Some Results on the Full Abstraction Problem for Restricted Lambda Calculi

Furio Honsell and Marina Lenisa

Dipartimento di Matematica e Informatica,
Università di Udine, via Zanon,6 - Italy
E-mail:{honsell,lenisa}@udmi5400.cineca.it

dedicated to Corrado Böhm
on the occasion of his 70^{th} *birthday*

Abstract. Issues in the mathematical semantics of two restrictions of the λ-calculus, i.e. λI-calculus and λ_V-calculus, are discussed. A *fully abstract* model for the natural evaluation of the former is defined using complete partial orders and strict Scott-continuous functions. A correct, albeit non-*fully abstract*, model for the SECD evaluation of the latter is defined using Girard's coherence spaces and stable functions. These results are used to illustrate the interest of the analysis of the *fine structure* of mathematical models of programming languages.

1 Introduction

D. Scott, in the late sixties, discovered a truly mathematical semantics for λ-calculus using continuous lattices. His D_∞ model was the first example of a mathematical structure, completely autonomous from syntax, which provided independent insight into the combinatorics of the evaluation of functional expressions. Since then, much energy has been devoted to develop and investigate mathematical structures which model denotations of programs. These are usually quite elaborate. It is therefore questionable if the denotational account of the semantics of programming languages is to be preferred to other forms of formal description of the semantics, e.g. operational semantics or rewrite semantics. We think that mathematical models, albeit involved, are useful to define and interesting to investigate. In the first place there is a psychological motivation. Mathematical structures have a far richer structure, e.g. topological or order theoretic, than merely that necessary to interpret the linguistic phrases. They provide independent and abstract insight in the structure of meanings, and this, being often geometric, can stimulate mental imagery, far better than pure symbol pushing. Secondly, mathematical models yield non trivial theories of *program equivalence*. This is particularly fruitful since modern methodologies of program development are often based on *program transformation*. This technique relies on powerful theories of *observational equivalence* of modules of programs, whereby one can replace a module in a program by an observationally

equivalent one, preserving the overall meaning of the main program. In practice the original modules are logically graspable but inefficient while the final ones are the opposite, being often obtained by iterated transformations of submodules. Formally, in the case of the λ-calculus, given a machine I for evaluating programs, i.e. closed λ-terms, M and N are said to be *observationally equivalent* if and only if:
$\forall C[\,].(\forall C[M], C[N] \in \Lambda^{\circ}.\ (I \text{ stops on input } C[M] \iff I \text{ stops on input } C[N]))$.
Of course, given an evaluation machine, the most useful models are those which provide a *fully abstract* semantics with respect to it, i.e. those which induce an equivalence on programs which coincides with the observational equivalence of the machine. This property however, is not necessary for the model to be worthwhile investigating. In [6] a denotational model for the language ISWIM of P. Landin is provided, which even if it induces a finer equivalence on terms, than the observational equivalence induced by the SECD machine, it allows nonetheless to show easily, say, that all fixed point operators are observationally equal. Of course one could prove the validity of each such equation using some involved *computation induction*, at least in principle. But, syntactic proofs are more *error-prone*. And, as it occurs in this case, we can prove easily the equations in question just by analyzing the *fine structure* of the model using mathematical tools, which work only because of the extra structure of the model. We can show in fact, an *ApproximationTheorem*, which allows to express the meaning of non-terminating processes in terms of limits of simpler ones. Topological and order theoretic techniques are at the core of the many results which characterize useful classes of program equivalences in denotational semantics. This motivation for building and investigating mathematical models is ultimately an instance of the general use of mathematical structures as a source of inspiration for developing interesting logical principles for reasoning about programs.

In this paper we present two constructions which exemplify the advantages that can be gained from infinite mathematical models in the study of the semantics of programming languages, of the kind discussed above. Both constructions deal with the semantics of variants of the classical λ-calculus (the λK-calculus), which axiomatize functions which *strictly* depend on all their arguments: the λI-calculus and the λ_V-calculus. The first construction has a foundational character, while the second is more in the way of applications.

The former calculus is the system originally introduced by Church in [3] in order to characterize and study the class of intuitively computable functions. The set of λI-terms is the subset of the λK-terms, consisting of those terms in which abstracted variables always occur in the body of the abstraction. This calculus has peculiar computational properties. For instance weakly normalizing terms are strongly normalizing, hence normal forms can be safely taken as values; but then, because of the lack of erasing combinators, all encodings are more cumbersome than in the λK-calculus, and one is forced to model recursive functions without fixed-point combinators. D. Scott, in [2], contrasted briefly the viewpoint of Church, with that of Curry who favored the λK-calculus. In particular he remarked: “In the λI-calculus a function is meant to strictly depend on all its arguments..." but "... what is a strict function? is it one which is undefined for undefined arguments? that seems a necessary condition, but is this all that we should say?" A complete satisfactory answer should illuminate on the general notion of *strictness* in relations. Very little has been done, up to now, even in the direction of investigating Scott's suggestion to model the λI-functions using strict functions in the context of domain semantics. In particular in [2] (p.368), an open problem concerning the mathematical semantics of the λI-calculus is raised: *Does there exist a mathematical model for the extensional theory where all terms without normal form are equated, which is known to be consistent by purely syntactic means?* The importance of this theory lies in the fact that it is the *natural* maximal theory. A positive answer to this question was announced in [5]. In this paper we will define such a mathematical model, thus providing partial evidence that Scott's original suggestion is indeed viable. To this end we shall investigate briefly the semantics of the λI-calculus.

The λ_V-calculus was introduced by Plotkin [12] in order to reason about equivalence and termination of ISWIM programs interpreted by Landin's SECD machine [10]. It is obtained from the λK-calculus by restricting the β-rule to redexes whose operand is a value (i.e. a constant, a variable or a function). The λ_V-calculus can be viewed as the abstract paradigm of functional programming languages with a lazy call-by-value evaluation mechanism. This calculus was extensively studied in [6]. It was shown there that, under very reasonable assumptions no fully abstract model exists in the category of C.P.O.'s and strict continuous function. This is due to the fact that there are too many continuous functions, some of which have a *parallel flavor* which allows to tell apart λ-terms which are observationally equal when evaluated sequentially. In this paper we investigate models for the λ_V-calculus in the category of coherence spaces [7, 8] and stable functions. Models built out of stable functions should not present the undesired phenomena of continuous functions, since parallel functions are not stable. The outcome however seems rather unsatisfactory at first. Not only it turns out that just modeling correctly the λ_V-calculus in coherence spaces is not immediate, for there is no standard way of modeling call-by-value evaluation and laziness. But moreover, even using this semantics we succeed only in defining a model which is correct but non-fully abstract with respect to the SECD machine. The model we analyze in detail turns out to be nonetheless very useful if taken together with the continuous one defined in [6]. The β_V-theories induced by the two models are in fact incomparable. Hence more sound equalities between λ-terms can be shown using the two semantics together. It comes as a pleasant and appropriate case at point, that the proof of the observational equivalence of the two terms which are used to show that the model of [6] is not fully abstract, can be readily obtained using the coherence semantics. This equivalence is proved in [6] using a lengthy and difficult syntactic computation induction, but it is immediate to check that the two terms have equal "coherent" denotations, and since the coherence model is correct they must also be observationally equal.

In conclusion, in this paper, we study the full abstraction problem for two models of different λ-calculi with respect to the observational equivalence induced by their natural evaluation machines. For simplicity we consider only calculi without constants. In particular for the λI-calculus we consider a machine that evaluates to β-normal form by reducing at each stage the leftmost redex and define a fully abstract model with respect to it. The correctness of the model will be shown using a computability argument inspired by that of [4]. To this end we define a finitary logical description of the model, i.e. an *intersection* type assignment system in which types can be seen as names for the compact elements of the semantic domain. As an interesting by-product of the correctness proof we obtain a new proof of the fact that in the λI-calculus the notion of strong normalization coincides with that of weak normalization. As far as the λ_V-calculus we follow quite closely [6] and prove an Approximation Theorem using indexed reductions. The novelty lies essentially in the construction of a correct coherence model using a particular form of stable functions as morphisms.

For lack of space we are forced to proceed directly in a terse definition/theorem style reducing proofs and comments to a minimum. We will use notation and conventions of [1] without explicit mention. In particular we will denote by $M[N/x]$ the capture avoiding substitution of all occurrences of the free variable x in M with N. We will denote with $FV(M)$ the set of free variables of the term M. Finally if $(D, \sqsubseteq)$ is a C.P.O. and $a, b \in D$, we denote by $f_{a,b}$ the step function $\boldsymbol{\lambda} d \in D. if\ a \sqsubseteq d\ then\ b\ else\ \perp$.

The authors would like to thank Mariangiola Dezani Ciancaglini and Simona Ronchi Della Rocca for many helpful discussions.

2 An Adequate Semantics for the $\boldsymbol{\lambda I}$-calculus

In this section we define an extensional model for the λI-calculus which models the $\lambda\eta$-theory which equates all λI-terms not having a normal form. Thus we solve positively the problem raised in [2]. What we achieve can be also viewed as the construction of a fully abstract model with respect to the observational equivalence induced by a natural reduction machine.

First we define the notion of model for the λI-calculus. Then we state the fundamental I-Separability Theorem, due to Barendregt [1]. Next we define the operational equivalence induced by the reduction machine which evaluates λI-terms to β-normal form by rewriting at each stage the leftmost redex. Then we define a model for the $\lambda\eta I$-calculus, D^I, as a particular solution, in the category of Complete Partial Orders (C.P.O.'s), of the domain equation $D \simeq [D \to_\perp D]$, where $[D \to_\perp D]$ is the space of strict continuous functions from D to D. Finally we show that the model D^I is fully abstract with respect to the above operational equivalence. For this purpose we introduce a finitary logical description of the model in the style of [4] and use an argument which is based on a generalization of Krivine's presentation of Tait's computability technique for showing strong normalization of the simply typed λ-calculus.

2.1 Preliminary Definitions and Results

Definition 1. i) The set Λ_I of terms of the λI-calculus is defined inductively as follows
- $x \in \Lambda_I$
- if $M \in \Lambda_I$ and $x \in FV(M)$ then $\lambda x.M \in \Lambda_I$
- if $M, N \in \Lambda_I$ then $(MN) \in \Lambda_I$.

ii) The theory λ_I of the λI-calculus is the equational theory defined as the congruence relation induced by the following reduction schemata:
$(\alpha)\quad \lambda x.M \to_I \lambda y.M[y/x],\quad$ if $y \notin FV(M)$;
$(\beta_I)\quad (\lambda x.M)N \to_I M[N/x]$.
A λ_I*-theory* is a binary congruence relation which extends λ_I. A λ_I-theory is said to be *extensional* if it is also closed under the following reduction schema
$(\eta)\quad \lambda x.Mx \to_I M,\quad$ if $x \notin FV(M)$.

We introduce now the definition of λI-model in terms of the notion of model for I-Combinatory Logic, cl_I (for the relations between λI-calculus and cl_I see [1]).

Definition 2. i) A model for cl_I is a structure $< D, \bullet >$, where $\bullet$ is a binary operation on D such that there are four elements $i, b, c, s \in D$ satisfying the following conditions:
- $i \bullet d = d$
- $((b \bullet d_0) \bullet d_1) \bullet d_2 = d_0 \bullet (d_1 \bullet d_2)$
- $((c \bullet d_0) \bullet d_1) \bullet d_2 = (d_0 \bullet d_2) \bullet d_1$
- $((s \bullet d_0) \bullet d_1) \bullet d_2 = (d_0 \bullet d_2) \bullet (d_1 \bullet d_2)$.

ii) A *combinatory model* for λI is a cl_I-model $< D, \bullet >$ such that there exists an element $\epsilon \in D$ with the following properties:
- $((\epsilon \bullet d_0) \bullet d_1) = d_0 \bullet d_1$
- $(\forall d \in D.(d_0 \bullet d = d_1 \bullet d)) \Rightarrow \epsilon \bullet d_0 = \epsilon \bullet d_1$
- $\epsilon \bullet \epsilon = \epsilon$.

iii) A model is said to be *trivial* if all closed terms have equal interpretation.
iv) A combinatory model for λI is *extensional* if $\epsilon = i$.

Contrary to what happens in the case of the standard λ-calculus, for a λI-model to be non trivial it is not sufficient that the model have more than one point. Moreover if any two elements among i, b, c, s are equal then i, b, c, s are all equal and the model is trivial. This can be seen also as an instance of the very important Theorem 1 below, due to H. Barendregt [1]. We need first a definition:

Definition 3. Let $\mathcal{F} = \{M_1, \ldots, M_n\} \subseteq \Lambda I$. The set $\mathcal{F}$ is said to be *I-separable* if and only if

$$\forall L_1, \ldots, L_n \in \Lambda I^\circ.(\exists F \in \Lambda I.FM_1 = L_1, \ldots, FM_n = L_n)\ .$$

Theorem 1 (I-Separability). *$\mathcal{F} \subseteq \Lambda I^\circ$ is I-separable if and only if $\mathcal{F}$ contains terms having distinct $\beta\eta$-normal forms.*

As a consequence of Theorem 1 we have also that a non trivial λI-model must be infinite. In fact, if a λI-model is finite, then there exist two distinct closed $\beta\eta$-normal forms which have the same interpretation; using the I-separability we can thus equate $\lambda x.x$ to $\lambda xyz.xz(yz)$ and get $i = s$. Another immediate consequence of Theorem 1 is the maximality of the theory that we model in this section.

We present now the operational semantics, in *S.O.S. style*, of a call-by-name sequential machine which reduces closed λI-terms to β-normal forms, by rewriting at each stage the leftmost redex.

Definition 4 (Operational Semantics). The set of programs is the set of closed λI-terms, Λ_I°. The set of values is the set of λI-terms in β-normal forms, NF_I. We define the leftmost reduction $\Downarrow_{LI}$ as the least binary relation over $\Lambda I \times NF_I$ satisfying the following rules:

$$(var) \qquad \frac{}{x \Downarrow_{LI} x}$$

$$(app1) \qquad \frac{M \Downarrow_{LI} \lambda x.P \quad P[N/x] \Downarrow_{LI} Q}{MN \Downarrow_{LI} Q}$$

$$(app2) \qquad \frac{M \Downarrow_{LI} P \quad P \text{ is not an abstraction} \quad N \Downarrow_{LI} Q}{MN \Downarrow_{LI} PQ}$$

$$(abstr) \qquad \frac{M \Downarrow_{LI} P}{\lambda x.M \Downarrow_{LI} \lambda x.P} \, .$$

We write simply $M \Downarrow_{LI}$ whenever M converges to a value, i.e. whenever the judgement $M \Downarrow_{LI} N$ can be established for some term N.

Once we have an operational semantics there is a natural operational equivalence to define on programs: the *observational equivalence*. This arises if we consider programs as *black boxes*. That is we take them to be equal if we cannot tell them apart by *observing* that for a given program context the machine halts when one is used as a subprogram but does not halt when the other is used as a subprogram. Thus we give:

Definition 5 (Observational Equivalence). i) Let $M, N \in \Lambda_I^\circ$. Then
$M \equiv_{LI} N$ iff $\forall C[\,].(C[M], C[N] \in \Lambda_I^\circ \Rightarrow (C[M] \Downarrow_{LI} \Leftrightarrow C[N] \Downarrow_{LI}))$.
ii) Let T_{LI} be the λI-theory $\{(M, N) \mid M, N \in \Lambda_I^\circ \wedge M \equiv_{LI} N\}$.

2.2 λI-calculus and λ_V-calculus Models Based on Strict Continuous Functions

In this subsection we study the problem of defining models for the λI-calculus and for the λ_V-calculus in the category of C.P.O.'s, by interpreting λ-abstractions as strict continuous functions. A continuous function is said to be *strict* if it maps $\perp$ to $\perp$. In particular we consider domains D such that the space $[D \to_\perp D]$ of strict continuous functions from D to D is a retract of D. We recall that a domain D' is a retract of a domain D ($D' \rhd D$) if there exist two continuous functions $\Phi : D' \to D$ and $\Psi : D \to D'$ such that $\Psi \circ \Phi = id_{D'}$. See [6] for details concerning the λ_V-calculus.

Definition 6. Let D be such that $[D \to_\perp D] \rhd D$. We denote with $\Phi : [D \to_\perp D] \to D$ and $\Psi : D \to [D \to_\perp D]$ the continuous functions such that $\Psi \circ \Phi = id_{[D\to_\perp D]}$. Let $\bullet : D \times D \to D$ be the function defined as $e \bullet d = Ap(\Psi(e), d)$. The interpretation function $[\![\ ]\!] : \Lambda I \times (Var \to D) \to D$ is defined as follows:
- $[\![x]\!]_\rho = \rho(x)$
- $[\![MN]\!]_\rho = [\![M]\!]_\rho \bullet [\![N]\!]_\rho$
- $[\![\lambda x.M]\!]_\rho = \Phi(strict(\boldsymbol{\lambda} d \in D.[\![M]\!]_{\rho[d/x]}))$,

where $\rho : Var \to D$ and $strict : [D \to D] \to [D \to_\perp D]$ is the function defined by: $strict(f) = \boldsymbol{\lambda} d.if\ d = \perp\ then\ \perp\ else\ f(d)$.

A weak version of Proposition 2 appears in [5]. We do not have enough space to introduce properly the λN°-calculus mentioned below. We limit ourselves to remark just that it is the calculus obtained from the λK-calculus by restricting the β-rule to redexes whose operand is a closed λK-normal form. It is a calculus in the spirit of the λ_V-calculus and constitutes the natural equational theory for reasoning about pure ISWIM programs, when these are evaluated by-value eagerly (i.e. reducing bodies of abstractions before application).

Proposition 2. *Let D be a domain such that $[D \to_\perp D] \rhd D$. Then*
i) D is a model for cl_I
ii) D is a model for the λN°-calculus
iii) If $\Phi \circ \Psi \in [D \to_\perp D]$, then D is a model for the λI-calculus; otherwise, if $\Phi \circ \Psi \notin [D \to_\perp D]$, then D is a λ_V-model.
iv) If D has at least three elements then it is a non trivial λI-model.

Proof. i) Take, for any ρ, $i = [\![\lambda x.x]\!]_\rho$, $b = [\![\lambda xyz.x(yz)]\!]_\rho$, $c = [\![\lambda xyz.xzy]\!]_\rho$, $s = [\![\lambda xyz.xz(yz)]\!]_\rho$.
ii) By induction on the structure of a λK-normal form N one can show that, if the model is not trivial, then $\exists \rho.[\![N]\!]_\rho \neq \perp$. Hence closed normal forms are erasable.
iii) If $\Phi \circ \Psi \in [D \to_\perp D]$ then $\Phi(\Phi \circ \Psi)$ satisfies the requirements for being the constant ϵ of Definition 2. If $\Phi \circ \Psi \notin [D \to_\perp D]$ then the interpretation of λ-abstractions is different from $\perp$ and hence we can model the λ_V-calculus by considering $V = D \setminus \{\perp\}$ as the set of semantic values and interpreting variables in V, see [6] for more details.
iv) One can easily see that say, $i \neq c$ since, taking a_1 to be the image of the identity function and a_2 to be image of $strict(\boldsymbol{\lambda} d.a_1)$, we have $a_2 a_1 \neq a_1 a_2$. $\square$

2.3 A Fully Abstract Model for the λI-calculus

We start by defining an extensional model for the λI-calculus as a particular solution of the domain equation $D \simeq [D \to_\perp D]$, in the category of C.P.O.'s and projection pairs.

Definition 7. The λI-model D^I is the structure $< D^I, \bullet^I >$, where:
- (D^I, Ψ, Φ) is the *inverse limit*, $lim_{\leftarrow}(D_n, i_n)$, where $D_0 = \{\bot, \phi, \top\}$ is a three point lattice with $\bot$ as the bottom element and $\top$ as the top element, and $i_0 : D_0 \to D_1$ is defined as
$i_0(\bot) = \bot$
$i_0(\phi) = f_{\top,\phi}$
$i_0(\top) = f_{\phi,\top}$;
- $\bullet^I : D^I \times D^I \to D^I$ is defined as $e \bullet^I d = (\Psi(e))(d)$, where $\Psi : D^I \to [D^I \to_\bot D^I]$;

The interpretation function in D^I will be denoted by $[\![\ ]\!]^I$.

Proposition 3. *D^I is an extensional combinatory model for the λI-calculus.*

Proof. Immediate using Proposition 2. □

Definition 8. Let T_{D^I} be the theory induced by the λI-model D^I, i.e.:

$$T_{D^I} = \{(M, N) \mid M, N \in \Lambda_I^\circ \wedge [\![M]\!]^I = [\![N]\!]^I\} \ .$$

We are now ready to state the *main theorem*

Theorem 4 (Full Abstraction). $T_{D^I} = T_{LI}$.

The rest of this section is devoted to the proof of the above theorem. We first prove that the model is correct, i.e. $T_{D^I} \subseteq T_{LI}$ by showing that if $M \in \Lambda I$, then: $(\exists \rho. [\![M]\!]^I_\rho \neq \bot)$ if and only if M is strongly normalizing. Continuity of contexts and the fact that M is strongly normalizing if and only if $M \Downarrow_{LI}$ then yield the result. We then show that the model is complete, i.e. $T_{D^I} \supseteq T_{LI}$, by showing that if M and N have different $\beta\eta$-normal forms then $\exists \rho. [\![M]\!]^I_\rho \neq [\![N]\!]^I_\rho$. The consistency of the model and Theorem 1 then yield the result. The proof of correctness is rather difficult. We use a generalization of the computability technique à la Krivine. To this end we need to define first a finitary logical presentation of the model D^I as intersection type assignment system, see [4] for more details.

The D^I Intersection Type Assignment System. Since the model D^I is an algebraic complete lattice we can define an *intersection type* language, $\mathcal{T}_I$, out of the names of the compact elements of the model different from $\bot$. We can then introduce a *type assignment system* S_I for assigning types of this language to λI-terms in such a way that a term M has type σ if and only if the compact element corresponding to σ is in the relation $\sqsubseteq$ with $[\![M]\!]^I_\rho$, for some ρ.

Definition 9. i) The set $\mathcal{T}_{\mathrm{I}}$ of types is the smallest set containing: the constants f and t and for all $\sigma, \tau \in \mathcal{T}_I$ the arrow type $\sigma \to \tau$ and the intersection type $\sigma \cap \tau$.
ii) The relation $\leq$ is the smallest reflexive and transitive relation on $\mathcal{T}_I$ satisfying

the following conditions for all $\tau, \sigma, \tau', \sigma', \rho \in \mathcal{T}_I$:

1. $\tau \leq f$
2. $\tau \geq t$
3. $t \to f \geq f$
4. $f \to t \leq t$
5. $(\sigma \leq \tau \wedge \sigma \leq \rho) \Rightarrow \sigma \leq (\rho \cap \tau)$
6. $\sigma \cap \tau \leq \sigma \quad \sigma \cap \tau \leq \tau$
7. $(\sigma \to \rho) \cap (\sigma \to \tau) \leq \sigma \to (\rho \cap \tau)$
8. $\sigma' \leq \sigma, \tau \leq \tau' \Rightarrow \sigma \to \tau \leq \sigma' \to \tau'$.

One can easily check that the relation $\leq$ makes $\mathcal{T}_I$ an inf-semilattice. Moreover the relation $\sim$ ($\tau \sim \sigma$ if and only if $\sigma \leq \tau$ and $\tau \leq \sigma$) is an equivalence relation.

Definition 10. i) Let a *basis* be a partial function from variables to types. A basis will be denoted by $\{x : \sigma \mid B(x) = \sigma\}$.
ii) The following type assignment system S_I establishes judgements of the shape $B \vdash M : \sigma$, where B is a basis, M is a λI-term and $\sigma \in T_I$. The rules are:

$$(var) \quad \frac{}{B \cup \{x : \sigma\} \vdash x : \sigma}$$

$$(\to E) \quad \frac{B \vdash M : \tau \to \sigma \quad B \vdash N : \tau}{B \vdash MN : \sigma}$$

$$(\to I) \quad \frac{B \cup \{x : \sigma\} \vdash M : \tau \quad x \in FV(M) \quad x \notin dom(B)}{B \vdash \lambda x.M : \sigma \to \tau}$$

$$(\cap I) \quad \frac{B \vdash M : \sigma \quad B \vdash M : \tau}{B \vdash M : \sigma \cap \tau} \qquad (\leq) \quad \frac{B \vdash M : \sigma \quad \sigma \leq \tau}{B \vdash M : \tau}$$

Definition 11. Let CI be the set of compact elements of $D^I \setminus \{\perp\}$. The function $* : T_I \to CI$ is inductively defined as follows:
- $f^* = \phi^*$
- $t^* = \top^*$
- $(\sigma \to \tau)^* = f_{\sigma^*, \tau^*}$
- $(\sigma \cap \tau)^* = \sigma^* \sqcup \tau^*$.

where ϕ^* and $\top^*$ are the images of ϕ and $\top$, respectively, under the canonical injection of D_0 into D^I .

Proposition 5. *i) The function $*$ is a surjection.*
ii) Let $\sigma, \tau \in \mathcal{T}_I$. Then $\sigma \leq \tau$ iff $\sigma^* \sqsupseteq \tau^*$.

Proof. i) The set CI can be expressed as $\bigsqcup_{n \in \omega} CI_n$, where CI_n is the image under the n-th projection of CI. The proof proceeds by induction on n, taking into account that an element of CI_n is a step function or a l.u.b of step functions.
ii) The implication ($\Rightarrow$) can be shown by induction on the length of the proof $\sigma \leq \tau$; the implication ($\Leftarrow$) can be shown by induction on the number of symbols in types. □

The following theorem, which asserts that the type assignment system S_I is a sound and complete logical description of the model D^I, motivates all the above definitions:

Theorem 6 (Faithfulness). *i) If $B \vdash M : \sigma$, then $\sigma^* \sqsubseteq [\![M]\!]^I_{\rho_B}$, where $\forall x \in dom(B).\rho_B(x) = B(x)^*$.*
ii) If $\sigma^* \sqsubseteq [\![M]\!]^I_\rho$, then there exists B such that $\forall x \in dom(B).B(x)^* \sqsubseteq \rho(x)$ and $B \vdash M : \sigma$.

Proof. i) By induction on the length of the proof $B \vdash M : \sigma$.
ii) By induction on the structure of M. □

Using a generalization of Krivine's computability technique we show now that if $(\exists \rho.[\![M]\!]^I_\rho \neq \perp)$ then M is strongly normalizing.

Definition 12. A set X of strongly normalizing λI-terms is *saturated* if:
i) if $x \in FV(M_1)$ and $(M_1[N/x])M_2 \ldots M_n \in X$, then $(\lambda x.M_1)NM_2 \ldots M_n \in X$;
ii) if $M_1, \ldots, M_n$ are strongly normalizing λI-terms, then $xM_1M_2 \ldots M_n \in X$.

Proposition 7. *The set Sat of saturated sets is a non empty complete lattice closed under the operation $\Rightarrow$ defined by:*

$$(B \Rightarrow C) = \{M \in \Lambda I \mid \forall N.(N \in B \Rightarrow MN \in C)\} \ .$$

Proof. The set of strongly normalizing terms is saturated. Saturated sets are closed under set-theoretic intersection. The rest is routine. □

We define now saturated sets, $Comp(\sigma)$, by induction on the structure of σ making sure that type equivalence is preserved. In particular, since $f \sim t \to f$ and $t \sim f \to t$, we must make sure that $Comp(f) = Comp(t \to f)$ and $Comp(t) = Comp(f \to t)$. We achieve this by defining a monotone operator over a suitable lattice derived from Sat and taking the pair $(Comp(f), Comp(t))$ to be a fixed point of that operator.

Definition 13. i)Let $(Satp, \leq)$ be the complete lattice consisting of all pairs (A, B) of saturated sets. We define

$$(A, B) \leq (A', B') \iff (A \supseteq A' \wedge B \subseteq B') \ .$$

ii)Let $R : (Satp, \leq) \to (Satp, \leq)$ be the operator defined as:

$$R(A, B) = ((B \Rightarrow A), (A \Rightarrow B)), \quad (A, B) \in Satp \ .$$

R is clearly monotonic, so it has least fixed point. It is worth noticing that the least fixed point is the pair consisting of the set of strongly normalizing λI-terms, and the set of those terms which yield strongly normalizing terms no matter how many times they are applied to strongly normalizing terms.

Definition 14. The sets $Comp(\sigma)$ are inductively defined as follows:
- $Comp(f) = A_0$
- $Comp(t) = B_0$
- $Comp(\sigma \to \tau) = (Comp(\sigma) \Rightarrow Comp(\tau))$
- $Comp(\sigma \cap \tau) = Comp(\sigma) \cap Comp(\tau)$,
where (A_0, B_0) is the least fixed point of the monotone operator R.

The above definition can be easily seen to be well posed using the following lemma, which is proved by induction on the structure of types:

Lemma 8. *If $\sigma \leq \tau$ then $Comp(\sigma) \subseteq Comp(\tau)$.*

Proof of the Full Abstraction of D^I.

Lemma 9. *i) Let $M, N_1, \ldots, N_n \in \Lambda I$ and $\{x_1, \ldots, x_n\} \supseteq FV(M)$. If $B \cup \{x_1 : \sigma_1, \ldots, x_n : \sigma_n\} \vdash M : \sigma$ is derivable in S_I and $\forall i. N_i \in Comp(\sigma_i)$, then $M[N_i/x_i] \in Comp(\sigma)$.*
ii)Let $M \in \Lambda I$. Then $\exists \rho. [\![M]\!]^I_\rho \sqsupseteq \phi$ if and only if M is strongly normalizing.

Proof. i) By induction on the length of the proof $B \cup \{x_1 : \sigma_1, \ldots, x_n : \sigma_n\} \vdash M : \sigma$, using Lemma 8.
ii) ($\Rightarrow$) Follows from faithfulness of S_I and from i) by considering the trivial substitution $M[x_i/x_i]$, since variables belong to every saturated set.
ii) ($\Leftarrow$) Consider the environment $\rho_\top$ mapping every variable in $\top$. By induction on the structure of normal forms M, it is easy to show that $[\![M]\!]^I_{\rho_\top} \sqsupseteq \phi$. □

The proof of the following Theorem is an interesting example of a *semantical* proof of a result which is normally proved *syntactically.*

Theorem 10 (Conservation). *Let $M \in \Lambda I$. M is weakly normalizing if and only if M is strongly normalizing.*

Proof. If M is weakly normalizing, then $\forall \rho. [\![M]\!]^I_\rho = [\![M']\!]^I_\rho$, for some normal form M'. But then by Lemma 9, M is strongly normalizing. □

Lemma 11. *Let $M \in \Lambda I$.*
i) $(\exists \rho. [\![M]\!]^I_\rho \neq \perp) \;\Leftrightarrow\; M \Downarrow_{LI}$.
ii) If M and N have different $\beta\eta$-normal forms, then $\exists \rho. [\![M]\!]^I_\rho \neq [\![N]\!]^I_\rho$.

Proof. i) From Lemma 9 and Theorem 10.
ii) Suppose that $\forall \rho. [\![M]\!]^I_\rho = [\![N]\!]^I_\rho$. We can assume that both M and N are closed, since otherwise we consider their closures, then, since M and N have different normal forms, using Theorem 1 we have that the model is trivial, but this contradicts Proposition 2. □

3 Modeling the λ_V-calculus in Coherence Spaces

Since the continuous semantics of λ_V-calculus is not entirely satisfactory *vis-à-vis* the full-abstraction problem, we investigate now the problem of modeling λ_V-calculus using coherence spaces and stable functions, see [8].

The λ_V-calculus is usually modeled in C.P.O's by taking the initial solution of the domain equation $D \simeq [D \rightarrow_\perp D]_\perp$, where $[D \rightarrow_\perp D]_\perp$ is the lifted space of strict continuous functions from D to D, see [6]. The new bottom

element, which results from lifting, is intended to denote closed terms which do not reduce to a value, i.e. undefined expressions, Thus λ_V-abstractions, which are values corresponding to *partial* functions, can be safely modeled as strict continuous functions. The immediate question which arises if we want to use coherence spaces, is then: how shall we denote undefined expressions and hence partial functions. The techniques used for Scott Domains cannot be mimicked directly in coherence spaces. In fact:
- if D, D' are coherence spaces, the natural operator *strict* defined as

$$strict(f) = \boldsymbol{\lambda} d.if\ d = \emptyset\ then\ \emptyset\ else\ f(d),$$

does not yield a stable function in general even if f is stable;
- the lifting operator takes us out of the category of coherence spaces.

We could try to define a different strict operator, manipulating directly the *traces* of the stable functions, but this would allow us just to model λI-calculus. Or else we could discuss models in the wider category of *D-I domains* and stable functions, but this would take us out of the mathematically attractive universe of coherence spaces. We will show in this section that, at the price of a slight modification of the category of coherence spaces, we can nevertheless succeed in defining an adequate notion of *partial* stable function.

In the next subsection, first we recall some definitions concerning the λ_V-calculus and we present the operational semantics determined by the SECD machine. We give then the main definitions and results concerning coherence spaces. These spaces are known in the literature also as binary qualitative domains. For further details and proofs concerning coherence spaces see [7, 8]. We introduce then a category derived from that of coherence spaces, where suitable notions of *lifted* coherence space and *partial* stable function can be defined, and where reflexive domain equations can be solved. We discuss a particular λ_V-model which is correct but not complete with respect to the observational equivalence induced by the SECD machine.

3.1 Operational Semantics and Observational Equivalence for the $\boldsymbol{\lambda_V}$-calculus

We will not give a formal introduction to the λ_V-calculus, the reader can see [12, 6] for more details. We just recall that the set of terms of the λ_V-calculus without constants is precisely the set of the λK-terms. The equational theory of the λ_V-calculus differs from that of the standard calculus in that the β-rule is replaced by the β_V-rule, i.e. $(\lambda x.M)N \rightarrow_V M[N/x]$ if and only if $N \in Val_V = \{\lambda x.M \mid M \in \Lambda\} \cup Var$. Where Val_V is called the set of *values*. Terms which reduce to values are called *valuable*.

The operational semantics induced by Landin's SECD machine on pure λ-terms can be defined as follows, in S.O.S. style:

Definition 15 (Operational Semantics). The set of programs is the set of closed λ-terms, Λ°. We define the lazy leftmost reduction $\Downarrow_{SECD}$ as the least binary relation over $\Lambda^\circ \times (Val_V \cap \Lambda^\circ)$ satisfying the following rules:

$$(abstr) \qquad \frac{}{\lambda x.P \Downarrow_{SECD} \lambda x.P}$$

$$(app) \qquad \frac{M \Downarrow_{SECD} \lambda x.P \quad N \Downarrow_{SECD} J \quad P[J/x] \Downarrow_{SECD} N}{MN \Downarrow_{SECD} N} \quad .$$

We write simply $M \Downarrow_{SECD}$ to indicate that M converges to a value.

Definition 16 (Observational Equivalence). i) $M \equiv_{SECD} N$ if and only if $\forall C[\ ].(C[M], C[N] \in \Lambda^\circ \Rightarrow (C[M] \Downarrow_{SECD} \Leftrightarrow C[N] \Downarrow_{SECD}))$.
ii) Let T_{SECD} be the λ-theory $\{(M,N) \mid M, N \in \Lambda^\circ \wedge M \equiv_{SECD} N\}$.

G.Plotkin, in [12] showed that the equational theory of the λ_V-calculus is strictly included in T_{SECD}. And hence that it is an equational theory for reasoning *correctly* about the observational equivalence.

3.2 Coherence Spaces and Stable Functions

Definition 17. i) A *coherence space* X is a set of sets which satisfies:
- *binary completeness*: if $M \subseteq A$ and if $\forall a, b \in M.(a \cup b \in A)$, then $\bigcup M \in A$.
- *down closure*: if $a \in X$ and $b \subseteq a$, then $b \in X$.

Definition 18. Let X be a coherence space:
i) We denote with X_{fin} the set of finite elements of X.
ii) Two elements $a, b \in X$ are said to be *compatible* iff $a \cup b \in X$.
iii) We denote with $\mid X \mid$ the set $\{z \mid \{z\} \in X\}$. The elements of $\mid X \mid$ are called the *atoms* of X.

A coherence space X is completely determined by the set $\mid X \mid$ of its atoms and by the compatibility relation $\frown$ between atoms of X defined as follows:

$$z \frown z'[modX] \text{ iff } \{z, z'\} \in X \quad .$$

Definition 19. Let X, Y be coherence spaces. A function $f : X \to Y$ is *stable* if and only if:

1. $\forall a, b \in X.(a \subseteq b \Rightarrow f(a) \subseteq f(b))$;
2. if $\{a_i\}_{i \in I}$ is a direct family of elements of X, then $f(\bigcup_{i \in I} a_i) = \bigcup_{i \in I} f(a_i)$;
3. $\forall a, b \in X.(a \cup b \in X \Rightarrow (f(a \cap b) = f(a) \cap f(b)))$.

The set of stable functions from X to Y will be denoted by $S(X, Y)$.

Definition 20. Let X, Y be coherence spaces.
i) Let $f : X \to Y$ be a stable function. The trace of f is the set:

$$Tr(f) = \{(a, z) \mid (a, z) \in X_{fin} \times \mid Y \mid \ , \ z \in f(a) \ , \ \forall a' \in X.(a' \subset a \Rightarrow z \notin f(a'))\}.$$

ii) The set of traces of stable functions from X to Y will be denoted by $[X \to Y]$.
iii) Stable functions can be ordered according to Berry's order relation $\sqsubseteq_s$ defined as follows: let $f, g : X \to Y$ be two stable functions, then

$$f \sqsubseteq_s g \ \text{ iff } \forall a, b \in X.(a \subseteq b \Rightarrow f(a) = f(b) \cap g(a)) \quad .$$

It is worth noticing that the pointwise ordering of functions is different from Berry's order. In fact if $f \sqsubseteq_s g$, then $\forall a \in X.(f(a) \subseteq g(a))$, but the converse is false.

Theorem 12 (Representation Theorem). *Let X, Y be coherence spaces.*
i) A stable function $f : X \to Y$ is determined by its trace $Tr(f)$, i.e.:
$f(a) = \{z \mid \exists d \in X_{fin}.(d \subseteq a \wedge (d, z) \in Tr(f))\},\ a \in X$.
ii) $[X \to Y]$ is a coherence space isomorphic to the set $S(X, Y)$ ordered by $\sqsubseteq_s$.

3.3 A "Coherent" Version of Lifted Spaces and Strict Functions

Since one cannot build a "lifted" coherence space simply by adding a new bottom element to the space, in order to define an adequate coherence semantics we decide to take an opposite view to that which inspires the construction in C.P.O.'s. Instead of adding a point which is intended to represent undefined expressions we *add* an atom whose rôle is to mark points which denote *defined expressions.* As a consequence we have to give up the requirement that there be only one semantic nonvalue. Formally we simulate the "lifted" coherence space by adding to a coherence space a new atom $*$ compatible with all the atoms of the space and take as set of *semantic values* the set of points to which the atom $*$ belongs.

Definition 21. Let X be a coherence space and let $* \notin \mid X \mid$. We denote with X_* the coherence space defined as follows:
- $\mid X_* \mid = \mid X \mid \cup \{*\}$;
- if $z_1, z_2 \in \mid X \mid$, then $z_1 \smile\!\!\!\!\frown z_2 [mod X_*]$ iff $z_1 \smile\!\!\!\!\frown z_2 [mod X]$;
- $* \smile\!\!\!\!\frown z [mod X_*]$, for all $z \in \mid X_* \mid$.

The new atom $*$ will be called the *added* atom of X_*. The space X_* has naturally associated constructor and destructor operators in_* and out_* which are stable and such that $(out_* \circ in_*)(a) = a$, for all $a \in X$:

Definition 22. Let X be a coherence space:
i) $in_* : X \to X_*$ is defined as $in_*(a) = a \cup \{*\}$, for $a \in X$
ii) $out_* : X_* \to X$ is defined as $out_*(a) = a \setminus \{*\}$, for $a \in X$.

Carrying further our idea for modeling λ_V-calculus, we try to interpret closed non valuable terms as $\emptyset$ and thus λ-abstractions as stable functions that return $\emptyset$ when the argument is a semantic nonvalue, i.e. a point to which the atom $*$ does not belong. Hence we define the analogue of the "strict" operator, as the operator which maps a stable function into a stable function which is extensionally equal to it on semantic values but which returns $\emptyset$ when the argument is a semantic nonvalue. We will call *partial stable* functions, stable functions that return $\emptyset$ when the argument is a semantic nonvalue. The results in the sequel will show that our technique is indeed adequate. Thus in order to model the λ_V-calculus, we try to solve the domain equation $D \simeq [D \to_{s*} D]_*$, where $[D \to_{s*} D]$ denotes the coherence space of traces of partial stable functions. The reason for considering partial stable functions instead of simply stable functions is motivated by the desire to keep superfluous points to a minimum. As a very interesting consequence we succeed in modeling also the η_V-rule, i.e. $\lambda x.Mx \rightarrow_V M$ if $x \notin FV(M)$ and $M \in Val_V$. In order to simplify the presentation we will solve the above domain equation in the category bqD_* of Pointed Coherence Spaces, defined below. One can also easily see that Girard's category bqD of coherence spaces and binary qualitative morphisms does not work immediately.

Proposition 13. *The category bqD_* is defined as follows:*
- the objects are the pairs $(D, *_D)$ where D is a coherence space and $*_D$ is an atom of D (the atom $*_D$ will be referred to as the base atom of the pointed space $(D, *_D)$);
- the arrows from $(D, *_D)$ to $(D', *_{D'})$ are the binary qualitative morphisms that map base atoms into base atoms, i.e. injective functions from $\mid D \mid$ to $\mid D' \mid$ which preserve compatibility and map base atoms to base atoms.

We will denote by $bqD_*((D, *_D), (D', *_{D'}))$ the set of binary qualitative morphisms from the pointed coherence space $(D, *_D)$ to the pointed coherence space $(D', *_{D'})$. Moreover we will denote by $S_{*_D}(D, D')$ the set of partial stable functions f from D to D', i.e. the set of stable functions such that, if $(a, z) \in Tr(f)$, then $*_D \in a$. We are now ready to define the function $partial_s$, which is the "coherent" analogue of the *strict* operator. One can easily check that:

Proposition 14. *Let $(D, *_D), (D', *_{D'})$ be pointed coherence spaces.*
i) The space of traces of partial stable functions from $(D, *_D)$ to $(D', *_{D'})$, denoted by $[D \rightarrow_{s*_D} D']$, is a coherence space.
ii) The function $partial_s : S(D, D') \rightarrow S_{*_D}(D, D')$ defined as follows, is stable:

$$partial_s(f) = \boldsymbol{\lambda} d \in (D, *_D).(if\ *_D \in d\ then\ f(d)\ else\ \emptyset), \quad f \in S(D, D') \ .$$

Finally we introduce the following useful binary qualitative morphisms and endofunctors over the category bqD_*.They are instrumental for the construction of the next subsection. It is easy to check that the definitions are well posed.

Definition 23. Let $(D, *_D), (D', *_{D'}), (E, *_E), (E', *_{E'})$ be objects of the category bqD_*. Let $f \in bqD_*((D, *_D), (D', *_{D'}))$ and $g \in bqD_*((E, *_E), (E', *_{E'}))$.
i) The morphism $(f \rightarrow_* g) \in bqD_*(([D \rightarrow_{s*_D} E], (\{*_D\}, *_E)), ([D' \rightarrow_{s*_{D'}} E'], (\{*_{D'}\}, *_{E'})))$ is defined as follows:

$$(f \rightarrow_* g)((a, z)) = (f^+(a), g(z)), \quad (a, z) \in \mid [D \rightarrow_{s*_D} E] \mid \ ,$$

where $f^+ : D \rightarrow D'$ is the stable function defined by $f^+(a) = \{f(z) \mid z \in a\}$.
ii) The morphism $f_* \in bqD((D_*, *), (D'_*, *))$ is defined as follows:
$f_*(z) = f(z), \quad \text{if } z \in \mid D \mid;$
$f_*(*) = *.$

Definition 24. i) Let $G_1 : bqD_* \times bqD_* \to bqD_*$ be the functor defined as follows:
$G_1((D,*_D),(E,*_E)) = ([D \to_{s*_D} E],(\{*_D\},*_E))$, for $(D,*_D),(E,*_E)$ pointed coherence spaces;
$G_1(f,g) = (f \to_* g)$, for $f \in bqD_*((D,*_D),(D',*_{D'}))$, $g \in bqD_*((E,*_E),(E',*_{E'}))$.
ii) Let $G_* : bqD_* \to bqD_*$ be the functor defined as follows:
$G_*((D,*_D)) = (D_*,*)$, for $(D,*_D)$ pointed coherence space;
$G_*(f) = f_*$, for $f \in bqD_*((D,*_D),(D',*_{D'}))$.
iii) Let $G : bqD_* \to bqD_*$ be the composition functor of G_1 and G_*:
$G = \boldsymbol{\lambda}((D,*_D) \in bqD_*).(G_* \circ G_1)((D,*_D),(D,*_D))$. Then
$G((D,*_D)) = ([D \to_{s*_D} D]_*,*)$, for $(D,*_D)$ pointed coherence space;
$G(f) = (f \to_* f)_*$, for $f \in bqD_*((D,*_D),(D,*_D))$.

3.4 A Coherence Model for the $\boldsymbol{\lambda_V}$-calculus

In this subsection we define a coherence model for the λ_V-calculus in the category bqD_* using the machinery introduced previously. More precisely we will take the initial solution in bqD_* of the reflexive equation $(D,*_D) \simeq ([D \to_{s*_D} D]_*,*)$, where the base atom of $([D \to_{s*_D} D]_*,*)$ is the added atom of $[D \to_{s*_D} D]_*$. Domain equations can be solved in this category with the usual "inverse limit" technique.

Proposition 15. *The domain equation $(D,*_D) \simeq ([D \to_{s*_D} D]_*,*)$ has initial solution in bqD_* equal to $lim_{\to}(D_n,i_n)$, where $(D_0,*_0)$ is a pointed coherence space with only one atom, $(D_{n+1},*_{n+1}) = ([D_n \to_{s*_n} D_n]_*,*)$, i_0 is the morphism which maps $*_0$ to $*_1$ and $i_{n+1} = (i_n \to_* i_n)_*$.*

Proof. Follows from the general Theorem in [13] since the category bqD_* is ω-complete and the functor G is ω-continuous (see [13] for further details). □

We have achieved our goal, finally! One can easily see, in fact, that the initial solution of the domain equation $(D,*_D) \simeq ([D \to_{s*_D} D]_*,*)$ in bqD_*, can be used to model adequately the λ_V-calculus. More formally one can show that the structure defined below is a *syntactical λ_V-model* in the sense of [6].

Definition 25. The λ_V-model D^V is the structure $< D^V, \{d \in D^V \mid * \in d\}, \bullet^V, [\![\ ]\!]^V >$, where:
- $((D^V,*),i_V)$ is the initial solution of the equation $(D,*_D) \simeq ([D \to_{s*_D} D]_*,*)$ in bqD_*;
- $\bullet^V : D^V \times D^V \to D^V$ is defined by:
$e \bullet^V d = if\ * \in d\ then\ Ap(out_*(i_V^+(e)),d)\ else\ \emptyset$,
where i_V^+ is the isomorphism from D^V to $[D^V \to_{s*} D^V]_*$ defined by $i_V^+(d) = \{i_V(z) \mid z \in d\}$.
- $[\![\ ]\!]^V : \Lambda \times (Var \to \{d \in D^V \mid * \in d\}) \to D^V$ is defined inductively as follows:
$[\![x]\!]^V_\rho = \rho(x)$
$[\![MN]\!]^V_\rho = [\![M]\!]^V_\rho \bullet^V [\![N]\!]^V_\rho$

$[\![\lambda x.M]\!]^V_\rho = i^-_V(in_*(partial_s(\boldsymbol{\lambda} d \in D^V.[\![M]\!]^V_{\rho_{ext}[d/x]})))$,
where $\rho_{ext} : Var \to D^V$ is the natural extension of ρ when the codomain is taken to be all of D^V and i^-_V is the isomorphism from $[D^V \to_{s*} D^V]_*$ to D^V defined by $i^-_V(d) = \{z \mid i_V(z) \in d\}$.

In the definition above and elsewhere we will take $[\![\ ]\!]^V$ to be sometimes defined on the whole $\Lambda \times (Var \to D^V)$, when necessary and clear from the context. The reader can check that this does not introduce ambiguities. In order to show that the structure D^V is well defined, one needs to prove that in general the function $\boldsymbol{\lambda} d_1 \ldots d_n.[\![M]\!]^V_{\rho_{ext}[d_1/x_1,\ldots,d_n/x_n]}$ is stable, for all $M \in \Lambda$, n-tuples of variables $x_1, \ldots, x_n$ and ρ's. This can be seen by structural induction on M. We omit this proof as well as the routine check that the model defined above is indeed a syntactical λ_V-model.

3.5 Correctness of the Coherence Semantics

In this subsection we outline the proof that the denotational semantics determined by D^V is correct with respect to the operational equivalence induced by Landin's SECD machine. In particular we show that if $T_{D^V} \subseteq T_{SECD}$, where T_{D^V} is the theory induced by D^V i.e. :

$$T_{D^V} = \{(M, N) \mid M, N \in \Lambda^\circ \wedge [\![M]\!]^V = [\![N]\!]^V\} \ .$$

First we need to recall that by the standardization theorem in [12], if $M \equiv_{SECD} N$ then $\forall C[\ ].(C[M], C[N] \in \Lambda^\circ \Rightarrow (C[M] \twoheadrightarrow_V Val_V \Leftrightarrow C[N] \twoheadrightarrow_V Val_V))\}$, where $\twoheadrightarrow_V$ denotes the reflexive, transitive and contextual closure of $\to_V$.
In order to prove the correctness result we now proceed as in [6], for the canonical λ_V-model in Scott domains. We start by proving an *Approximation Theorem* for the λ_V-model D^V, whereby the interpretation of a term can be expressed as the l.u.b. of the interpretations of its *approximants*. Approximants are defined in an extension of the λ_V-calculus, the λ^*_V-calculus, obtained by adding a distinguished constant Ω, whose intended meaning is the least element of the coherence space.

Definition 26 (λ^*_V-calculus). The set of λ^*_V-terms Λ^* is inductively defined as Λ, starting from $Var \cup \{\Omega\}$. The set of values is $Val^* = Var \cup \{\lambda x.M \mid M \in \Lambda^*\}$.
The rules of the approximate reduction are:
(α^*) $(\lambda x.M) \to^*_V (\lambda y.M[y/x])$, if $y \notin FV(M)$;
(β^*_V) $(\lambda x.M)N \to^*_V M[N/x]$, if $N \in Val^*$;
(Ωl) $\Omega M \to^*_V \Omega$;
(Ωr) $M\Omega \to^*_V \Omega$.
The set NF^*_V of terms of Λ^* in $\to^*_V$-normal is defined as follows:
- $\Omega \in NF^*_V$;
- $x \in NF^*_V$;
- $\xi M_1 \ldots M_m \in NF^*_V$, if $\xi \in Var$ and $M_i \not\equiv \Omega$, $M_i \in NF^*_V$ $\forall i = 1, \ldots, m$, or $\xi \equiv \boldsymbol{\lambda} x.P$, $P \in NF^*_V$, $M_i \not\equiv \Omega$, $M_i \in NF^*_V$ $\forall i = 1, \ldots, m$ and $M_1 \notin Val^*$;
- $\lambda x_1 \ldots x_n.M \in NF^*_V$, if $M \in NF^*_V$.

Definition 27. Let $M \in \Lambda^*$, the set of *approximants* of M is defined as follows: $A_V(M) = \{A \mid A \in NF_V^* \text{ and } A \text{ is obtained from } M',\ M' =_V M, \text{ by substituting } \Omega \text{ for some subterms of } M'\}$.

In order to prove the Approximation Theorem we need to introduce the notion of indexed term and indexed $\lambda\beta_V^*$-reduction. This is a standard tool for analyzing the fine structure of models obtained as limits, in that it allows to reason about n^{th}-*approximations* of interpretations of λ-terms.

Definition 28. i) The set of indexed λ-terms, Λ_{in}^*, is the language generated by the following grammar: $M \ ::= \ \Omega^n \mid x^n \mid (\lambda x.M)^n \mid (MM)^n$, for $n \in \omega$ and $x \in Var$.
ii) We denote by $er : \Lambda_{in}^* \to \Lambda^*$ the function that maps any term $M \in \Lambda_{in}^*$ to the term which is obtained from the indexed term M by erasing indices.
iii) We denote by $\Lambda_{in,exter}^*$ the set of terms obtained from indexed terms erasing only the outermost index.
iv) Let $M, N \in \Lambda_{in,exter}^*$, $x \in Var$, we define by $M^b[N^n/x^c]$ the result of the substitution of N for any free occurrence of x in M , in such a way that, if c is the index of a free occurrence of x, c is eliminated and the index of the term N is substituted for that occurrence of x is n.

Definition 29 (Indexed $\lambda\beta_V^*$-reduction). i) The rules of the indexed $\lambda\beta_V^*$-reduction are the following:
(I_α) $(\lambda x.M^n)^m \to_{IV} (\lambda y.(M^n[y^c/x^c]))^m$, where $M \in \Lambda_{in,exter}^*$, $y \notin FV(M)$;
$(I_{\beta V})$ $((\lambda x.M^n)^{m+1} N^p)^q \to_{IV} M^b[N^a/x^c]$, where $M, N \in \Lambda_{in,exter}^*$, $er(N^p) \in Val^*$, $p \geq 0$, $a = min(m,p,c)$, $b = min(m,n,q)$;
$(I_{\Omega l})$ $((\lambda x.M^n)^0 N^p)^q \to_{IV} \Omega^0$, where $M, N \in \Lambda_{in,exter}^*$.
ii) We denote with $\twoheadrightarrow_{IV}$ the contextual, reflexive and transitive closure of $\to_{IV}$.

Indexed λ-terms can be naturally interpreted in D^V:

Definition 30. Let $M \in \Lambda_{in}^*$. The interpretation $[\![M]\!]_\rho^V$ of M is inductively defined as follows:
- $[\![\Omega^n]\!]_\rho^V = \emptyset$
- $[\![x^n]\!]_\rho^V = (\rho(x))_n$
- $[\![(MN)^n]\!]_\rho^V = ([\![M]\!]_\rho^V \bullet^V [\![N]\!]_\rho^V)_n$
- $[\![(\lambda x.M)^n]\!]_\rho^V = (i_V^-(in_*(partial_s(\boldsymbol{\lambda} d \in D^V.[\![M]\!]_{\rho_{ext}[d/x]}^V))))_n$,

where the notation $(d)_n$ used above indicate the injection into D^V of the n-th projection of $d \in D^V$ onto D_n.

We can now prove the following results:

Proposition 16. *1. Let $M, M' \in \Lambda_{in}^*$, if $M \twoheadrightarrow_{IV} M'$, then $\forall \rho. [\![M]\!]_\rho^V = [\![M']\!]_\rho^V$.*
2. Let $M \in \Lambda^*$. Then, $\forall \rho$, $[\![M]\!]_\rho^V = \bigcup_{M':er(M')=M} [\![M']\!]_\rho^V$.
3. If $M \in \Lambda_{in}^*$, then there exists $A \in A_V(er(M))$ s.t. $\forall \rho. [\![M]\!]_\rho^V \subseteq [\![A]\!]_\rho^V$.
4. Let $M \in \Lambda^*$, then $\forall C[\ ] \forall \rho. ([\![C[\Omega]]\!]_\rho^V \subseteq [\![C[M]]\!]_\rho^V)$.

5. *For all $M \in \Lambda$, for all $A \in \mathcal{A}_V(M)$ and all ρ, $[\![A]\!]^V_\rho \subseteq [\![M]\!]^V_\rho$.*

Proof. 1. By induction on the number of reduction steps.
2. By induction on the structure of M, since the following is a direct family

$$\{Tr(\boldsymbol{\lambda} d.[\![M']\!]^V_{\rho_{ext}[d/x]})\}_{er(M')=M}$$

3. Follows form the facts that $\rightarrow_{IV}$-convertible indexed λ-terms have the same denotation and that indexed terms are normalizing. A proof of normalization for the indexed $\lambda\beta^*_V$-reduction can be achieved by showing that indexed β^*-reduction is strongly normalizing. Indexed $\lambda\beta^*_V$-reduction can be seen, in fact, as a particular reduction strategy of indexed β^*-reduction.
4. By induction on the structure of $C[\,]$, for any n-tuple $x_1,\ldots,x_n$, we have

$$Tr(\boldsymbol{\lambda} d_1\ldots d_n[\![C[\Omega]]\!]^V_{\rho_{ext}[d_1/x_1,\ldots,d_n/x_n]})\subseteq Tr(\boldsymbol{\lambda} d_1\ldots d_n[\![C[M]]\!]^V_{\rho_{ext}[d_1/x_1,\ldots,d_n/x_n]}).$$

5. Use the fact above and that β_V-convertible terms have the same denotation. □

Using Proposition 16 we can show immediately:

Theorem 17 (Approximation Theorem). *Let $M \in \Lambda$. Then, for all ρ,*

$$[\![M]\!]^V_\rho = \bigcup\{[\![A]\!]^V_\rho \mid A \in \mathcal{A}_V(M)\}.$$

Using the Approximation theorem we can now show that many interesting equations hold in the model. See [6] for some examples. In particular we can show the following lemma, which implies immediately the correctness of the denotational semantics w.r.t. the observational equivalence:

Lemma 18. *Let $M \in \Lambda$.*
i) If M is not valuable, then $\exists\rho.[\![M]\!]^V_\rho = \emptyset$.
ii) If M is valuable, then $\forall\rho.[\![M]\!]^V_\rho \neq \emptyset$.

Proof. i) Let M be not valuable. If $A \in \mathcal{A}_V(M)$, then $A \in NF^*_V$ and $A \notin Val^*$. By induction on the structure of $A \in NF^*_V \setminus Val^*$ one can show that, if ρ is the environment which maps all the variables to $\{*\}$, then $[\![A]\!]^V_\rho = \emptyset$. Hence using the Approximation theorem we have $[\![M]\!]^V_\rho = \emptyset$.
ii) From the definition of the λ_V-model D^V. □

Theorem 19 (Correctness Theorem). $T_{D^V} \subseteq T_{SECD}$.

A remark concerning coherence models like D^V is in order here. Contrary to what happens in Scott domains the relation $M \sqsubseteq^V N$ defined as $\forall\rho.[\![M]\!]^V_\rho \subseteq [\![N]\!]^V_\rho$ is not a congruence over Λ. In fact there exist λ-terms M, N such that $M \sqsubseteq^V N$, but $C[M] \not\sqsubseteq^V C[N]$, for some context $C[\,]$. For instance take $S_1 \equiv (\lambda xy.\Delta\Delta)(x(\lambda x.\Delta\Delta))$ and $S_2 \equiv \lambda x.\Delta\Delta$, where $\Delta \equiv \lambda x.xx$. Then for any ρ, $[\![S_1]\!]^V_\rho \subseteq [\![S_2]\!]^V_\rho$, but if we consider the context $C[\,] \equiv \lambda x.[\,]$ we have that $[\![C[S_1]]\!]^V_\rho$ and $[\![C[S_2]]\!]^V_\rho$ are incomparable. The reason is that there exist pointwise comparable functions with incomparable traces i.e. incomparable under Berry's order. In any case the relation $\sqsubseteq^V$ restricted to Λ° is a congruence relation.

3.6 The Interplay Between Different Correct but Incomplete Semantics

In this final subsection we prove first that the λ_V-model D^V is not fully-abstract with respect to the SECD operational equivalence. In particular we exhibit two operationally equivalent λ-terms, Q_1 and Q_2, which have different interpretations in the λ_V-model D^V. In order to show the operational equivalence of Q_1 and Q_2, we shall not embark in a cumbersome syntactical proof but show instead immediately that Q_1 and Q_2 have the same denotation in the canonical λ_V-model P, defined in [6] using C.P.O.'s. The counterexample to the completeness of D^V appears because there exist λ-definable stable functions which are pointwise comparable but incomparable under Berry's order.

Theorem 20 (Incompleteness Theorem). *The following λ-terms*
$Q_1 \equiv \lambda x.(\lambda xyz.\Delta\Delta)(xR_1)(xR_2)$ *and* $Q_2 \equiv \lambda x.(\lambda xy.\Delta\Delta)(xR_1)$,
where $R_1 \equiv \lambda z.((\lambda xy.\Delta\Delta)(z(\lambda x.\Delta\Delta)))$ *and* $R_2 \equiv \lambda xy.\Delta\Delta$,
are observationally equivalent but have different denotations in D^V.

Proof. The λ-terms $R_1 \equiv \lambda z.((\lambda xy.\Delta\Delta)(z(\lambda x.\Delta\Delta)))$ and $R_2 \equiv \lambda xy.\Delta\Delta$ when interpreted in D^V are incomparable under Berry's order. Hence there exists $d \in [D^V \rightarrow_{s*} D^V]_*$, say $\{*, ([\![R_1]\!]^V, *)\}$, such that $\emptyset = [\![Q_1]\!]^V \bullet^V d \neq [\![Q_2]\!]^V \bullet^V d = \{*\}$. But Q_1 and Q_2 are observationally equal, since in the canonical continuous λ_V-model P one has $[\![R_1]\!]^P \subseteq [\![R_2]\!]^P$. In fact, for all d $[\![xR_1]\!]^P_{\rho[d/x]} =\perp$ holds if and only if either $[\![xR_1]\!]^P_{\rho[d/x]} =\perp$ or $[\![xR_2]\!]^P_{\rho[d/x]} =\perp$, and hence $[\![Q_1]\!]^P = [\![Q_2]\!]^P$. □

What we have shown can be stated also as $T_{D^V} \not\supseteq T_{SECD}$ since $T_P \not\subseteq T_{D^V}$, where T_P is the theory induced by the model P. There is indeed a *chiasma* between P and D^V, both in their theories and in the technique used to show their non full abstraction. Along the same line of reasoning as above, we can in fact prove immediately that $T_P \not\supseteq T_{SECD}$, and hence the incompleteness of the λ_V-model P, by showing that the λ-terms M_1 and M_2 defined in [6], which have different denotations in P, have the same interpretation in D^V, i.e. $T_P \not\supseteq T_{D^V}$. The following semantic proof of the operational equivalence of M_1 and M_2 is certainly more easier than the intricate syntactical proof done in [6].

Theorem 21. *The following λ-terms*
$M_1 \equiv \lambda x.((\lambda xyz.\Delta\Delta)(x(\lambda x.\Delta\Delta)(\lambda xy.\Delta\Delta)))(x(\lambda xy.\Delta\Delta)(\lambda x.\Delta\Delta))$ *and*
$M_2 \equiv \lambda x.(\lambda xy.\Delta\Delta)(x(\lambda x.\Delta\Delta)(\lambda x.\Delta\Delta))$
have the same denotation in D^V and hence are observationally equivalent.

Proof. First, using the Approximation Theorem one computes that
$[\![\lambda x.\Delta\Delta]\!]^V_{\rho[d/x]} = \{*\}$ and $[\![\lambda xy.\Delta\Delta]\!]^V_{\rho[d/x]} = \{*, (\{*\}, *)\}$.
Then one shows that as far as the subterms of M_1 and M_2 :
$[\![x(\lambda x.\Delta\Delta)(\lambda xy.\Delta\Delta)]\!]^V_{\rho[d/x]}$ and $[\![x(\lambda xy.\Delta\Delta)(\lambda x.\Delta\Delta)]\!]^V_{\rho[d/x]}$ are semantic values if and only if $[\![x(\lambda x.\Delta\Delta)(\lambda x.\Delta\Delta)]\!]^V_{\rho[d/x]}$ is a semantic value. In fact:

- $[\![x(\lambda x.\Delta\Delta)(\lambda xy.\Delta\Delta)]\!]^V_{\rho[d/x]}$ is a semantic value iff $* \in d$ and $(\{*\},*) \in d$ and either $(\{*\},(\{*\},*)) \in d$ or $(\{*\},(\{(\{*\},*)\},*)) \in d$ or $(\{*\},(\{*,(\{*\},*)\},*)) \in d$;
- $[\![x(\lambda xy.\Delta\Delta)(\lambda x.\Delta\Delta)]\!]^V_{\rho[d/x]}$ is a semantic value iff $* \in d$ and $((\{*\},*) \in d \vee (\{(\{*\},*)\},*) \in d \vee (\{*,(\{*\},*)\},*) \in d)$ and $((\{*\},(\{*\},*)) \in d \vee (\{(\{*\},*)\},(\{*\},*)) \in d \vee (\{*,(\{*\},*)\},(\{*\},*)) \in d)$,
hence $(* \in [\![x(\lambda x.\Delta\Delta)(\lambda xy.\Delta\Delta)]\!]^V_{\rho[d/x]} \wedge * \in [\![x(\lambda xy.\Delta\Delta)(\lambda x.\Delta\Delta)]\!]^V_{\rho[d/x]})$ iff $(* \in d \wedge (\{*\},*) \in d \wedge (\{*\},(\{*\},*)) \in d)$, that is iff $* \in [\![x(\lambda x.\Delta\Delta)(\lambda x.\Delta\Delta)]\!]^V_{\rho[d/x]}$. $\square$

In conclusion we have that indeed the "coherent" model succeeds in equating observationally equivalent terms that the "continuous" model tells apart, as our original intuition suggested. And this is because the Scott-continuous functions which separate those terms are *indeed* parallel and hence not stable. But somewhat surprisingly Berry's order introduces perverse stable functions which tell apart observationally equivalent terms which are instead equated in the continuous model. Although neither the coherent model nor the continuous one are fully abstract, nevertheless the investigation we have carried out is rewarding.

References

1. Barendregt, H.: The Lambda Calculus, its Syntax and Semantics. North Holland. Amsterdam (1984)
2. Böhm, C.: Lambda Calculus and Computer Science Theory. LNCS **37** Springer-Verlag (1975)
3. Church, A.: The Calculi of Lambda Conversion. Princeton University Press Princeton (1941)
4. Coppo, M., Dezani-Ciancaglini, M., Zacchi, M.: Type Theories, Normal Forms and D_∞-Lambda-Models. Information and Computation **72(2)** (1987) 85–116
5. Dezani-Ciancaglini, M., Honsell, F., Ronchi Della Rocca, S.: Models for theories of functions depending on all their arguments (abstract). Journal of Symbolic Logic **51(3)** (1986) 399–402
6. Egidi, L., Honsell, F., Ronchi Della Rocca, S.: Operational, Denotational and Logical descriptions: a case study. Fundamenta Informaticae **16(2)** (1992) 149–169
7. Girard, J.Y.: The system F of variable types, fifteen years later. TCS **45** (1986) 159–192
8. Girard, J. Y., Lafont, Y., Taylor, P.: Proofs and Types. Cambridge University Press. Cambridge (1989)
9. Hindley, J. R., Longo, G.: Lambda Calculus Models and Extensionality. Z. Math. Logik Grundlag. Math. **26** (1980) 289–310
10. Landin, P. J.: The mechanical evaluation of expressions. Computer J. **6(4)** (1964) 308–320
11. Meyer, A.: What is a Model of the Lambda Calculus? Information and Control **52** (1982) 87–122
12. Plotkin, G. D.: Call-by-name, call-by-value and the λ-calculus. TCS **1** (1975) 125–159
13. Smith, M. B., Plotkin, G. D.: The category-theoretic solution of recursive domain equations. SIAM J. of Computing **11(5)** (1982) 761–783