Fully abstract models of typed λ-calculi

Theoretical Computer Science 4 (1977) 1–22.

FULLY ABSTRACT MODELS OF TYPED λ-CALCULI

Robin MILNER

Computer Science Department, Edinburgh University, Edinburgh, Scotland

Communicated by Maurice Nivat
Received October 1975
Revised June 1976

Abstract. A semantic interpretation $\mathscr{A}$ for a programming language L is fully abstract if, whenever $\mathscr{A}[\![\mathscr{C}[M]]\!] \sqsubseteq \mathscr{A}[\![\mathscr{C}[N]]\!]$ for two program phrases M, N and for all program contexts $\mathscr{C}[\ ]$, it follows that $\mathscr{A}[\![M]\!] \sqsubseteq \mathscr{A}[\![N]\!]$. A model $\mathscr{M}$ for the language is fully abstract if the natural interpretation $\mathscr{A}$ of L in $\mathscr{M}$ is fully abstract.

We show that under certain conditions there exists, for an extended typed λ-calculus, a unique fully abstract model.

1. Introduction

We are concerned with the problem of finding, for a programming language, a denotational semantic definition which is not over-generous in a certain sense. We can describe quite informally what we mean by 'over-generosity'. Suppose that L is the set of well-formed phrases of the language. Often it is the case that not every such phrase is a whole program; for example, a procedure declaration may not be one, though of course may be part of one.

Now a denotational semantic definition of L consists of a semantic domain D of meanings, and a semantic interpretation $\mathscr{A} : L \to D$. We assume that we are mainly interested in the semantics of (whole) programs. Denote by $\mathscr{C}[\ ]$ a program context — that is, a program with a hole in it, to be filled by a phrase of some kind. One desirable property of $\mathscr{A}$ is that for all phrases M and N (of the right kind) we have $\mathscr{A}[\![\mathscr{C}[M]]\!] = \mathscr{A}[\![\mathscr{C}[N]]\!]$ whenever $\mathscr{A}[\![M]\!] = \mathscr{A}[\![N]\!]$. This is not hard to achieve, particularly if $\mathscr{A}$ is given as a homomorphism. But it is unfortunate if for some M and N such that $\mathscr{A}[\![M]\!] \neq \mathscr{A}[\![N]\!]$ it nevertheless holds for *all* program contexts that $\mathscr{A}[\![\mathscr{C}[M]]\!] = \mathscr{A}[\![\mathscr{C}[N]]\!]$; it means that $\mathscr{A}$ distinguishes too finely among nonprogram phrases.

The reason for describing this situation as 'over-generous' is that it typically arises when there are many objects in D which cannot be realized (i.e. denoted by a phrase). For example, $\mathscr{A}[\![M]\!]$ and $\mathscr{A}[\![N]\!]$ may be functions which only differ at an unrealizable argument, which can never be supplied to the functions in a program context.

So we wish to find D and $\mathscr{A}$ such that

$$\mathscr{A}[\![M]\!] \sqsubseteq \mathscr{A}[\![N]\!] \quad \text{iff} \quad \forall \mathscr{C}[\ ].\, \mathscr{A}[\![\mathscr{C}[M]]\!] \sqsubseteq \mathscr{A}[\![\mathscr{C}[N]]\!]$$

(we use $\sqsubseteq$ in place of $=$ since we shall always have a partial order over D); we call such a semantic definition *fully abstract*. In [3] we said that $\mathscr{A}$ was fully abstract w.r.t a given operational semantics if, in addition, $\mathscr{A}$ agreed with the latter (on whole programs) but here we are concerned with full abstraction as an intrinsic property of $\mathscr{A}$ and D. In fact, in this paper we are concerned with extensional models of typed λ-calculi (or equivalently, of typed combinations built from the combinators **S**, **K** and a set of constants) and we discuss fully abstract models rather than fully abstract interpretations $\mathscr{A}$, since we shall assume that the interpretation is the natural one in which **S** denotes $\lambda x.\lambda y.\lambda z.xz(yz)$, **K** denotes $\lambda x.\lambda y.x$ and combination means function application. Thus we have replaced the constraint that $\mathscr{A}$ agrees with a given operational semantics with the constraint that it is a natural interpretation.

Plotkin [4] considered the language PCF — the typed λ-calculus with arithmetic, truth-values and the fixed-point operator. The programs are closed terms of ground type. He showed that the obvious choice of D — that is, all continuous functions at each type — yields an interpretation which is not fully abstract. There are functions in D, like binary disjunction, which can only be realized by computing more than one argument simultaneously, while PCF admits of a purely sequential evaluation method. However as soon as a new function constant, a "parallel" conditional operator, is added to the language, the interpretation becomes fully abstract.

We wish not to extend the language, but to diminish the model. One would like to find a concept of *sequential* continuous function, and to show that the model of sequential functions exists and yields a fully abstract interpretation. But attempts to find such a concept for functions of higher type have hitherto failed as far as I know, though Vuillemin [10] and I independently found (different!) notions of sequentiality for first-order functions. We do not succeed in this here, but the present results are in another direction more general. In Theorem 2 we show how, given ground domains and first-order functions satisfying certain conditions, to construct an extensional model of the typed λ-calculus with the property that all its "finite" elements are definable. This is the property which ensures full abstraction. Moreover, Theorem 3 shows that with a few more (still not very restrictive) conditions this model is unique.

It must be emphasised that the construction is syntactic in nature. In outline, it consists in establishing the appropriate quasi-order over syntactic combinations of the given ground objects and first-order functions (or more precisely, of constants which stand for them) and dividing out by the induced equivalence relation; extra limit points are added to the model to ensure that the resulting partial order is complete under limits of directed sets. Perhaps because of the generality of the construction — it works for such a wide variety of ground domains and given

functions — it is difficult to give a semantic characterization of the models. But I hope that the existence proof will encourage the search for such characterizations in particular cases.

The approach also yields models of the type-free λ-calculus, found by a construction analogous to that of Scott [7], in which the domains D of the models satisfy $D \subseteq [D \to D]$, rather than $D = [D \to D]$, where $[D \to D]$ is the continuous function domain. However, we consider only the typed calculus in this paper. It is not immediately clear what full abstraction should mean for the type-free calculus, since it depends on designating a subset of the language as programs, and in the typed calculus one naturally chooses the terms of ground type to be programs.

2. Models of the typed λ-calculus

In this section we introduce some terminology, relying on some familiarity with typed λ-calculus and combinators.

Assume a set of ground types and the normal hierarchy of functional types. κ ranges over ground types and ρ, σ, τ over all types.

A *model* of the typed λ-calculus consists of:

(i) A set D_σ for each type σ; these are the *domains.*

(ii) For each σ and τ, a two-place application operation $(\cdot\cdot)$ such that for $x \in D_{\sigma\to\tau}$ and $y \in D_\sigma$, $(xy) \in D_\tau$.

(iii) A family of elements S and K in appropriate D_σ such that for all x, y and z in appropriate domains, $Sxyz = xz(yz)$ and $Kxy = x$, where as usual parentheses are omitted and application is taken to be left associative.

A model $\mathcal{M}$ is *extensional* if there is a partial order (*po*) $\sqsubseteq$ on each domain such that $(\forall z.\, xz \sqsubseteq yz) \iff x \sqsubseteq y$. (We shall omit mention of types when whatever we say is to be understood at all appropriate types.)

$\mathcal{M}$ is *monotone* if $x \sqsubseteq y \implies zx \sqsubseteq zy$.

$\mathcal{M}$ is *continuous* if it is monotone, each po $\sqsubseteq$ is a *cpo* — i.e. each directed set X has a lub $\sqcup X$ — and moreover for each such X, z $(\sqcup X) = \sqcup\{zx \mid x \in X\}$.

An element d in a cpo is *finite* if for all directed X, $d \sqsubseteq \sqcup X \implies \exists x \in X.\, d \sqsubseteq x$.

A cpo is *ω-algebraic* if it has at most denumerably many finite elements, and for each $x \in D$ $\{d \mid d$ finite and $\sqsubseteq x\}$ is directed and has lub x.

A cpo is *consistently complete* if each pair x, y having an upper bound has a lub, which we write $x \sqcup y$.

$\mathcal{M}$ is *ω-algebraic* if it is continuous and each domain is ω-algebraic.

$\mathcal{M}$ is *consistently complete* if each domain is consistently complete.

An (n-ary) *first-order function* over ground domains D_κ is one with type of the form $\kappa^{(1)} \to \kappa^{(2)} \to \cdots \to \kappa^{(n)} \to \kappa^{(n+1)}$.

Given ground domains D_κ and a set F of first-order functions over the D_κ, $\mathcal{M}$ is *a model for* F if for each $f \in F$ of type σ, $f \in D_\sigma$.

We are concerned only with extensional models. It is worth remarking that none of the other properties of models (continuity etc.) defined above imply extensionality. However, to avoid tedious repetition we ask the reader to interpret the phrases "model", "continuous model" etc. as meaning "extensional model", "continuous extensional model" etc.; we shall of course always prove extensionality when a model is constructed.

3. Discussion

We proceed to construct, for consistently complete ω-algebraic D_κ and a given set F of continuous first-order functions over the D_κ, an ω-algebraic model $\mathscr{M}$ for F with the property that under a certain condition on F every finite element in $\mathscr{M}$ is λ-definable in terms of the D_κ and F.

It is an immediate corollary that $\mathscr{M}$ is fully abstract. Suprisingly perhaps, the model is not always consistently complete, though we shall not trouble to present the rather pathological counter-example. However, further simple conditions on F ensure consistent completeness, and also ensure that $\mathscr{M}$ is the only continuous fully abstract model up to isomorphism.

The restriction to given *first-order* functions deserves comment. Once the D_κ are fixed, a first-order function may be specified unambiguously. But as long as we have not settled the membership or structure of the higher-order domains, a higher-order function cannot be so specified; it may only be axiomatized — as for example we axiomatize the fixed-point operation Y — and it is then necessary to construct a model in which a function exists (perhaps uniquely) satisfying the axioms. It is no accident that the primitive procedures of a programming language are, almost without exception, first-order; the language designer understands his ground domains but does not usually take the trouble to consider exactly which functions are in his universe of discourse.

Before embarking on the construction, it may help the reader to consider the example of PCF [4] in more detail. Here the ground types are o and ι; D_o (the truth-values) and D_ι (the natural numbers) are given, with their structure, in Fig. 1.

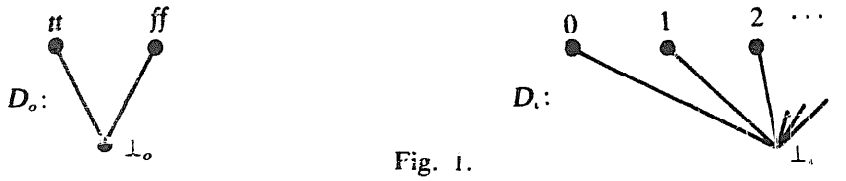

Fig. 1.

The set F in this case is $\{+1, -1, Z, \supset, \supset\}$, where

$+1, -1 : \iota \to \iota \to \iota$ are successor, predecessor,

$Z : \iota \to o$ is 'test for zero', and

$\supset : o \to \iota \to \iota \to \iota$
$\supset : \iota \to o \to o \to o$
} are the usual conditional operations.

We omit the (standard) definitions of these functions; they are monotonic and hence continuous since D_o, D_ι are flat domains.

For this example, F satisfies our conditions and our result yields as a corollary a unique fully abstract semantics for PCF.

In outline, the construction falls into two parts. We first build a 'partial model' in which for each σ we build, not D_σ itself, but a family $\{D_\gamma\}$ of finite domains, where γ ranges over a set of 'partial types' which 'approximate' σ. This model results from a more general construction of monotone models (Theorem 1, Section 4). The point of the partial model is that the elements of each (finite) domain are all definable, and are to be the finite elements of the full model. In the second part (Section 5) we telescope the $\{D_\gamma\}$ for each σ into the limit domain D_σ by a straightforward inverse limit construction.

The partial model is gained by defining an appropriate quasi-order over combinations and then taking the induced equivalence classes of terms as elements of the model. That is, we define a term model; see for example Stenlund [9] or Hyland [1] for discussions of term models of the (type-free) λ-calculus.

It is unfortunate that we have not been able to define our models as *retractions* of the model consisting of all continuous functions, rather than building them up as we do from syntactic material. The retractions would be pleasant and probably useful (they would provide an easy way of discussing the smaller models within a single framework), but they do not exist in general when the ground domains are consistently complete ω-algebraic cpo's — at least if we require (rather naturally) that the element defined by a combination M in the smaller model is the image under retraction of the element which it defines in the larger. Indeed, PCF itself provides a counter-example to this possibility, though we shall not demonstrate this here in detail. It is an open question whether the retractions can be found when the ground domains are lattices — that is, when we restore the "overdefined" element $\perp$. If they can be found, then we shall have some ground for retaining this element.

4. Monotone models

Theorem 1. *Given partially ordered ground domains D_κ and a set F of monotonic first-order functions over the D_κ, there exists a monotone model for F such that every element is λ-definable in terms of F and the elements of the D_κ.*

Before embarking on the proof we need some machinery and a Lemma. We consider typed combinations built from a set $\{\mathbf{c}\}$ of constants of ground type, a set $\{\mathbf{f}\}$ of constants of first-order type, and the combinators $\mathbf{S}$ and $\mathbf{K}$ at all appropriate types. We will consistently present syntactic elements in bold type. We will use M, N, P to range over all combinations, and w to range over ground-type combinations not containing $\mathbf{S}$ or $\mathbf{K}$ — i.e. the word algebra over $\{\mathbf{c}\} \cup \{\mathbf{f}\}$.

We define a reduction relation $\rightarrow$ over combinations as follows:

($\rightarrow$1) $\mathbf{S}MNP \rightarrow MP(NP)$

($\rightarrow$2) $\mathbf{K}MN \rightarrow M$

($\rightarrow$3) If $M \rightarrow M'$ then $MN \rightarrow M'N$

($\rightarrow$4) If $M \rightarrow M'$ then $\mathbf{f}w_1 \cdots w_i M \rightarrow \mathbf{f}w_1 \cdots w_i M'$

where, as always, we tacitly assume that types are respected.

Denote by $\rightarrow^*$ the transitive reflexive closure of $\rightarrow$. It is not hard to show that $\rightarrow$ is monogenic; using this we claim that every M of ground type has under $\rightarrow^*$ a unique normal form w. First, it is a well known property of typed reductions that every reduction sequence terminates; so we need only show that every M containing **S** or **K** (and of ground type) has a reduction. But since the {**f**} are first-order, the leftmost occurrence of **S** or **K** must be in a subterm of form $\mathbf{S}N_1N_2N_3$ or $\mathbf{K}N_1N_2$, which yields a reduction.

Remark. Not *every* combination reduces to its usual normal form; consider $\mathbf{K}(\mathbf{K}c_1c_2)$. This is because we do not include a rule

($\rightarrow$3′) If $N \rightarrow N'$ then $MN \rightarrow MN'$.

The omission of this rule is, as far as I can see, a purely technical device. It ensures that $\rightarrow$ is monogenic, and is also essential for the application of the First Context Lemma which follows.

Definition. An occurrence of N in M' is a *son* of an occurrence of N in M, w.r.t. the reduction $M \rightarrow M'$, if

either (i) N's occurrence in M is not in the redex, and its occurrence in M' corresponds textually,

or (ii) N's occurrence in M is in P_i in the redex $\mathbf{S}P_1P_2P_3$ or $\mathbf{K}P_1P_2$, and its occurrence in M' is the textually corresponding occurrence in a P_i in the contractum $P_1P_3(P_2P_3)$ or P_1.

Definition. A (*ground*) *context* $\mathscr{C}[\ ]$ is a (ground) combination with zero or more holes, to be filled by a combination of appropriate type. The identity context is denoted by $[\ ]$; that is, $[M] \equiv M$.

We shall use $\equiv$ to mean syntactic identity of combinations, and also of contexts.

We now prove a lemma concerning quasi-orders over combinations; it will be needed again in a later section. We remark that every type σ may be expressed in the form $\sigma_1 \rightarrow \cdots \rightarrow \sigma_h \rightarrow \kappa$ for some h and ground κ.

First Context Lemma. *Let* $\prec$ *be a quasi-order over combinations such that*

$$(\prec 1)\ w_i \prec w_i' (i \leq n) \Rightarrow \mathbf{f}w_1 \cdots w_n \prec \mathbf{f}w_1' \cdots w_n'$$

(*where* **f** *is n-ary*)

$$(<2)\ M \to M' \Rightarrow M \lhd\rhd M',$$

where $\lhd\rhd$ *is the equivalence induced by* $<$. *Then for combinations* M, M' *of type* $\sigma_1 \to \cdots \to \sigma_h \to \kappa$ *the following are equivalent*:

(i) $\forall P_i$ *of type* σ_i, $i \leq h$, $MP_1 \cdots P_h < M'P_1 \cdots P_h$,

(ii) $\forall$ *ground* $\mathscr{C}[\ ]$. $\mathscr{C}[M] < \mathscr{C}[M']$.

Proof. (ii) $\Rightarrow$ (i) is immediate. In the other direction the proof is by induction on the length m of the reduction

$$\mathscr{C}[M'] \equiv \mathscr{C}_0[M'] \to \cdots \to \mathscr{C}_m[M'] \equiv w'$$

of $\mathscr{C}[M']$ to normal form, and (for fixed m) the size of $\mathscr{C}[M']$. In the above reduction, each $\mathscr{C}_{i+1}[\ ]$ distinguishes the sons of occurrences of M' distinguished in $\mathscr{C}_i[\ ]$.

The basis $m = 0$ is simple, and we omit details — but note that there are two cases: $\mathscr{C}[\ ] \equiv w'$, or $\mathscr{C}[\ ] \equiv [\ ]$ and $M' \equiv w'$.

For $m > 0$, take the least $i \leq m$, if any, for which

either

$$\mathscr{C}_i[\ ] \equiv [\ ](\mathscr{D}_1[\ ]) \cdots (\mathscr{D}_n[\ ]) \tag{A}$$

or

$$\mathscr{C}_i[\ ] \equiv \mathbf{f}(\mathscr{D}_1[\ ]) \cdots (\mathscr{D}_n[\ ]) \tag{B}$$

for some $n \geq 0$.

Case 1. Neither (A) nor (B) obtains. Then each $\mathscr{C}_j[\ ]$, $j < m$, starts with **S** or **K**, which also heads the redex, and $\mathscr{C}_m[\ ] \equiv w'$. Hence also

$$\mathscr{C}_0[M] \to \cdots \to \mathscr{C}_m[M] \equiv w'$$

whence, by (<2),

$$\mathscr{C}[M] \lhd\rhd \mathscr{C}[M'].$$

Case 2. (A) obtains. Then also (by the same argument as in Case 1)

$$\mathscr{C}[M] \to^* M(\mathscr{D}_1[M]) \cdots (\mathscr{D}_n[M])$$
$$< M'(\mathscr{D}_1[M]) \cdots (\mathscr{D}_n[M]) \text{ by (i)},$$

and of course

$$\mathscr{C}[M'] \to^* M'(\mathscr{D}_1[M']) \cdots (\mathscr{D}_n[M']).$$

Case 2.1 $M' \equiv \mathbf{c}'$. Then $n = 0$, $m = i$, $w' = \mathbf{c}'$ and the result follows.

Case 2.2 $M' \equiv \mathbf{f}N_1 \cdots N_p$. Then each $\mathscr{D}_j[M']$ is ground, and either $i = 0$ and each $\mathscr{D}_j[M']$ is strictly smaller than $\mathscr{C}[M']$ with a reduction of length $\leq m$ to some w_j', or $i > 0$ and each $\mathscr{D}_j[M']$ has a reduction of length $< m$ to some w_j'; in either case by the induction hypothesis $\mathscr{D}_j[M] \to^* w_j < w_j'$ for some w_j. Moreover each $N_k \to^* w_k''$ for some w_k'', so

$$\mathscr{C}[M] < \mathbf{f}w_1'' \cdots w_p'' w_1 \cdots w_m, \quad < \mathbf{f}w'' \cdots w_p'' w_1' \cdots w_n' \quad \text{by } (<1), \diamond \mathscr{C}[M'].$$

Case 2.3 M' starts with **S** or **K**. Then the redex in $\mathscr{C}_i[M']$ is initial, so the leading occurrence of M' has no sons in $\mathscr{C}_{i+1}[M']$; hence

$$\mathscr{C}[M] < M'(\mathscr{D}_1[M]) \cdots (\mathscr{D}_n[M]) \rightarrow \mathscr{C}_{i+1}[M],$$

and the result follows by applying the induction hypothesis to $\mathscr{C}_{i+1}[\ ]$.

Case 3 (B) obtains. Then also (similar argument to Case 1)

$$\mathscr{C}[M] \rightarrow^i \mathbf{f}(\mathscr{D}_1[M]) \cdots (\mathscr{D}_n[M]),$$

and of course

$$\mathscr{C}[M'] \rightarrow^i \mathbf{f}(\mathscr{D}_1[M']) \cdots (\mathscr{D}_n[M'])$$

and we treat the two cases $i = 0$, $i > 0$ as we did in Case 2. □

To prepare for the proof of Theorem 1, we now take as constants

$$\{\mathbf{c}\} = \{\mathbf{c} \mid c \in \cup\{D_\kappa\}\}, \quad \{\mathbf{f}\} = \{\mathbf{f} \mid f \in F\}$$

and we allow ourselves to write $w \sqsubseteq w'$ whenever this is true in the natural interpretation interpreting combination as application. (We can only adopt this interpretation for first-order functions at present, since only the ground domains D_κ are given). We define the quasi-order $\leqslant$ over combinations, with induced equivalence $\triangleq$, by induction on types:

($\leqslant$A) For M, M' of type κ, $M \leqslant M'$ iff $M \rightarrow^* w$, $M' \rightarrow^* w'$ and $w \sqsubseteq w'$.

($\leqslant$B) For M, M' of type $\sigma \rightarrow \tau$, $M \leqslant M'$ iff $\forall N.\, MN \leqslant M'N$.

The properties of $\leqslant$ which we shall need are as follows:

($\leqslant$1) $\leqslant$ is transitive and reflexive,

($\leqslant$2) $\mathbf{c} \leqslant \mathbf{c}'$ iff $c \sqsubseteq c'$,

($\leqslant$3) $\mathbf{f}\mathbf{c}_1 \cdots \mathbf{c}_n \triangleq \mathbf{c}$ iff $fc_1 \cdots c_n = c$,

($\leqslant$4) $\mathbf{S}MNP \triangleq MP(NP)$,

($\leqslant$5) $\mathbf{K}MN \triangleq M$,

($\leqslant$6) $M \leqslant M'$ iff $\forall N.\, MN \leqslant M'N$,

($\leqslant$7) if $M \leqslant M'$ then $NM \leqslant NM'$.

All except the last have straightforward verifications, and we omit the details. For ($\leqslant$7), first verify that $\leqslant$ satisfies the two conditions of the First Context Lemma. Then assume $M \leqslant M'$; by ($\leqslant$6) we obtain $\forall P_i.\, MP_1 \cdots P_m \leqslant M'P_1 \cdots P_m$, whence by the lemma $\forall$ ground $\mathscr{C}[\ ]$. $\mathscr{C}[M] \leqslant \mathscr{C}[M']$. In particular, $\forall P_i.\, NMP_1 \cdots P_n \leqslant NM'P_1 \cdots P_n$ where n is such that the combinations are ground; then ($\leqslant$6) yields $NM \leqslant NM'$.

Proof of Theorem 1. Denote by $[M]$ the $\triangleq$ equivalence class of M, and define $\mathscr{M} = \langle \{D_\sigma\}, S, K, (\cdot\,\cdot), \sqsubseteq \rangle$ by

(a) $D_\sigma = \{[M] \mid M \text{ has type } \sigma\}$,
(b) $S = [\mathbf{S}]$,
(c) $K = [\mathbf{K}]$,
(d) $(xy) = [MN]$ where $M \in x$, $N \in y$,
(e) $[M] \sqsubseteq [N]$ iff $M \leqslant N$,

remarking that (d) is a good definition by ($\leqslant 6$) and ($\leqslant 7$).

To check that $\mathcal{M}$ is the right model, first observe that each 'new' D_κ is order-isomorphic to the given one under $[\mathbf{c}] \leftrightarrow c$. Certainly each member of the new D_κ has form $[\mathbf{c}]$ since every M of ground type is $=$ some $\mathbf{c}$, and (e) together with ($\leqslant 2$) ensures the isomorphism.

Next, each $f \in F$ is faithfully represented by $[\mathbf{f}]$, since by ($\leqslant 3$) and (d)

$$[\mathbf{f}][\mathbf{c}_1] \cdots [\mathbf{c}_n] = [\mathbf{c}] \quad \text{iff } fc_1 \cdots c_n = c.$$

Also the required equations for S and K are easily checked, $\mathcal{M}$ is extensional and monotone by ($\leqslant 6$) and ($\leqslant 7$), and each $x = [M]$ of the model is defined by the combination M. □

Remark. It is not hard to show that the conditions of Theorem 1 determine $\mathcal{M}$ completely (up to isomorphism), but we shall not need this fact.

5. Algebraic models

There is no guarantee that the model of Theorem 1 is continuous; hence in particular the least fixed-point operation may not be present. It is, of course, fully abstract (since every element is definable). It remains an open question whether it is possible in general to extend it to a continuous model by adding limit points, while maintaining full abstraction; we would start with cpos D_κ and continuous functions F.

However, when the D_κ are all finite the model is trivially continuous, since all its domains will be finite by extensionality. We shall use this fact in Theorem 2; as a corollary we then obtain a fully abstract continuous model for F provided that certain projection functions are definable from F, even when the D_κ are infinite.

Let each D_κ now be ω-algebraic and consistently complete, with an enumeration $c_0, c_1, \ldots$ of its finite elements. Define $\Psi_i^{(\kappa)} : D_\kappa \to D_\kappa$ for each $i > 0$ by

$$\Psi_i^{(\kappa)} x = \bigsqcup \{c_j \mid j \leqslant i,\ c_j \sqsubseteq x\}.$$

The lub is guaranteed by consistent completeness; also it is easy to check that $\bigsqcup$ preserves finiteness, so each $\Psi_i^{(\kappa)} x$ is finite. Moreover, $\Psi_i^{(\kappa)} x \sqsubseteq x$ and $\Psi_i^{(\kappa)} \circ \Psi_i^{(\kappa)} = \Psi_i^{(\kappa)}$; that is the functions are projections, and are easily seen to be continuous. (Note that it may occur for some i and some x that $\Psi_i^{(\kappa)} x = \bigsqcup \emptyset = \bot$).

Theorem 2. *Let consistently complete ω-algebraic ground domains D_κ, and a set F of continuous first-order functions over D_κ, be given. Then there exists an ω-algebraic model for F in which all finite elements are λ-definable in terms of F, $\{\Psi_i^{(\kappa)}\}$ and the finite elements of the D_κ.*

Proof. Define the set of *partial types* as follows: for each κ and $i \geq 0$ κ_i is a partial (ground) type, and if γ, δ are partial types so is $\gamma \to \delta$. The *full types* are generated from the (full) ground types κ as before.

We let β, γ, δ range over partial types and ρ, σ, τ over full types. Define a partial order $\leqslant$ over all types as follows:

(i) $\kappa_i \leqslant \kappa_{i+1} \leqslant \kappa$.

(ii) If $\gamma \leqslant \gamma' \leqslant \sigma$ and $\delta \leqslant \delta' \leqslant \tau$ then $\gamma \to \delta \leqslant \gamma' \to \delta' \leqslant \sigma \to \tau$.

(iii) Transitive reflexive closure.

It is easy to check that $\{\gamma \mid \gamma \leqslant \sigma\}$ is a lattice for each σ, but we shall only need the pairwise lub operation $\vee$.

Let D_{κ_i} be the range of $\Psi_i^{(\kappa)}$. Define injection-projection pairs $\phi_i^{(\kappa)}, \psi_i^{(\kappa)} : D_{\kappa_i} \rightleftarrows D_\kappa$ and $\phi_{i,j}^{(\kappa)}, \psi_{j,i}^{(\kappa)} : D_{\kappa_i} \rightleftarrows D_{\kappa_j}$ $(j \geq i)$ by

$$\phi_i c = \phi_{i,j} c = c, \qquad (c \in D_{\kappa_i})$$

$$\psi_i x = \Psi_i x, \ \psi_{j,i} c = \Psi_i c \qquad (x \in D_\kappa, \ c \in D_{\kappa_j}),$$

where we have (as we shall when no confusion arises) dropped superscript κ.

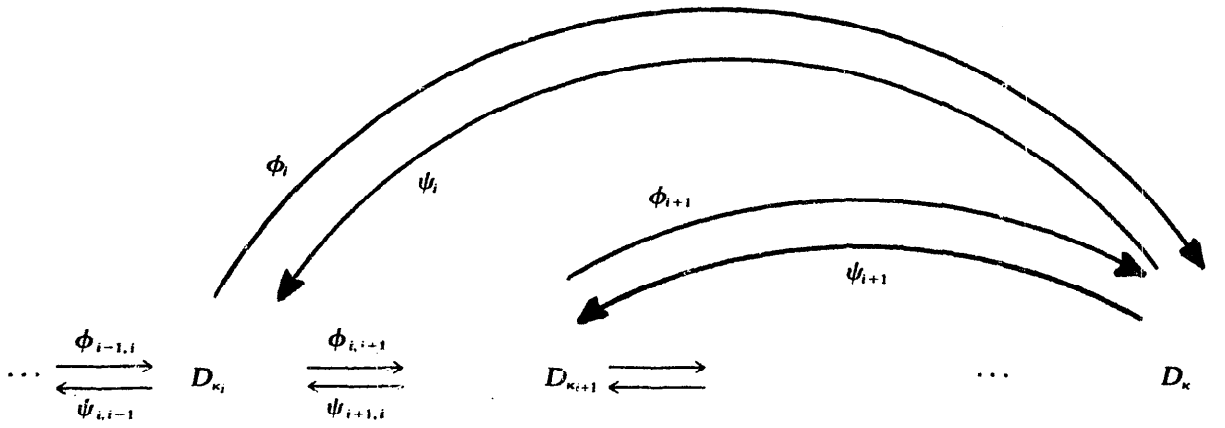

Fig. 2.

The following relations are evident for $i \leqslant j \leqslant k$ (writing $: \sigma$ for $\in D_\sigma$ etc.):

$$\psi_i \circ \phi_i = \lambda c : \kappa_i . c, \quad \phi_i \circ \psi_i \sqsubseteq \phi_{i+1} \circ \psi_{i+1} \sqsubseteq \lambda x : \kappa . x \tag{1}$$

$$\phi_{i,\kappa} = \phi_{j,\kappa} \circ \phi_{i,j}, \quad \psi_{\kappa,i} = \psi_{j,i} \circ \psi_{\kappa,j} \tag{2}$$

$$\psi_{j,i} \circ \phi_{i,j} = \lambda c : \kappa_i . c, \quad \phi_{i,j} \circ \psi_{j,i} \sqsubseteq \lambda c' : \kappa_j . c'. \tag{3}$$

Next we define approximants for each $f \in F$. If f is n-ary, take f_i over the partial ground domains D_{κ_i} to be given by

$$f_i x_1 \cdots x_n = \psi_i(f(\phi_i x_1) \cdots (\phi_i x_n)) \tag{4}$$

where the κ implicit in each ϕ_i and ψ_i may differ.

Now equipped with the (partial) ground types κ_i, the finite domains D_{κ_i} and the f_i, Theorem 1 gives us a monotone model for the set F' of monotonic functions

$$F' = \{f_i\} \cup \{\phi_{i,i+1}\} \cup \{\phi_{i+1,i}\}.$$

We call this the *partial model*; it has a domain for each partial type.

In the partial model can be found the injection-projection pairs $\phi_{\gamma\to\gamma'}, \psi_{\gamma'\to\gamma}: D_\gamma \rightleftarrows D_{\gamma'}$ $(\gamma \leqslant \gamma')$ given by:

$$\phi_{\kappa_i\to\kappa_j} = \phi_{i,j}^{(\kappa)}, \quad \phi_{(\gamma\to\delta)\to(\gamma'\to\delta')} = \lambda d : \gamma \to \delta . (\phi_{\delta\to\delta'} \circ d \circ \psi_{\gamma'\to\gamma}),$$

$$\psi_{\kappa_j\to\kappa} = \psi_{i,j}^{(\kappa)}, \quad \psi_{(\gamma'\to\delta')\to(\gamma\to\delta)} = \lambda d' : \gamma' \to \delta' . (\psi_{\delta'\to\delta} \circ d' \circ \phi_{\gamma\to\gamma'}),$$

and relations (2) and (3) easily generalise; for $\gamma \leqslant \gamma' \leqslant \gamma''$

$$\phi_{\gamma\to\gamma''} = \phi_{\gamma'\to\gamma''} \circ \phi_{\gamma\to\gamma'}, \quad \psi_{\gamma''\to\gamma} = \psi_{\gamma'\to\gamma} \circ \psi_{\gamma''\to\gamma'} \tag{5}$$

$$\psi_{\gamma'\to\gamma} \circ \phi_{\gamma\to\gamma'} = \lambda d : \gamma . d, \quad \phi_{\gamma\to\gamma'} \circ \psi_{\gamma'\to\gamma} \sqsubseteq \lambda d' : \gamma' . d'. \tag{6}$$

We can now begin to build the required *full model*. For D_σ take as members all sets

$$x = \{x_\gamma \mid \gamma \leqslant \sigma\}$$

such that $\gamma \leqslant \gamma' \Rightarrow x_\gamma = \psi_{\gamma\to\gamma} x_{\gamma'}$; the second equation of (5) ensures that this is a good definition. Note that the indexing set $\{\gamma \leqslant \sigma\}$ is $\leqslant$-directed.

The ordering in D_σ is defined pointwise:

$$x \sqsubseteq y \quad \text{iff} \quad \forall \gamma \leqslant \sigma . x_\gamma \sqsubseteq y_\gamma.$$

We can readily check that we have recovered the given (full) ground domains D_κ under the isomorphism $x \leftrightarrow \{\psi_i^{(\kappa)} x \mid i \geqslant 0\}$.

Our construction proceeds analogously to Scott's [7] for a model of the type-free λ calculus, but we are going "in parallel" vertically, while he goes horizontally, with respect to the following picture, whose nodes may be thought of as domains or as types:

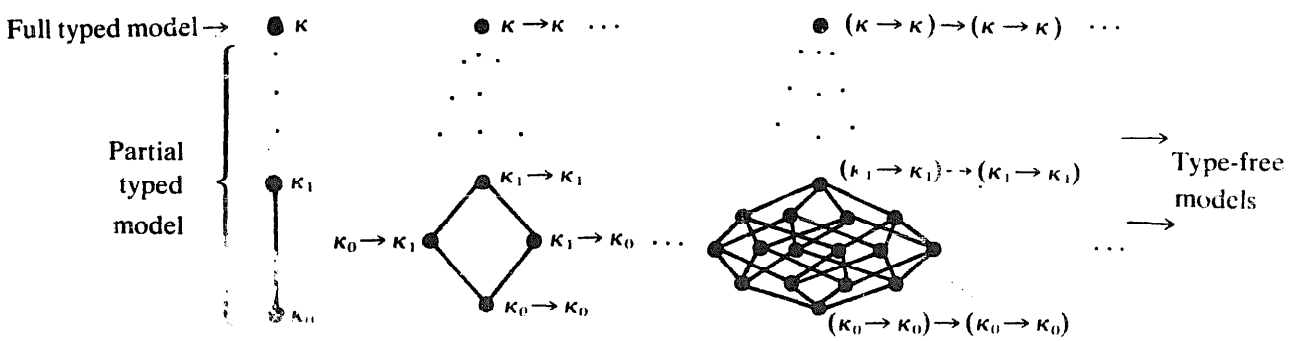

Fig. 3.

In fact the analogy is close enough for us to omit some details and proofs. The reader may also consult Wadsworth [11] for a clear summary of Scott's construction.

We now give the injection-projection pairs $\phi_{\gamma\to\sigma}$, $\psi_{\sigma\to\gamma}$ between partial and full types $\gamma \leqslant \sigma$:

$$\phi_{\gamma\to\sigma}d = \{\psi_{\gamma\vee\gamma'\to\gamma'}(\phi_{\gamma\to\gamma\vee\gamma'}d) \mid \gamma' \leqslant \sigma\},$$

$$\psi_{\sigma\to\gamma}\{x_\gamma \mid \gamma \leqslant \sigma\} = x_\gamma,$$

and relations (2) and (3) again generalize; for $\gamma \leqslant \gamma' \leqslant \sigma$

$$\phi_{\gamma\to\sigma} = \phi_{\gamma'\to\sigma} \circ \phi_{\gamma\to\gamma'}, \qquad \psi_{\sigma\to\gamma} = \psi_{\gamma'\to\gamma} \circ \psi_{\sigma\to\gamma'}, \tag{7}$$

$$\psi_{\sigma\to\gamma} \circ \phi_{\gamma\to\sigma} = \lambda d : \gamma . d, \qquad \phi_{\gamma\to\sigma} \circ \psi_{\sigma\to\gamma} \sqsubseteq \lambda x : \sigma . x \tag{8}$$

Now each D_σ is directly complete (from the pointwise ordering and the finiteness of each D_γ) and each $\phi_{\gamma\to\sigma}$, $\psi_{\sigma\to\gamma}$ is continuous; we may therefore define application for the full model by

$$(xy) = \bigsqcup\{\phi_{\delta\to\tau}(x_{\gamma\to\delta}y_\gamma) \mid \gamma \leqslant \sigma,\ \delta \leqslant \tau\}$$

where $x : \sigma \to \tau$, $y : \sigma$.

With application defined, we next verify that the following are satisfied, for $d : \gamma \to \delta$ and $x : \sigma \to \tau$:

$$\phi_{(\gamma\to\delta)\to(\sigma\to\tau)}d = \phi_{\delta\to\tau} \circ d \circ \psi_{\sigma\to\gamma} \tag{9}$$

$$\psi_{(\sigma\to\tau)\to(\gamma\to\delta)}x = \psi_{\tau\to\delta} \circ x \circ \phi_{\gamma\to\sigma}. \tag{10}$$

It is possible now to consider $D_\gamma \subseteq D_\sigma$ by informally identifying $d \in D_\gamma$ with $\phi_{\gamma\to\sigma}d$ in D_σ. With this identification, we first see that

$$x : \sigma = \bigsqcup_{\gamma\to\sigma} x_\gamma. \tag{11}$$

To justify the identification, it is important that application of partial elements may be done in either the partial or the full domains — that is, for $d : \gamma \to \delta$, $e : \gamma$, assuming no identifications:

$$\phi_{\delta\to\tau}(de) = (\phi_{(\gamma\to\delta)\to(\sigma\to\tau)}d)(\phi_{\gamma\to\sigma}e)$$

which indeed follows from (9) and (8). So with the identifications we can state succinctly (omitting proof) the properties of application that we need. For $x : \sigma \to \tau$, $y : \sigma$

$$xy = \bigsqcup_{\gamma,\delta}\{x_{\gamma\to\delta}y_\gamma\} \tag{12}$$

$$x_{\gamma\to\delta}y = x_{\gamma\to\delta}y_\gamma \tag{13}$$

$$(xy_\gamma)_\delta = x_{\gamma\to\delta}y_\gamma. \tag{14}$$

The remaining properties of the D_σ to be checked, to ensure a model, are that each $f \in F$ is faithfully represented and that S and K exist. Further, we must show the model to be extensional and ω-algebraic, and that all its finite elements are definable.

For extensionality, we argue as follows, following Scott: we require

$$(\forall z.\, xz \sqsubseteq yz) \Rightarrow x \sqsubseteq y, \quad (x, y : \sigma \to \tau).$$

Assume the antecedent. Then for all z, and $\gamma \leqslant \sigma$, $xz_\gamma \sqsubseteq yz_\gamma$, hence for each $\delta \leqslant \tau$ by (14) $x_{\gamma\to\delta} z_\gamma \sqsubseteq y_{\gamma\to\delta} z_\gamma$. So by extensionality in the partial model $x_{\gamma\to\delta} \sqsubseteq y_{\gamma\to\delta}$ for all γ, δ, and $x \sqsubseteq y$ follows by (11). We also need $x \sqsubseteq y \Rightarrow xz \sqsubseteq yz$ and, for monotony, $x \sqsubseteq y \Rightarrow zx \sqsubseteq zy$. These easily follow from the corresponding properties of the partial model, using (11) and (12). Continuity is similarly verified.

For ω-algebraicity, we will show (more accurately) that

$$x : \sigma \text{ is finite} \quad \text{iff for some } \gamma \leqslant \sigma,\ x \in D_\gamma. \tag{15}$$

For then, first $X = \{d \mid d \text{ finite} \sqsubseteq x\}$ is directed, since if $d_1, d_2 \in X$ then $d_1 \in D_{\gamma_1}$, $d_2 \in D_{\gamma_2}$ and so $d_1, d_2 \sqsubseteq x_{\gamma_1 \vee \gamma_2} \in X$. Second, since each $x : \sigma$ is a lub of elements in $\bigcup\{D_\gamma \mid \gamma \leqslant \sigma\}$, which contains only denumerably many elements, D_σ is ω-algebraic.

To establish (15), suppose first that x is finite. Then since $x = \bigsqcup_\gamma x_\gamma$, $x = x_\gamma$ for some γ, so $x \in D_\gamma$. Conversely let $d \in D_\gamma$, and let $d \sqsubseteq \bigsqcup X$, X directed $\subseteq D_\sigma$. Then $d = d_\gamma \sqsubseteq \bigsqcup\{x_\gamma \mid x \in X\}$, and since D_γ is finite, $d \sqsubseteq x_\gamma \sqsubseteq x$ for some $x \in X$. So d is finite.

Let us now check that $f \in F$ is faithfully represented by $\bigsqcup_i f_i$. For $x, y, \ldots$ in appropriate ground domains

$$\begin{aligned}
\left(\bigsqcup_i f_i\right) xy\ldots &= \bigsqcup_i (f_i xy \cdots) \\
&= \bigsqcup_i (f_i x_i y_i \cdots) \quad \text{by repeated use of (13), (14)} \\
&= \bigsqcup_i (\psi_i (f x_i y_i \cdots) \text{ by (4) and identification of } x_i \text{ with } \phi_i x_i \\
&= \bigsqcup_i \psi_i \left(f \left(\bigsqcup_i x_i\right)\left(\bigsqcup_i y_i\right) \cdots\right) \\
&= \bigsqcup_i (fxy \cdots)_i = fxy \text{ as required.}
\end{aligned}$$

Now for S and K. Let us just consider S; K is easier. Abbreviate the type $(\beta \to (\gamma \to \delta)) \to (\beta \to \gamma) \to (\beta \to \delta)$ by $\beta\gamma\delta$ just for now, and the corresponding full type by $\rho\sigma\tau$. We define

$$S : \rho\sigma\tau = \bigsqcup\{S_{\beta\gamma\delta} \mid \beta\gamma\delta \leqslant \rho\sigma\tau\}; \tag{16}$$

the $S_{\beta\gamma\delta}$ are known to exist in the partial model. That $\{S_{\beta\gamma\delta}\}$ is directed follows from the fact that *in the partial model*, when $\beta\gamma\delta < \beta'\gamma'\delta'$,

$$S_{\beta\gamma\delta} = \psi_{\beta'\gamma'\delta'\to\beta\gamma\delta} S_{\beta'\gamma'\delta'} \tag{17}$$

which we verify as follows, repeatedly using the inductive definitions of the ϕ's and ψ's of the partial model (and omitting types):

$$(\psi S)xyz = \psi(S(\phi x)(\phi y)(\phi z)) = \psi((\phi x)(\phi z)((\phi y)(\phi z)))$$

$$= \psi(\phi(xz(yz))) = xz(yz) = Sxyz.$$

Now to check that the S of (16) is the right function, we compute

$$Sxyz = \bigsqcup_{\beta\gamma\delta} (S_{\beta\gamma\delta} xyz) = \bigsqcup_{\beta\gamma\delta} S_{\beta\gamma\delta} x_{\beta\to\gamma\to\delta} y_{\beta\to\gamma} z_\beta \quad \text{(using (13) and (14))},$$

$$= \bigsqcup_{\beta\gamma\delta} (x_{\beta\to\gamma\to\delta} z_\beta (y_{\beta\to\gamma} z_\beta)) \quad \text{(from partial model)}$$

$$= xz(yz) \text{ by distributing } \sqcup \text{ and using (11).}$$

All that now remains is the definability of the finite elements in terms of the finite elements of D_κ, F and the $\Psi_i^{(\kappa)}$. But they are all definable in the *partial* model, in terms of the D_α, f_i, $\phi_{i,i+1}$ and $\psi_{i+1,i}$ (and of course S and K). To define a finite element in the full model, we need only take a term which defines it in the partial model and replace each atomic component by a term defining *it* in the full model (since application can be done in either). With the aid of one last definition, we show that all these terms need involve only S, K, F, $\Psi_i^{(\kappa)}$ and the finite members of D_κ.

Let $\Psi_\gamma^{(\sigma)} : \sigma \to \sigma = \phi_{\gamma\to\sigma} \circ \psi_{\sigma\to\gamma}$, for each σ and $\gamma \leqslant \sigma$. At ground type $\Psi_{\kappa_i}^{(\kappa)}$ so defined is just $\Psi_i^{(\kappa)}$. Also it is easily found that $\Psi_{\gamma\to\delta}^{(\sigma\to\tau)} = \lambda x : \sigma\to\tau.\ \Psi_\delta^{(\tau)} \circ x \circ \Psi_\gamma^{(\sigma)}$, so that all the Ψ are definable by S, K and the $\Psi_i^{(\kappa)}$. Moreover

(i) Each element of D_{κ_i} is a finite element of D_κ.

(ii) Each $f_i : \sigma$ is $\Psi_\gamma^{(\sigma)} f$ for some γ.

(iii) $\phi_{i,i+1}^{(\kappa)}$ is $\phi_{(\kappa_i\to\kappa_{i+1})\to(\kappa\to\kappa)}(\phi_{\kappa_i\to\kappa_{i+1}})$ in $D_{\kappa\to\kappa}$, $= \Psi_i^{(\kappa)}$ using (7), (8) and (9).

(iv) $\psi_{i+1,i}^{(\kappa)}$ is also $\Psi_i^{(\kappa)}$ in $D_{\kappa\to\kappa}$ by a similar argument.

(v) From (17), $S_{\beta\gamma\delta}$ is $\Psi_{\beta\gamma\delta}^{(\rho\sigma\tau)} S$ in $D_{\rho\sigma\tau}$, and similarly for κ.

This concludes the proof of Theorem 2. □

6. Fully abstract models

In what follows we shall use overbars ($\bar{}$) to represent the natural interpretation of combinations in a model; that is, $\bar{S} = S$, $\bar{K} = K$, $\bar{f} = f$, $\bar{c} = c$ and $\overline{MN} = (\bar{M}\bar{N})$. When only one model is under discussion we sometimes omit the overbar and write

$M \sqsubseteq (=) M'$ to mean $\bar{M} \sqsubseteq (=) \bar{M}'$, but otherwise we retain distinct overbars, e.g. $\bar{(\ )}$ and $\bar{\bar{(\ )}}$.

Definition. A monotone model $\mathcal{M}$ is *fully abstract* if $M \sqsubseteq M'$ whenever $\forall$ ground $\mathscr{C}[\ ]$. $\mathscr{C}[M] \sqsubseteq \mathscr{C}[M']$.

Corollary 1. *The model of Theorem 2 is fully abstract provided the $\Psi_i^{(\kappa)}$ are definable (from* **S**, **K**, {**f**} *and* {**c**}).

Proof. Let M, M' have type $\sigma_1 \to \cdots \to \sigma_m \to \kappa$, and assume $\forall$ ground $\mathscr{C}[\ ]$. $\mathscr{C}[M] \sqsubseteq \mathscr{C}[M']$. Then for all $x_i \in D_{\sigma_i}$

$$\bar{M}x_1 \cdots x_m = \bigsqcup\{\bar{M}(x_1)_{\gamma_1} \cdots (x_m)_{\gamma_m} \mid \gamma_i \leqslant \sigma_i\}$$
$$\sqsubseteq \bigsqcup\{\bar{M}'(x_1)_{\gamma_1} \cdots (x_m)_{\gamma_m} \mid \gamma_i \leqslant \sigma_i\}$$

by assumption, since the $(x_i)_{\gamma_i}$ are finite and so definable,

$$= \bar{M}'x_1 \cdots x_m.$$

Hence $M \sqsubseteq M'$ by extensionality. □

The next lemma shows that two fully abstract monotone models are essentially the same when restricted to their definable elements.

Full Abstraction Lemma. *Let $\mathcal{M}$, $\mathcal{M}'$ be fully abstract monotone models for given D_κ and (monotone) F. Then $\bar{M}_1 \sqsubseteq \bar{M}_2$ iff $\bar{\bar{M}} \sqsubseteq \bar{\bar{M}}_2$, where $\bar{(\ )}$ and $\bar{\bar{(\ )}}$ are the interpretations of combinations in $\mathcal{M}$ and $\mathcal{M}'$.*

Proof. Assume $\bar{M}_1 \sqsubseteq \bar{M}_2$, and let $\mathscr{C}[\ ]$ be any ground context. Then $\overline{\mathscr{C}[M_1]} \sqsubseteq \overline{\mathscr{C}[M_2]}$ by monotonicity. Let $\mathscr{C}[M_i] \to^* w_i$, $i = 1, 2$. Then $\bar{\bar{w}}_1 \sqsubseteq \bar{\bar{w}}_2$, since $\to$ preserves interpretation. But $\bar{\bar{w}}_i = \bar{w}_i \in D_\kappa$, so $\bar{w}_1 \sqsubseteq \bar{w}_2$, hence $\overline{\mathscr{C}[M_1]} \sqsubseteq \overline{\mathscr{C}[M_2]}$, and since $\mathscr{C}[\ ]$ was arbitrary $\bar{M}_1 \sqsubseteq \bar{M}_2$ follows by the full abstraction of $\mathcal{M}'$. □

We conclude this section with two lemmas which will be needed for the uniqueness proof in the next section ; the lemmas are not directly concerned with full abstraction.

Second Context Lemma. *In any monotone model for (monotone) F, the following are equivalent, for combinations M, M' of type $\sigma_1 \to \cdots \to \sigma_m \to \kappa$:*

(i) *$\forall P_i$ of type σ_i, $i \leqslant m$, $MP_1 \cdots P_m \sqsubseteq M'P_1 \cdots P_m$,*

(ii) *$\forall$ ground $\mathscr{C}[\ ]$. $\mathscr{C}[M] \sqsubseteq \mathscr{C}[M']$.*

Proof. It is immediate that the relation $\sqsubseteq$ over combinations (which is a quasi-

order) satisfies the conditions of the First Context Lemma, using the monotonicity of the functions F. □

Now in any model containing the projections $\Psi_i^{(\kappa)}$ the projections $\Psi_\gamma^{(\sigma)}$ at all types exist, where $\Psi_{\gamma\to\delta}^{(\sigma\to\tau)}x = \Psi_\delta^{(\tau)} \circ \chi \circ \Psi_\gamma^{(\sigma)}$. Of course they are definable if the $\Psi_i^{(\kappa)}$ are so.

Algebraicity Lemma. *Given D_κ and F as in Theorem 2, in any continuous model $\mathcal{M}$ for F which contains the $\Psi_i^{(\kappa)}$, the finite elements in D_σ are exactly $\{\Psi_\gamma^{(\sigma)}x \mid x \in D_\phi, \gamma \leqslant \sigma\}$, and $\mathcal{M}$ is also ω-algebraic.*

Proof. Write x_γ for $\Psi_\gamma^{(\sigma)}x$, and let D_γ be the range of $\Psi_\gamma^{(\sigma)}$. Now each D_γ is finite; for ground types we know this to be true, and at higher types it follows easily by induction using $x_{\gamma\to\delta}y = (xy_\gamma)_\delta$. To show each x_γ finite, let $x_\gamma \sqsubseteq \bigsqcup Z$. Then

$$\begin{aligned} x_\gamma &\sqsubseteq \bigsqcup\{z_\gamma \mid z \in Z\}, \\ &= z_\gamma \quad \text{for some } z \in Z \text{ since } D_\gamma \text{ is finite}, \\ &\sqsubseteq z. \end{aligned}$$

Conversely, let $x \in D_\sigma$ be finite. Since $x = \bigsqcup\{x_\gamma \mid \gamma \leqslant \sigma\}$ is easily proved, $x = x_\gamma$ for some $\gamma \leqslant \sigma$.

Finally, $\mathcal{M}$ is ω-algebraic because $x = \bigsqcup\{x_\gamma \mid \gamma \leqslant \sigma\}$, and because $\bigcup\{D_\gamma \mid \gamma \leqslant \sigma\}$ is denumerable. □

7. Uniqueness

Is there more than one fully abstract continuous model for D_κ and F (as in Theorem 2)? The answer is yes in general, but certain natural conditions on F do ensure uniqueness.

Note first that over ω-algebraic consistently complete ground domains the pairwise glb operation $\sqcap$ is continuous; we leave the proof to the reader.

Definition. *F articulates* the D_κ if the following first order functions are definable from F and the finite elements of the D_κ:

(i) the $\Psi_i^{(\kappa)}: \kappa \to \kappa$,

(ii) $\sqcap: \kappa \to \kappa \to \kappa$,

(iii) $[\sqsupseteq c]: \kappa \to \kappa^{(0)}$, for each finite $c \in D_\kappa$, where $\kappa^{(0)}$ is some fixed ground type, and for some arbitrary fixed finite element $tt \neq \bot$ in $D_{\kappa^{(0)}}$:

$$[\sqsupseteq c]x = \begin{cases} tt & \text{if } x \sqsupseteq c, \\ \bot & \text{otherwise.} \end{cases}$$

We also say that F is *articulate* (for the D_κ).

Remark. The functions $(\sqsupseteq c]$ are just the functions $c \Rightarrow tt$ in Plotkin's notation [4]; we use a different notation since we are not concerned with all of the functions $c \Rightarrow c'$, c and c' finite.

Theorem 3. (*Uniqueness*) *Under the conditions of Theorem 2, if F articulates the D_κ then there is only one fully abstract continuous model for F (up to order isomorphism) and it is ω-algebraic and consistently complete.*

Proof. By the Algebraicity Lemma, all fully abstract continuous models for F are ω-algebraic since they contain the projections, and by the Full Abstraction Lemma they are all order-isomorphic when restricted to their definable elements. We shall show that in every such model the finite elements are all definable; it follows that they are all isomorphic, since an algebraic cpo is determined by its finite elements.

Suppose, to the contrary, that $\mathcal{M}'$ is such a model containing a non-definable finite element. Let $\bar{=}$ be the interpretation of combinations in $\mathcal{M}'$. We shall define combinations H_1, H_2 such that $\bar{\bar{H}}_1 \neq \bar{\bar{H}}_2$, but $\overline{\overline{\mathscr{C}[H_1]}} = \overline{\overline{\mathscr{C}[H_2]}}$ for all ground $\mathscr{C}[\ ]$, contradicting full abstraction. Let $\mathcal{M}'$ have domains D'_σ ($D'_\kappa = D_\kappa$).

Let $\sigma = \sigma_1 \to \cdots \to \sigma_m \to \kappa$ be a minimal type such that D'_σ has a non-definable finite element $d^* \in D'_\gamma$; that is, assume that the finite elements in each D'_{σ_i} are definable (σ cannot be a ground type).

Let $d_1, \ldots, d_n$ be the definable elements $\sqsupseteq d^*$ in D'_γ, and $e_1, \ldots, e_p$ the other definable elements of D'_γ.

Case 1. $n \neq 0$. $\bigsqcap_{j<n} d_j = x$ is definable, hence so is $d = x_\gamma$; in fact, $d = d_j$ for some j. That is, d is the glb of definable elements in $D'_\gamma \sqsupseteq \iota$. Since d^* is not definable, $d \sqsupset d^*$.

Hence there is an m-vector $\hat{f}$ of necessarily definable finite elements, $f_i \in D'_{\sigma_i}$, such that $d\hat{f} \sqsupseteq d^*\hat{f}$ in D_κ. Also, since each $e_k \not\sqsupseteq d^*$, there are m-vectors $\hat{g}^{(k)}$ $(k \leqslant p)$ such that $e_k\hat{g}^{(k)} \not\sqsupseteq d^*\hat{g}^{(k)} \neq \bot$ in D_κ. All of these elements in D_κ are finite.

Define combinations H_1, H_2 such that

$$\bar{\bar{H}}_1 x = \bigsqcap \{[\sqsupseteq d^*\hat{g}^{(k)}](x_\gamma \hat{g}^{(k)}) \mid k \leqslant p\} \sqcap [\sqsupseteq d^*\hat{f}](x_\gamma \hat{f})$$

$$\bar{\bar{H}}_2 x = \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad \sqcap [\sqsupseteq d\hat{f}](x_\gamma \hat{f}),$$

i.e. the definitions agree except in the last term.

Now for all definable x *either* $x_\gamma = d_i$ and $\bar{\bar{H}}_1 x = \bar{\bar{H}}_2 x = tt$, *or* $x_\gamma = e_k$ and

$\bar{\bar{H}}_1 x = \bar{\bar{H}}_2 x = \perp$. Hence by the Second Context Lemma, $\overline{\overline{\mathscr{C}[H_1]}} = \overline{\overline{\mathscr{C}[H_2]}}$ for all ground $\mathscr{C}[\ ]$.

On the other hand, $\bar{\bar{H}}_1 d^* = tt$ and $\bar{\bar{H}}_2 d^* = \perp$.

Case 2. $n = 0$. Then no definable d in $D_\gamma \sqsupseteq d^*$. So the following combinations achieve a similar result:

$$\bar{\bar{H}}_1 x = \bigsqcap\{[\sqsupseteq d^* \hat{g}^{(k)}](x_\gamma \hat{g}^{(k)}) \mid k \leqslant p\},$$

$$\bar{\bar{H}}_2 x = \perp.$$

In either case, we have a contradiction, since $\mathscr{M}'$ was assumed fully abstract.

For consistent completeness, it is enough to demonstrate that any pair d, e of *finite* elements with an upper bound possesses a lub. For then if x, y is any pair in D_σ with an upper bound, each pair x_γ, y_γ $(\gamma \leqslant \sigma)$ is upper bounded and so has lub z_γ; moreover $\{z_\gamma \mid \gamma \leqslant \sigma\}$ is easily seen to be directed, and its lub is the lub of x and y.

Now let d, e be finite (in D_γ say) and upper bounded. Then $\{x_\gamma \mid x \sqsupseteq d, e\}$ is non-empty, and finite (since D_γ is finite), and its glb if it exists is a lub for d and e. But since $\sqcap$ is definable at ground type it is definable at all types, hence $\bigsqcap\{x_\gamma \mid x \sqsupseteq d, e\}$ exists in the model $\mathscr{M}$. □

8. Fixed points and applications

We have so far avoided the fixed point combinator **Y** at all types $(\sigma \to \sigma) \to \sigma$ and its interpretation; our combinations have contained only **S**, **K**, {**f**} and {**c**}.

Now $Y = \bigsqcup_n Y_n$ where $Y_n = \lambda z.\, z^n \perp$, so Y exists in all continuous models. Moreover $Y_\gamma = \bigsqcup_n (Y_n)_\gamma$, $= (Y_n)_\gamma$ for some n since D_γ is always finite. So each Y_γ is definable if the $\Psi_\gamma^{(\sigma)}$ are so, and $Y = \bigsqcup_\gamma Y_\gamma$.

We must make sure that adding **Y**, with $\bar{\mathbf{Y}} = Y$, does not affect full abstraction. (In fact the argument is not specific to Y; it applies whenever we wish to add a combinator — or constant — denoting $\bigsqcup\{\bar{M} \mid M \in \mathscr{S}\}$ where $\mathscr{S}$ is a set of combinations whose interpretations form a directed set in every continuous model.)

Fixed-point Lemma. *A continuous model $\mathscr{M}$ for F is fully abstract for **Y**-free combinations iff it is so for all combinations, if the projections $\Psi_\gamma^{(\sigma)}$ are definable; i.e. the following are equivalent:*

(i) $(\forall$ *ground* $\mathscr{C}[\ ].\ \mathscr{C}[M] \sqsubseteq \mathscr{C}[M']) \Rightarrow M \sqsubseteq M'$, *where M, M' and the $\mathscr{C}[\ ]$ may contain **Y**.*

(ii) *The same with **Y** excluded.*

Proof. First we show that every combination P defining a finite element is equivalent to a **Y**-free combination. Write $P = P[\mathbf{Y}, \ldots, \mathbf{Y}]$ to distinguish the

occurrences of $\mathbf{Y}$, at types $\sigma_1, \sigma_2 \cdots$ say. Then $P = \bigsqcup\{P[\mathbf{Y}_{\gamma_1}, \mathbf{Y}_{\gamma_2} \cdots] \mid \gamma_i \leqslant \sigma_i\}$, $= P[\mathbf{Y}_{\gamma_1}, \mathbf{Y}_{\gamma_2} \cdots]$ for some γ_i since P is finite; in the latter we merely replace each $\mathbf{Y}_{\gamma_i}$ by its $\mathbf{Y}$-free equivalent.

(ii) $\Rightarrow$ (i). Assuming the antecedent of (i), and choosing $\mathscr{C}[\ ]$ to be $\mathscr{D}[([\ ])_\gamma]$, remembering that $(\ )_\gamma = \Phi^\sigma_\gamma$ is definable, we have $\mathscr{D}[M_\gamma] \sqsubseteq \mathscr{D}[M'_\gamma]$ for all $\mathscr{D}$ and all γ. But since M_γ and M'_γ have $\mathbf{Y}$-free equivalents, from (ii) we obtain $M_\gamma \sqsubseteq M'_\gamma$. Since γ was arbitrary, we conclude $M \sqsubseteq M'$.

(i) $\Rightarrow$ (ii). For this part we do not need the projections. Let $\mathscr{C}[\ ]$ be any ground context, perhaps containing $\mathbf{Y}$, and assume that M, M' do not contain $\mathbf{Y}$ and that $\forall$ $\mathbf{Y}$-free $\mathscr{D}[\ ].\mathscr{D}[M] \sqsubseteq \mathscr{D}[M']$. It will be enough to show that $\mathscr{C}[M] \sqsubseteq \mathscr{C}[M']$, since the consequent of (ii) will follow using (i).

Write $\mathscr{C}[M] = \mathscr{C}[M, \mathbf{Y}, \mathbf{Y} \cdots]$ to distinguish the occurrences of $\mathbf{Y}$. Then

$$\mathscr{C}[M] = \bigsqcup\{\mathscr{C}[M, Y_{n_1}, Y_{n_2} \cdots] \mid n_i \geqslant 0\}$$

$$\sqsubseteq \bigsqcup\{\mathscr{C}[M', Y_{n_1}, Y_{n_2} \cdots] \mid n_i \geqslant 0\} \text{ by antecedent of (ii)}$$

since $\mathscr{C}[[\ ], Y_{n_1}, Y_{n_2}, \ldots]$ is $\mathbf{Y}$-free

$$= \mathscr{C}[M'], \text{ as required.} \quad \square$$

We are now at last in a position to look at some examples. We restrict ourselves to two; they should serve as evidence that the conditions that F be articulate is likely to be satisfied in practice. Indeed, F will normally be richer still; it appears that the indispensable conditional operation will not be definable from the articulating functions alone.

Corollary 3. *PCF has a unique fully abstract continuous model.*

Proof. Recall that PCF has ground domains D_ι, D_o and functions F as described in Section 3. It has constants $\mathbf{0}, \mathbf{1}, \ldots$ for the integers and $\mathbf{tt}, \mathbf{ff}$ for the truth values, and the fixed-point combinator $\mathbf{Y}$. (We can assume constants $\perp_\iota, \perp_o$ if we like, but they are definable by $\mathbf{Y}(\lambda x.\, x)$.)

Trivially, D_ι and D_o are consistently (and directedly) complete, and $\sqcap$ is continuous over them. We leave it mainly to the reader to show that F articulates D_ι and D_o. The natural enumerations $\perp_o, tt, ff, ff, \ldots$ and $\perp_\iota, 0, 1, 2, \ldots$ will do. Using $\mathbf{Y}$ an equality predicate $=$ may be defined such that in D_ι and D_o

$$(x = y) \text{ is } \begin{cases} tt & \text{if } x \text{ and } y \text{ are identical and } \neq \perp, \\ \perp_o & \text{otherwise.} \end{cases}$$

Then $\sqcap$ is defined by $\lambda x.\, \lambda y.\, (x = y) \supset x, \perp$; $[\sqsupseteq c]$ is defined by $\lambda x.\, (x = \mathbf{c})$ if c is not $\perp$; $[\sqsupseteq \perp]$ is defined by $\lambda x.\, tt$. The reader will have no trouble defining the projections $\Psi_i^{(\iota)}, \Psi_i^{(o)}$.

With the addition of the parallel conditional $:\supset$, which differs from $\supset$ in that $\perp_o :\supset x, x = x$, the result also holds; F still articulates D_ι and D_o. □

Plotkin showed that the model of *all* continuous functions is fully abstract for PCF with $\cdot\supset$. Since he also showed that all finite elements in this model are definable, he had already demonstrated uniqueness of the continuous model in this case, provided the ground domains are kept fixed.

PCF without $:\supset$ is different. Plotkin showed that the model of all continuous functions is not fully abstract, but we now have a unique fully abstract model. What continuous functions are missing?

We can at least partly answer this question. For every definable element z of type $\sigma_1 \to \cdots \to \sigma_n \to \kappa$ may be shown to have the property that *either* z is a constant function, i.e. $zx_1 \cdots x_n$ is independent of the x_i, *or* z is strict in some argument, i.e. for some i $x_i = \perp \Rightarrow zx_1 \cdots x_n = \perp$. Hence in our model all finite elements have this property, and so *every* element has the property also, since the property is preserved by directed $\bigsqcup$. Many continuous functions, in particular $:\supset$, do not possess the property and so are missing from the model.

Another example. Consider a single ground domain $D_\iota = \Sigma^* \cup \Sigma^\omega$ where Σ is a finite alphabet, (i.e. the domain of finite and infinite strings), under the order $s \sqsubseteq s'$ iff $s' = ss''$ for some s''. For the alphabet $\{0, 1\}$ D_ι is an infinite binary tree, with limit points (Σ^ω) added. The finite elements are just Σ^* of course.

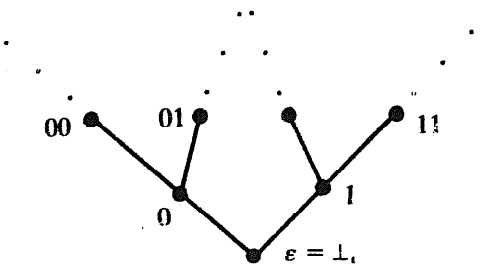

Fig. 4.

A natural set of primitive functions F over D_ι, when $\Sigma = \{0, 1\}$, is $\{S_0, S_1, T, \mathbb{D}\}$ where

$$S_0 s = 0s,\ S_1 s = 1s,\ T(1s) = T(0s) = s,\ T\varepsilon = \varepsilon,$$

and

$$\begin{cases} 0s\ \mathbb{D}\ s', s'' = s' \\ 1s\ \mathbb{D}\ s', s'' = s'' \\ \varepsilon\ \mathbb{D}\ s', s'' = \varepsilon. \end{cases}$$

($\mathbb{D}$ has type $\iota \to \iota \to \iota \to \iota$, and we write $x\ \mathbb{D}\ y, z$ for $\mathbb{D}\ xyz$).

It is easy to show that F is articulate (with the help of Y, which is justified by the lemma of Section 8). In particular,

$$[\sqsupseteq \varepsilon] = \lambda x.0,\ [\sqsupseteq 0s] = \lambda x.x \ \mathbb{D}\ [\sqsupseteq s](Tx), \varepsilon,$$
$$[\sqsupseteq 1s] = \lambda x.x \ \mathbb{D}\ \varepsilon, [\sqsupseteq s](Tx)$$

and $\sqcap$ is recursively defined by

$$x \sqcap y = x \ \mathbb{D}\ (y \ \mathbb{D}\ S_0(Tx \sqcap Ty), \varepsilon), (y \ \mathbb{D}\ \varepsilon, S_1(Tx \sqcap Ty)).$$

The projections $\Psi_i^{(\iota)}$ may be defined without recursion, taking for example the enumeration $\varepsilon, 0, 1, 00, 01, 10, 11, \ldots$ of Σ^*.

Hence there is again a unique fully abstract continuous model. And again many continuous functions are missing — for example

$$f : \iota \to \iota \to \iota \text{ for which } fs_1s_2 = \begin{cases} \varepsilon & \text{if } s_1 = s_2 = \varepsilon \\ 0 & \text{otherwise} \end{cases}.$$

Note that f is finite. I am grateful to Gordon Plotkin for showing me, using the technique of logical relations, that f is not definable (see [5]).

9. Conclusion

The interest in finding models smaller than that of all continuous functions is at least two-fold. First, the smaller the model, the stronger one expects the proof rules to be. Full abstraction may provide useful proof rules; an obvious example is the so called ω-rule, which infers $M = N$ from the hypothesis that $MZ = NZ$ for all (closed) combinations Z. Another rule — valid in our model of PCF — would say that every function is either constant or else strict in some argument.

Second, many programming languages — almost all real ones, in fact, and in addition PCF and the string language of the previous section — admit a 'sequential' evaluation method. By looking harder at these models we may hope to characterize sequentiality property.

Apart from this, the notion of full abstraction is intuitively a compelling one, as we attempted to show in the introduction.

We would not like to give the impression that the step from fully abstract semantics for typed λ-calculi to fully abstract semantics for programming languages is a small one. There are reasons for expecting the opposite. First, the semantic domains [8] which one naturally chooses for most programming languages are reflexive ones — that is, they are solutions of sets of recursive domain equations. Second, for many of the abstract objects which are elements of these domains — as examples we may consider environments, stores, continuations, processes — the programming language can typically express only a small repertoire of operations

over them. It remains to be seen therefore whether equivalence of programs and program phrases may be expressed as identity of meaning in suitable semantic domains, or whether it will continue to be necessary to express it by some equivalence relation over the domains. The latter may be done using inclusive predicates (Milne [2]) or equivalently directed-complete relations (Reynolds [6]). The present work therefore indicates only that there is still a chance for the first alternative.

I am indebted to Gordon Plotkin for helpful discussions during this work. His study of full abstraction in [4], in a special case, added to my motivation to study it under more general conditions, and he encouraged me to look for conditions under which the fully abstract model is unique.

References

[1] J. M. E. Hyland, A survey of some useful partial order relations on terms of the lambda calculus, *Proc. Symp. on λ-calculus Comput. Sci. Theory*, Rome (March 1975).

[2] R. Milne, The formal semantics of computer languages and their implementations, Oxford University Computing Laboratory, Programming Research Group, Technical Monograph PRG–13 (1975).

[3] R. Milner, Processes; a mathematical model for computing agents in: Rose and Shepherdson (Eds.), *Logic Colloquium '73, Studies in Logic and the Foundations of Mathematics*, Vol. 80 (North Holland/American Elsevier, 1975).

[4] G. Plotkin, LCF as a programming language, *Proc. Conf. Program Proving and Improving*, Arc-et-Senans (1975; *Theoret. Comput. Sci.* (to appear)).

[5] G. Plotkin, Lambda definability and logical relations, Memo SAI–RM–4, School of Artificial Intelligence, Edinburgh (1973).

[6] J. C. Reynolds, On the relation between direct and continuation semantics, in: Loeckx (Ed.), *Automata, Languages and Programming*, 2nd Colloquium, University of Saarbrucken, Lecture Notes in Computer Science Vol. 14 (Springer-Verlag, 1974).

[7] D. Scott, Lattice theoretic models for various type-free calculi, *Proc. 4th Internl. Congress for Logic, Methodology and Philosophy of Science*, Bucharest (1972).

[8] D. Scott and C. Strachey, Towards a Mathematical semantics for computer languages, *Proc. Symp. on Comput. Automata*, Microwave Research Institute Symposia Series, Vol. 21, Polytechnic Institute of Brooklyn (1972).

[9] S. Stenlund, *Combinators, λ-terms and Proof Theory* (Reidel Co., Holland, 1972).

[10] J. Vuillemin, Proof techniques for recursive programs, Research Report, IRIA, 78150 Le Chesnay, France (1973).

[11] C. Wadsworth, Relation between computational and denotational properties for Scott's D^∞ models of the λ-calculus, *SIAM J. Comput.* **5** (3) (1976).