BRICS RS-06-12 K. Støvring.: Higher-Order Beta Matching with Solutions in Long Beta-Eta Normal Form

BRICS

Basic Research in Computer Science

Higher-Order Beta Matching with Solutions in Long Beta-Eta Normal Form

Kristian Støvring.

BRICS Report Series **RS-06-12**

ISSN 0909-0878 **June 2006**

See back inner page for a list of recent BRICS Report Series publications.
Copies may be obtained by contacting:

BRICS
Department of Computer Science
University of Aarhus
IT-parken, Aabogade 34
DK–8200 Aarhus N
Denmark
Telephone: +45 8942 9300
Telefax: +45 8942 5601
Internet: BRICS@brics.dk

BRICS publications are in general accessible through the World Wide Web and anonymous FTP through these URLs:

`http://www.brics.dk`
`ftp://ftp.brics.dk`
This document in subdirectory `RS/06/12/`

Higher-Order Beta Matching with Solutions in Long Beta-Eta Normal Form

Kristian Støvring

BRICS [*]
Department of Computer Science
University of Aarhus [†]

May 28, 2006

Abstract

Higher-order matching is a special case of unification of simply-typed lambda-terms: in a matching equation, one of the two sides contains no unification variables. Loader has recently shown that higher-order matching up to beta equivalence is undecidable, but decidability of higher-order matching up to beta-eta equivalence is a long-standing open problem.

We show that higher-order matching up to beta-eta equivalence is decidable if and only if a restricted form of higher-order matching up to beta equivalence is decidable: the restriction is that solutions must be in long beta-eta normal form.

[*]Basic Research in Computer Science (www.brics.dk), funded by the Danish National Research Foundation.

[†]IT-parken, Aabogade 34, DK-8200 Aarhus N, Denmark
E-mail: kss@brics.dk

1 Introduction and related work

The *higher-order matching problem* [7, 12, 20] is the following decision problem:

> Given closed, simply-typed lambda-terms $M : A \to B$ and $N : B$, is there a closed term $X : A$ such that $MX =_{\beta\eta} N$?

Algorithms performing higher-order matching, as well as more general forms of *higher-order unification*, are core parts of systems for automated deduction and higher-order logic programming [8].

Much effort has gone into determining whether the higher-order matching problem is decidable, a question first asked by Huet in 1976 [7, page 2-40]. The problem is decidable in the case where the order of the type A is 4 or less [2, 4, 9, 14, 22, 24], as well as in other special cases [3, 13, 16, 17, 18, 19]. A terminating algorithm has been proposed [25], but it is not known to be complete. In the words of Huet [8], "This vexing but important problem is thus still open after 30 years of intense investigation."

In contrast, the variant of the higher-order matching problem where one considers β-equivalence instead of $\beta\eta$-equivalence has recently been shown to be undecidable [11, 12]. Therefore it seems natural to investigate relations between β-equivalence and $\beta\eta$-equivalence in order to try to shed some light on the complications of higher-order matching. In this article we show that the higher-order matching problem is decidable if and only if the following "hybrid" problem is decidable:

> Given closed, simply-typed lambda-terms $M : A \to B$ and $N : B$, is there a closed term $X : A$ *in long $\beta\eta$-normal form* such that $MX =_{\beta} N$?

For convenience we call this problem the *long β-matching problem*. Here the equivalence relation on terms is β-equivalence, but solutions must be in long $\beta\eta$-normal form. Lifting the latter restriction gives the (general) *β-matching problem* which, as mentioned above, is undecidable.

The most interesting direction of the proof, a reduction from the long β-matching problem to the higher-order matching problem, proceeds by a simply-typed encoding of a "thunk translation" [5], which intuitively has the effect of blocking η-redexes.

What are the consequences of the equivalence between higher-order matching and long β-matching? Assume for the moment that the higher-order matching problem is in fact decidable, as conjectured by Huet. Then the long β-matching problem is also decidable, unlike the general β-matching problem. So if higher-order matching is decidable, the equivalence shown in this article informally indicates that the "computational power" of general β-matching lies in the unbounded search for terms in β-normal—but not necessarily long $\beta\eta$-normal—form.

Here is an example of the difference between the general β-matching problem and the long β-matching problem: let $M = \lambda f^{(b\to b)\to(b\to b)}.\, f$ and $N = \lambda g^{b\to b}.\, g$. Then $M\,N =_\beta N$, but there is no X in long $\beta\eta$-normal form such that $M\,X =_\beta N$. Indeed, let X be in long $\beta\eta$-normal form; then the β-normal form of $M\,X$ is X. The term N is already in β-normal form, but not in long $\beta\eta$-normal form. Therefore X and N are different terms, which means that $M\,X$ and N have different β-normal forms.

2 Background and notation

The reader is assumed to be familiar with basic properties of the simply-typed lambda calculus, as presented for example in the first three sections of Barendregt's handbook chapter [1].

We consider the simply-typed lambda calculus with a countable set of ground types. Let b range over ground types; the set of types is then given by the following grammar:

$$A \quad ::= \quad b \mid A \to A$$

Variables are explicitly typed in the style of Church [1, p. 159], but with a minor variation: first, fix an infinite set of objects called *pre-variables*. A *variable* is then a pair consisting of a pre-variable x and a simple type A; such a pair is written x^A. In particular, if A_1 and A_2 are different types, then x^{A_1} and x^{A_2} are different variables.

Lambda-terms are constructed from variables in the standard way:

$$M \quad ::= \quad x^A \mid \lambda x^A.\, M \mid M\; M$$

Terms are identified up to α-equivalence; note that with the present definition of variables, the standard definitions of substitution and α-equivalence (as well as η-equivalence) naturally accommodate that a pre-variable may occur in different positions with different associated types. The constructions in this article do not depend on such occurrences of the same pre-variable with different types in a single term. Rather, allowing a pre-variable to be associated with different types in different terms is technically convenient in the definition of the thunk translation.

We often omit writing the type of a variable if the type is clear from the context or unimportant.

The two usual equivalences between lambda-terms are used, *β-equivalence* (written $=_\beta$) and *$\beta\eta$-equivalence* (written $=_{\beta\eta}$). One-step β-reduction is written $\longrightarrow_\beta$ and its reflexive-transitive closure is written $\longrightarrow_\beta^*$; similarly for $\beta\eta$-reduction. Every simply-typed term is $\beta\eta$-equivalent to a unique term in *long*

$\beta\eta$-normal form [7]: a term of the form $\lambda x_1 \ldots \lambda x_k \,.\, z^A\ M_1 \ldots M_n$ where A has the form $A_1 \to A_2 \to \ldots \to A_n \to b$ and where each of $M_1, \ldots, M_n$ is in long $\beta\eta$-normal form. In this article, it is convenient to use a slightly more detailed definition of long $\beta\eta$-normal forms:

Definition 1. The set of simply-typed terms in *long $\beta\eta$-normal form* and the set of simply-typed terms in *atomic form* are defined inductively as follows:

(a) Every variable x^A is in atomic form.

(b) If $M : A \to B$ is in atomic form and $N : A$ is in long $\beta\eta$-normal form, then $M\,N$ is in atomic form.

(c) If M is in atomic form and M is of ground type, then M is in long $\beta\eta$-normal form.

(d) If M is in long $\beta\eta$-normal form, then $\lambda x^A.\, M$ is in long $\beta\eta$-normal form.

There is a simple algorithm for computing the long $\beta\eta$-normal form of a given term in β-normal form [7, p. 4-3]. Therefore, by strong normalization of β-reduction, the function mapping each simply-typed term to its long $\beta\eta$-normal form is computable.

3 The thunk translation

We first define a *thunk translation* [5] from lambda-terms to lambda-terms. (The notation used for the translation is different from the original notation [5].) For our purposes, the key property of the translation is that it "blocks" η-redexes, as formalized in Proposition 4 below.

Definition 2 (Thunk translation)**.**

1. Let 0 be a distinguished ground type. For each simple type A, a type $\widetilde{A}$ is defined as follows:

$$\begin{array}{rcl} \widetilde{b} & = & b \\ \widetilde{A \to B} & = & (0 \to \widetilde{A}) \to \widetilde{B} \end{array}$$

2. Let $\bullet^0$ and d^0 be two different, distinguished variables. For each simply-typed term M of type A, a simply-typed term $\widetilde{M}$ of type $\widetilde{A}$ is defined as follows:

$$\begin{array}{rcl} \widetilde{x^A} & = & x^{0 \to \widetilde{A}}\, \bullet^0 \\ \widetilde{M_1 M_2} & = & \widetilde{M_1}\, (\lambda d^0.\, \widetilde{M_2}) \\ \widetilde{\lambda x^A.\, M} & = & \lambda x^{0 \to \widetilde{A}}.\, \widetilde{M} \end{array}$$

If $x_1^{A_1}, \ldots, x_n^{A_n}$ are the free variables of M, then $\bullet^0, x_1^{0 \to \widetilde{A_1}}, \ldots, x_n^{0 \to \widetilde{A_n}}$ are the free variables of $\widetilde{M}$. Therefore, in every subterm $\lambda d^0. M$ of a translated term, d^0 is not free in M: by assumption, d^0 is different from $\bullet^0$, and moreover, d^0 is different from each $x_i^{0 \to \widetilde{A_i}}$ since the types are different.

The analogy with programming-language implementation is that in a translated term, every argument to a function is a "thunk" $\lambda d^0. M$ which must be "forced", by applying it to $\bullet^0$, before use.

The translation gives rise to an equational correspondence. In particular, the translation is sound and complete with respect to β-equivalence:

Proposition 3. $M =_\beta N \quad \Longleftrightarrow \quad \widetilde{M} =_\beta \widetilde{N}$.

Proof. See the original report on the thunk translation [6, Theorem 7.3]. Adapting the proof to a simply-typed calculus presents no problems. Completeness, i.e., the implication from right to left, is the difficult part: the idea is to consider a partial left inverse to the thunk translation, show that it preserves (many-step) β-reduction, and conclude by the Church-Rosser Theorem. □

Now, the key property: η-reduction never applies to a translated term, or to any of its β-reducts:

Proposition 4. $\widetilde{M} \longrightarrow^*_{\beta\eta} N \quad \Rightarrow \quad \widetilde{M} \longrightarrow^*_\beta N$.

Proof. Consider the set of terms $\mathcal{M}_\mathrm{t}$ defined by the following grammar:

$$M_\mathrm{t} \ ::= \ x^{0 \to \widetilde{A}} \, \bullet^0 \ \mid \ M_\mathrm{t} \, (\lambda d^0. M_\mathrm{t}) \ \mid \ \lambda x^{0 \to \widetilde{A}}. M_\mathrm{t} \ \mid \ (\lambda d^0. M_\mathrm{t}) \, \bullet^0$$

As observed by Hatcliff and Danvy [5, 6], every thunk-translated term $\widetilde{M}$ belongs to $\mathcal{M}_\mathrm{t}$, and furthermore $\mathcal{M}_\mathrm{t}$ is closed under β-reduction. Now observe that no term in $\mathcal{M}_\mathrm{t}$ contains an η-redex. Therefore: if $\widetilde{M} \longrightarrow^*_{\beta\eta} N$ then $\widetilde{M} \longrightarrow^*_\beta N$, since no η-redex occurs during the reduction. □

Proposition 5. $\widetilde{M} =_{\beta\eta} \widetilde{N} \quad \Longleftrightarrow \quad \widetilde{M} =_\beta \widetilde{N}$.

Proof. The implication from right to left is trivial. For the other direction, let $\widetilde{M} =_{\beta\eta} \widetilde{N}$. By the Church-Rosser Theorem for $\beta\eta$-reduction on simply typed terms [21] there is a V such that $\widetilde{M} \longrightarrow^*_{\beta\eta} V$ and $\widetilde{N} \longrightarrow^*_{\beta\eta} V$. By the previous proposition $\widetilde{M} \longrightarrow^*_\beta V$ and $\widetilde{N} \longrightarrow^*_\beta V$, hence $\widetilde{M} =_\beta \widetilde{N}$. □

Corollary 6. $M =_\beta N \quad \Longleftrightarrow \quad \widetilde{M} =_{\beta\eta} \widetilde{N}$.

Proof. By Propositions 3 and 5. □

4 Thunk translation using simply typed terms

Even though the thunk translation is sound and complete, in the sense of Corollary 6, it is not "fully complete": there is a type A and a term N of type $\widetilde{A}$ such that N is not $\beta\eta$-equivalent to any translated term. Intuitively, this means that a search for a term $X : A$ can not immediately be reduced to a search for a term $\widetilde{X} : \widetilde{A}$. To see that there are such A and N, choose for example $A = (b \to b) \to (b \to b)$ and $N = \lambda f^{0\to((0\to b)\to b)}.\, \lambda x^{0\to b}.\, f\ \bullet^0\ (\lambda d^0.\, f\, d\, x)$. The only properties of N which are needed here are that N is normal with respect to $\beta\eta$-reduction, and that N is not generated from the grammar used in the proof of Proposition 4. Assume now that $N =_{\beta\eta} \widetilde{M}$. Then by the Church-Rosser Theorem, $\widetilde{M} \longrightarrow^*_{\beta\eta} N$. By Proposition 4, $\widetilde{M} \longrightarrow^*_{\beta} N$, and therefore N can be generated from the grammar used in proof of Proposition 4. But N can in fact not be generated from that grammar, hence N is not $\beta\eta$-equivalent to $\widetilde{M}$ after all. One could modify the thunk translation by requiring that the types of the terms to be translated do not contain the special ground type 0; the above choice of A and N works under that restriction as well.

Since the thunk translation is not fully complete in this sense, it does not immediately appear useful for reducing the *general* β-matching problem to the higher-order matching problem: given closed terms M and N, one can construct a term M' such that $M\, X =_\beta N$ if and only if $M'\, \widetilde{X} =_{\beta\eta} \widetilde{N}$; but even if $M'\, Y =_{\beta\eta} \widetilde{N}$ for some Y, there is no guarantee that this Y is $\beta\eta$-equivalent to a translated term $\widetilde{X}$.

It is, however, possible to define simply-typed terms $T_A : A \to \widetilde{A}$ which "implement" the thunk translation on source terms in long $\beta\eta$-normal form in the following sense: if X is a closed term in long $\beta\eta$-normal form, then $T_A X =_{\beta\eta} \widetilde{X}$. Therefore, with the notation used above, $M\, X =_\beta N$ if and only if $M'\, (T_A\, X) =_{\beta\eta} \widetilde{N}$, provided that X is in long $\beta\eta$-normal form. This is the idea behind our reduction from the long β-matching problem to the higher-order matching problem.

We now turn to the definition of the terms T_A. These terms are strictly speaking not combinators; they contain the free variable $\bullet^0$.

Definition 7. By induction on the type A, define two families of terms $T_A : A \to \widetilde{A}$ and $U_A : \widetilde{A} \to A$:

$$\begin{array}{rcl} T_b & = & \lambda x^b.\, x \\ U_b & = & \lambda x^b.\, x \\ \\ T_{A\to B} & = & \lambda f^{A\to B}.\, \lambda x^{0\to\widetilde{A}}.\, T_B\, (f\, (U_A\, (x\, \bullet))) \\ U_{A\to B} & = & \lambda f^{\widetilde{A\to B}}.\, \lambda x^{A}.\, U_B\, (f\, (\lambda d^0.\, T_A\, x)) \end{array}$$

Such mutually inductive definitions of type-indexed families of lambda-terms are well-known [12, 20].

Proposition 8. *Let M be a term of type A and let the free variables of M be included in the set $\{x_1^{A_1}, \dots, x_n^{A_n}\}$.*

1. *If M is in long $\beta\eta$-normal form, then*

$$T_A\,(M\ [(U_{A_i}\,(x_i^{0\to\widetilde{A_i}}\,\bullet^0))/x_i^{A_i}]_{i=1}^n) =_\beta \widetilde{M}.$$

2. *If M is in atomic form, then*

$$M\ [(U_{A_i}\,(x_i^{0\to\widetilde{A_i}}\,\bullet^0))/x_i^{A_i}]_{i=1}^n =_\beta U_A\,(\widetilde{M}).$$

Here $M\ [N_i/x_i^{A_i}]_{i=1}^n$ denotes the substitution $M\ [N_1/x_1^{A_1} \dots N_n/x_n^{A_n}]$.

Proof. By induction on the definition of long $\beta\eta$-normal forms and atomic forms, using straightforward calculations. First, introduce the abbreviation

$$N^* = N\ [(U_{A_i}\,(x_i\,\bullet))/x_i]_{i=1}^n.$$

Using this abbreviation, we have to show: (1) If M is in long $\beta\eta$-normal form, then $T_A\,(M^*) =_\beta \widetilde{M}$, and (2) if M is in atomic form, then $M^* =_\beta U_A\,(\widetilde{M})$.

1. Let M be in long $\beta\eta$-normal form. There are two cases:
   (a) M is in atomic form and M has some ground type b. Then $T_b\,(M^*) = (\lambda x^b.\,x)\,M^* =_\beta M^*$, and by part (2) of the induction hypothesis, $M^* =_\beta U_b\,(\widetilde{M}) = (\lambda x^b.\,x)\,\widetilde{M} =_\beta \widetilde{M}$.
   (b) $M = \lambda x^{B_1}.\,N$ and $A = B_1 \to B_2$ where $N : B_2$ is in long $\beta\eta$-normal form and x^{B_1} is chosen distinct from the $x_i^{A_i}$. Then, using part (1) of the induction hypothesis:

$$\begin{array}{rcl} T_{B_1\to B_2}\,((\lambda x^{B_1}.\,N)^*) & = & T_{B_1\to B_2}\,(\lambda x^{B_1}.\,N^*) \\ & =_\beta & \lambda x^{0\to\widetilde{B_1}}.\,T_{B_2}\,(N^*[(U_{B_1}\,(x\,\bullet))/x]) \\ & =_\beta & \lambda x^{0\to\widetilde{B_1}}.\,\widetilde{N} \\ & =_\beta & \widetilde{M}. \end{array}$$

2. Let M be in atomic form. There are two cases:
   (c) $M = x_j{}^{A_j}$ and $A = A_j$. Then $M^* = x_j\ [(U_{A_i}\,(x_i\,\bullet))/x_i]_{i=1}^n = U_{A_j}\,(x_j\,\bullet) = U_A\,(\widetilde{M})$.
   (d) $M = N_1\,N_2$ where $N_1 : B \to A$ is in atomic form and $N_2 : B$ is in long $\beta\eta$-normal form. Then, using both parts of the induction hypothesis:

$$\begin{array}{rcl} (N_1\,N_2)^* = N_1^*\,N_2^* & =_\beta & (U_{B\to A}\,\widetilde{N_1})\,N_2^* \\ & =_\beta & U_A\,(\widetilde{N_1}\,(\lambda d^0.\,T_B\,N_2^*)) \\ & =_\beta & U_A\,(\widetilde{N_1}\,(\lambda d^0.\,\widetilde{N_2})) \\ & = & U_A\,(\widetilde{M}). \qquad \square \end{array}$$

□

Corollary 9. *For every closed term M of type A in long $\beta\eta$-normal form,*

$$T_A\, M =_\beta \widetilde{M}.$$

Remark. The condition in Corollary 9 that M is in long $\beta\eta$-normal form is necessary. In fact, one can show that if M is an arbitrary closed term of type A, then $T_A\, M =_\beta \widetilde{N}$ where N is the long $\beta\eta$-normal form of M.

5 Long β-matching reduces to higher-order matching

We now use the results of the two previous sections to show that the long β-matching problem reduces to the higher-order matching problem.

Proposition 10. *Let $M : A \to B$ and $N : B$ be closed terms. Define*

$$\begin{aligned} P &= (\lambda x^A.\, \lambda \bullet^0.\, \widetilde{M}\, (\lambda d^0.\, T_A\, x)) \quad \textit{and} \\ Q &= \lambda \bullet^0.\, \widetilde{N} \end{aligned}$$

Then for every closed term X of type A in long $\beta\eta$-normal form, $M\,X =_\beta N$ if and only if $P\,X =_{\beta\eta} Q$.

Proof. By Corollary 6, $M\,X =_\beta N$ if and only if $\widetilde{M\,X} =_{\beta\eta} \widetilde{N}$. By Corollary 9 (and the definition of the thunk translation), $\widetilde{M\,X} = \widetilde{M}\,(\lambda d.\, \widetilde{X}) =_{\beta\eta} \widetilde{M}\,(\lambda d.\, T_A X)$. Therefore:

$$\begin{aligned} M\,X =_\beta N \quad &\iff \quad \widetilde{M\,X} =_{\beta\eta} \widetilde{N} \\ &\iff \quad \widetilde{M}\,(\lambda d.\, \widetilde{X}) =_{\beta\eta} \widetilde{N} \\ &\iff \quad \widetilde{M}\,(\lambda d.\, T_A X) =_{\beta\eta} \widetilde{N} \\ &\iff \quad \lambda \bullet.\, \widetilde{M}\,(\lambda d.\, T_A X) =_{\beta\eta} \lambda \bullet.\, \widetilde{N} \\ &\iff \quad P\,X =_{\beta\eta} Q. \qquad \square \end{aligned}$$

□

Theorem 11. *If the higher-order matching problem is decidable, then the long β-matching problem is decidable.*

Proof. By reducing the long β-matching problem to the higher-order matching problem. Let $M : A \to B$ and $N : B$ be closed terms, and let the closed terms P and Q be defined as in Proposition 10. Notice that P and Q are indeed closed: they contain no free occurrences of the variable $\bullet^0$.

Assume that a closed term X in long $\beta\eta$-normal term satisfies that $M\,X =_\beta N$. Then by Proposition 10, $P\,X =_{\beta\eta} Q$. Conversely, assume that a closed term Y satisfies that $P\,Y =_{\beta\eta} Q$. Let X be the long $\beta\eta$-normal form of Y; then $P\,X =_{\beta\eta} Q$ and $M\,X =_\beta N$ by Proposition 10. Finally, observe that the terms P and Q are computable from M and N. □

6 Higher-order matching reduces to long β-matching

Consider now the other direction of the equivalence: in this section we show that the higher-order matching problem reduces to the long β-matching problem. The reduction proceeds in the same way as a reduction outlined by Joly [11, p. 137] from the higher-order matching problem to the general β-matching problem.

The reduction is based on the following property: if $M : A \to B$, $X : A$, and $N : B$ are closed, simply-typed lambda-terms in long $\beta\eta$-normal form, then $M\,X =_{\beta\eta} N$ if and only if $M\,X =_\beta N$. This follows from the fact that the set of terms in normal form with respect to *restricted η-expansion* [10] is closed under substitution and β-reduction [15, p. 108]. We instead outline a slightly simpler proof, based on similar closure properties observed by Huet [7, p. 4-4 to 4-5].

Definition 12. The *expanded terms* are the simply-typed terms defined inductively as follows:

1. If $A = A_1 \to \cdots \to A_n \to b$ and if $M_1, \ldots, M_n$ are expanded terms of appropriate types, then $x^A\,M_1\,\ldots\,M_n$ is expanded.
2. If $M : A \to B$ and $N : A$ are expanded, then $M\,N$ is expanded.
3. If M is expanded, then $\lambda x^A.\,M$ is expanded.

Intuitively, a term is expanded if every occurrence of a variable in the term is "fully applied".

Proposition 13.

1. *If M is expanded and $M \longrightarrow_\beta N$, then N is expanded.*
2. *A term is in long $\beta\eta$-normal form if and only if it is in β-normal form and expanded.*

Proof. For part (1), one first shows that the set of expanded terms is closed under substitution, using induction on the definition of expanded terms. For part (2), the "if" direction follows by induction on the definition of expanded terms, while the "only if" direction follows by induction on the definition of long $\beta\eta$-normal forms. □

Proposition 14. *If $M : A \to B$ and $X : A$ are closed terms in long $\beta\eta$-normal form, then the β-normal form of $M\,X$ is in long $\beta\eta$-normal form.*

Proof. Use the previous proposition: by part (2), M and X are expanded, therefore, by the definition of expanded terms, $M\,X$ is expanded. By part (1), the β-normal form of $M\,X$ is expanded, hence by part (2) it is also in long $\beta\eta$-normal form. □

Theorem 15. *If the long β-matching problem is decidable, then the higher-order matching problem is decidable.*

Proof. By reducing the higher-order matching problem to the long β-matching problem. Let $M : A \to B$ and $N : B$ be closed terms; we are interested in whether there exists a closed term X such that $M\,X =_{\beta\eta} N$. Since every closed, simply-typed term P has a long $\beta\eta$-normal form (which is computable from P), we can assume without loss of generality that M and N are in long $\beta\eta$-normal form, and that we only ask for solutions X in long $\beta\eta$-normal form. It is then enough to show the following property: for every X in long $\beta\eta$-normal form, $M\,X =_{\beta\eta} N$ if and only $M\,X =_{\beta} N$. So assume that $M\,X =_{\beta\eta} N$. Then by the previous proposition, the β-normal form of $M\,X$ is actually the long $\beta\eta$-normal form of $M\,X$, which is N (by uniqueness of long $\beta\eta$-normal forms). □

7 Conclusion

Decidability of higher-order matching is a long-standing open problem motivated by algorithms used in systems for automated deduction and higher-order logic programming. We have shown that higher-order matching is decidable if and only if a restricted variant of β-matching is decidable. General β-matching is known to be undecidable.

The proof relies on a translation from lambda-terms to lambda-terms. A natural direction for future work would be to investigate other translations in connection with other variants of higher-order matching. Another idea is to look for negative results on the existence of fully complete translations, thereby determining inherent limitations of the translation approach.

Acknowledgements. Thanks are due to Małgorzata Biernacka, Olivier Danvy, Andrzej Filinski, and the anonymous referees for comments on this work.

Added in final version. Stirling has recently proposed a game-theoretic argument that higher-order matching is decidable [23].

References

[1] Henk Barendregt. Lambda calculi with types. In Samson Abramsky, Dov M. Gabbay, and Thomas S. E. Maibaum, editors, *Handbook of Logic in Computer Science, Vol. 2*, chapter 2, pages 118–309. Oxford University Press, Oxford, 1992.

[2] Hubert Comon and Yan Jurski. Higher-order matching and tree automata. In Mogens Nielsen and Wolfgang Thomas, editors, *Computer Science Logic, 11th International Workshop, CSL '97*, volume 1414 of *Lecture Notes in Computer Science*, pages 157–176. Springer-Verlag, 1998.

[3] Dan Dougherty and Tomasz Wierzbicki. A decidable variant of higher order matching. In Sophie Tison, editor, *Rewriting Techniques and Applications, 13th International Conference, RTA 2002*, volume 2378 of *Lecture Notes in Computer Science*, pages 340–351. Springer-Verlag, 2002.

[4] Gilles Dowek. Third order matching is decidable. *Annals of Pure and Applied Logic*, 69(2–3):135–155, 1994.

[5] John Hatcliff and Olivier Danvy. Thunks and the λ-calculus. *Journal of Functional Programming*, 7(3):303–319, 1997.

[6] John Hatcliff and Olivier Danvy. Thunks and the λ-calculus (extended version). Technical Report BRICS RS-97-7, DAIMI, Department of Computer Science, University of Aarhus, March 1997.

[7] Gérard Huet. *Résolution d'équations dans les langages d'ordre 1, 2, ..., ω*. Thèse d'état, Université de Paris VII, Paris, France, 1976.

[8] Gérard Huet. Higher order unification 30 years later. In Victor Carreño, César Muñoz, and Sofiène Tashar, editors, *Theorem Proving in Higher Order Logics*, volume 2410 of *Lecture Notes in Computer Science*, pages 3–12. Springer-Verlag, 2002.

[9] Gérard Huet and Bernard Lang. Proving and applying program transformations expressed with second-order patterns. *Acta Informatica*, 11:31–55, 1978.

[10] C. Barry Jay and Neil Ghani. The virtues of eta-expansion. *Journal of Functional Programming*, 5(2):135–154, 1995.

[11] Thierry Joly. On lambda-definability I: the fixed model problem and generalizations of the matching problem. *Fundamenta Informaticae*, 65(1-2):135–151, 2005.

[12] Ralph Loader. Higher order beta matching is undecidable. *Logic Journal of the IGPL*, 11(1):51–68, 2003.

[13] Vincent Padovani. Decidability of all minimal models. In Stefano Berardi and Mario Coppo, editors, *Types for Proofs and Programs, International Workshop TYPES '95*, volume 1158 of *Lecture Notes in Computer Science*, pages 201–215. Springer-Verlag, 1996.

[14] Vincent Padovani. Decidability of fourth-order matching. *Mathematical Structures in Computer Science*, 10(3):361–372, 2000.

[15] Femke van Raamsdonk. *Confluence and Normalization for Higher-Order Rewriting*. PhD thesis, Vrije Universiteit, Amsterdam, 1996.

[16] Sylvain Salvati and Philippe de Groote. On the complexity of higher-order matching in the linear lambda-calculus. In Robert Nieuwenhuis, editor, *Rewriting Techniques and Applications, 14th International Conference, RTA 2003*, volume 2706 of *Lecture Notes in Computer Science*, pages 234–245. Springer-Verlag, 2003.

[17] Manfred Schmidt-Schauß. Decidability of arity-bounded higher-order matching. In Franz Baader, editor, *Automated Deduction – CADE-19: 19th International Conference on Automated Deduction*, volume 2741 of *Lecture Notes in Computer Science*, pages 488–502. Springer-Verlag, 2003.

[18] Aleksy Schubert. Linear interpolation for the higher-order matching problem. In Michel Bidoit and Max Dauchet, editors, *TAPSOFT '97: Theory and Practice of Software Development*, volume 1214 of *Lecture Notes in Computer Science*, pages 441–452. Springer-Verlag, 1997.

[19] Aleksy Schubert. A self-dependency constraint in the simply typed lambda calculus. In Maciej Liskiewicz and Rüdiger Reischuk, editors, *Fundamentals of Computation Theory, 15th International Symposium, FCT 2005*, volume 3623 of *Lecture Notes in Computer Science*, pages 352–364. Springer-Verlag, 2005.

[20] Richard Statman. Completeness, invariance and lambda-definability. *Journal of Symbolic Logic*, 47(1):17–26, 1982.

[21] Richard Statman. Logical relations and the typed λ-calculus. *Information and Control*, 65:85–97, 1985.

[22] Colin Stirling. Higher-order matching and games. In Luke Ong, editor, *Computer Science Logic, 19th International Workshop, CSL 2005*, volume 3634 of *Lecture Notes in Computer Science*, pages 119–134. Springer-Verlag, 2005.

[23] Colin Stirling. A game-theoretic approach to deciding higher-order matching. To be presented at the 33rd International Colloquium on Automata, Languages and Programming (ICALP 2006), July 2006.

[24] Tomasz Wierzbicki. Complexity of the higher order matching. In Harald Ganzinger, editor, *Automated Deduction — CADE-16: 16th International Conference on Automated Deduction*, volume 1632 of *Lecture Notes in Computer Science*, pages 82–96. Springer-Verlag, 1999.

[25] David A. Wolfram. *The Clausal Theory of Types*, volume 21 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1993.

Recent BRICS Report Series Publications

RS-06-12 Kristian Støvring. *Higher-Order Beta Matching with Solutions in Long Beta-Eta Normal Form*. June 2006. 13 pp. To appear in *Nordic Journal of Computing*, 2006.

RS-06-11 Kim G. Larsen, Ulrik Nyman, and Andrzej Wasowski. *An Interface Theory for Input/Output Automata*. June 2006.

RS-06-10 Christian Kirkegaard and Anders Møller. *Static Analysis for Java Servlets and JSP*. June 2006. 23 pp.

RS-06-9 Claus Brabrand, Giegerich Robert, and Anders Møller. *Analyzing Ambiguity of Context-Free Grammars*. April 2006.

RS-06-8 Christian Kirkegaard and Anders Møller. *Static Analysis for Java Servlets and JSP*. April 2006. 22 pp.

RS-06-7 Petr Jančar and Jiří Srba. *Undecidability Results for Bisimilarity on Prefix Rewrite Systems*. April 2006. 20 pp. Presented at *FoSSaCS 2006*, LNCS 3921:277–291.

RS-06-6 Luca Aceto, Willem Jan Fokkink, Anna Ingólfsdóttir, and Bas Luttik. *A Finite Equational Base for CCS with Left Merge and Communication Merge*. March 2006. 22 pp.

RS-06-5 Kristian Støvring. *Extending the Extensional Lambda Calculus with Surjective Pairing is Conservative*. March 2006. 18 pp. To appear in *Logical Methods in Computer Science*. Supersedes RS-05-35.

RS-06-4 Olivier Danvy and Kevin Millikin. *A Rational Deconstruction of Landin's J Operator*. February 2006. ii+26 pp. To appear in the post-reviewed proceedings of the 17th International Workshop on the *Implementation and Application of Functional Languages* (IFL'05), Dublin, Ireland, September 2005.

RS-06-3 Małgorzata Biernacka and Olivier Danvy. *A Concrete Framework for Environment Machines*. February 2006. ii+29 pp. To appear in the *ACM Transactions on Computational Logic*. Supersedes BRICS RS-05-15.

RS-06-2 Mikkel Baun Kjærgaard and Jonathan Bunde-Pedersen. *A Formal Model for Context-Awareness*. February 2006. 26 pp.

RS-06-1 Luca Aceto, Taolue Chen, Willem Jan Fokkink, and Anna Ingólfsdóttir. *On the Axiomatizability of Priority*. January 2006. 25 pp.