Logique & Analyse **137-138** (1992), 69-79

COMBINATORY LOGIC AND LAMBDA CALCULUS WITH CLASSICAL TYPES

M.W. BUNDER

Type Assignment in Lambda Calculus

The primitive terms of lambda calculus are variables x, y, ... and possibly some constants. Other λ-terms are formed by application (if X and Y are λ-terms then (XY) is a λ-term) and lambda abstraction (If X is a λ-term and x a variable then (λx.X) is a λ-term).

(XY) can be interpreted as the value of the function X at Y and (λx.X) as the function whose value at x is X.

There is an equality relation between λ-terms which satisfies the following axiom (amongst others):

(β) $(\lambda x.M)N = [N/x]M$([1]).

Most λ-terms can be assigned an associated "type". The intention is that the type tells us what kind of term we have. We will describe types and the rules for assigning them to λ-terms.

Types are constructed from a class of type variables and/or certain type constants using the binary operation →. $X: \mathscr{A} \to \mathscr{B}$ will be read "If $U \in \mathscr{A}$ then $(XU) \in \mathscr{B}$."

Given an arbitrary assignment of types to the free variables (and constants) the types of compound λ-terms are usually assigned by the following rules:([2]).

([1]) [N/x]M is the result of substituting N for the free occurrences of x in M. For details of the definition and the full axioms and rules for equality see Hindley and Seldin [6].

([2]) For full details see Hindley and Seldin [6].

$$\rightarrow_i \quad \frac{\begin{array}{l}(\lambda)\\ x:\mathscr{A}\\ \vdots\\ Y:\mathscr{B}\end{array}}{(\lambda x.Y):\mathscr{A}\rightarrow\mathscr{B}} \quad -(1)$$

$$\rightarrow_e \quad \frac{X:\mathscr{A}\rightarrow\mathscr{B} \qquad Y:\mathscr{A}}{(XY):\mathscr{B}}$$

Here is a formal definition of a type of a λ-term and also of the principal type scheme of a λ-term.

Definition of Type and Principal Type Scheme (p.t.s.)

If X: $\mathscr{A}$ can be derived by $\rightarrow_e$ and $\rightarrow_i$ in a deduction that has no uncancelled hypotheses then X is said to have type $\mathscr{A}$.

Every λ-term that has a type has what is called a *principal type scheme* (p.t.s.) built up from type variables and →s . All other types of a λ-term X will be substitution instances of the p.t.s.

Note that some λ-terms such as (λx.λy.xyy)(λz.z) have no types assigned by the rules.

Type Assignment in Combinatory Logic

The primitive terms of a combinatory logic are a finite set of operators called combinators. Other combinators are formed by application.

Each combinator has associated with it an axiom; for example:

$$KXY = X$$
$$SXYZ = XZ(YZ)$$
$$IX = X$$

These combinators correspond to the following λ-terms: K to λx.λy.x, S to λx.λy.λz.xz(yz) and I to λx.x.

Given this identification and the type assignment rules of the lambda

calculus we can derive the following p.t.s.s :

$K : a \to (b \to a)$
and $S : (a \to (b \to c)) \to ((a \to b) \to (a \to c))$.

In this paper we will take these assignments of p.t.s.s to K and S as axioms and not use, directly, any of their substitution instances. To compensate for this we replace $\underline{\to}_e$ by the following rule:

Condensed Detachment (D)

If X: $\mathscr{A}$ and Y: $\mathscr{B}$ and σ_1 and σ_2 are the substitutions, of those for which $\sigma_1(\mathscr{A}) = \sigma_2(\mathscr{B}) \to \mathscr{C}$ has the least number of $\to$s , for which $\sigma_1(\mathscr{A})$ has the maximal number of distinct type variables, then (XY) : $\mathscr{C}$.
(This formulation is a somewhat more consice one than that in Hindley and Meredith [6].)

The type assignment axioms for K and S and rule (D) lead exactly to the p.t.s.s of combinators. In the same way p.t.s.s. of λ-terms can be generated using $\to$i and a generalised version of (D), provided that all hypotheses are initially of the form x:a.

Formulas as Types

A special case of (D), where v_1 and v_2 are identity substitutions, is given by:

$$\frac{X : \mathscr{A} \to \mathscr{B} \qquad Y : \mathscr{A}}{(XY) : \mathscr{B}}$$

which, ignoring the terms, looks like modus ponens if $\to$ is read as implication and $\mathscr{A}$ and $\mathscr{B}$ are formulas rather than types.

As the types for K and S are exactly the usual axioms for intuitionistic logic and (D) encorporates some substitution and modus ponens it is not surprising (for details see Hindley and Meredith [5]) that the types are exactly the theorems of implicational intuitionistic propositional logic.

Classical Theorems as Types

The aim of this paper is to extend the type theory and if necessary the combinatory logic or lambda calculus in such a way that the type theory becomes a full classical propositional logic rather than just an intuitionistic implicational one.

The extension to classical logic can be achieved most simply by adding a constant proposition f and an axiom:

$$((a \to f) \to f) \to a.$$

The usual logical connectives can then be defined by:

$$\sim \mathscr{A} = \mathscr{A} \to f$$

$$\mathscr{A} \vee \mathscr{B} = \sim \mathscr{A} \to \mathscr{B}$$

and

$$\mathscr{A} \wedge \mathscr{B} = \sim(\mathscr{A} \to \sim \mathscr{B}).$$

We will assign a "new" combinator or λ-term U the above type as p.t.s. i.e.

$$U : ((a \to f) \to f) \to a$$

The constant f seen in type terms, will be an empty type. What U does as a combinator will be investigated later.

First we will look at certain classical theorems and their corresponding (extended) combinators. We will use some further combinators (definable using S and K) and their types (derivable from those for S and K):

$IX = X$		$I = SKK$
$BXYZ = X(YZ)$	where	$B = S(KS)K$
$CXYZ = XZY$	"	$C = S(BBS)(KK)$
$WXY = XYY$	"	$W = CSI$

I : $a \to a$

B : $(a \to b) \to ((c \to a) \to (c \to b))$

C : $(a \to (b \to c)) \to (b \to (a \to c))$

W : $(a \to (a \to b)) \to (a \to b)$.

Using (D) we obtain successively:

$$BU : (c \to ((a \to f) \to f)) \to (c \to a)$$
$$BUK : f \to a$$

Thus the p.t.s. of *BUK* is the well known theorem that false implies all propositions. Note also that *BUK* gives us exactly the proof of $f \to a$ using the axioms or theorems *B*, *U*, and *K* and rule (D).

The classical theorem $a \wedge b \to a$ can be written as

$$((a \to (b \to f)) \to f) \to a.$$

Its proof given by *BU*(*CB*(*BK*)) can be written out as follows using (D).

B:	$(a \to b) \to ((c \to a) \to (c \to b))$
K:	$e \to (d \to e)$
BK:	$(c \to e) \to (c \to (d \to e))$
C:	$(g \to (h \to i)) \to (h \to (g \to i))$
CB:	$(c \to a) \to ((a \to b) \to (c \to b))$
CB(*BK*):	$((c \to (d \to e)) \to b) \to ((c \to e) \to b)$
U:	$((j \to f) \to f) \to j$
BU:	$(k \to ((j \to f) \to f)) \to (k \to j)$
BU(*CB*(*BK*)):	$((c \to (d \to f)) \to f) \to c$

The following are the results of the substitutions induced by the use of (D) in the above proof.

$$k = (c \to (d \to e)) \to b,$$
$$j = c,$$
$$f = e = b$$

The theorem of classical implicational logic (Peirce's law) which reduces intuitionistic logic to classical logic corresponds to the following combinator:

$$BU\{C[BS(CB)]\ [B(BUK)]\} : ((a \to b) \to a) \to a\ .$$

Proof Transformations

Combinators (or equivalently λ-terms) do not only give a precise representation of a proof, they also allow a proof to be transformed easily into one where the more basic axioms (of some weaker system) are used first and ones needed to extend the system to a stronger one as late as possible. We will treat the p.t.s.s of I, B, C, K and W as potential axioms for these weaker systems.

The proof of $a \wedge b \rightarrow a$ can be rewritten as follows using the properties of B and C:

$$\begin{aligned} BU(CB(BK)) &= CB(CB(BK))U \\ &= CB(B(CB)BK)U \\ &= B(CB)(B(CB)B)KU. \end{aligned}$$

As the equations for B and C preserve type (see Curry and Feys [3]) the combinator $B(CB)(B(CB)B)KU$ also represents a proof of $a \wedge b \rightarrow a$. Its initial part $B(CB)(B(CB)B)$ represents the proof of a theorem in the very weak logic based on the p.t.s.s of B and C and Rule (D) and the proof is completed by just two applications of (D) with respectively the types of K and U as minor premises. Note that as W is not used this is a theorem of a "classical BCK logic".

We can summarise results of this kind in the following theorem.

Theorem 1 Any proof in a (sub-) classical logic having at least B and C as theorems can be reorganised to the form:

$$ZI \ldots IK \ldots KW \ldots WU \ldots U \qquad -(1)$$

where Z is a BC combinator and each of I, K, W and U will appear zero or more times.

Proof A proof can be represented by a combinator involving zero or more of B, C, K, I, W and U . Using B and C this combinator can be rewritten in the form (1). The two combinators will be equal and will as above have the same p.t.s. i.e. prove the same theorem.

Note 1. In a logic that has B, C and K, I can be defined so the I's can be left out of (1).

Note 2. $ZI...I$ represents a proof in a *BCI* (or linear) logic. Logics similar to this have been studied by C.A. Meredith, J.Y. Girard and others.

$ZI...IK...K$ represents a proof in a *BCK* logic, versions of which have been studied by Meredith, Ono and Komori and the author.

$ZI...IK...KW...W$ represents an intuitionistic proof, $ZI...IW...W$ a proof in the relevance logic $R_{\rightarrow}$.

In each case the *I*s, *K*s *W*s and *U*s are minor premises in (D) steps.

Note 3. The order of the combinators in (1) can be altered to for example $Z_1\ I...IK...KU...UW...W$, so that the theorems proved are first *BC* then *BCI* then *BCU* then "classical *BCK*" and then classical. Also we could have $Z_2\ I...IU...UW...WU...U$ where an intuitionistic theorem is proved before the classical etc.

Note 4. It might be thought that using W (1) could be simplified even further to Z_3 IK W ...WU , but while this is true for the combinator, a W step does not necessarily preserve type. III for example has type $a \rightarrow a$, but WII has no type.

Note 5. It was shown in [1] and [2] and also by Hindley in [4] that every *BCK*-term has a p.t.s. However not every *BCKW* or *BCKU* term has a p.t.s. A term $ZI...IK...K$, if *Z* is a *BC* term, therefore always represents a proof, but given a term of type (1) one of the *W* or *U* (D) steps may not be able to be carried out.

U as a Combinator

One choice for the "new" combinator *U* is a *unifier* with the property

$$UX = T$$

where *T* is a *terminator* combinator with the property

$$TZ = T.$$

U can clearly be defined as *KT*.

T can be defined as *YK* where *Y* is the paradoxical (or fixed point) combinator defined by *WS*(*BWB*) with property

$$YX = X(YX).$$

Alternatively T can be the λ-term (λxy.xx)(λxy.xx) and U the λ-term λz.((λxy.xx)(λxy.xx)).

Types for U and T by Type Assignment

If Rule (D) and the axioms for K and S are used to assign types, neither Y, T nor U will have a type. In fact any reasonable type assigned, as an axiom, to Y will cause the inconsistency of the corresponding propositional logic. For example the choice Y: (a → a) → a gives YI : a, which in terms of the logic gives an arbitrary propositional variable as a theorem. This problem fortunately need not arise with T (nor obviously with U as classical logic is consistent). In fact the following is sufficient as a type assignment rule for T:

$$Ti \qquad \frac{X : \mathscr{A}}{T : \mathscr{A}}$$

The rule says that:

Theorem 2 T has every nonempty type.

Also we have:

Theorem 3 KT has every nonempty type.

Proof Every nonempty type is of the form $\mathscr{A} \to \mathscr{B}$. Thus for some term X:

$$\dfrac{\dfrac{\dfrac{X : \mathscr{A} \to \mathscr{B} \qquad x : \mathscr{A}}{Xx : \mathscr{B}}\,(I)}{T : \mathscr{B}}\,\text{-(1)}}{KT : \mathscr{A} \to \mathscr{B}}$$

This however does not give KT the type $((\mathscr{A} \to f) \to f) \to \mathscr{A}$ unless we

know this to be nonempty.

The further rule:

$$fe \qquad \frac{X : f}{Y : \mathscr{A}}$$

allows us to prove:

Theorem 4 KT: $((\mathscr{A} \to \mathscr{B}) \to \mathscr{B}) \to \mathscr{A}$

(i) If $\mathscr{A}$ is nonempty.

(ii) If $\mathscr{A}$ is f.

Proof

(i) If $\mathscr{A}$ is a nonempty type then so is $\mathscr{B} \to \mathscr{A}$ for any $\mathscr{B}$. Hence by Theorem 3

$$KT : ((\mathscr{A} \to f) \to f) \to \mathscr{A}.$$

(ii)

$$\frac{I : f \to f \qquad\qquad x : (f \to f) \to f}{xI : f}\ (I)$$

$$\frac{xI : f}{T : f}$$

$$KT : ((f \to f) \to f) \to f .$$

The type assignment for KT (=U) can be derived generally (as above) if the following rule is added:

$$\frac{\begin{array}{l} X : \mathscr{A} \\ \quad\vdots \\ Y : \mathscr{B} \end{array} \qquad\qquad \begin{array}{l} Z : \mathscr{A} \to f \\ \quad\vdots \\ Y : \mathscr{B} \end{array}}{Y : \mathscr{B}}\ -(1)\ (2)$$

As $Z : \mathscr{A} \to f$ only if $\mathscr{A}$ is f, we can read this as:

If $Y : \mathscr{B}$ follows if $\mathscr{A}$ is nonempty ($X : \mathscr{A}$) and if $\mathscr{A}$ is empty

(Z : $\mathscr{A}\rightarrow$ f) then Y : $\mathscr{B}$ holds.

It can therefore be seen as an "or elimination" rule using excluded middle (P or not P).

Types as sets

If { $A_1,\ldots, A_n$} is a set of basis types and J is the closure of this set under $\rightarrow$, the standard set of types of say *K* could be seen as

$$\{(\mathscr{A}\rightarrow (\mathscr{B}\rightarrow \mathscr{A}) \mid \mathscr{A}, \mathscr{B} \in \mathrm{J}\},$$

and that of I as

$$\{\mathscr{A}\rightarrow \mathscr{A} \mid \mathscr{A} \in \mathrm{J}\} .$$

In the extended type theory J , ϕ will be a basis type and *every* old type will be augmented by $\{T\}$. The new sets of types for K and I will therefore be

$$\{\{T\}\cup(\mathscr{A}\rightarrow (\mathscr{B}\rightarrow \mathscr{A})) \mid \mathscr{A}, \mathscr{B} \in \mathrm{J}\}$$

and

$$\{\{T\}\cup(\mathscr{A}\rightarrow \mathscr{A}) \mid \mathscr{A} \in \mathrm{J}\}$$

In this set of sets T is therefore a *universal element*.

The University of Wollongong

References

[1] Bunder, M.W., "Corrections to some results for *BCK* logics and algebras" Logique et Analyse, Vol. 31 (1991) pp 115-122.

[2] Bunder, M.W. and Meyer, R.K. "A result for combinators, *BCK* logics and *BCK* algebras" Logique et Analyse, Vol. 109 (1985) pp 33-40.

[3] Curry, H.B. and Feys, R. Combinatory Logic, Vol. 1, Amsterdam, North Hollland (1958).

[4] Hindley, J.R. "*BCK*-combinators and linear λ-terms have types" Theoretical Computer Science Vol. 64 (1989) pp 97-105.
[5] Hindley, J.R. and Meredith D. "Principal type-schemes and condensed detachment" Journal of Symbolic Logic, Vol. 55 (1990) pp 90-105.
[6] Hindley, J.R. and Seldin, J.P. Introduction to Combinators and λ-Calculus, Cambridge University Press (1986).