Theoretical Computer Science 100 (1992) 303-324
Elsevier

Intersection types for combinatory logic*

Mariangiola Dezani-Ciancaglini
Dipartimento di Informatica, Corso Svizzera 185, 10149 Torino, Italy

J. Roger Hindley
Mathematics Division, University College, Swansea SA2 8PP, UK

Communicated by A.R. Meyer
Received June 1989
Revised February 1991

Abstract

Dezani-Ciancaglini, M. and J. R. Hindley, Intersection types for combinatory logic, Theoretical Computer Science 100 (1992) 303-324.

Two different translations of the usual formulation of intersection types for λ-calculus into combinatory logic are proposed; in the first one the rule ($\leqslant$) is unchanged, while in the second one the rule ($\leqslant$) is replaced by three new rules and four axiom-schemes, which seem to be simpler than rule ($\leqslant$) itself.

Introduction

Intersection types were first introduced around 1977 as a generalization of the type discipline of Church and Curry, mainly with the aim of describing the functional behaviour of all solvable λ-terms. The usual $\rightarrow$-based type-language was extended by adding a constant ω as a universal type and a new connective $\wedge$ for the intersection of two types. With suitable axioms and rules to assign types to λ-terms, this gave a system in which (i) the set of types given to a λ-term does not change under β-conversion, and (ii) the sets of normalizing and solvable λ-terms can be characterized very neatly by the types of their members. (An introduction and motivation of $\wedge$ and ω is given in [4], another in [12], and all the most important basic properties of the system's syntax are given in [2].)

Moreover, in the new type-language we can build λ-models in which the interpretation of a λ-term coincides with the set of all types that can be assigned to it. These are called *filter models*. Filter models turn out to be a very rich class containing in

* Research partially supported by Italian M.P.I. Fondi 40% Comitato per la Matematica.

particular each inverse-limit space, and have been widely used to study properties of D_∞-λ-models; see [2, 3, 5].

More recently, intersection types have been introduced in the programming language Forsythe, which is a descendent of Algol 60, to simplify the structure of types; see Reynolds' report [18].

Systems of combinators are designed to perform the same tasks as systems of λ-calculus, but without using bound variables. Curry's type discipline has turned out to be significantly simpler in combinatory logic than in λ-calculus. (For an introduction see [13, Chapter 14].) Is the same true for intersection types?

We shall propose here two different formulations of intersection types for combinatory logic. They will be given for an arbitrary complete combinator basis (as defined in Section 2 below) and illustrated by examples using the most common abstraction algorithms. Conditions on bases and abstraction algorithms will be introduced in Section 3 which will imply that these formulations are exact translations of the λ-system in [2].

However, there are extra complications in combinatory logic. In the case of λ-calculus, the type-assignment rule $(\leqslant)$ in [2] is well known to be replaceable by the simpler rule (η) (Section 1 below). But in combinatory logic some more care must be taken in choosing a rule to replace $(\leqslant)$, and we do not know whether the alternative system presented below is the simplest possible (see Section 4).

A first version of the present paper was included in the collection [15] dedicated to J.W. de Bakker in honour of his 25 years of work in semantics.

1. λ-Calculus

To prepare for the combinatory system we first summarize the intersection type-assignment system for λ-calculus, following [2]. We use the λ-calculus notation of [13, Chapter 1].

Definition 1.1 (i) The set $\boldsymbol{T}$ of *intersection types* $(\rho, \sigma, \tau, \zeta, \ldots)$ is inductively defined by

$$\phi_0, \phi_1, \ldots \in \boldsymbol{T} \quad \text{(type-variables)},$$

$$\omega \in \boldsymbol{T} \quad \text{(one type-constant)},$$

$$\sigma, \tau \in \boldsymbol{T} \Rightarrow (\sigma \to \tau) \in \boldsymbol{T}, (\sigma \wedge \tau) \in \boldsymbol{T}.$$

(ii) A (*type-assignment*) *statement* is of the form $M{:}\sigma$ with $\sigma \in \boldsymbol{T}$ and M a λ-term, called its *subject.*

Definition 1.2. A (*type-assignment*) *basis* $\boldsymbol{B}$ is a set of statements with only variables as subjects. If x does not occur in $\boldsymbol{B}$, then "$\boldsymbol{B}, x{:}\sigma$" denotes $\boldsymbol{B} \cup \{x{:}\sigma\}$.

Note. We do not assume that the subjects in a basis are distinct.

Notation. To avoid parentheses we assume that "$\wedge$" takes precedence over "$\rightarrow$" and that "$\rightarrow$" associates to the right.

Next, we define a pre-order relation $\leqslant$ on intersection types which formalizes the subset relation and will be used in a type-assignment rule.

Definition 1.3. The $\leqslant$ *relation* on intersection types is inductively defined by:

$$\tau \leqslant \tau, \qquad \tau \leqslant \tau \wedge \tau,$$

$$\tau \leqslant \omega, \qquad \sigma \wedge \tau \leqslant \sigma, \quad \sigma \wedge \tau \leqslant \tau,$$

$$\omega \leqslant \omega \rightarrow \omega, \qquad (\sigma \rightarrow \rho) \wedge (\sigma \rightarrow \tau) \leqslant \sigma \rightarrow \rho \wedge \tau,$$

$$\sigma \leqslant \rho \leqslant \tau \Rightarrow \sigma \leqslant \tau,$$

$$\sigma \leqslant \sigma', \tau \leqslant \tau' \Rightarrow \sigma \wedge \tau \leqslant \sigma' \wedge \tau',$$

$$\sigma \leqslant \sigma', \tau \leqslant \tau' \Rightarrow \sigma' \rightarrow \tau \leqslant \sigma \rightarrow \tau'.$$

Definition 1.4 (*The intersection type system* $\mathrm{TA}_\lambda(\wedge, \omega, \leqslant)$).

(i) $\mathrm{TA}_\lambda(\wedge, \omega, \leqslant)$ is the type assignment system defined by the following natural-deduction rules and axioms.

Axioms: (ω) $M{:}\omega$ (one axiom for each λ-term M).

Rules:

$$(\rightarrow\mathrm{I}) \ \frac{\begin{array}{c}[x{:}\sigma] \\ \vdots \\ M{:}\tau\end{array}}{\lambda x.M{:}\sigma \rightarrow \tau}\,(*) \qquad (\rightarrow\mathrm{E}) \ \frac{M{:}\sigma \rightarrow \tau \quad N{:}\sigma}{MN{:}\tau}$$

$$(\wedge\mathrm{I}) \ \frac{M{:}\sigma \quad M{:}\tau}{M{:}\sigma \wedge \tau} \qquad (\wedge\mathrm{E}) \ \frac{M{:}\sigma \wedge \tau}{M{:}\sigma} \quad \frac{M{:}\sigma \wedge \tau}{M{:}\tau}$$

$$(\leqslant) \ \frac{M{:}\sigma \quad \sigma \leqslant \tau}{M{:}\tau}$$

(*) rule ($\rightarrow$I) may only be used when x is not free in assumptions above $M{:}\tau$ other than $x{:}\sigma$.

(ii) We write $\boldsymbol{B} \vdash_\lambda M{:}\sigma$ iff $M{:}\sigma$ is derivable from the basis $\boldsymbol{B}$ in this system.

Theorem 1.5 (*Type-preservation by β-equality and η-reduction*). *In* $\mathrm{TA}_\lambda(\wedge, \omega, \leqslant)$:

(i) *type-assignment statements are preserved by β-equality, that is, if $\boldsymbol{B} \vdash_\lambda M{:}\tau$ and $M =_\beta N$, then $\boldsymbol{B} \vdash_\lambda N{:}\tau$;*

(ii) *type-assignment statements are preserved by η-reduction, that is, if $x \notin \mathrm{FV}(M)$ and $\boldsymbol{B} \vdash_\lambda (\lambda x.Mx){:}\tau$ then $\boldsymbol{B} \vdash_\lambda M{:}\tau$.*

Proof. (i) [2, Corollary 3.8].

(ii) Induction on the deduction of $(\lambda x.Mx):\tau$. The only nontrivial case is when the last step in this deduction is $(\to\mathrm{I})$; then $\tau \equiv \rho\to\sigma$ and x does not occur in $\boldsymbol{B}$ and

$$\boldsymbol{B}, x:\rho \vdash_\lambda Mx:\sigma.$$

Then $\boldsymbol{B}\vdash_\lambda M:\rho\to\sigma$ by the η-lemma in [11, Section 5]. □

Interestingly, the above theorem can be strengthened; there is an equality relation that characterizes the invariance of types in an exact way, namely equality in the model $P\omega$. Define

$$M =_{P\omega} N \Leftrightarrow P\omega \models M = N.$$

Then the following characterization theorem holds; we shall use it later.

Theorem 1.6 (*Characterization of type preservation*). *In* $\mathrm{TA}_\lambda(\wedge, \omega, \leqslant)$:

$$M =_{P\omega} N \Leftrightarrow (\textit{for all } \boldsymbol{B}, \tau)\{\boldsymbol{B}\vdash_\lambda M:\tau \Leftrightarrow \boldsymbol{B}\vdash_\lambda N:\tau\};$$

in particular two closed terms are $P\omega$-equal iff they have the same set of intersection types.

Proof. By [19, Theorem 4], we have

$$(\text{for all } \boldsymbol{B}, \tau)\{\boldsymbol{B}\vdash_\lambda M:\tau \Leftrightarrow \boldsymbol{B}\vdash_\lambda N:\tau\} \Leftrightarrow \mathscr{A}(M) = \mathscr{A}(N),$$

where $\mathscr{A}(M)$ is the set of all approximate normal forms of M ([24, Section 5], or [1, Definition 14.3.5], or [19, Definition 8] where they are called approximants of M). By [1, Section 14.3], we have

$$\mathscr{A}(M) = \mathscr{A}(N) \Leftrightarrow \mathrm{BT}(M) = \mathrm{BT}(N),$$

where $\mathrm{BT}(M)$ is defined in [1, Chapter 10]. Then by [1, Theorem 19.1.19],

$$\mathrm{BT}(M) = \mathrm{BT}(N) \Leftrightarrow P_\omega \models M = N. \qquad \square$$

The property of invariance under η-reduction in Theorem 1.5 suggests that rule $(\leqslant)$ can be replaced by an η-rule, as follows.

Definition 1.7 (*Type-assignment with an η-rule*).

(i) Let $\mathrm{TA}_\lambda(\wedge, \omega, \eta)$ be the system obtained from $\mathrm{TA}_\lambda(\wedge, \omega, \leqslant)$ by replacing rule $(\leqslant)$ by

$$(\eta)\ \frac{(\lambda x.Mx):\tau}{M:\tau} \quad (\text{if } x \text{ is not free in } M).$$

(ii) Let $\vdash_{\lambda\eta}$ denote derivability in the resulting system.

Theorem 1.8. $\mathrm{TA}_\lambda(\wedge, \omega, \leqslant)$ *and* $\mathrm{TA}_\lambda(\wedge, \omega, \eta)$ *are equivalent*; *that is*,

$$\boldsymbol{B}\vdash_\lambda M:\sigma \Leftrightarrow \boldsymbol{B}\vdash_{\lambda\eta} M:\sigma.$$

Proof. A direct proof is fairly easy, or one can use [2] (in particular Lemma 4.2, Remark 2.10, and the remark just before 4.3). □

Another important property is that, given a closed λ-term A in approximate normal form ([24, Section 5], [19, Definition 8] or [20, Definition 2.8]), we can always build a principal type Π, depending on A, from which all derivable types can be obtained, i.e. $\vdash_\lambda A{:}\sigma$ implies that there is a chain c of substitutions, expansions and raises such that $\sigma = c(\Pi)$. We shall not give here the definition of expansions and raises: the interested reader can find definitions and proofs in [20, Sections 3-4]. We shall give here only the definition of principal type (and of principal basis, since subterms may be open) by induction on the definition of approximate normal form.

Definition 1.9. Let A be an approximate normal form ([24, Section 5], [19, Definition 8] or [20, Definition 2.8]). The *principal basis* $\boldsymbol{B}$ and the *principal type* Π of A are defined as follows (see [20, Definition 4.1]). They are unique modulo alphabetic variation (the relation $\approx$ in [20, Definition 3.3]).

(i) If $A \equiv \Omega$ then $\boldsymbol{B}$ is empty and $\Pi \equiv \omega$;

(ii) If $A \equiv x$ then $\boldsymbol{B} = \{x{:}\varphi\}$ and $\Pi \equiv \varphi$, where φ is a type variable;

(iii) If $A \equiv \lambda x.A'$ and $\boldsymbol{B}'$, Π' are respectively the principal basis and the principal type of A', then:

(1) if $\boldsymbol{B}'$ does not contain premises whose subject is x then

$$\boldsymbol{B} = \boldsymbol{B}', \qquad \Pi \equiv \varphi \to \Pi',$$

where φ is a type variable not occurring in $\boldsymbol{B}'$ and Π';

(2) if $x{:}\sigma_1, \ldots, x{:}\sigma_n$ are all the premises in $\boldsymbol{B}'$ whose subject is x, then

$$\boldsymbol{B} = \boldsymbol{B}' - \{x{:}\sigma_1, \ldots, x{:}\sigma_n\}, \qquad \Pi \equiv \sigma_1 \wedge \cdots \wedge \sigma_n \to \Pi';$$

(iv) if $A \equiv xA_1 \ldots A_n$ and $\boldsymbol{B}_i$, Π_i are the principal bases and the principal types of A_i for $1 \leqslant i \leqslant n$ (we choose alphabetic variants of them such that the same type variable does not occur in two principal bases and types), then

$$\boldsymbol{B} = \bigcup_{1 \leqslant i \leqslant n} \boldsymbol{B}_i \cup \{x{:}\, \Pi_1 \to \cdots \to \Pi_n \to \varphi\}, \qquad \Pi \equiv \varphi,$$

where φ is a type variable which does not occur in $\boldsymbol{B}_i$ and Π_i $(1 \leqslant i \leqslant n)$.

Exercise 1.10. The names and principal types of some λ-terms that will be used later are as follows. (Not to be confused with principal types in the Curry system!)

$\mathbf{S}_\lambda \equiv \lambda xyz.xz(yz)$: $\quad (\varphi \to \varphi_1 \to \varphi_2) \to (\varphi' \to \varphi_1) \to \varphi \wedge \varphi' \to \varphi_2$,

$\mathbf{S}^*_\lambda \equiv \lambda wxyz.w(xz)(yz)$: $\quad (\varphi_1 \to \varphi_2 \to \varphi_3) \to (\varphi \to \varphi_1) \to (\varphi' \to \varphi_2) \to \varphi \wedge \varphi' \to \varphi_3$,

$\mathbf{K}_\lambda \equiv \lambda xy.x$: $\quad \varphi \to \varphi_1 \to \varphi$,

$\mathbf{I}_\lambda \equiv \lambda x.x$: $\quad \varphi \to \varphi$,

$\mathbf{B}_\lambda \equiv \lambda xyz.x(yz)$: $\quad (\varphi \to \varphi_1) \to (\varphi_2 \to \varphi) \to \varphi_2 \to \varphi_1$,

$\mathbf{B}'_\lambda \equiv \lambda xyz.y(xz)$: $(\varphi \to \varphi_1) \to (\varphi_1 \to \varphi_2) \to \varphi \to \varphi_2$,

$\mathbf{B}^*_\lambda \equiv \lambda wxyz.wx(yz)$: $(\varphi \to \varphi_1 \to \varphi_2) \to \varphi \to (\varphi_3 \to \varphi_1) \to \varphi_3 \to \varphi_2$,

$\mathbf{C}_\lambda \equiv \lambda xyz.xzy$: $(\varphi \to \varphi_1 \to \varphi_2) \to \varphi_1 \to \varphi \to \varphi_2$,

$\mathbf{C}^*_\lambda \equiv \lambda wxyz.w(xz)y$: $(\varphi \to \varphi_1 \to \varphi_2) \to (\varphi_3 \to \varphi) \to \varphi_1 \to \varphi_3 \to \varphi_2$,

$\mathbf{W}_\lambda \equiv \lambda xy.xyy$: $(\varphi \to \varphi' \to \varphi_1) \to \varphi \wedge \varphi' \to \varphi_1$.

2. Correspondence between λ and CL

This section outlines the known properties of the standard transformations from combinatory logic (CL) to λ-calculus and vice versa. The account is independent of the choice of basic combinators, but apart from this the results are not new, except for Lemma 2.16.

The reader is assumed to know the main definitions in combinatory logic ([1, Chapter 7] or [13, Chapter 2]).

Assumptions and notation 2.1. We assume that a finite or infinite *combinator basis* $\mathscr{B} = \{C_1, C_2, \ldots\}$ has been given. CL$\mathscr{B}$*-terms* are built from variables and $C_1, C_2, \ldots$ by application as usual, and CL$\mathscr{B}$-terms without variables are called CL$\mathscr{B}$*-combinators.* Each C_i is called an *atomic combinator* and is assumed to have an *axiom scheme for reduction*

(i) $\quad C_i x_1 \ldots x_{n_i} \rhd D_i \quad (n_i \geqslant 1)$,

where $x_1, \ldots, x_{n_i}$ are distinct and D_i is a combination of some or all of $x_1, \ldots, x_{n_i}$ and no other atoms. That is, C_i is a proper combinator [7, p. 161].

Weak reduction $\rhd_w$ and *weak equality* $=_w$ are defined as usual by replacements with form

(ii) $\quad C_i X_1 \ldots X_{n_i} \rhd_w [X_1/x_1, \ldots, X_{n_i}/x_{n_i}]D_i$.

Note 2.2. Although $\rhd_w$ is more general than the usual reduction in [1] and [13], it satisfies the Church–Rosser theorem and other main theorems (see the table on p. 270 and footnote 4a on p. 269 in [10]). But we shall not need these theorems here; all we shall need is that

(i) $\quad X \rhd_w Y \Rightarrow [U/v]X \rhd_w [U/v]Y$.

Definition 2.3. A combinator basis $\mathscr{B}$ is *complete* iff for each sequence of distinct variables $x_1, \ldots, x_n$ and each CL$\mathscr{B}$-term Y there exists a CL$\mathscr{B}$-term $\lambda^* x_1 \ldots x_n.Y$ containing none of $x_1, \ldots, x_n$, such that

(i) $\quad (\lambda^* x_1 \ldots x_n.Y)x_1 \ldots x_n \rhd_w Y$.

An *abstraction algorithm* is any algorithm that computes such a $\lambda^* x_1 \ldots x_n.Y$ for all $x_1, \ldots x_n$, Y. It is *multi-sweep* iff it procedes by first giving an algorithm to compute $\lambda\, x.Y$ for all x and Y, and then by applying this algorithm repeatedly to compute

(ii) $\quad \lambda^* x_1. \ldots x_n.Y \equiv \lambda^* x_1.(\lambda^* x_2. \ldots (\lambda^* x_n.Y) \ldots).$

Note 2.4. Even though the definition of completeness does not require that $\lambda\, x_1 \ldots x_n.Y$ be computed by an algorithm, an abstraction algorithm always exists if $\mathscr{B}$ is complete. (One will be given in the proof of Lemma 2.16.)

In the last few years there has been a new interest in different abstraction algorithms, since they can be used in compiling programs written in functional languages. Some algorithms abstract one variable at a time (*multi-sweep algorithms* as defined above) and others abstract all variables simultaneously (*one-sweep algorithms*).

We shall not consider one-sweep algorithms here. But it is interesting to remark that Curry's first abstraction algorithm in 1930 [6] was one-sweep, and in [17] a modification of this algorithm has been proposed which has many interesting features (see also [8]).

Because of the interest in different algorithms we now describe some of the multi-sweep algorithms which have occurred most frequently in the literature.

Example 2.5. The complete bases below are subbases of the basis $\mathscr{B}_0 = \{\mathbf{S}, \mathbf{K}, \mathbf{I}, \mathbf{B}, \mathbf{C}, \mathbf{S}^*, \mathbf{B}^*, \mathbf{C}^*, \mathbf{W}\}$. These combinators are assumed to come with the following axiom schemes for reduction:

$$\mathbf{S}xyz \rhd_w xz(yz) \qquad \mathbf{K}xy \rhd_w x \qquad \mathbf{I}x \rhd_w x$$

$$\mathbf{B}xyz \rhd_w x(yz) \qquad \mathbf{C}xyz \rhd_w xzy \qquad \mathbf{S}^*wxyz \rhd_w w(xz)(yz)$$

$$\mathbf{B}^*wxyz \rhd_w wx(yz) \qquad \mathbf{C}^*wxyz \rhd_w w(xz)y \qquad \mathbf{W}xy \rhd_w xyy.$$

In more detail: we shall consider $\mathscr{B}_0$ and the bases

$$\mathscr{B}_1 = \{\mathbf{S}, \mathbf{K}, \mathbf{I}\},$$

$$\mathscr{B}_2 = \mathscr{B}_1 \cup \{\mathbf{B}\} = \{\mathbf{S}, \mathbf{K}, \mathbf{I}, \mathbf{B}\},$$

$$\mathscr{B}_3 = \mathscr{B}_2 \cup \{\mathbf{C}\} = \{\mathbf{S}, \mathbf{K}, \mathbf{I}, \mathbf{B}, \mathbf{C}\},$$

$$\mathscr{B}_4 = \mathscr{B}_3 \cup \{\mathbf{S}^*, \mathbf{B}^*, \mathbf{C}^*\} = \{\mathbf{S}, \mathbf{K}, \mathbf{I}, \mathbf{B}, \mathbf{C}, \mathbf{S}^*, \mathbf{B}^*, \mathbf{C}^*\}.$$

The basis will not be written explicitly in each algorithm below, since we shall assume that the current basis is the set of atomic combinators which appear on the right hand side of the identities which define the abstraction algorithm. (Clearly one could choose other bases for the same abstraction algorithms and even always the basis $\{\mathbf{S}, \mathbf{K}\}$.)

We present eight alternative algorithms for $\lambda^* x.Y$, with notes on their origins later. In each algorithm the clauses must be used in the order of precedence shown;

for example in λ^η, to compute $\lambda^\eta x.UV$ one may only use clause (f) when (a), (b) and (c) do not apply.

λ^η: (a) $\lambda^\eta x.Y \equiv \mathbf{K}Y$ if $x \notin \mathrm{FV}(Y)$,
(b) $\lambda^\eta x.x \equiv \mathbf{I}$,
(c) $\lambda^\eta x.Ux \equiv U$ if $x \notin \mathrm{FV}(U)$,
(f) $\lambda^\eta x.UV \equiv \mathbf{S}(\lambda^\eta x.U)(\lambda^\eta x.V)$.

λ^{abf}: (a), (b), (f) as above.

λ^{fab}: (f) $\lambda^{\mathrm{fab}} x.UV \equiv \mathbf{S}(\lambda^{\mathrm{fab}} x.U)(\lambda^{\mathrm{fab}} x.V)$,
(a*) $\lambda^{\mathrm{fab}} x.y \equiv \mathbf{K}y$ if y is an atom and $y \not\equiv x$,
(b) $\lambda^{\mathrm{fab}} x.x \equiv \mathbf{I}$.

λ^{abcdf}: (a), (b), (c) as above,
(d) $\lambda^{\mathrm{abcdf}} x.UV \equiv \mathbf{B}U(\lambda^{\mathrm{abcdf}} x.V)$ if $x \notin \mathrm{FV}(U)$,
(f) as above.

λ^{S}: (a), (b), (c), (d) as above,
(e) $\lambda^{\mathrm{S}} x.UV \equiv \mathbf{C}(\lambda^{\mathrm{S}} x.U)V$ if $x \notin \mathrm{FV}(V)$,
(f) as above.

λ^β: (a), (b) as above,
(c_β) $\lambda^\beta x.Ux \equiv U$ if $x \notin \mathrm{FV}(U)$ and U is functional
(i.e. U has form $\mathbf{S}VW$, $\mathbf{S}V$, $\mathbf{S}$, $\mathbf{K}V$, $\mathbf{K}$ or $\mathbf{I}$),
(f_β) $\lambda^\beta x.UV \equiv \mathbf{S}(\lambda^\eta x.U)(\lambda^\eta x.V)$ (yes, we mean λ^η!).

λ^{T1}: (a), (b), (c) as above,
(d*) $\lambda^{\mathrm{T1}} x.UVZ \equiv \mathbf{B}^* UV(\lambda^{\mathrm{T1}} x.Z)$ if $x \notin \mathrm{FV}(UV)$
(e*) $\lambda^{\mathrm{T1}} x.UVZ \equiv \mathbf{C}^* U(\lambda^{\mathrm{T1}} x.V)Z$ if $x \notin \mathrm{FV}(UZ)$
(f*) $\lambda^{\mathrm{T1}} x.UVZ \equiv \mathbf{S}^* U(\lambda^{\mathrm{T1}} x.V)(\lambda^{\mathrm{T1}} x.Z)$ if $x \notin \mathrm{FV}(U)$
(d), (e), (f) as above.

λ^{T2}: as λ^{T1} but inserting between (c) and (d*):
(g) $\lambda^{\mathrm{T2}} x.Ux \equiv \mathbf{W}(\lambda^{\mathrm{T2}} x.U)$.

Note 2.6. The first five algorithms were described by Curry in [7, Section 6A]. λ^{fab} is the simplest of these but produces the longest results. λ^{S} is the oldest one, being due to Schönfinkel [21] and dating from 1924.

The sixth, λ^β, was discussed in [13, Sections 9.34 and 9.35]; it also originated with Curry.

The last two, λ^{T1} and λ^{T2}, have been proposed by Turner respectively in [22] and [23], though the above way of writing the rules for $\mathbf{B}^*$, $\mathbf{C}^*$ and $\mathbf{S}^*$ is from [14].

Mulder in [16] compares the length of the translations obtained using many different abstraction algorithms, and it turns out that Turner's algorithms seem to work well in practice.

Definition 2.7. (*From* CL *to* λ*-calculus: the* λ*-transformation*). To each CL$\mathscr{B}$-term X we associate a λ-term X_λ thus:

$$(C_i)_\lambda \equiv \lambda x_1 \ldots x_{n_i}.D_i,$$

$$x_\lambda = x,$$

$$(XY)_\lambda \equiv X_\lambda Y_\lambda.$$

Lemma 2.8. *If* $X \rhd_w Y$ *then* $X_\lambda \rhd_\beta Y_\lambda$.

Proof. Consider a contraction of form 2.1(ii); by the definition of $(C_i)_\lambda$ and the fact that D_i is only a combination of variables, one has

$$\begin{aligned}(C_iX_1 \ldots X_{n_i})_\lambda &\equiv (\lambda x_1 \ldots x_{n_i}.D_i)(X_1)_\lambda \ldots (X_{n_i})_\lambda \\ &\rhd_\beta [(X_1)_\lambda / x_1, \ldots, (X_{n_i})_\lambda / x_{n_i}]D_i \\ &\equiv ([X_1/x_1, \ldots, X_{n_i}/x_{n_i}]D_i)_\lambda. \qquad \square\end{aligned}$$

Using the λ-transformation one can carry some key notions over from λ-calculus to CL, as follows.

Definition 2.9. (*Combinatory* β *and* $P\omega$*-equalities*, [13, *Section* 9C]).

(i) $X =_{c\beta} Y \Leftrightarrow X_\lambda =_\beta Y_\lambda$;
(ii) $X =_{P\omega} Y \Leftrightarrow X_\lambda =_{P\omega} Y_\lambda$.

Note 2.10. If desired, more on $c\beta$-equality can be found in [7, Section 6C; 1, Section 7.3; 13, Section 9C] and on the model P_ω in [1, Sections 18.1 and 19.1]. Note that

$$X =_w Y \Rightarrow X =_{c\beta} Y \Rightarrow X =_{P\omega} Y.$$

Definition 2.11. (*From* λ *to* CL; *the H-transformations*). Each pair $\langle \mathscr{B}, \lambda^* \rangle$ determines a transformation $H = H_{\langle \mathscr{B}, \lambda^* \rangle}$ from λ-terms to CL$\mathscr{B}$-terms, defined thus:

$$x_H \equiv x,$$

$$(\lambda x.M)_H \equiv \lambda^* x.(M_H),$$

$$(MN)_H \equiv (M_H N_H).$$

Note. H depends on λ^*; in contrast the λ-transformation only depends on $\mathscr{B}$.

Example 2.12. The eight λ^*-algorithms in Example 2.5 give eight distinct H-transformations, which we shall call respectively

$$H_\eta, \quad H_{\mathrm{abf}}, \quad H_{\mathrm{fab}}, \quad H_{\mathrm{abcdf}}, \quad H_{\mathrm{S}}, \quad H_\beta, \quad H_{\mathrm{T1}}, \quad H_{\mathrm{T2}}.$$

Definition 2.13. (*Relations between the H- and λ-transformations*). Let "=" denote any binary relation, for example $\equiv$, $=_w$, $=_{c\beta}$ or $=_{P\omega}$, on the set of all $CL\mathscr{B}$-terms; define

(i) *H cancels λ modulo* $=$ $\Leftrightarrow$ $X_{\lambda H} = X$ (for all $CL\mathscr{B}$-terms X).

Let "=" denote any binary relation on the set of all λ-terms; define

(ii) *λ cancels H modulo* $=$ $\Leftrightarrow$ $M_{H\lambda} = M$ (for all λ-terms M).

Discussion 2.14. If H and λ both cancel each other modulo an equality, then H is the inverse of λ modulo that equality. Given a complete basis $\mathscr{B}$, can one find a λ^* such that $H_{\langle \mathscr{B}, \lambda^* \rangle}$ is the inverse of the λ-transformation modulo one of the standard equalities? For $\beta\eta$-equality $=_{\beta\eta}$, the answer is that any λ^* will do; it is not hard to prove that every $H_{\langle \mathscr{B}, \lambda^* \rangle}$ is the inverse of λ with respect to $=_{\beta\eta}$. But what about β-equality or even identity?

The following example will show how close the eight H-transformations introduced above come to being inverses of the λ-transformation, and the lemma after it will generalise this.

Example 2.15. For the H-mappings defined by the algorithms in Example 2.5 we have

(i) H_η, H_{abcdf}, H_S, H_β, H_{T1} and H_{T2} (i.e. all except H_{abf} and H_{fab}) cancel λ modulo $\equiv$ (and hence modulo $=_{c\beta}$ and $=_{P\omega}$);

(ii) H_{abf} and H_{fab} cancel λ modulo $=_{c\beta}$ and hence $=_{P\omega}$, but not $\equiv$;

(iii) λ cancels H_{abf}, H_{fab} and H_β modulo $=_\beta$ and hence modulo $=_{P\omega}$,

(iv) λ cancels H_η, H_{abcdf}, H_S, H_{T1} and H_{T2} modulo $=_{\beta\eta}$ but not $=_\beta$.

Proofs. (Mainly from [7, Section 6E] and [1, Section 7.3]).

(i) It is sufficient to check that for all these H-transformations and for all the combinators X belonging to the corresponding basis: $X_{\lambda H} \equiv X$. And the checking is routine. (The key is that clause (c) is in each algorithm.)

(ii) For H_{abf} see [1, Theorem 7.3.10] or [13, Theorem 9.28(a)]. For H_{fab} the proof is similar. Moreover notice that for these H-mappings $\mathbf{S}_{\lambda H} \not\equiv \mathbf{S}$.

(iii) A proof for H_{abf} is in [13, Theorem 9.28(b)], and the others are similar; see [13, Section 9.35] for hints on the proof for H_β.

(iv) For H_η a proof is in [13, Theorem 9.14(b)] and the others are similar. For all these H-mappings $(\lambda xy.xy)_H \equiv \mathbf{I}$, so

$$(\lambda xy.xy)_{H\lambda} \equiv \lambda x.x \neq_\beta \lambda xy.xy. \qquad \square$$

The following lemma shows that an inverse of the λ-transformation can always be found modulo β-equality. It will be used in Section 3.

Lemma 2.16. *Given any complete combinator basis* $\mathscr{B} = \{C_1, C_2, \ldots\}$, *one can always*

define a λ^* (called here λ^{**}) such that $H_{(\mathscr{B},\lambda^{**})}$ and the λ-transformation cancel each other modulo β-equality.

Proof. Since $\mathscr{B}$ is complete there exist combinations $S_{\mathscr{B}}$ and $K_{\mathscr{B}}$ of its members such that

(i) $\quad S_{\mathscr{B}}xyz \rhd_w xz(yz), \qquad K_{\mathscr{B}}xy \rhd_w x.$

By Lemma 2.8 these two reductions can be translated into λ-calculus to give

(ii) $\quad S_{\mathscr{B}\lambda}xyz \rhd_\beta xz(yz), \qquad K_{\mathscr{B}\lambda}xy \rhd_\beta x.$

By the standardization theorem for $\rhd_\beta$ ([1, Theorem 11.4.7]), these reductions can be made by contracting the leftmost redex at each step. Then the reduction of $K_{\mathscr{B}\lambda}xy$ has form

(iii) $\quad K_{\mathscr{B}\lambda}xy \rhd_\beta (\lambda x.U_1)xy \quad$ where $K_{\mathscr{B}\lambda} \rhd_\beta \lambda x.U_1$,

$\rhd_\beta U_1 y$

$\rhd_\beta (\lambda y.U_2)y \quad$ where $U_1 \rhd_\beta \lambda y.U_2$,

$\rhd_\beta U_2$

$\rhd_\beta x.$

Hence $K_{\mathscr{B}\lambda} \rhd_\beta \lambda x.U_1 \rhd_\beta \lambda xy.U_2 \rhd_\beta \lambda xy.x$. Similarly for $S_{\mathscr{B}\lambda}$. Thus

(iv) $\quad K_{\mathscr{B}\lambda} \rhd_\beta \lambda xy.x, \qquad S_{\mathscr{B}\lambda} \rhd_\beta \lambda xyz.xz(yz).$

Define $\lambda^{**}x.Y$ for all x and Y, using $K_{\mathscr{B}}$ and $S_{\mathscr{B}}$, like $\lambda^{\mathrm{fab}}x.Y$ in Example 2.5. It is easy to prove that $\lambda^{**}x.Y$ does not contain x, and

(v) $\quad (\lambda^{**}x.Y)x \rhd_w Y,$

and $\lambda^{**}x.Y$ always has one of the forms

(vi) $\quad K_{\mathscr{B}}U, \qquad S_{\mathscr{B}}UV.$

Hence

(vii) $\quad (\lambda^{**}x.Y)_\lambda =_\beta \lambda x.(Y_\lambda)$

because

$$
\begin{aligned}
(\lambda^{**}x.Y)_\lambda &= K_{\mathscr{B}\lambda}U_\lambda \text{ or } S_{\mathscr{B}\lambda}U_\lambda V_\lambda && \text{with } x \notin \mathrm{FV}(U_\lambda V_\lambda)\\
&=_\beta \lambda x.U_\lambda \text{ or } \lambda x.U_\lambda x(V_\lambda x) && \text{by (iv)}\\
&=_\beta \lambda x.((\lambda x.U_\lambda)x) \text{ or } \lambda x.((\lambda x.U_\lambda x(V_\lambda x))x) && \text{by } \beta\text{-expansion}\\
&=_\beta \lambda x.(K_{\mathscr{B}\lambda}U_\lambda x) \text{ or } \lambda x.(S_{\mathscr{B}\lambda}U_\lambda V_\lambda x) && \text{by (iv)}\\
&= \lambda x.((\lambda^{**}x.Y)_\lambda x)\\
&=_\beta \lambda x.(Y_\lambda) && \text{by (v) and Lemma 2.8.}
\end{aligned}
$$

Now define $H = H_{\langle \mathscr{B}, \lambda^{**} \rangle}$. Then by induction on M one can prove

(viii) $\quad M_{H\lambda} =_\beta M.$

In fact the only non-trivial case is $M \equiv \lambda x.P$, and this is treated as follows:

$$M_{H\lambda} \equiv (\lambda^{**} x.(P_H))_\lambda =_\beta \lambda x.(P_{H\lambda}) \quad \text{by (vii)}$$

$$=_\beta \lambda x.P \quad \text{by induction hypothesis.}$$

Finally, apply (viii) with $M \equiv X_\lambda$. This gives $X_{\lambda H\lambda} =_\beta X_\lambda$; that is

(ix) $\quad X_{\lambda H} =_{c\beta} X.$ □

Warning 2.17. If one tried to go further and define a λ^* so that $H_{\langle \mathscr{B}, \lambda^* \rangle}$ was the inverse of the λ-transformation in the strongest sense, i.e. modulo $\equiv$, then the resulting λ^* would have some awkward properties relating to substitution and reduction, because these concepts in λ-calculus do not correspond exactly to combinatory logic. See [9, Sections 1-3].

3. Intersection types for CL-terms

We shall now introduce a family of type-assignment systems, one for each basis, which can be viewed as translations of $\mathrm{TA}_\lambda(\wedge, \omega, \leqslant)$ into combinatory logic.

Note 3.1. In what follows $\mathscr{B} = \{C_1, C_2, \ldots\}$ is any complete combinator basis, and we use the notation

$$\pi_i \equiv \textit{principal type of } (C_i)_\lambda.$$

Note that π_i always exists, by Definition 1.9, since $(C_i)_\lambda$ is a closed approximate normal form by the assumptions in Section 2.1.

Definition 3.2. (*Type-assignment in* CL).

(i) $\mathrm{TA}_{\mathrm{CL}\mathscr{B}}(\wedge, \omega, \leqslant)$ is the type-assignment system defined by:
statements: $X{:}\tau$ where X is a CL$\mathscr{B}$-term and τ is an intersection type;
rules: $(\to\mathrm{E})$, $(\wedge\mathrm{I})$, $(\wedge\mathrm{E})$, $(\leqslant)$ as in Definition 1.4;
axioms: for each CL$\mathscr{B}$-term X the following is an axiom:

(ω) $\quad X{:}\omega,$

and for each C_i $(i = 1, 2, \ldots)$ and each type π_i^* obtained by substituting types for variables in π_i the following is an axiom:

$(\to C_i)$ $\quad C_i{:}\pi_i^*.$

(ii) We write $\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} X{:}\tau$ iff $X{:}\tau$ is derivable from $\boldsymbol{B}$ in this system.

Example 3.3. (i) For the basis $\mathscr{B}_1 = \{\mathbf{S}, \mathbf{K}, \mathbf{I}\}$ the system $\mathrm{TA}_{\mathrm{CL}\mathscr{B}_1}(\wedge, \omega, \leqslant)$ has rules $(\to\mathrm{E})$, $(\wedge\mathrm{I})$, $(\wedge\mathrm{E})$, $(\leqslant)$, and the axioms

(ω)	$X:\omega$	(an axiom for each $\mathrm{CL}\mathscr{B}_1$-term X),
$(\to\mathbf{S})$	$\mathbf{S}:(\rho_1\to\sigma\to\tau)\to(\rho_2\to\sigma)\to\rho_1\wedge\rho_2\to\tau$	(an axiom for each choice of $\rho_1, \rho_2, \sigma, \tau$),
$(\to\mathbf{K})$	$\mathbf{K}:\tau\to\sigma\to\tau$	(an axiom for each pair σ, τ),
$(\to\mathbf{I})$	$\mathbf{I}:\sigma\to\sigma$	(an axiom for each σ).

(ii) For the other bases in Example 2.5 we take as combinator-axioms of the corresponding $\mathrm{TA}_{\mathrm{CL}}$ system all the substitution instances of the principal types of the appropriate combinators in Exercise 1.10.

The next lemma states three fundamental properties of $\vdash_{\mathrm{CL}\mathscr{B}}$. Then afterwards we shall focus on the relation between $\vdash_{\mathrm{CL}\mathscr{B}}$ and $\vdash_\lambda$. The first few results will hold for all bases $\mathscr{B}$, but later we shall concentrate on the particular bases in Example 2.5.

Lemma 3.4. *Let $\mathscr{B}$ be any complete combinator basis; let $\boldsymbol{B}$ be any type-assignment basis and z be a type-variable not in $\boldsymbol{B}$. Then*

(i) $\boldsymbol{B}, z:\sigma\vdash_{\mathrm{CL}\mathscr{B}} z:\tau \Rightarrow \sigma\leqslant\tau$.

(ii) $\boldsymbol{B}\vdash_{\mathrm{CL}\mathscr{B}} XY:\tau \Rightarrow \exists\sigma$ *such that* $\boldsymbol{B}\vdash_{\mathrm{CL}\mathscr{B}} X:\sigma\to\tau$ *and* $\boldsymbol{B}\vdash_{\mathrm{CL}\mathscr{B}} Y:\sigma$.

(iii) *If* $\boldsymbol{B}, z:\sigma\vdash_{\mathrm{CL}\mathscr{B}} Yz:\tau$ *and* $z\notin\mathrm{FV}(Y)$ *then* $\boldsymbol{B}\vdash_{\mathrm{CL}\mathscr{B}} Y:\sigma\to\tau$.

Note. The above are true also for $\vdash_\lambda$; for (i) and (ii) see [2, Lemmas 2.7(ii) and 2.8(i)], and for (iii) see the η-lemma in [11, Section 5].

Proof. (i) Follows by an easy induction on deductions.

(ii) By induction on the deduction of $XY:\tau$. The only interesting case is when the last applied rule is $(\wedge\mathrm{I})$. Say $\tau\equiv\tau_1\wedge\tau_2$ and we have

$$\frac{XY:\tau_1 \qquad XY:\tau_2}{XY:\tau_1\wedge\tau_2}.$$

By induction hypothesis there are σ_1 and σ_2 such that

$$\boldsymbol{B}\vdash_{\mathrm{CL}\mathscr{B}} X:\sigma_i\to\tau_i, \quad \boldsymbol{B}\vdash_{\mathrm{CL}\mathscr{B}} Y:\sigma_i \qquad \text{for } i=1,2.$$

So by $(\wedge\mathrm{I})$,

$$\boldsymbol{B}\vdash_{\mathrm{CL}\mathscr{B}} X:(\sigma_1\to\tau_1)\wedge(\sigma_2\to\tau_2), \qquad \boldsymbol{B}\vdash_{\mathrm{CL}\mathscr{B}} Y:\sigma_1\wedge\sigma_2.$$

Then the result follows from rule $(\leqslant)$ since $(\sigma_1\to\tau_1)\wedge(\sigma_2\to\tau_2)\leqslant\sigma_1\wedge\sigma_2\to\tau_1\wedge\tau_2$.

(iii) $\boldsymbol{B}, z:\sigma\vdash_{\mathrm{CL}\mathscr{B}} Yz:\tau$ implies by (ii) that there is a ρ such that

$$\boldsymbol{B}\vdash_{\mathrm{CL}\mathscr{B}} Y:\rho\to\tau, \qquad z:\sigma\vdash_{\mathrm{CL}\mathscr{B}} z:\rho.$$

(Note that z does not occur in $\boldsymbol{B}$.) Hence $\sigma \leqslant \rho$ by (i). It follows that $\rho \to \tau \leqslant \sigma \to \tau$, so by rule ($\leqslant$), we have

$$\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} Y : \sigma \to \tau. \qquad \square$$

Note 3.5. We have defined $\mathrm{TA}_{\mathrm{CL}\mathscr{B}}(\wedge, \omega, \leqslant)$ in the hope that it will correspond to $\mathrm{TA}_{\lambda}(\wedge, \omega, \leqslant)$ in the following strong sense. Our hope is to prove, first, that

(i) $\qquad \boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} X : \tau \Leftrightarrow \boldsymbol{B} \vdash_{\lambda} X_{\lambda} : \tau;$

and second, that there exists a λ^*-algorithm such that

(ii) $\qquad \boldsymbol{B} \vdash_{\lambda} M : \tau \Leftrightarrow \boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} M_{H_{(\mathscr{B},\lambda^*)}} : \tau.$

For (i) "$\Rightarrow$", the proof is trivial because the axioms for C_i have been chosen to be provable in the λ-system. In contrast, for (i) "$\Leftarrow$" and (ii) we shall not try to prove the properties for a general $\mathscr{B}$, but shall state general conditions under which they hold, see Theorem 3.7, and then show that these conditions are satisfied by all the bases and some of the λ^*'s introduced in Example 2.5; see Theorem 3.11.

Definition 3.6. *λ^* satisfies* ($\to$I) iff, for all $\boldsymbol{B}$ and all x not in $\boldsymbol{B}$,

$$\boldsymbol{B}, x:\sigma \vdash_{\mathrm{CL}\mathscr{B}} Y : \tau \Rightarrow \boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} (\lambda^* x.Y) : \sigma \to \tau.$$

Theorem 3.7. *Let $\mathscr{B}$ be any complete combinator basis.*

(i) *We have*

$$\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} X : \tau \Rightarrow \boldsymbol{B} \vdash_{\lambda} X_{\lambda} : \tau.$$

(ii) *If there is a λ^* satisfying* ($\to$I) *and such that $H_{(\mathscr{B},\lambda^*)}$ cancels the λ-transformation modulo identity, then*

$$\boldsymbol{B} \vdash_{\lambda} X_{\lambda} : \tau \Rightarrow \boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} X : \tau.$$

(iii) *For all λ^*: if λ^* satisfies* ($\to$I), *then*

$$\boldsymbol{B} \vdash_{\lambda} M : \tau \Rightarrow \boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} M_H : \tau \quad (H = H_{(\mathscr{B},\lambda^*)}).$$

(iv) *For all λ^*: if the λ-transformation cancels H modulo Pω-equality, then*

$$\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} M_H : \tau \Rightarrow \boldsymbol{B} \vdash_{\lambda} M : \tau \quad (H = H_{(\mathscr{B},\lambda^*)}).$$

(v) *There is always an H-transformation such that*

$$\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} M_H : \tau \Rightarrow \boldsymbol{B} \vdash_{\lambda} M : \tau.$$

Proof. (i) This is Note 3.5(i) "$\Rightarrow$", and holds because the type-axioms for the members of $\mathscr{B}$ were chosen to be λ-provable.

(iii) This comes by induction on $\vdash_{\lambda}$, see Definition 1.4. The only non-trivial case is Rule ($\to$I); but λ^{η} has been assumed to satisfy ($\to$I).

(ii) Let $\boldsymbol{B}\vdash_\lambda X_\lambda:\tau$. Now (iii) can be applied because λ^η satisfies $(\to\mathrm{I})$, so $\boldsymbol{B}\vdash_{\mathrm{CL}_{\mathscr{B}}} X_{\lambda H}:\tau$. But $X_{\lambda H}\equiv X$ by assumption, so $\boldsymbol{B}\vdash_{\mathrm{CL}_{\mathscr{B}}} X:\tau$.

(iv) Let $\boldsymbol{B}\vdash_{\mathrm{CL}_{\mathscr{B}}} M_H:\tau$. Then by (i), $\boldsymbol{B}\vdash_\lambda M_{H\lambda}:\tau$. But λ cancels H modulo $P\omega$-equality, so $M_{H\lambda}=_{P\omega} M$. Hence by Theorem 1.6, $\boldsymbol{B}\vdash_\lambda M:\tau$.

(v) This follows immediately from (iv) by choosing $H=H_{\langle\mathscr{B},\lambda^{**}\rangle}$ where λ^{**} is defined in Lemma 2.16. □

Corollary 3.8. (*Type-invariance under $P\omega$-equality in* CL). *If $\mathscr{B}$ is a complete combinator basis and* $\{\boldsymbol{B}\vdash_\lambda X_\lambda:\tau \Rightarrow \boldsymbol{B}\vdash_{\mathrm{CL}_{\mathscr{B}}} X:\tau\}$, *then*

$$\boldsymbol{B}\vdash_{\mathrm{CL}_{\mathscr{B}}} X:\tau \text{ and } X=_{P\omega} Y \;\Rightarrow\; \boldsymbol{B}\vdash_{\mathrm{CL}_{\mathscr{B}}} Y:\tau.$$

Proof. Definition 2.9(ii) and Theorems 1.6 and 3.7(i). □

Lemma 3.9. *All the eight λ^{**}'s defined in Example* 2.5 *satisfy* $(\to\mathrm{I})$.

Proof. We shall prove by induction on the deduction of $Y:\tau$ that

$$\boldsymbol{B}, x:\sigma\vdash_{\mathrm{CL}_{\mathscr{B}}} Y:\tau \;\Rightarrow\; \boldsymbol{B}\vdash_{\mathrm{CL}_{\mathscr{B}}} (\lambda^* x.Y):\sigma\to\tau.$$

We shall prove this result for all eight λ^*'s at once, and will use the induction hypothesis for λ^η in proving the induction step for λ^β.

Case 1. $Y:\tau$ is $x:\sigma$. Hence $Y\equiv x$, so $\lambda^* x.Y\equiv \mathbf{I}$. But $\mathbf{I}:\sigma\to\sigma$ is an axiom.

Case 2. $Y:\tau$ is either in $\boldsymbol{B}$, or is an $\mathbf{S}, \mathbf{K}, \mathbf{I}$, etc. axiom, or is an ω-axiom with Y an atom $\not\equiv x$. Hence Y is an atom and $x\notin \mathrm{FV}(Y)$, so $\lambda^* x.\ Y\equiv \mathbf{K}Y$. Hence, by the axiom $\mathbf{K}$: $\tau\to\sigma\to\tau$ and rule $(\to\mathrm{E})$, $\boldsymbol{B}\vdash_{\mathrm{CL}_{\mathscr{B}}}\mathbf{K}Y$: $\sigma\to\tau$.

Case 3. $Y:\tau$ is an ω-axiom, so $\tau\equiv\omega$. Now $(\lambda^* x.Y):\omega$ is an ω-axiom. And, since $\sigma\leqslant\omega$, we have $\omega\leqslant\omega\to\omega\leqslant\sigma\to\omega$. Hence $(\lambda^* x.Y)$: $\sigma\to\omega$ by rule $(\leqslant)$

Case 4. The last step in the deduction of $Y:\tau$ is $(\leqslant)$ or $(\wedge\mathrm{E})$:

$$\begin{array}{c} \boldsymbol{B} \quad x:\sigma \\ \vdots \\ \dfrac{Y:\rho}{Y:\tau}\ (\rho\leqslant\tau) \end{array}$$

Then $\sigma\to\rho\leqslant\sigma\to\tau$, so we use the induction hypothesis and rule $(\leqslant)$.

Case 5. (Rule $(\wedge\mathrm{I})$).

$$\frac{Y:\tau_1 \qquad Y:\tau_2}{Y:\ \tau_1\wedge\tau_2}\ (\tau\equiv\tau_1\wedge\tau_2)$$

By induction hypothesis, $\boldsymbol{B}\vdash_{\mathrm{CL}_{\mathscr{B}}} (\lambda^* x.Y)$: $\sigma\to\tau_i$ for $i=1,2$. But $(\sigma\to\tau_1)\wedge(\sigma\to\tau_2)\leqslant\sigma\to\tau_1\wedge\tau_2$, so rules $(\wedge\mathrm{I})$ and $(\leqslant)$ give the result.

Case 6 (Rule (→E)). Say $Y \equiv UV$, and we have

$$\dfrac{\begin{array}{cc} \boldsymbol{B}\ x{:}\sigma & \boldsymbol{B}\ x{:}\sigma \\ \vdots & \vdots \\ U{:}\rho \to \tau & V{:}\rho \end{array}}{UV{:}\tau}$$

Now $\lambda^* x.UV$ is defined by one of the clauses (a), (c), (d), (d*), (e), (e*), (f), (f*), (g) in Example 2.5, using one of the combinators **K**, **B**, **B***, **C**, **C***, **S**, **S***, **W**. Thus there are nine sub-cases. We show here only (c), (f) and (f*); the others are similar.

Subcase 6c. $V \equiv x$, $x \notin \mathrm{FV}(U)$, and $\lambda^* x.UV \equiv U$. Since $\boldsymbol{B}$, $x{:}\sigma \vdash_{\mathrm{CL}_{\mathscr{B}}} x{:}\rho$, we have $\sigma \leqslant \rho$ by 3.4(i). Hence $\rho \to \tau \leqslant \sigma \to \tau$. But $\boldsymbol{B} \vdash_{\mathrm{CL}_{\mathscr{B}}} U{:}\ \rho \to \tau$ since $x \notin \mathrm{FV}(U)$; hence by (≤), $\boldsymbol{B} \vdash_{\mathrm{CL}_{\mathscr{B}}} U{:}\ \sigma \to \tau$.

Subcase 6f. $\lambda^{**} x.UV \equiv \mathbf{S}(\lambda^{**} x.U)(\lambda^* x.V)$, where λ^{**} is λ^η if λ^* is λ^β, otherwise λ is λ. By the induction hypothesis for λ, we have

$$\boldsymbol{B} \vdash_{\mathrm{CL}_{\mathscr{B}}} (\lambda^{**} x.U){:}\ \sigma \to \rho \to \tau, \qquad \boldsymbol{B} \vdash_{\mathrm{CL}_{\mathscr{B}}} (\lambda^{**} x.V){:}\ \sigma \to \rho.$$

Now the axioms for **S** are exactly the substitution instances of the principal type of $\mathbf{S}_\lambda$ in Exercise 1.10. Hence the following is an axiom:

$$\mathbf{S}{:}\ (\sigma \to \rho \to \tau) \to (\sigma \to \rho) \to \sigma \wedge \sigma \to \tau.$$

Therefore by rules (≤) and (→E),

$$\boldsymbol{B} \vdash_{\mathrm{CL}_{\mathscr{B}}} \mathbf{S}(\lambda^{**} x.U)(\lambda^{**} x.V){:}\ \sigma \to \tau.$$

Subcase 6f*. $U \equiv U_1 U_2$, $x \notin \mathrm{FV}(U_1)$, and $\lambda^* x.UV \equiv \mathbf{S}^* U_1(\lambda^* x.U_2)(\lambda^* x.V)$. Now $\boldsymbol{B}$, $x{:}\sigma \vdash_{\mathrm{CL}_{\mathscr{B}}} U_1 U_2{:}\ \rho \to \tau$, so by 3.4(ii) there exists a ζ such that

$$\boldsymbol{B},\ x{:}\sigma \vdash_{\mathrm{CL}_{\mathscr{B}}} U_1{:}\ \zeta \to \rho \to \tau, \qquad \boldsymbol{B},\ x{:}\sigma \vdash_{\mathrm{CL}_{\mathscr{B}}} U_2{:}\zeta.$$

By the induction hypothesis, we have

$$\boldsymbol{B} \vdash_{\mathrm{CL}_{\mathscr{B}}} (\lambda^* x.U_2){:}\ \sigma \to \zeta, \qquad \boldsymbol{B} \vdash_{\mathrm{CL}_{\mathscr{B}}} (\lambda^* x.V){:}\ \sigma \to \rho.$$

Now the axioms for $\mathbf{S}^*$ are exactly the substitution instances of the principal type of $\mathbf{S}^*_\lambda$ in Exercise 1.10. Hence the following is an axiom:

$$\mathbf{S}^*{:}\ (\zeta \to \rho \to \tau) \to (\sigma \to \zeta) \to (\sigma \to \rho) \to \sigma \wedge \sigma \to \tau.$$

Therefore by rules (≤) and (→E),

$$\boldsymbol{B} \vdash_{\mathrm{CL}_{\mathscr{B}}} \mathbf{S}^* U_1(\lambda^* x.U_2)(\lambda^* x.V){:}\ \sigma \to \tau. \qquad \square$$

The above lemma implies that also the λ^{**} defined in Lemma 2.16 satisfies (→I) under suitable conditions, as follows.

Corollary 3.10. *Let $\mathscr{B}$ be a complete combinator basis such that*

$$\vdash_{\mathrm{CL}_{\mathscr{B}}} K_{\mathscr{B}}{:}\pi_{\mathbf{K}}, \qquad \vdash_{\mathrm{CL}_{\mathscr{B}}} S_{\mathscr{B}}{:}\pi_{\mathbf{S}},$$

where $K_{\mathscr{B}}$ and $S_{\mathscr{B}}$ are defined as in the proof of Lemma 2.16 *and $\pi_{\mathbf{K}}$ and $\pi_{\mathbf{S}}$ are the principal types of $\mathbf{K}_\lambda$ and $\mathbf{S}_\lambda$, see Exercise* 1.10. *Then λ^{**} as defined in the proof of Lemma* 2.16 *satisfies* $(\to\mathrm{I})$.

Proof. It is sufficient to observe that λ^{**} is λ^{abf} relative to $S_{\mathscr{B}}$ and $K_{\mathscr{B}}$, and so the proof of Lemma 3.9 applies since $S_{\mathscr{B}}$ and $K_{\mathscr{B}}$ have the suitable types. □

Theorem 3.11. *For the five bases $\mathscr{B}_i$ in Example* 2.5, *where $i = 0, \ldots, 4$, the $\mathrm{CL}\mathscr{B}_i$-system is equivalent to the λ-one; more precisely*

(i) *for $i = 0, \ldots, 4$:*

$$\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}i} X{:}\tau \;\Leftrightarrow\; \boldsymbol{B} \vdash_\lambda X_\lambda{:}\tau;$$

(ii) *for $i = 1$ and $H = H_{\mathrm{abf}}, H_{\mathrm{fab}}, H_\beta$:*

$$\boldsymbol{B} \vdash_\lambda M{:}\tau \;\Leftrightarrow\; \boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}_1} M_H{:}\tau;$$

(iii) *for H_η, H_{abcdf}, H_{S}, H_{T1}, H_{T2} one has "$\Rightarrow$" in* (ii) (*with respect to the corresponding basis*) *but not "$\Leftarrow$";*

(iv) *for $i = 0, \ldots, 4$ and $H = H_{\mathrm{abf}}$,*

$$\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}_i} M_H{:}\tau \;\Rightarrow\; \boldsymbol{B} \vdash_\lambda M{:}\tau.$$

Proof. (i) By Theorem 3.7(i) and (ii), noting that λ^η, λ^{abcdf}, λ^{S}, λ^{T1} and λ^{T2} satisfy $(\to\mathrm{I})$ by Lemma 3.9 and that H_η, H_{abcdf}, H_{S}, H_{T1} and H_{T2} cancel λ modulo identity by Example 2.15. (λ^β could be used instead of λ^η.)

(ii) By Theorem 3.7(iii) and (iv), Lemma 3.9 and Example 2.15.

(iii) By Theorem 3.7(iii) and Lemma 3.9 one gets "$\Rightarrow$" in (ii). A counterexample to "$\Leftarrow$" is $M \equiv \lambda xy.xy$; one has $M_H \equiv \mathbf{I}$ which has type $\varphi \to \varphi$ in the $\mathrm{CL}\mathscr{B}_i$-system (φ being a type-variable), but it can be shown that M does not have this type in the λ-system. (The principal type of M is $(\varphi_1 \to \varphi_2) \to \varphi_1 \to \varphi_2$.)

(iv) It suffices to observe that if λ^{**} is built according to Lemma 2.16 we have $H_{\langle \mathscr{B}, \lambda^{**} \rangle} = H_{\mathrm{abf}}$. □

For the five bases $\mathscr{B}_i$ in Example 2.5, the following theorem shows that $\mathrm{TA}_{\mathrm{CL}\mathscr{B}_i}(\wedge, \omega, \leqslant)$ is invariant under β-equality, also that type preservation is exactly characterized by $P\omega$-equality.

Theorem 3.12 (*Preservation of types in* $\mathrm{CL}\mathscr{B}_i$). *For the five bases $\mathscr{B}_i$ in Example* 2.5, *where $i = 0, \ldots, 4$ we have*:

(i) *type-assignment statements are preserved by combinatory β-equality; that is, if $\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}_i} X{:}\tau$ and $X =_{c\beta} Y$, then $\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}_i} Y{:}\tau$; and*

(ii) *type-preservation is exactly characterized by $P\omega$-equality; that is*

$$X =_{P\omega} Y \;\Leftrightarrow\; (\textit{for all } \boldsymbol{B}, \tau)\{\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}_i} X{:}\tau \;\Leftrightarrow\; \boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}_i} Y{:}\tau\}.$$

Proof. For (i), use Theorem 3.11(i), Theorem 1.5(i) and Definition 2.9(i) (the definition of $c\beta$-equality), and for (ii), use Corollary 3.8 and Theorem 3.11(i). □

Incidental Note 3.13. The CL-system for each of the five bases has been proved equivalent to the λ-system in a fairly strong sense (Theorem 3.11(i)). But the equivalence is not as neat as one might like (see Theorem 3.11(ii)-(iv)). And here is another point where CL and λ diverge.

Following [11], let the set ***NTS*** of *normal types* be the set of all types σ such that: either $\sigma \equiv \omega$ or $\sigma \equiv \sigma_1 \wedge \cdots \wedge \sigma_n$ with some bracketing and with each σ_i having the form

$$\sigma_i \equiv \sigma_{i,1} \to \cdots \to \sigma_{i,m_i} \to \varphi_i.$$

Normal types correspond closely to the original formulation of intersection types, see [4], which were slightly more restricted than those in [2] and later papers, including this one. In [11] it was proved that (as Coppo and Dezani already knew informally) the restriction was trivial, in the sense that every deduction $\boldsymbol{B} \vdash_\lambda M:\tau$ could be paralleled by a deduction $\boldsymbol{B}^* \vdash_\lambda M:\tau^*$ containing only normal types, where the map $*:\boldsymbol{T} \to \boldsymbol{NTS}$ applied to a type gave its "normal form".

But in CL the restriction seems not to be so trivial. For example, in CL there is a problem with the axiom

$$\mathbf{I}:\sigma \wedge \tau \to \sigma \wedge \tau.$$

The type in this is not normal, and the nearest normal type to it is

$$(\sigma \wedge \tau \to \sigma) \wedge (\sigma \wedge \tau \to \tau).$$

So if types were restricted to being normal, quite a complicated form of the axiom scheme for **I** would be needed to give a reasonable equivalence to the λ-system, and similarly for **S** and **K**.

4. Replacing rule ($\leqslant$)

In this section we propose an alternative formulation of intersecton type-assignment to CL-terms in which rule ($\leqslant$) has been replaced by something simpler.

Throughout the section $\mathscr{B}$ will be any complete basis whose combinators have the "right" types in the following sense.

Definition 4.1. A combinator basis $\mathscr{B}$ is *well-typed for* **B**, **B′**, **I** iff there exist CL$\mathscr{B}$-combinators **I**, **B** and **B′**, not necessarily atoms, such that

$$\mathbf{I}x \rhd_w x, \qquad \mathbf{B}xyz \rhd_w x(yz), \qquad \mathbf{B}'xyz \rhd_w y(xz),$$

and the following are provable in $\mathrm{TA}_{\mathrm{CL}\mathscr{B}}(\wedge, \omega, \leqslant)$ (cf. Exercise 1.10) without using

rule ($\leqslant$):

$$\mathbf{I}\colon \sigma\to\sigma, \qquad \mathbf{B}\colon (\sigma\to\rho)\to(\tau\to\sigma)\to\tau\to\rho, \qquad \mathbf{B}'\colon (\sigma\to\rho)\to(\rho\to\tau)\to\sigma\to\tau$$

for each choice of σ, ρ, τ.

Motivation 4.2. The aim is to find axioms to replace rule ($\leqslant$). Now the $\leqslant$-relation on types is a formal analogue of the subset relation $\subseteq$ on sets, so we shall look for axioms of form

(i) $\qquad \mathbf{I}\colon \sigma\to\tau.$

This is because, informally, $\mathbf{I}\colon \sigma\to\tau$ says that $\sigma\subseteq\tau$. More precisely, if σ and τ represent sets $[\![\sigma]\!]$ and $[\![\tau]\!]$ in some way, then $\mathbf{I}\colon \sigma\to\tau$ says that if x is in $[\![\sigma]\!]$, then $\mathbf{I}x$ is in $[\![\tau]\!]$. Since $\mathbf{I}x = x$, this says that every member of $[\![\sigma]\!]$ is in $[\![\tau]\!]$.

The system below will contain four axiom-schemes of form $\mathbf{I}\colon \sigma\to\tau$. However it is not possible to replace rule ($\leqslant$) entirely by such axiom-schemes, because in order to use an axiom $\mathbf{I}\colon \sigma\to\tau$ to deduce $X:\tau$ from $X:\sigma$, one needs also a rule that gives

$$\mathbf{I}X:\tau \vdash X:\tau.$$

This will be rule ($\mathbf{I}_5$) below.

Besides this rule the system will contain two special cases of the η-rule, so it will be partly like the $\lambda\eta$-system in Definition 1.7 and partly an axiomatization (see Note 4.4).

Definition 4.3. (i) $\mathrm{TA}_{\mathrm{CL}\mathscr{B}}(\wedge, \omega, \eta)$ is the system for $\mathrm{CL}\mathscr{B}$-terms whose axiom-schemes are the axiom schemes of $\mathrm{TA}_{\mathrm{CL}\mathscr{B}}(\wedge, \omega, \leqslant)$ and

($\mathbf{I}_1$) $\qquad \mathbf{I}\colon \sigma\to\omega$

($\mathbf{I}_2$) $\qquad \mathbf{I}\colon \omega\to\omega\to\omega$

($\mathbf{I}_3$) $\qquad \mathbf{I}\colon \sigma_1\wedge\sigma_2\to\sigma_i \quad (i=1,2)$

($\mathbf{I}_4$) $\qquad \mathbf{I}\colon (\sigma\to\tau)\wedge(\sigma\to\rho)\to\sigma\to\tau\wedge\rho$

and whose rules are ($\to$E), ($\wedge$I), ($\wedge$E) and

($\mathbf{I}_5$) $\dfrac{\mathbf{I}X:\sigma}{X:\sigma}$ $\qquad$ (η_1) $\dfrac{\mathbf{B}\mathbf{I}:\sigma}{\mathbf{I}:\sigma}$ $\qquad$ (η_2) $\dfrac{\mathbf{B}'\mathbf{I}:\sigma}{\mathbf{I}:\sigma}$

for all choices of σ, σ_1, σ_2, ρ and τ.

(ii) We write $\boldsymbol{B}\vdash_{\mathrm{CL}\mathscr{B}\eta} X:\sigma$ if $X:\sigma$ is derivable from the basis $\boldsymbol{B}$ in this system.

Note 4.4. Rules (η_1) and (η_2) will be used in the proof of equivalence to the ($\leqslant$) system. It would be nice to weaken them and still prove equivalence, and rule (η_2) is an obvious candidate for redundancy, because it seems to do essentially the same work as (η_1). In particular, $\mathbf{B}\mathbf{I} =_{c\beta} \mathbf{B}'\mathbf{I}$, so in each $\mathrm{CL}\mathscr{B}$-system equivalent to the

λ-one (i.e. such that $\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} X{:}\sigma$ iff $\boldsymbol{B} \vdash_{\lambda} X_{\lambda}{:}\sigma$) these combinators will have the same types. Therefore (η_1) and (η_2) both say nearly the same thing. But this is deceptive; in reality the contributions of **BI** and **B'I** to the equivalence proof will not be through their types, but through deductions of form

$$\mathbf{I}{:}\,\sigma \to \tau \;\vdash\; \mathbf{BI}{:}\,(\rho \to \sigma) \to \rho \to \tau,$$

$$\mathbf{I}{:}\,\sigma \to \tau \;\vdash\; \mathbf{B'I}{:}\,(\tau \to \rho) \to \sigma \to \rho,$$

and these are distinct. Simplifying the two η-rules may be possible, but it will not be so easy.

We shall prove after the following lemma that $\mathrm{TA}_{\mathrm{CL}\mathscr{B}}(\wedge, \omega, \leqslant)$ and $\mathrm{TA}_{\mathrm{CL}\mathscr{B}}(\wedge, \omega, \eta)$ are equivalent.

Lemma 4.5. *Let $\mathscr{B}$ be complete and well-typed for* **B**, **B'**, **I**. *If $\sigma \leqslant \sigma'$, then*

$$\vdash_{\mathrm{CL}\mathscr{B}\eta} \mathbf{I}{:}\,\sigma \to \sigma'.$$

Proof. Induction on the proof of $\sigma \leqslant \sigma'$ using Definition 4.1. We consider only the non-trivial cases.

Axiom $\sigma \leqslant \sigma \wedge \sigma$.

$$\dfrac{\dfrac{(\mathbf{I}_4)\text{-ax}}{\mathbf{I}{:}\,(\sigma\to\sigma)\wedge(\sigma\to\sigma)\to\sigma\to\sigma\wedge\sigma} \qquad \dfrac{\dfrac{\mathbf{I}{:}\,\sigma\to\sigma \qquad \mathbf{I}{:}\,\sigma\to\sigma}{\mathbf{I}{:}\,(\sigma\to\sigma)\wedge(\sigma\to\sigma)}(\wedge\mathrm{I})}{\mathbf{II}{:}\,\sigma\to\sigma\wedge\sigma}(\to\mathrm{E})}{\mathbf{I}{:}\,\sigma\to\sigma\wedge\sigma}(\mathbf{I}_5)$$

Transitivity. Suppose $\mathbf{I}{:}\,\sigma \to \tau$ and $\mathbf{I}{:}\,\tau \to \rho$. Deduce $\mathbf{I}{:}\,\sigma \to \rho$ thus:

$$\dfrac{\dfrac{\dfrac{\mathbf{B}{:}\,(\tau\to\rho)\to(\sigma\to\tau)\to\sigma\to\rho \qquad \mathbf{I}{:}\,\tau\to\rho}{\mathbf{BI}{:}\,(\sigma\to\tau)\to\sigma\to\rho}(\to\mathrm{E})}{\mathbf{I}{:}\,(\sigma\to\tau)\to\sigma\to\rho}(\eta_1) \qquad \mathbf{I}{:}\,\sigma\to\tau}{\mathbf{II}{:}\,\sigma\to\rho}(\to\mathrm{E})}{\mathbf{I}{:}\,\sigma\to\rho}(\mathbf{I}_5)$$

Replacement in $\wedge$. Assume $\mathbf{I}{:}\,\sigma \to \sigma'$ and $\mathbf{I}{:}\,\tau \to \tau'$. Deduce $\mathbf{I}{:}\,\sigma \wedge \tau \to \sigma' \wedge \tau'$ thus:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathbf{B}{:}\,(\sigma\to\sigma')\to(\sigma\wedge\tau\to\sigma)\to\sigma\wedge\tau\to\sigma' \quad \mathbf{I}{:}\,\sigma\to\sigma'}{\mathbf{BI}{:}\,(\sigma\wedge\tau\to\sigma)\to\sigma\wedge\tau\to\sigma'}(\to\mathrm{E})}{\mathbf{I}{:}\,(\sigma\wedge\tau\to\sigma)\to\sigma\wedge\tau\to\sigma'}(\eta_1) \qquad (\mathbf{I}_3)\text{-ax}}{\mathbf{I}{:}\,\sigma\wedge\tau\to\sigma}(\to\mathrm{E}) \qquad \dfrac{\dfrac{\mathbf{II}{:}\,\sigma\wedge\tau\to\sigma'}{\mathbf{I}{:}\,\sigma\wedge\tau\to\sigma'}(\mathbf{I}_5) \qquad \dfrac{\text{SIMILAR}}{\mathbf{I}{:}\,\sigma\wedge\tau\to\tau'}}{\mathbf{I}{:}\,(\sigma\wedge\tau\to\sigma')\wedge(\sigma\wedge\tau\to\tau')}(\wedge\mathrm{I})}{\mathbf{I}{:}\,\sigma\wedge\tau\to\sigma'\wedge\tau'}(\to\mathrm{E}) \qquad (\mathbf{I}_4)\text{-ax}}{\mathbf{II}{:}\,\sigma\wedge\tau\to\sigma'\wedge\tau'}(\mathbf{I}_5)$$

$$\mathbf{I}{:}\,(\sigma\wedge\tau\to\sigma')\wedge(\sigma\wedge\tau\to\tau')\to\sigma\wedge\tau\to\sigma'\wedge\tau'$$

$$\mathbf{I}{:}\,\sigma\wedge\tau\to\sigma'\wedge\tau'$$

Replacement in $\rightarrow$. Assume $\mathbf{I}: \sigma \rightarrow \sigma'$ and $\mathbf{I}: \tau \rightarrow \tau'$. Deduce $\mathbf{I}: (\sigma' \rightarrow \tau) \rightarrow \sigma \rightarrow \tau'$ as follows. In this deduction, let $\xi \equiv \sigma \rightarrow \tau$, $\eta \equiv \sigma \rightarrow \tau'$, and $\zeta \equiv \sigma' \rightarrow \tau$.

$$\dfrac{\dfrac{\mathbf{B}: (\zeta \rightarrow \xi) \rightarrow \zeta \rightarrow \eta}{\mathbf{BI}: (\zeta \rightarrow \xi) \rightarrow \zeta \rightarrow \eta}(\eta_1)}{\mathbf{I}: (\zeta \rightarrow \xi) \rightarrow \zeta \rightarrow \eta} \qquad \dfrac{\dfrac{\mathbf{B}: (\tau \rightarrow \tau') \rightarrow (\sigma \rightarrow \tau) \rightarrow \sigma \rightarrow \tau' \quad \mathbf{I}: \tau \rightarrow \tau'}{\mathbf{BI}: (\sigma \rightarrow \tau) \rightarrow \sigma \rightarrow \tau'}(\eta_1)}{\mathbf{I}: \xi \rightarrow \eta}(\rightarrow \mathrm{E}) \qquad \dfrac{\dfrac{\mathbf{B}': (\sigma \rightarrow \sigma') \rightarrow (\sigma' \rightarrow \tau) \rightarrow \sigma \rightarrow \tau \quad \mathbf{I}: \sigma \rightarrow \sigma'}{\mathbf{B}'\mathbf{I}: (\sigma' \rightarrow \tau) \rightarrow \sigma \rightarrow \tau}(\eta_2)}{\mathbf{I}: \zeta \rightarrow \xi}(\rightarrow \mathrm{E})$$

$$\dfrac{}{\mathbf{I}: \zeta \rightarrow \eta}(\mathbf{I}_5)$$

This completes the proof of Lemma 4.5. □

Theorem 4.6. *Let* $\mathscr{B}$ *be any complete combinator basis, well-typed for* $\mathbf{B}, \mathbf{B}', \mathbf{I}$. *Then*

$$\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} X:\sigma \Leftrightarrow \boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}_\eta} X:\sigma.$$

Proof. "$\Rightarrow$": The only thing to show is that $(\leqslant)$ is an admissible rule in $\mathrm{TA}_{\mathrm{CL}\mathscr{B}}(\wedge, \omega, \eta)$; that is, to show that if $\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}_\eta} X:\sigma$ and $\sigma \leqslant \tau$, then $\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}_\eta} X:\tau$. By the preceding lemma, $\vdash_{\mathrm{CL}\mathscr{B}_\eta} \mathbf{I}: \sigma \rightarrow \tau$. Then we can deduce

$$\dfrac{\dfrac{\mathbf{I}: \sigma \rightarrow \tau \qquad X:\sigma}{\mathbf{I}X:\tau}(\rightarrow \mathrm{E})}{X:\tau}(\mathbf{I}_5)$$

"$\Leftarrow$": We must show that all the axioms and rules introduced in Definition 4.3 are admissible in $\mathrm{TA}_{\mathrm{CL}\mathscr{B}}(\wedge, \omega, \leqslant)$. For the axioms, rule $(\leqslant)$ and Definition 4.1 give admissibility immediately. For rule $(\mathbf{I}_5)$, $\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} \mathbf{I}X:\sigma$ implies by Lemma 3.4 (ii) that there exists τ such that

$$\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} \mathbf{I}: \tau \rightarrow \sigma, \qquad \boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} X:\tau.$$

From $\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} \mathbf{I}: \tau \rightarrow \sigma$ we have $\boldsymbol{B} \vdash_{\lambda} \lambda x.x: \tau \rightarrow \sigma$ by Theorem 3.7(i). It is easy to verify that

$$\boldsymbol{B} \vdash_{\lambda} \lambda x.x: \tau \rightarrow \sigma \Rightarrow \sigma \leqslant \tau.$$

So we can conclude $\boldsymbol{B} \vdash_{\mathrm{CL}\mathscr{B}} X:\sigma$ by rule $(\leqslant)$.

The admissibility of rules (η_1) and (η_2) follows from a simple case analysis using Lemma 3.4 (ii) and Definition 4.1. □

Note 4.7. Rule $(\leqslant)$ can also be replaced by a strengthened $\mathbf{I}$-axiom-scheme saying

$$\mathbf{I}: \sigma \rightarrow \tau \quad (\sigma \leqslant \tau),$$

together with the single $\mathbf{I}$-rule $(\mathbf{I}_5)$. (The proof is like that of the theorem above.) But to replace rule $(\leqslant)$ by an axiom-scheme in which the $\leqslant$-relation plays such a strong role is hardly a simplification.

Acknowledgement

In the first version of the present paper, we did not consider general properties of bases and H-mappings, but only particular cases. This improvement is due to the suggestion of a referee.

References

[1] H.P. Barendregt, *The Lambda Calculus*, 2nd edition (North-Holland, Amsterdam, 1984).
[2] H.P. Barendregt, M. Coppo and M. Dezani-Ciancaglini, A filter lambda model and the completeness of type assignment, *J. Symbolic Logic* **48** (1983) 931-940.
[3] M. Coppo, M. Dezani-Ciancaglini, F. Honsell and G. Longo, Extended type structures and filter lambda models, in: G. Longo et al., *Logic Colloquium '82* (North-Holland, Amsterdam, 1983) 241-262.
[4] M. Coppo, M. Dezani-Ciancaglini and B. Venneri, Functional characters of solvable terms, *Z. Math. Logik Grundlag. Math.* **27** (1981) 45-58.
[5] M. Coppo, M. Dezani-Ciancaglini and M. Zacchi, Type-theories, normal forms and D_∞-λ-models, *Inform. and Comput.* **72** (1987) 85-116.
[6] H.B. Curry, Grundlagen der kombinatorischen Logik, *Amer. J. Math.* **52** (1930) 509-536, 789-834.
[7] H.B. Curry, R. Feys, *Combinatory logic*, Vol. I (North-Holland, Amsterdam, 1958).
[8] A. Diller, *Compiling Functional Languages* (Wiley, Chichester, 1988).
[9] J.R. Hindley, Combinatory reductions and lambda reductions compared, *Z. Math. Logik Grundlag. Math.* **23** (1977) 169-180.
[10] J.R. Hindley, Standard and normal reductions, *Trans. Amer. Math. Soc.* **241** (1978) 253-271.
[11] J.R. Hindley, The simple semantics for Coppo-Dezani-Sallé types, Lecture Notes in Computer Science, Vol. 137 (Springer, Berlin, 1982) 212-226.
[12] J.R. Hindley, Types with intersection, an introduction, *Formal Aspects of Computing*, to appear.
[13] J.R. Hindley, J.P. Seldin, *Introduction to Combinators and λ-Calculus* (Cambridge University Press, Cambridge, 1986).
[14] J.R. Kennaway, The complexity of a translation of λ-calculus to combinators, Internal Report CSA/13/1984, University of East Anglia (1984).
[15] J.W. Klop, J.-J. Meijer, J. Rutten, eds, *J.W. de Bakker, 25 Jaar Semantiek*, Centrum voor Wiskunde en Informatica, Amsterdam (1989).
[16] J.C. Mulder, Complexity of combinatory code, Internal Report, University of Utrecht (1985).
[17] A. Piperno, Abstraction problems in combinatory logic: a compositive approach, *Theoret. Comput. Sci.* **66** (1989) 27-43.
[18] J.C. Reynolds, Preliminary design of the programming language Forsythe, Report CMU-CS-88-159, Computer Science Dept., Carnegie-Mellon University, Schenley Park, Pittsburgh, USA.
[19] S. Ronchi della Rocca, Characterization theorems for a filter lambda model, *Inform. and Control* **54** (1982) 201-216.
[20] S. Ronchi della Rocca, B. Venneri, Principal type schemes for an extended type theory, *Theoret. Comput. Sci.* **28** (1984) 151-171.
[21] M. Schönfinkel, Über die Bausteine der mathematischen Logik, *Math. Ann.* **92** (1924) 305-316.
[22] D.A. Turner, Another algorithm for bracket abstraction, *J. Symbolic Logic* **44** (1979) 267-270.
[23] D.A. Turner, Aspects of the implementation of programming languages: the compilation of an applicative language to combinatory logic, Ph.D. Thesis, University of Oxford (1981).
[24] C. Wadsworth, The relation between computational and denotational properties for D_∞-models of the lambda calculus, *SIAM J. Comput.* **5** (1976) 488-521.