Intersection Types and Lambda Models

Fabio Alessi [a]

[a] *Dip. di Matematica e Informatica Via delle Scienze, 206*
33100 Udine (Italy) `alessi@dimi.uniud.it`

Franco Barbanera [b]

[b] *Dip. di Matematica e Informatica Viale A.Doria, 6*
95125 Catania (Italy) `barba@dmi.unict.it`

Mariangiola Dezani-Ciancaglini [c]

[c] *Dip. di Informatica Corso Svizzera, 185*
10149 Torino (Italy) `dezani@di.unito.it`

Research Report UDMI/06/2004/RR

Abstract

The invariance of the meaning of a λ-term by reduction/expansion w.r.t. the considered computational rules is one of the minimal requirements for a λ-model. Being the intersection type systems a general framework for the study of semantic domains for the Lambda-calculus, the present paper provides a characterisation of "meaning invariance" in terms of characterisation results for intersection type systems enabling typing invariance of terms w.r.t. various notions of reduction/expansion, like β, η and a number of relevant restrictions of theirs. Building on these results, the paper provides also necessary and sufficient conditions an (intersection) filter λ-structure needs to satisfy in order to be models of (restricted) λ-calculi.

1 Introduction

A fundamental notion in any computational model is that of *basic* computational step. For the Lambda-calculus, the computational model at the basis of the functional programming paradigm, such a notion is embodied by the β-reduction rule

$$(\lambda x.M)N \to_\beta M[x := N]$$

Even if such a notion of reduction can undergo a deeper analysis, as it has been done in various calculi which "decompose" it in more "atomic" steps (like the calculi of explicit substitutions, among which [1], [27], [26]), yet, up to now, no other reduction rule has been widely recognised as the core of the computational process for functional programs.

It is no surprise at all that any reasonable notion of computational step does not change the meaning of the object on which it operates, being a computational process intended simply to make such a meaning more *explicit*. It is hence clear why any classical formalisation of a semantics for the Lambda-calculus be required to be *adequate*, that is to provide the same "meaning" for *convertible* terms $M =_\beta N$, i.e. terms that can be obtained one out of the other by means of a number of β-reductions and β-expansions (where the notion of β-expansion is the obvious inverse of that of β-reduction).
"Meaning preservation" by β-reduction and "meaning preservation" by β-expansion are therefore very basic requirements whose deep investigation cannot be overlooked in any foundational study of the denotational semantics of functional programs. Moreover, a deeper understanding of these notions separately cannot lead but to a deeper understanding of the notion of adequacy and hence to a better understanding, in general, of the models for the Lambda-calculus. Such an investigation is indeed the general context of the present paper. However, it is very difficult to start investigating unless one does not manage to identify a finitary and natural framework where most of the many models proposed in the literature for the Lambda-calculus can be "embedded" and analysed.

Type systems for intersection types are definitely a framework with the qualities we are looking for: they form a class of type assignment systems for the untyped λ-calculus which allow to express, in a natural and finitary way, many of the most important *denotational* properties of terms (as a matter of fact also many relevant *operational* properties can be characterised by means of intersection types).
Indeed intersection types are a powerful tool both for the analysis and the synthesis of λ-models (see *e.g.* [9] [11], [18], [24], [23], [30], [15], [6] and the references there): on the one hand, intersection type disciplines provide finitary inductive definitions of interpretation of λ-terms in models. On the other hand, they are suggestive for the shape the domain model has to have in order to exhibit certain properties, see [11], [24], [5], [7], [17], [14].

Intersection types can be also looked at as a restriction of the domain theory in logical form, see [2], to the special case of modelling pure Lambda-calculus by means of ω-algebraic complete lattices. Many properties of these models can be proved using this paradigm, which goes back to Stone duality.

Different finitary characterisations of models for the Lambda-calculus can be obtained by introducing specific constants, typing rules and type preorders in the basic intersection type assignment system. An element of a particular domain, representing the denotational meaning of a term M, comes out to correspond to the set of types that can be inferred for M.

It is then clear that, in the framework of intersection type systems, the requirements of "meaning preservation" by β-reduction/expansion can be fully formalised in terms of typing invariance, namely, in type theory terminology, by the so called Subject reduction/expansion properties.

It is worth noticing that there have been devised in the literature also many restrictions of the β-rule, in order to formalise particular sorts of computations. Interesting examples of such restrictions are the rule of Plotkin's "call-by-value" λ-calculus [29], the rule of $\lambda\mathbf{I}$-calculus [13] and the rule of $\lambda\mathbf{KN}$-calculus [23]. In this paper we shall prove our results also for the restricted notions of computations embodied by the above mentioned calculi.

The contribution of the present paper to the investigation of the above mentioned "meaning preservation" requirements, is to try and isolate necessary and sufficient conditions to be satisfied by those domains that enable the "meaning" of a λ-term to be preserved by β-reduction/expansion. From our "intersection" perspective this will amount to proving in the paper a number of characterisation results for those intersection type systems enjoying Subject β-reduction/expansion properties.

Even if it possesses no real computational meaning, the η-rule (the formalisation of the notion of extensionality) plays a crucial role in denotational semantics. We then show also how to characterise the intersection type systems enjoying Subject η-reduction/expansion properties.

The present paper will also provide evidence for the statement we made before, that is that the sort of above mentioned characterisation results proved in the paper cannot but lead to a better understanding of the notion of λ-model itself. To this aim we consider the set of filters build on intersection types and we introduce a notion of application between filters. The λ-terms are naturally interpreted as the sets of types derivable for them. We call *filter λ-structures* the so obtained triple: set of filters, application, term interpretation. Building on the characterisation results for Subject β-reduction/expansion, we shall be able to provide necessary and sufficient conditions that have to be satisfied in order a filter (intersection) λ-structure be also a λ-model.

Even if of no real computational meaning, the η-rule (the formalisation of the notion of extensionality) plays a crucial role in denotational semantics. We then show also how to characterise the intersection type systems enjoying Subject η-reduction/expansion properties, as well as the filter λ-structures that are extensional λ-models for the considered calculi.

The paper is structured as follows: in Section 2 we recall the definitions of intersection types and intersection type preorders. We shall briefly recall also the main systems proposed in the literature, in particular those related to the use of intersection types for describing λ-models. We shall introduce conditions on type preorders to be used in our characterisation results. Section 3 discusses intersection type assignment systems and their properties. Section 4 is devoted to the characterisation results concerning β- and η-reduction/expansion. Our characterisations of filter λ-structures that are λ-models will be given in Section 5. A conclusions section (Section 6) will provide a few remarks on possible further research on the argument of the paper.

The present paper extends [4] by providing also the results concerning the characterisation of filter λ-structures. Moreover all the proofs omitted in [4] are present here.

2 Intersection Type Languages and Type Preorders

In this section we shall recall the main notions concerning intersection type languages and type preorders.

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

Definition 1 (Intersection Type Language) *The* intersection type language *over* $\mathbb{C}$*, denoted by* $\mathbb{T} = \mathbb{T}(\mathbb{C})$ *is defined by the following abstract syntax:*

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

NOTATION. Upper case Roman letters i.e. $A, B, \ldots$, will denote arbitrary types. When writing intersection types we shall use the following convention: the constructor $\cap$ takes precedence over the constructor $\rightarrow$ and $\rightarrow$ associates to the right. For example

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

In this paper we shall be concerned with several different intersection type languages arising from taking different sets of type atoms, depending on which typing invariance properties we want to capture. Typical choices for the set of type atoms are $\mathbb{C}_\infty$, a countable set of constants, or finite sets like $\{\Omega, \varphi, \omega\}$ or $\{\nu\}$.

Most of the expressive power of intersection type languages comes from the fact that they are endowed with a *preorder relation*, $\leq$, which induces, on the set of types, the structure of a meet semi-lattice with respect to $\cap$. This appears natural when we think of types as sets of denotations and interpret $\cap$ as set-theoretic intersection, and $\leq$ as set inclusion.

Definition 2 (Intersection Type Preorder)
An intersection type preorder $\Sigma = (\mathbb{C}^\Sigma, \leq_\Sigma)$ *is a binary relation* $\leq_\Sigma$ *on* $\mathbb{T}(\mathbb{C}^\Sigma)$ *satisfying the set of axioms and rules of Figure 1.*

(refl) $A \leq A$	(idem) $A \leq A \cap A$
(incl$_L$) $A \cap B \leq A$	(incl$_R$) $A \cap B \leq B$
(mon) $\dfrac{A \leq A' \quad B \leq B'}{A \cap B \leq A' \cap B'}$	(trans) $\dfrac{A \leq B \quad B \leq C}{A \leq C}$
(Ω) if $\Omega \in \mathbb{C}$ $\quad A \leq \Omega$	(ν) if $\nu \in \mathbb{C}$ $\quad A \to B \leq \nu$

Fig. 1. Basic Axioms and Rules of Type Preorders.

NOTATION. We will write $A \sim_\Sigma B$ for $A \leq_\Sigma B$ and $B \leq_\Sigma A$.
When clear from the context the notation $(\mathbb{C}, \leq)$ will be used for $(\mathbb{C}^\Sigma, \leq_\Sigma)$ and $\mathbb{T}$ for $\mathbb{T}(\mathbb{C})$.

Axiom (Ω) states that the type preorders containing the constant Ω have Ω itself as top element. It is particularly meaningful when used in combination with the Ω-type assignment systems, which essentially treat Ω as the universal type of all λ-terms (see Definition 14).

Axiom (ν) states that ν is above any arrow type. This axiom agrees with the ν-type assignment systems, which treat ν as the universal type of all λ-abstractions (see Definition 16). Notice that the role of ν may be played by the type $\Omega \to \Omega$, when Ω is in $\mathbb{C}$. For this reason it is of no use to have at the same time ν and Ω, hence we impose as pragmatic rule that the two constants do not occur together in any $\mathbb{C}$.

Notice that associativity and commutativity of $\cap$ (as always modulo $\sim$) follow easily from the above axioms and rules. For instance, commutativity is immediate:

$$A \cap B \leq (A \cap B) \cap (A \cap B) \leq B \cap A.$$

Being $\cap$ commutative and associative, we will write $\bigcap_{i \leq n} A_i$ for $A_1 \cap \ldots \cap A_n$. Similarly we shall write $\cap_{i \in I} A_i$, where I denotes always a finite set. Moreover we convene that $\cap_{i \in \emptyset} A_i$ is Ω when $\Omega \in \mathbb{C}$ and we forbid intersections on the empty set when $\Omega \notin \mathbb{C}$.

Remark 3 *It is not required that* $\sim$ *be congruent with the constructor* $\to$*. For many type preorders this will be implied by the extra axiom* (η) *or* $(\eta^\sim)$ *below.*

All the type preorders considered so far in the literature are defined for languages

over finite or countable sets of atoms and they are "generated" by recursive sets $\bigtriangledown$ of axioms and rules of the shape $A \leq B$ (where $\bigtriangledown$ it is said to generate $\leq$ when $A \leq B$ holds if and only if it can be derived from the axioms and rules of $\bigtriangledown$ together with those in Definition 2). Such generated preorders have been referred to as *type theories*. We will denote them by $\Sigma^{\bigtriangledown} = (\mathbb{C}^{\bigtriangledown}, \leq_{\bigtriangledown})$.
Note that there are only countably many possible $\bigtriangledown$; hence, there are uncountably many preorders which cannot be represented this way. Note also that the correspondence $\bigtriangledown \mapsto \leq_{\bigtriangledown}$ is not injective.
In this paper, instead, we try to be as general as possible, sticking to our notion of type preorder which indeed extends the notion of type preorders usually considered in the literature, where rules (Ω) and (ν) are not taken into account and are instead postulated inside the recursive sets generating the type preorder.

Figure 2 shows a list of special purpose axioms and rules which have been considered in the literature, and which we shall briefly discuss in the following.

$(\Omega\text{-}\eta)$	$\Omega \leq \Omega\to\Omega$	$(\to\text{-}\cap)$	$(A\to B)\cap(A\to C) \leq A\to B\cap C$
$(\Omega\text{-}lazy)$	$A\to B \leq \Omega\to\Omega$	$(\to\text{-}\cap^{\sim})$	$(A\to B)\cap(A\to C) \sim A\to B\cap C$
(η)	$\dfrac{A' \leq A \quad B \leq B'}{A\to B \leq A'\to B'}$	$(\eta^{\sim})$	$\dfrac{A' \sim A \quad B \sim B'}{A\to B \sim A'\to B'}$
$(\omega\text{-}Scott)$	$\Omega\to\omega \sim \omega$	$(\omega\text{-}Park)$	$\omega\to\omega \sim \omega$
$(\omega\varphi)$	$\omega \leq \varphi$	$(\varphi\to\omega)$	$\varphi\to\omega \sim \omega$
$(\omega\to\varphi)$	$\omega\to\varphi \sim \varphi$	(I)	$(\varphi\to\varphi)\cap(\omega\to\omega) \sim \omega$

Fig. 2. Possible Axioms and Rules concerning $\leq$.

The meaning of axioms and rules of Figure 2 can be grasped if we take types to denote subsets of a domain of discourse and we look at $\to$ as the function space constructor in the light of Curry-Scott semantics, see [32]. Thus the type $A\to B$ denotes the set of *total* functions which map each element of A into an element of B.

Since Ω represents the maximal element, i.e. the whole universe, then $\Omega\to\Omega$ is the set of functions which applied to an arbitrary element return again an arbitrary element. Thus, axiom $(\Omega\text{-}\eta)$ expresses the fact that all the objects in our domain of discourse are total functions, i.e. that Ω is equal to $\Omega\to\Omega$ [9]. If now we want to capture only those terms which truly represent functions, as we do for example in the lazy λ-calculus, we cannot assume axiom $(\Omega\text{-}\eta)$. One still may postulate the weaker property $(\Omega\text{-}lazy)$ to make all functions total [3]. It simply says that an element which is a function, because it maps A into B, maps also the whole universe into itself.

The intended interpretation of arrow types motivates axiom ($\to$-$\cap$), which implies that if a function maps A into B, and the same function maps also A into C, then, actually, it maps the whole A into the intersection between B and C (i.e. into $B\cap C$), see [9].

Rule (η) is also very natural in view of the set-theoretic interpretation. It implies that the arrow constructor is contra-variant in the first argument and covariant in the second one. It is clear that if a function maps A into B, and we take a subset A' of A and a superset B' of B, then this function will map also A' into B', see [9].

The rules ($\to$-$\cap^{\sim}$) and ($\eta^{\sim}$) are similar to the rules ($\to$-$\cap$) and (η). They capture properties of the graph models for the untyped lambda calculus, see [30] and [19].

The remaining axioms express peculiar properties of D_∞-like inverse limit models, see [9], [12], [11], [24], [22], [14].

We can introduce now a list of significant intersection type preorders which have been extensively considered in the literature. All these preorders have been introduced mainly to obtain corresponding filter models of (restricted) λ-calculi, as we will discuss in Section 5. The order is logical, rather than historical, and some references define the models, others deal with the corresponding filter models: [33], [10], [23], [24], [18], [3], [9], [31], [28], [11], [30], [19], [17].

These preorders are of the form $\Sigma^\nabla = (\mathbb{C}^\nabla, \leq_\nabla)$, with various different names ∇, picked for mnemonic reasons. In Figure 3 we list their sets of constants $\mathbb{C}^\nabla$ and their sets ∇ of extra axioms and rules taken from Figure 2. Here $\mathbb{C}_\infty$ is an infinite set of fresh atoms (i.e. different from $\Omega, \nu, \varphi, \omega$).

We define two conditions on type preorders to be used in our characterisation results for rule β.

Definition 4 (Beta and ν-sound Preorders)

(1) *A type preorder Σ is* beta *iff for all I, $A_i, B_i, C, D\in\mathbb{T}(\mathbb{C}^\Sigma)$:*

$$\bigcap_{i\in I}(A_i{\to}B_i) \leq_\Sigma C{\to}D \quad\Leftrightarrow\quad \bigcap_{i\in J} B_i \leq_\Sigma D \textit{ where } J = \{i{\in}I \mid C \leq A_i\}.$$

(2) *A type preorder Σ is ν-*sound *iff* $\nu \not\leq_\Sigma A{\to}B$ *for all* $A,\ B\in\mathbb{T}(\mathbb{C}^\Sigma)$.

A few comments on the previous definition. In the definition of beta preorders if J is empty and $\Omega\in\mathbb{C}^\Sigma$ we get $\Omega \sim_\Sigma D$. Instead by assumption J can never be empty when $\Omega \notin \mathbb{C}^\Sigma$. If we look at $\cap$ as representing join and arrow types as representing step functions, then the condition for a type preorder of being beta is exactly the relation which holds between sups of step functions [20].
The condition of being ν-sound is used to prevent both ν from being a redundant type and from assigning too many types to a λ-abstraction (assigning ν amounts

$\mathbb{C}^{\mathcal{B}a}$	$= \mathbb{C}_\infty$	$\mathcal{B}a$	$= \emptyset$
$\mathbb{C}^{\mathcal{CDV}}$	$= \mathbb{C}_\infty$	$\mathcal{CDV}$	$= \{(\to\text{-}\cap), (\eta)\}$
$\mathbb{C}^{\mathcal{HL}}$	$= \{\varphi, \omega\}$	$\mathcal{HL}$	$= \mathcal{CDV} \cup \{(\omega\varphi), (\varphi{\to}\omega), (\omega{\to}\varphi)\}$
$\mathbb{C}^{\mathcal{HR}}$	$= \{\varphi, \omega\}$	$\mathcal{HR}$	$= \mathcal{CDV} \cup \{(\omega\varphi), (\omega{\to}\varphi), (I)\}$
$\mathbb{C}^{\mathcal{EHR}}$	$= \{\nu\}$	$\mathcal{EHR}$	$= \mathcal{CDV} \cup \{(\nu)\}$
$\mathbb{C}^{\mathcal{AO}}$	$= \{\Omega\}$	$\mathcal{AO}$	$= \mathcal{CDV} \cup \{(\Omega), (\Omega\text{-}lazy)\}$
$\mathbb{C}^{\mathcal{BCD}}$	$= \{\Omega\} \cup \mathbb{C}_\infty$	$\mathcal{BCD}$	$= \mathcal{CDV} \cup \{(\Omega), (\Omega\text{-}\eta)\}$
$\mathbb{C}^{\mathcal{S}c}$	$= \{\Omega, \omega\}$	$\mathcal{S}c$	$= \mathcal{BCD} \cup \{(\omega\text{-}Scott)\}$
$\mathbb{C}^{\mathcal{P}a}$	$= \{\Omega, \omega\}$	$\mathcal{P}a$	$= \mathcal{BCD} \cup \{(\omega\text{-}Park)\}$
$\mathbb{C}^{\mathcal{CDZ}}$	$= \{\Omega, \varphi, \omega\}$	$\mathcal{CDZ}$	$= \mathcal{HL} \cup \mathcal{BCD}$
$\mathbb{C}^{\mathcal{P}l}$	$= \{\Omega, \varphi\}$	$\mathcal{P}l$	$= \{(\Omega), (\eta^\sim)\}$
$\mathbb{C}^{\mathcal{E}n}$	$= \{\Omega\} \cup \mathbb{C}_\infty$	$\mathcal{E}n$	$= \mathcal{P}l \cup \{(\to\text{-}\cap^\sim), (\Omega\text{-}\eta)\}$
$\mathbb{C}^{\mathcal{DHM}}$	$= \{\Omega, \varphi, \omega\}$	$\mathcal{DHM}$	$= \mathcal{BCD} \cup \{(\omega\varphi), (\omega\text{-}Scott), (\omega{\to}\varphi)\}$

Fig. 3. Particular Atoms, Axioms and Rules.

exactly to discriminating an abstraction and nothing more). Notice that Σ is trivially ν-sound when $\nu \notin \mathbb{C}^\Sigma$.

When $\Sigma = \Sigma^\nabla$, for some ∇, it is usually possible to prove the above defined conditions by induction on the derivation showing that two given types are in the preorder relation. The following notion of *strong beta* is handy in proving that some of the type preorders of Figure 3 are beta.

Definition 5 (Strong Beta Preorders)
A type preorder Σ^∇ is strong beta *iff $\nabla = \mathcal{BCD} \cup \nabla^-$ and:*

(1) ∇^- contains no rule and only axioms of one of the following two shapes:
- $\psi \leq \psi'$,
- $\psi \sim \bigcap_{i\in I}(\psi_i^{(1)} \to \psi_i^{(2)})$,

where $\psi, \psi', \psi_i^{(1)}, \psi_i^{(2)} \in \mathbb{C}^\nabla$, and $\psi, \psi', \psi_i^{(2)} \not\equiv \Omega$ for all $i \in I$;

(2) for each $\psi \in \mathbb{C}^\nabla$ such that $\psi \not\equiv \Omega$ there is exactly one axiom in ∇^- of the shape $\psi \sim \bigcap_{i\in I}(\psi_i^{(1)} \to \psi_i^{(2)})$;

(3) let ∇^- contain $\psi \sim \bigcap_{i\in I}(\psi_i^{(1)} \to \psi_i^{(2)})$ and $\psi' \sim \bigcap_{j\in J}(\psi_j'^{(1)} \to \psi_j'^{(2)})$. Then ∇^- contains also $\psi \leq \psi'$ iff for each $j \in J$ there exists $i \in I$ such that $\psi_j'^{(1)} \leq \psi_i^{(1)}$ and $\psi_i^{(2)} \leq \psi_j'^{(2)}$ are both in ∇^-.

For example the preorders $\Sigma^{\mathcal{HL}}$, $\Sigma^{\mathcal{S}c}$, $\Sigma^{\mathcal{P}a}$, $\Sigma^{\mathcal{CDZ}}$, $\Sigma^{\mathcal{HR}}$, and $\Sigma^{\mathcal{DHM}}$ are strong beta.

Lemma 6 *Each strong beta type preorder is beta.*

PROOF. We will denote by $\psi, \xi, \varphi, \zeta$ (possibly with indexes) elements of $\mathbb{C}^{\triangledown}$. By assumption for each constant $\psi \in \mathbb{C}^{\triangledown}$ there is exactly one axiom stating that ψ is equivalent to an intersection of arrow types. We denote by $\bigcap_{l \in L^{(\psi)}}(\xi^{(\psi)} \rightarrow \zeta^{(\psi)})$ such an intersection. Moreover, notice that the most general form of an intersection type is a finite intersection of arrow types and type constants. We can prove by simultaneous induction on the definition of $\leq$ two statements, the first of which implies the beta condition:

- if $(\bigcap_{i \in I}(A_i \rightarrow B_i)) \cap (\bigcap_{h \in H} \psi_h) \leq_{\triangledown} (\bigcap_{j \in J}(C_j \rightarrow D_j)) \cap (\bigcap_{k \in K} \varphi_k)$, then $(\bigcap_{i \in I'} B_i) \cap (\bigcap_{h \in H'}(\bigcap_{l \in L^{(\psi_h)'}} \zeta_l^{(\psi_h)})) \leq_{\triangledown} D_j$ where $I' = \{i \in I \mid C_j \leq_{\triangledown} A_i\}$, $H' = \{h \in H \mid C_j \leq_{\triangledown} \xi_l^{(\psi_h)}$ for some $l \in L^{(\psi_h)}\}$, $L^{(\psi_h)'} = \{l \in L^{(\psi_h)} \mid C_j \leq_{\triangledown} \xi_l^{(\psi_h)}\}$;
- if $(\bigcap_{i \in I}(A_i \rightarrow B_i)) \cap (\bigcap_{h \in H} \psi_h) \leq_{\triangledown} (\bigcap_{j \in J}(C_j \rightarrow D_j)) \cap (\bigcap_{k \in K} \varphi_k)$, then $(\bigcap_{i \in I'} B_i) \cap (\bigcap_{h \in H'}(\bigcap_{l \in L^{(\psi_h)'}} \zeta_l^{(\psi_h)})) \leq_{\triangledown} \zeta_m^{(\varphi_k)}$ where $I' = \{i \in I \mid \xi_m^{(\varphi_k)} \leq_{\triangledown} A_i\}$, $H' = \{h \in H \mid \xi_m^{(\varphi_k)} \leq_{\triangledown} \xi_l^{(\psi_h)}$ for some $l \in L^{(\psi_h)}\}$, $L^{(\psi_h)'} = \{l \in L^{(\psi_h)} \mid \xi_m^{(\varphi_k)} \leq_{\triangledown} \xi_l^{(\psi_h)}\}$.

Proposition 7

(1) All the type preorders of Figure 3 are beta.
(2) All the type preorders of Figure 3 are ν-sound.

PROOF. (1) For $\triangledown \in \{\mathcal{B}a, \mathcal{CDV}, \mathcal{EHR}, \mathcal{AO}, \mathcal{BCD}, \mathcal{P}l, \mathcal{E}n\}$ we can prove, by induction on the definition of $\leq_{\triangledown}$, that if $(\bigcap_{i \in I}(A_i \rightarrow B_i)) \cap (\bigcap_{h \in H} \psi_h) \leq_{\triangledown} (\bigcap_{j \in J}(C_j \rightarrow D_j)) \cap (\bigcap_{k \in K} \xi_k)$ then $\bigcap_{i \in I'} B_i \leq_{\triangledown} D_j$ where $I' = \{i \in I \mid C_j \leq_{\triangledown} A_i\}$.
The preorders $\Sigma^{\triangledown}$ for $\triangledown \in \{\mathcal{HL}, \mathcal{S}c, \mathcal{P}a, \mathcal{CDZ}, \mathcal{HR}, \mathcal{DHM}\}$ are beta by Lemma 6.
(2) For $\Sigma^{\mathcal{EHR}}$ one can easily show, by induction on $\leq_{\mathcal{EHR}}$, that $\nu \leq_{\mathcal{EHR}} A$ implies that A is an intersection of ν.

Example 8 *An example of a not beta preorder is $\Sigma^{\diamondsuit}$ defined by $\mathbb{C}^{\diamondsuit} = \{\Omega, \diamondsuit, \heartsuit\}$ and $\diamondsuit = \mathcal{BCD} \cup \{(\diamondsuit)\}$, where*

$$(\diamondsuit)\ A \leq A[\diamondsuit := \heartsuit].$$

$\Sigma^{\diamondsuit}$ is not beta, since $\diamondsuit \rightarrow \diamondsuit \leq_{\diamondsuit} \heartsuit \rightarrow \heartsuit$, but $\heartsuit \not\leq_{\diamondsuit} \diamondsuit$.

NOTATION. We write "the type preorder Σ validates $\triangledown$" to mean that all axioms and rules of $\triangledown$ are admissible in Σ.

In order to characterise the invariance of typing under η-expansion, we need to introduce a further condition on type preorders, which essentially says that each atomic type either is greater than or equal to a type which can be deduced for all terms which are abstractions or it is between two intersections of arrow types strictly related.

Definition 9 (Eta Preorders) *A type preorder* Σ *is* eta *iff for all* $\psi \in \mathbb{C}^{\Sigma}$ *either* $\bigcap_{i\in I}(A_i \to B_i) \leq_{\Sigma} \psi$ *for some* $I, A_i, B_i \in \mathbb{T}(\mathbb{C}^{\Sigma})$ *such that* $B_i \sim_{\Sigma} \Omega$ *for all* $i \in I$ *or* $\nu \leq_{\Sigma} \psi$ *or there exist non-empty families of types* $\{A_i, B_i\}_{i\in I}$, $\{D_{i,j}, E_{i,j}\}_{j\in J_i}$ *in* $\mathbb{T}(\mathbb{C}^{\Sigma})$ *such that*

$$\bigcap_{i\in I}(A_i \to B_i) \leq_{\Sigma} \psi \leq_{\Sigma} \bigcap_{i\in I}(\bigcap_{j\in J_i}(D_{i,j} \to E_{i,j})) \quad \&$$
$$\forall i \in I.\ A_i \leq_{\Sigma} \bigcap_{j\in J_i} D_{i,j} \ \&\ \bigcap_{j\in J_i} E_{i,j} \leq_{\Sigma} B_i.$$

It is easy to verify that if Σ validates $\mathcal{CDV}$ then the condition of the above definition simplifies to the requirement that all atomic types are either greater than $\Omega \to \Omega$ or greater than ν, or they are equivalent to a suitable intersection of arrow types, namely

$$\forall \psi \in \mathbb{C}^{\Sigma}. \Omega \to \Omega \leq_{\Sigma} \psi \text{ or } \nu \leq_{\Sigma} \psi \text{ or } \exists I, \{A_i, B_i\}_{i\in I}.\ \bigcap_{i\in I}(A_i \to B_i) \sim_{\Sigma} \psi.$$

The following proposition singles out all type preorders of Figure 3 which are eta: the proof is trivial.

Proposition 10 *If* $\triangledown \in \{\mathcal{HL}, \mathcal{EHR}, \mathcal{AO}, \mathcal{S}c, \mathcal{P}a, \mathcal{CDZ}, \mathcal{DHM}\}$, *then* $\Sigma^{\triangledown}$ *is an eta preorder.*

3 Intersection Type Assignments

We are now ready to introduce the crucial notion of *intersection type assignment system*. First we need a few preliminary definitions.

Definition 11 *(1) A* basis *over* $\mathbb{C}$ *is a set of statements of the shape* $x{:}B$, *where* $B \in \mathbb{T}(\mathbb{C}^{\Sigma})$, *all whose variables are distinct.*

(2) An intersection-type assignment system *relative to* Σ, *denoted by* $\lambda\cap^{\Sigma}$, *is a formal system for deriving judgements of the form* $\Gamma \vdash^{\Sigma} M : A$, *where the* subject M *is an untyped* λ*-term, the* predicate A *is in* $\mathbb{T}(\mathbb{C}^{\Sigma})$, *and* Γ *is a basis over* $\mathbb{C}^{\Sigma}$.

We will consider λ-terms up to α-conversion and we will assume the Barendregt convention on variables [8] to be fulfilled. The Barendregt convention for judgments $\Gamma \vdash^{\Sigma} M : A$ implies that variables occurring in the basis Γ cannot occur

$$(\mathrm{Ax}) \quad \Gamma \vdash^{\Sigma}_{\mathcal{B}} x{:}A \qquad \text{if } (x{:}A{\in}\Gamma)$$

$$(\to\mathrm{I}) \ \frac{\Gamma, x{:}A \vdash^{\Sigma}_{\mathcal{B}} M : B}{\Gamma \vdash^{\Sigma}_{\mathcal{B}} \lambda x.M : A{\to}B} \qquad (\to\mathrm{E}) \ \frac{\Gamma \vdash^{\Sigma}_{\mathcal{B}} M : A \to B \quad \Gamma \vdash^{\Sigma}_{\mathcal{B}} N : A}{\Gamma \vdash^{\Sigma}_{\mathcal{B}} MN : B}$$

$$(\cap\mathrm{I}) \ \frac{\Gamma \vdash^{\Sigma}_{\mathcal{B}} M : A \quad \Gamma \vdash^{\Sigma}_{\mathcal{B}} M : B}{\Gamma \vdash^{\Sigma}_{\mathcal{B}} M : A \cap B} \qquad (\leq) \ \frac{\Gamma \vdash^{\Sigma}_{\mathcal{B}} M : A \quad A \leq_{\Sigma} B}{\Gamma \vdash^{\Sigma}_{\mathcal{B}} M : B}$$

Fig. 4. The Axioms and Rules of the Basic Type Assignment System.

bound in the term M.

NOTATION. If Γ is a Σ-basis then $x{\in}\Gamma$ is short for $(x : A){\in}\Gamma$ for some A.
If Γ is a Σ-basis and $A{\in}\mathbb{T}(\mathbb{C}^{\Sigma})$ then $\Gamma, x : A$ is short for $\Gamma \cup \{x : A\}$ when $x \notin \Gamma$.
When $\Sigma = \Sigma^{\nabla}$ we shall denote $\lambda\cap^{\Sigma}$ and $\vdash^{\Sigma}$ by $\lambda\cap^{\nabla}$ and $\vdash^{\nabla}$, respectively.

Various type assignment systems can be defined, each of them parametrized w.r.t. a particular type preorder Σ. The simplest system is given in the following definition.

Definition 12 (Basic Type Assignment System)
Given a type preorder Σ, the axioms and rules of the basic type assignment system*, denoted by $\lambda\cap^{\Sigma}_{\mathcal{B}}$, for deriving judgements $\Gamma \vdash^{\Sigma}_{\mathcal{B}} M : A$, are shown in Figure 4.*

Example 13 *Self-application can be easily typed in $\lambda\cap^{\Sigma}_{\mathcal{B}}$, as follows.*

$$\dfrac{\dfrac{\dfrac{x{:}(A{\to}B) \cap A \vdash^{\Sigma}_{\mathcal{B}} x{:}(A{\to}B) \cap A}{x{:}(A{\to}B) \cap A \vdash^{\Sigma}_{\mathcal{B}} x{:}A{\to}B}\,(\leq) \quad \dfrac{x{:}(A{\to}B) \cap A \vdash^{\Sigma}_{\mathcal{B}} x{:}(A{\to}B) \cap A}{x{:}(A{\to}B) \cap A \vdash^{\Sigma}_{\mathcal{B}} x{:}A}\,(\leq)}{x{:}(A{\to}B) \cap A \vdash^{\Sigma}_{\mathcal{B}} xx{:}B}\,(\to\mathrm{E})}{\vdash^{\Sigma}_{\mathcal{B}} \lambda x.xx : (A{\to}B) \cap A{\to}B}\,(\to\mathrm{I})$$

If $\Omega{\in}\mathbb{C}$, a natural choice is to set Ω as the universal type of all λ-terms. This amounts to modify the basic type assignment system by adding a suitable axiom for Ω.

Definition 14 (Ω-type Assignment System)
Given a type preorder Σ with $\Omega{\in}\mathbb{C}^{\Sigma}$, the axioms and rules of the Ω-type assignment system *(denoted $\lambda\cap^{\Sigma}_{\Omega}$), for deriving judgements of the form $\Gamma \vdash^{\Sigma}_{\Omega} M : A$, are those of the basic one, plus the further axiom*

$$(Ax\text{-}\Omega) \ \ \Gamma \vdash^{\Sigma}_{\Omega} M : \Omega.$$

Example 15 *Also non-strongly normalising terms can be typed in $\lambda\cap^{\Sigma}_{\Omega}$ even with* *a type different from Ω. Note the usage of the axiom $(Ax\text{-}\Omega)$. Let $\Delta \equiv \lambda x.xx$.*

$$\dfrac{\dfrac{\dfrac{x{:}A, y{:}\Omega \vdash^{\Sigma}_{\Omega} x : A}{y{:}\Omega \vdash^{\Sigma}_{\Omega} \lambda x.x : A{\to}A}\,(\to\mathrm{I})}{\vdash^{\Sigma}_{\Omega} \lambda yx.x : \Omega{\to}A{\to}A}\,(\to\mathrm{I}) \qquad \vdash^{\Sigma}_{\Omega} \Delta\Delta : \Omega}{\vdash^{\Sigma}_{\Omega} (\lambda yx.x)(\Delta\Delta) : A{\to}A}\,(\to\mathrm{E})$$

Analogously to the case of Ω, when $\nu{\in}\mathbb{C}$, it is natural to consider ν as the universal type for abstractions, hence modifying the basic system by the addition of a special axiom for ν.

Definition 16 (ν-type Assignment System)
*Given a type preorder Σ with $\nu{\in}\mathbb{C}^{\Sigma}$, the axioms and rules of the ν-*type assignment system *(denoted $\lambda{\cap}^{\Sigma}_{\nu}$), for deriving judgements of the form $\Gamma \vdash^{\Sigma}_{\nu} M : A$, are those of the basic one, plus the further axiom*

$$(Ax\text{-}\nu)\;\; \Gamma \vdash^{\Sigma}_{\nu} \lambda x.M : \nu.$$

Example 17 *Using axiom $(Ax\text{-}\nu)$ we can again type non-strongly normalising terms. Notice that the term of Example 15 is not typable in $\lambda{\cap}^{\mathcal{EHR}}_{\nu}$, as proved in [18].*

$$\dfrac{\dfrac{\dfrac{x{:}A, y{:}\nu \vdash^{\Sigma}_{\nu} x : A}{y{:}\nu \vdash^{\Sigma}_{\nu} \lambda x.x : A{\to}A}\,(\to\mathrm{I})}{\vdash^{\Sigma}_{\nu} \lambda yx.x : \nu{\to}A{\to}A}\,(\to\mathrm{I}) \qquad \vdash^{\Sigma}_{\nu} \lambda z.\Delta\Delta : \nu}{\vdash^{\Sigma}_{\nu} (\lambda yx.x)(\lambda z.\Delta\Delta) : A{\to}A}\,(\to\mathrm{E})$$

For simplicity we assume the symbols Ω and ν to be reserved for the universal type constants respectively used in the systems $\lambda{\cap}^{\Sigma}_{\Omega}$ and $\lambda{\cap}^{\Sigma}_{\nu}$. i.e. we forbid $\Omega{\in}\mathbb{C}^{\Sigma}$ or $\nu{\in}\mathbb{C}^{\Sigma}$ when we deal with $\lambda{\cap}^{\Sigma}_{\mathcal{B}}$.

NOTATION. In the following $\lambda{\cap}^{\Sigma}$ will range over $\lambda{\cap}^{\Sigma}_{\mathcal{B}}$, $\lambda{\cap}^{\Sigma}_{\Omega}$ and $\lambda{\cap}^{\Sigma}_{\nu}$. More precisely we assume that $\lambda{\cap}^{\Sigma}$ stands for $\lambda{\cap}^{\Sigma}_{\Omega}$ whenever $\Omega{\in}\mathbb{C}^{\Sigma}$, for $\lambda{\cap}^{\Sigma}_{\nu}$ whenever $\nu{\in}\mathbb{C}^{\Sigma}$, and for $\lambda{\cap}^{\Sigma}_{\mathcal{B}}$ otherwise. Similarly for $\vdash^{\Sigma}$. If there is no danger of confusion, we write simply $\vdash$ for $\vdash^{\Sigma}$.

The subterm property does not hold in general for $\lambda{\cap}^{\Sigma}_{\nu}$. In fact $\lambda x.M$ is typable also when M is not typable. Moreover, in $\lambda{\cap}^{\Sigma}_{\Omega}$ and $\lambda{\cap}^{\Sigma}_{\nu}$, a judgement $\Gamma \vdash^{\Sigma} M : A$ does not imply $FV(M) \subseteq \Gamma$.

One of the most interesting features of intersection type systems is that of enabling precise characterisation results of many important sets of Lambda-terms, among which the one of Strongly-Normalizing terms. We state in the following theorem such a result that shall be needed in the next section, for a proof see [17]. [1]

Theorem 18 *(Characterisation of Strongly Normalising Terms) A λ-term $M{\in}\mathsf{SN}$ if and only if for all type preorders Σ there exist $A{\in}\mathbb{T}(\mathbb{C}^{\Sigma})$ and a $\mathbb{C}^{\Sigma}$-basis Γ such that $\Gamma \vdash^{\Sigma}_{\mathcal{B}} M : A$.*

We end this subsection defining the union between basis which requires some care in the presence of the intersection type constructor.

Definition 19
$$\begin{aligned}\Gamma_1 \uplus \Gamma_2 = {} & \{(x{:}A) \mid (x{:}A){\in}\Gamma_1 \textit{ and } x \notin \Gamma_2\} \cup \\ & \{(x{:}A) \mid (x{:}A){\in}\Gamma_2 \textit{ and } x \notin \Gamma_1\} \cup \\ & \{(x{:}A_1{\cap}A_2) \mid (x{:}A_1){\in}\Gamma_1 \textit{ and } (x{:}A_2){\in}\Gamma_2\}\end{aligned}$$

3.1 Admissible Rules

In this subsection we introduce a few relevant properties of intersection types, which we shall need for our characterisation results in the next section.

Many interesting type assignment rules can be proved to be admissible.

Proposition 20 (Admissible Rules) *For arbitrary intersection type preorders Σ the following rules are admissible in the intersection type assignment system $\lambda{\cap}^{\Sigma}$.*

$$(\cap \mathrm{E}_l)\ \frac{\Gamma \vdash^{\Sigma} M : A \cap B}{\Gamma \vdash^{\Sigma} M : A} \qquad (\cap \mathrm{E}_r)\ \frac{\Gamma \vdash^{\Sigma} M : A \cap B}{\Gamma \vdash^{\Sigma} M : B}$$

$$(\mathrm{W})\ \frac{\Gamma \vdash^{\Sigma} M : A \quad x \notin \Gamma}{\Gamma, x{:}B \vdash^{\Sigma} M : A} \qquad (\mathrm{S})\ \frac{\Gamma, x{:}B \vdash^{\Sigma} M : A \quad x \notin FV(M)}{\Gamma \vdash^{\Sigma} M : A}$$

$$(\mathrm{C})\ \frac{\Gamma, x{:}B \vdash^{\Sigma} M : A \quad \Gamma \vdash^{\Sigma} N : B}{\Gamma \vdash^{\Sigma} M[x := N] : A} \qquad (\leq \mathrm{L})\ \frac{\Gamma, x{:}B \vdash^{\Sigma} M : A \quad C \leq_{\Sigma} B}{\Gamma, x{:}C \vdash^{\Sigma} M : A}$$

In the following we shall freely use the rules of the above Proposition.

[1] The type systems considered in [17] are induced by type theories instead of type preorders, but the arguments given there to show the characterisation of strongly normalising terms extend without changes to type preorders.

We introduce now a few properties enabling to "reverse" some of the rules of the type assignment systems $\lambda\cap^{\Sigma}$, so as to achieve some form of generation (or inversion) lemmas (see Theorems 21 and 22).

Such properties are not trivial. For instance, for the arrow elimination rule, in general we can only say that when $\Gamma \vdash^{\Sigma} MN : A$, then there are a non-empty, finite set I and types B_i, C_i, such that for each $i{\in}I$, $\Gamma \vdash^{\Sigma} M : B_i{\to}C_i$, $\Gamma \vdash^{\Sigma} N : B_i$, and moreover $\bigcap_{i\in I} C_i \leq A$. Reasoning similarly on the rule $(\to\mathrm{I})$, one can conclude again that it cannot be reversed. More formally, we get the following theorem.

NOTATION. When we write "...assume $A \not\sim_{\Sigma} \Omega$..." we mean that this condition is always true when we deal with $\vdash^{\Sigma}_{\mathcal{B}}$ and $\vdash^{\Sigma}_{\nu}$, while it must be checked for $\vdash^{\Sigma}_{\Omega}$. Similarly, the condition $\nu \not\leq_{\Sigma} A$ must be checked just for $\vdash^{\Sigma}_{\nu}$.

Theorem 21 (Generation Lemma I) *Let Σ be a type preorder.*

(1) Assume $A \not\sim_{\Sigma} \Omega$. Then $\Gamma \vdash^{\Sigma} MN : A$ iff $\Gamma \vdash^{\Sigma} M : B_i{\to}C_i$, $\Gamma \vdash^{\Sigma} N : B_i$, and $\bigcap_{i\in I} C_i \leq_{\Sigma} A$ for some I non-empty and $B_i, C_i{\in}\mathbb{T}(\mathbb{C}^{\Sigma})$.
(2) Assume $\nu \not\leq_{\Sigma} A$. Then $\Gamma \vdash^{\Sigma} \lambda x.M : A$ iff $\Gamma, x{:}B_i \vdash^{\Sigma} M : C_i$, and $\bigcap_{i\in I}(B_i{\to}C_i) \leq_{\Sigma} A$ for some I non-empty and $B_i, C_i{\in}\mathbb{T}(\mathbb{C}^{\Sigma})$.

PROOF. The proof of each $(\Leftarrow)$ is easy. So we only treat $(\Rightarrow)$.

(1) By induction on derivations. The only interesting case is when $A \equiv A_1 \cap A_2$ and the last applied rule is $(\cap\mathrm{I})$:

$$(\cap\mathrm{I})\ \frac{\Gamma \vdash^{\Sigma} MN : A_1 \quad \Gamma \vdash^{\Sigma} MN : A_2}{\Gamma \vdash^{\Sigma} MN : A_1 \cap A_2}.$$

The condition $A{\not\sim_{\Sigma}}\Omega$ implies that we cannot have $A_1 \sim_{\Sigma} A_2 \sim_{\Sigma} \Omega$. We do the proof for $A_1{\not\sim_{\Sigma}}\Omega$ and $A_2{\not\sim_{\Sigma}}\Omega$, the other cases can be treated similarly. By induction there are I, B_i, C_i, J, D_j, E_j such that

$$\begin{array}{lll}
\forall i{\in}I. & \Gamma \vdash^{\Sigma} M : B_i{\to}C_i,\ \Gamma \vdash^{\Sigma} N : B_i, & \\
\forall j{\in}J. & \Gamma \vdash^{\Sigma} M : D_j{\to}E_j,\ \Gamma \vdash^{\Sigma} N : D_j, & \\
\bigcap_{i\in I} C_i \leq_{\Sigma} A_1 & \qquad\& & \bigcap_{j\in J} E_j \leq_{\Sigma} A_2.
\end{array}$$

So we are done since $(\bigcap_{i\in I} C_i) \cap (\bigcap_{j\in J} E_j) \leq A$.

(2) The proof is very similar to the proof of (1). It is again by induction on deriva-

tions and again the only interesting case is when the last applied rule is $(\cap \mathrm{I})$:

$$(\cap \mathrm{I})\ \frac{\Gamma \vdash^{\Sigma} \lambda x.M : A_1 \quad \Gamma \vdash^{\Sigma} \lambda x.M : A_2}{\Gamma \vdash^{\Sigma} \lambda x.M : A_1 \cap A_2}.$$

The condition $\nu \not\leq_{\Sigma} A$ implies that we cannot have $\nu \leq_{\Sigma} A_1$ and $\nu \leq_{\Sigma} A_2$. We do the proof for $\nu \not\leq_{\Sigma} A_1$ and $\nu \not\leq_{\Sigma} A_2$. By induction there are I, B_i, C_i, J, D_j, E_j such that

$$\forall i \in I.\ \Gamma, x{:}B_i \vdash^{\Sigma} M : C_i, \forall j \in J.\ \Gamma, x{:}D_j \vdash^{\Sigma} M : E_j,$$

$$\bigcap_{i \in I}(B_i \to C_i) \leq_{\Sigma} A_1 \quad \& \quad \bigcap_{j \in J}(D_j \to E_j) \leq_{\Sigma} A_2.$$

So we are done since $(\bigcap_{i \in I}(B_i \to C_i)) \cap (\bigcap_{j \in J}(D_j \to E_j)) \leq_{\Sigma} A$. The other two cases are easier. For instance, if $\nu \not\leq_{\Sigma} A_1$ and $\nu \sim_{\Sigma} A_2$, it is sufficient to take $\bigcap_{i \in I}(B_i \to C_i)$ above to conclude.

Using the properties introduced in Definition 4, we can give now a rather powerful generation lemma for $\lambda\cap^{\Sigma}$, which is one of the essential ingredients for the proofs of our results.

Special cases of this theorem have been previously proved in [9], [12], [11], [24], [18].

Theorem 22 (Generation Lemma II) *Let Σ be a type preorder.*

(1) Assume $A \not\sim_{\Sigma} \Omega$. Then $\Gamma \vdash^{\Sigma} x : A$ iff $(x{:}B) \in \Gamma$ and $B \leq_{\Sigma} A$ for some $B \in \mathbb{T}(\mathbb{C}^{\Sigma})$.
(2) Assume $A \not\sim_{\Sigma} \Omega$ and let Σ validate $\mathcal{CDV}$. Then $\Gamma \vdash^{\Sigma} MN : A$ iff $\Gamma \vdash^{\Sigma} M : B \to A$, and $\Gamma \vdash^{\Sigma} N : B$ for some $B \in \mathbb{T}(\mathbb{C}^{\Sigma})$.
(3) Let Σ be ν-sound and beta. Then $\Gamma \vdash^{\Sigma} \lambda x.M : B \to C$ iff $\Gamma, x{:}B \vdash^{\Sigma} M : C$.

PROOF. The proof of each $(\Leftarrow)$ is easy. So we only treat $(\Rightarrow)$.

(1) Easy by induction on derivations, since only the axioms (Ax), (Ax-Ω), and the rules $(\cap \mathrm{I})$, $(\leq)$ can be applied. Notice that the condition $A \not\sim_{\Sigma} \Omega$ implies that $\Gamma \vdash^{\Sigma} x : A$ cannot be obtained just using axioms (Ax-Ω).

(2) Let I, B_i, C_i be as in Theorem 21(1). Applying rule $(\cap \mathrm{I})$ to $\Gamma \vdash^{\Sigma} M : B_i \to C_i$ we can derive $\Gamma \vdash^{\Sigma} M : \bigcap_{i \in I}(B_i \to C_i)$, so by $(\leq)$ we have $\Gamma \vdash^{\Sigma} M : \bigcap_{i \in I} B_i \to \bigcap_{i \in I} C_i$, since $\bigcap_{i \in I}(B_i \to C_i) \leq_{\Sigma} \bigcap_{i \in I}(\bigcap_{i \in I} B_i \to C_i) \leq_{\Sigma} \bigcap_{i \in I} B_i \to \bigcap_{i \in I} C_i$ by rule (η) and axiom $(\to$-$\cap)$. We can choose $B = \bigcap_{i \in I} B_i$ and conclude $\Gamma \vdash^{\Sigma} M : B \to A$ since $\bigcap_{i \in I} C_i \leq_{\Sigma} A$.

(3) By the ν-soundness of Σ we cannot have $\nu \sim_{\Sigma} B \to C$. Let I, B_i, C_i be as in Theorem 21(2), where $A \equiv B \to C$. Then $\bigcap_{i \in I}(B_i \to C_i) \leq_{\Sigma} B \to C$ implies that

$\bigcap_{i\in J} C_i \leq_\Sigma C$ where $J = \{i\in I \mid B \leq_\Sigma B_i\}$, since Σ is beta. From $\Gamma, x{:}B_i \vdash^\Sigma M : C_i$ we can derive $\Gamma, x{:}B \vdash^\Sigma M : C_i$ using rule $(\leq \mathrm{L})$, so by $(\cap\mathrm{I})$ we have $\Gamma, x{:}B \vdash^\Sigma M : \bigcap_{i\in J} C_i$. Finally applying rule $(\leq)$ we can conclude $\Gamma, x{:}B \vdash^\Sigma M : C$.

4 Characterisation of Subject Reduction and Expansion

In the literature, to which we have provided many references in the previous sections, many models for the Lambda-calculus and a number of its restrictions have been shown to be finitary representable by means of (intersection) types. We now address the general issue of "meaning preservation" by reduction/expansion, dealt with in the Introduction, by characterising the intersection type systems in which types are preserved under various notions of reductions and expansions: β, η, together with some of their *restrictions* inspired by λ-calculi considered in the literature.

Let us first give the definitions of these redexes.

Definition 23 (Restricted Redexes)

(1) A redex $(\lambda x.M)N$ is a $\mathsf{var}\beta$-redex *if N is a variable.*
(2) A redex $(\lambda x.M)N$ is a $\mathsf{fun}\beta$-redex *if N is an abstraction.*
(3) A redex $(\lambda x.M)N$ is an $\mathsf{id}\beta$-redex *if $x\in FV(M)$.*
(4) A redex $(\lambda x.M)N$ is a $\mathsf{norm}\beta$-redex *if N is a closed strongly normalising term.*

The "call-by-value" λ-calculus is obtained by restricting to $\mathsf{var}\beta$- and $\mathsf{fun}\beta$-redexes (usually called β_v-redexes)[29], the $\lambda\mathbf{I}$-calculus by restricting to $\mathsf{id}\beta$-redexes (usually called $\beta\mathbf{I}$-redexes)[13] and the $\lambda\mathbf{K}\mathsf{N}$-calculus by restricting to $\mathsf{var}\beta$-, $\mathsf{id}\beta$- and $\mathsf{norm}\beta$-redexes [23].

We introduce rules of the form

$$(\mathsf{R}\text{-}exp)\quad \frac{M \to_{\mathsf{R}} N \qquad \Gamma \vdash N : A}{\Gamma \vdash M : A} \qquad (\mathsf{R}\text{-}red)\quad \frac{M \to_{\mathsf{R}} N \qquad \Gamma \vdash M : A}{\Gamma \vdash N : A}$$

where $\to_{\mathsf{R}}$ denote the reduction relation obtained by restricting the contraction to the set of R-redexes. Admissibility of the above rules in a type assignment is usually referred to as *subject expansion* and *subject reduction*, respectively.

Theorem 24 (Characterisation of Subject R-reduction/expansion)

(1) If $\Gamma \vdash^\Sigma M[x := N] : A$ then $\Gamma \vdash^\Sigma (\lambda x.M)N : A$ iff N is typable in the context Γ.

(2) *(R-expansion) Rule (R-exp) is admissible in $\lambda\cap^{\Sigma}$ iff the condition of* (1) *holds for all pairs of* R*-redexes and corresponding* R*-contracta.*

(3) *(R-reduction) Rule (R-red) is admissible in $\lambda\cap^{\Sigma}$ iff rule ($\to$I) can be reversed, i.e. for all Γ, M, A, B such that $(\lambda x.M)N$ is a* R*-redex for some N:*

$$\Gamma \vdash^{\Sigma} \lambda x.M : B{\to}A \;\Rightarrow\; \Gamma, x{:}B \vdash^{\Sigma} M : A.$$

PROOF. (1) ($\Rightarrow$) Clearly if N is not typable in the context Γ then also $(\lambda x.M)N$ has no type in Γ by Theorem 21(1).

($\Leftarrow$) Let D be a derivation of $\Gamma \vdash^{\Sigma} M[x := N] : A$ and $\Gamma_i \vdash^{\Sigma} N : B_i$ for $i{\in}I$ be all the statements in D whose subject is N. Without loss of generality we can assume that x does not occur in Γ.
If I is non-empty, notice that $\Gamma \subseteq \Gamma_i$ but $\Gamma \upharpoonright FV(N) = \Gamma_i \upharpoonright FV(N)$ (by $\Gamma \upharpoonright \mathcal{X}$ we denote $\{x : A{\in}\Gamma \mid x{\in}\mathcal{X}\}$). So using rules (S) and ($\cap$I), we have that $\Gamma \vdash^{\Sigma} N : \bigcap_{i\in I} B_i$. Moreover, one can easily see, by induction on M, that $\Gamma, x : \bigcap_{i\in I} B_i \vdash^{\Sigma} M : A$. Thus, by rule ($\to$I), we have $\Gamma \vdash^{\Sigma} \lambda x.M : \bigcap_{i\in I} B_i{\to}A$. Hence, by ($\to$E) we can conclude $\Gamma \vdash^{\Sigma} (\lambda x.M)N : A$.
If I is empty, we get from D a derivation of $\Gamma \vdash^{\Sigma} M : A$ by replacing each N by x. By assumption there exists a B such that $\Gamma \vdash^{\Sigma} N : B$. By rule (W) we get $\Gamma, x : B \vdash^{\Sigma} M : A$ and we can conclude as in previous case.

(2) The proof, by double induction on $\to_{\mathsf{R}}$ and on derivations, is straightforward.

(3) ($\Rightarrow$) Assume $\Gamma \vdash^{\Sigma} \lambda x.M : B{\to}A$, which implies $\Gamma, y{:}B \vdash^{\Sigma} (\lambda x.M)y : A$ by rule ($\to$E) for a fresh y. The admissibility of rule (R-red) gives us $\Gamma, y{:}B \vdash^{\Sigma} M[x := y] : A$. Hence $\Gamma, x{:}B \vdash^{\Sigma} M : A$.

($\Leftarrow$) It suffices to show that $\Gamma \vdash^{\Sigma} (\lambda x.M)N : A$ implies $\Gamma \vdash^{\Sigma} M[x := N] : A$. The case $A \sim_{\Sigma} \Omega$ is trivial for $\lambda\cap^{\Sigma}_{\Omega}$. Otherwise, by Theorem 21(2), there exists a finite set I and types B_i, C_i such that $\Gamma \vdash^{\Sigma} \lambda x.M : B_i{\to}C_i$, $\Gamma \vdash^{\Sigma} N : B_i$ and $\bigcap_{i\in I} C_i \leq_{\Sigma} A$. By hypothesis we get $\Gamma, x{:}B_i \vdash^{\Sigma} M : C_i$. Then $\Gamma \vdash^{\Sigma} M[x := N] : C_i$ follows by an application of rule (C), and so we can conclude $\Gamma \vdash^{\Sigma} M[x := N] : A$ using rules ($\cap$I) and ($\leq$).

Using Theorem 22(3) we immediately get a condition which assures Subject β-reduction.

Corollary 25 *If Σ is ν-sound and beta then rule (β-red) is admissible in $\lambda\cap^{\Sigma}$.*

The rather contrived statement given in Theorem 24(1) above is immediately met in $\lambda\cap^{\Sigma}_{\Omega}$, in $\lambda\cap^{\Sigma}_{\mathcal{B}}$ when $x{\in}FV(M)$ and in $\lambda\cap^{\Sigma}_{\nu}$ when N is an abstraction. We can prove the admissibility and non admissibility of restricted β-expansions for our type systems.

Corollary 26

(1) Rule ($\mathsf{var}\beta$-exp) is admissible in all $\lambda\cap^{\Sigma}_{\Omega}$, but never in $\lambda\cap^{\Sigma}_{\mathcal{B}}$ and $\lambda\cap^{\Sigma}_{\nu}$.
(2) Rule ($\mathsf{fun}\beta$-exp) is admissible in all $\lambda\cap^{\Sigma}_{\Omega}$ and $\lambda\cap^{\Sigma}_{\nu}$, but never in $\lambda\cap^{\Sigma}_{\mathcal{B}}$.
(3) Rule ($\mathsf{id}\beta$-exp) is admissible in all $\lambda\cap^{\Sigma}_{\mathcal{B}}$ and $\lambda\cap^{\Sigma}_{\Omega}$, but never in $\lambda\cap^{\Sigma}_{\nu}$.
(4) Rule ($\mathsf{norm}\beta$-exp) is admissible in all $\lambda\cap^{\Sigma}$.
(5) Rule (β-exp) is admissible in all $\lambda\cap^{\Sigma}_{\Omega}$, but never in $\lambda\cap^{\Sigma}_{\mathcal{B}}$ and $\lambda\cap^{\Sigma}_{\nu}$.

PROOF. Each of the four items but (4) follow from Theorem 24.

Item (4) is a consequence of Theorem 18, stating that each strongly normalising term is typable in all intersection type systems from a suitable basis. So all closed strongly normalising terms are typable in all intersection type systems starting from the empty basis.

For the non admissibility of rules ($\mathsf{var}\beta$-exp) and (β-exp) in $\lambda\cap^{\Sigma}_{\mathcal{B}}$ and in $\lambda\cap^{\Sigma}_{\nu}$, notice that we can always derive $\vdash^{\Sigma} \lambda x.x : A{\to}A$, but by the Generation Lemmas I and II (Theorems 21(1) and 22(1)) we cannot derive the same type for $(\lambda yx.x)z$ from the empty basis without using (Ax-Ω).

An example showing that ($\mathsf{fun}\beta$-exp) is not admissible in $\lambda\cap^{\Sigma}_{\mathcal{B}}$ is $\vdash^{\Sigma}_{\mathcal{B}} \lambda x.x : A \to A$ and $\not\vdash^{\Sigma}_{\mathcal{B}} (\lambda yx.x)(\lambda t.z) : A \to A$.

An example showing that ($\mathsf{id}\beta$-exp) is not admissible in $\lambda\cap^{\Sigma}_{\nu}$ is $\vdash^{\Sigma}_{\nu} \lambda x.z : \nu$ and $\not\vdash^{\Sigma}_{\nu} (\lambda yx.y)z : \nu$.

Notice that there are β-redexes that, without being $\mathsf{norm}\beta$-redexes, are typable whenever their contracta are. As an example take $(\lambda x.y)y$.

We end this section with the characterisation of Subject reduction/expansion for the η-rule.

Theorem 27 (Characterisation of Subject η-reduction/expansion)

(1) Rule (η-exp) is admissible in $\lambda\cap^{\Sigma}$ iff Σ is eta.
(2) Rule (η-red) is admissible in $\lambda\cap^{\Sigma}_{\mathcal{B}}$ iff Σ validates $\mathcal{CDV}$, in $\lambda\cap^{\Sigma}_{\Omega}$ iff Σ validates $\mathcal{BCD}$, and it is never admissible in $\lambda\cap^{\Sigma}_{\nu}$.

PROOF. (1) ($\Rightarrow$) Let $\diamondsuit{\in}\mathbb{C}^{\Sigma}$ be a constant that does not satisfy the first two conditions in Definition 9. We can derive $x{:}\diamondsuit \vdash^{\Sigma} x : \diamondsuit$. To derive $x{:}\diamondsuit \vdash^{\Sigma} \lambda y.xy : \diamondsuit$ by Theorem 21(2) we need I, A_i, B_i such that $x{:}\diamondsuit, y{:}A_i \vdash^{\Sigma} xy : B_i$ for all $i{\in}I$ and $\bigcap_{i\in I}(A_i{\to}B_i) \leq_{\Sigma} \diamondsuit$. This implies $B_i{\not\sim}_{\Sigma}\Omega$ for all $i{\in}I$, otherwise the first condition of Definition 9 would be satisfied. Now by Theorem 21(1) we get $x{:}\diamondsuit, y{:}A_i \vdash^{\Sigma} x : D_{i,j}{\to}E_{i,j}$, $x{:}\diamondsuit, y{:}A_i \vdash^{\Sigma} y : D_{i,j}$, and $\cap_{j\in J_i}E_{i,j} \leq_{\Sigma} B_i$ for some

$J_i, D_{i,j}, E_{i,j}$. By Theorem 22(1) we have $\diamondsuit \leq_\Sigma D_{i,j}{\to}E_{i,j}$ and $A_i \leq_\Sigma D_{i,j}$ for all $i{\in}I$ and $j{\in}J_i$. So we conclude

$$\begin{array}{c}\bigcap_{i\in I}(A_i{\to}B_i){\leq_\Sigma}\diamondsuit \leq_\Sigma \bigcap_{i\in I}(\bigcap_{j\in J_i}(D_{i,j}{\to}E_{i,j}))\\ \forall i{\in}I.\ A_i \leq_\Sigma \bigcap_{j\in J_i} D_{i,j}\ \&\ \bigcap_{j\in J_i} E_{i,j} \leq_\Sigma B_i.\end{array}$$

($\Leftarrow$) The proof that $\Gamma \vdash^\Sigma M : A$ implies $\Gamma \vdash^\Sigma \lambda x.Mx : A$, where x is fresh, is by induction on the structure of A. If A is a type constant, then we use the fact that Σ is eta in order to do the derivation discussed in the proof of ($\Rightarrow$). Suppose that $\Gamma \vdash^\Sigma M : \psi$ for some $\psi{\in}\mathbb{C}^\Sigma$ such that:

$$\begin{array}{c}\bigcap_{i\in I}(A_i{\to}B_i){\leq_\Sigma}\psi \leq_\Sigma \bigcap_{i\in I}(\bigcap_{j\in J_i}(D_{i,j}{\to}E_{i,j}))\\ \forall i{\in}I.\ A_i \leq_\Sigma \bigcap_{j\in J_i} D_{i,j}\ \&\ \bigcap_{j\in J_i} E_{i,j} \leq_\Sigma B_i.\end{array}$$

By rule ($\leq$) we can derive $\Gamma \vdash^\Sigma M : D_{i,j}{\to}E_{i,j}$ for all $i{\in}I, j{\in}J_i$, and so $\Gamma, x{:}D_{i,j} \vdash^\Sigma Mx : E_{i,j}$ by rule ($\to$E). From ($\leq$ L), ($\cap$I) and ($\leq$) we get $\Gamma, x{:}A_i \vdash^\Sigma Mx : B_i$ and this implies $\Gamma \vdash^\Sigma \lambda x.Mx : A_i{\to}B_i$ using rule ($\to$I). So we can conclude by ($\cap$I) and ($\leq$) that $\Gamma \vdash^\Sigma \lambda x.Mx : \psi$. The other cases are easy.

(2) ($\Rightarrow$) Let us assume that Σ does not validate axiom ($\to$-$\cap$), i.e. that there are types A, B, C such that $(A{\to}B) \cap (A{\to}C) \not\leq_\Sigma A{\to}B \cap C$. We can derive $x{:}(A{\to}B) \cap (A{\to}C) \vdash^\Sigma_{\mathcal{B}} \lambda y.xy : A{\to}B{\cap}C$ using ($\leq$), ($\to$E), ($\cap$I), and ($\to$I), but $x : A{\to}B{\cap}C$ cannot be derived from $x{:}(A{\to}B) \cap (A{\to}C)$ by Theorem 22(1). Now suppose that Σ does not validate rule (η), i.e. that there are types A, B, C, D such that $A \leq_\Sigma B$ and $C \leq_\Sigma D$ but $B{\to}C \not\leq_\Sigma A{\to}D$. We can derive $x{:}B{\to}C \vdash^\Sigma_{\mathcal{B}} \lambda y.xy : A{\to}D$ using ($\leq$), ($\to$E), and ($\to$I), but $x{:}B{\to}C \not\vdash^\Sigma_{\mathcal{B}} x : A{\to}D$ by Theorem 22(1).
If $\Omega{\in}\mathbb{C}^\Sigma$ we get $x{:}\Omega \vdash^\Sigma_\Omega \lambda y.xy : \Omega{\to}\Omega$ by axiom (Ax-Ω) and rule ($\to$I). By Theorem 22(1) we can derive $x{:}\Omega \vdash^\Sigma_\Omega x : \Omega{\to}\Omega$ iff $\Omega \leq_\Sigma \Omega{\to}\Omega$, i.e. iff Σ validates axiom (Ω-η).
If $\nu{\in}\mathbb{C}^\Sigma$ we get $\vdash^\Sigma_\nu \lambda y.xy : \nu$ by axiom (Ax-ν), but we cannot derive $x : \nu$ from the empty basis by Theorem 22(1).

($\Leftarrow$) We prove that under the given conditions on type preorders $\Gamma \vdash^\Sigma \lambda x.Mx : A$ and $x \notin FV(M)$ imply $\Gamma \vdash^\Sigma M : A$. We give the proof for $\lambda{\cap}^\Sigma_\Omega$, that one for $\lambda{\cap}^\Sigma_{\mathcal{B}}$ being similar and simpler. By Theorem 21(2) $\Gamma \vdash^\Sigma_\Omega \lambda x.Mx : A$ implies that there are I, B_i, C_i such that $\Gamma, x{:}B_i \vdash^\Sigma_\Omega Mx : C_i$ and $\bigcap_{i\in I}(B_i{\to}C_i) \leq_\Sigma A$. If for some i we get $C_i \sim_\Sigma \Omega$, then we can obtain $B_i{\to}C_i \sim_\Sigma \Omega$ by axiom (Ω-η) and rule (η). Therefore we can forget those $B_i{\to}C_i$. Otherwise $\Gamma, x{:}B_i \vdash^\Sigma_\Omega Mx : C_i$ implies by Theorem 22(2) and rule (S) that $\Gamma \vdash^\Sigma_\Omega M : D_i{\to}C_i$, and $\Gamma, x{:}B_i \vdash^\Sigma_\Omega x : D_i$, for some D_i. By Theorem 22(1) we get $B_i \leq_\Sigma D_i$, so we can derive $\Gamma \vdash^\Sigma_\Omega M : B_i{\to}C_i$ using rule ($\leq$), since $D_i{\to}C_i \leq_\Sigma B_i{\to}C_i$ by rule (η). Rule ($\cap$I) implies $\Gamma \vdash^\Sigma_\Omega M : \bigcap_{i\in I}(B_i{\to}C_i)$. So we can conclude $\Gamma \vdash^\Sigma_\Omega M : A$ using rule ($\leq$).

5 Filter λ-structures and Filter Models

In this section we shall see how the results obtained in the previous sections can be used to prove characterisation results concerning domains defined by means of intersection types, the so called filter λ-structures. In particular necessary and sufficient conditions will be given that characterize those filter λ-structures that are also models for the (restricted) λ-calculi.

Let us begin with a short discussion about how it is possible to interpret types. There are essentially *two* semantics for intersection types.

One is the *set-theoretical* semantics, originally introduced in [9], generalizing the one given by Scott for simple types. The meanings of types are subsets of the domain of discourse, arrow types are defined as *logical predicates* and intersection is set-theoretic intersection.

The second semantics, which arises in the wake of Stone Duality results (see [2], [12], [34]), views types as *compact elements* of Plotkin's λ-structures ([30]). According to this interpretation, the type Ω denotes the least element, intersections denote joins of compact elements, and arrow types allow to internalize the space of continuous endomorphisms. By duality, type preorders give rise to *filter λ-structures*, where the interpretation of λ-terms can be given through a finitary logical description.

In order to introduce filter λ-structures, let us give the appropriate notion of filter over a type preorder. This is a particular case of filter over a generic $\top$-meet semilattice (see [25]).

Definition 28 (Σ-filters) *Let Σ be a type preorder.*

(1) A Σ-filter (or a filter over $\mathbb{T}(\mathbb{C}^\Sigma)$) is a set $\Xi \subseteq \mathbb{T}(\mathbb{C}^\Sigma)$ such that
  - *(a) if $\Omega \in \mathbb{C}^\Sigma$ then $\Omega \in \Xi$;*
  - *(b) if $A \leq_\Sigma B$ and $A \in \Xi$, then $B \in \Xi$;*
  - *(c) if $A, B \in \Xi$, then $A \cap B \in \Xi$.*

(2) $\mathcal{F}^\Sigma$ the set of Σ-filters over $\mathbb{T}(\mathbb{C}^\Sigma)$;

(3) if $\Xi \subseteq \mathbb{T}(\mathbb{C}^\Sigma)$, $\uparrow \Xi$ denotes the Σ-filter generated by Ξ;

(4) a Σ-filter is principal *if it is of the shape $\uparrow \{A\}$, for some type A. We shall denote $\uparrow \{A\}$ simply by $\uparrow A$.*

It is well known that $\mathcal{F}^\Sigma$ is an ω-algebraic lattice, whose poset of compact (or finite) elements is isomorphic to the reversed poset obtained by quotienting the preorder on $\mathbb{T}(\mathbb{C}^\Sigma)$ by $\sim_\Sigma$. That means that compact elements are the filters of the form $\uparrow A$ for some type A, the top element is $\mathbb{T}(\mathbb{C}^\Sigma)$, and the bottom element is $\uparrow \Omega$ when $\Omega \in \mathbb{C}^\Sigma$ and $\emptyset$ otherwise. Moreover the join of two filters is the filter induced by their union and the meet of two filters is their intersection, i.e.:

$$\Xi \sqcup \Upsilon = \uparrow (\Xi \cup \Upsilon)$$
$$\Xi \sqcap \Upsilon = \Xi \cap \Upsilon.$$

We turn now the space of filters into an applicative structure.

Definition 29 (Application) *Application* $\_ \cdot \_ : \mathcal{F}^\Sigma \times \mathcal{F}^\Sigma \to \mathcal{F}^\Sigma$ *is defined as*

$$\Xi \cdot \Upsilon = \uparrow \{B \mid \exists A \in \Upsilon. A \to B \in \Xi\}.$$

Taking the Stone duality view point the interpretation of terms coincides with the sets of types which are deducible for them:

Definition 30 *For any* λ*-term* M *and environment* $\rho : Var \to \mathcal{F}^\Sigma$*-*$\{\emptyset\}$*,*

$$[\![M]\!]^\Sigma_\rho = \{A \mid \exists \Gamma \models \rho.\ \Gamma \vdash^\Sigma M : A\},$$

where Var *is the set of term variables and* $\Gamma \models \rho$ *if and only if* $(x : B) \in \Gamma$ *implies* $B \in \rho(x)$*.*

We call filter λ-structure *the triple* $\langle \mathcal{F}^\Sigma, \cdot, [\![\ ]\!]^\Sigma \rangle$.

By rules (Ω), ($\leq$) and ($\cap$I) the interpretations of all λ-terms are filters.

Dropping the empty set from the codomain of environments is necessary for obtaining models. First of all, notice that the empty set is a filter only if $\Omega \notin \mathbb{C}^\Sigma$. Clearly any reasonable interpretation of λ-terms must give the same meaning to the terms z and $(\lambda y.z)x$. If we would allow $\rho(z) = \Xi \neq \emptyset$ and $\rho(x) = \emptyset$ we would get $[\![z]\!]^\Sigma_\rho = \Xi$ and $[\![(\lambda y.z)x]\!]^\Sigma_\rho = \emptyset$: in fact no type is derivable for x from a basis which does not contain x when $\Omega \notin \mathbb{C}^\Sigma$. This example is obviously related to the fact that rule (idβ-exp) is admissible only when $\Omega \in \mathbb{C}^\Sigma$.

Remark 31 *In the literature (see for instance [16]) filter* λ*-structures are often referred to as triples* $\langle \mathcal{F}^\Sigma, \mathbb{F}^\Sigma, \mathbb{G}^\Sigma \rangle$ *with* $\mathbb{F}^\Sigma : \mathcal{F}^\Sigma \to [\mathcal{F}^\Sigma \to \mathcal{F}^\Sigma]$ *and* $\mathbb{G}^\Sigma : [\mathcal{F}^\Sigma \to \mathcal{F}^\Sigma] \to \mathcal{F}^\Sigma$ *defined by:*

$$\begin{aligned} \mathbb{F}^\Sigma(\Xi) &= \lambda\!\!\lambda \Upsilon \in \mathcal{F}^\Sigma. \Xi \cdot \Upsilon; \\ \mathbb{G}^\Sigma(f) &= \begin{cases} \uparrow \{A \to B \mid B \in f(\uparrow A)\} \sqcup \uparrow \nu & \textit{if } \nu \in \mathbb{C} \\ \uparrow \{A \to B \mid B \in f(\uparrow A)\} & \textit{otherwise.} \end{cases} \end{aligned}$$

Actually our definition of filter λ*-structure coincides with this last one, since application "$\cdot$" allows to recover both* $\mathbb{F}^\Sigma$ *and* $\mathbb{G}^\Sigma$*. Moreover, the interpretation* $[\![\ ]\!]^\Sigma$

coincides with the interpretation of λ-terms induced in the obvious way by $\mathbb{F}^\Sigma$ and $\mathbb{G}^\Sigma$. We prefer here the definition of filter λ-structures as triples $\langle \mathcal{F}^\Sigma, \cdot, [\![\ ]\!]^\Sigma \rangle$ since it is closer to the syntactic perspective of the previous sections.

The notion of restricted redexes introduced in Definition 23 leads us to consider correspondingly notions of restricted λ-models: first we adapt the classical definition of λ-model à la Hindley-Longo [21] to encompass the various notions of reduction, then we characterize the filter structures which induce these models. To accommodate "call-by-value" λ-calculus we allow the codomains of environments to be proper subsets of the whole domains of models.

Definition 32 (λ-models) *A* λ-model *for the (restricted) λ-calculus λ_R consists of a triple $\langle \mathcal{D}, \cdot, [\![\ ]\!]^\mathcal{D} \rangle$ such that $\mathcal{D}$ is a set, $\cdot : \mathcal{D} \times \mathcal{D} \to \mathcal{D}$, Env : $Var \to \mathcal{V}$ for some $\mathcal{V} \subseteq \mathcal{D}$ and the interpretation function $[\![\ ]\!]^\mathcal{D} : \Lambda \times Env \to \mathcal{D}$ satisfies:*

(1) $[\![x]\!]^\mathcal{D}_\rho = \rho(x)$*;*
(2) $[\![MN]\!]^\mathcal{D}_\rho = [\![M]\!]^\mathcal{D}_\rho \cdot [\![N]\!]^\mathcal{D}_\rho$*;*
(3) $[\![\lambda x.M]\!]^\mathcal{D}_\rho \cdot [\![N]\!]^\mathcal{D}_\rho = [\![M]\!]^\mathcal{D}_{\rho[x:=[\![N]\!]^\mathcal{D}_\rho]}$ *for* $\langle(\lambda x.M)N, M[x := N]\rangle \in R$*;*
(4) If $\rho(x) = \rho'(x)$ *for all* $x \in \mathrm{FV}(M)$*, then* $[\![M]\!]^\mathcal{D}_\rho = [\![M]\!]^\mathcal{D}_{\rho'}$*;*
(5) If $y \notin \mathrm{FV}(M)$*, then* $[\![\lambda x.M]\!]^\mathcal{D}_\rho = [\![\lambda y.M[x := y]]\!]^\mathcal{D}_\rho$*;*
(6) If $\forall d \in \mathcal{D}. [\![M]\!]^\mathcal{D}_{\rho[x:=d]} = [\![N]\!]^\mathcal{D}_{\rho[x:=d]}$*, then* $[\![\lambda x.M]\!]^\mathcal{D}_\rho = [\![\lambda x.N]\!]^\mathcal{D}_\rho$.

$\langle \mathcal{D}, \cdot, [\![\ ]\!]^\mathcal{D} \rangle$ *is* extensional *if moreover*

$$[\![\lambda x.Mx]\!]^\mathcal{D}_\rho = [\![M]\!]^\mathcal{D}_\rho.$$

Actually, using the Generation Lemmata, we can prove that all filter λ-structures satisfy all the points of the previous definition but the third one.

Lemma 33 *For all type preorders Σ the interpretation function $[\![\ ]\!]^\Sigma$ satisfies conditions (1), (2), (4), (5), (6) of Definition 32.*

PROOF. (1) follows immediately from Definition 30 and Theorem 22(1).
(2) Let $A \in [\![MN]\!]^\Sigma_\rho$. Then there exists $\Gamma \models \rho$ such that $\Gamma \vdash^\Sigma MN : A$. By Theorem 22(2), there exists $B \in \mathbb{T}$ such that $\Gamma \vdash^\Sigma M : B \to A$ and $\Gamma \vdash^\Sigma N : B$, hence $B \in [\![N]\!]^\Sigma_\rho$ and $B \to A \in [\![M]\!]^\Sigma_\rho$. By definition of application it follows $A \in [\![M]\!]^\Sigma_\rho \cdot [\![N]\!]^\Sigma_\rho$.
Let now $A \in [\![M]\!]^\Sigma_\rho \cdot [\![N]\!]^\Sigma_\rho$. Then there exists $B \in \mathbb{T}$ such that $B \to A \in [\![M]\!]^\Sigma_\rho$ and $B \in [\![N]\!]^\Sigma_\rho$, hence there exist two Σ-bases, Γ and Γ', such that $\Gamma \models \rho$, $\Gamma' \models \rho$, and moreover $\Gamma \vdash^\Sigma M : B \to A$, $\Gamma' \vdash^\Sigma N : B$. Consider the basis $\Gamma'' = \Gamma \uplus \Gamma'$. We have $\Gamma'' \models \rho$, $\Gamma'' \vdash^\Sigma M : B \to A$ and $\Gamma'' \vdash^\Sigma N : B$. From the last two judgments we deduce $\Gamma'' \vdash^\Sigma MN : A$, which, along with the first judgment, implies $A \in [\![MN]\!]^\Sigma_\rho$.

(4) and (5) are trivial.

(6) Suppose that the premise holds and $A \in [\![\lambda x.M]\!]^{\Sigma}_{\rho}$. Then there is $\Gamma \models \rho$ such that $\Gamma \vdash^{\Sigma} \lambda x.M : A$. Since $x \notin \mathrm{FV}(\lambda x.M)$ by rule (S) we can assume $x \notin \Gamma$. By Theorem 22(2) there exist I and $B_i, C_i \in \mathbb{T}$ such that $\Gamma \vdash^{\Sigma} \lambda x.M : B_i \to C_i$ for all $i \in I$ and $\bigcap_{i \in I}(B_i \to C_i) \leq_{\Sigma} A$. So we have, for each $i \in I$, by Theorem 22(3) $\Gamma, x : B_i \vdash^{\Sigma} M : C_i$. By the premise, we get, for each $i \in I$, $\Gamma, x : B_i \vdash^{\Sigma} N : C_i$, which implies $[\![\lambda x.M]\!]^{\Sigma}_{\rho} \subseteq [\![\lambda x.N]\!]^{\Sigma}_{\rho}$. Similarly one proves $[\![\lambda x.N]\!]^{\Sigma}_{\rho} \subseteq [\![\lambda x.M]\!]^{\Sigma}_{\rho}$.

Due to the previous lemma, a filter λ-structure is a *model* (i.e. a *filter model*) *of the λ-calculus* λ_{R} (i.e. of the calculus whose redexes are exactly the R-redexes) iff the interpretation function $[\![\ ]\!]^{\Sigma}$ equates the R-redexes with their contracta, that is it satisfies the condition of Definition 32(3):

$$[\![(\lambda x.M)N]\!]^{\Sigma}_{\rho} = [\![M]\!]^{\Sigma}_{\rho[x:=[\![N]\!]^{\Sigma}_{\rho}]} \text{ for } \langle(\lambda x.M)N, M[x := N]\rangle \in \mathsf{R}.$$

We end the section by giving the characterisations of the type preorders inducing (restricted) λ-models. These characterisations, which are generalizations of the corresponding result in [11], follow easily from Theorem 24, Corollary 26 and Lemma 33.

Theorem 34 (Characterisations of (Restricted) Filter λ-models) *The filter λ-structure* $\langle \mathcal{F}^{\Sigma}, \cdot, [\![\ ]\!]^{\Sigma}\rangle$

(1) is a model of the "call-by-value" λ-calculus iff $\Gamma \vdash^{\Sigma} \lambda x.M : A \to B$ *implies* $\Gamma, x : A \vdash^{\Sigma} M : B$ *and either* $\Omega \in \mathbb{C}^{\Sigma}$ *or* $\nu \in \mathbb{C}^{\Sigma}$*;*
(2) is a model of the $\lambda\mathbf{I}$-calculus iff $\Gamma \vdash^{\Sigma} \lambda x.M : A \to B$ *where* $x \in \mathrm{FV}(M)$ *implies* $\Gamma, x : A \vdash^{\Sigma} M : B$ *and* $\nu \notin \mathbb{C}^{\Sigma}$*;*
(3) is a model of the $\lambda\mathbf{K}\mathsf{N}$-calculus iff it is a model of the $\lambda\mathbf{I}$-calculus;
(4) is a λ-model iff it is a model of the $\lambda\mathbf{K}\mathsf{N}$-calculus and $\Omega \in \mathbb{C}^{\Sigma}$.

PROOF. Notice that $[\![(\lambda x.M)y]\!]^{\Sigma}_{\rho} \supseteq [\![M[x := y]]\!]^{\Sigma}_{\rho}$ for all x, y, M, ρ, Σ, also if $x \notin FV(M)$, since by Definition 30 $\rho(y)$ is never the empty set. This gives us for free the invariance under expansion of $\mathsf{var}\beta$-redexes.

For point (3), observe that $\Gamma \vdash^{\Sigma} \lambda x.M : A \to B$ implies $\Gamma, x : A \vdash^{\Sigma} M : B$ when $x \notin FV(M)$.

As an immediate consequence of Theorem 34(4) and Theorem 22(3), $\langle \mathcal{F}^{\Sigma}, \cdot, [\![\ ]\!]^{\Sigma}\rangle$ is a λ-model whenever Σ is a beta theory and $\Omega \in \mathbb{C}^{\Sigma}$. We can also characterize filter λ-models which are extensional using Theorem 27.

Theorem 35 (Characterisation of Extensional Filter λ-models) *The filter λ-structure* $\langle \mathcal{F}^{\Sigma}, \cdot, [\![\ ]\!]^{\Sigma}\rangle$ *is an extensional model of the calculus* λ_{R} *if it is a model of the cal-*

culus λ_{R} *and moreover* Σ *is an eta type preorder such that* $\nu \notin \mathbb{C}^{\Sigma}$ *and which validates* $\mathcal{CDV}$ *if* $\Omega \notin \mathbb{C}^{\Sigma}$*,* $\mathcal{BCD}$ *if* $\Omega \in \mathbb{C}^{\Sigma}$ *.*

Using Proposition 7(1), Lemma 21 and previous theorems we get:

- $\langle \mathcal{F}^{\nabla}, \cdot, [\![\ ]\!]^{\nabla} \rangle$ with $\nabla \in \{\mathcal{B}a, \mathcal{CDV}\}$ is a model of the $\lambda\mathbf{I}$-calculus,
- $\langle \mathcal{F}^{\nabla}, \cdot, [\![\ ]\!]^{\nabla} \rangle$ with $\nabla \in \{\mathcal{HL}, \mathcal{HR}\}$ is an extensional model of the $\lambda\mathbf{I}$-calculus,
- $\langle \mathcal{F}^{\mathcal{EHR}}, \cdot, [\![\ ]\!]^{\mathcal{EHR}} \rangle$ is a model of the "call-by-value" λ-calculus,
- $\langle \mathcal{F}^{\nabla}, \cdot, [\![\ ]\!]^{\nabla} \rangle$ with $\nabla \in \{\mathcal{AO}, \mathcal{BCD}, \mathcal{P}l, \mathcal{E}n\}$ is a λ-model,
- $\langle \mathcal{F}^{\nabla}, \cdot, [\![\ ]\!]^{\nabla} \rangle$with $\nabla \in \{\mathcal{S}c, \mathcal{P}a, \mathcal{CDZ}, \mathcal{DHM}\}$ is an extensional λ-model.

6 Conclusion

We have provided characterisation results concerning intersection type systems for the Lambda calculus, for a number of its restrictions and for their corresponding extensional versions. Some results characterize those intersection type systems for which typing invariance holds w.r.t. reduction/expansion. Other ones concerns the characterisation of (intersection) filter structures which are also models of (restricted) λ-calculi. These characterisations possess an interest *per se* in the syntactical theory of intersection types. However, in our intentions, these results aim at being a first step towards the investigation of possible semantic domains modelling exactly one computational reduction/expansion rules, together with the possibility of combining them in order to get models of complete conversions. All this, of course, exploiting the finitary representability of semantic domains offered by intersection types.

Acknowledgements The authors are grateful to Henk Barendregt and Wil Dekkers for enlightening discussions on the subject of the present paper. Moreover, they would like to thank the referees of WoLLIC'03 for careful reading a preliminary version of this paper and for their useful suggestions. The second author wishes to express his thanks to Angela Gravina.

References

[1] M. Abadi, L. Cardelli, P. Curien, and J.-J. Levy. Explicit substitutions. *J. of Functional Progr.*, 1(4):375–416, 1991.

[2] S. Abramsky. Domain theory in logical form. *Ann. Pure Appl. Logic*, 51(1-2):1–77, 1991.

[3] S. Abramsky and C.-H. L. Ong. Full abstraction in the lazy lambda calculus. *Inform. and Comput.*, 105(2):159–267, 1993.

[4] F. Alessi, F. Barbanera, and M. Dezani-Ciancaglini. Intersection types and

computational rules. In *WoLLIC'03*, number 84 in El. Notes in Theoret. Comput. Sci. Elsevier, 2003.

[5] F. Alessi and M. Dezani-Ciancaglini. Filter models and easy terms. In *ICTCS'01*, volume 2202 of *LNCS*, pages 17–37. Springer-Verlag, 2001.

[6] F. Alessi, M. Dezani-Ciancaglini, and S. Lusin. Intersection types and domain operators. *Theoret. Comput. Sci.*, 2003. to appear.

[7] F. Alessi and S. Lusin. Simple easy terms. In *ITRS'02*, volume 70 of *ENTCS*. Elsevier, 2002.

[8] H. Barendregt. *The Lambda Calculus: its Syntax and Semantics*. North-Holland, Amsterdam, revised edition, 1984.

[9] 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.

[10] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Functional characters of solvable terms. *Z. Math. Logik Grundlag. Math.*, 27(1):45–58, 1981.

[11] M. Coppo, M. Dezani-Ciancaglini, and M. Zacchi. Type theories, normal forms, and D_∞-lambda-models. *Inform. and Comput.*, 72(2):85–116, 1987.

[12] M. Coppo, F. Honsell, M. Dezani-Ciancaglini, and G. Longo. Extended type structures and filter lambda models. In *Logic colloquium '82*, pages 241–262. North-Holland, 1984.

[13] H. Curry and R. Feys. *Combinatory Logic*, volume I of *Studies in Logic and the Foundations of Mathematics*. North-Holland, 1958.

[14] M. Dezani-Ciancaglini and S. Ghilezan. Two behavioural lambda models. In *TYPES'02*, volume 2646 of *LNCS*, pages 127–147. Springer-Verlag, 2003.

[15] M. Dezani-Ciancaglini, S. Ghilezan, and S. Likavec. Behavioural inverse limit models. *Theoret. Comput. Sci.*, 2003. to appear.

[16] M. Dezani-Ciancaglini, F. Honsell, and F. Alessi. A complete characterization of complete intersection-type preorders. *ACM TOCL*, 4(1):120–147, 2003.

[17] M. Dezani-Ciancaglini, F. Honsell, and Y. Motohama. Compositional characterization of λ-terms using intersection types. *Theoret. Comput. Sci.*, 2003. to appear. An extended abstract appeared in *MFCS'00*, volume 1893 of *LNCS*, pages 304–313. Springer-Verlag, 2000.

[18] L. Egidi, F. Honsell, and S. Ronchi Della Rocca. Operational, denotational and logical descriptions: a case study. *Fund. Inform.*, 16(2):149–169, 1992.

[19] E. Engeler. Algebras and combinators. *Algebra Universalis*, 13(3):389–392, 1981.

[20] G. Gierz, K. Hoffmann, K. Keimel, J. Mislove, and D. Scott. *A Compendium of Continuous Lattices*. Springer-Verlag, 1980.

[21] R. Hindley and G. Longo. Lambda-calculus models and extensionality. *Z. Math. Logik Grundlag. Math.*, 26(4):289–310, 1980.

[22] F. Honsell and M. Lenisa. Some results on the full abstraction problem for restricted lambda calculi. In *MFCS'93*, volume 711 of *LNCS*, pages 84–104. Springer-Verlag, 1993.

[23] F. Honsell and M. Lenisa. Semantical analysis of perpetual strategies in λ-calculus. *Theoret. Comput. Sci.*, 212(1-2):183–209, 1999.

[24] F. Honsell and S. Ronchi Della Rocca. An approximation theorem for topological lambda models and the topological incompleteness of lambda calculus. *J. Comput. System Sci.*, 45(1):49–75, 1992.

[25] P. T. Johnstone. *Stone spaces*. Cambridge University Press, Cambridge, 1986. Reprint of the 1982 edition.

[26] F. Kamareddine, A. Rios, and J. Wells. Calculi of generalized beta-reduction and explicit substitution: Type free and simply typed versions. *J. of Functional and Logic Progr.*, 5:1–44, 1998.

[27] P. Lescanne. From $\lambda\sigma$ to $\lambda\upsilon$: a journey through calculi of explicit substitutions. In *POPL'94*, pages 60–69. ACM Press, 1994.

[28] D. Park. The Y-combinator in Scott's λ-calculus models (revised version). Theory of Computation Report 13, Department of Computer Science, University of Warick, 1976.

[29] G. D. Plotkin. Call-by-name, call-by-value and the λ-calculus. *Theoret. Comput. Sci.*, 1(2):125–159, 1975.

[30] G. D. Plotkin. Set-theoretical and other elementary models of the λ-calculus. *Theoret. Comput. Sci.*, 121(1-2):351–409, 1993.

[31] D. Scott. Continuous lattices. In *Toposes, algebraic geometry and logic*, volume 274 of *LNM*, pages 97–136. Springer-Verlag, 1972.

[32] D. Scott. Open problem. In *Lambda Calculus and Computer Science Theory*, volume 37 of *LNCS*, page 369. Springer-Verlag, 1975.

[33] S. van Bakel. Complete restrictions of the intersection type discipline. *Theoret. Comput. Sci.*, 102(1):135–163, 1992.

[34] S. Vickers. *Topology via logic*. Cambridge University Press, Cambridge, 1989.