

Intersection Types and Lambda Theories ^{*}

M.Dezani-Ciancaglini[†]

S.Lusin[‡]

Abstract

We illustrate the use of intersection types as a semantic tool for showing properties of the lattice of λ -theories. Relying on the notion of *easy intersection type theory* we successfully build a filter model in which the interpretation of an arbitrary simple easy term is any filter which can be described in a uniform way by a recursive predicate. This allows us to prove the consistency of a well-know λ -theory: this consistency has interesting consequences on the algebraic structure of the lattice of λ -theories.

Introduction

Intersection types were introduced in the late 70's by Dezani and Coppo [6, 8, 5], to overcome the limitations of Curry's type discipline. They are a very expressive type language which allows to describe and capture various properties of λ -terms. For instance, they have been used in [15] to give the first type theoretic characterization of *strongly normalizable* terms and in [9] to capture *persistently normalizing terms* and *normalizing terms*. See [10] for a more complete account of this line of research.

Intersection types have a very significant realizability semantics with respect to applicative structures. This is a generalization of Scott's natural semantics [16] of simple types. According to this interpretation types denote subsets of the applicative structure, an arrow type $A \rightarrow B$ denotes the sets of points which map all points belonging to the interpretation of A to points belonging to the interpretation of B , and an intersection type $A \cap B$ denotes the intersections of the interpretation of A and the interpretation of B . Building on this, intersection types have been used in [5] to give a proof of the completeness of the natural semantics of Curry's simple type assignment system in applicative structures, introduced in [16].

Intersection types have also an alternative semantics based on *duality* which is related to Abramsky's *Domain Theory in Logical Form* [1]. Actually it amounts to the application of that paradigm to the special case of ω -algebraic complete lattice models

^{*}Partially supported by MURST Cofin'00 AITCFA Project, MURST Cofin'01 COMETA Project, and by EU within the FET - Global Computing initiative, project DART IST-2001-33477.

[†]Dipartimento di Informatica, Università di Torino, Corso Svizzera 185, 10149 Torino, Italy
dezani@di.unito.it

[‡]Dipartimento di Informatica, Università di Venezia, via Torino 153, 30170 Venezia, Italy
slusin@oink.dsi.unive.it

of pure λ -calculus, [7]. Namely, types correspond to *compact elements*: the type Ω denoting the least element, intersections denoting *joins* of compact elements, and arrow types denoting *step functions* of compact elements. A typing judgment then can be interpreted as saying that a given term belongs to a pointed compact open set in a ω -algebraic complete lattice model of λ -calculus. By duality, type theories give rise to *filter λ -models*. Intersection type assignment systems can then be viewed as *finitary logical* descriptions of the interpretation of λ -terms in such models, where the meaning of a λ -term is the set of types which are deducible for it. This duality lies at the heart of the success of intersection types as a powerful tool for the analysis of λ -models, see [14] and the references there.

A key observation is that the λ -models we build out of intersection types differ only for the *preorder relations* between types. Changing these preorders in fact allow us to give different interpretations to λ -terms. In all these preorders crucial are the equivalences between atomic types and intersections of arrow types: therefore *type isomorphisms* are the corner stones of filter model constructions.

In [3] Alessi and Lusin faced the issue of easiness proofs of λ -terms from the semantic point of view (we recall that a closed term e is *easy* if, for any other closed term t , the theory $\lambda\beta + \{t = e\}$ is consistent). Actually the mainstream of easiness proofs is based on the use of syntactic tools (see [12] and the references there). Instead, very little literature can be found on easiness issues handled by semantic tools, we can mention the papers [17], [4], [2], [3].

Going in the direction of [2], in [3] Alessi and Lusin introduced the notion of *simple easiness*: roughly speaking, an unsolvable term e is simple easy if, for each filter model \mathcal{F}^∇ built on an easy intersection type theory Σ^∇ , any type Z in Σ^∇ , we can expand Σ^∇ to a new easy intersection type theory $\Sigma^{\nabla'}$ such that the interpretation of e in $\mathcal{F}^{\nabla'}$ is the sup of the old interpretation of e in \mathcal{F}^∇ and of the filter generated by Z .

A consequence is that simple easiness is a stronger notion than easiness. A simple easy term e is easy, since, given an arbitrary closed term t , it is possible to build (in a canonical way) a non-trivial filter model which equates the interpretation of e and t .

Besides of that, simple easiness is interesting in itself, since it has to do with minimal sets of axioms which are needed in order to give the easy terms certain types. This can be put at work to interpret easy terms by filters which can be described in an uniform way by recursive predicates.

Building on the duality between type intersections and joins, arrows and step functions, given an arbitrary simple easy term we build a λ -model in which this term is interpreted as the join. In this way we can prove the consistency of an interesting λ -theory. This consistency has been used in [13] to show that there exists a sublattice of the lattice of λ -theories which satisfies a restricted form of distributivity, called meet semidistributivity, and a nontrivial congruence identity (i.e., an identity in the language of lattices enriched by the relative product of binary relations).

The present paper is organized as follows. In Section 1 we present easy intersection type theories and type assignment systems for them. In Section 2 we introduce λ -models based on spaces of filters in easy intersection type theories. Section 3 gives the main contribution of the present paper: each simple easy term can be interpreted as an arbitrary filter which can be described in an uniform way by a recursive predicate. Finally in Section 4 we apply our result to show the consistency of a λ -theory which

has interesting consequences on the algebraic properties of the lattice of λ -theories.

1 Intersection Type Assignment Systems

Intersection types are syntactical objects built inductively by closing a given set \mathbb{C} of *type atoms* (constants) which contains the universal type Ω under the *function type* constructor \rightarrow and the *intersection type* constructor \cap .

Definition 1.1 (Intersection Type Language).

Let \mathbb{C} be a countable set of constants such that $\Omega \in \mathbb{C}$. The intersection type language over \mathbb{C} , denoted by $\mathbb{T} = \mathbb{T}(\mathbb{C})$ is defined by the following abstract syntax:

$$\mathbb{T} = \mathbb{C} \mid \mathbb{T} \rightarrow \mathbb{T} \mid \mathbb{T} \cap \mathbb{T}.$$

Notice that the most general form of an intersection type is a finite intersection of arrow types and type constants.

Notation Upper case Roman letters i.e. A, B, \dots , will denote arbitrary types. Greek letters will denote constants in \mathbb{C} . When writing intersection types we shall use the following convention: the constructor \cap takes precedence over the constructor \rightarrow and it associates to the right.

Much of the expressive power of intersection type disciplines comes from the fact that types can be endowed with a *preorder relation* \leq , which induces the structure of a meet semi-lattice with respect to \cap , the top element being Ω . We recall here the notion of *easy* intersection type theory as first introduced in [3].

Definition 1.2 (Easy intersection type theories).

Let $\mathbb{T} = \mathbb{T}(\mathbb{C})$ be an intersection type language. The easy intersection type theory (eitt for short) $\Sigma(\mathbb{C}, \nabla)$ over \mathbb{T} is the set of all judgments $A \leq B$ derivable from ∇ , where ∇ is a collection of axioms and rules such that (we write $A \sim B$ for $A \leq B$ & $B \leq A$):

1. ∇ contains the set $\overline{\nabla}$ of axioms and rules:

$$\begin{array}{ll}
(\text{refl}) & A \leq A & (\text{idem}) & A \leq A \cap A \\
(\text{incl}_L) & A \cap B \leq A & (\text{incl}_R) & A \cap B \leq B \\
(\text{mon}) & \frac{A \leq A' \quad B \leq B'}{A \cap B \leq A' \cap B'} & (\text{trans}) & \frac{A \leq B \quad B \leq C}{A \leq C} \\
(\Omega) & A \leq \Omega & (\Omega\text{-}\eta) & \Omega \leq \Omega \rightarrow \Omega \\
(\rightarrow\text{-}\cap) & (A \rightarrow B) \cap (A \rightarrow C) \leq A \rightarrow B \cap C & (\eta) & \frac{A' \leq A \quad B \leq B'}{A \rightarrow B \leq A' \rightarrow B'}
\end{array}$$

2. further axioms can be of the following two shapes only:

$$\begin{array}{l}
\psi \leq \psi', \\
\psi \sim \bigcap_{h \in H} (\xi_h \rightarrow E_h).
\end{array}$$

where $\psi, \psi', \xi_h \in \mathbb{C}$, $E_h \in \mathbb{T}$, and $\psi, \psi' \neq \Omega$;

3. ∇ does not contain further rules;
4. for each $\psi \neq \Omega$ there is exactly one axiom in ∇ of the shape $\psi \sim A$;
5. let ∇ contain $\psi \sim \bigcap_{h \in H} (\xi_h \rightarrow E_h)$ and $\psi' \sim \bigcap_{k \in K} (\xi'_k \rightarrow E'_k)$. Then ∇ contains also $\psi \leq \psi'$ iff for each $k \in K$, there exists $h_k \in H$ such that $\xi'_k \leq \xi_{h_k}$ and $E_{h_k} \leq E'_k$ are both in ∇ .

Notice that:

- (a) since $\Omega \sim \Omega \rightarrow \Omega \in \Sigma(\mathbb{C}, \nabla)$ by (Ω) and $(\Omega\text{-}\eta)$, it follows that all atoms in \mathbb{C} are equivalent to suitable (intersections of) arrow types;
- (b) \cap (modulo \sim) is associative and commutative;
- (c) in the last clause of the above definition E'_k and E_{h_k} must be constant types for each $k \in K$.

Notation When we consider an eitt $\Sigma(\mathbb{C}, \nabla)$, we will write \mathbb{C}^∇ for \mathbb{C} , \mathbb{T}^∇ for $\mathbb{T}(\mathbb{C})$ and Σ^∇ for $\Sigma(\mathbb{C}, \nabla)$. Moreover $A \leq_\nabla B$ will be short for $(A \leq B) \in \Sigma^\nabla$ and $A \sim_\nabla B$ for $A \leq_\nabla B \leq_\nabla A$. We will consider syntactic equivalence “ \equiv ” of types up to associativity and commutativity of \cap .

A nice feature of eitts is that the order between intersections of arrows agrees with the order between joins of step functions. This is stated in the following theorem, whose proof can be found in [3].

Theorem 1.3.

For all I , and $A_i, B_i, C, D \in \mathbb{T}^\nabla$,

$$\bigcap_{i \in I} (A_i \rightarrow B_i) \leq_\nabla C \rightarrow D \Rightarrow \exists J \subseteq I. C \leq_\nabla \bigcap_{i \in J} A_i \ \& \ \bigcap_{i \in J} B_i \leq_\nabla D,$$

provided that $D \not\leq_\nabla \Omega$.

Before giving the crucial notion of *intersection-type assignment system*, we introduce bases and some related definitions.

Definition 1.4 (Bases).

1. A ∇ -basis is a (possibly infinite) set of statements of the shape $x:A$, where $A \in \mathbb{T}^\nabla$, with all variables distinct.
2. If Γ is a ∇ -basis and $A \in \mathbb{T}^\nabla$ then $\Gamma, x:A$ is short for $\Gamma \cup \{x:A\}$ when $x \notin \Gamma$.

Definition 1.5 (The type assignment system).

The intersection type assignment system relative to the eitt Σ^∇ , notation $\lambda \cap^\nabla$, is a formal system for deriving judgements of the form $\Gamma \vdash^\nabla t : A$, where the subject t is an untyped λ -term, the predicate A is in \mathbb{T}^∇ , and Γ is a ∇ -basis. Its axioms and rules are

the following:

$$\begin{array}{ll}
(Ax) \frac{(x:A) \in \Gamma}{\Gamma \vdash^\nabla x:A} & (Ax-\Omega) \Gamma \vdash^\nabla t : \Omega \\
(\rightarrow I) \frac{\Gamma, x:A \vdash^\nabla t : B}{\Gamma \vdash^\nabla \lambda x.t : A \rightarrow B} & (\rightarrow E) \frac{\Gamma \vdash^\nabla t : A \rightarrow B \quad \Gamma \vdash^\nabla u : A}{\Gamma \vdash^\nabla tu : B} \\
(\cap I) \frac{\Gamma \vdash^\nabla t : A \quad \Gamma \vdash^\nabla t : B}{\Gamma \vdash^\nabla t : A \cap B} & (\leq_\nabla) \frac{\Gamma \vdash^\nabla t : A \quad A \leq_\nabla B}{\Gamma \vdash^\nabla t : B}
\end{array}$$

As usual we consider λ -terms modulo α -conversion. Notice that intersection elimination rules

$$(\cap E) \frac{\Gamma \vdash^\nabla t : A \cap B}{\Gamma \vdash^\nabla t : A} \quad \frac{\Gamma \vdash^\nabla t : A \cap B}{\Gamma \vdash^\nabla t : B}$$

can be immediately proved to be derivable in all $\lambda\cap^\nabla$.

We end this section by stating a Generation Theorem (proved in [3]).

Theorem 1.6 (Generation Theorem).

1. Assume $A \not\leq_\nabla \Omega$. $\Gamma \vdash^\nabla x : A$ iff $(x:B) \in \Gamma$ and $B \leq_\nabla A$ for some $B \in \mathbb{T}^\nabla$.
2. $\Gamma \vdash^\nabla tu : A$ iff $\Gamma \vdash^\nabla t : B \rightarrow A$, and $\Gamma \vdash^\nabla u : B$ for some $B \in \mathbb{T}^\nabla$.
3. $\Gamma \vdash^\nabla \lambda x.t : A$ iff $\Gamma, x : B_i \vdash^\nabla t : C_i$ and $\bigcap_{i \in I} (B_i \rightarrow C_i) \leq_\nabla A$, for some I and $B_i, C_i \in \mathbb{T}^\nabla$.
4. $\Gamma \vdash^\nabla \lambda x.t : B \rightarrow C$ iff $\Gamma, x : B \vdash^\nabla t : C$.

2 Filter Models

In this section we discuss how to build λ -models out of type theories. We start with the definition of *filter* for eitt's. Then we show how to turn the space of filters into an applicative structure. Finally we will define a notion of interpretation of λ -terms and state that we get λ -models (*filter models*).

Filter models arise naturally in the context of those generalizations of Stone duality that are used in representing domain theory in logical form (see [1], [7]). This approach provides a conceptually independent semantics to intersection types, the *lattice semantics*. Types are viewed as *compact elements* of domains. The type Ω denotes the least element, intersections denote joins of compact elements, and arrow types allow to internalize the space of continuous endomorphisms. Following the paradigm of Stone duality, type theories give rise to filter models, where the interpretation of λ -terms can be given through a finitary logical description.

Definition 2.1.

1. A ∇ -*filter* (or a *filter over* \mathbb{T}^∇) is a set $X \subseteq \mathbb{T}^\nabla$ such that:

- $\Omega \in X$;
 - if $A \leq_{\nabla} B$ and $A \in X$, then $B \in X$;
 - if $A, B \in X$, then $A \cap B \in X$;
2. \mathcal{F}^{∇} denotes the set of ∇ -filters over \mathbb{T}^{∇} ;
 3. if $X \subseteq \mathbb{T}^{\nabla}$, $\uparrow^{\nabla} X$ denotes the ∇ -filter generated by X ;
 4. a ∇ -filter is principal if it is of the shape $\uparrow^{\nabla} \{A\}$, for some type A . We shall denote $\uparrow^{\nabla} \{A\}$ simply by $\uparrow^{\nabla} A$.

It is well known that \mathcal{F}^{∇} is an ω -algebraic cpo, whose compact (or finite) elements are the filters of the form $\uparrow^{\nabla} A$ for some type A and whose bottom element is $\uparrow^{\nabla} \Omega$.

Next we endow the space of filters with the notions of application and of λ -term interpretation. Let $\text{Env}_{\mathcal{F}^{\nabla}}$ be the set of all mappings from the set of term variables to \mathcal{F}^{∇} .

Definition 2.2.

1. Application $\cdot : \mathcal{F}^{\nabla} \times \mathcal{F}^{\nabla} \rightarrow \mathcal{F}^{\nabla}$ is defined as

$$X \cdot Y = \{B \mid \exists A \in Y. A \rightarrow B \in X\}.$$

2. The interpretation function: $\llbracket \cdot \rrbracket^{\nabla} : \Lambda \times \text{Env}_{\mathcal{F}^{\nabla}} \rightarrow \mathcal{F}^{\nabla}$ is defined by

$$\llbracket t \rrbracket_{\rho}^{\nabla} = \{A \in \mathbb{T}^{\nabla} \mid \exists \Gamma \models \rho. \Gamma \vdash^{\nabla} t : A\},$$

where ρ ranges over $\text{Env}_{\mathcal{F}^{\nabla}}$ and $\Gamma \models \rho$ if and only $(x : B) \in \Gamma$ implies $B \in \rho(x)$.

3. The triple $\langle \mathcal{F}^{\nabla}, \cdot, \llbracket \cdot \rrbracket^{\nabla} \rangle$ is called the filter model over Σ^{∇} .

Notice that previous definition is sound, since it is easy to verify that $X \cdot Y$ is a ∇ -filter. The key property of \mathcal{F}^{∇} is to be a λ -model. This is proved in [3].

Theorem 2.3.

The filter model $\langle \mathcal{F}^{\nabla}, \cdot, \llbracket \cdot \rrbracket^{\nabla} \rangle$ is a λ -model, in the sense of Hindley-Longo [11], that is:

1. $\llbracket x \rrbracket_{\rho}^{\nabla} = \rho(x)$;
2. $\llbracket tu \rrbracket_{\rho}^{\nabla} = \llbracket t \rrbracket_{\rho}^{\nabla} \cdot \llbracket u \rrbracket_{\rho}^{\nabla}$;
3. $\llbracket \lambda x. t \rrbracket_{\rho}^{\nabla} \cdot X = \llbracket t \rrbracket_{\rho[x/x]}^{\nabla}$;
4. $(\forall x \in FV(t). \llbracket x \rrbracket_{\rho}^{\nabla} = \llbracket x \rrbracket_{\rho'}^{\nabla}) \Rightarrow \llbracket t \rrbracket_{\rho}^{\nabla} = \llbracket t \rrbracket_{\rho'}^{\nabla}$;
5. $\llbracket \lambda x. t \rrbracket_{\rho}^{\nabla} = \llbracket \lambda y. t[y/x] \rrbracket_{\rho}^{\nabla}$, if $y \notin FV(t)$;
6. $(\forall X \in \mathcal{F}^{\nabla}. \llbracket t \rrbracket_{\rho[x/x]}^{\nabla} = \llbracket u \rrbracket_{\rho[x/x]}^{\nabla}) \Rightarrow \llbracket \lambda x. t \rrbracket_{\rho}^{\nabla} = \llbracket \lambda x. u \rrbracket_{\rho}^{\nabla}$.

Moreover it is extensional, that is $\llbracket \lambda x. tx \rrbracket_{\rho}^{\nabla} = \llbracket t \rrbracket_{\rho}^{\nabla}$ when $x \notin FV(t)$.

3 Simple easy terms and filters

In this section we give the main notion of the paper, namely *simple easiness*. A term e is simple easy if, given any eitt Σ^∇ and a type $Z \in \mathbf{T}^\nabla$, we can extend in a conservative way Σ^∇ to $\Sigma^{\nabla'}$, so that $\llbracket e \rrbracket^{\nabla'} = \uparrow^{\nabla'} Z \sqcup \llbracket e \rrbracket^\nabla$. This allows to build with an uniform technique filter models in which the interpretation of e is a *filter of types induced by a recursive predicate* (see Definition 3.4).

Definition 3.1.

1. Let be Σ^∇ and $\Sigma^{\nabla'}$ two eitts. We say that $\Sigma^{\nabla'}$ is a conservative extension of Σ^∇ (notation $\Sigma^\nabla \sqsubseteq \Sigma^{\nabla'}$) iff $\mathbf{C}^\nabla \subset \mathbf{C}^{\nabla'}$ and for all $A, B \in \mathbf{T}^\nabla$,

$$A \leq_\nabla B \Leftrightarrow A \leq_{\nabla'} B.$$

2. A pointed eitt is a pair (Σ^∇, Z) with $Z \in \mathbf{T}^\nabla$.
3. A filter scheme is a mapping $S : PEITT \rightarrow EITT$, such that for any (Σ^∇, Z)

$$\Sigma^\nabla \sqsubseteq S(\Sigma^\nabla, Z),$$

where *EITT* and *PEITT* denote respectively the classes of eitts and pointed eitts.

We now give the central notion of *simple easy* term.

Definition 3.2.

An unsolvable term e is simple easy if there exists a filter scheme S_e such that for any pointed eitt (Σ^∇, Z) ,

$$\vdash^{\nabla'} e : B \iff \exists C \in \mathbf{T}^\nabla. C \cap Z \leq_{\nabla'} B \ \& \ \vdash^\nabla e : C,$$

where $\Sigma^{\nabla'} = S_e(\Sigma^\nabla, Z)$.

A first key property of easy terms is showed in [3].

Theorem 3.3.

With the same notation of previous definition, we have $\llbracket e \rrbracket^{\nabla'} = \uparrow^{\nabla'} Z \sqcup \llbracket e \rrbracket^\nabla$.

The last notion we need is that of filters induced by a recursive predicate.

Definition 3.4.

Let P be a recursive predicate defined on \mathbf{T}^∇ for all ∇ . The ∇ -filter induced by P is the filter defined by:

$$X_P^\nabla = \uparrow^\nabla \{A \in \mathbf{T}^\nabla \mid P(A)\}.$$

Theorem 3.5.

Let e be a simple easy term and P be as in previous definition. Then there is a filter model in which the interpretation of e is the filter induced by P .

Proof. Let $\langle \cdot, \cdot \rangle$ denotes any fixed bijection between $\mathbb{N} \times \mathbb{N}$ and \mathbb{N} such that $\langle r, s \rangle \geq r$.

We will define a denumerable sequence of eitts $\Sigma^{\nabla_0}, \dots, \Sigma^{\nabla_r}, \dots$. For each r we will consider a fixed enumeration $\langle W_s^{(r)} \rangle_{s \in \mathbb{N}}$ of the set $\{A \in \mathbb{T}^{\nabla_r} \mid A \notin \mathbb{T}^{\nabla_{r-1}} \& P(A)\}$ (for $r = 0$ the clause $A \notin \mathbb{T}^{\nabla_{r-1}}$ is vacuously true).

We can construct the model as follows.

step 0: take the eitt Σ^{∇_0} whose filter model is isomorphic to Scott D_∞ (see [7]):

- $\mathbb{C}^{\nabla_0} = \{\Omega, \omega\}$;
- $\nabla_0 = \overline{\nabla} \cup \{\omega \sim \Omega \rightarrow \omega\}$.

step $(n+1)$: if $n = \langle r, s \rangle$ we define $\Sigma^{\nabla_{n+1}} = \mathcal{S}_e(\Sigma^{\nabla_n}, W_s^{(r)})$ (notice that $\Sigma^{\nabla_n} \sqsubseteq \Sigma^{\nabla_{n+1}}$);

final step: take $\Sigma^{\nabla^*} = \Sigma(\bigcup_n \mathbb{C}^{\nabla_n}, \bigcup_n \nabla_n)$.

We prove first that the model \mathcal{F}^{∇^*} is non-trivial by showing that $\llbracket i \rrbracket^{\nabla^*} \neq \llbracket k \rrbracket^{\nabla^*}$, where $i \equiv \lambda x.x \ k \equiv \lambda xy.x$. Let $D \equiv (\omega \rightarrow \omega) \rightarrow (\omega \rightarrow \omega)$. Since $\vdash^{\nabla^*} i : D$, we have that $D \in \llbracket i \rrbracket^{\nabla^*}$. On the other hand, if it were $D \in \llbracket k \rrbracket^{\nabla^*}$, then it should exist n such that $D \in \llbracket k \rrbracket^{\nabla_n}$. This would imply (by applying several times the Generation Theorem) $\omega \rightarrow \omega \leq_{\nabla_n} \omega$. Since we have $\Sigma^{\nabla_n} \sqsubseteq \Sigma^{\nabla_{n+1}}$ for any n , we should have $\omega \rightarrow \omega \leq_{\nabla_0} \omega$. Since $\omega \sim_{\nabla_0} \Omega \rightarrow \omega$, we should conclude, by Theorem 1.3, $\Omega \leq_{\nabla_0} \omega$, which is a contradiction. Therefore we cannot have $D \in \llbracket k \rrbracket^{\nabla^*}$ and the model \mathcal{F}^{∇^*} is non-trivial.

Now we prove that $\llbracket e \rrbracket^{\nabla^*} = \uparrow^{\nabla^*} \{W_s^{(r)} \mid r, s \in \mathbb{N}\}$ by showing that $\llbracket e \rrbracket^{\nabla_n} = \uparrow^{\nabla_n} \{W_s^{(r)} \mid \langle r, s \rangle < n\}$ for all n . The inclusion (\supseteq) is immediate by construction. We prove (\subseteq) by induction on n . If $n = 0$, then $\llbracket e \rrbracket^{\nabla_0} = \uparrow^{\nabla_0} \Omega$, since \mathcal{F}^{∇_0} is the Scott D_∞ model, where all unsolvable terms are equated to bottom. Suppose the thesis true for $n = \langle r_n, s_n \rangle$ and let $B \in \llbracket e \rrbracket^{\nabla_{n+1}}$. Then $\vdash^{\nabla_{n+1}} e : B$. This is possible only if there exists $C \in \mathbb{T}^{\nabla_n}$ such that $C \cap W_{s_n}^{(r_n)} \leq_{\nabla_{n+1}} B$ and moreover $\vdash^{\nabla_n} e : C$. By induction we have $C \in \uparrow^{\nabla_n} \{W_s^{(r)} \mid \langle r, s \rangle < n\}$, hence $W_{s_1}^{(r_1)} \cap \dots \cap W_{s_k}^{(r_k)} \leq_{\nabla_n} C$ for some $r_1, \dots, r_k, s_1, \dots, s_k$ with $\langle r_i, s_i \rangle < n$ ($1 \leq i \leq k$). We derive $W_{s_1}^{(r_1)} \cap \dots \cap W_{s_k}^{(r_k)} \cap W_{s_n}^{(r_n)} \leq_{\nabla_{n+1}} B$, i.e. $B \in \uparrow^{\nabla_{n+1}} \{W_s^{(r)} \mid \langle r, s \rangle < n+1\}$.

Finally we show that $A \in \mathbb{T}^{\nabla^*}$ and $P(A)$ iff $A \equiv W_s^{(r)}$ for some r, s . If $A \in \mathbb{T}^{\nabla^*}$ then there is an r such that $A \in \mathbb{T}^{\nabla_r}$ and $A \notin \mathbb{T}^{\nabla_{r-1}}$. Moreover if $P(A)$ holds there is s such that $A \equiv W_s^{(r)}$. The vice versa is immediate. So we can conclude $\llbracket e \rrbracket^{\nabla^*} = \uparrow \{A \in \mathbb{T}^{\nabla^*} \mid P(A)\}$, i.e. $\llbracket e \rrbracket^{\nabla^*} = X_P^{\nabla^*}$.

4 An application to the consistency of λ -theories

We introduce now a λ -theory whose consistency has been first proved using a suitable filter model [13]. We obtain the same model here as a consequence of Theorem 3.5. Let $\Delta \equiv \lambda x.xx$.

Definition 4.1. *The λ -theory \mathcal{J} is axiomatized by*

$$\Delta \Delta x x = x; \quad \Delta \Delta x y = \Delta \Delta y x; \quad \Delta \Delta x (\Delta \Delta y z) = \Delta \Delta (\Delta \Delta x y) z.$$

It is clear that the previous equations hold if the interpretation of $\Delta\Delta$ is the join operator on filters. For using Theorem 3.5 we need:

- $\Delta\Delta$ to be simple easy;
- the join operator on filters to be a filter generated by a recursive predicate defined on all types.

The first condition is proved in [3]. For the second one it is easy to check that the join relative to \mathcal{F}^∇ is represented by the filter:

$$\uparrow^\nabla \{A \rightarrow B \rightarrow A \cap B\}.$$

Therefore the required predicate is

$$P(C) = (C \equiv A \rightarrow B \rightarrow A \cap B).$$

We can conclude:

Theorem 4.2. *The λ -theory \mathcal{J} is consistent.*

Previous result is used in [13] to show that there exists a sublattice of the lattice of λ -theories which satisfies a restricted form of distributivity, called meet semidistributivity, and a nontrivial congruence identity (i.e., an identity in the language of lattices enriched by the relative product of binary relations).

5 Conclusion

The notions of simple easy terms and filter models have been successfully applied to show easiness of λ -terms [3]. The present paper is a first step toward the application of this methodology for proving consistency of λ -theories. As a side-effect we showed that simple easiness is more general than easiness. The question whether easiness implies simple easiness remains open. An interesting research direction which we plan to follow is the characterization of the λ -theories whose consistency can be shown using the present approach or some generalizations of it.

References

- [1] S. Abramsky. Domain theory in logical form. *Ann. Pure Appl. Logic.* 51(1-2):1-77, 1991.
- [2] F. Alessi, M. Dezani-Ciancaglini. Filter models and easy terms. In *ICTCS'01*, vol. 2202 of *LNCS*, pages 17-37. Springer, Berlin, 2001.
- [3] F. Alessi, S. Lusin. Simple easy terms. In *ITRS'02*, vol. 70 of *ENTCS*. Elsevier, 2002.
- [4] J.Baeten and B. Boerboom. Ω can be anything it shouldn't be. *Indag. Math.*, 41:111-120, 1979.

- [5] H. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *J. Symbolic Logic*, 48(4):931-940, 1983.
- [6] M.Coppo and M. Dezani-Ciancaglini. An extension of the basic functionality theory for the λ -calculus. *Notre Dame J. Formal Logic*, 21(4):685-693, 1980.
- [7] M.Coppo, M. Dezani-Ciancaglini, F. Honsell, and G. Longo. Extended type structures and filter lambda models. In *Logic colloquium '82*, pages 241-262. North-Holland, Amsterdam, 1984.
- [8] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Principal type schemes and λ -calculus semantics. In *To H. B. Curry: essays on combinatory logic, lambda calculus and formalism*, pages 535-560. Academic Press, London, 1980.
- [9] M.Coppo, M. Dezani-Ciancaglini, and M. Zacchi. Type theories, normal forms, and D_∞ -lambda-models. *Inform. and Comput.*, 72(2):85-116, 1987.
- [10] M. Dezani-Ciancaglini, F. Honsell, and Y. Motohama. Compositional characterization of λ -terms using Intersection Types. In *MFCS'00*, vol. 1893 of *LNCS*, pages 304-313, Springer, 2000.
- [11] R. Hindley and G. Longo. Lambda-calculus models and extensionality. *Z. Math. Logik Grundlag. Math.*, 26(4):289-310, 1980.
- [12] J. Kuper. On the Jacopini technique. *Inform. and Comput.*, 138:101-123, 1997.
- [13] S. Lusin and A. Salibra, The lattice of lambda theories, submitted, 2002.
- [14] G. D. Plotkin. Set-theoretical and other elementary models of the λ -calculus. *Theoret. Comput. Sci.*, 121(1-2):351-409, 1993.
- [15] G. Pottinger. A type assignment for the strongly normalizable λ -terms. In *To H. B. Curry: essays on combinatory logic, lambda calculus and formalism*, pages 561-577. Academic Press, London, 1980.
- [16] D. Scott. Continuous lattices. In *Toposes, algebraic geometry and logic*, pages 97-136. vol. 274 of *LNM*. Springer, Berlin, 1972.
- [17] C. Zylberajch. *Syntaxe et Semantique de la Facilité en Lambda-calcul*. PhD thesis, Université Paris VII, 1991.