2000

Expedited Broda-Damas bracket abstraction

Martin W. Bunder
University of Wollongong, mbunder@uow.edu.au

Expedited Broda-Damas bracket abstraction

Abstract

A bracket abstraction algorithm is a means of translating λ-terms into combinators. Broda and Damas, in [1], introduce a new, rather natural set of combinators and a new form of bracket abstraction which introduces at most one combinator for each λ-abstraction. This leads to particularly compact combinatory terms. A disadvantage of their abstraction process is that it includes the whole Schonfinkel [4] algorithm plus two mappings which convert the Schonfinkel abstract into the new abstract. This paper shows how the new abstraction can be done more directly, in fact, using only 2n - 1 algorithm steps if there are n occurrences of the variable to be abstracted in the term. Some properties of the Broda-Damas combinators are also considered.

Keywords

bracket, damas, expedited, abstraction, broda

Disciplines

Engineering | Science and Technology Studies

Publication Details

Bunder, M. W. (2000). Expedited Broda-Damas bracket abstraction. The Journal of Symbolic Logic, 65 (4), 1850-1857.

THE JOURNAL OF SYMBOLIC LOGIC
Volume 65. Number 4. Dec. 2000

EXPEDITED BRODA-DAMAS BRACKET ABSTRACTION

M. W. BUNDER

Abstract. A bracket abstraction algorithm is a means of translating λ-terms into combinators. Broda and Damas, in [1], introduce a new, rather natural set of combinators and a new form of bracket abstraction which introduces at most one combinator for each λ-abstraction. This leads to particularly compact combinatory terms. A disadvantage of their abstraction process is that it includes the whole Schönfinkel [4] algorithm plus two mappings which convert the Schönfinkel abstract into the new abstract. This paper shows how the new abstraction can be done more directly, in fact, using only $2n - 1$ algorithm steps if there are n occurrences of the variable to be abstracted in the term. Some properties of the Broda-Damas combinators are also considered.

§1. The Broda-Damas combinators. The class of Broda-Damas combinators $\mathfrak{K}$ and the class of combinatory terms $\mathfrak{C}l$ are defined below.

DEFINITION 1. The set of combinator indices $\mathfrak{A}$

(i) The empty word $\emptyset \in \mathfrak{A}$
(ii) $\alpha, \beta \in \mathfrak{A} \Rightarrow c \cdot \alpha, b \cdot \alpha, (\alpha, \beta) \in \mathfrak{A}$.

Thus $b, c, b \cdot c,\ c \cdot b \cdot b, (b \cdot c, (c, b) \cdot b)$ etc are combinator indices.

If $\alpha \in \mathfrak{A}$ we let $\#\alpha$ be the number of b's and c's in α.
The indices b and c are related to the standard combinators **B** and **C**.

DEFINITION 2. The class of Broda-Damas (or B-D) combinators $\mathfrak{K}$

(i) $\boldsymbol{K} \in \mathfrak{K}$
(ii) $\alpha \in \mathfrak{A} \Rightarrow \Phi_\alpha \in \mathfrak{K}$.

$\Phi_{b.b}, \Phi_{b.c}$ etc will often be written as Φ_{bb}, Φ_{bc} etc.

In the following we assume that we have a class of variables V.

DEFINITION 3. The class of combinatory terms $\mathfrak{C}l$

(i) $x \in V \Rightarrow x \in \mathfrak{C}l$
(ii) $\mathfrak{K} \subseteq \mathfrak{C}l$
(iii) $X, Y \in \mathfrak{C}l \Rightarrow (XY) \in \mathfrak{C}l$.

As usual we assume association to the left for combinatory terms.

Weak reduction is defined by the following axioms as well as the usual rules:

DEFINITION 4. Weak reduction $\triangleright_{1w}$ is given by:

Received April 22, 1999; accepted July 29, 1999.

0022-4812/00/6504-0023/$1.80

$\mathbf{K}$ $\quad \boldsymbol{K}XY \rhd_{1w} X$

$\Phi_\emptyset$ $\quad \Phi_\emptyset X \rhd_{1w} X$

$\Phi_{c\cdot\alpha}$ $\quad \Phi_{c\cdot\alpha}XX_1 \dots X_{\#\alpha+1} \rhd_{1w} \Phi_\alpha X_1 \dots X_{\#\alpha+1}X$

$\Phi_{b\cdot\alpha}$ $\quad \Phi_{b\cdot\alpha}XX_1 \dots X_{\#\alpha+1} \rhd_{1w} X(\Phi_\alpha X_1 \dots X_{\#\alpha+1})$

$\Phi_{(\alpha.\beta)}$ $\quad \Phi_{(\alpha.\beta)}X_1 \dots X_{\#\alpha}Y_1 \dots Y_{\#\beta}Z \rhd_{1w} \Phi_\alpha X_1 \dots X_{\#\alpha}Z(\Phi_\beta Y_1 \dots Y_{\#\beta}Z).$

$\rhd_w$ is the reflexive, transitive closure of $\rhd_{1w}$ and $=_w$ the symmetric closure of $\rhd_w$.

§2. The Broda-Damas bracket abstraction algorithm. Lambda terms are defined by:

DEFINITION 5. The class of λ-terms Λ

(i) $V \subseteq \Lambda$
(ii) $M, N \in \Lambda \Rightarrow (MN) \in \Lambda$
(iii) $x \in V \& M \in \Lambda \Rightarrow (\lambda x.M) \in \Lambda$.

These can be transformed into combinators by the transformation $(\ )_H$ defined in terms of a bracket abstraction $\lambda^* x$:

DEFINITION 6. $(\ )_H$

$$(\ )_H : \Lambda \to \mathfrak{Cl}$$
$$\text{where } (x)_H \equiv x,$$
$$(MN)_H \equiv M_H N_H,$$
$$\text{and } (\lambda x.M)_H \equiv \lambda^* x.M_H.$$

Definition 6 applies for most translations from λ-terms into combinators, in [1] the bracket abstraction $\lambda^* x$ is defined using the Schönfinkel [4] algorithm and two mappings.

DEFINITION 7. $[x]$ (Schönfinkel bracket abstraction)

(i) $[x]x \equiv \boldsymbol{I}$
(k) $[x]U \equiv \boldsymbol{K}U \quad$ if $x \notin U$
(η) $[x]Ux \equiv U \quad$ if $x \notin U$
(b) $[x]UV \equiv \boldsymbol{B}U([x]V) \quad$ if $x \notin U$
(c) $[x]UV \equiv \boldsymbol{C}([x]U)V \quad$ if $x \notin V$
(s) $[x]UV \equiv \boldsymbol{S}([x]U)([x]V)$
$[x_1, \dots, x_n]X \equiv [x_1, \dots, x_{n-1}]([x_n]X)$.

Here $\boldsymbol{I}, \boldsymbol{K}, \boldsymbol{B}, \boldsymbol{C}$ and $\boldsymbol{S}$ are the standard Schönfinkel or Curry (see Curry and Feys [3]) combinators, with weak reduction given by:

(I) $\boldsymbol{I}X \rhd X$
(K) $\boldsymbol{K}XY \rhd X$
(B) $\boldsymbol{B}XYZ \rhd X(YZ)$
(C) $\boldsymbol{C}XYZ \rhd XZY$
(S) $\boldsymbol{S}XYZ \rhd XZ(YZ)$

i, which maps Schönfinkel combinatory terms into $\mathfrak{A}$, is given by:

DEFINITION 8. i

$$\begin{aligned} i(\boldsymbol{I}) &\equiv \emptyset \\ i(\boldsymbol{C}XY) &\equiv c \cdot i(X) \\ i(\boldsymbol{B}XY) &\equiv b \cdot i(Y) \\ i(\boldsymbol{S}XY) &\equiv (i(X), i(Y)) \\ iX &\equiv b \qquad \text{otherwise.} \end{aligned}$$

Another function r maps these terms into a sequence of $\mathfrak{C}l$ terms.

DEFINITION 9. r

$$\begin{aligned} r(\boldsymbol{I}) &\equiv \emptyset \\ &\text{where } \emptyset \text{ represents the empty sequence} \\ r(\boldsymbol{C}XY) &\equiv Y, r(X) \\ r(\boldsymbol{B}XY) &\equiv X, r(Y) \\ r(\boldsymbol{S}XY) &\equiv r(X), r(Y) \\ r(X) &\equiv X \qquad \text{otherwise.} \end{aligned}$$

We can now define $\lambda^* x.X$.

DEFINITION 10. $\lambda^* x.X$

(i) If $[x]X \in \mathfrak{C}l$ (i.e. it doesn't contain $\boldsymbol{I}, \boldsymbol{B}, \boldsymbol{C}$ or $\boldsymbol{S}$) then

$$\lambda^* x.X \equiv [x]X.$$

(ii) If $[x]X \notin \mathfrak{C}l$, $\alpha = i([x]X)$ and $X_1, \ldots, X_n \equiv r([x]X)$ then

$$\lambda^* x.X \equiv \Phi_\alpha X_1 \ldots X_n.$$

Most abstracts are simpler in terms of B-D combinators, a few are not.

EXAMPLE 11.

$$\begin{aligned} [x].xyz &\equiv \boldsymbol{C}(\boldsymbol{CI}y)z \\ i([x].xyz) &\equiv c \cdot c \\ r([x].xyz) &\equiv z, y \\ \text{so by Definition 10(ii):} \quad \lambda^* x.xyz &\equiv \Phi_{c\cdot c} zy \\ \text{by Definition 10(i):} \quad \lambda^* zyx.xyz &\equiv \Phi_{c\cdot c} \\ \text{while} \quad [z, y, x].xyz &\equiv \boldsymbol{C}(\boldsymbol{BC}(\boldsymbol{CI})) \end{aligned}$$

EXAMPLE 12.

$$\begin{aligned} [z]y(xz) &\equiv \boldsymbol{B}yx \\ i([z]y(xz)) &\equiv b \cdot b \\ r([z]y(xz)) &\equiv y, x \\ \text{so} \quad \lambda^* z.y(xz) &\equiv \Phi_{bb} yx \\ [y]\lambda^* z.y(xz) &\equiv \boldsymbol{C}\Phi_{bb} x \\ i([y]\lambda^* z.y(xz)) &\equiv c \cdot b \\ r([y]\lambda^* z.y(xz)) &\equiv x, \Phi_{bb} \end{aligned}$$

$$\lambda^* yz.y(xz) \equiv \Phi_{cb} x \Phi_{bb}$$
$$\text{Similarly} \quad \lambda^* xyz.y(xz) \equiv \Phi_{cb}\Phi_{bb}\Phi_{cb}$$
$$\text{while} \quad [x,y,z]y(xz) \equiv \boldsymbol{CB}$$

§3. An expedited algorithm. Definition 7 is simple and convenient because it defines $[x]X$ in terms of bracket abstractions of simpler terms. We can do the same for $\lambda^* x.X$.

THEOREM 13. *If* $x \in U, x \in V$, $x \notin T$, $\lambda^* x.U \equiv \Phi_\alpha X_1 \dots X_{\#\alpha}$ *and* $\lambda^* x.V \equiv \Phi_\beta Y_1 \dots Y_{\#\beta}$ *then*:

(i) $\lambda^* x.TU \equiv \Phi_{b\cdot\alpha} T X_1 \dots X_{\#\alpha}$
(ii) $\lambda^* x.UT \equiv \Phi_{c\cdot\alpha} T X_1 \dots X_{\#\alpha}$
(iii) $\lambda^* x.UV \equiv \Phi_{(\alpha,\beta)} X_1 \dots X_{\#\alpha} Y_1 \dots Y_{\#\beta}$.

If $\lambda^* x.U$ *is not of the above form we have* (*i*), (*ii*) *and* (*iii*) *with* $\alpha \equiv b$ *and* $X_1 \equiv \lambda^* x.U$. *If* $\lambda^* x.V$ *is not of the above form we have* (*iii*) *with* $\beta = b$ *and* $Y_1 \equiv \lambda^* x.V$.

PROOF. (i) $[x]TU \equiv \boldsymbol{B}T([x]U)$.
If $\lambda^* x.U$ has the above form then

$$i([x]TU) \equiv b \cdot i([x]U) = b \cdot \alpha$$
$$\text{and } r([x]TU) \equiv T, r([x]U) \equiv T, X_1, \dots, X_{\#\alpha}$$

If $\lambda^* x.U$ is not of that form

$$i([x]TU) \equiv b \cdot b$$
$$\text{and } r([x]TU) \equiv T, [x]U$$

So in the former case $\lambda^* x.TU$ is as in (i) and in the latter case as in (i) but with $\alpha \equiv b$ and $X_1 \equiv \lambda^* x.U$.
(ii) and (iii) are similar. ⊣

In bracket abstraction, as defined in Theorem 7, the first clause used determines the leftmost combinator of the abstract (if any) and later uses of clauses determine other combinators. In the abstraction given by Theorem 13, where only one combinator is formed, this combinator is being enhanced with further subscripts at each use of a clause after the first.

EXAMPLE 14. By Theorem 13(i) and (ii):

$$\lambda^* x.z(xyz) \equiv \Phi_{b\cdot\alpha} z X_1 \dots X_{\#\alpha}$$
$$\text{where} \quad \lambda^* x.xyz \equiv \Phi_\alpha X_1 \dots X_{\#\alpha}$$
$$\lambda^* x.xyz \equiv \Phi_{c\cdot\beta} z Y_1 \dots Y_{\#\beta}$$
$$\text{where} \quad \lambda^* x.xy \equiv \Phi_\beta Y_1 \dots Y_{\#\beta}$$
$$\lambda^* x.xy \equiv \Phi_{c\cdot\gamma} y Z_1 \dots Z_{\#\gamma}$$
$$\text{where} \quad \lambda^* x.x \equiv \Phi_\gamma Z_1 \dots Z_{\#\gamma}.$$

Then $\gamma \equiv \emptyset$, $\#\gamma = 0$, $\beta \equiv c$, $Y_1 \equiv y$, $\#\beta = 1$, $\alpha = c \cdot c$ and $X_1 \equiv z, X_2 \equiv y$.

So

$$\lambda^* x.z(xyz) \equiv \Phi_{b\cdot c\cdot c} zzy.$$

This process can be carried out more simply in reverse, by noting that the simplest directly abstractable forms are, for $x \notin Y$:

$$\lambda^* x.xY \equiv \Phi_c Y$$
$$\text{and} \qquad \lambda^* x.Yx \equiv Y.$$

Example 14 now becomes:

$$\lambda^* x.xy \equiv \Phi_c y$$
$$\text{then by Theorem 13 (ii),} \qquad \lambda^* x.xyz \equiv \Phi_{c\cdot c} zy$$
$$\text{and by Theorem 13 (i),} \qquad \lambda^* x.z(xyz) \equiv \Phi_{b\cdot c\cdot c} zzy.$$

This example makes it clear that Theorem 13 can be used as part of an algorithm, provided we abstract from the inside of a term outwards. Such an algorithm is described below.

The first expedited Broda-Damas algorithm (EBDA1)

AIM. To evaluate the B-D abstract $\lambda^* x.X$.

METHOD. STEP 1. (i) If $x \notin Y$ and $X \equiv Yx$ then $\lambda^* x.X \equiv Y$
(ii) $x \notin X$ then $\lambda^* x.X \equiv \boldsymbol{K}X$
(iii) if $X \equiv x$ then $\lambda^* x.X \equiv \Phi_\emptyset$.

STEP 2. For any parts UT, TU or UV of X where $x \in U$, $x \in V$ and $x \notin T$, if

$$\lambda^* x.U \equiv \Phi_\alpha X_1 \ldots X_{\#\alpha}$$
$$\text{and} \qquad \lambda^* x.V \equiv \Phi_\beta Y_1 \ldots Y_{\#\beta}$$
$$\text{then (i)} \qquad \lambda^* x.TU \equiv \Phi_{b\cdot\alpha} TX_1 \ldots X_{\#\alpha}$$
$$\text{(ii)} \qquad \lambda^* x.UT \equiv \Phi_{c\cdot\alpha} TX_1 \ldots X_{\#\alpha}$$
$$\text{(iii)} \qquad \lambda^* x.UV \equiv \Phi_{(\alpha.\beta)} X_1 \ldots X_{\#\alpha} Y_1 \ldots Y_{\#\beta}.$$

If $\lambda^* x.U$ is not of the above form $\alpha \equiv b$ and $X_1 \equiv \lambda^* x.U$ in (i), (ii) and (iii). If $\lambda^* x.V$ is not of the above form $\beta \equiv b$ and $Y_1 \equiv \lambda^* x.V$ in (iii).

Step 2 is repeated till it applies to X.

It is clear from Theorem 13 and the definition of $\lambda^* x.X$ that:

THEOREM 15. *EBDA*1 *evaluates* $\lambda^* x.X$.

The new algorithm has the advantage that if x appears in disjoint subterms of X, the abstractions with respect to x of these subterms can be done in parallel.

EXAMPLE 16. Evaluate $\lambda^* x.x(xy)(zxy)$.

In parallel: $\lambda^* x.xy \equiv \Phi_c y$, $\lambda^* x.x \equiv \Phi_\emptyset$ and $\lambda^* x.zx \equiv z$.

Then in parallel: $\lambda^* x.zxy \equiv \Phi_{c\cdot b} yz$
and $\lambda^* x.x(xy) \equiv \Phi_{(\emptyset.c)} y$.
Then $\lambda^* x.x(xy)(zxy) \equiv \Phi_{((\emptyset.c),c\cdot b)} yyz$.

The algorithm can be made to work faster still if we combine several uses of Step 2 in EBDA1. For example if $x \notin X_1 \ldots X_n$, we have:

$$(1) \qquad \lambda^* x.xX_1 \ldots X_n \equiv \Phi_{c^n} X_n X_{n-1} \ldots X_1$$

where Φ_{c^n} stands for $\Phi_{c\cdot c\ldots c}$ where there are n c's in the subscript.

Similarly if $x \notin Y_1 \dots Y_n$, we have:

$$\lambda^* x.Y_1(Y_2 \dots Y_{n-1}(Y_n x)\dots) \equiv \Phi_{b^n} Y_1 \dots Y_n.$$

The second expedited Broda-Damas algorithm (EBDA2)

AIM. To evaluate the B-D abstract $\lambda^* x.X$.

METHOD. STEP 1. If $x \notin X$ then $\lambda^* x.X \equiv \boldsymbol{K}X$, otherwise consider all parts Z of X which have only one occurrence of x. Such a Z takes the form:

$$Z \equiv Y_k(\dots(Y_2(Y_1(xX_1 \dots X_n)Y_{11} \dots Y_{1m_1})Y_{21} \dots Y_{2m_2})\dots Y_{km_k}$$

then

$$\lambda^* x.Z \equiv \Phi_{c^{m_k}\cdot b\dots c^{m_2}\cdot b\cdot c^{m_1}\cdot b\cdot c^n} Y_{km_k} \dots Y_{k1} Y_k \dots Y_{2m_2} \dots Y_{21} Y_2 Y_{1m_1} \dots Y_{11} Y_1 X_n \dots X_1$$

unless $n = 0$, $k = 1$ and $m_1 = 0$, in which case $\lambda^* x.Z \equiv Y_1$.

STEP 2. For parts Z of X where

$$Z \equiv Y_k(\dots(Y_2(Y_1(UVX_1 \dots X_n)Y_{11} \dots Y_{1m_1})Y_{21} \dots Y_{2m_2})\dots Y_{km_k}$$

where $k \geq 0$, each $m_i \geq 0$, x appears only in UV;

$$\lambda^* x.U \equiv \Phi_\alpha W_1 \dots W_{\#\alpha},$$

and $$\lambda^* x.V \equiv \Phi_\beta R_1 \dots R_{\#\beta},$$

have been evaluated by Step 1 or a previous Step 2,

$$\lambda^* x.Z \equiv \Phi_{c^{m_k}\cdot b\dots c^{m_2}\cdot b\cdot c^{m_1}\cdot c^n\cdot(\alpha,\beta)} Y_{km_k} \dots Y_{k_1} Y_k \dots Y_{2m_2} \dots Y_{21} Y_2 Y_{1m_1} \dots Y_{11} Y_1 X_n \dots X_1 W_1 \dots W_{\#\alpha} R_1 \dots R_{\#\beta}.$$

NOTE. 1. A special case of Step 1 is

$$\lambda^* x.x \equiv \Phi_\emptyset.$$

2. Abstractions of disjoint subterms Z of X as in Steps 1 and 2 can be done in parallel.

EXAMPLE 17.

$$\lambda^* x.uv(uy(xyuv)wv)uvw \equiv \Phi_{c^3\cdot b\cdot c^2\cdot b\cdot c^3} wvu(uv)vw(uy)vuy$$

$$\lambda^* x.u(xvv)ww \equiv \Phi_{c^2\cdot b\cdot c^2} wwuvv.$$

So

$$\lambda^* x.uv(uy(xyuv)wv)uvw(u(xvv)ww) \equiv \Phi_{(c^3\cdot b\cdot c^2\cdot b\cdot c^3, c^2\cdot b\cdot c^2)} wvu(uv)vw(uy)vuywwuvv.$$

The following lemma proves the correctness of Steps 1 and 2 of EBDA2.

LEMMA 18. *If* $x \notin Y_1 \dots Y_k X_1 \dots X_n Y_{11} \dots Y_{k1} \dots Y_{km_k}$ *and*

$$\lambda^* x.W \equiv \Phi_\alpha Z_1 \dots Z_{\#\alpha},$$

then unless $n = 0$, $k = 1$, $m_1 = 0$ *and* $W \equiv x$,

$$\lambda^* x.Y_k(\dots(Y_2(Y_1(WX_1 \dots X_n)Y_{11} \dots Y_{1m_1})Y_{21} \dots Y_{2m_2})\dots Y_{km_k} \equiv \Phi_{c^{m_k}\cdot b\dots c^{m_1}\cdot b\cdot c^n\cdot\alpha} Y_{km_k} \dots Y_{2m_2} \dots Y_{21} Y_2 Y_{1m_1} \dots Y_{11} Y_1 X_n \dots X_1 Z_1 \dots Z_{\#\alpha}.$$

PROOF. By induction on k.

By EDBA1,

$$\lambda^* x.WX_1 \dots X_n \equiv \Phi_{c^n \cdot \alpha} X_n \dots X_1 Z_1 \dots Z_{\#\alpha},$$

the required result for $k = 0$.

For $k > 0$, unless $n = 0$ and $W \equiv x$,

$$\lambda^* x.Y_1(WX_1 \dots X_n) \equiv \Phi_{b \cdot c^n \cdot \alpha} Y_1 X_n \dots X_1 Z_1 \dots Z_{\#\alpha},$$

and, unless $n = 0$, $W \equiv x$ and $m_1 = 0$,

$$\lambda^* x.Y_1(WX_1 \dots X_n) Y_{11} \dots Y_{1m_1} \equiv \Phi_{c^{m_1} \cdot b \cdot c^n \cdot \alpha} Y_{1m_1} \dots Y_{11} Y_1 X_n \dots X_1 Z_1 \dots Z_{\#\alpha}.$$

For any n, W and m_1 we have

$$\lambda^* x.Y_2(Y_1(WX_1 \dots X_n) Y_{11} \dots Y_{1m_1}) \equiv \Phi_{b \cdot c^{m_1} \cdot b \cdot c^n \cdot \alpha} Y_2 Y_{1m_1} \dots Y_{11} Y_1 X_n \dots X_1 Z_1 \dots Z_{\#\alpha}.$$

If we assume the result for $k = r > 1$, we have by Theorem 13(i):

$$\lambda^* x.Y_{r+1}(Y_r \dots Y_1(WX_1 \dots X_n) \dots Y_{rm_r}) \equiv \Phi_{b \cdot c^{m_r} \cdot b \dots b \cdot c^n \cdot \alpha} Y_{r+1} Y_{rm_r} \dots Y_{r1} Y_r \dots Y_1 X_n \dots X_1 Z_1 \dots Z_{\#\alpha}$$

and by Theorem 13(ii) m_{r+1} times:

$$\lambda^* x.Y_{r+1}(Y_r \dots Y_1(WX_1 \dots X_n) \dots Y_{rm_r}) Y_{r+11} \dots Y_{r+1m_{r+1}} \equiv \Phi_{c^{m_{r+1}} \cdot b \cdot c^{m_r} \dots b \cdot c^n \cdot \alpha} Y_{r+1m_{r+1}} \dots Y_{r+11} Y_{r+1} Y_{rm_r} \dots Y_r \dots Y_1 X_n \dots X_1 Z_1 \dots Z_{\#\alpha}.$$

So the result holds. ⊣

EBDA2 requires far fewer algorithm steps for evaluations of $\lambda^* x.X$ than EBDA1.

THEOREM 19. *EBDA2 evaluates $\lambda^* x.X$ with, if there are $n(> 0)$ occurrences of x in X, exactly $2n - 1$ uses of Steps 1 and 2.*

PROOF. Lemma 10 confirms that EBDA2 evaluates $\lambda^* x.X$. The rest of the theorem is proved by induction on n.

If X is in the form (of Z) given in EBDA2 Step 1, x appears once in X and only one Step 1 is required to evaluate $\lambda^* x.X$. The only other case of $n = 1$ has $X \equiv Y_1 x$ $(x \notin Y_1)$ and again only one use of Step 1 is required.

If X is in the form (of Z) given in EBDA2 Step 2, where there are k occurrences of x in U and $n - k$ in V $(0 < k < n)$, there are, by the induction hypothesis altogether $2k - 1 + 2(n - k) - 1 + 1 = 2n - 1$ uses of Steps 1 and 2 required. ⊣

NOTE. The method employed in EBDA2 is not restricted to B-D abstraction. We can construct a similar algorithm for Schönfinkel abstraction by using:

If $[x]W \equiv \boldsymbol{S}UV$ and $x \notin Y_1 \dots Y_k X_1 \dots X_n Y_{11} \dots Y_{k_1} \dots Y_{km_k}$ then

$$[x]Y_k(\dots(Y_2(Y_1(WX_1 \dots X_n) Y_{11} \dots Y_{1m_1}) Y_{21} \dots Y_{2m_2}) \dots Y_{km_k}) \equiv \boldsymbol{C}^{m_k}(\boldsymbol{B}Y_k(\boldsymbol{C}^{m_{k-1}}(\boldsymbol{B}Y_{k-1} \dots (\boldsymbol{C}^{m_1}(\boldsymbol{B}Y_1(\boldsymbol{C}^n(\boldsymbol{S}UV)X_1 \dots X_n)) Y_{11} \dots Y_{1m_1}) \dots) Y_{k1} \dots Y_{km_k}$$

in Step 2 and something similar but simpler in Step 1.

This is useful if the speed of abstraction rather than the simplicity of the abstract is important.

§4. Broda-Damas and standard combinators. As all standard combinators can be defined in terms of **K** and **S**, it is clear that all (standard or **K-S**) combinators can be defined in terms of **K** and $\Phi_{(b.b)}$. This leads to interrelations between the B-D combinators such as

$$\Phi_{(b.b)}\boldsymbol{KK} = \Phi_{\emptyset}$$
$$\text{and } \Phi_{b.b}\Phi_{\emptyset} = \Phi_b.$$

These are η-equalities - all we really have is

$$\Phi_{(b.b)}\boldsymbol{KK}x =_w \Phi_{\emptyset}x$$
$$\text{and } \Phi_{b.b}\Phi_{\emptyset}x =_w \Phi_b x.$$

Other bases of combinators are often considered (see [2]), here we will just look at **BCIW** (or **BCIS**) - the combinatory counterpart to the Church lambda calculus. This in fact corresponds exactly to the B-D combinators without **K**. We can define

$$\boldsymbol{B} = \Phi_{b.b}$$
$$\boldsymbol{C} = \Phi_{c.b}\Phi_{c.b}\Phi_{c.b}$$
$$\boldsymbol{I} = \Phi_{\emptyset}$$
$$\boldsymbol{W} = \Phi_{(b.\emptyset)}$$
$$\text{and } \boldsymbol{S} = \Phi_{(b.b)}.$$

The **BCIW**-definable λ-terms are exactly terms in which every subterm $\lambda x.Y$ has x appear at least once in Y.

It is clear from the EBDA1 or 2 that all **BCIW**-definable λ-terms can be abstracted without the use of **K**.

REFERENCES

[1] S. Broda and L. Damas, *Compact bracket abstraction in combinatory logic*, this Journal, vol. 62 (1997), pp. 729–740.

[2] M. W. Bunder, *Lambda terms definable as combinators*, ***Theoretical Computer Science***, vol. 169 (1996), pp. 3–21.

[3] Haskell B. Curry and Robert Feys, ***Combinatory logic. Vol. I***, North-Holland Publishing Co., Amsterdam, 1958.

[4] M. Schönfinkel, *über die bausteine der mathematische logik*, ***Mathematische Annalen***, (1924), pp. 305–316.

SCHOOL OF MATHEMATICS AND APPLIED STATISTICS
UNIVERSITY OF WOLLONGONG
WOLLONGONG, NSW 2522, AUSTRALIA
E-mail: martin_bunder@uow.edu.au