Electronic Notes in Theoretical Computer Science 70 No. 1 (2002)
URL: http://www.elsevier.nl/locate/entcs/volume70.html 19 pages

Intersection Typed λ-calculus

Simona Ronchi Della Rocca [1,2,3,4]

Dipartimento di Informatica
Università di Torino
10149 – Torino, Italy

Abstract

The aim of this paper is to discuss the design of an explicitly typed λ-calculus corresponding to the Intersection Type Assignment System (IT), which assigns intersection types to the untyped λ-calculus. Two different proposals are given. The logical foundation of all of them is the Intersection Logic IL.

1 Introduction

The intersection type assignment system (IT) is a set of rules for assigning types to terms of the untyped λ-calculus. The types of IT are formulae of the implicational and conjuntive fragment of the predicate logic, which will be denoted (with an abuse of notation) by LJ (Fig. 2). Formulae of LJ are generated by the grammar: $\sigma ::= V \mid (\sigma \to \sigma) \mid (\sigma \wedge \sigma)$ where V is a denumerable set of constants. The system is in Fig. 1.

[1] partially supported by MIUR-cofin-2000 "Linear Logic and Beyond"
[2] partially supported by EC-TMR project "Linear Logic"
[3] partially supported by IST-2001-33477 DART Project
[4] Email:ronchi@di.unito.it

$$(Ax_{IT})\frac{x:\sigma \in \Gamma}{\Gamma \vdash_{IT} x:\sigma} \qquad (\wedge I_{IT})\frac{\Gamma \vdash_{IT} M:\sigma \quad \Gamma \vdash_{IT} M:\tau}{\Gamma \vdash_{IT} M:\sigma \wedge \tau}$$

$$(\wedge E^l_{IT})\frac{\Gamma \vdash_{IT} M:\sigma \wedge \tau}{\Gamma \vdash_{IT} M:\sigma} \qquad (\wedge E^r_{IT})\frac{\Gamma \vdash_{IT} M:\sigma \wedge \tau}{\Gamma \vdash_{IT} M:\tau}$$

$$(\to I_{IT})\frac{\Gamma \cup \{x:\sigma\} \vdash_{IT} M:\tau}{\Gamma \vdash_{IT} \lambda x.M:\sigma \to \tau} \qquad (\to E_{IT})\frac{\Gamma \vdash_{IT} M:\sigma \to \tau \quad \Gamma \vdash_{IT} N:\sigma}{\Gamma \vdash_{IT} MN:\tau}$$

Fig. 1. The Intersection Type assignment system. Types are formulae of LJ. Contexts are sets $\{x_1:\sigma_1,\ldots,x_n:\sigma_n\}$, where $i \neq j$ implies $x_i \not\equiv x_j$.

IT was introduced in the early Eighties by Mariangiola Dezani and Mario Coppo [4], to extend the set of λ-terms typable by the Curry type assignment system. The system has nice syntactical features. In particular, it has been proved that IT assigns types to all and only the strong normalizing terms [7].

The aim of this paper is to discuss the design of an explicitly typed λ-calculus corresponding to IT.

The design of a typed version of IT has been an open problem for a long time. Indeed, from a foundational point of view, terms of a typed language encode proofs of a logical system. For example, the simply typed λ-calculus and the second order λ-calculus encode the proofs of the implicational fragment of the propositional and the second order intuitionistic logic respectively; their corresponding type assignments were defined only in a second step.

On the other hand, IT was directly defined as a type assignment system. The problem of finding a logical foundation for it as been first proposed by Hindley [5]: the main issue in order to solve it is to find the logical form of the conjunction introduction:

$$(\wedge I_{IT})\frac{\Gamma \vdash_{IT} M : \sigma \quad \Gamma \vdash_{IT} M : \tau}{\Gamma \vdash_{IT} M : \sigma \wedge \tau}.$$

In accordance with the "Curry-Howard isomorphism" between λ-terms and proofs, the rule says that an intersection type $\sigma \wedge \tau$ can be built from the two components σ and τ only in case they can be proved by two "isomorphic" proofs, in accordance with a notion of isomorphism that relates proofs, encoded by the same term. This corresponds to a meta-theoretical restriction on the introduction of the conjunction, and for this reason LJ is not the logic supporting a foundation for IT.

A solution to the problem is to find a logic that internalizes the meta restrictions on $(\wedge I_{IT})$. Some solutions appeared recently. [2,3] introduce the notion of *hyperformulae*, i.e., sequences of formulae of LJ proved in parallel. [8] introduces Intersection Logic (IL) where formulae are trees of formulae of LJ. Moreover, [2] introduces also a typed λ-calculus, whose types are intersection types.

Also, an independent proposal about an explicitly typed λ-calculus whose types are, in fact, a superset of intersection types, is in [11] and [12]. Differently from the approaches of [2] and of the present paper, this language has not been designed out of a logical system.

This paper introduces a couple of further proposal of intersection typed λ-calculi. The starting point is IL [8]. The first proposal yields Λ_{IT}, obtained as decoration of proofs of IL. Λ_{IT} is quite similar to the language in [11][12], when types are restricted to $\rightarrow$ and $\wedge$. However, its logical foundation allows for a deep analysis of its operational semantics, which is intrinsecally parallel. A variant of Λ_{IT}, Λ'_{IT} is studied too, where a more difficult formation rule of terms allows for a completely sequential reduction system. Moreover a further

$$(A_{LJ})\frac{\sigma \in \Gamma}{\Gamma \vdash_{LJ} \sigma} \qquad (\wedge I_{LJ})\frac{\Gamma \vdash_{LJ} \sigma \quad \Gamma \vdash_{LJ} \tau}{\Gamma \vdash_{LJ} \sigma \wedge \tau}$$

$$(\wedge E^{l}_{LJ})\frac{\Gamma \vdash_{LJ} \sigma \wedge \tau}{\Gamma \vdash_{LJ} \sigma} \qquad (\wedge E^{r}_{LJ})\frac{\Gamma \vdash_{LJ} \sigma \wedge \tau}{\Gamma \vdash_{LJ} \tau}$$

$$(\rightarrow I_{LJ})\frac{\Gamma \cup \{\sigma\} \vdash_{LJ} \tau}{\Gamma \vdash_{LJ} \sigma \rightarrow \tau} \quad (\rightarrow E_{LJ})\frac{\Gamma \vdash_{LJ} \sigma \rightarrow \tau \quad \Gamma \vdash_{LJ} \sigma}{\Gamma \vdash_{LJ} \tau}$$

Fig. 2. Implicative and Conjunctive Fragment of Intuitionistic Logic (LJ). Γ is a set of formulae.

language, Λ^2_{IT}, is studied. Λ^2_{IT} is a simplification of Λ_{IT}, that partially gets rid of unnecessary differentiations among the derivations of IL due to the commutative and associative properties of the connective $\wedge$. This language is similar, but not identical, to that one proposed in [2].

The opinion of the author is that Λ^2_{IT} cannot be further simplified, without losing its logical foundation. Some justifications to this claim are given in Section 4.

The paper is organized as follows. In Section 2 the system IL and its properties are recalled. In Section 3 and 4 the two languages Λ_{IT} and Λ^2_{IT} are defined and their properties are given. In Section 5 a comparison with related work is made.

2 Intersection Logic

This section is a survey of [8]. It introduces Intersection Logic (IL) and formalizes its correspondence with the intersection type assignment.

The formulae of IL are binary trees, with leaves labelled by formulae of LJ. The relation between IL and LJ can be informally described as follows: a derivation Π of IL groups a set, say $LJ(\Pi)$, of derivations of LJ. Every derivation $\Pi' \in LJ(\Pi)$ can be obtained by taking the leaf of a given path in every tree of Π. In particular, the elements of $LJ(\Pi)$ share both the number of instances and the application order of the rules that introduce and eliminate $\rightarrow$.

IL is obtained from the pre-Intersection Logic (pIL), recalled in Fig. 3, by an equivalence relation.

Some necessary notions to understand the definition of pIL are in order.

Definition 2.1 i) A *kit* is a binary tree in the language generated by the following grammar:

$$H, K ::= \sigma \mid [K, K]$$

The leaves of any kit, which we call also *atoms*, are *formulae of LJ*.

ii) A *path* is a string built over the set $\{l, r\}$; p, q, possibly indexed, denote paths, and ϵ denotes the empty path. The subtree of a kit H at the path p

$$(A_{pIL})\frac{\forall K \in \Gamma . K \simeq H \quad H \in \Gamma}{\Gamma \vdash_{pIL} H}$$

$$(P_{pIL})\frac{\Gamma \vdash_{pIL} K \quad s \in \{l, r\} \quad ps \in P_T(K)}{\Gamma \backslash^{ps} \vdash_{pIL} K \backslash^{ps}}$$

$$(\to E_{pIL})\frac{\Gamma \vdash_{pIL} H \to K \quad \Gamma \vdash_{pIL} H}{\Gamma \vdash_{pIL} K} \qquad (\to I_{pIL})\frac{\Gamma \cup \{H\} \vdash_{pIL} K}{\Gamma \vdash_{pIL} H \to K}$$

$$(\wedge E^l_{pIL})\frac{\Gamma \vdash_{pIL} K[p := \sigma \wedge \tau]}{\Gamma \vdash_{pIL} K[p := \sigma]} \qquad (\wedge E^r_{pIL})\frac{\Gamma \vdash_{pIL} K[p := \sigma \wedge \tau]}{\Gamma \vdash_{pIL} K[p := \tau]}$$

$$(\wedge I_{pIL})\frac{\{H_1[p := [\sigma_1, \sigma_1]], ..., H_n[p := [\sigma_n, \sigma_n]]\} \vdash_{pIL} K[p := [\sigma, \tau]]}{\{H_1[p := \sigma_1], ..., H_n[p := \sigma_n]\} \vdash_{pIL} K[p := \sigma \wedge \tau]}$$

Fig. 3. The *pre Intersection Logic*. The context Γ is a set of kits. In rule (P), the notation $\Gamma\backslash^{ps}$ stands for the distribution of the pruning to the elements of Γ.

is inductively defined as follows:

$$H^{\epsilon} = H, [H_1, H_2]^{lp} = H_1^p, [H_1, H_2]^{rp} = H_2^p$$

Otherwise, H^p is undefined.

A path p is *defined in* H if H^p is defined, and it is *terminal in* H if H^p is an atom; the set of terminal paths of a kit H is denoted by $P_T(H)$. $H[p := K]$ denotes the kit resulting from the substitution of K for H^p in H. The path obtained by concatenation of the two paths p and q is simply denoted by pq.

iii) $H \simeq K$ if $P_T(H) = P_T(K)$, i.e., H and K are two trees with exactly the same structure, but which may differ only on the formulae at their leaves.

iv) Let $H \simeq K$. $H \to K$ denotes the kit such that, for every $p \in P_T(H) = P_T(K)$, $(H \to K)^p$ is the atom $H^p \to K^p$.

v) Let $s \in \{l, r\}$, and let ps be a path, defined in H. The *pruning* of H at path ps, is defined as follows:

$H\backslash^{ps} = H[p := H^{ps}]$

vi) $\equiv$ is the syntactical identity of both atoms, kits and paths.

The natural deduction system $\vdash_{pIL}$ is given in Fig. 3.

Notation 2.1 *Formulae of LJ are ranged over by σ, τ. Kits are ranged over by H, K. Terms of (typed) λ-calculus are ranged over by M, N, P, Q. $\equiv$ denotes the syntactical identity of both formulae, terms and kits.*

The judgments of $\vdash_{pIL}$ enjoy an invariant, i.e., $\{H_1, \ldots, H_n\} \vdash_{pIL} K$ implies $H_i \simeq K$ $(1 \leq i \leq n)$. Let us call informally "corresponding" two leaves at the same path of two different tree. As will be formally stated later, if $\{H_1, ..., H_n\} \vdash_{pIL} K$, and $\sigma_1, ..., \sigma_n, \sigma$ are corresponding nodes in $H_1, ..., H_n, H$

respectively, then $\{\sigma_1, ..., \sigma_n\} \vdash_{LJ} \sigma$. Note the condition on the rule $(\wedge I_{pIL})$: intersection between two leaves can be introduced just in case the corresponding leaves in every tree of the context are labelled by the same atom. This condition realizes the internalization of the meta-theoretical restriction on the rule $(\wedge I_{IT})$, discussed in the introduction. The structural rule (P_{pIL}) allows for a manipulation of the structure of the trees involved in the derivation, and is essential for the normalization procedure. The Intersection Logic (IL) is obtained from pIL by eliminating unnecessary differentiations among the deductions of $\vdash_{pIL}$. In particular, deductions in IL are the deductions of $\vdash_{pIL}$ up to the order of applications of the rules dealing with the intersection, when applied on different paths.

Definition 2.2 [Intersection Logic] *Intersection Logic*, abbreviated as IL, is the set ${}^{\vdash_{pIL}}/_{\sim}$ of all the deductions of $\vdash_{pIL}$, quotiented by the congruence $\sim$, which is the reflexive and transitive closure of the following rules:

$$(\wedge E^{s'}_{pIL})\frac{(\wedge E^{s}_{pIL})\dfrac{\Gamma \vdash_{pIL} K[p := \sigma_l \wedge \sigma_r][q := \tau_l \wedge \tau_r]}{\Gamma \vdash_{pIL} K[p := \sigma_s][q := \tau_l \wedge \tau_r]}}{\Gamma \vdash_{pIL} K[p := \sigma_s][q := \tau_{s'}]} \sim$$

$$(\wedge E^{s}_{pIL})\frac{(\wedge E^{s'}_{pIL})\dfrac{\Gamma \vdash_{pIL} K[p := \sigma_l \wedge \sigma_r][q := \tau_l \wedge \tau_r]}{\Gamma \vdash_{pIL} K[p := \sigma_l \wedge \sigma_r][q := \tau_{s'}]}}{\Gamma \vdash_{pIL} K[p := \sigma_s][q := \tau_{s'}]}$$

$$(\wedge E^{s}_{pIL})\frac{(\wedge I_{pIL})\dfrac{\Gamma \vdash_{pIL} K[p := \sigma_l \wedge \sigma_r][q := [\tau_l, \tau_r]]}{\Gamma\backslash^{qs'} \vdash_{pIL} K[p := \sigma_l \wedge \sigma_r][q := \tau_l \wedge \tau_r]}}{\Gamma\backslash^{qs'} \vdash_{pIL} K[p := \sigma_s][q := \tau_l \wedge \tau_r]} \sim$$

$$(\wedge I_{pIL})\frac{(\wedge E^{s}_{pIL})\dfrac{\Gamma \vdash_{pIL} K[p := \sigma_l \wedge \sigma_r][q := [\tau_l, \tau_r]]}{\Gamma \vdash_{pIL} K[p := \sigma_s][q := \tau_l, \tau_r]}}{\Gamma\backslash^{qs'} \vdash_{pIL} K[p := \sigma_s][q := \tau_l \wedge \tau_r]}$$

being $s, s' \in \{l, r\}$, and p, q two different paths.

Let π be a derivation in IL, i.e., an equivalence class of derivations in pIL. We will denote by $\Pi \in \pi$ the fact that the derivation Π in pIL belongs to this class.

In order to reason about the relationship between LJ, IT and IL, a "decoration" of all three systems is used. The decoration is developed in analogy to the, so called, "Curry-Howard isomorphism". Every deduction Π is associated to a λ-term that records some structural properties of Π. Note that the decoration is not standard, since the λ-term associated to Π is *untyped*, and does not encode *the whole* structure of Π. On the contrary, it just records the order of the occurrences of the rules introducing and eliminating $\to$, inside Π. Observe that decorations of derivations in pIL are invariant under the equivalence introduced in Definition 2.2; so they can be seen as decorations of derivations of IL.

Definition 2.3 Let $* \in \{LJ, pIL\}$, and $A, B, C, ...$ be meta-variables for denoting either atoms or kits and let $\Delta \subseteq \{A, B, C, ...\}$.

i) Every Π proving $\Delta \vdash_* A$ can be decorated by a λ-term $T_{dom(\Delta^*)}(\Pi)$, where Δ^* is a decoration of Δ, and, if $\Delta^* \equiv \{x_1 : A_1, \ldots, x_n : A_n\}$, then $dom(\Delta^*)$ is the set $\{x_1, \ldots, x_n\}$. The decoration of $\vdash_*$ is denoted by $\vdash_*^+$ and is inductively defined as:

- $\Pi : (A_*)\dfrac{A \in \Delta}{\Delta \vdash_* A} \Rightarrow (A_*^+)\dfrac{x : A \in \Delta^*}{\Delta^* \vdash_*^+ x : A}$ where, if $\Delta = \{A_1, ..., A_n\}$ then $\Delta^* = \{x_1 : A_1, ..., x_n : A_n\}$, where $i \neq j$ implies $x_i \not\equiv x_j$ and $T_x(\Pi) \equiv x$;
- $\Pi : (\to I_*)\dfrac{\Pi_1 : \Delta \cup \{A\} \vdash_* B}{\Delta \vdash_* A \to B} \Rightarrow$
$$(\to I_*^+)\frac{\Delta^* \cup \{x : A\} \vdash_*^+ T_{dom(\Delta^*)\cup\{x\}}(\Pi_1) : B \quad x \notin dom(\Delta^*)}{\Delta^* \vdash_*^+ \lambda x.T_{dom(\Delta^*)\cup\{x\}}(\Pi_1) : A \to B}$$
and $T_{dom(\Delta^*)}(\Pi) \equiv \lambda x.T_{dom(\Delta^*)\cup\{x\}}(\Pi_1)$;
- $\Pi : (\to E_*)\dfrac{\Pi_1 : \Delta \vdash_* A \to B \quad \Pi_2 : \Delta \vdash_* A}{\Delta \vdash_* B} \Rightarrow$
$$(\to E_*^+)\frac{\Delta^* \vdash_*^+ T_{dom(\Delta^*)}(\Pi_1) : A \to B \quad \Delta^* \vdash_*^+ T_{dom(\Delta^*)}(\Pi_2) : A}{\Delta^* \vdash_*^+ T_{dom(\Delta^*)}(\Pi_1)T_{dom(\Delta^*)}(\Pi_2) : B}$$
and $T_{dom(\Delta^*)}(\Pi) \equiv T_{dom(\Delta^*)}(\Pi_1)T_{dom(\Delta^*)}(\Pi_2)$;
- $\Pi : (P_{pIL})\dfrac{\Pi_1 : \{K_1, \ldots, K_n\} \vdash_{pIL} H}{\{K_1\backslash^{ps}, \ldots, K_n\backslash^{ps}\} \vdash_{pIL} H\backslash^{ps}} \Rightarrow$
$$(P_{pIL}^+)\frac{\{x_1 : K_1, \ldots, x_n : K_n\} \vdash_{pIL}^+ T_{\{x_1,\ldots,x_n\}}(\Pi_1) : H}{\{x_1 : K_1\backslash^{ps}, \ldots, x_n : K_n\backslash^{ps}\} \vdash_{pIL}^+ T_{\{x_1,\ldots,x_n\}}(\Pi) : H\backslash^{ps}}$$
and $T_{\{x_1,\ldots,x_n\}}(\Pi) \equiv T_{\{x_1,\ldots,x_n\}}(\Pi_1)$;
- $\Pi : (\wedge E_*^s)\dfrac{\Pi_1 : \Delta \vdash_* A_l \wedge A_r}{\Delta \vdash_* A_s} \Rightarrow (\wedge(E_*^s)^+)\dfrac{\Delta^* \vdash_*^+ T_{dom(\Delta^*)}(\Pi_1) : A_l \wedge A_r}{\Delta^* \vdash_*^+ T_{dom(\Delta^*)}(\Pi_1) : A_s}$
and $T_{dom(\Delta^*)}(\Pi) \equiv T_{dom(\Delta^*)}(\Pi_1)$.
- $\Pi : (\wedge I_{pIL})\dfrac{\Pi_1 : \Delta \equiv \{H_1[p := [\sigma_1, \sigma_1]], ..., H_n[p := [\sigma_n, \sigma_n]]\} \vdash_{pIL} K[p := [\sigma, \tau]]}{\{H_1[p := \sigma_1], ..., H_n[p := \sigma_n]\} \vdash_{pIL} K[p := \sigma \wedge \tau]} \Rightarrow$
$$(\wedge I_{pIL}^+)\frac{\Delta^* \equiv \{x_i : H_i[p := [\sigma_i, \sigma_i] \mid 1 \leq i \leq n\} \vdash_{pIL}^+ T_{dom(\Delta^*)}(\Pi_1) : K[p := [\sigma, \tau]]}{\{x_1 : H_1[p := \sigma_1], ..., x_n : H_n[p := \sigma_n]\} \vdash_{pIL} T_{dom(\Delta^*)}(\Pi) : K[p := \sigma \wedge \tau]}$$
and $T_{dom(\Delta^*)}(\Pi) \equiv T_{dom(\Delta^*)}(\Pi_1)$.
- $\Pi : (\wedge I_{LJ})\dfrac{\Pi_1 : \Delta \vdash_{LJ} \sigma \quad \Pi_2 : \Delta \vdash_{LJ} \tau}{\Delta \vdash_{LJ} \sigma \wedge \tau} \Rightarrow$
$$(\wedge I_{LJ}^+)\frac{\begin{array}{c}\Delta^* \vdash_{LJ}^+ T_{dom(\Delta^*)}(\Pi_1) : \sigma \\ \Delta^* \vdash_{LJ}^+ T_{dom(\Delta^*)}(\Pi_2) : \tau \\ T_{dom(\Delta^*)}(\Pi_1) =_\alpha T_{dom(\Delta^*)}(\Pi_2)\end{array}}{\Delta \vdash_{LJ}^+ T_{dom(\Delta^*)}(\Pi) : \sigma \wedge \tau}$$
and $T_{dom(\Delta^*)}(\Pi) \equiv T_{dom(\Delta^*)}(\Pi_1) \equiv T_{dom(\Delta^*)}(\Pi_2)$.

ii) $U(\Pi) = \{T_{dom(\Delta^*)}(\Pi) \mid \Pi : \Delta \vdash_* A$ and Δ^* is a decoration of $\Delta\}$

Notice that, by construction, $M, N \in U(\Pi)$ implies M and N can differ each other by renaming of both free and bound variables. We will call $U(\Pi)$ the *form* of Π.

The next theorem shows that each derivation of $\vdash_{IL}$ corresponds to a set of derivations in LJ with the same form.

Theorem 2.4 (From IL to LJ) *Let $\Pi : \{K_1,\ldots,K_n\} \vdash_{IL} H$. For all paths p terminal in H, $\Pi^p : \{K_1^p,\ldots,K_n^p\} \vdash_{LJ} H^p$, and $U(\Pi^p) = U(\Pi)$.*

Definition 2.5 A judgment $\{K_1,\ldots,K_n\} \vdash_{IL} H$ is *proper* if and only if H and every K_i are atoms ($1 \leq i \leq n$).

Theorem 2.6 (IL and IT) *Let π be a proper derivation. $\pi : \{\sigma_1,\ldots,\sigma_n\} \vdash_{IL} \tau$ if and only if $\{x_1 : \sigma_1,\ldots,x_n : \sigma_n\} \vdash_{IT} T_{\{x_1\ldots x_n\}}(\pi) : \tau$.*

On the system IL some reduction rules can be defined.

Definition 2.7 • A $\wedge$-*IL-rewriting step* on Π is:

$$(\wedge E_s)\dfrac{(\wedge I)\dfrac{\{H_1[p:=[\sigma_1,\sigma_1]],\ldots,H_n[p:=[\sigma_n,\sigma_n]]\} \vdash_{pIL} K[p:=[\sigma_l,\sigma_r]]}{\{H_1[p:=\sigma_1],\ldots,H_n[p:=\sigma_n]\} \vdash_{pIL} K[p:=\sigma_l \wedge \sigma_r]}}{\{H_1[p:=\sigma_1],\ldots,H_n[p:=\sigma_n]\} \vdash_{pIL} K[p:=\sigma_s]} \rightsquigarrow_\wedge$$

$$(P)\dfrac{\{H_1[p:=[\sigma_1,\sigma_1]],\ldots,H_n[p:=[\sigma_n,\sigma_n]]\} \vdash_{pIL} K[p:=[\sigma_l,\sigma_r]]}{\{(H_1[p:=\sigma_1]),\ldots,(H_n[p:=\sigma_n])\} \vdash_{pIL} K[p:=\sigma_s]}$$

• A $\rightarrow$-*IL-rewriting* step on Π is:

$$(\rightarrow E)\dfrac{(\rightarrow I)\dfrac{\Pi_0 : \Gamma, H \vdash_{pIL} K}{\Gamma \vdash_{pIL} H \rightarrow K} \qquad \Pi_1 \ : \ \Gamma \ \vdash_{pIL} H}{\Gamma \vdash_{pIL} K} \rightsquigarrow_\rightarrow S(\Pi_1,\Pi_0) : \Gamma \vdash K$$

where $S(\Pi,\Pi')$ is the deductive structure obtained from Π_0 by

- replacing every axiom of the shape $\Gamma' \cup \{H\} \vdash_{pIL} H$ by the derivation $\Pi_1 : \Gamma \cup \{H\} \vdash_{pIL} H$ (it is always possible since $\Gamma' \subseteq \Gamma$);
- changing the contexts according with the previous substitution.

In [8] it has been proved that rule (P) is redundant, i.e., each derivation in IL can be transformed into an equivalent one which does not have instances of this rule. Using this property, it can be proved that IL is strongly normalizing, with respect to the rewriting rules given in the previous definition (the proof has been given in [8]).

Example 2.8 Let σ and τ denote respectively the formula $(\alpha \rightarrow \beta)\wedge(\mu \rightarrow \nu)$ and $\alpha \wedge \mu$, and let $\Gamma = \{x : \sigma, y : \tau\}$. Let also the following deduction be given:

$$(\rightarrow E_{IT})\dfrac{(\wedge E^l_{IT})\dfrac{(A_{IT})\dfrac{}{\Gamma \vdash_{IT} x : \sigma}}{\Gamma \vdash_{IT} x : \alpha \rightarrow \beta} \quad (\wedge E^l_{IT})\dfrac{(A_{IT})\dfrac{}{\Gamma \vdash_{IT} y : \tau}}{\Gamma \vdash_{IT} y : \alpha}}{\Gamma \vdash_{IT} xy : \beta}$$

A quite similar deduction proves $\Gamma \vdash_{IT} xy : \nu$, so by rule $(\wedge I_{IT})$, $\Gamma \vdash_{IT} xy : \beta \wedge \nu$. The corresponding derivation in IL is the following, where $\Delta = \{[\sigma, \sigma], [\tau, \tau]\}$:

$$
(\wedge I_{IL})\dfrac{(\to E_{IL})\dfrac{(\wedge E^r_{IL})\dfrac{(\wedge E^l_{IL})\dfrac{(A_{IL})\dfrac{}{\Delta \vdash_{IL} [\sigma, \sigma]}}{\Delta \vdash_{IL} [\alpha \to \beta, \sigma]}}{\Delta \vdash_{IL} [\alpha \to \beta, \mu \to \nu]} \quad (\wedge E^r_{IL})\dfrac{(\wedge E^l_{IL})\dfrac{(A_{IL})\dfrac{}{\Delta \vdash_{IL} [\tau, \tau]}}{\Delta \vdash_{IL} [\alpha, \tau]}}{\Delta \vdash_{IL} [\alpha, \mu]}}{\Delta \vdash_{IL} [\beta, \nu]}}{\{\sigma, \tau\} \vdash_{IL} \beta \wedge \nu}
$$

3 Intersection Typed λ-calculus I

The first proposal of an intersection typed λ-calculus (Λ_{IT}) is obtained by completely decorating the derivations of IL with terms.

It will turn out that Λ_{IT} is a *proper* sub-set of the language $\mathcal{L}$, generated by the grammar:

$$
\begin{aligned}
M, N ::= x^\sigma \mid (M^{\sigma \to \rho} N^\tau)^\rho \mid (\lambda x^\sigma . M^\tau)^{\sigma \to \tau} \mid (M^\sigma \wedge N^\tau)^{\sigma \wedge \tau} \mid \\
(D_l(M^{\sigma \wedge \tau}))^\sigma \mid (D_r(M^{\sigma \wedge \tau}))^\tau
\end{aligned}
$$

where ρ, σ and τ are formulae of LJ. In particular, the terms with form $(M^\sigma \wedge N^\tau)^{\sigma \wedge \tau}$ represent a pair. We have not used the more standard notation $\langle M^\sigma, N^\tau \rangle^{\sigma \wedge \tau}$ to highlight that the pairs of Λ_{IT} are not at all standard. Indeed, we shall prove that the erasure of the types of both M^σ and N^τ in $(M^\sigma \wedge N^\tau)^{\sigma \wedge \tau}$ necessarily yields the same *untyped* λ-term.

The next definition describes the algorithm $\mathcal{D}$ decorating IL. $\mathcal{D}$ takes as input a derivation in IL and gives as output its decoration. The decorated system is called $\vdash_\lambda$. The algorithm is quite standard, but:

- every leaf of a kit is decorated by a term, in such a way that decorations of different leaves of the same kit have identical erasures,

- in the rule $(\to E)$ a control is made on variable names, in order to assure an "hygiene" condition which is useful for defining the operation semantics.

In the following, we will use the notations $FV(M)$, $BV(M)$ and $FBV(M)$ for denoting respectively the set of free, bound and both free and bound variables variables of a (typed or untyped) term M.

Definition 3.1 i) A *typed kit* $\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi)$ is a tree whose leaves are labelled by λ-terms, typed with formulae of LJ, and whose set of free variables is $\{x_1, \ldots, x_n\}$.

ii) Let σ be a formula of LJ: x^σ denotes a typed variable whose type is σ and whose erasure is x. Let H be a kit, and x a variable. $t(H, x)$ is a tree such that $t(H, x) \simeq H$, and $(t(H, x))^p \equiv x^{H^p}$, for every p terminal in H.

iii) The algorithm $\mathcal{D}$ decorating IL and the erasure function $E : \Lambda_{IT} \to \Lambda$ are mutually defined in the following.

· For every Π and π, such that $\Pi \in \pi$ and $\pi : \{K_1, \ldots, K_n\} \vdash_{IL} H$, $\mathcal{D}$ maps Π to the judgment $\{t(K_1, x_1), \ldots, t(K_n, x_n)\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) : H$, where $\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) \simeq H$. If Γ is the context $\{K_1, ..., K_n\}$ then $t(\Gamma)$ denotes the context $\{t(K_1, x_1), ..., t(K_n, x_n)\}$. $\mathcal{D}$ is defined by cases on the last rule of Π, as follows:

$$\mathcal{D}(\Pi : (A_{pIL})\frac{}{\Gamma \cup \{H\} \vdash_{pIL} H}) = (A)\frac{}{t(\Gamma) \cup \{t(H, x)\} \vdash_\lambda t(H, x) : H}$$

with $\mathcal{T}_x(\Pi) \equiv t(H, x)$;

$$\mathcal{D}(\Pi : (P_{pIL})\frac{\Pi_1 : \{K_1, \ldots, K_n\} \vdash_{pIL} K \quad s \in \{l, r\} \quad ps \in P_T(K)}{\{K_1\backslash^{ps}, \ldots, K_n\backslash^{ps}\} \vdash_{pIL} K\backslash^{ps}}) =$$

$$(P)\frac{\{t(K_1, x_1), \ldots, t(K_n, x_n)\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1) : K \quad s \in \{l, r\} \quad ps \in P_T(K)}{\{t(K_1, x_1)\backslash^{ps}, \ldots, t(K_n, x_n)\backslash^{ps}\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) : K\backslash^{ps}}$$

with $\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) \equiv \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1)\backslash^{ps}$;

$$\mathcal{D}(\Pi : (\to I_{pIL})\frac{\Pi_1 : \{K_1, \ldots, K_n, H\} \vdash_{pIL} K}{\{K_1, \ldots, K_n\} \vdash_{pIL} H \to K}) =$$

$$(\to I)\frac{\{t(K_1, x_1), \ldots, t(K_n, x_n), t(H, x)\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n,x\}}(\Pi_1) : K}{\{t(K_1, x_1), \ldots, t(K_n, x_n)\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) : H \to K}$$

with $\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi)$ such that, for every path $p \in P_T(H)$, $(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi))^p \equiv (\lambda t(H, x)^p.(\mathcal{T}_{\{x_1,\ldots,x_n,x\}}(\Pi_1))^p)^{(H\to K)^p}$;

$$\mathcal{D}(\Pi : (\to E_{pIL})\frac{\Pi_1 : \{K_1, \ldots, K_n\} \vdash_{pIL} H \to K \quad \Pi_2 : \{K_1, \ldots, K_n\} \vdash_{pIL} H}{\{K_1, \ldots, K_n\} \vdash_{pIL} K}) =$$

$$(\to E)\frac{\begin{array}{c}\{t(K_1, x_1), \ldots, t(K_n, x_n)\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1) : H \to K \\ \{t(K_1, x_1), \ldots, t(K_n, x_n)\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_2) : H \\ BV(E(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1))) \cap BV(E(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_2))) = \emptyset\end{array}}{\{t(K_1, x_1), \ldots, t(K_n, x_n)\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) : K}$$

with $\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi)$ such that, for every path $p \in P_T(H)$, $(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi))^p \equiv ((\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1))^p(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_2))^p)^{K^p}$;

$$\mathcal{D}(\Pi : (\wedge I_{pIL})\frac{\Pi_1 : \{H_1[p := [\sigma_1, \sigma_1]], \ldots, H_n[p := [\sigma_n, \sigma_n]]\} \vdash_{pIL} K[p := [\sigma, \tau]]}{\{H_1[p := \sigma_1], \ldots, H_n[p := \sigma_n]\} \vdash_{pIL} K[p := \sigma \wedge \tau]}) =$$

$$(\wedge I)\frac{\begin{array}{c}\{t(H_1[p := [\sigma_1, \sigma_1]], x_1), \ldots, t(H_n[p := [\sigma_n, \sigma_n]], x_n)\} \vdash_\lambda \\ \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1) : K[p := [\sigma, \tau]]\end{array}}{\{t(H_1[p := \sigma_1], x_1), \ldots, t(H_n[p := \sigma_n], x_n)\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) : K[p := \sigma \wedge \tau]}$$

with $(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi))^p \equiv ((\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1))^{pl} \wedge (\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1))^{pr})^{\sigma\wedge\tau}$.
Let $s \in \{l, r\}$. Then:

$$\mathcal{D}(\Pi : (\wedge E^s_{pIL})\frac{\Pi_1 : \{K_1, \ldots, K_n\} \vdash_{pIL} K[p := \sigma_l \wedge \sigma_r]}{\{K_1, \ldots, K_n\} \vdash_{pIL} K[p := \sigma_s]}) =$$

$$(\wedge E^s)\frac{\{t(K_1, x_1), \ldots, t(K_n, x_n)\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1) : K[p := \sigma_l \wedge \sigma_r]}{\{t(K_1, x_1), \ldots, t(K_n, x_n)\} \vdash_\lambda \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) : K[p := \sigma_s]} \text{ with}$$

$(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi))^p \equiv (D_s((\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1))^p))^{\sigma_s}$,

· The erasing function E maps every $M \in \Lambda_{IT}$ to an *untyped* λ-term, as follows:

$$\begin{aligned} E(x^\sigma) &= x \\ E((M^{\sigma\to\tau} N^\sigma)^\tau) &= E(M^{\sigma\to\tau})E(N^\sigma) \\ E((\lambda x^\sigma.M^\tau)^{\sigma\to\tau}) &= \lambda x.E(M^\tau) \\ E((M^\sigma \wedge N^\tau)^{\sigma\wedge\tau}) &= E(M^\sigma) \text{ if } E(M^\sigma) \equiv E(N^\tau), \text{ undefined otherwise} \\ E((D_s(M^{\sigma_l\wedge\sigma_r}))^{\sigma_s}) &= E(M^{\sigma_l\wedge\sigma_r}) \qquad s \in \{l, r\} \end{aligned}$$

Λ_{IT} is the set of terms t such that $t \equiv (\mathcal{T}_{\{x_1,\dots,x_n\}}(\Pi))^p$, for some $\Pi : \{K_1, \dots, K_n\} \vdash_{pIL} K$, some set $\{x_1, .., x_n\}$, and some $p \in P_T(K)$.

The next property states that the erasing function E is totally defined. Its proof states that $\mathcal{D}$, applied to a derivation Π, generates a typed kit such that the erasure of all the λ-terms attached to its leaves yield identical untyped λ-terms belonging to the form of Π.

Property 3.1 *Let $\Pi : \{H_1, \dots, H_n\} \vdash_{pIL} K$ be given. Then:*

i) $E(M) \in U(\Pi)$*, for every M which is a leaf of $\mathcal{T}_{\{x_1,\dots,x_n\}}(\Pi)$.*

ii) $E(M) \equiv E(N)$*, for every M and N, which are leaves of $\mathcal{T}_{\{x_1,\dots,x_n\}}(\Pi)$.*

iii) $E(M^\sigma) \equiv E(N^\tau)$*, for every $(M^\sigma \wedge N^\tau)^{\sigma\wedge\tau}$ which is a leaf of $\mathcal{T}_{\{x_1,\dots,x_n\}}(\Pi)$.*

Proof. i) and ii) by induction on Π. iii) follows from ii). □

Moreover Property 3.1 assures that $\mathcal{D}$ well behaves with respect to the equivalence classes of IL, i.e., typed terms encode derivations in IL.

If S is a set of typed variable, let $E(S)$ be the set of its erasures. Terms of Λ_{IT} satisfy a kind of hygiene condition on the names of variables, stronger than the one defined by Barendregt [1]. In order to define formally such a condition, let us introduce a notation for speaking about subterms of a given term, using the already defined notion of path. For sake of clarity, we will not specify the whole type decoration in writing terms of Λ_{IT}.

Definition 3.2 Let $M \in \Lambda_{IT}$, and let p be a path. The *subterm* of M at path p is inductively defined as follows:

- $(M)^\epsilon = M$;
- $(MN)^{lp} = (M)^p$,$(MN)^{rp} = (N)^p$;
- $(\lambda x^\sigma.M)^p = (M)^p$;
- $(M \wedge N)^{lp} = (M)^p$, $(M \wedge N)^{rp} = (N)^p$;
- $(D_s(M))^p = (M)^p$ $(s \in \{l, r\})$.

Property 3.2 *Let $\pi : \{H_1, \dots, H_n\} \vdash_{IL} K$. For every $\{x_1, \dots, x_n\}$ and $\Pi \in \pi$, every term M which is a leaf of $\mathcal{T}_{\{x_1,\dots,x_n\}}(\Pi))$ is such that:*

i) $E(FV(M))$ *and* $E(BV(M))$ *are disjoint;*

ii) Let $\lambda x^\sigma.N$ and $\lambda y^\tau.P$ be two different subterms of M. $E(x^\sigma) \equiv E(y^\tau)$ if and only if there is $Q \wedge Q' \equiv (M)^p$, for some p, and a path q such that

$\lambda x^{\sigma}.N \equiv (Q \wedge Q')^{lq}$ *and* $\lambda x^{\tau}.P \equiv (Q \wedge Q')^{rq}$.

Proof.

i) Easy, since the system $\vdash_{\lambda}$ is a natural deduction system, without an explicit weakening rule;

ii) By the condition on variable names in the rule $(\rightarrow E)$.

$\square$

Point i) of the previous property says that erasures of free and bound variables are different. Point ii) is peculiar, and it is quite important for defining the operational semantics of Λ_{IT}: different bound variables have the same erasure if and only if they are "corresponding" subterms of an intersection term, i.e., if they correspond to the same untyped subterm.

The reduction rules of the system $\vdash_{\lambda}$ derive from those of the system $\vdash_{IL}$, as in the next definition. The renaming of variables in the definition of the $\rightarrow$-rewriting step assures that the hygiene condition on variable names is preserved by rewriting.

Definition 3.3 • The $\wedge$-rewriting step on the system $\vdash_{\lambda}$ is just the decorated version of the $\wedge$-IL-rewriting step on $\vdash_{IL}$, defined in Definition 2.7.

• A $\rightarrow$-*rewriting* step on on the system $\vdash_{\lambda}$ is:

$$\dfrac{\Pi : (\rightarrow I)\dfrac{\Pi_0 : \Gamma^*, t(H,x) \vdash_{\lambda} \mathcal{T}_{dom(\Gamma^*)\cup\{x\}}(\Pi_0) : K}{\Gamma^* \vdash_{\lambda} \mathcal{T}_{dom(\Gamma^*)}(\Pi) : H \rightarrow K} \qquad \Pi_1 : \Gamma^* \vdash_{\lambda} \mathcal{T}_{dom(\Gamma^*)}(\Pi_1) : H}{\Gamma \vdash_{\lambda} K}(\rightarrow E) \rightsquigarrow_{\rightarrow}$$

$$\rightsquigarrow_{\rightarrow} S(\Pi_1, \Pi_0) : \Gamma \vdash_{\lambda} K$$

where $S(\Pi_1, \Pi_0)$ is the deductive structure obtained from Π_0 by:

- replacing every axiom of the shape $\Gamma' \cup \{H\} \vdash_{pIL} H$ by a copy of the derivation $\Pi_1 : \Gamma \cup \{H\} \vdash_{pIL} H$, in such a way that different copies have disjoint set of bound variables (it is always possible since $\Gamma' \subseteq \Gamma$);
- changing the contexts according with the previous substitution.

The rewriting steps of IL, defined in Definition 2.7, induce a rewriting system for Λ_{IT}.

Definition 3.4 i) The rewriting rules for Λ_{IT} are the following:

$$((\lambda x^{\sigma}.M^{\tau})^{\sigma\rightarrow\tau} N^{\sigma})^{\tau} \rightarrow_{\beta} (M[N^{\sigma}/x^{\sigma}])^{\tau}$$
$$(D_l((M^{\sigma} \wedge N^{\tau})^{\sigma\wedge\tau})^{\sigma} \rightarrow_{\wedge} M^{\sigma}$$
$$(D_r((M^{\sigma} \wedge N^{\tau})^{\sigma\wedge\tau})^{\tau} \rightarrow_{\wedge} N^{\tau}$$

where $((\lambda x^{\sigma}.M^{\tau})^{\sigma\rightarrow\tau} N^{\sigma})^{\tau}$ and $(D_s((M^{\sigma} \wedge N^{\tau})^{\sigma\wedge\tau})^{\sigma}$ ($s \in \{r, l\}$) are called respectively a β-pre-redex and a $\wedge$-redex.

ii) Let $M, N \in \Lambda_{IT}$. Let $R_1, ..., R_n$ be all and only the β-pre-redexes in M such that $E(R_1) \equiv E(R_2) \equiv ... \equiv E(R_n)$, and let $R_i \equiv (\lambda x^{\sigma_i}.M_i[x_{i1}, ..., x_{im}])P_i$, where x_{ij} denotes the j-th occurrence of x^{σ_i} in M_i $(1 \leq i \leq n, 1 \leq j \leq m)$.

Let $P_{i1}, ..., P_{im}$ be m copies of P_i such that for all i, j, $BV(P_{ih}) = BV(P_{jh})$ and $h \neq k$ implies $BV(P_{ih}) \cap BV(P_{ik}) = \emptyset$. Let N be obtained from M by replacing every R_i by $M_i[P_{i1}/x_{i1}, ..., P_{im}/x_{im}]$. Then $M \rightarrow_{IT_\beta} N$. The set $\{R_1, ..., R_n\}$ is called a β-redex.

iii) $\rightarrow_{IT}$ is the union of the two reductions $\rightarrow_{IT_\beta}$ and $\rightarrow_{IT_\wedge}$. $\rightarrow^*_{IT}$ denotes the reflexive and transitive closure of $\rightarrow_{IT}$.

The definition of $\rightarrow_{IT_\beta}$ looks involved, but it is very natural, as it can be seen in the following example. The renaming of variables in the substitution is necessary for mantaining the hygiene condition.

Example 3.5 Let a, b, c, d denote respectively the types $\sigma \rightarrow \sigma$, $\tau \rightarrow \tau$, $(\sigma \rightarrow \sigma) \rightarrow (\sigma \rightarrow \sigma) \rightarrow \mu$ and $(\tau \rightarrow \tau) \rightarrow (\tau \rightarrow \tau) \rightarrow \nu$. Then $M \equiv ((\lambda y^a.x^c yy)(\lambda z^\sigma.z) \wedge (\lambda y^b.x^d yy)(\lambda z^\tau.z))^{\mu\wedge\nu}$ is a well formed term of Λ_{IT}. Then $\{(\lambda y^a.x^c yy)(\lambda z^\sigma.z), (\lambda y^b.x^d yy)(\lambda z^\tau.z)\}$ is a β-redex and
$M \rightarrow_{IT_\beta} x^c(\lambda t^\sigma.t)(\lambda v^\sigma.v) \wedge x^d(\lambda t^\tau.t)(\lambda v^\tau.v)$.

The proof of strong normalization of derivations of IL assures the strong normalization of $\vdash_\lambda$, and consequently the following theorem holds:

Theorem 3.6 *Λ_{IT} is strongly normalizing with respect to the reduction $\rightarrow_{IT}$.*

The operational behaviour of Λ_{IT}, as defined before, needs some comments. In the system IL the connective $\rightarrow$ has a *global behaviour*: indeed both its introduction and elimination act in parallel in all the leaveas of the involved kits. On the contrary the connective $\wedge$ has a standard local behaviour. This is reflected in the rewriting rules defined in Definition 2.7. The global behaviour of $\rightarrow$ gives rise, in the typed setting, to the rule $\rightarrow_{IT_\beta}$. Property 3.1 assures us that, if two subterms come from the same kit, then their erasures are identical, and so the definition of the reduction $\rightarrow_{IT_\beta}$ is correct. But it is important to point out that α-rule is not correct in this setting: fortunately we don't need it, since Property 3.2.i).

Terms of Λ_{IT} can be defined independently from the logical system, in the following way:

Definition 3.7 The language Λ_{IT} is inductively defined as follows:

- $\sigma \in LJ$ and $x \in Var$ imply $x^\sigma \in \Lambda_{IT}$;
- $M^\tau \in \Lambda_{IT}$ and $x^\sigma \notin BV(M^\tau)$ and there is no $\rho \not\equiv \sigma$ such that $x^\rho \in FBV(M^\tau)$ imply $(\lambda x^\sigma.M^\tau)^{\sigma\rightarrow\tau} \in \Lambda_{IT}$;
- $M^{\sigma\rightarrow\tau} \in \Lambda_{IT}$ and $N^\sigma \in \Lambda_{IT}$ and $(E(BV(M^{\sigma\rightarrow\tau}))$ and $E(BV(N^\sigma))$ disjoint) and $(x^\rho \in FV(M^{\sigma\rightarrow\tau})$ and $y^\gamma \in FV(N^\sigma)$ and $\rho \not\equiv \gamma$ imply $x \not\equiv y)$ imply $(M^{\sigma\rightarrow\tau}N^\sigma)^\tau \in \Lambda_{IT}$.
- $M^\sigma \in \Lambda_{IT}$ and $N^\tau \in \Lambda_{IT}$ and $E(M^\sigma) \equiv E(N^\tau)$ imply $(M^\sigma \wedge N^\tau)^{\sigma\wedge\tau} \in \Lambda_{IT}$;
- $M^{\sigma\wedge\tau} \in \Lambda_{IT}$ implies $(D_s(M^{\sigma\wedge\tau}))^\phi \in \Lambda_{IT}$, where $\phi \equiv \sigma$ if $s \equiv l$, and $\phi \equiv \tau$ if $s \equiv r$.

Definition 3.7 generates terms that satisfy property 3.2, and so the reduction is correctly defined.

The definition of Λ_{IT} given independently from the logical system is not standard, because of the constraints assuring the hygiene condition on variables names. But it is possible to give a different definition of both the language and the reduction, in such a way that the reduction is defined in the standard way as contextual closure of some reduction rules. The resulting language, which we call Λ'_{IT}, is defined in the next definition.

Definition 3.8 i) The language Λ'_{IT} is inductively defined as follows:

- $\sigma \in LJ$ and $x \in Var$ imply $x^\sigma \in \Lambda'_{IT}$;
- $M^\tau \in \Lambda'_{IT}$ implies $(\lambda x^\sigma.M^\tau)^{\sigma\to\tau} \in \Lambda'_{IT}$;
- $M^{\sigma\to\tau} \in \Lambda'_{IT}$ and $N^\sigma \in \Lambda'_{IT}$ imply $(M^{\sigma\to\tau} N^\sigma)^\tau \in \Lambda'_{IT}$.
- $M^\sigma \in \Lambda'_{IT}$ and $N^\tau \in \Lambda'_{IT}$ and $E(M^\sigma) = E(N^\tau)$ imply $(M^\sigma \wedge N^\tau)^{\sigma\wedge\tau} \in \Lambda'_{IT}$ where $=$ is the untyped α-β-equality;
- $M^{\sigma\wedge\tau} \in \Lambda'_{IT}$ implies $(D_s(M^{\sigma\wedge\tau}))^\phi \in \Lambda'_{IT}$, where $\phi \equiv \sigma$ if $s \equiv l$, and $\phi \equiv \tau$ if $s \equiv r$.

ii) The rewriting relation for Λ'_{IT} ($\to_{IT'}$) is the contextual closure of the following rules:

$$(\lambda x^\sigma.M^\tau)^{\sigma\to\tau} \to_\alpha (\lambda y^\sigma.M[y^\sigma/x^\sigma]^\tau)^{\sigma\to\tau} (y^\sigma \notin FV(M^\tau))$$
$$((\lambda x^\sigma.M^\tau)^{\sigma\to\tau} N^\sigma)^\tau \to_\beta (M[N^\sigma/x^\sigma])^\tau$$
$$(D_l((M^\sigma \wedge N^\tau)^{\sigma\wedge\tau})^\sigma \to_\wedge M^\sigma$$
$$(D_r((M^\sigma \wedge N^\tau)^{\sigma\wedge\tau})^\tau \to_\wedge N^\tau$$

.

Property 3.3 *Both Λ'_{IT} and its operational semantics are well defined.*

Proof. Since terms typable in IT are all strongly normalizing, $=$ is decidable. □

Clearly Λ_{IT} and Λ'_{IT} are equivalent, in the following sense. Two translation functions can be defined, $F : \Lambda_{IT} \to \Lambda'_{IT}$ and $G : \Lambda'_{IT} \to \Lambda_{IT}$ such that:

- $M, N \in \Lambda_{IT}$ and $M \to_{IT} N$ imply $F(M) \to^*_{IT'} F(N)$ (where $\to^*_{IT'}$ is the transitive closure of $\to_{IT'}$);
- $M, N \in \Lambda'_{IT}$ and $M \to_{IT'} N$ implies there is $P \in \Lambda_{IT'}$ such that $M \to^*_{IT'} P$ and $G(M) \to_{IT} G(P)$.

4 Intersection Typed λ-calculus II

From a proof-theoretical point of view, Λ_{IT}, and its variant Λ'_{IT}, is certainly a correct typed version of IT. However, Λ_{IT} might not be completely satisfactory from a user point of view. The associativity and the commutativity of the connective $\wedge$ introduce annoying and useless syntactical differences between terms with the same semantics. Here below there is an example of this:

Example 4.1 The terms P_1, P_2, P_3 defined as follows belong to Λ_{IT}.

$P_1 \equiv ((\lambda x^\sigma.x^\sigma)^{\sigma\to\sigma} \wedge ((\lambda x^\tau.x^\tau)^{\tau\to\tau} \wedge (\lambda x^\rho.x^\rho)^{\rho\to\rho})^{((\tau\to\tau)\wedge(\rho\to\rho))})^{(\sigma\to\sigma)\wedge((\tau\to\tau)\wedge(\rho\to\rho))}$
$P_2 \equiv (((\lambda x^\sigma.x^\sigma)^{\sigma\to\sigma} \wedge (\lambda x^\tau.x^\tau)^{\tau\to\tau})^{((\sigma\to\sigma)\wedge(\tau\to\tau))} \wedge (\lambda x^\rho.x^\rho)^{\rho\to\rho})^{((\sigma\to\sigma)\wedge(\tau\to\tau))\wedge(\rho\to\rho)}$.
$P_3 \equiv (((\lambda x^\tau.x^\tau)^{\tau\to\tau} \wedge (\lambda x^\rho.x^\rho)^{\rho\to\rho})^{((\tau\to\tau)\wedge(\rho\to\rho))} \wedge (\lambda x^\sigma.x^\sigma)^{\sigma\to\sigma}))^{((\tau\to\tau)\wedge(\rho\to\rho))\wedge(\sigma\to\sigma))}$.
P_1, P_2 and P_3 are different typed terms, but their types are semantically equivalent.

We will try to define a new typed language, in order to better deal with the associative and commutative property of $\wedge$: the idea is to allow the connective $\wedge$ to have not fixed arity.

Definition 4.2 [Generalized systems gLJ, gIT and gIL]

i) Let the *generalized LJ formulae* be defined by the following grammar:

$$\sigma ::= V \mid \sigma \to \sigma \mid \underbrace{\sigma \wedge ... \wedge \sigma}_{n\geq 2}$$

ii) The *generalized system* gLJ is obtained from that one in Fig. 2, by replacing rules $(\wedge I_{LJ})$ and $(\wedge E^s_{LJ})$ $(s \in \{l, r\})$ by the following rules:

$$(\wedge I_{gLJ})\frac{\Gamma \vdash_{gLJ} \sigma_1 \quad ... \quad \Gamma \vdash_{gLJ} \sigma_n}{\Gamma \vdash_{gLJ} \sigma_1 \wedge ... \wedge \sigma_n}$$

$$(\wedge E^{i_1,...,i_m}_{gLJ})\frac{\Gamma \vdash_{gLJ} \sigma_1 \wedge ... \wedge \sigma_n \quad \{i_1, ..., i_m\} \subseteq \{1, ..., n\}}{\Gamma \vdash_{gLJ} \sigma_{i_1} \wedge ... \wedge \sigma_{i_m}}$$

iii) The *generalized type assignment system* gIT is obtained from that one in Fig. 1, by replacing rules $(\wedge I_{IT})$ and $(\wedge E^s_{IT})$ $(s \in \{l, r\})$ by the following rules:

$$(\wedge I_{gIT})\frac{\Gamma \vdash_{gIT} M : \sigma_1 \quad ... \quad \Gamma \vdash_{gIT} M : \sigma_n}{\Gamma \vdash_{gIT} M : \sigma_1 \wedge ... \wedge \sigma_n}$$

$$(\wedge E^i_{gIT})\frac{\Gamma \vdash_{gIT} M : \sigma_1 \wedge ... \wedge \sigma_n \quad \{i_1, ..., i_m\} \subseteq \{1, ...n\}}{\Gamma \vdash_{gIT} M : \sigma_{i_1} \wedge ... \wedge \sigma_{i_m}}$$

iv) A *generalized kit* is a binary tree whose leaves are labelled by generalized formulae of gLJ. Let $K_\wedge \equiv \sigma_1 \wedge ... \wedge \sigma_n$, where $\sigma_1, ..., \sigma_n$ are the names of the leaves of K, read in pre-order. The generalized system $\vdash_{gpIL}$ derives judgments $\Gamma \vdash_{gpIL} K$, where Γ is a sequence of generalized kits and K is a generalized kit. Its rules are the rules of the system pIL, but for the rules $(\wedge I_{pIL})$ and $(\wedge E^s_{pIL})$, which are replaced by the rules:

$$(\wedge I_{gpIL})\frac{\{H_1[p := H], ..., H_n[p := H]\} \vdash_{gpIL} K[p := K'] \qquad \forall q \in P_T(H).H^q \equiv \tau}{\{H_1[p := \tau], ..., H_n[p := \tau]\} \vdash_{gpIL} K[p := K'_\wedge]}$$

$$(\wedge E^i_{gpIL})\frac{\Gamma \vdash_{gpIL} K[p := \sigma_1 \wedge ... \wedge \sigma_n] \quad \{i_1, ..., i_m\} \subseteq \{1, ...n\}}{\Gamma \vdash_{gpIL} K[p := \sigma_{i_1} \wedge ... \wedge \sigma_{i_m}]}$$

v) The *generalized Intersection Logic* gIL is the set $\vdash_{gpIL} / \sim$, where the congruence relation $\sim$ between derivations is the obvious extension to the generalized case of that one defined in Definition 2.2.

It is easy to check that the relationship between the three systems is preserved, i.e., the following theorems holds:

Theorem 4.3 *Let* $\Pi : \{K_1,\ldots, K_n\} \vdash_{gIL} H$. *For all path* p *terminal in* H, $\{\Pi^p : K^p_1, \ldots, K^p_n\} \vdash_{gLJ} H^p$, *and* $U(\Pi^p) = U(\Pi)$.

Theorem 4.4 *Let* π *be a proper derivation in* gIL. *Then* $\pi : \{\sigma_1, \ldots, \sigma_n\} \vdash_{gIL} \tau$ *if, and only if,* $\{x_1 : \sigma_1, \ldots, x_n : \sigma_n\} \vdash_{gIT} T_{\{x_1,\ldots,x_n\}}(\pi) : \tau$.

In both the systems the notion of normalization is extended in the obvious way, by taking into account the generalized arity of the connective $\wedge$. Clearly the strong normalization property is preserved too. The intersection typed λ-calculus Λ^2_{IT} will be defined through a decoration of the derivations of the system $\vdash_{gpIL}$.

Definition 4.5 i) A *generalized typed kit* $\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi)$ is a generalized tree whose leaves are labelled by λ-terms, typed with generalized formulae of gLJ, and whose set of free variables is $\{x_1, \ldots, x_n\}$. $t(H, x)$ is defined as in Definition 3.1.ii).

ii) The algorithm $\mathcal{D}'$decorating gIL and the erasure function $E : \Lambda'_{IT} \to \Lambda$ are mutually defined as follows.

- For every Π and π, such that $\Pi \in \pi$ and $\pi : \{K_1, \ldots, K_n\} \vdash_{gIL} H$, $\mathcal{D}'$ maps Π to the judgment $\{t(K_1, x_1), \ldots, t(K_n, x_n)\} \vdash_{g\lambda} \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi)$, where $\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) \simeq H$. The definition of $\mathcal{D}'$ coincides with the one of $\mathcal{D}$, but for the following cases:

  $\mathcal{D}'(\Pi :$
  $$(\wedge I_{gpIL})\frac{\{H_1[p := H], ..., H_n[p := H]\} \vdash_{gpIL} K[p := K'] \quad \forall q \in P_T(H).H^q \equiv \tau}{\{H_1[p := \tau], ..., H_n[p := \tau]\} \vdash_{gpIL} K[p := K'_\wedge]}) =$$
  $$(\wedge I_g)\frac{\begin{array}{c}\{t(H_1[p := H], x_1), \ldots, t(H_n[p := H], x_n)\} \vdash_{g\lambda} \\ \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1) : K[p := K'] \\ (E(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1)))^{pq} \equiv (E(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1)))^{pq} \quad (pq \in P_T(K))\end{array}}{\{t(H_1[p := \tau], x_1), \ldots, t(H_n[p := \tau], x_n)\} \vdash_{g\lambda} \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) : K[p := K'_\wedge]}$$
  with $(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi))^p \equiv (M^{\sigma_1}_1 \wedge ... \wedge M^{\sigma_n}_n)^{\sigma_1 \wedge \ldots \wedge \sigma_n}$, if $(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1))^{pq_i} \equiv M^{\sigma_i}_i$, where q_i is the i-th subpath of p in the pre-order reading, such that $pq_i \in P_T(K)$. Moreover, for every q disjoint from p and $q \in P_T(K)$, $(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi))^q \equiv (\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1))^q$;

$$\mathcal{D}'(\Pi : (\wedge E_{pIL}^{i_1,\ldots,i_m})\frac{\Delta \vdash_{pIL} K[p := \sigma_1 \wedge \ldots \wedge \sigma_n]}{\Delta \vdash_{pIL} K[p := \sigma_{i_1} \wedge \ldots \wedge \sigma_{i_m}]}) =$$

$$(\wedge E_g^{i_1,\ldots,i_m})\frac{\Pi_1 : t(\Delta) \vdash_{g\lambda} \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1) : K[p := \sigma_1 \wedge \ldots \wedge \sigma_n]}{t(\Delta) \vdash_{g\lambda} \mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi) : K[p := \sigma_{i_1} \wedge \ldots \wedge \sigma_{i_m}]}$$

where, if $\Delta = \{K_1, \ldots, K_n\}$, then $t(\Delta) = \{t(K_1, x_1), \ldots, t(K_n, x_n)\}$ and if $(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi_1))^p \equiv M^{\sigma_1 \wedge \ldots \wedge \sigma_n}$, then $(\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi))^p \equiv (D_{i_1,\ldots,i_m}(M^{\sigma_1 \wedge \ldots \wedge \sigma_n}))^{\sigma_{i_1} \wedge \ldots \wedge \sigma_{i_m}}$.

- The erasing function E maps every $M \in \Lambda_{IT}^2$ to an *untyped* λ-term, as follows:

$$\begin{aligned}
E(x^\sigma) &= x \\
E((M^{\sigma\to\tau} N^\sigma)^\tau) &= E(M^{\sigma\to\tau})E(N^\sigma) \\
E((\lambda x^\sigma . M^\tau)^{\sigma\to\tau}) &= \lambda x . E(M^\tau) \\
E((M_1^{\sigma_1} \wedge \ldots \wedge M_n^{\sigma_n})^{\sigma_1 \wedge \ldots \wedge \sigma_n}) &= E(M_1^{\sigma_1}) \text{ if } E(M_i^{\sigma_i}) \equiv E(M_j^{\sigma_j}), \\
&\qquad \text{undefined otherwise} \\
E((D_{i_1,\ldots,i_m}(M_1^{\sigma_1} \wedge \ldots \wedge M_n^{\sigma_n}))^{\sigma_{i_1} \wedge \ldots \wedge \sigma_{i_m}}) &= E(M_{i_1}^{\sigma_{i_1}} \wedge \ldots \wedge M_{i_m}^{\sigma_{i_m}})
\end{aligned}$$

Λ_{IT}^2 is the set of typed terms of $\mathcal{L}$ which are leaves of $\mathcal{T}_{\{x_1,\ldots,x_n\}}(\Pi)$, for some Π and some $\{x_1, \ldots, x_n\}$. The rewriting steps of the generalized system $\vdash_{gpIL}$ induce a rewriting system for Λ_{IT}^2, obtained from Definition 3.4 by replacing rule $\to_\wedge$ by the following rule:

$$(D_{i_1,\ldots,i_m}((M_1^{\sigma_1} \wedge \ldots \wedge M_n^{\sigma_n})^{\sigma_1 \wedge \ldots \wedge \sigma_n})^{\sigma_{i_1} \wedge \ldots \wedge \sigma_{i_m}} \to_\wedge (M_{i_1}^{\sigma_{i_1}} \wedge \ldots \wedge M_{i_m}^{\sigma_{i_m}})^{\sigma_{i_1} \wedge \ldots \wedge \sigma_{i_m}}$$

The strong normalization of derivations of IL assures that the following theorem holds:

Theorem 4.6 *Λ_{IT}^2 is strongly normalizing.*

It is possible to define terms of Λ_{IT}^2 independently from the logical system, in a similar way as we did in the previous section for language Λ_{IT}.

Example 4.7 The three terms in the Example 4.1 belong to Λ_{IT}^2. The term: $P_4 \equiv ((\lambda x^\sigma . x^\sigma)^{\sigma\to\sigma} \wedge (\lambda x^\tau . x^\tau)^{\tau\to\tau} \wedge (\lambda x^\rho . x^\rho)^{\rho\to\rho})^{(\sigma\to\sigma)\wedge(\tau\to\tau)\wedge(\rho\to\rho)}$
belongs to Λ_{IT}^2 but not to Λ_{IT}.

It seems now natural to ask for a further simplification of the typed language, to avoid reduntant terms. More precisely, we would like to have a language such that the term P_4 belongs to it, but neither P_1 nor P_2 nor P_3 do.

It is easy to check that the nested subterms of P_i $(1 \leq i \leq 3)$ arise from the fact that the derivations encoded by these terms have two consecutive applications of rule $(\wedge I_{gIL})$. So, in order to impose to typed terms that components of intersection terms cannot be in their turn intersection terms, while preserving the property that typed terms encode derivations, it is necessary to restrict the set of derivations, considering just that ones having not consecutive applications of rule $(\wedge I_{gIL})$. It is easy to check that every derivation can be transformed in a derivation of this kind. However, such condition on derivations is not preserved under reduction. In fact, let consider the following two terms (where, for reasons of readability, not all the subterms are decorated with types):

$$Q_1 \equiv (\lambda x^{\sigma\wedge\tau}.\lambda y^{((\sigma\wedge\tau)\wedge(\sigma\wedge\tau))\to\mu}.y(x\wedge x))(N^\sigma \wedge P^\tau)^{\sigma\wedge\tau}$$

and

$$Q_2 \equiv (\lambda x^\rho.(M_1^\sigma \wedge M_2^\tau)^{\sigma\wedge\tau})P^\rho \wedge (\lambda x^{\rho'}.(N_1^{\sigma'} \wedge N_2^{\tau'})^{\sigma'\wedge\tau'})Q^{\rho'}$$

Both Q_1 and Q_2 satisfy the requirement (components of an intersection term are not in their turn intersection terms), but they reduce respectively to:

$$Q_1' \equiv (\lambda y^{((\sigma\wedge\tau)\wedge(\sigma\wedge\tau))\to\mu}.y((N^\sigma \wedge P^\tau)^{\sigma\wedge\tau} \wedge (N^\sigma \wedge P^\tau)^{\sigma\wedge\tau}))$$

and

$$Q_2' \equiv ((M_1[P^\rho/x^\rho])^\sigma \wedge (M_2[P^\rho/x^\rho])^\tau)^{\sigma\wedge\tau} \wedge ((N_1[Q^{\rho'}/x^{\rho'}])^{\sigma'} \wedge (N_2[Q^{\rho'}/x^{\rho'}])^{\tau'})^{\sigma'\wedge\tau'}$$

which both do not obey the constraint.

5 Conclusions and comparison with related works

In this paper the design of an intersection typed λ-calculus is discussed, starting from the assumption that a typed language *must* be defined as decoration of derivations in some logic. In particular two different proposals are made, all obtained from the Intersection Logic (IL), introduced in [8], which has been proved to be a correct logical foundation for the Intersection Type Assignment System. Λ_{IT} is obtained by decorating IL. The operational semantics of Λ_{IT} is inherited from the normalization procedure of IL, where both the introduction and the elimination of the arrow have a global behaviour. So some typed β-reductions in Λ_{IT}, involving redexes which are corresponding in different components of an intersection term, must be performed in parallel: in fact they are copies of the same untyped redex. Λ'_{IT} is a variant of Λ_{IT}, with a completely sequential behaviour, obtained by relaxing the formation rules of terms. The second language, Λ^2_{IT}, is obtained from a modification of IL, which allows the connective $\wedge$ to have not fixed arity, in order to better deal with its commutative and associative properties.

The typed language proposed in [2] is similar to Λ^2_{IT}, having $\wedge$ not fixed arity, but it differs from it in the $(\wedge E)$ rule, in the sense that it just allows to extract one of the components of an intersection. I.e., the following formation rule of Λ^2_{IT}:

- $M^{\sigma_1\wedge\ldots\wedge\sigma_n} \in \Lambda'_{IT}$ and $\{i_1,...,i_m\} \subseteq \{1,...,n\}$ imply $(D_{i_1,...,i_m}((M_1^{\sigma_1} ... \wedge M_n^{\sigma_n})^{\sigma_1...\wedge\sigma_n})^{\sigma_{i_1}\wedge\ldots\wedge\sigma_{i_m}} \in \Lambda'_{IT}$.

is replaced by:

- $M^{\sigma_1\wedge\ldots\wedge\sigma_n}$ belongs to the language implies $(D_i((M_1^{\sigma_1}\ldots\wedge M_n^{\sigma_n})^{\sigma_i\ldots\wedge\sigma_n})^{\sigma_i}$ belongs to the language.

While this language is apparently easier, derivations in it can be more difficult. In fact, let consider the term in Λ_{IT}: $(D_l(x)^{(\sigma\wedge\tau)\wedge(\mu\wedge\nu))})^{\sigma\wedge\tau}$. In Λ_{IT}^2, it can be simplified in $(D_{1,2}(x)^{\sigma\wedge\tau\wedge\mu\wedge\nu})^{\sigma\wedge\tau}$. In the language in [2] the term corresponding to it, built from the same axioms, is $((D_1(x)^{\sigma\wedge\tau\wedge\mu\wedge\nu})^{\sigma}\wedge(D_2(x)^{\sigma\wedge\tau\wedge\mu\wedge\nu})^{\tau})^{\sigma\wedge\tau}$. The operational semantics of this language is not studied.

The typed language λ^{CIL}, proposed in [11] and [12], while built from a bigger set of connectives, when restricted to the connectives $\to$ and $\wedge$, is similar to Λ_{IT}. λ^{CIL} has not a logical foundation. The gain of the logical foundation of Λ_{IT} is essentially in the definition of its operational semantics. In fact, the decoration of IL generates terms of Λ_{IT} that obey to an hygiene condition on the names of variables. Thanks to that condition, the scope of the parallel typed β- reduction can be very easily identified, while in λ^{CIL} its definition is quite complex.

The language defined in [6] is not a typed language, but it is related to this work, since it could be easily transformed into an intersection typed one. It allows $\wedge$ having not fixed arity, and it succeds in asking that intersection terms cannot be immediate components of intersection terms. So syntactically it is quite easy. But it has been defined by using strict intersection types, as defined in [9], whose logical foundation is unclear, since $\wedge$ is not treated as a logical connective.

Acknowledgments

The author would like to thank Luca Roversi and Betti Venneri for stimulating discussions about the topic of this paper, and an anonymous referee for pointing out a problem in the definition of typed reduction in the first version of this paper.

References

[1] Barendregt H., The Λ-calculus: its syntax and semantics, North Holland, 1984.

[2] Capitani B., “A Logical Characterization of Intersection Types”, PhD thesis, Department of Mathematics, University of Siena, (2001).

[3] Capitani B., Venneri B.,“Hyperformulae, Parallel Deductions and Intersection Types”, Proc. BOTH 2001, *Electronic Notes in Theoretical computer Science*, 50(2), 2001, pp.180-198.

[4] Coppo M. , Dezani-Ciancaglini M., “An extension of the basic functionality theory for the λ-calculus”, *Notre Dame J. Formal Logic*, 21(4):685–693, 1980.

[5] Hindley J. R., “Coppo Dezani types do not correspond to propositional logic”, *Theoretical Computer Science*,28,1-2, (1984),pp.235-236.

[6] Kfoury A., “Beta-reduction as Unification”, *Logic Algebra and Computer Science*, Polish Academy of Science, Warsaw, (1999), pp. 241-262.

[7] Pottinger G.,“A type assignment for the strongly normalizable λ- terms”, in: *To H. B. Curry: essays on combinatory logic, lambda calculus and formalism*,Academic Press, London, (1980), pp.561-577.

[8] Ronchi Della Rocca S. and Roversi L. “Intersection logic”, *Proc. of CSL’01*, LNCS, 2142, Springer-Verlag,(2001), pp.414-428

[9] van Bakel Steffen,“Complete restrictions of the intersection type discipline”, *Theoretical Computer Scienc*e,102,1,(1992),pp.135-163.

[10] Venneri B.,“Intersection types as logical formulae”, *J. Logic Comput.*,4,2,(1994),pp.109-124.

[11] Wells J.B., Dimock A., Muller R., and Turbak F., “A typed intermediate language for flow-directed compilation”, In *7th International Joint Conference on Theory and Practice of Software Development (TAPSOFT97)*, pages 757–771.

[12] Wells J.B., Dimock A., Muller R., and Turbak F., “A Calculus with Polymorphic and Polyvariant Flow Types”, *Journal of Functional Programming*, May 2002.