Notre Dame Journal of Formal Logic
Volume 37, Number 1, Winter 1996

Strong Normalization and Typability with Intersection Types

SILVIA GHILEZAN

Abstract A simple proof is given of the property that the set of strongly normalizing lambda terms coincides with the set of lambda terms typable in certain intersection type assignment systems.

1 Introduction Intersection type assignment systems were introduced and developed in the 1980s by Barendregt, Coppo, Dezani-Ciancaglini and Venneri (see [2], [3], and [4]). They are meant to be extensions of Curry's basic functional theory which will provide types for a larger class of lambda terms. On the one hand this aim was fulfilled, and on the other hand they became of interest for their other properties as well.

We shall deal with four intersection type assignment systems: the original ones $\mathcal{D}$ and $\mathcal{D}\Omega$ introduced in [3] and [4] and their extensions $\mathcal{D}_{\leq}$ and $\mathcal{D}\Omega_{\leq}$ with the rule $(\leq)$, which involves partial ordering on types.

The problem of typability in a type system is whether there is a type for a given term. The problem of typability in the full intersection type assignment system $\mathcal{D}\Omega_{\leq}$ is trivial, since every lambda term is typable by the type ω. For the same reasons typability in $\mathcal{D}\Omega$ is trivial as well. This property changes essentially when the (ω)-rule is left out. It turns out that all strongly normalizing lambda terms are typable in $\mathcal{D}_{\leq}$ and $\mathcal{D}$, and they are the only terms typable in these systems (see Krivine [9] and van Bakel [15]).

The idea that strongly normalizing lambda terms are exactly the terms typable in the intersection type assignment systems without the (ω)-rule first appeared in [4], Pottinger [11], and Leivant [10]. Further, this subject is treated in [15], [9], and Ronchi della Rocca et al. [12], with different approaches. We shall present a modified proof of this property and compare it with the proofs mentioned above.

Section 2 is an overview of the systems considered. In Section 3 we shall present a proof à la Tait of strong normalization for $\mathcal{D}$ and $\mathcal{D}_{\leq}$ based on the proof of strong

Received April 5, 1995; revised September 15, 1995

normalization for the simply typed lambda calculus and the polymorphic lambda calculus, as given in Barendregt [1]. The strong normalization holds for all eight systems of Barendregt's cube. The systems of Barendregt's cube are given in the Church style, whereas the intersection type systems are given in the Curry style. The very essential and outstanding property of the intersection type systems $\mathcal{D}_{\leq}$ and $\mathcal{D}$ is the converse of the strong normalization property, i.e., the fact that all strongly normalizing terms are typable in $\mathcal{D}_{\leq}$ and $\mathcal{D}$. In Section 4 we shall present a simple proof of this property and compare it with the proofs mentioned above. The undecidability of typability is a consequence of this property. It will also be mentioned.

2 Basic intersection type systems Intersection types were introduced in [2], [3], and [4]. They are introduced as a generalization of Curry's type inference system in order to characterize a larger class of terms. The main idea is the introduction of a new type-forming operator, namely intersection $\cap$.

The *types* are propositional formulas with connectives $\rightarrow$ and $\cap$, where $\cap$ is a *specific conjunction* whose properties are in accordance with its interpretation as intersection of types. The basic notions of the intersection type assignment system are given in [2], [3], and [4] and can be found in the survey of typed lambda calculi in [1] and [9]. Let us recall some basic notions in order to fix the notation.

The set of types T of is defined in the following way.

Definition 2.1

- $V = \{\alpha, \beta, \gamma, \alpha_1, \ldots\} \subset T$ (V is a denumerable set of propositional variables).
- $\omega \in T$.
- If $\sigma, \tau \in T$, then $(\sigma \rightarrow \tau) \in T$.
- If $\sigma, \tau \in T$, then $(\sigma \cap \tau) \in T$.

Let $\alpha, \beta, \gamma, \alpha_1, \ldots$ be schematic letters for type variables, and let $\sigma, \tau, \varphi, \psi, \sigma_1, \ldots$ be schematic letters for types.

Definition 2.2

- A *pre-order* $\leq$ is introduced on T in the following way:

1. $\sigma \leq \sigma$	5. $(\sigma \rightarrow \rho) \cap (\sigma \rightarrow \tau) \leq \sigma \rightarrow (\rho \cap \tau)$
2. $\sigma \leq \tau, \quad \tau \leq \rho \Rightarrow \sigma \leq \rho$	6. $\sigma \cap \tau \leq \sigma, \quad \sigma \cap \tau \leq \tau$
3. $\sigma \leq \omega$	7. $\sigma \leq \tau, \quad \sigma \leq \rho \Rightarrow \sigma \leq \tau \cap \rho$
4. $\omega \leq \omega \rightarrow \omega$	8. $\sigma \leq \sigma_1, \quad \tau \leq \tau_1 \Rightarrow \sigma_1 \rightarrow \tau \leq \sigma \rightarrow \tau_1$.

- $\sigma \sim \tau$ if and only if $\sigma \leq \tau$ and $\tau \leq \sigma$.

Definition 2.3 The following rules determine the intersection type systems:

(*start rule*) $\dfrac{(x : \sigma) \in \Gamma}{\Gamma \vdash x : \sigma}$;

$(\rightarrow E)$ $\dfrac{\Gamma \vdash M : \sigma \rightarrow \tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau}$;

$(\rightarrow I)$ $\dfrac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash (\lambda x.M) : \sigma \rightarrow \tau}$;

$$(\cap E) \qquad \frac{\Gamma \vdash M : \sigma \cap \tau}{\Gamma \vdash M : \sigma}, \qquad \frac{\Gamma \vdash M : \sigma \cap \tau}{\Gamma \vdash M : \tau};$$

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

$$(\omega) \qquad \frac{}{\Gamma \vdash M : \omega};$$

$$(\leq) \qquad \frac{\Gamma \vdash M : \sigma \quad \sigma \leq \tau}{\Gamma \vdash M : \tau}.$$

The *simply typed lambda calculus* $\lambda \rightarrow$ is obtained by the (*start rule*), $(\rightarrow E)$, and $(\rightarrow I)$. We shall deal with the following intersection type assignment systems:

- $\mathcal{D}$ is $\lambda \rightarrow$ plus $(\cap E)$ and $(\cap I)$, the corresponding turnstile is denoted by $\vdash$,
- $\mathcal{D}_{\leq}$ is $\mathcal{D}$ plus $(\leq)$, the corresponding turnstile is denoted by $\vdash_{\leq}$.

Let us mention two other intersection type systems, $\mathcal{D}\Omega$ which is $\mathcal{D}$ plus (ω), and the full intersection type system $\mathcal{D}\Omega_{\leq}$, which is $\mathcal{D}$ plus $(\leq)$ and (ω) and is also called $\lambda\cap$ in the literature.

The basic combinators $\mathsf{I} \equiv \lambda x.x$, $\mathsf{K} \equiv \lambda xy.x$ and $\mathsf{S} \equiv \lambda xyz.xz(yz)$ are typable in the simply typed lambda calculus, and hence they are typable in the intersection type systems. Self-application is not typable in $\lambda \rightarrow$, but it is in $\mathcal{D}$. For example, $\lambda x.xx$ has no type in $\lambda \rightarrow$, but $\vdash \lambda x.xx : (\sigma \cap (\sigma \rightarrow \tau)) \rightarrow \tau$. Also, the term $\Omega \equiv (\lambda x.xx)(\lambda x.xx)$ has no type in $\lambda \rightarrow$, nor in $\mathcal{D}$, but it is trivially typable by ω in $\mathcal{D}\Omega$. These examples give the intuition about the difference between the classes of lambda terms which are typable in the simply typed lambda calculus and in the intersection type systems.

Further, let us consider reductions in the type systems. The main axioms of β- and η-reductions are $(\lambda x.M)N \rightarrow_{\beta} M[N/x]$ and $\lambda x.Mx \rightarrow_{\eta} M$, $x \notin FV(M)$, respectively. If a term M is typable by a certain type σ and if M β-reduces to N, $M \twoheadrightarrow_{\beta} N$, or if M η-reduces to N, $M \twoheadrightarrow_{\eta} N$, the question is whether N is typable by σ as well. The answer for $\lambda \rightarrow$ is yes, and we say that the *subject reduction property* holds for $\lambda \rightarrow$. It holds for intersection type systems as well and for all systems of Barendregt's cube (see [1]).

Proposition 2.4 *If* $\Gamma \vdash_{(\leq)} M : \sigma$ *and* $M \twoheadrightarrow_{\beta\eta} N$, *then* $\Gamma \vdash_{(\leq)} N : \sigma$.

The converse question is whether the type systems are closed under expansion. $\mathcal{D}\Omega_{\leq}$ and $\mathcal{D}\Omega$ are closed under β-expansion, and the reason why it is so is the characteristic of these systems that a term can have more types, as shown in [1]. On the other hand $\lambda \rightarrow$ is not closed under β-expansion, since $\mathsf{KI}\Omega \rightarrow_{\beta} \mathsf{I}$ and I is typable in $\lambda \rightarrow$, i.e., $\vdash_{\lambda\rightarrow} \mathsf{I} : \alpha \rightarrow \alpha$, but $\mathsf{KI}\Omega$ is even not typable in $\lambda \rightarrow$. The systems $\mathcal{D}$ and $\mathcal{D}_{\leq}$ are not closed under β-expansion. This will be discussed in Section 4.

None of the systems considered is closed under η-expansion since in all systems $\lambda x.x : \alpha \rightarrow \alpha$, α is a type variable, whereas its η-expansion $\lambda xy.xy$ cannot be typed by $\alpha \rightarrow \alpha$ in any of these systems.

3 Strong normalization for intersection type systems The proof of strong normalization for $\mathcal{D}$ and $\mathcal{D}_{\leq}$, which we present here in Theorem 3.7, is in the manner of the proof of strong normalization for the simply typed lambda calculus given by Tait [13].

A lambda term is *normalizing* if at least one of its reduction paths is finite. A lambda term is *strongly normalizing* if each of its reduction paths is finite. For example:

- $\Omega \equiv (\lambda x.xx)(\lambda x.xx)$ is not normalizing,
- $\mathsf{KI}\Omega$ is normalizing, but not strongly normalizing,
- KIS is strongly normalizing.

All terms typable in the simply typed lambda calculus $\lambda \rightarrow$ are strongly normalizing. This was first proved in [13]. The same property for the polymorphic lambda calculus $\lambda 2$ (system F) was proved by Girard [7]. This proof cannot be performed within second-order Peano arithmetic. A general approach towards the proof of strong normalization for both $\lambda \rightarrow$ and $\lambda 2$ is given in [1]. It is on the lines of the proofs in [7] and Tait [14]. We shall use this method in order to prove strong normalization for the intersection type systems $\mathcal{D}$ and $\mathcal{D}_{\leq}$.

First of all, let us recall some notions and notation given in [1]. The set of all strongly normalizing lambda terms is denoted by SN. If A and B are sets of lambda terms, then

$$A \rightarrow B =_{def} \{M \in \Lambda | \forall N \in A \;\; MN \in B\}.$$

The *interpretation of types* $\| \; \| : T \rightarrow \mathcal{P}(\Lambda)$ is defined inductively.

Definition 3.1

1. $\|\alpha\| = SN$, for all type variables α;
2. $\|\varphi \rightarrow \psi\| = \|\varphi\| \rightarrow \|\psi\|$, for all types φ and ψ ;
3. $\|\varphi \cap \psi\| = \|\varphi\| \cap \|\psi\|$, for all types φ and ψ.

Let us notice that for each type $\varphi, \psi \in T$ we have that $\|\varphi\| \subseteq SN$, and if $\varphi \leq \psi$, then $\|\varphi\| \subseteq \|\psi\|$ (see [1]). The notion of a *saturated set* of lambda terms is defined inductively as follows.

Definition 3.2 A set $X \subseteq \Lambda$ is called *saturated*, $X \in SAT$, if

1. $(\forall n \geq 0) \;\; (\forall M_1, \ldots, M_n \in SN) \;\; xM_1 \ldots M_n \in X$, where x is any term variable;
2. $(\forall n \geq 0)(\forall M_1, \ldots, M_n \in SN)(\forall N \in SN)$
   $M[N/x]M_1 \ldots M_n \in X \Rightarrow (\lambda x.M)NM_1 \ldots M_n \in X$.

Proposition 3.3

1. *SN is saturated.*
2. *If A and B are saturated, then $A \rightarrow B$ is saturated.*
3. *If A and B are saturated, then $A \cap B$ is saturated.*
4. *$\|\varphi\|$ is saturated for all $\varphi \in T$.*

Proposition 3.3.3 (Lemma 4.3.3 in [1]) is stated for an infinite intersection, hence it holds for a finite intersection as well, which will be needed here.

Let ρ be a mapping of term variables into lambda terms, i.e., $\rho : \mathrm{V} \rightarrow \Lambda$. Then the *valuation of terms* $\| \; \|_\rho : \Lambda \rightarrow \Lambda$ is defined inductively.

Definition 3.4

1. $\|x\|_\rho = \rho(x)$, for all term variables x;
2. $\|M\|_\rho = M[\rho(x_1)/x_1, \ldots, \rho(x_n)/x_n]$, where $FV(M) = \{x_1, \ldots, x_n\}$.

The relation of satisfaction, $\models$, is defined in terms of the type interpretation $\| \| : T \to \mathcal{P}(\Lambda)$ and the term valuation $\| \|_\rho : \Lambda \to \Lambda$ induced by a map $\rho : \mathrm{V} \to \Lambda$.

Definition 3.5

1. ρ satisfies $M : \sigma$, notation $\rho \models M : \sigma$, if $\|M\|_\rho \in \|\sigma\|$.
2. ρ satisfies a basis Γ, notation $\rho \models \Gamma$, if

$$\rho \models x : \sigma, \quad \text{for all} \quad (x : \sigma) \in \Gamma.$$

3. A basis Γ satisfies $M : \varphi$, notation $\Gamma \models M : \sigma$, if

$$\forall \rho (\rho \models \Gamma \Rightarrow \rho \models M : \sigma).$$

Now, the proof of soundness of $\lambda \to$ given in [1] can be extended to the proof of soundness of $\mathcal{D}$ and $\mathcal{D}_{\leq}$.

Proposition 3.6 (Soundness of $\mathcal{D}$ and $\mathcal{D}_{\leq}$) *If* $\Gamma \vdash_{(\leq)} M : \sigma$, *then* $\Gamma \models M : \sigma$.

Proof: By induction on the derivation of $\Gamma \vdash M : \sigma$. We shall point out only the essential cases for $\mathcal{D}$.

Case $(\cap E)$*:* The last rule applied is $(\cap E)$, i.e.,

$$\frac{\Gamma \vdash M : \varphi \cap \psi}{\Gamma \vdash M : \varphi (M : \psi)}.$$

In order to show $\Gamma \models M : \varphi$ and $\Gamma \models M : \psi$, let us take an arbitrary ρ and suppose that $\rho \models \Gamma$. Then by the induction hypothesis,

$$\rho \models M : \varphi \cap \psi \ \text{, i.e.} \ \|M\|_\rho \in \|\varphi \cap \psi\|.$$

Thus $\|M\|_\rho \in \|\varphi\|$ and $\|M\|_\rho \in \|\psi\|$. Hence $\rho \models M : \varphi$ and $\rho \models M : \psi$.

Case $(\cap I)$*:* The last rule applied is $(\cap I)$, i.e.,

$$\frac{\Gamma \vdash M : \varphi \ \ \Gamma \vdash M : \psi}{\Gamma \vdash M : \varphi \cap \psi}.$$

If we proceed as in the previous cases and suppose that $\rho \models \Gamma$ for an arbitrary ρ, then by the induction hypothesis,

$$\rho \models M : \varphi \ \text{ and } \ \rho \models M : \psi.$$

Thus $\|M\|_\rho \in \|\varphi\|$ and $\|M\|_\rho \in \|\psi\|$. It follows that $\|M\|_\rho \in \|\varphi\| \cap \|\psi\|$, i.e., $\rho \models M : \varphi \cap \psi$.

Case $(\leq)$*:* The proof in this case is straight forward since we noticed that if $\varphi \leq \psi$, then $\|\varphi\| \subseteq \|\psi\|$. □

□

Once we have the soundness of $\mathcal{D}$ and $\mathcal{D}_{\leq}$, then in order to obtain strong normalization for $\mathcal{D}$ and $\mathcal{D}_{\leq}$, we proceed as in the case of $\lambda\rightarrow$.

Theorem 3.7 (Strong Normalization for $\mathcal{D}$ and $\mathcal{D}_{\leq}$) *If $\Gamma \vdash_{(\leq)} M : \sigma$, then M is strongly normalizing.*

Proof: Suppose $\Gamma \vdash M : \sigma$, then by Proposition 3.6 $\Gamma \models M : \sigma$, thus for each ρ,

$$\rho \models \Gamma \;\Rightarrow\; \|M\|_\rho \in \|\sigma\|.$$

Define $\rho_0(x) = x$ for all term variables x. Then we have $\rho_0 \models \Gamma$ since by Proposition 3.3 $\|\tau\|$ is saturated for each type τ, and hence $x \in \|\tau\|$. Therefore $\|M\|_{\rho_0} \in \|\sigma\|$, but $\|M\|_{\rho_0} = M$ and $\|\sigma\| \subseteq SN$. Therefore $M \in SN$. □

The proofs of Proposition 3.6 and Theorem 3.7 are in a sense dual to the proofs of the same properties presented in [9]. In the sequel we shall try to explain what this duality is about. It is necessary to introduce various type interpretations as well in order to extend the proof of soundness of $\lambda\rightarrow$ to the proof of soundness of $\lambda 2$. This is done in [1] and Girard et al. [8] by taking various mappings ξ, which map *type variables* into *saturated sets*, thereby obtaining the type interpretation $\|\ \|_\xi$ from ξ by setting

$$\|\alpha\|_\xi = \xi(\alpha) \in SAT, \quad \text{for all type variables } \alpha,$$

and proceeding as in Definition 3.1.

The same is done in the proof of soundness of $\mathcal{D}$ given in [9] (lemme d'adéquation). On the other hand, there is a fixed term (variable) valuation

$$\rho(x) = x, \quad \text{for all term variables } x,$$

and hence a fixed term valuation $\|\ \| : \Lambda \rightarrow \Lambda$, for which $\|M\| = M$. This is dual to the proof of Proposition 3.6, where we had a single type interpretation and a variety of term valuations.

Strong normalization for $\mathcal{D}$ in [9] is proved by choosing a single type (variable) interpretation

$$\xi_0(\alpha) = SN.$$

This is dual again to the choice of a single term valuation in the proof of Theorem 3.7.

This technique is generalized in Gallier [5] using a "variant of realizability argument known as reducibility." By a suitable choice of type interpretation $\|\sigma\|$ which satisfies a unary predicate $\mathcal{P}$ describing the desired property of lambda terms (e.g., $\|\alpha\| = SN$, as in our case, or $\|\alpha\| \in SAT$ as in [9]) the type characterizations of solvable and normalizing terms are given.

The strong normalization for $\mathcal{D}_{\leq}$ is not considered in [9], since the partial ordering on types is not mentioned in an explicit way. The notion of computability defined in [11] is used in [15] in order to prove the strong normalization theorem for $\mathcal{D}_{\leq}$. It seems doubtful that the strong normalization of $\mathcal{D}$ can be proved via computability. One of the reasons to believe this is the fact that η-conversion is involved in the notion of computability, whereas it is a "property" of the rule $(\leq)$ as shown in Ghilezan [6]. For the analysis of other proofs of strong normalization for intersection type systems see [15].

4 Typability of strongly normalizing terms The very essential and outstanding property of the intersection type systems $\mathcal{D}_{\leq}$ and $\mathcal{D}$ is the converse of the strong normalization property, i.e., the fact that all strongly normalizing terms are typable in $\mathcal{D}_{\leq}$ and $\mathcal{D}$. The idea that strongly normalizing lambda terms are exactly the terms typable in the intersection type assignment systems without the (ω)-rule first appeared in [11], [4], and [10]. Further, this subject is treated in [12], [9], and [15] with different approaches. Obviously, this property does not hold for $\mathcal{D}\Omega_{\leq}$, since all lambda terms are typable in it. In other words, by leaving out the rule (ω) the situation changes essentially: not only that terms typable in $\mathcal{D}$ and $\mathcal{D}_{\leq}$ are strongly normalizing, but strongly normalizing terms are exactly all terms typable in $\mathcal{D}$ and $\mathcal{D}_{\leq}$. Here we present a modified proof of this property in Theorem 4.3 and compare it with the proofs mentioned above.

In order to show that every strongly normalizing term is typable in $\mathcal{D}_{\leq}$ and $\mathcal{D}$, first it is necessary to notice that every normal form is typable in these systems. It is known that a lambda term N in normal form can be represented in the following way:

$$N \equiv \lambda x_1 \dots x_n.yN_1 \dots N_k, \quad \text{for some normal forms } N_1, \dots, N_k.$$

Also it is known that every term in normal form can be typed in $\mathcal{D}$ (see [12] and [9]).

Proposition 4.1 *If N is a normal form, then there is a basis Γ and a type σ such that*

$$\Gamma \vdash N : \sigma.$$

Proof: By induction on the construction of a normal form M. □

There is another problem that is to be overcome in order to show that every strongly normalizing term is typable in $\mathcal{D}$. It is known that $\mathcal{D}$ is closed for β-reduction, which is not closed for β-expansion, i.e., types are preserved under β-reduction, but this is not the case for β-expansion. The counterexample given in [9] takes into account the terms $\lambda y.(\lambda x.y)(yy)$ and $\lambda y.y$ since $\lambda y.(\lambda x.y)(yy) \to_\beta \lambda y.y$ and $\vdash \lambda y.y : \alpha \to \alpha$, where α is a type variable, whereas $\not\vdash \lambda y.(\lambda x.y)(yy) : \alpha \to \alpha$.

The term variable y has to have an arrow type in order to type self-application yy in $\mathcal{D}$. This problem is avoided in $\mathcal{D}\Omega$, since yy can be typed by ω. But still, the term $\lambda y.(\lambda x.y)(yy)$ is typable in $\mathcal{D}$ by $(\alpha \cap (\alpha \to \beta)) \to (\alpha \cap (\alpha \to \beta))$. Thus both of the considered terms are typable in $\mathcal{D}$, i.e., $\vdash \lambda y.y : \alpha \to \alpha$ and $\vdash \lambda y.(\lambda x.y)(yy) : (\alpha \cap (\alpha \to \beta)) \to (\alpha \cap (\alpha \to \beta))$, but $\lambda y.(\lambda x.y)(yy) : \alpha \to \alpha$ cannot be derived without the rule (ω).

The counterexample in [15] deals with the terms $\lambda yz.(\lambda x.z)(yz)$ and $\lambda yz.z$ typable in $\mathcal{D}$, since

$$\lambda yz.(\lambda x.z)(yz) \to_\beta \lambda yz.z$$

and for which $\vdash \lambda yz.z : \beta \to (\alpha \to \alpha)$ and $\vdash \lambda yz.(\lambda x.z)(yz) : (\alpha \to \beta) \to (\alpha \to \alpha)$. Again, $\lambda yz.(\lambda x.z)(yz)$ cannot be typed with $\beta \to (\alpha \to \alpha)$ without the rule (ω) because y has to be of an arrow type in order to type the application yz in $\mathcal{D}$.

Since each normal form is typable in $\mathcal{D}$ we need some property that "pastes together" the steps of β-reduction in a reduction tree with the steps in the type assignment. This is done by the following statement proved in [9] and [15].

Proposition 4.2 *Let $M \to_\beta N$ by contracting the β-redex $(\lambda x.P)Q$. Then if $\Gamma \vdash N : \sigma$ and Q is typable in the same basis Γ, it follows that $\Gamma \vdash M : \sigma$.*

Proof: By induction on the construction of M and length of σ. □

The main point in Proposition 4.2 is the fact that Q is typable in $\mathcal{D}$ in the given basis. In the previous counterexamples subterms yy and yz were not typable in $\mathcal{D}$ in the given bases. That is the reason why those β-expansions do not preserve types. This property trivially holds for $\mathcal{D}\Omega$ since Q is typable in any basis by ω.

Theorem 4.3 (Converse of Strong Normalization Theorem) *If M is strongly normalizing, then there is a basis Γ and a type σ such that*

$$\Gamma \vdash M : \sigma.$$

Proof: By induction on the size of reduction of M, i.e., the sum of the lengths of all possible reductions of M, as defined in [9] notation $n(M)$.

If $n(M) = 0$, then M is a normal form. Hence by Proposition 4.1 there is a basis Γ and a type σ such that $\Gamma \vdash M : \sigma$.

Otherwise let $(\lambda x.P)Q$ be a redex in M. Let N be the result of contracting this redex in M. Since $n(Q) < n(M)$ and $n(N) < n(M)$ by the induction hypothesis Q and N can be typed in $\mathcal{D}$, i.e.,

$$\Gamma' \vdash Q : \sigma \text{ and } \Gamma'' \vdash N : \tau.$$

Now Q and N can be typed in $\mathcal{D}$ in the same basis Γ which is obtained from the bases Γ' and Γ'' in the following way: $x : \sigma \in \Gamma$ if $x : \sigma \in \Gamma'$ or $x : \sigma \in \Gamma''$ or $\sigma \equiv \sigma' \cap \sigma''$ and $x : \sigma' \in \Gamma'$, $x : \sigma'' \in \Gamma''$. Hence by Proposition 4.2 M can be typed in $\mathcal{D}$. □

This strong normalization property is proved in [15] by using "the inside-out reduction" of lambda terms. The inductive argument in [9] is the sum of all possible reductions of a term, using the property that each term can be expressed in the form $\lambda x_1 \ldots x_n.NM_1 \ldots M_k$, where N is a variable or a redex. For the analysis of other proofs of this property see [15].

A consequence of Theorems 3.7 and 4.3 is that strongly normalizing terms are exactly the terms typable in $\mathcal{D}$. The systems $\mathcal{D}$ and $\mathcal{D}_\leq$ are equivalent from the point of view of typability, as shown in [6], so the same property holds for the system $\mathcal{D}_\leq$.

Corollary 4.4 *Let M be a lambda term. There are Γ and σ such that $\Gamma \vdash_{(\leq)} M : \sigma$ if and only if M is strongly normalizing.*

A consequence of this equivalence is the undecidability of the typability in the systems $\mathcal{D}$ and $\mathcal{D}_\leq$, since the set of strongly normalizing terms, SN, is not recursive.

REFERENCES

[1] Barendregt, H. P., "Lambda calculi with types," pp. 117–309 in *Handbook of Logic in Computer Science*, vol. II, edited by S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, Oxford University Press, Oxford, 1992. MR 97m:03026 1, 2, 2, 2, 3, 3, 3, 3, 3, 3

[2] Barendregt, H. P., M. Coppo, and M. Dezani–Ciancaglini, "A filter model and the completeness of type assignment," *The Journal of Symbolic Logic*, vol. 48 (1983), pp. 931–940. Zbl 0545.03004 MR 85j:03014 1, 2, 2

[3] Coppo, M., M. Dezani-Ciancaglini, and B. Venneri, "Principal type schemes and λ-calculus semantics," pp. 480–490 in *To H. B. Curry: Essays on Combinatory Logic, Typed Lambda Calculus and Formalism*, edited by J. R. Hindley and J. P. Seldin, Academic Press, London, 1980. MR 82d:03022 1, 1, 2, 2

[4] Coppo, M., M. Dezani-Ciancaglini, and B. Venneri, "Functional characters of solvable terms," *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, vol. 27 (1981), pp. 45–58. Zbl 0479.03006 MR 82j:03011 1, 1, 1, 2, 2, 4

[5] Gallier, J. H., "Reducibility strikes again I," *Technical report* (1993), University of Pennsylvania, Philadelphia. 3

[6] Ghilezan, S., "Typability and inhabitation in intersection type assignment systems," forthcoming. 3, 4

[7] Girard, J.-Y., *Interprétation fonctionelle et élimination des coupures dans l'arithmétique d'ordre supérieur*, Ph. D. Thesis, Université Paris VII, 1972. 3, 3

[8] Girard, J.-Y., Y. Lafont, and P. Taylor, *Proofs and Types*, Tracts in Theoretical Computer Science 7, Cambridge University Press, Cambridge, 1990. Zbl 0671.68002 MR 90g:03013 3

[9] Krivine, J. L., *Lambda-calcul, types et modèles*, Masson, Paris, 1990. Zbl 0697.03004 1, 1, 2, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 4

[10] Leivant, D., "Polymorphic type inference," pp. 88–98 in *Proceedings 10th ACM Symposium on Principles of Programming Languages*, Austin, Texas, 1983. 1, 4

[11] Pottinger, G., "A type assignment for the strongly normalizable λ-terms," pp. 561–577 in *To H. B. Curry: Essays on Combinatory Logic, Typed Lambda Calculus and Formalism*, edited by J. R. Hindley and J. P. Seldin, Academic Press, London, 1980. MR 82j:03014 1, 3, 4

[12] Ronchi della Rocca, S., and B. Venneri, "Principal type schemes for an extended type theory," *Theoretical Computer Science*, vol. 28 (1984), pp. 151–169. Zbl 0535.03007 MR 86d:03017 1, 4, 4

[13] Tait, W. W., "Intensional interpretation of functionals of finite type I," *The Journal of Symbolic Logic*, vol. 32 (1967), pp. 198–212. MR 36:2483 3, 3

[14] Tait, W. W., "A realizability interpretation of the theory of species," pp. 240–251 in *Logic Colloquium (Boston)*, edited by R. Parikh, Lecture Notes in Mathematics 453, Springer-Verlag, Berlin, 1975. Zbl 0328.02014 MR 52:5380 3

[15] van Bakel, S., "Complete restrictions of the intersection type discipline," *Theoretical Computer Science*, vol. 102 (1992), pp. 136–163. Zbl 0762.03006 MR 94b:03025 1, 1, 3, 3, 4, 4, 4, 4, 4

Faculty of Engineering
University of Novi Sad
Trg Dositeja Obradovića 6
21000 Novi Sad
Yugoslavia
email: gsilvia@uns.ns.ac.yu