . B(f(A(x))).$$

The equations between maps are the equations we can prove in the theory. [Actually, it might be better to construe maps as triples (A,f,B), but never mind.] It is not hard to show that this is a category where

$$1_A = A, \text{ and}$$
$$f \circ g = \lambda x . f(g(x)).$$

[More properly spoken, the maps should be equivalence classes of terms based on the equations of theory, but never mind.]

To show this construction gives a c.c.c. we need to define:

$$A \times B = \lambda u\lambda z . z(A(u(\lambda x\lambda y . x)))(B(u(\lambda x\lambda y . y))),$$

where

$$p_{AB} = \lambda u . (A \times B)(u)(\lambda x\lambda y . x),$$
$$q_{AB} = \lambda u . (A \times B)(u)(\lambda x\lambda y . y),$$

and if $f : C \to A$ and $g : C \to B$, then

$$\langle f,g\rangle = \lambda t\lambda z . z(f(t))(g(t)).$$

All of this is based on the familiar pairing functions of λ-calculus.

For function spaces, we define:

$$(A \to B) = \lambda f.\ B \circ f \circ A$$

where

$$\varepsilon_{BC} = \lambda u.\ C(u(\lambda x\lambda y.x)(B(u(\lambda x\lambda y.y)))),$$

and

$$\Lambda_{ABC}\, h = \lambda x\lambda y.h(\lambda z.z(x)(y)),$$

provided $h : (A \times B) \to C$.

There are a jolly lot of equations to verify, but the work is all straight-forward conversion. The method of retracts as a c.c.c. has in any case been exposed before with respect to the $P\omega$ model in Scott (1976). Note here, however, we are to verify the required equations in a theory (not a model) making use of nothing but the "logical" axioms of λ-calculus.

It remains to identify the domain U in the constructed category. We define:

$$U = \lambda x.\ x$$

Clearly

$$U = \lambda x.\ U(U(x)) = U \circ U.$$

Note that every A in the category is a retract of U; indeed, for retractions define:

$$A : A \to U \quad \text{and} \quad A : U \to A \quad \text{and}$$

$$A \circ A = A = 1_A.$$

We thus speak of these A's also as retractions. We can write:

$$(U \to U) = \lambda f \lambda x.\ f(x),$$

and it is thus easy to verify now that $(U \to U)$ is a retraction of U. As U is in fact the identity function, the reinterpretation *via* U of the type-free calculus will obviously translate every term into itself.

I just note in writing down the definition of the c.c.c., I forgot to define $\mathbb{1}$ – because it is so dull, I suppose! For this we have to map everything onto a constant:

$$\mathbb{1} = \lambda u\ \lambda x.x, \quad \text{and}$$
$$0 = \mathbb{1}.$$

I think the calculations suggested provide an argument that type-free λ-calculus takes *second place* to typed λ-calculus – foundationally speaking. Type-free domains are *special kinds* of types. As I have said before in other writings, to get $(U \to U)$ inside U, we have to pass to an *infinite type*. I thought this was made very clear in the so-called D_∞-construction. The category of *continuous lattices and continuous functions* is a c.c.c. Starting with any domain D_0, in that category the sequence of types D_n where $D_{n+1} = (D_n \to D_n)$ has a certain limit D_∞ with D_0 (and all the D_n's) as retracts, and with $(D_\infty \to D_\infty)$ not only a retract but an isomorph of D_∞. That is one choice of an U, and I showed many variations are possible for other type-free domains U in this one category.

We hasten to note that in the c.c.c. of *sets and arbitrary functions*, a non-trivial domain U with $(U \to U)$ a retract is *impossible* (by cardinality considerations). This means that not all c.c.c. lead directly to interpretations of the type-free theory. Hence, we must conclude, *the typed theory is the more general one, and the prior one.*

Such a conclusion will not be welcome, however. The type-free theory from our experience *seems* general enough. Even though we have shown two good ways of relating the two kinds of theories, we would like something more. We do not want just *some* c.c.c. related to a given type-free theory, but we would like to find a relation that achieved *any desired* c.c.c., provided we cook up the type-free one properly. This problem is the topic of the next section.

Before we turn to this new relationship, a word about *models* of the type-free calculus would be to the point. There is considerable discussion of the notion of a model in Hindley and Longo (1980) and Barendregt (1980) (where other references are given). We should state how this all fits in with the present view.

When presenting a *theory* in the usual λ-notation, free variables are permitted as well as full use of the rule (ξ). *But*, when thinking of elements (relative to a theory) only terms (or better: equivalence classes of terms) *without free variables* should be considered. As is known from many examples, there may not be enough of them. This can of course be so even if w allow in our language many non-logical constants. What does "enough" mean? Well, if f and g are closed terms, it may be that $f(a) = g(a)$ is provable for *all* closed a, but $f = g$ is no provable. The fact that this happens for some theories should come as no surprise. (For the explicit examples consult Baren dregt (1980).)

The remedy is to adjoin indeterminates (constants without new axioms) until "enough" is reached. (A proof is also found in Barendregt (1980).) As with the typed calculus, every theory has a model which satisfies exactly the same equations as are provable in the theory (one might call it a conservative model).

The notion of a λ-model has not struck people as quite satisfactory because the extensionality principle in the "enough" clause is not very algebraic. A suggestion of mine is mentioned in the cited references, but I think it would be useful to recast the idea in the light of the present discussion.

In typed λ-calculus, the categorical formulation is one way of eliminating all use of variables. In type-free λ-calculus, the usual plan is to use the combinators – and the plan leads to awfully long formulae. Let us not try to give a variable-free formulation, but talk in terms of first-order models. What is unalgebraic in the model definition is the λ-operator, since a bound variable is of the essence of the use of λ. So let us replace λ by the combinators in the usual way. We take S and K as primitive, and a λ-model is (at least), a structure of the form $\langle U, \cdot(\cdot), S, K\rangle$, with a domain, a binary operation and two distinguished constants. The problem is: *what are th axioms?* Clearly we want:

$$(*)\begin{cases} K(x)(y) = x, \text{ and} \\ S(u)(v)(x) = u(x)(v(x)), \end{cases}$$

s usual. But these are not sufficient to express extensionalty, which in λ-notation reads:

$$\forall x.\ \tau = \sigma \rightarrow \lambda x.\tau = \lambda x.\sigma$$

f we convert out the variable x, we are tempted to write:

$$\forall x.\ f(x) = g(x) \rightarrow f = g.$$

ut this is too strong. (It corresponds to $U = (U \rightarrow U)$ rather han the weaker: $(U \rightarrow U)$ is a retract of U.) If we wrote:

$$\forall x.\ f(x) = g(x) \rightarrow \lambda x.f(x) = \lambda x.g(x),$$

e statement would at least be correct – even if containing e unwanted λ. Well, we just have to define this λ in terms S and K. Introduce the standard definitions:

$$I = S(K)(K)$$

$$B = S(K(S))(K).$$

en (with λ-notation)

$$\lambda x.\ f(x) = B(I)(f).$$

the desired axiom now reads

$$(**)\quad \forall x.\ f(x) = g(x) \rightarrow B(I)(f) = B(I)(g)$$

We are not quite done, however. We want S and K to correspond to λ-expressions (eventually), so we need an axiom which kes them suitably unique. Now we note intuitively that

$$\lambda x_0\lambda x_1\ldots\lambda x_{n-1}.f(x_0)(x_1)\ldots(x_{n-1}) = B^n(I)(f).$$

us, what we need to say is:

$$(***)\begin{cases} S = B(B(B(I)))(S), \text{ and} \\ K = B(B(I))(K) \end{cases}$$

To see that (*), (**), and (***) are adequate, we note firs that

$$B(I)(f)(x) = f(x)$$

by (*). From (**) it then follows that

$$B(I)(B(I)(f)) = B(I)(f).$$

This means we can reformulate (**) as:

$$(**_1)\quad f = B(I)(f) \wedge g = B(I)(g) \wedge \forall x. f(x) = g(x) \;\rightarrow\; f = g.$$

[This does not seem to be equivalent to (**) unless we have the equation about B(I)(B(I)(f)) just noted – the retraction equation.] We now generalize $(**_1)$ to n variables:

$$(**_n)\begin{cases} f = B^n(I)(f) \wedge g = B^n(I)(g) \wedge \\ \forall x_0, x_1, \ldots, x_{n-1}. f(x_0)(x_1)\ldots(x_{n-1}) = g(x_0)(x_1)\ldots(x_{n-1}) \\ \rightarrow f = g. \end{cases}$$

If we prove this, then by (***) we see that the original axiom (*) uniquely determine S and K; further we have the uniqueness required to define $\lambda x.\tau$ for any term (cf. the references cited

To establish $(**_n)$, we need some lemmas. From (*) and (*** and the definitions, we can easily prove:

$$S(u) = B^2(I)(S(u))$$
$$S(u)(v) = B(I)(S(u)(v))$$
$$B(u) = S(K(u)).$$

We then establish for $n \geq 1$:

$$B(I)(B^n(I)(f)) = B^n(I)(f),$$

because $B^n(I)(f)$ has the form S(u)(v). Suppose then that, e.g.

$$\forall x,y,z.\ f(x)(y)(z) = g(x)(y)(z).$$

By (**) we find:

$$\forall x,y.\ B(I)(f(x)(y)) = B(I)(g(x)(y)).$$

This can be rewritten as

$$\forall x,y.\ B^2(I)(f(x))(y) = B^2(I)(g(x))(y).$$

But again by (**) we find:

$$\forall x.\ B(I)(B^2(I)(f(x))) = B(I)(B^2(I)(f(x))).$$

By the lemma, drop the B(I). Throw on another B, use (**), drop off the B(I), and get:

$$B^3(I)(f) = B^3(I)(g).$$

The method is perfectly general and proves $(**_n)$.

The import of this axiomatization is that B(I) is the retraction of the universe U onto $(U \rightarrow U)$ and $B^n(I)$ retracts onto

$$\underbrace{(U \rightarrow (U \rightarrow (U \rightarrow \ldots\ (U}_{n \text{ times}} \rightarrow U)\ldots))).$$

e need (***) to show, e.g.:

$$S : U \rightarrow (U \rightarrow (U \rightarrow U))).$$

e need $(**_n)$ to show that the maps in these function spaces re *uniquely* determined by their values.

We have just been speaking in terms of models; but the calulations just carried out were formal. The axiomatic question, hen, is: what is the relationship between the equational theries and the first-order theories? We shall now see the reation is a close one – even if the logic is allowed to go beond the first order.

. A RÔLE FOR INTUITIONISTIC LOGIC.

The (rather cheap) method of adjoining indeterminates proves hat every typed or untyped theory of λ-calculus has an exteninonal model. This can also be put as a conservative extension esult for theories: a λ-theory is an equational theory, and very such equational theory can be expanded to a first-order theory *without* forcing any new equations on us. In the untyp case, the style of first-order theory is that of axioms (*), (**), (***) of the previous section. These are the "logical" axioms (i.e. common to all such tneories); the non-logical ax- ioms would be all the equations between closed λ-terms demand by whatever equational theory we started with - and these spe cial equations could involve special "non-logical" constants.

In the typed case, we would get a many-sorted theory with sort for each domain in the given category. As we have alrea pointed out, an untyped theory can always be reformulated as typed theory by the method of retracts. So we now concentrat on typed theories - that is to say, cartesian closed categorie

But the writing down of *first-order* theories is not all th interesting: we clearly have nice axioms for a theory of fun tions, but first-order theories do not impress us as being ver categorical. Such theories do not really capture the idea of the "arbitrary" function. We began our discussion with set theory, where the intention was that function spaces did real contain "all" functions - they did not just appear as an "alg bra" of functions. Leaving aside for the moment the (philoso- phical) question of whether the desire for the ALL is a ratio al one, we can ask the (formal) question of whether there is a conservative extension result for *higher-order* theories. Surprisingly, category theorists have known the answer for so time.

Now we cannot hope to embed the theory of a typed λ-calcul into a *classical* higher-order theory with a full comprehensio axiom of the form

$$\forall x : A\ \exists! y : B.\varphi(x,y) \to \exists f : A \to B \forall x : A.\ \varphi(x,f(x)).$$

Because in higher-order logic we can prove Cantor's Theorem which implies that the only type U which has a surjective map $j : U \to (U \to U)$ is the one-element type. Thus, if a typed the ory had such a type (and we know many), then the adding of the

tandard higher-order axioms (where we construe $(A \to B)$ as the otal function space of *all* functions) would not at all be con- ervative. Something else has to be tried, and the answer is igher-order *intuitionistic* logic.

As we shall now have to consider more than one category, let call our given c.c.c. the category C. To fix ideas, the onstructions to be carried out will be done in ordinary set eory - with classical logic! The models obtained, however, ll only satisfy intuitionistic logic. The obvious lack of armony can be repaired, but it would take too much explanation ere. Moreover, we are also going to assume that the given tegory C is a *set*. This is not much of a restriction, since were thinking of C as a *theory* and usually a theory has a mited number of symbols in any case.

Before saying where the intuitionistic logic comes from, let give the *construction*. Let S be the category of all sets d arbitrary functions; we know it is a c.c.c. The construc- on we need here is the well-known one of the *functor category* ^{op} of all contravariant functors from C into S with the nat- al transformations as the maps - full definitions follow. e *result* is that the functor category is a model for higher- der intuitionistic logic, in particular it is a c.c.c.; more- er the original category C has a full and faithful embedding $S^{C^{op}}$, and this shows the conservative extension property. much for the outline of the method, now for the details. edless to say this represents a *very early* chapter in topos eory; it should be more widely known.

What is a *contravariant functor?* It is a mapping $F : C \to S$ at associates to every domain A of C a set $F(A)$ of S and to ery map $f : B \to A$ of C a function $F(f) : F(A) \to F(B)$ (and e the change of order!) so that:

$$F(1_A) = 1_{F(A)}, \text{ and}$$

$$F(f \circ g) = F(g) \circ F(f),$$

provided $f : B \to A$ and $g : C \to B$ in $\mathcal{C}$. It was one of the major early insights of category theory to see that the functors form a category in themselves. What is needed is a definition of transformation between functors. We call such maps $\nu : F \to G$ "natural transformations" for reasons explained in category theory books.

What is a *natural transformation* $\nu : F \to G$? It is an association with every domain A of $\mathcal{C}$ of a function $\nu_A : F(A) \to G(A)$ so that whenever $f : B \to A$ in $\mathcal{C}$, then the following diagram commutes in $\mathcal{S}$:

$$\begin{tikzcd} F(A) \arrow[r, "F(f)"] \arrow[d, "\nu_A"'] & F(B) \arrow[d, "\nu_B"] \\ G(A) \arrow[r, "G(f)"] & G(B) \end{tikzcd}$$

This means $\nu_B \circ F(f) = G(f) \circ \nu_A$. An example will help explain this.

For each C of $\mathcal{C}$, let

$$H_C(A) = \{h \mid h : A \to C\}$$

and if $f : B \to A$ in $\mathcal{C}$, let $H_C(f)$ be the map taking $h \in H_C(A)$ into $h \circ f \in H_C(B)$. It is easy to show H_C is a (contravariant) functor. It is often called the *representable functor* (corresponding to C), and we shall see that it is very "representative".

Now let $g : C \to D$ in $\mathcal{C}$. There is a natural transformation $H_g : H_C \to H_D$; because, for each $h \in H_C(A)$ we can map it to $g \circ h \in H_D(A)$, naturally. The composite map for $f : B \to A$ takes $h \in H_C(A)$ into $g \circ h \circ f \in H_D(B)$, and there are two equal ways to calculate it owing to the associativity of composition (in $\mathcal{C}$); that is why the necessary diagram commutes.

Not only are the H_C pleasant functors with cooperative natural transformations between them, but the by now classic *Yoneda Lemma* proves for us that the *only* natural transformations $\nu : H_C \to H_D$ are those of the form H_g for some $g : C \to D$. If we remark that if $k : D \to E$, then

$$H_k \circ H_g = H_{k \circ g},$$

this shows us that $H : \mathcal{C} \to \mathcal{S}^{\mathcal{C}^{op}}$ is a (*covariant*) functor between these categories. Of course H_C uniquely determines C and H_g determines g, so we conclude from this and the *Yoneda Lemma* that H is a full and faithful embedding of $\mathcal{C}$ into the functor category (all of this on p.2 of Johnstone (1977)!).

All of this discussion is "abstract nonsense" in the sense that its validity is perfectly general for *any* category $\mathcal{C}$. If we assume that $\mathcal{C}$ is a c.c.c., then we can say more. The point is that $\mathcal{S}^{\mathcal{C}^{op}}$ is a very powerful category. For example, it is always a c.c.c. even if $\mathcal{C}$ is not. The cartesian closed structure of the functor category is obtained through the following definitions.

Before getting down to details, however, some more-vivid terminology might help. Think of a functor U in $\mathcal{S}^{\mathcal{C}^{op}}$, following Lawvere, as a "variable domain". That is to say, for each $A \in \mathcal{C}$ we have an associated domain (set) $U_A = U(A)$. The maps $f : B \to A$ in $\mathcal{C}$ give us transitions between "stages" A and "later" stages B; and each such transition "restricts" elements in U_A to elements in U_B "along" the map f. To save writing, let us set $a{\upharpoonright}f = (Uf)(a)$ when the functor U is understood. That U is indeed a functor comes down to these equations:

$$a{\upharpoonright}1_A = a \quad \text{and} \quad (a{\upharpoonright}f){\upharpoonright}g = a{\upharpoonright}(f \circ g),$$

where $f : B \to A$ and $g : C \to B$ in $\mathcal{C}$. To define a functor U, then, we just have to give the domains and the restrictions. For example the unit functor $\mathbb{1}$ has $\mathbb{1}_A = \{0\}$, a one-point set, and all restrictions constant $0 \upharpoonright f = 0$. And all natural transformations into $\mathbb{1}$ are constant.

Now suppose U and V are two functors. We define $U \times V$ so that for all A in $\mathcal{C}$

$$(U \times V)_A = U_A \times V_A$$

and whenever $a \in U_A$ and $b \in U_A$ and $f : B \to A$, then

$$(a,b) \upharpoonright f = (a \upharpoonright f, b \upharpoonright f).$$

(Note that the restriction symbol above is used in three different senses.) The natural transformations $p : U \times V \to U$ and $q : U \times V \to V$ have obvious pointwise definitions (e.g. $p_A = p_{U_A V_A} : U_A \times V_A \to U_A$) and they clearly commute with restrictions. Similarly, if $\mu : W \to U$ and $\nu : W \to V$ are given natural transformations, then $\langle\mu,\nu\rangle : W \to U \times V$ is also defined pointwise:

$$\langle\mu,\nu\rangle_A = \langle\mu_A,\nu_A\rangle : W_A \to U_A \times V_A.$$

Again it is obvious that these maps commute with restrictions, so $\langle\mu,\nu\rangle$ is natural. As all of this is pointwise; the verification of such equations as $p \circ \langle\mu,\nu\rangle = \mu$ is easy.

Again suppose U and V are given. In defining $(U \to V)$, we cannot be quite as pointwise. That is, $(U \to V)_A$ cannot be taken simply as the set of functions $(U_A \to V_A)$, the function space in sets. The reason, roughly, is that when we have a function at *one* stage, we also have to know how it restricts at *later* stages; a simple mapping from U_A into V_A does not give us enough information for that. When $f : B \to A$, restriction on U_A maps into U_B; this is the wrong direction for us to be able to pass from an arbitrary function defined on U_A to one defined on U_B. So an element of $(U \to V)_A$ has to be a whole *family* of functions

$$\varphi_f : U_B \to V_B ,$$

one for each $f : B \to A$. (Note: A is fixed, f and B are variable.) Moreover, we must assume that all is harmonious with restrictions: $c \upharpoonright g = \phi_{f\circ g}(b \upharpoonright g)$ whenever $c = \varphi_f(b)$, for $b,c \in U_B$ and $g : C \to B$ in $\mathcal{C}$. In words, φ_{1_A} is the "present" function; while φ_f is what becomes of it in the "future", supposing time evolves along f. Now families φ of this kind in $(U \to V)_A$ have to be restricted. By what we just said, the following is more or less forced upon us:

$$(\varphi \upharpoonright f)_g = \varphi_{f\circ g}$$

where $f : B \to A$ and $g : C \to B$, so that $\varphi \upharpoonright f$ is a family in $(U \to V)_B$.

That defines $(U \to V)$ as a functor. To have $\mathcal{S}^{\mathcal{C}^{op}}$ be a c.c.c., certain maps ε and Λ are, alas, still required. The evaluation map $\varepsilon : ((V \to W) \times V) \to W$ is fortunately rather clear (as a natural transformation). Suppose $\varphi \in (V \to W)_A$ and $a \in V_A$. Then

$$\varepsilon_A(\varphi,a) = \varphi_{1_A}(a),$$

so $\varepsilon_A : ((V \to W)_A \times V_A) \to W_A$. If $f : B \to A$, then

$$\begin{aligned}
\varepsilon_A(\varphi,a) \upharpoonright f &= \varphi_{1_A}(a) \upharpoonright f \\
&= \varphi_f(a \upharpoonright f) \\
&= (\varphi \upharpoonright f)_{1_B}(a \upharpoonright f) \\
&= \varepsilon_B(\varphi \upharpoonright f, a \upharpoonright f)
\end{aligned}$$

This proves that ε is natural.

Next, suppose $\psi : U \times V \to W$ is natural. Define $\Lambda\psi : U \to (V \to W)$ by

$$(\Lambda\psi)_A : U_A \to (V \to W)_A$$

where for $a \in U_A$, $b \in V_B$, and $f : B \to A$ we have:

$$(\Lambda\psi)_A(a)_f(b) = \psi_B(a \upharpoonright f, b).$$

To show $\Lambda\psi$ is natural, we must calculate:

$$\begin{aligned}((\Lambda\psi)_A(a) \upharpoonright f)_g(c) &= (\Lambda\psi)_A(a)_{f \circ g}(c)\\ &= \psi_C(a \upharpoonright f \upharpoonright g, c)\\ &= (\Lambda\psi)_B(a \upharpoonright f)_g(c)\end{aligned}$$

for $a \in U_A$, $f : B \to A$, $g : C \to B$, $c \in U_C$. It follows that

$$(\Lambda\psi)_A(a) \upharpoonright f = (\Lambda\psi)_B(a \upharpoonright f).$$

We have to leave to the reader the verification of the two basic equations of c.c.c.'s involving ε, Λ, p and q. As there was only one way that the definitions could be written, the verification is quite mechanical, however.

As I said before, the functor category is "powerful", and indeed it is much more than a c.c.c. For instance, we can define the analogue of the *power set* for arbitrary functors. For any U, let $(\mathcal{P}U)_A$ be the collection of all families S_f indexed by $f : B \to A$ where $S_f \subseteq U_B$ and such that $b \upharpoonright g \in S_{f \circ g}$ whenever $b \in S_f$ and $g : C \to B$ in $\mathcal{C}$. Restriction is defined by

$$(S \upharpoonright f)_g = S_{f \circ g}.$$

The significance of the power operator will become clear when we speak about higher-order logic.

Having seen why the functor category is a c.c.c., it is good to pause a moment to appreciate the difference between the elements $a \in U_A$ as sets and the "elements" of U in the categorical sense. If $\alpha : \mathbb{1} \to U$ is natural, it means that $\alpha_A : \mathbb{1}_A \to U_A$ in $\mathcal{S}$. Let $a_A = \alpha_A(0)$, then $a_A \in U_A$. If $f : B \to A$, then because α is natural we find:

$$\begin{aligned}a_A \upharpoonright f &= \alpha_A(0) \upharpoonright f\\ &= \alpha_B(0 \upharpoonright f)\\ &= \alpha_B(0)\\ &= a_B\end{aligned}$$

This is very strong indeed, since usually if $a \in U_A$ and $f_0, f_1 : B \to A$, there is no reason why $a \upharpoonright f_0 = a \upharpoonright f_1$. So the number of "elements" of U will very likely be rather small. (And, even worse, a_A has to be chosen for *all* A in $\mathcal{C}$.)

In the special case $\sigma : \mathbb{1} \to \mathcal{P}U$ we can simplify the choices out of $(\mathcal{P}U)_A$ even further. Write $S_A = \sigma_A(0)_{1_A}$, then $S_A \subseteq U_A$ for all A in $\mathcal{C}$. Moreover, when $f : B \to A$, then

$$\begin{aligned}S_B &= \sigma_B(0)_{1_B}\\ &= \sigma_B(0 \upharpoonright f)_{1_B}\\ &= (\sigma_A(0) \upharpoonright f)_{1_B}\\ &= \sigma_A(0)_f\end{aligned}$$

This means that $\sigma_A(0)$ is determined from the S_B's. And if they are chosen so that

$$b \in S_B \text{ implies } b \upharpoonright g \in S_C$$

whenever $g : C \to B$, then the σ_A so defined from them provides a natural transformation. Again, we see the elements of $\mathcal{P}U$ are rather special. We can say that elements of a functor provide information about the "global" nature of the functor; but this is far from determining it, for there can be considerable "local" activity that cannot be sensed globally. For example, the sets U_B can be empty for a long "time", only becoming non-empty in the "future". The functor U is non trivial, but it has no global elements.

We should also pause to see why the functor H maps C into a subcartesian closed category of $S^{C^{op}}$ (up to isomorphism). It is easy to check that the functors $H_A \times H_B$ and $H_{A\times B}$ are naturally isomorphic. We also have to do the same for $(H_A \to H_B)$ and $H_{A\to B}$. Consider an element of $(H_A \to H_B)_C$. It is a family of maps

$$\varphi_f : H_A(D) \to H_B(D),$$

for $f : D \to C$. In particular consider the standard maps $p : (C \times A) \to C$ and $q : (C \times A) \to A$. Then $\varphi_p(q) : (C \times A) \to B$. So, since C is a c.c.c., we find $\Lambda\varphi_p(q) \in (H_{A\to B})_C$. In the other direction, let $t : C \to (A \to B)$. Define τ_f for $f : D \to C$ by

$$(\tau_f)(g) = \varepsilon \circ \langle t \circ f, g \rangle$$

where $g : D \to A$. We see this lies in $H_B(D)$. Now

$$\varepsilon \circ \langle t \circ f, g\rangle \circ k = \varepsilon \circ \langle t \circ f \circ k,\ g \circ k\rangle$$

whenever $k : E \to D$. Thus,

$$(\tau_f)(g) \restriction k = \tau_{f\circ k}(g \restriction k).$$

This proves that the family τ_f lies in $(H_A \to B_B)_C$. It has to be left without proof that these two correspondences are inverse to one another and provide a natural isomorphism.

Well, this is a rather heavy construction starting from one little category C. The question: *what does it prove?* Why worry about the functor category? The answer is that the functors give an interpretation of higher-order logic, as we hinted earlier, and now we have to pay up and demonstrate how to construe logical formulae. The idea from topos theory when specialized to the functor category looks very much like Kripke models of intuitionistic logic – except that the "times" form a category C rather than just a partially ordered set, as has often been emphasized by Lawvere (see, e.g. Lawvere (1975)).

To make the logical language more definite, let us think of the domains A in C as being (in a one-one correspondence with) *type symbols*. Introduce *new* type symbols built from the ones in C (the "ground" types) by forming $\mathbb{1}$, $(T \times S)$, $(T \to S)$, $\mathcal{P}T$ for all type symbols. (Note: $A \times B$ in C is being distinguished from the type symbol $A \times B$. But the "meaning" of the symbol $A \times B$ will turn out to be something "isomorphic" to $A \times B$ in C. The trouble is that the domain $A \times B$ does not in itself determine the A and the B; whereas the type symbol does.) We extend the notation H_A to H_T for any type symbol in the obvious way; that is, $H_{T\times S}$ is the product $H_T \times H_S$ in the functor category. This is the first step in treating the functor category as an interpretation of a higher-order theory.

Next we must imagine a logical language with a supply of variables of each type. Atomic formulae will be of these forms: $\bot$; $x = y$, where x and y have the same type; $y = fx$ where f is a constant symbol corresponding to a map $f : A \to B$ in C and x has type A and y type B; $z = (x,y)$ where x has type T, y type S, z type $T \times S$; $z = x(y)$, where z has type S, y has type T, and x type $(T \to S)$; $y \in x$, where y has type T and x type $\mathcal{P}T$. Atomic formulae are then made into compound formulae by the usual constructs: $\Phi \wedge \Psi$, $\Phi \vee \Psi$, $\Phi \to \Psi$, $\forall x.\ \Phi$, $\exists x.\ \Psi$.

Suppose A is a domain of C, Φ is a formula, and s is a valuation of the free variables of Φ. We are going to define what Joyal-Reyes (1980) call the *forcing-satisfaction* relation $A \Vdash \Phi[s]$. The definition here will be in one respect simpler than theirs since the category C carries no topology; in another respect it is more complicated because we have the whole higher-order language. But the adaptation is straight forward. Before we can give the clauses, we must say what kind of a creature s is. We must make s *relative* to A in the first place. So if x has type T, then s(x) is to belong to the set $H_T(A)$. Now here are the clauses:

$A \Vdash \bot\ [s]$	iff	*false*
$A \Vdash x = y\ [s]$	iff	$s(x) = s(y)$
$A \Vdash y = fx\ [s]$	iff	$s(y) = f \circ s(x)$
$A \Vdash z = (x,y)\ [s]$	iff	$s(z) = (s(x),s(y))$
$A \Vdash z = x(y)\ [s]$	iff	$s(z) = s(x)\upharpoonright_{1_A}(s(y))$
$A \Vdash y \in x\ [s]$	iff	$s(y) \in s(x)\upharpoonright_{1_A}$

These were for the atomic cases and the reader should stop and think how the types are supposed to match. For the compound cases we have:

$A \Vdash [\Phi \wedge \Psi][s]$	iff	$A \Vdash \Phi[s]$ and $A \Vdash \Psi[s]$
$A \Vdash [\Phi \vee \Psi][s]$	iff	$A \Vdash \Phi[s]$ or $A \Vdash \Psi[s]$
$A \Vdash [\Phi \rightarrow \Psi][s]$	iff	whenever $f : B \rightarrow A$ and $B \Vdash \Phi[s \upharpoonright f]$, then $B \Vdash \Psi[s \upharpoonright f]$
$A \Vdash \forall x.\ \Phi[s]$	iff	whenever $f : B \rightarrow A$ and $b \in H_T(B)$, then $B \Vdash \Phi[s \upharpoonright f\ (b/x)]$
$A \Vdash \exists x.\ \Phi[s]$	iff	there is an $a \in H_T(A)$ such that $A \Vdash \Phi[s(a/x)]$

In the above, the notation $s(a/x)$ means the valuation is fixed so that a matches x; of course the type of x must be T. By $s \upharpoonright f$ we mean the valuation that matches $s(x) \upharpoonright f$ with each of the relevant variables x. In each case the restriction operation must be made appropriate to the functor H_T, where T is the type of x.

This is so much like Kripke models, the reader will have no problem in showing every intuitionistic quantificational validity Φ is such that $A \Vdash \Phi[s]$ for all A and all appropriate s.

We only have to take care that we remember that some ranges of variables can be empty (that a set $H_T(A)$ may be empty), and so the logic is the so-called "free" logic (cf. Scott (1979) for a discussion).

In order to verify the special axioms of higher-order logic, we need to remark first on what Joyal-Reyes call the "functorial" character of $\Vdash$:

$$\text{if } A \Vdash \Phi[s] \text{ and } f : B \rightarrow A, \text{ then } B \Vdash \Phi[s \upharpoonright f].$$

This, too, is a property familiar from Kripke models. It plays a direct rôle in the verification of the comprehension axiom:

$$\forall u_0,\ldots,u_{n-1}\ \exists x \forall y\ [y \in x \leftrightarrow \Phi]$$

where the free variables of Φ are among $u_0,\ldots,u_{n-1},y$ and x is a new variable not free in Φ of type $\mathcal{P}T$ where T is the type of y.

To show the above valid in the interpretation we only have to show that for every A of $\mathcal{C}$ and for all $b_0,\ldots,b_{n-1}$ in the $H_S(A)$ of the appropriate types S, there is an element $c \in H_{\mathcal{P}(T)}(A)$ such that

$$A \Vdash \forall y[y \in x \leftrightarrow \Phi][s].$$

Here s is the valuation where $s(x) = c$ and $s(u_i) = b_i$. We have to define c. For each $f : B \rightarrow A$, let

$$c_f = \{t \in H_T(B) \mid B \Vdash \Phi[s \upharpoonright f(t/y)]\}$$

The functorial character of $\Vdash$ proves for us that $c \in H_{\mathcal{P}(T)}(A)$. It is now easy to check from the clauses of the definition of $\Vdash$ that at A the above formula is indeed forced.

In a similar way we can verify the functional version of comprehension:

$$\forall x \exists y \forall z[z = y \leftrightarrow \Phi] \rightarrow \exists f \forall x,z[z = f(x) \leftrightarrow \Phi],$$

where we have x of type T, y and z of type S; f of type $(T \to S)$ and y not free in Φ. Again, the functorial character of $\Vdash$ connects with the way we had to define $(H_T \to H_S)$.

We also have to veriry such extensionality properties as:

$$\forall f,g[\forall x,y[y = f(x) \leftrightarrow y = g(x)] \to f = g],$$

$$\forall x,y[\forall z[z \in x \leftrightarrow z \in y] \to x = y],$$

where the variables have to be given the appropriate types. But in defining the function spaces and the powersets in the functor category, we only put in just enough of a mapping or a set to get an appropriate functorial character. Hence, if two such objects are extensionally equal by the formulae above, they will be equal. This has to be spelled out via $\Vdash$, but it is not surprising.

The higher-order axioms for ordered pairs are obvious, and their satisfaction relates at once to the definition of product of functors. As for the embedding of C into the higher-order theory we find

$$\forall x,y[y = fx \leftrightarrow y = gx]$$

is valid if and only if $f = g$ in C. Also, when $h = g \circ f$, we have as valid

$$\forall x,y,z[[y = fx \wedge z = gy] \to z = hx].$$

Further, functions like f are well defined:

$$\forall x\ \exists y\ .\ y = fx$$

There are many principles of identity that should be mentioned, but we will not write them down here. Among them we would also find the statements that there is a unique element of type $\mathbb{1}$ and all maps of type $(T \to \mathbb{1})$ are constant. (Perhaps the constant 0 should figure in the language, but it is not all that essential.)

As for questions of uniqueness, if the sentence

$$\forall x \exists y \forall z[z = y \leftrightarrow \Phi]$$

is valid (i.e. forced at all A), then provided x and y have ground types B and C, respectively, there is an $f : B \to C$ in C such that

$$\forall x,z[z = fx \leftrightarrow \Phi]$$

is valid too. The validities, then give us an exact picture of C at the level of ground types: the higher-order theory is conservative over C.

But the higher-order intuitionistic theory of the functor category is much more than just a conservative extension; it is a full-blown *higher-order* theory with full comprehension axioms. That is to say, we started out with a category C we regarded algebraically as a theory of functions. Well, the construction of the functor category shows us that we can indeed construe C and its maps as normal, everyday functions in a normal, everyday higher-order logic. This works as long as we agree to keep our logic intuitionistic. But experience with intuitionistic logic really shows that the system is a natural one and that it leads to very, very interesting theories. Even if C is a c.c.c., we can show that the embedding of C in the higher-order logic preserves all the cartesian closed structure, so that the function spaces in C really become spaces of all possible functions in the higher-order theory. The principles of λ-calculus are thus consequences of the standard logical axioms. This seems to me to establish complete harmony between (intuitionistic) logic and (typed) λ-calculus.

The next step in this investigation would be to see what other properties of the higher-order logic could be enforced and still preserve the conservative extension over the given category C. The functor category is just a very first stage of the investigation: In topos theory the categories of *sheaves* result from putting a kind of modal operator into the logic, and making a reinterpretation of the logical connectives and quantifiers. The passage from $S^{C^{op}}$ is one of finding c.c.c. as cartesian closed subcategories of the functor category. There are many of them and many still contain C as a cartesian closed subcategory. So, there is much to look for, and - I am sure - much left to be discovered of definite logical interest.

5. TYPE-FREE DOMAINS REVISITED

Having made *any* given c.c.c. C "honest" as a theory of functions in higher-order logic, we can conclude from the method of retracts of Section 3 that any type-free λ-theory can similarly be made honest. Intuitionistic logic is very tolerant of types U where $(U \to U)$ is a retract; so tolerant in fact that *any* λ-theory can be embedded in a suitable higher-order logic. Self-application is no longer odd: it is something that may very well turn up when we weaken our logic to be intuitionistic but still require that functions spaces like $(U \to U)$ contain *all* functions.

This provides a certain kind of rescue for the type-free calculus, but the move fails to give it a universal rôle: the creators of the type-free theory hoped that such a universe U could be thought of as containing all the functions there were. We shall not try to go so far in the present context, but various constructions can be used to show that not only is it possible to have *one* such type-free domain, but it is always possible to find them being richer and richer and containing more and more functions. Only a sketch of the construction can be given here.

Suppose, for the sake of illustration, we have some types A, B, C that we happen to like, and that we are interested in the functions between them - possibly also in functions of the type $(A^2 \times B) \to C$, and similar multivariable types. We could probably work up a c.c.c. containing A, B, C and these functions, but the straight-forward construction would contain no type-free domains (cf. the category of sets and maps in ordinary logic). We need a new method. My first approach is to use the idea of continuous lattices. I do not want to go into a lot of detail (cf. Gierz et al. (1980) for just such details), but there is an easy definition that can be invoked at least to make the statements precise.

We shall employ what are not technically lattices but "half" lattices without unit elements (top elements). Fortunately we do not have to go into a long list of definitions, since I have been able to characterize them neatly as special *topological spaces*. They are in fact T_0-spaces (i.e. spaces where points are uniquely determined by their neighborhoods) D such that whenever X is a dense subspace of a topological space Y, and $f : X \to D$ is a continuous function, then f has a continuous extension $\bar{f} : Y \to D$. What I proved is that the category of such spaces D together with continuous maps between them is a c.c.c. (There are very many intriguing c.c.c.'s related to the category of topological spaces!) Let us employ the temporary name "injective" for these spaces.

As an example of injective spaces, consider one of our given types A, which for simplicity we construe just as a *set*. The injective space A_* corresponding to A results from adding one new point $*$. Or, if classical logic is not assumed, we take A_* as the space of subsets of A with *at most one element*. The topology is generated by sets of the form $\{x \in A_* \mid a \in x\}$ where $a \in A$. Thus, a function $f : X \to A_*$ is *continuous* iff $\{x \in X \mid a \in f(x)\}$ is *open* in X for each $a \in A$. Now if $X \subseteq Y$ is a dense subspace, we have only to define

$$\bar{f}(y) = \bigcup \{\{a \in A \mid \forall x \in N \cap X.\ a \in f(x)\} \mid y \in N\},$$

where N ranges over the open sets of Y. Because every non-empty open set has a non-empty intersection with X, it follows that $\bar{f} : Y \to A_*$. To prove $\bar{f}$ continuous, we remark that $\{y \in Y \mid a \in \bar{f}(y)\}$ is the largest open subset of Y whose intersection with X gives $\{x \in X \mid a \in f(x)\}$. It is also easy to calculate that $\bar{f}$ extends f. So A_* is injective. Note, too, that A may be regarded as a dense subspace of A_* if we map a to $\{a\}$. Hence, every function $g : A \to B$ has a unique counterpart $g_* : A_* \to B_*$ so that the "restriction" of g_* to A gives g back again (indeed $g_*(\{a\}) = \{g(a)\}$). This really means that the $*$-construction is a faithful functor from the category of our sets A, B, C into the category of injective spaces and continuous functions.

But now we can apply my construction of λ-calculus models to find an injective space U which, in the category of injective spaces, has $(U \to U)$ as a (continuous) retract *and* in addition has A_*, B_*, and C_* as retracts. (In fact, for those who know the method, we solve the domain equation

$$U = A_* \times B_* \times C_* \times (U \to U),$$

where of course a factor is always a retract of a product; because in the category of injective spaces the one-point space is a retract of every space.) This idea could be extended to obtain any given *set* of injective spaces as retracts of a *single* space U.

Next we invoke the plan of the previous section using as the category C the retracts of U (and continuous functions), which we can regard as a small category (as a set). The functor category has all higher-order logic as well as a full and faithful picture of C. We are definitely going to take advantage of the higher-order structure in looking at subtypes of the functors H_V where V is a domain in C - more precisely, we will look at the category generated by certain of these subtypes or subfunctors in the functor category.

In the first place, consider H_{A_*}. Let K_A be the functor here $K_A(V)$ is just the set of all continuous functions : $V \to A_*$ where for some $a \in A$, we have $a \in f(x)$ for all $x \in V$. (The retracts of U are simply being regarded as injective paces, and we do not distinguish between A_* as a constructed pace based on the set A and as a retract of U.) The restriction operation $\restriction f : K_A(V) \to K_A(W)$ is the one for the functor H_{A_*} with the domain cut down: K_A is a *subfunctor* of H_{A_*}. We an think of the maps in $K_A(V)$ as being the *constant* maps with alues in A. So what then can we have for natural transformaions $\nu : K_A \to K_B$, the maps in the functor category? Well, magine one. Now if $a \in A$, we can take the appropriate contant map $k_a \in K_A(\mathbb{1})$, where $\mathbb{1}$ is the one-element space. Then $(k_a) \in K_B(\mathbb{1})$. But this too is a constant map and determines a nique $b \in B$; so ν defines a function $|\nu| : A \to B$. And, since very constant map factors through the one-element space $\mathbb{1}$, the p $|\nu|$ uniquely determines ν. But *any* map from A into B can made to turn up in this way by trivially fooling around with nstants. We conclude, therefore, that K_A as a functor - of r given sets A - is a full and faithful embedding.

What have we done? First, starting in sets - or perhaps, tter, in higher-order logic - we found (or gave ourselves) category of types we liked. To be more definite, they could ve been unioned together so they were all subtypes (subsets) a single set, V, say. We then embedded faithfully this tegory of sets and maps into the category of injective spaces a the very elementary A_* construction. Of course, the spaces are very special, so the "universal" space U with $(U \to U)$ as retract is much more messy than V_*. But the category of reacts of U contains all the maps between the A_*. Finally this tegory of retracts is fully and faithfully embedded in the

the functor category. The latter has the advantage of subtypes in profusion, so we were able to recapture the *original* category of subsets of V as a full and faithful subcategory of the functor category.

And having done all this, what have we bought? Well, the (pictures of) the A's were subtypes of the A_*'s which are both retracts and subtypes of U, and in the functor category U is a model for the untyped λ-calculus. So that means that starting with our original notion of function, we have – in the logic of the functor category – consistently been able to assume that there are types giving models for the "type-free" λ-calculus, and further, that these types are rich enough to contain our original category in a full and faithful way. In more detail: the new logic allows us to think of A and B as subtypes of U, where $(U \to U)$ is a retract, so that *any* function from A into B is the result of restricting a function in $(U \to U)$ down to the subset A. *Warning*: this does not hold for *all* subtypes of U, the A, B, C were given *in advance* and U was constructed relative to them. Still, this means that even in models for type-free λ-calculus (which can be regarded as ordinary function spaces), we are *not* losing sight of the standard idea of function. To have $(U \to U)$ as a retract of U, the functions have to "bend" a little, but we have kept them "straight" as far as the given A, B, and C go.

We have just shown how a type-free domain U can incorporate given domains as well as the "arbitrary" functions on them. In Engeler (1979) it is shown that λ-calculus models can also incorporate any *algebra*; specifically it is shown that any algebra can be made isomorphic to a subset of the model where the operation is functional application itself. We shall give a proof here using the constructs we have mentioned.

Let A be set. An "algebra" can be regarded as any binary operation $\bullet : A \times A \to A$. A *partial algebra* can be taken to be any *continuous* $\bullet : A_* \times A_* \to A_*$. It is easy to argue that any algebra on A determines a partial algebra on A_*.

Now let the λ-calculus model U be taken so that $U = A_* \times (U \to U)$. We regard elements $x \in U$ as pairs (x_0, x_1). The application operation $f(x)$ on U can be defined as $f_1(x)$ because $f_1 \in (U \to U)$. Now recall that λ-calculus models satisfy the fixed-point theorem; so we can define a map $\rho : A_* \to U$ by the functional equation:

$$\rho(a) = (a, \lambda x{:}U.\ \rho(a \bullet x_0)),$$

here the λ-operator gives an element in $(U \to U)$. This map is ontinuous and one-one into. Now calculate in U:

$$\begin{aligned}\rho(a)(\rho(b)) &= \rho(a \bullet \rho(b)_0) \\ &= \rho(a \bullet b)\end{aligned}$$

o the image of ρ is closed under application, and the resulting applicative subalgebra is *isomorphic* to the given algebra $A_*, \bullet$>.

This result would seem to have a potentially useful implicaion for *non-extensional* models of combinatory algebra: any ich can be embedded in an application-preserving way into an tensional model. This works even if we regard application as *partial* operation. *Warning*: we do *not* obtain a combinatoreserving embedding, however. That is, if the algebra $A_*, \bullet$> has elements S and K, satisfying the usual equations in , we *cannot* conclude that the embedding $\rho : A_* \to U$ will map e S and K of A_* to the "true" S and K of the λ-calculus model . The "functions" in A_* operate only on A_*, which is quite a mited part of U; clearly ρ does not give elements $\rho(a) \in U$ ry broad rôles. But at least we can say that anything that en looks a little like application can be assumed to be apication in a suitable domain.

6. SUMMARY AND CONCLUSIONS

The constructions reviewed and outlined here have been rather lengthy, so it would seem best to summarize the principal conclusions we have reached.

1. *A theory in typed λ-calculus is just the same as a cartesian closed category.*

As was stated, this has been known for well over ten years from the work of Lambek. It should be stressed, however, that category theory achieves a greater generality than the usual logical presentations, because in category theory the type constructions are *axiomatized*. Thus, the types form an "algebra" under the operations $T \times U$ and $(T \to U)$. We need not assume that we always have a "free" algebra of types built out of "ground" types.

2. *In a c.c.c. a reflexive domain provides an interpretation of the "type-free" theory.*

We can call U "reflexive" if $(U \to U)$ is a retract. The last statement is of course obvious. What makes it interesting is:

3. *Every type-free theory is the theory of a reflexive domain in a c.c.c.*

The proof of this result was by the author's method of *retracts*. The use of idempotents in a category as forming a category is well known, but the author believes that he was the first to note that in a c.c.c. we really have a calculus of retracts – especially when there are reflexive domains available.

Then some remarks were made about theories and models and the significance of adding indeterminants (also well known). What might not have been clear from other works was the type-free theory in terms of application, S, and K. Up to that point the theories had been equational; and, though the first-order version (with extensionality) was pleasant, it was not of great philosophical interest since it does not relate the idea of λ-calculus to any broad notion of functions. This desire was taken care of by:

4. *Every c.c.c. can be fully and faithfully embedded in an intuitionistic theory of types with the full (impredicative) power-set construct and function spaces (higher-order intuitionistic logic).*

The domains of the c.c.c. become types in the theory. The word "fully" means that the definable maps between the types all come from maps in the category; "faithfully" means that in the higher-order theory no new equations between these maps are introduced over what we already had in the category. In other words, this is a conservative extension result. It has been known for quite a time in category theory, and the functor category we employed in the construction is one of the very first examples of a *topos*; there must be considerable use possible of more interesting examples of topoi.

However, there was already enough philosophical interest in this easy construction. Namely, it was seen that equational λ-calculus is perfectly consistent with higher-order logic where – provided we only employ intuitionistic logic – we can speak of function spaces in the normal way in type theory. Some people can, if they like, stick to λ-terms and equations; but others can use whatever logical means they like for discussing functions. However, if the logician *proves* in his higher-order theory that a certain property picks out a function $f : A \to B$, then, if A and B are from our given category, this definable f must be given by a standard λ-term. So the logic in that sense gives nothing new, but at least we know that the sense for λ-calculus is exactly that it can always be taken to be talking about functions and *full* function spaces in a higher-order theory.

and they are satisfied in certain topoi. It would be an important next step for λ-calculus to relate these model constructions to interpretations of λ-calculus. The author hopes that the present paper will encourage others to look further.

REFERENCES

Barendregt, H., (1980). "The Lambda Calculus: its Syntax and Semantics", North-Holland, Amsterdam, to appear.

Engeler, E., (1979). Algebras and combinators, *preprint, ETH Zürich.*

Gierz, G., Hofmann, K.H., Keimel, K., Lawson, J.D., Mislove, M., and Scott, D.S., (1980). "A Compendium of Continuous Lattices", Springer-Verlag, Heidelberg, to appear.

Goldblatt, R., (1979). "Topoi: The Categorial Analysis of Logic", North-Holland, Amsterdam.

Hindley, R., and Longo, G., (1980). Lambda calculus models and extensionality, *Math. Logik u. Grundl. d. Math,* to appear.

Johnstone, P., (1977). "Topos Theory", London Math. Soc. Monographs No. 10, Academic Press, London.

Joyal, A., and Reyes, G.E., (1980). Forcing and generic models in categorical logic, *J. Pure and Applied Algebra,* to appear.

Lawvere, F.W., (1975). Continuously variable sets: algebraic geometry = geometric logic, *In* "Logic Colloquium '73", (Eds. H.E. Rose and J.C. Shepherdson), North-Holland, Amsterdam, pp. 135-156.

MacLane, S. (1971). "Categories for the Working Mathematician", Springer-Verlag, Heidelberg.

Plotkin, G. (1972). A set-theoretical definition of application, *Memo,* MIP-R-95, *School of Artificial Intelligence, Univ. of Edinburgh.*

Scott, D.S., (1976). Data types as lattices, *Siam J. Computing,* Vol. 5, pp. 522-587.

Scott, D.S., (1979). Identity and existence in intuitionistic logic, *In* "Applications of Sheaves", Springer-Verlag, Lecture Notes in Math. Vol. 753, pp. 660-696.

Scott, D.S., (1980). Lambda calculus: Some models, some philosophy, *In* "The Kleene Symposium", (Eds. J. Barwise, H.J. Keisler, and K. Kunen), North-Holland, Amsterdam, pp. 381-421.

Szabo, M.E., (1978). "Algebra of proofs", North-Holland, Amsterdam.