Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Symbolic Computation 40 (2005) 905–954
Journal of Symbolic Computation
www.elsevier.com/locate/jsc

Decidability of bounded higher-order unification

Manfred Schmidt-Schauß[a,*], Klaus U. Schulz[b,1]

[a] *Institut für Informatik, J.-W.-Goethe-Universität, Postfach 11 19 32, D-60054 Frankfurt, Germany*
[b] *CIS, University of Munich, Oettingenstrasse 67, D-80538 München, Germany*

Received 14 January 2004; accepted 11 January 2005
Available online 26 February 2005

Abstract

It is shown that unifiability of terms in the simply typed lambda calculus with β and η rules becomes decidable if there is a bound on the number of bound variables and lambdas in a unifier in η-expanded β-normal form.

Keywords: Higher-order unification; Decision algorithms; Simply typed lambda calculus; Bounded unification problems; Exponent of periodicity

1. Introduction

First-order unification (Baader and Siekmann, 1994) is a fundamental operation in several areas of computer science, e.g. automated deduction, term rewriting, logic programming and type-checking. The generalization to higher-order unification increases the expressiveness and the applicability and improves the level of abstraction. This explains the interest in higher-order systems such as higher-order logics and higher-order deduction systems (Andrews, 1986; Paulson, 1994; Goubault-Larrecq and Mackie, 1997; Andrews, 2001; Pfenning, 2001), higher-order (functional) programming languages

* Corresponding address: J.W.Goethe-University Frankfurt, Institut fuer Informatik, Fachbereich Biologie und Informatik, Robert Mayer-Strasse 11-15, 60054 Frankfurt, Germany. Tel.: +49 69 798 28597; fax: +49 69 798 28919.

E-mail addresses: schauss@ki.informatik.uni-frankfurt.de (M. Schmidt-Schauß), schulz@cis.uni-muenchen.de (K.U. Schulz).

1 Tel.: +49 89 2180 9700; fax: +49 89 2180 9701.

0747-7171/$ - see front matter
doi:10.1016/j.jsc.2005.01.005

(Burstall et al., 1980; Turner, 1985; Paulson, 1991; Barendregt, 1990; Bird, 1998), higher-order logic programming languages (Miller, 1991; Hanus et al., 1995), higher-order rewriting (Nipkow, 1991; Klop, 1992; Dershowitz and Jouannaud, 1990) and higher-order unification (Huet, 1975; Dowek, 2001).

Higher-order unification procedures were already described in Huet (1975) and Jensen and Pietrzykowski (1976). It is well known that second-order unification — and hence higher-order unification — is undecidable (Goldfarb, 1981; Farmer, 1991; Levy and Veanes, 2000). In order to introduce natural restrictions that lead to decidable unification problems, at least two orthogonal directions can be followed. On the one hand, we may restrict the syntactic form of the input problems. A well-known syntactic restriction that leads to a decidable unification problem is the unification of higher-order patterns (Miller, 1991). Clearly, a brute force syntactic restriction of the input problems is often not possible. On the other hand, we may also impose restrictions on substitutions that may be used to solve unification problems. From an intuitive point of view that means that we are willing to ignore solutions that are "too complex" with respect to a given measure. One variant of this idea is "bounded unification" where for each input unification problem an upper bound on the number of occurrences of lambda bindings in the substitution terms is fixed. The naturalness and attractiveness of bounded versions of higher-order formalisms relies on the fact that any input is possible and the respective first-order fragments are not restricted at all. In particular, the number of occurrences of first-order function symbols is unrestricted.

In Schmidt-Schauß (1999a, 2004) it was shown that second-order unification becomes decidable if an upper bound on the number of occurrences of bound variables in the substitution terms is fixed, which has as a corollary the well-known result that second-order unification with monadic function symbols is decidable (Huet, 1975; Zhezherun, 1979; Farmer, 1988). On the negative side, the monadic restriction fails to yield a decidable unification problem if generalized to all types. Restricting third-order unification to monadic types, i.e. where every function has at most one argument, was shown to be undecidable in Narendran (1990). A further restriction of second-order unification that restricts the number of bound variables in the substitution terms to be one is context unification. The conjecture is that context unification is decidable. There are several results on decidability of fragments of context unification (Comon, 1998; Schmidt-Schauß, 1994, 1999b, 2001, 2002; Schmidt-Schauß and Schulz, 2002b; Levy, 1996) or variants of context unification (Cervesato and Pfenning, 1997).

In this paper we generalize the result on decidability of bounded second-order unification to higher-order unification in the simply typed lambda calculus with β and η rules (Barendregt, 1984; Hindley and Seldin, 1986). We show that solvability of bounded higher-order unification problems (BHOUPs) is decidable. By a BHOUP, we mean a higher-order unification problem where for any variable a bound on the number of lambda-binders and occurrences of bound variables in the image of the variable under a unifier is given. Here each image is assumed to be in η-expanded β-normal form. Note that this notion of boundedness is slightly more restrictive than the straightforward generalization from the second-order case (see Remark 14.2).

A comparable approach is the k-duplicating higher-order unification problem (see Dougherty and Wierzbicki, 2002), which is a generalization of k-duplicating higher-order matching together with a bound on the number of occurrences of function symbols in substitutions. However, bounded higher-order unification is strictly more general, since the number of occurrences of first-order function symbols is unrestricted. Furthermore, k-duplicating higher-order unification is equivalent to first computing a finite set of substitutions, and then checking whether this set contains a unifier.

Our result implies that undecidability proofs for higher-order unification require an unbounded number of lambda-bound variables or lambdas in a unifier in η-expanded normal form. It can be used to define a semi-decision procedure for ordinary higher-order unification where we start with given bounds for the variables in the problem and increase the bounds as long as we have an unsolvable problem.

Omitting many details, the decision procedure can be described in the following way. In an initial step, a given input BHOUP is translated into a finite number of BHOUPs of a special form, called reduced BHOUPs (RBHOUPs). We show that the input BHOUP is solvable iff a successor RBHOUP is solvable. Ignoring types, RBHOUPs are similar to bounded second-order unification problems. Our treatment of RBHOUPs adapts methods from Schmidt-Schauß (1999a, 2004) (see also Remark 6.8). On the basis of some simple syntactic criteria, we distinguish between four kinds of RBHOUPs, called "xy", "amb", "nocycle", and "unique". RBHOUPs of kind "xy" are shown to be solvable. For a given RBHOUP of kind "amb", "nocycle", or "unique" we apply non-deterministic transformation steps that transform a given RBHOUP into a finite number of possible successor RBHOUPs. We prove that each transformation step reduces a fixed well-founded measure. By König's Lemma, iterated transformation of a given input RBHOUP defines a finite search tree. In each branch of the tree, the transformation stops if either a RBHOUP of kind "xy" is found, or we reach a RBHOUP with an empty set of successors. In a sense to be made precise, each transformation rule is sound and complete. In particular, the set of successors of a solvable RBHOUP of type "amb", "nocycle" or "unique" is always non-empty. It follows that a given RBHOUP resulting from the initial translation step is solvable if and only if a RBHOUP of kind "xy" is found in some branch of the search tree. Since the search tree is finite such a successful branch can be effectively found for each solvable input problem.

A needed assumption for the decision algorithm is that each transformation step is finitely branching. In order to achieve this goal, the algorithm does not try to generate a complete set of unifiers for the given input BHOUP. Instead, the unifiers that are taken into account by the transformation rules satisfy two characteristic restrictions:

1. The terms in the codomain of the unifier are built over a finite signature determined by the input problem.
2. The "exponent of periodicity" of a unifier is bound by a constant determined by the input problem.

The exponent of periodicity of a unifier, roughly, is the maximal number of periodical repetitions of a non-empty tree-pattern that occurs in the solution values of the variables. It should be mentioned that the bound for the exponent of periodicity that is used represents a non-elementary recursive function. For transformation of BHOUPs of type "unique", the bound restricts the number of possible successor BHOUPs. For this reason the branching degree of the search tree, as well as the complexity of the decision algorithm, is non-elementary recursive.

The structure of the paper is as follows. After a brief description of the problem context and related work we start with an introduction to the simply typed lambda-calculus with β and η rules in Section 3. We prove that there is a computable non-elementary upper bound for the size of the η-expanded β-normal form of a given term. Since this result provides the basis for restricting the exponent of periodicity of a given input BHOUP it can be considered as one source of the non-elementary complexity of our decision algorithm, the second being the intermediate β-reductions (see Section 13). In Section 4 we formally introduce bounded higher-order unification problems (BHOUPs) and their unifiers. Section 5 defines the exponent of periodicity of a unifier of a BHOUP, describes properties of minimal solutions of BHOUPs and provides a justification for imposing the above two restrictions on unifiers. Section 6 describes the aforementioned initial translation step of BHOUPs into RBHOUPs, which restricts free variables to be first-order or second-order (plus further constraints). In Section 7 the notions of soundness and completeness that are used for proving correctness of the remaining steps of the decision algorithm are made precise. A well-founded measure is defined that yields a termination order for the algorithm. The four kinds of RBHOUPs mentioned above are formally defined. We then give a compact description of the complete algorithm and prove the central result of this paper: unifiability of BHOUPs is decidable. To simplify the orientation, all details of the rather technical transformation of RBHOUPs of type “amb”, “nocycle” and “unique”, as well as the corresponding subparts of the proof, are omitted at this point. The missing transformation rules are described in Sections 8–12, where we also prove that the rules have the needed properties. In Section 13 we show a non-elementary lower bound for the complexity, and in Section 14 we summarize the results obtained.

A preliminary and shortened version of the results in this paper appeared in Schmidt-Schauß and Schulz (2002a).

2. Context and related work

In order to justify the bound for the exponent of periodicity that leads to a finite branching of the search tree, we use a lemma on an upper bound for the exponent of periodicity for a minimal unifier for context unification from Schmidt-Schauß and Schulz (1998). The lemma in Schmidt-Schauß and Schulz (1998) generalizes a proposition that appeared in the decidability proof of word unification by Makanin (1977). An improvement of the latter result was given in Kościelski and Pacholski (1996). This link to word unification (Makanin, 1977; Schulz, 1990, 1993; Gutierrez, 1998; Plandowski, 1999) is not accidental. The relationship between word unification and bounded higher-order unification is indicated in Fig. 1 where we also mention some other problems in order to position the results of this paper.

The problems mentioned in the figure can be divided into two classes. The class of “complete structural alignment problems” comprehends word unification and context unification. The class of “functional equality problems” contains all other problems. The principal difference between the two classes may be exemplified using the equation

$$xa \doteq xb.$$

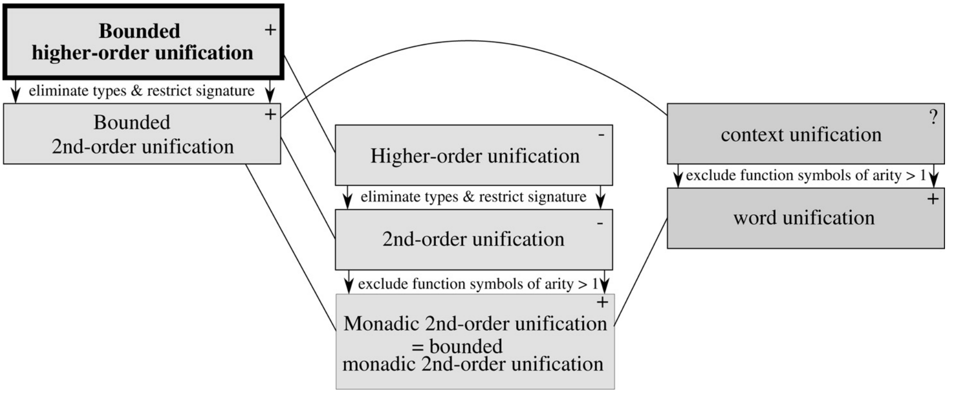

Fig. 1. Decidable (+) and undecidable (−) unification problems and their relationship.

When treating the equation as a complete structural alignment problem, as it is done in word unification, we ask for a word W, representing the solution value for x, such that the complete strings Wa and Wb are identical. Obviously, this is not possible and the equation is unsolvable. From the functional equality perspective, where the equation can be rewritten in the form $x(a) = x(b)$, we ask for a function F such that $F(a) = F(b)$. Assuming that the constant function $\lambda y.a$ represents a possible value for x, the equation has a trivial solution. More generally, in complete structural alignment problems the value of each subterm occurring in an equation influences the final identity of both sides under a solution. In contrast, when solving functional equality problems, constant functions can be used and the values of arguments may become irrelevant.

The possible values for functional variables in second-order monadic unification are unary (resp. constant) term functions of the form $f(\dots g([\cdot])\dots)$ (resp. $f(\dots g(a)\dots)$) composed of unary first-order function symbols and individual constants. From one perspective, all these functions have *one argument* that has *at most one* occurrence/position. Second-order and higher-order unification problems are more general in the sense that the functions assigned to variables may have an arbitrary number of arguments, and each argument may have an arbitrary number of argument positions/occurrences. For the bounded versions, an upper limit for the number of arguments and positions is fixed. Still, since for each argument the number of occurrences/positions may be zero, solution values may be constant functions, which means that equations where both sides start with variables, also called flexible–flexible, are always trivially solvable. This effect is heavily used in the decidability results for bounded second-order/higher-order unification given in Schmidt-Schauß (1999a, 2004) and in the present paper. It shows that the above-mentioned BHOUPs of kind "xy" are always solvable.

In context unification (Niehren et al., 1997; Vorobyov, 1998; Niehren et al., 2000; Schmidt-Schauß, 1999b, 2002; Schmidt-Schauß and Schulz, 2002b; Levy and Villaret, 2000), solution values of context variables are tree functions with one argument that has *exactly one* occurrence/position. Hence solving a context equation leads to a complete structural alignment of both sides as trees.

As we explain in Section 14.3, (bounded) second-order unification can be considered as a special case of (bounded) higher-order unification where characteristic restrictions on signatures and types are imposed. From (bounded) second-order unification we get monadic second-order unification by exclusion of function symbols of arity > 1. In the same way, word unification is obtained from context unification.

Lemmas on the exponent of periodicity, roughly, give a bound on the maximal number of periodical repetitions of words/trees that occur in the solution values of "minimal" solutions. All known decision procedures for word unification are based on periodicity bounds. For the above-mentioned "functional equality problems", periodicity bounds are used for simplifying equations of a particular kind. The idea can be illustrated using a monadic second-order equation $X(s) \doteq f(X(t))$. Obviously, under any solution, X is mapped to a function of the form $f(f(\ldots([\cdot])\ldots))$. If we have an upper bound for the number of periodical repetitions of f's in a minimal solution, it is possible to eliminate X in the equation, using a finite subcase analysis. A more complex, but similar argument is used in the present paper for simplifying bounded higher-order unification problems of kind "unique".

3. Simply typed lambda-calculus

We present the simply typed lambda-calculus (see Barendregt, 1984; Wolfram, 1993; Hindley, 1997).

3.1. Types and terms

Definition 3.1. The language of types is defined according to the grammar

$$T ::= T_0 \mid (T \to T)$$

where $T_0 \neq \emptyset$ is the set of *elementary types*. The symbols α, τ range over types, and ι ranges over elementary types.

A shorter notation for types of the form $\tau = (\alpha_1 \to (\alpha_2 \ldots (\alpha_n \to \iota)\ldots))$ is $(\alpha_1 \to \alpha_2 \to \cdots \to \alpha_n \to \iota)$. The number $n \geq 0$ is called the *arity* of the type τ, denoted $ar(\tau)$, and ι is called the *target type* of τ.

The background signature Σ for building higher-order terms is a set of *function symbols*, where every function symbol f comes with a type $type(f)$. The arity of f is defined as $ar(f) := ar(type(f))$. Function symbols f of elementary type (i.e., $ar(f) = 0$) are called *elementary constant symbols*. We assume that Σ contains for every type τ a countably infinite set of function symbols. For every type τ there is in addition a (countably) infinite set of variables V_τ. The union of all these sets is denoted as $\mathcal{V}$. Variables are denoted as x, y, z, expressions $\overrightarrow{x}$ denote finite sequences of variables. As for function symbols, with $type(x)$ we denote the type of x. The arity $ar(x)$ of x is $ar(x) := ar(type(x))$. If necessary, the type is indicated as a superscript. Since for every type there are infinitely many variables (or function symbols, respectively), we can always use fresh variables (or function symbols, respectively). A variable of elementary type is also called a *first-order variable*.

Definition 3.2. For every type τ we define the set $Term^\tau$ of terms of type τ according to the grammar

$$Term^\tau ::= f^\tau \mid x^\tau \mid (Term^{\tau' \to \tau}\ Term^{\tau'}) \mid \lambda x^{\tau_1}.Term^{\tau_2}$$

where f is a function symbol of type τ, and x is a variable of type τ. The term $\lambda x^{\tau_1}.Term^{\tau_2}$ is only valid if $\tau_1 \to \tau_2 = \tau$.

If $t \in Term^\tau$, we say t *has type* τ and denote this by $type(t) = \tau$. Terms of the form $(t_1\ t_2)$ are called *applications*, and terms of the form $\lambda x.t$ are called *abstractions*. The *target type* of a term t is the target type of $type(t)$.

The notions of *bound* and *free variables* in a term and *open* and *closed* (or *ground*) *terms* are as usual. The set of free variables in a term t is denoted as $FV(t)$. We say that s, t are *α-equal* ($=_\alpha$), if s and t differ only by a sequence of renamings of bound variables of equal types. To avoid too clumsy notation and to avoid distraction from the essential, we assume the *disjoint variable convention*: all bound variables in terms are distinct, and whenever an operation makes it necessary to rename bound variables, this is done.

To avoid excessive bracketing, we write applications in flat form: $(t_1\ t_2 \dots t_n)$ means the term $(\dots((t_1\ t_2)\ t_3)\dots t_n)$. If a term is of the form $(f\ t_1 \dots t_n)$, where $n = ar(f)$, then we may write this term as $f(t_1, \dots, t_n)$. We also write nested lambda-expressions in a shorter form. $\lambda x_1, x_2, \dots, x_n.t$ means $\lambda x_1.\lambda x_2.\dots\lambda x_n.t$. Expressions $\lambda \overrightarrow{x}.t$ are shorthand for $\lambda x_1, x_2, \dots, x_n.t$. We also omit top level brackets and assume that application has a higher priority than λ, which means that the scope of a lambda extends as far to the right as possible. For example, $\lambda y.x\ y\ z$ denotes $\lambda y.((x\ y)\ z)$. We will use positions in terms, which are tree addresses corresponding to occurrences of subterms, where the kind of positions used is dependent on the representation used (non-flattened or flattened).

A *maximal application* in a term is a subterm of the form $(t_1\ t_2 \dots t_n)$ with $n \geq 2$ that is not the left subterm of an application.

The *head* of an application is the subterm that is in the leftmost position in the flat representation. For example, f is the head of $(f\ t_1 \dots t_n)$.

For a set of types T, let $subt(T)$ be the smallest superset of T with the condition: If $\alpha \to \beta \in subt(T)$, then $\alpha, \beta \in subt(T)$. For a term t, let $types(t)$ be the set of all the types of subterms, and let $subt(t)$ be $subt(types(t))$.

3.2. First-order terms and first-order contexts

For each type τ let $[\cdot]^\tau$ denote a new function symbol of type τ that does not belong to Σ. $[\cdot]^\tau$ is called the *hole* of type τ. *Contexts* are terms built over the enlarged signature that have exactly one occurrence of a hole. The expression $C[t]^\tau$ for a context C and a term/context t of type τ denotes the term/context that is constructed from C by replacing the hole $[\cdot]^\tau$ with t. Since descriptions such as $C[t]^\tau$ are used as a kind of meta-syntax free variables of t may be bound in C, i.e., variable capture is permitted for contexts. A context is *trivial* iff it has the form $[\cdot]^\tau$. A context B is a *prefix of a context* C iff there is a context B' such that $C = B[B']$. A context B is a *suffix of a context* C iff there is a context B' such that $C = B'[B]$. A context B is a *subcontext of a term* t iff there is a context B' and a term t' such that $t = B'[B[t']]$. A context B is a *subcontext of a context* C iff it is a subcontext of a subterm of C, or there are contexts B', B'' such that $C = B'[B[B'']]$.

Definition 3.3. A *first-order function symbol* is a function symbol of type $\iota_1 \to \iota_2 \to \cdots \to \iota_m \to \iota$ with $m \geq 0$. A *first-order term* is a term generated by the grammar

$$FOT ::= x^\iota \mid f(FOT_1, \ldots, FOT_n) \mid a$$

where f denotes a first-order function symbol of arity $n \geq 1$, and a is an elementary constant. A *first-order context* is defined using the grammar

$$FOC ::= [\cdot]^\iota \mid f(t_1, \ldots, t_{i-1}, FOC, t_{i+1}, \ldots, t_n)$$

where f is a first-order function symbol of arity $n \geq 1$ and the t_i are first-order terms.

For first-order terms/contexts we will use positions as for conventional terms of first-order logic and call them *first-order positions*. For example, i is the first-order position of t_i in $f(t_1, \ldots, t_n)$. We will use the number 0 to point to the position of the function symbol. The length of the first-order position of the hole of a first-order context C is called *main depth* of C, denoted as $|C|$. The first digit of the position of the hole of a nontrivial first-order context C is denoted as $\mathit{firstdpos}(C)$. If n is a nonnegative integer and C is a first-order context of type ι with hole of type ι, then $C^0 := [\cdot]^\iota$, and $C^{n+1} := C[C^n]$. Note that C^n is of type ι.

3.3. Measures

For the following proofs, several measures are needed.

Definition 3.4.

- The *order* $\mathit{ord}(\tau)$ of a type τ is defined as follows:
  - $\mathit{ord}(\iota) = 1$.
  - If $\tau = \alpha_1 \to \cdots \alpha_n \to \iota$, then $\mathit{ord}(\tau) = 1 + \mathit{max}\{\mathit{ord}(\alpha_1), \ldots, \mathit{ord}(\alpha_n)\}$.

  The *degree* of a term t is $\mathit{deg}(t) := \mathit{max}\{(\mathit{ord}(\tau) - 1) \mid \tau \in \mathit{subt}(t)\}$.[2]
- The *size* of type τ is defined as follows:
  - $\mathit{size}(\iota) = 1$.
  - $\mathit{size}(\tau_1 \to \tau_2) := \mathit{size}(\tau_1) + \mathit{size}(\tau_2) + 1$.
- The *size* of a term t is defined as follows: $\mathit{size}(x) = 1, \mathit{size}(f) = 1, \mathit{size}(\lambda x.t) = \mathit{size}(t) + 2$, and $\mathit{size}(s\ t) = 1 + \mathit{size}(s) + \mathit{size}(t)$.
- The *length* $\mathit{len}(t)$ of a term t is defined as follows (Beckmann, 2001): $\mathit{len}(x) = 1, \mathit{len}(f) = 1, \mathit{len}(\lambda x.t) = \mathit{len}(t) + 1$, and $\mathit{len}(s\ t) = \mathit{len}(s) + \mathit{len}(t)$. Note that $\mathit{len}(t) \leq \mathit{size}(t)$.

Note that $\mathit{size}(\alpha_1 \to \cdots \alpha_n \to \iota) = 1 + n + \sum_{i=1}^{n} \mathit{size}(\alpha_i)$.
The last measure is used to formally define bounded higher-order unification problems — it plays a central role:

Definition 3.5. For a term t, we define $\#\mathit{bvl}(t)$ to be the number of occurrences of bound variables in t plus the number of lambda-binders in t.

For example, $\#\mathit{bvl}(\lambda x. f\ \lambda y.x\ y\ z) = 4$.

[2] In Beckmann (2001), the degree of a term is defined similarly to the order in papers on unification; however, *degree* $=$ *order* $- 1$.

3.4. Instantiation and substitutions

An *instantiation* of a variable x^τ in s by the term t^τ, written $s[t/x]$, replaces free occurrences of the variable x in s by t, where before replacement, the bound variables in s have to be renamed to avoid variable capture. See, e.g., Hindley and Seldin (1986) for a precise definition. After the replacement, it may be necessary to rename bound variables in the different copies of t, since we use the disjoint variable convention. The notation $s[t/x]$ is only used if t and x have the same type.

A *closed substitution* is a mapping from variables to closed terms, represented as $\{x_i \to t_i \mid i = 1, \ldots, n\}$, where t_i for $i = 1, \ldots, n$ is a closed term with $type(x_i) = type(t_i)$ for $i = 1, \ldots, n$. To apply the closed substitution σ to the term s means to simultaneously instantiate each free variable x_i by t_i $(1 \le i \le n)$. We tacitly assume that a closed substitution σ is only applied to terms t such that every free variable in t is replaced; hence we assume that the result of applying a closed substitution to a term results in a closed term. This makes a closed substitution a mapping from terms to closed terms. In the following, by a substitution we always mean a closed substitution. The *domain* of σ, denoted as $dom(\sigma)$, is the set $\{x_i \mid i = 1, \ldots, n\}$, and the *codomain* of σ (denoted as $cod(\sigma)$) is the set $\{t_i \mid i = 1, \ldots, n\}$. If all function symbols occurring as subterms in $cod(\sigma)$ are in the set $\Sigma_0 \subseteq \Sigma$, then σ is called a Σ_0-*substitution*.

3.5. Equality and normal forms

Since we use the $\beta\eta$ rules for the simply typed lambda-calculus, there are the following equations between terms:

$$\begin{array}{llcll}
(\alpha) & \lambda x.t & = & \lambda y.t[y/x] & y \text{ is a fresh variable} \\
(\beta) & ((\lambda x.t)\ s) & = & t[s/x] & \\
(\eta) & t & = & \lambda x^\tau.t\ x & \text{if } type(t) = \tau \to \tau_1 \text{ and } x \notin FV(t).
\end{array}$$

Of course we also assume that the thus defined equality $=_{\beta\eta}$ is an equivalence relation and a congruence, i.e. $s =_{\beta\eta} t \Rightarrow C[s] =_{\beta\eta} C[t]$. Note that $s =_\alpha t \Rightarrow s =_{\beta\eta} t$.

The equations for (β), (η) are applied in a directed way for normalizing terms. We will employ η-expansion, denoted as $\overline{\eta}$.

(β)	$C[(\lambda x.t)\ s]$	$\to$	$C[t[s/x]]$	for all contexts C.
(η)	$C[\lambda y.t\ y]$	$\to$	$C[t]$	If $y \notin FV(t)$. The rule is applicable for all contexts $C[]$.
$(\overline{\eta})$	$C[t]$	$\to$	$C[\lambda y.t\ y]$	if t is not an abstraction, $type(t)$ is not an elementary type, and t in $C[t]$ is a maximal application. The variable y must be a fresh variable of appropriate type. This reduction is valid for all contexts C.

Since there are wrong definitions in the literature, we give some examples to clarify what we mean by $(\overline{\eta})$-reduction.

Example 3.6. The term $x^{\iota\to\iota}$ can be $(\overline{\eta})$-reduced (η-expanded) to $\lambda y^\iota.x^{\iota\to\iota}\ y$.

The term $\lambda x_1^{(\iota\to\iota)\to\iota\to\iota}, x_2^{\iota\to\iota}.x_1\ x_2$ will be $(\overline{\eta})$-reduced in two steps to $\lambda x_1, x_2, y_1^{\iota}.x_1\ (\lambda y_2^{\iota}.x_2\ y_2)\ y_1$.

If a term cannot be further reduced by $\beta\overline{\eta}$ (resp. $\overline{\eta}$), then it is in $\beta\overline{\eta}$-normal form (resp. $\overline{\eta}$-normal form). It is well known that the reduction relations defined by $\beta, \overline{\eta}$ or β, η are both strongly terminating and Church–Rosser (Wolfram, 1993; Huet, 1976; Barendregt, 1984). Hence for every term t, there is a $\beta\overline{\eta}$-normal form $t{\downarrow_{\beta\overline{\eta}}}$, which is unique up to $=_\alpha$. Similarly the $\beta\eta$-normal form of t is denoted as $t{\downarrow_{\beta\eta}}$.

Remark 3.7. Let t be a term in $\overline{\eta}$-normal form. Let t' result from t by a series of β-reductions. Then t' is in $\overline{\eta}$-normal form.

Remark 3.8. Let t be a term of type τ in $\beta\overline{\eta}$-normal form. Let $m = ar(\tau)$. Then t has the form $\lambda y_1, \ldots, y_m.t'$. In particular $\#bvl(t) \geq m$. The head of any maximal application in t' is either a variable or a function symbol $f \in \Sigma$. Hence maximal applications can be written in the form $x(t_1, \ldots, t_n)$ or $f(t_1, \ldots, t_n)$. This leads to a tree representation of terms in $\beta\overline{\eta}$-normal form that closely resembles the usual tree representation of terms in first-order logic. Compare Fig. 2 below.

Proposition 3.9. *The following equivalence holds:*

$$s =_{\beta\eta} t \Leftrightarrow s{\downarrow_{\beta\overline{\eta}}} =_\alpha t{\downarrow_{\beta\overline{\eta}}} \Leftrightarrow s{\downarrow_{\beta\eta}} =_\alpha t{\downarrow_{\beta\eta}}.$$

Lemma 3.10. *A term t is in $\beta\overline{\eta}$-normal form, iff the following hold:*

- *t is in β-normal form, and*
- *every proper subterm s of t such that s is not an abstraction and s has a non-elementary type is embedded in a superterm of the form $(s\ s')$.*

Proposition 3.11. *Let $s, t,$ be terms and f, g be function symbols of appropriate types, such that g does not occur in s, t. Then $s =_{\beta\eta} t$ iff $f\ s =_{\beta\eta} f\ t$ iff $s\ g =_{\beta\eta} t\ g$.*

Proof. The first equivalence follows from reduction to $\beta\eta$-normal form. To prove the second equivalence, the direction $s =_{\beta\eta} t \Rightarrow s\ g =_{\beta\eta} t\ g$ is trivial, since it follows from the congruence property. To prove the other direction, let $s\ g =_{\beta\eta} t\ g$. Note that g does not occur in s, t. Since reduction makes no difference between function symbols and free variables, this implies that $s\ x =_{\beta\eta} t\ x$, where x is fresh variable. From congruence it follows that $\lambda x.s\ x =_{\beta\eta} \lambda x.t\ x$. Then we can use (η) on both sides of the equation and obtain $s =_{\beta\eta} t$. $\square$

Lemma 3.12. *Let x be a variable of type τ. Then the $\beta\overline{\eta}$-normal form of x has size at most $3 * size(\tau)$.*

Proof. We use induction on the size of τ. If x has type ι, then we are ready. If x has type $\tau = \alpha_1 \to \cdots \to \alpha_n \to \iota$, then $(\overline{\eta})$-reductions transform x into $\lambda x_1^{\alpha_1}, \ldots, x_n^{\alpha_n}.x\ x_1 \ldots x_n$. The size is $4n + 1$. The final $\beta\overline{\eta}$-normal form is

$$\lambda x_1^{\alpha_1}, \ldots, x_n^{\alpha_n}.x\ x_1{\downarrow_{\beta\overline{\eta}}} \ldots x_n{\downarrow_{\beta\overline{\eta}}}.$$

This gives a size of $3n + 1 + \sum_i (size(x_i{\downarrow_{\beta\overline{\eta}}}))$. By induction this is smaller than $3 \cdot (n + 1 + \sum(size(\alpha_i))) = 3 \cdot size(\tau)$. $\square$

Lemma 3.13. *If a ground term t is in $\beta\overline{\eta}$-normal form, and $\#bvl(t) = 0$, then t is ground first-order term and of elementary type.*

This is wrong for non-ground terms, since $x^{\iota\to\iota}\ a$ is in $\beta\overline{\eta}$-normal form, but not a first-order term.

3.6. Upper bounds on sizes of normal forms

There are estimations on the length of reduction sequences for various lambda-calculi (see Beckmann, 2001; Gandy, 1980; Schwichtenberg, 1982, 1991). We adapt this to our purposes and argue that there is a computable upper bound on the size of a $\beta\overline{\eta}$-normal form of a term t. Note that there are also lower bounds for the complexity (Statman, 1979; Beckmann, 2001).

We need an estimation on the size of normal forms depending on the starting term. Let $2_0(n) := n$ and $2_m(n) = 2^{2_{m-1}(n)}$ for $m > 0$. Let $maxtypesize(t)$ be the maximal size of the types of subterms of t, i.e. $maxtypesize(t) := max\{size(\tau) \mid \tau \in types(t)\}$.

Lemma 3.14. *Let t be a term. Then the size of the $\overline{\eta}$-normal form of t is at most $seqnf(t) := 3 \cdot size(t) \cdot maxtypesize(t)$.*

Proof. Lemma 3.12 shows the claim for variables. For each subterm of t that is a non-maximal application in the sense that some arguments are not made explicit, we may add $\overline{\eta}$-normal forms of appropriate variables as arguments and corresponding lambda-binders, as in the proof of Lemma 3.12. The enlargement of the size that results from treating one subterm in this way is bound by $3 \cdot maxtypesize(t) - 1$. Since the total number of subterms that represent non-maximal applications in the above sense does not exceed $size(t)$, the size of the final term is bound by $size(t) \cdot (3 \cdot maxtypesize(t) - 1) + size(t) = seqnf(t)$. □

Theorem 3.15. *Let t be a term. Then the size of the $\beta\overline{\eta}$-normal form of t is not greater than $sbeqnf(t) := seqnf(t)^{2_{deg(t)+1}(seqnf(t))}$.*

Proof. We first transform t into its $\overline{\eta}$-normal form t'. Lemma 3.14 shows that the size of t' is bound by $seqnf(t)$. It is simple to see that $deg(t) = deg(t')$. Remark 3.7 shows that only β-reductions are needed to reach the $\beta\overline{\eta}$-normal form t'' of t. Using the result in (Beckmann, 2001), who shows that the number of reductions of a term r is at most $2_{deg(r)}(len(r))$, we obtain an upper bound $2_{deg(t)}(seqnf(t))$ for the number of β-reductions that are needed to reach t''. Since every β-reduction step may increase the size of a term at most by squaring it, it follows that $size(t'') \leq sbeqnf(t)$. □

4. Bounded higher-order unification problems

In this section we formally introduce the decision problems that are studied in this paper. In the following, let $\Sigma_0 \subseteq \Sigma$ denote a subsignature.

Definition 4.1. A *higher-order unification problem (HOUP)* is a finite multiset S of equations $\{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$ where s_i, t_i are terms with $type(s_i) = type(t_i)$ for all i. A closed (Σ_0-) substitution σ such that $\sigma(s_i) =_{\beta\eta} \sigma(t_i)$ for $1 \leq i \leq n$ is called a (Σ_0-) *unifier* of S.

At several places in the algorithm we will exploit the symmetry of equations by replacing $s \doteq t$ by its symmetric equivalent $t \doteq s$. If this is done, we state it explicitly. We use the notation S^T for the HOUP where all equations are transposed, i.e. $S^T := \{s \doteq t \mid t \doteq s \in S\}$.

Definition 4.2. Let S be a HOUP; let $b : FV(S) \to \mathbb{N}_0$ be a function. Then the pair (S, b) is called a *bounded HOUP (BHOUP)*. A (Σ_0-) substitution σ is a (Σ_0-) *unifier* of (S, b) iff all terms in the codomain of σ are in $\beta\overline{\eta}$-normal form, σ is a (Σ_0-) unifier of S and for every variable $x \in FV(S)$ the inequality $\#bvl(\sigma(x)) \leq b(x)$ holds.

Note that in a BHOUP the size of unifiers is *not* bounded, since for example for $t = \lambda x.\, \underbrace{f(\ldots(f}_{k}\ x)\ldots)$ we have $\#bvl(t) = 2$, but the size of t grows with k. We remark that the upper bound b also provides an (implicit) upper bound on the size of types in $subt(t)$ for terms in $cod(\sigma)$ for unifiers σ.

Lemma 4.3. *Let (S, b) be a BHOUP with unifier σ. Then for every variable x with $b(x) = 0$, the variable x has elementary type and $\sigma(x)$ is a ground first-order term.*

Proof. By assumption, the terms in the codomain of σ are ground and in $\beta\overline{\eta}$-normal form. The result follows from Lemma 3.13. □

5. Properties of minimal unifiers

In this section we shall see that for solvable BHOUPs there exists a unifier that fulfills characteristic restrictions on the signature and on the number of periodical repetitions of subcontexts in the codomain terms (cf. Lemma 5.8).

A *minimal* unifier σ of a BHOUP (S, b) is a unifier such that the sum $\sum_{x \in FV(S)} size(\sigma(x))$ is minimal with respect to all unifiers of the problem.

5.1. Sufficient signatures

First it is shown that for any given BHOUP a finite signature can be described that suffices to unify the BHOUP if there is any unifier. This result will be important when formulating non-deterministic transformation rules where we "guess" function symbols occurring in unifiers. In such a context it helps to guarantee a finite branching of the search tree.

Lemma 5.1. *Let σ be a minimal unifier of the BHOUP (S, b). Then the following hold:*

1. *Every function symbol that is not an elementary constant occurring as a subterm in the codomain of σ also occurs in S.*
2. *All types of elementary constants, variables and applications occurring as subterms in the codomain of σ are in $subt(S)$.*
3. *The maximal arity of types of variables occurring as subterms in $cod(\sigma)$ is not greater than the maximal arity of types in $subt(S)$.*
4. *If $\Sigma_0 \subseteq \Sigma$ is any subsignature that contains all function symbols occurring in S and in addition at least one elementary constant a^ι for each (elementary) target type ι in $subt(S)$, then there exists a minimal unifier σ' of (S, b) that is a Σ_0-unifier.*

Proof. 1. Let (S, b) be a BHOUP and σ be a minimal unifier of (S, b). Assume there exists a non-elementary function symbol f in the codomain of σ that does not occur in S. Since the terms in the codomain of σ are in $\beta\overline{\eta}$-normal form each occurrence of f is the head of a term of the form $f(t_1, \ldots, t_{ar(f)})$. Let ι be the target type of f, let a be an elementary constant of type ι. Then replace every occurrence of a term of the form $f(t_1, \ldots, t_{ar(f)})$ in the codomain of σ by a. The constructed substitution σ' remains in $\beta\overline{\eta}$-normal form. If $s \doteq t$ is an equation of S, then $\sigma'(s)\downarrow_{\beta\overline{\eta}}$ is obtained from $\sigma(s)\downarrow_{\beta\overline{\eta}}$ by replacing each subterm of the form $f(t_1, \ldots, t_{ar(f)})$ by a, and similarly for $\sigma'(t)\downarrow_{\beta\overline{\eta}}$ and $\sigma(t)\downarrow_{\beta\overline{\eta}}$. This follows from the fact that f does not occur in s, t. Now $\sigma(s)\downarrow_{\beta\overline{\eta}} =_\alpha \sigma(t)\downarrow_{\beta\overline{\eta}}$ implies $\sigma'(s)\downarrow_{\beta\overline{\eta}} =_\alpha \sigma'(t)\downarrow_{\beta\overline{\eta}}$, since in the normal forms all the subterms starting with f are replaced by the subterm a. This shows that σ is a unifier for (S, b). Since σ' has a smaller size, this is a contradiction.
2. First of all, the top terms in the codomain have a type in *subt*(S).
We use induction on the structure of the terms in the codomain. Suppose a subterm $\lambda x_1, \ldots, x_m. f(t_1, \ldots, t_n)$ of a term in the codomain has a type in *subt*(S). Then the type of each variable x_i is in *subt*(S) $(1 \le i \le m)$. Part 1 shows that for $n > 0$ we may assume that the type of f, and hence the type of any t_i, is in *subt*(S) $(1 \le i \le n)$. Suppose a subterm $\lambda x_1, \ldots, x_m.x_i(t_1, \ldots, t_n)$ has a type in *subt*(S) where $1 \le i \le n$. Then the type of x_i, and hence the type of any argument t_i, is in *subt*(S) $(1 \le i \le n)$. The result follows by induction.
3. Follows from the previous part.
4. Let a^ι be an elementary constant occurring in the codomain of σ that does not occur in S. Part 2 shows that ι is a target type in *subt*(S). Hence, as in Part 1 of the proof a^ι can be replaced by an elementary constant $b^\iota \in \Sigma_0$ to obtain σ' from σ. □

5.2. The exponent of periodicity

The *exponent of periodicity* (see also Schmidt-Schauß and Schulz, 1998) of a unifier σ of (S, b) is the maximal number n such that for some variable x occurring in S the image $\sigma(x)$ contains a subterm of the form $C^n[t]$, where C is a nontrivial ground first-order context. Note that $n > 1$ implies that the type of the context C and the type of the hole of C are equal.

Definition 5.2. Let t be a ground term in $\beta\overline{\eta}$-normal form. Assume that we color in t the positions of

1. each of the n lambda-binders in expressions $\lambda x_1, \ldots, x_n$ occurring in t,
2. each occurrence of a bound variable in t,
3. each occurrence of a function symbol f in an expression $f(t_1, \ldots, t_n)$ where either
   (a) f contains an argument of non-elementary type, i.e. f is not first-order, or
   (b) there are at least two subterms t_{i_1}, t_{i_2} $(i_1 \neq i_2)$ such that t_{i_j} for $j = 1, 2$ contains an occurrence of a variable or a lambda.

The uncolored positions of t can be considered as the nodes of a Böhm-like tree where links correspond to immediate subterm relationship. Each maximal connected uncolored component either defines a ground first-order term or a ground first-order context. They are called the *maximal first-order subterms/subcontexts* of t. The *representation*

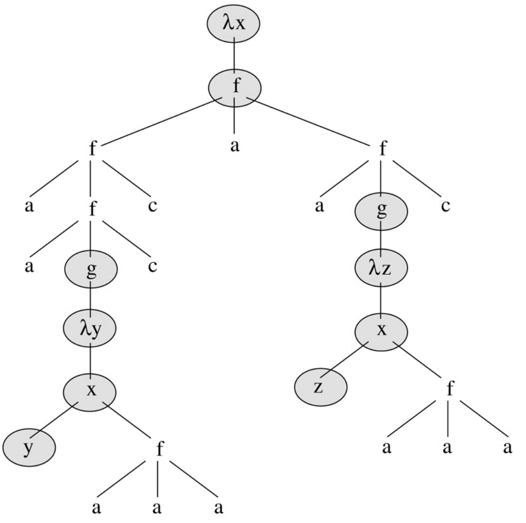

Fig. 2. Colored positions and maximal first-order subterms and subcontexts.

size of t, *repsize*(t), is defined similarly to the size of t, but each maximal first-order subterm/subcontext yields a uniform contribution of 1.

Intuitively, in the *repsize*-measure, maximal first-order subterms/subcontexts are treated as primitive symbols.

Example 5.3. The ground term t depicted in Fig. 2 is colored in the above sense. There are two occurrences of the maximal first-order subterm $f(a, a, a)$, one occurrence of the maximal first-order subterm a, one occurrence of the maximal first-order context $f(a, f(a, [\cdot], c), c)$, and one occurrence of the maximal first-order context $f(a, [\cdot], c)$. Hence the maximal first-order subterms/subcontexts yield a contribution of 5 to *repsize*(t). Note that coloring first-order function symbols of type 3(b) helps to avoid the use of maximal first-order *multi*-contexts. This may be seen from the topmost colored symbol f in the figure.

Lemma 5.4. *Let t be a ground term in $\beta\overline{\eta}$-normal form, colored as above. Let $k := \#bvl(t)$. Then there are at most $3k$ colored positions. The number of maximal first-order (uncolored) subcontexts of t does not exceed $3k$. The number of maximal first-order (uncolored) subterms of t does not exceed $1+3k\cdot(1+maxar(t))$ where $maxar(t)$ is the maximal arity of a type in $subt(t)$. The representation size $repsize(t)$ does not exceed $2+22k+6k\cdot maxar(t)$.*

Proof. In t we have at most k colored positions corresponding to lambda-binders or occurrences of bound variables. If f has an argument of non-elementary type, then this argument starts with a lambda-binder. Hence there are at most k colored positions of type 3(a). In addition there are at most $k-1$ colored positions of type 3(b). This gives a total of at most $3k-1$ colored positions. For each colored position π, there is at most one maximal first-order subcontext of t that ends at the position (in the sense that the subterm with top position π represents the argument of the context). Hence the number of maximal first-order subcontexts of t does not exceed $3k-1$. The number of maximal first-order subterms of t is at most 1 (for the root) plus the sum of the number of immediate subterms of colored positions. It can be estimated by $1 + (3k-1) \cdot (1 + \mathit{maxar}(t))$. As to $\mathit{repsize}(t)$, the total contribution of maximal first-order subterms/subcontexts is $3k - 1 + 1 + (3k-1) \cdot (1 + \mathit{maxar}(t)) = (3k-1) \cdot \mathit{maxar}(t) + 6k - 1$. The maximal contribution of λ-binders and bound variables is $\leq 2k$ (recall that a binder λx yields a size contribution of 2). The function symbols at colored positions can contribute $3k-1$, each single symbol yielding a size contribution of 1. Ignoring applications, this yields a total bound of $(3k-1) \cdot \mathit{maxar}(t) + 11k - 2$. Since each application yields an additional contribution of 1, a bound for $\mathit{repsize}(t)$ is $(6k-2) \cdot \mathit{maxar}(t) + 22k - 4$. □

Definition 5.5. Let (S, b) be a BHOUP. Let $\mathit{maxar}(S)$ denote the maximal arity of a type in $\mathit{subt}(S)$; let maxb be the maximal value $b(x)$ for variables in $FV(S)$. Then the number

$$\mathit{repn}(S, b) := (6 \cdot \mathit{maxb} - 2) \cdot \mathit{maxar}(S) + 22 \cdot \mathit{maxb} - 4$$

is called the *representation number* of (S, b).

Lemma 5.6. *Let (S, b) be a BHOUP, and σ be a minimal unifier of (S, b). Then the representation size of any term in the codomain of σ is at most $\mathit{repn}(S, b)$.*

Proof. This follows from Lemmas 5.4 and 5.1. □

The important point to note is that the above estimate for the representation size does not depend on σ. In the following we use some of the previously introduced measuring functions also for HOUPs S as follows. If $S = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$, then $\mathit{terms}(S)$ is the multiset of all terms s_i and t_i $(i = 1, \ldots, n)$. Now we can use the functions $\mathit{ord}, \mathit{deg}, \mathit{size}, \mathit{maxtypesize}, \mathit{subt}, \mathit{seqnf}, \mathit{sbeqnf}$ also for S by applying them to $\mathit{terms}(S)$, and use the obvious operators for extending the functions to multisets.

Lemma 5.7. *There is a positive real constant c_0 such that for every unifiable BHOUP (S, b) the exponent of periodicity of any minimal unifier of (S, b) is less than $2^{(c_0 + 2.14 \cdot \mathit{finsize}(S))}$, where*

$$\mathit{finsize}(S) := 2_{\mathit{deg}(S)+1}(\mathit{repn}(S, b) \cdot \mathit{sbeqnf}(S)).$$

Proof. Let (S, b) be a BHOUP and let σ be a minimal unifier of (S, b). Let $\mathit{terms}(\sigma(S))$ denote the multiset of image terms $\{\sigma(s) \mid s \in \mathit{terms}(S)\}$. In each term $\sigma(s)$ we consider the occurrences of codomain terms $\sigma(x)$ that represent the images of the variables occurring in s under σ. For each such occurrence we consider the maximal first-order subterms/subcontexts of the respective codomain term as primitive symbols. Each such subterm/subcontext will be called an *inner codomain subterm/subcontext*, stressing its origin in a codomain term. By Lemma 5.6, the sum of the sizes of all terms in $\mathit{terms}(\sigma(S))$ with respect to this representation is bound by $\mathit{size}(S) \cdot \mathit{repn}(S, b)$.

When we compute the $\beta\overline{\eta}$-normal form of the terms in $\mathit{terms}(\sigma(S))$, the inner codomain subterms/subcontexts are not destroyed. For the reduction they can be considered as primitive symbols as well. Hence it follows from Theorem 3.15 that the corresponding representation for the normalized image terms in the set $\{\sigma(s)\downarrow_{\beta\overline{\eta}} \mid s \in terms(S)\}$ has representation size not exceeding $\mathit{finsize}(S)$ as defined in the lemma.

Now we use the fact that the $\beta\overline{\eta}$-normal forms of the left- and right-hand side of equations are α-equal to extract a context unification problem (Schmidt-Schauß, 1999b, 2002; Schmidt-Schauß and Schulz, 2002b). This can be done by equating the following:

- the maximal ground first-order terms in equations $\sigma(s)\downarrow_{\beta\overline{\eta}} =_\alpha \sigma(t)\downarrow_{\beta\overline{\eta}}$ at corresponding positions,
- the maximal ground first-order contexts in equations $\sigma(s)\downarrow_{\beta\overline{\eta}} =_\alpha \sigma(t)\downarrow_{\beta\overline{\eta}}$ at corresponding positions,

for $s \doteq t \in S$. Note that all the inner codomain subterms/subcontexts are contained in some maximal first-order term/context. The context unification problem CUP is formed from the equations $\sigma(s)\downarrow_{\beta\overline{\eta}} =_\alpha \sigma(t)\downarrow_{\beta\overline{\eta}}$ ($s \doteq t \in S$) as follows: The inner codomain contexts are replaced consistently by context variables, and the inner codomain terms are consistently replaced by first-order variables.

The total number of occurrences of variables and function symbols in CUP does not exceed $\mathit{finsize}(S)$. The results in Schmidt-Schauß and Schulz (1998) show that there exists a fixed real constant c_0 such that the exponent of periodicity of a minimal unifier for CUP is smaller than $2^{c_0+2.14\cdot \mathit{finsize}(S)}$.

We consider the unifier σ' of CUP that assigns to each context variable the corresponding inner codomain context, and to each first-order variable the corresponding inner codomain term. We show that σ' is a minimal unifier for CUP. It then follows that the exponent of periodicity of σ', and hence the exponent of periodicity of σ, does not exceed $2^{c_0+2.14\cdot \mathit{finsize}(S)}$.

Assume that σ' is not a minimal unifier for CUP. In Schmidt-Schauß and Schulz (1998), in order to turn a non-minimal unifier for CUPs into a minimal one, subcontexts of the form CC^mC occurring in the images of variables are replaced by similar subcontexts of the form CC^nC. An appropriate selection of the numbers n guarantees that the new values define a unifier for the CUP that satisfies the above bound. Since for a context C of type ι, C^k always has type ι for $k \geq 0$, this shows that the same techniques as were used for modifying non-minimal unifiers can be applied in the situation of bounded higher-order unification, where types have to be respected. This shows that a smaller unifier of CUP could be retranslated into a smaller unifier for (S, b), which yields a contradiction. $\square$

From Lemmas 5.1 and 5.7 we obtain:

Lemma 5.8. *Let (S, b) be a BHOUP. Let $\Sigma_0 \subseteq \Sigma$ be a finite signature that contains all function symbols occurring in S and in addition at least one elementary constant a^ι for each target type ι in $subt(S)$. Let $E := 2^{(c_0+2.14\cdot \mathit{finsize}(S))}$ where $\mathit{finsize}(S) := 2_{deg(S)+1}(repn(S, b) \cdot sbeqnf(S))$. Then (S, b) has a Σ-unifier iff (S, b) has a minimal Σ_0-unifier where the exponent of periodicity does not exceed E.*

6. Order reduction

Assume that we want to decide unifiability of a BHOUP (S, b) over a finite signature Σ_0. Consider a minimal Σ_0-unifier σ of (S, b) and let $\sigma(x)$ be the value of $x \in FV(S)$.

Lemma 5.6 shows that there is an upper bound for the representation size of $\sigma(x)$. Assume now that we replace in $\sigma(x)$ maximal first-order subterms by first-order variables and maximal first-order subcontexts by unary second-order variables with bound 2. The resulting term will be called a *solution scheme* for the variable x of (S, b). We will see that — modulo an inessential renaming of variables — there are only a finite number of possible solution schemes for a given variable $x \in FV(S)$, and all solution schemes for x are effectively computable. We may use this idea to simplify a given BHOUP, guessing possible solution schemes. This motivates the following definitions and steps.

Definition 6.1. Let (S, b) be a BHOUP. A *I-variable* is a first-order variable $x \in FV(S)$ with bound $b(x) = 0$. A *II-variable* is a variable $x \in FV(S)$ of type $\iota \to \iota'$ with bound $b(x) = 2$.

Definition 6.2. Let (S, b) be a BHOUP; let $\Sigma_0 \subseteq \Sigma$ be a finite signature; let $x \in FV(S)$. The term t_x in $\beta\overline{\eta}$-normal form is a Σ_0*-solution scheme* for x iff the following conditions hold:

1. the function symbols of t_x are from signature Σ_0,
2. for each occurrence of a subterm $f(t_1, \ldots, t_n)$ ($f \in \Sigma_0$) in t_x, f always contains an argument of non-elementary type, or there are at least two distinct subterms t_{i_1}, t_{i_2} ($i_1 \neq i_2$) such that both subterms contain an occurrence of a variable or a lambda,
3. the free variables in t_x are I-variables or II-variables,
4. $\#bvl(t_x) \leq b(x)$,
5. each free II-variable has as its argument either a term starting with a lambda, a bound variable, or a term of the form $f(t_1, \ldots, t_n)$,
6. the types of all symbols in t_x are from the set $subt(S)$,
7. the size of t_x is not greater than $repn(S, b)$.

Lemma 6.3. *Let (S, b), Σ_0 and $x \in FV(S)$ as above. Then, modulo renaming of variables there exist only a finite number of Σ_0-solution schemes for x. The set of Σ_0-solution schemes for x is effectively computable.*

Proof. By Lemma 5.6, the size of any Σ_0-solution scheme t_x is not greater than $repn(S, b)$. Hence there is an upper bound on the number of positions of Σ_0-solution schemes t_x. The function symbols occurring in t_x are from the finite signature Σ_0. There are only a finite collection of possible types for the variables occurring in t_x; given S, the set of possible types is effectively computable. Hence the result follows. □

Definition 6.4. A *reduced BHOUP* (RBHOUP) is a BHOUP (S, b) where each variable $x \in FV(S)$ is a I-variable or a II-variable.

Definition 6.5. Let $\Sigma_0 \subseteq \Sigma$. A Σ_0-unifier σ of the RBHOUP (S, b) is *pure* iff the following conditions hold:

1. for each I-variable x in $FV(S)$, $\sigma(x)$ is always a ground first-order term,
2. for each II-variable x in $FV(S)$, $\sigma(x)$ is always a term of the form $\lambda u.s$ where u is a first-order variable, s is a first-order term with at most one occurrence of u. The term s does not have any occurrence of another variable.

Definition 6.6 (*Order-Reduction*). Let the BHOUP (S, b) be given.

1. Define $E := 2^{(c_0+2.14\cdot \mathit{finsize}(S))}$ where $\mathit{finsize}(S) := 2_{deg(S)+1}(\mathit{repn}(S, b) \cdot \mathit{sbeqnf}(S))$ as a bound for the exponent of periodicity.
2. Select ("don't care") a finite signature $\Sigma_0 \subseteq \Sigma$ that contains all function symbols occurring in S and in addition at least one elementary constant a^ι for each target type ι in $\mathit{subt}(S)$.
3. For each variable $x \in FV(S)$, guess a Σ_0-solution scheme t_x for x (among a finite set of possible schemes for Σ_0; cf. Lemma 6.3) and replace all occurrences of x by t_x. Then reduce all terms to $\beta\overline{\eta}$-normal form.

Theorem 6.7. *Let (S, b) be a BHOUP. Let Σ_0 be a signature selected in Step* 2 *of the rule (Order-Reduction).*

1. *(Finite branching degree of (Order-Reduction)): For Step* 3 *there are a finite number of output systems.*
2. *(Soundness of (Order-Reduction)): Let (S', b') denote a RBHOUP resulting from Step* 3. *If (S', b') has a pure Σ_0-unifier, then (S, b) has a unifier.*
3. *(Completeness of Order-Reduction)): If (S, b) has a unifier, then there exists an output system (S', b') that may be selected in Step* 3 *that has a pure Σ_0-unifier σ' with exponent of periodicity not greater than E.*

Proof. Part 1 is trivial.

Part 2: Let t_x denote the image of $x \in FV(S)$ under the Σ_0-solution scheme selected in Step 3. Let σ' denote a pure Σ_0-unifier of (S', b'). Let σ map each $x \in FV(S)$ to the $\beta\overline{\eta}$-normal form of $\sigma'(t_x)$. Obviously σ solves all equations of S. It remains to show that $\#\mathit{bvl}(\sigma(x)) \leq b(x)$ for all $x \in FV(S)$. Clearly the images $\sigma'(z)$ of I-variables z introduced in t_x do not yield any contribution to $\#\mathit{bvl}(\sigma(x))$. Let z be a II-variable introduced in t_x. Here $\sigma'(z)$ has the form $\lambda u.s$ and s is a first-order term with at most one occurrence of a variable, which is then u. Since each occurrence of z in t_x has a subterm of t_x as its argument, after $\beta\overline{\eta}$-reduction the binder λu and the occurrence of u (if there is such an occurrence) disappear. This shows that the images $\sigma'(z)$ of II-variables z do not contribute to $\#\mathit{bvl}(\sigma(x))$. Since $\#\mathit{bvl}(t_x) \leq b(x)$ it follows that $\#\mathit{bvl}(\sigma(x)) \leq b(x)$.

Part 3: Let (S, b) be unifiable. Then, by Lemma 5.8, (S, b) has a minimal Σ_0-unifier σ where the exponent of periodicity does not exceed E. Let $x \in FV(S)$. By Lemma 5.6 the representation size of $\sigma(x)$ is at most $\mathit{repn}(S, b)$. Replacing in $\sigma(x)$ maximal first-order subterms by I-variables and maximal first-order contexts by II-variables x_j of the corresponding type we receive the Σ_0-solution scheme t_x: it follows from Lemma 5.1 that t_x satisfies the conditions of Definitions 6.6 and 6.2. Let (S', b') denote the problem obtained from Step 2 of (Order-Reduction) where each variable $x \in FV(S)$ is replaced by t_x. Then (S', b') has the obvious Σ_0-unifier σ' where each I-variable (resp. II-variable) z is mapped to the corresponding maximal first-order subterm (resp. subcontext) of $\sigma(x)$. Clearly σ' is pure. The exponent of periodicity of σ' does not exceed the exponent of periodicity of σ. $\square$

The preceding theorem permits to reduce decidability of BHOUPs to the question of decidability of RBHOUPs.

Remark 6.8. Now the problem is similar to a bounded second-order unification problem; however, there are the following differences. An RBHOUP is typed and it may contain abstractions. The following examples illustrate the problems when trying to encode RBHOUPs as bounded second-order unification problems:

1. The RBHOUP $X(y) \doteq f(X(z))$ is unifiable in an untyped signature, a family of solutions is $X = f^n[], z = a, y = f(a)$. But it is not unifiable in a typed signature with the following typing: $X^{\iota_1 \to \iota_2}, y^{\iota_1}, z^{\iota_1}, f^{\iota_2 \to \iota_2}$: X cannot be a constant function and cannot be the identity; therefore, it must be of the form $X = \lambda u. f(X'\ u)$, where X' has the same type as X.
2. The BHOUP $f(\lambda x.x, a) \doteq f(\lambda y.y, a)$ is unifiable, contains abstractions, and if it is treated as a bounded second-order problem, the names of the bound variables have to be made equal in advance, which appears to be not appropriately encodable.
3. The BHOUP $\lambda x.x \doteq \lambda x.y$ with appropriate types has to be encoded as $a_x \doteq y$ with the condition that a_x does not occur in the instantiation of y. This is not treated in the proof of the decidability of bounded second-order unification in Schmidt-Schauß (1999a, 2004).

In the following, we almost exclusively treat RBHOUPs and are looking for pure unifiers of RBHOUPs. Since in this case the bound b in (S, b) is implicit, we simplify notation in this case and omit b.

7. The decision algorithm

Order reduction represents the first step of our decision algorithm for bounded higher-order unification. The remaining steps are based on decomposition and transformation rules of a particular form. In this section we first introduce the concepts that yield the background for these rules. In the following, Σ_0 and E respectively represent a fixed finite signature and a bound for the exponent of periodicity selected in Steps 1 and 2 of the (Order-Reduction) rule for a given input BHOUP (S, b).

The decision algorithm is described in the last subsection.

7.1. Surface positions and cycles

In order to describe the main reduction techniques, the following notions play a central role. Henceforth, by an elementary term, we mean a term with elementary type.

We define surface positions in elementary terms like for bounded second-order unification: arguments of II-variables are below the surface. In addition all abstractions are also below the surface.

Definition 7.1. Let t be an elementary term. The *surface positions* of t and the subterms at these positions are defined as follows:

– ε is a surface position of t, and t is the subterm at surface position ε;[3]

[3] ε denotes the empty sequence.

– if $t = f(t_1, \ldots, t_n)$, the subterm t_i is elementary and s is the subterm of t_i at surface position p of t_i, then s is the subterm of t at surface position $i.p$;
– if $t = (x\ t_1)$ then 0, the position of x, is a surface position of t.

The *depth* of a surface position p is the length of p.

We say that a term s is *on the surface* of the elementary term t if there exists a surface position p of t such that s is the term at this position. We use the notation $t\lfloor s \rfloor$ to indicate that t has a surface occurrence of the term s.

Example 7.2. Let f be of type $(\iota \to (\iota \to \iota) \to \iota)$ and g be of type $\iota \to \iota$. Then the surface positions of $f(x^\iota, y^{\iota \to \iota})$ are $\{\varepsilon, 1\}$, the term $(f\ x^\iota)$ has no surface positions, the term $f(x^\iota, g)$ has $\{\varepsilon, 1\}$ as surface positions, $f(g(x^\iota), g)$ has $\{\varepsilon, 1, 1.1\}$, and $f((y^{\iota \to \iota} x^\iota), g)$ has $\{\varepsilon, 1, 1.0\}$ as surface positions. Note that in this example not all terms are in $\beta\overline{\eta}$-normal form.

Remark 7.3. Assume that the variable x occurring on the surface of t is replaced by a term s of type *type*(x) where the variable y occurs on the surface of s. Then y occurs on the surface of $t[s/x]$.

Note that every position in a first-order term is a surface position, and that every surface position is elementary or the position of a variable representing the head of an elementary term. Moreover, in the latter case, each node on the path from the root to the variable is labelled by a function symbol, and the argument positions determined by the direction of this path are of elementary type. In the following, to simplify index notation for cycles of equations we use expressions $i \ \mathtt{mod}{*}\ n$ where

$$i \ \mathtt{mod}{*}\ n = \begin{cases} i \ \mathtt{mod}\ n & \text{if } i \ \mathtt{mod}\ n \neq 0, \\ n & \text{if } i \ \mathtt{mod}\ n = 0. \end{cases}$$

The algorithm that is used to analyze RBHOUPs does not operate on single equations. Rather it tries to reduce combinations of equations of a particular form, called cycles.

Definition 7.4. Let S be a RBHOUP. A *cycle* is a sequence of equations between elementary terms of the form $s_1 \doteq t_1, \ldots, s_h \doteq t_h$ of length $h \geq 1$ where the following conditions hold:

1. for all $1 \leq i \leq h$: $s_i \doteq t_i \in S \cup S^T$,
2. for all $1 \leq i \leq h$: s_i has the form $(x_i\ r_i)$ or x_i, where x_i occurs on the surface of $t_{i-1 \ \mathtt{mod}{*}\ h}$,
3. there is at least one term t_i of the form $f(t_{i,1}, \ldots, t_{i,n})$ and at least one term s_i of the form $(x_i\ r_i)$.

A cycle is *path-unique* iff for every $1 \leq i \leq h$ there is only one occurrence of x_i on the surface of $t_{(i-1) \ \mathtt{mod}{*}\ h}$.

Let L be a cycle in S of the form $s_1 \doteq t_1, \ldots, s_h \doteq t_h$. For each of the terms t_i, $1 \leq i \leq h$, let C_i be the context determined as follows: Let q_i be the smallest subterm of t_i such that all surface occurrences of $x_{(i+1\ mod*\ h)}$ in t_i are also contained in q_i. The *relevant context* C_i of equation i is uniquely determined by $t_i = C_i\lfloor q_i \rfloor$.

The *length of a cycle* L is the number of equations in L. If for some cycle L, there is no other cycle in S with a smaller length, then we say L is a *minimal-length* cycle.

A cycle $s_1 \doteq t_1, \ldots, s_h \doteq t_h$ is called *compressed* iff there is no i such that s_i or t_i is a I-variable.

Example 7.5. We give some examples for cycles and non-cycles in RBHOUPs.
The sequence $x \doteq h((y\ s_1)), (y\ s_2) \doteq x$ is a (non-compressed) cycle of length 2. Note that $x, (y\ s_1)$ and $(y\ s_2)$ are elementary since y must be a II-variable. When instantiating x by $h((y\ s_1))$ we receive from the second equation a shorter cycle of the form $(y\ s_2') \doteq h((y\ s_1))$ which is compressed and path-unique. The sequence $(x_1\ s_1) \doteq f((x_1\ s_2), x_2((x_1\ s_3)))$ is a path-unique and compressed cycle of length 1. Note that $(x_1\ s_1), (x_1\ s_2)$ are elementary since x_1 must be a II-variable. The sequence $(x_1\ y_1) \doteq x_2\ (x_1\ y_1)$ is not a cycle.

7.2. A well-founded measure for termination

We now introduce the measure that is used to prove termination of the decision algorithm.

Definition 7.6. The lexicographic measure $\psi(L) = (\psi_1(L), \psi_2(L), \psi_3(L))$ of a cycle L of a RBHOUP S has the following three components:

$\psi_1 :=$ the length h of L.
$\psi_2 :=$ 0, if L is non-path-unique, 1, if L is path-unique.
$\psi_3 :=$ – if L is non-path-unique, then the minimal main depth of the relevant contexts C_j of L where t_j contains at least two different surface occurrences of $x_{(j+1) \text{ mod}* h}$.
– if L is path-unique, then the number of indices $1 \leq i \leq h$ such that C_i is not trivial.

Definition 7.7. The measure μ of a RBHOUP S is a lexicographic one with the components $\mu_1, \mu_2, \mu_3, \mu_4, \mu_5$:

$\mu_1 :=$ the number of distinct II-variables occurring in S.
$\mu_2 :=$ if there is a cycle in S, then $min\{\psi(L) \mid L$ is a cycle in $S\}$; otherwise, ∞.
$\mu_3 :=$ the number of lambda-bound variables occurring in S.
$\mu_4 :=$ the number of occurrences of function symbols in S on surface positions of the terms that constitute the left-hand and right-hand sides of the equations in S.
$\mu_5 :=$ the number of I-variables in S.

Lemma 7.8. *The measure μ for RBHOUPs is well-founded.*

7.3. Definitions of soundness and completeness

Definition 7.9. Let $\Sigma_0 \subseteq \Sigma$ and E as above. A non-deterministic transformation rule $\mathcal{T}$ that transforms a RBHOUP S into another RBHOUP S, offering a finite number of alternatives, is called:

– *Sound (for subsignature Σ_0)* if whenever S is transformed by $\mathcal{T}$ into S', and S' has a pure Σ_0-unifier, then S has a pure Σ_0-unifier.

– *Complete (for bound E and subsignature Σ_0)* iff the following holds: If S has a pure Σ_0-unifier σ with exponent of periodicity not greater than E, then there exists an RBHOUP S' such that $\mathcal{T}$ can transform S into S' where S' has a pure Σ_0-unifier with exponent of periodicity not greater than E.
– *Deterministically complete (for bound E and subsignature Σ_0)* iff the following holds: If S has a pure Σ_0-unifier σ with exponent of periodicity not greater than E, then for all RBHOUPs S': If $\mathcal{T}$ transforms S into S', then S' has a pure Σ_0-unifier with exponent of periodicity not greater than E.

Since Σ_0 and E are selected in the (Order-Reduction) step and fixed for the rest of the algorithm we may simply talk about "soundness", "completeness" and "deterministic completeness".

7.4. Decomposition rules

The decision algorithm for BHOU operates on systems of a special kind, called "decomposed" RBHOUPs. The decomposition of an intermediate system constitutes the final step of each transformation rule.

Definition 7.10. Let Σ_0 as above. The *decomposition rules* are defined in Table 1. The rule (extend) is intended to be deterministic: the chosen name of the symbol does not matter. That is, the selection of the function symbol is "don't care". Note that in order to ensure soundness of (extend) the new function symbol f is not permitted in the codomain of unifiers (see also Example 7.15). For the application of the rules we presuppose that all terms in the RBHOUP are in $\beta\overline{\eta}$-normal form.

In every application of decomposition the failure rules have highest priority.

Definition 7.11. A RBHOUP S is *decomposed* if no decomposition rule (in particular no failure rule) is applicable.

Lemma 7.12. *Let S be a decomposed RBHOUP. Then each minimal-length cycle of S is compressed.*

Proof. Follows from the fact that rule (decomp-repvt) cannot be applied. □

Remark 7.13. Let S be a decomposed RBHOUP and L be a minimal-length (and hence compressed) cycle in S of the form $s_1 \doteq t_1, \ldots, s_h \doteq t_h$ where s_i has the form $(x_i\ r_i)$. Then the relevant context C_i of equation i does not have any surface occurrence of a variable x_j for $j = 1, \ldots, h$. In fact, by definition, C_i cannot contain a surface occurrence of x_{i+1}. If C_i would have a surface occurrence of a variable $x_j \neq x_{i+1}$, then L cannot be a minimal-length cycle.

Lemma 7.14. *The decomposition rules are sound and deterministically complete.*

Proof. *Soundness* of (decomp), (repvv) and (decomp-repvt) is trivial. Let S' result from S by the application of (extend), and let σ be a pure Σ_0-unifier for S'. Since σ is ground, the variable u does not occur in its codomain. We have $\sigma(s[f/u]) =_{\beta\eta} \sigma(t[f/u])$. We obtain $\sigma(s)$ by replacing in $\sigma(s[f/u])$ all symbols f by u. In the same way we obtain $\sigma(t)$ from

Table 1
The decomposition rules

(decomp)	$\dfrac{\{f(s_1,\ldots,s_n) \doteq f(t_1,\ldots,t_n)\} \cup S}{\{s_1 \doteq t_1,\ldots,s_n \doteq t_n\} \cup S}$	
(extend)	$\dfrac{\{\lambda u^\tau.s \doteq \lambda u^\tau.t\} \cup S}{\{s[f/u] \doteq (t[f/u])\} \cup S}$	where $f \in \Sigma \setminus \Sigma_0$ is a fresh function symbol of type τ.
(repvv)	$\dfrac{\{x \doteq y\} \cup S}{S[y/x]}$	If x, y are I-variables.
(decomp-repvt)	$\dfrac{\{s_1 \doteq s_2\} \cup S}{\{s_1 \doteq s_2\} \cup S'}$	If $s_1 \doteq s_2$ is $x \doteq t$ or $t \doteq x$, where x is a I-variable. S' is constructed from S by replacing all surface occurrences of x by t. Conditions for applications are: The rule (occurs-check) is not applicable; there must be a minimal-length cycle L that is not compressed, and $x \doteq t$ must be an equation in the cycle L.
Failure rules:		
(clash)	$\dfrac{\{f(s_1,\ldots,s_n) \doteq g(t_1,\ldots,t_m)\} \cup S}{Fail}$	if $f \neq g$
(occurs-check)	$\dfrac{S}{Fail}$	if there is a chain of equations $x_1 \doteq t_1\lfloor x_2\rfloor,\ldots,x_{n-1} \doteq t_{n-1}\lfloor x_n\rfloor, x_n \doteq t_n\lfloor x_1\rfloor$ in $S \cup S^T$, such that for all $i = 1,\ldots,n$, x_i is a I-variable, and x_i occurs on the surface of $t_{(i-1) \mathtt{\ mod*\ } n}$; and for some $i = 1,\ldots,n$, the term t_i is of the form $f(t_{i,1},\ldots,t_{i,ar(f)})$.

$\sigma(t[f/u])$. Since f does not occur in the codomain of σ, this implies $\sigma(s) =_{\beta\eta} \sigma(t)$. Since $u \notin dom(\sigma)$, this implies $\sigma(\lambda u.s) =_{\beta\eta} \lambda u.\sigma(s) =_{\beta\eta} \lambda u.\sigma(t) =_{\beta\eta} \sigma(\lambda u.t)$.

For *deterministic completeness* first note that if a failure rule applies, then the input system does not have a unifier. Deterministic completeness of rules (decomp), (repvv) and (decomp-repvt) is trivial.

Let σ be a pure Σ_0-unifier of the input system. Assume that (extend) is applied in the form described in Table 1. We have $\sigma(s) =_{\beta\eta} \sigma(t)$. Since σ is ground, variable u does not occur in its codomain. It follows that $\sigma(s[f/u]) =_{\beta\eta} \sigma(t[f/u])$. This shows that σ unifies the system reached with (extend). Deterministic completeness of (extend) follows. □

Example 7.15. Soundness of the algorithm is an issue. In particular the rule (extend) enforces a careful usage of signatures. Consider the equation

$$\lambda x.y \doteq \lambda x.fx,$$

where y is a I-variable, and f is a function symbol. This equation has no unifier, since y cannot be instantiated with a term containing the free variable x, since instantiation is capture-free.

After applying (extend), the new equation is

$$y \doteq fg$$

where g is a function symbol from $\Sigma \setminus \Sigma_0$.

After an imitation instantiation $y \to f\ y'$ and a subsequent decomposition the system is

$$y' \doteq g.$$

This is unifiable as a first-order unification problem, but there is no unifier using symbols from Σ_0.

Hence soundness of decomposition requires restricting imitation instantiations to symbols from Σ_0.

Lemma 7.16. *If in a RBHOUP S all terms are in $\beta\overline{\eta}$-normal form, then each non-failing decomposition leaves μ_1 invariant and strictly reduces the measure μ. Thus the repeated application of decomposition rules terminates.*

Proof. Rule (decomp) does not modify μ_1. Since equations $f(s_1, \ldots, s_n) \doteq f(t_1, \ldots, t_n)$ do not occur in cycles, μ_2 can only be decreased. The rule does not modify μ_3. Since terms are in $\beta\overline{\eta}$-normal form, $f(s_1, \ldots, s_n)$ and $f(t_1, \ldots, t_n)$ are elementary and the given occurrence of f is on the surface. Hence the rule strictly reduces μ_4.
Rule (extend) does not modify μ_1. It strictly reduces μ_3. It may also decrease μ_2.

Rule (repvv) does not modify μ_1, μ_3, μ_4. It may reduce μ_2 in different ways: a minimal-length cycle may become shorter after application, or a path-unique cycle may become non-path-unique. Rule (repvv) also reduces the number of I-variables, i.e. μ_5.

Rule (decomp-repvt) does not modify μ_1. If (decomp-repvt) is applied, then $x \doteq t$ is an equation of a minimal-length cycle L. The term t has a surface occurrence of a variable y which is distinct from x since otherwise (occurs-check) would lead to failure. Looking at the predecessor and successor equations of $x \doteq t$ in L it is obvious that S' has a shorter cycle; cf. Remark 7.3. Hence μ_2 strictly decreases. □

Corollary 7.17. *Let S' result from S by applying one non-failing decomposition rule. If S contains a cycle L, then also S' contains a cycle L' such that $\psi(L') \leq \psi(L)$.*

Proof. Follows from Lemma 7.16, since μ_1 is not modified by the unfailing rules, and since S contains a cycle L. □

Lemma 7.18. *Let the RBHOUP S' result from the RBHOUP S in $\beta\overline{\eta}$-normal form by applying one non-failing decomposition rule. Then all terms of S' are in $\beta\overline{\eta}$-normal form.*

Proof. It is simple to check that the non-failing rules do not enable any new application of β-reduction or η-expansion. For (decomp-repvt) note that t cannot be an abstraction since x has elementary type. □

The following example shows that a unifiable RBHOUP may have surface occurrences of a variable and also an occurrence on a non-surface position. Moreover, it shows that the rule (decomp-repvt) does not necessarily eliminate all occurrences of a variable x.

Example 7.19. The RBHOUP $\{x^\iota \doteq y^{\iota \to \iota}\ x, \ldots\}$ is unifiable. A unifier is: $\{x \to a, y \to \lambda u_2^\iota.a\}$, where a is a constant of type ι. In this RBHOUP it is not possible to eliminate all the occurrences of x by (decomp-repvt) using $x \doteq y\ x$.

Lemma 7.20. *Let $s \doteq t$ be an equation of a decomposed RBHOUP in $\beta\overline{\eta}$-normal form. Then the type of s and t is elementary.*

Proof. Since s and t are in $\beta\overline{\eta}$-normal form, this follows from the use of the rule (extend), which replaces equations between terms of non-elementary type by terms with simpler types. □

7.5. Different types of RBHOUPs

The following definition describes a partition of the class of decomposed RBHOUPs:

Definition 7.21. A decomposed RBHOUP S is of

- *type "xy"* if S does not have any cycles, and if there is no function symbol f on the surface of S (also called pre-unified in the literature on higher-order unification),[4]
- *type "nocycle"* if S does not have any cycles, and if there exists a function symbol f on the surface of S,
- *type "amb"* (ambiguous) if S contains a cycle and if there is a ψ-minimal cycle that is non-path-unique,
- *type "unique"* if S contains a cycle and if all ψ-minimal cycles are path-unique.

Lemma 7.22. *Let S be a decomposed RBHOUP of type "xy". Let Σ_0 be as above. Assume that for every elementary type $\iota \in subt(S)$ there exists a constant of type ι in Σ_0. Then S is unifiable by a pure Σ_0-unifier.*

Proof. Let S be decomposed and of type "xy". Since decomposition rule (extend) replaces equations between abstractions, all equations of S are between terms of the form x or $(x\ t)$. Instantiate every I-variable x^ι with a^ι, and every II-variable y with a constant function of the form $\lambda y_1.a^\iota$ where ι is the target type of y. This is a unifier since it transforms every equation into an identity between elementary constants. □

Lemma 7.23. *Let S be a RBHOUP in $\beta\overline{\eta}$-normal form. After decomposition, the resulting RBHOUP S' has either type "xy", or type "nocycle", or type "amb", or type "unique". S' is in $\beta\overline{\eta}$-normal form.*

7.6. The algorithm BHOU

Let Σ_0 and E as above. The main backbone of our algorithm is the following observation on the properties of the unification rules described in Sections 10–12:

Proposition 7.24. *For each of the types "nocycle", "amb", and "unique", a transformation rule can be given that accepts a RBHOUP S in $\beta\overline{\eta}$-normal form of the given type and nondeterministically computes a successor system S' such that the following properties hold:*

[4] Since constant symbols count as function symbols, this includes there also being no constant symbol on the surface of S.

1. *The rule leads to a finite branching, i.e., for each input RBHOUP there are only a finite set of possible successor systems. The set of all successors of an input RBHOUP is effectively computable.*
2. *Each successor RBHOUP is decomposed and in $\beta\overline{\eta}$-normal form.*
3. *For every successor RBHOUP S' we have $\mu(S') < \mu(S)$.*
4. *The transformation rules are sound for Σ_0 and complete for bound E and Σ_0.*
5. *For every successor RBHOUP S' we have $subt(S') \subseteq subt(S)$.*

The transformation rule for RBHOUPs of type "nocycle" is described in Section 10. The above properties for the rule are proved in Lemmas 10.3–10.5. The transformation rule for RBHOUPs of type "amb" is given in Section 11. The above properties are shown in Lemmas 11.2–11.4. The transformation rule for RBHOUPs of type "unique" is given in Section 12. The above properties are shown in Lemmas 12.10 and 12.12–12.14. Compare Remark 12.15.

The decision procedure is described as a non-deterministic algorithm. The tree spanned by all possible rule applications is considered in the following proof.

Definition 7.25 (*Algorithm BHOU*). The *input* is a BHOUP (S_{inp}, b_{inp}).

1. Transform (S_{inp}, b_{inp}) using (Order-Reduction). We obtain
   (a) a bound E for the exponent of periodicity,
   (b) an output RBHOUP S'_{inp} in $\beta\overline{\eta}$-normal form,
   (c) a finite subsignature $\Sigma_0 \subseteq \Sigma$ that contains all function symbols occurring in S_{inp}, S'_{inp} and in addition at least one elementary constant a^ι for each target type ι in $subt(S_{inp}) \cup subt(S'_{inp})$.
   Signature Σ_0 and bound E are fixed for the rest of the algorithm.
2. Decompose the RBHOUP S'_{inp}. We obtain the RBHOUP S''_{inp} that is in $\beta\overline{\eta}$-normal form and decomposed.

Then perform the following steps:

1. Iteratively transform the current RBHOUP S using the appropriate transformation rule as described in Sections 10–12 into a successor problem S', which is again decomposed and in $\beta\overline{\eta}$-normal form.
2. The repetition stops if either a fail occurs or it signals success: a RBHOUP of type "xy" is generated.

The input (S_{inp}, b_{inp}) is recognized as unifiable iff there exists an execution possibility of BHOU such that success results.

Using Proposition 7.24, we are now able to prove the main result. As mentioned above, the missing proof of the proposition is postponed to the following sections.

Theorem 7.26. *Unifiability of BHOUPs is decidable.*

Proof. Let (S_{inp}, b_{inp}) be an input BHOUP. Let E denote the bound for the exponent of periodicity fixed in Step 1 of (Order-Reduction). Let Σ_0 denote the signature that is selected ("don't care") in Step 2 of (Order-Reduction). Define an unordered tree $\mathcal{T}(S_{inp}, b_{inp})$ in the following way. The root of $\mathcal{T}(S_{inp}, b_{inp})$ is labelled with (S_{inp}, b_{inp}). The nodes in level 1 are labelled with the possible RBHOUPs S'_{inp} that may be selected in Step 3 of (Order-Reduction). Each node of level 1 has at most one child in level 2 that is labelled with the RBHOUP S''_{inp} obtained from S'_{inp} via decomposition. If decomposition fails, the node is a blind leaf.

Let η be any node of $\mathcal{T}(S_{inp}, b_{inp})$ in level $l \geq 2$ labelled with the RBHOUP S of type "xy", "nocycle", "amb", or "unique". If S has type "xy", then η is a leaf. In the other case, for each possible successor S' of S under the appropriate transformation rule (cf. Proposition 7.24), we introduce a child η' of η labelled with S'. Each transformation uses the subsignature Σ_0 and the bound E specified above. Note that Property 5 mentioned in Proposition 7.24 guarantees that Σ_0 has the desired properties, for each transformation step.

It follows from Part 1 of Theorem 6.7 and Proposition 7.24 that $\mathcal{T}(S_{inp}, b_{inp})$ is finitely branching. Moreover, since μ is well-founded each path of $\mathcal{T}(S_{inp}, b_{inp})$ is finite. König's Lemma shows that $\mathcal{T}(S_{inp}, b_{inp})$ is finite. Part 1 of Proposition 7.24 shows that a complete traversal of $\mathcal{T}(S_{inp}, b_{inp})$ is effectively possible.

Assume that (S_{inp}, b_{inp}) has a unifier. Part 3 of Theorem 6.7 shows that there exists a node in level 1 that is labelled with a RBHOUP S'_{inp} such that S'_{inp} has a pure Σ_0-unifier with exponent of periodicity not greater than E. Lemma 7.14 shows that the unique successor RBHOUP S''_{inp} in level 2 has a pure Σ_0-unifier with exponent of periodicity not greater than E. Since the transformation rules are complete for Σ_0 and E and reduce the well-founded measure μ there exists a leaf of $\mathcal{T}(S_{inp}, b_{inp})$ that is labelled with a RBHOUP of type "xy".

Conversely, assume that $\mathcal{T}(S_{inp}, b_{inp})$ has a leaf that is labelled with a RBHOUP S of type "xy". Since we have $subt(S) \subseteq subt(S_{inp})$ it follows from Lemma 7.22 that S has a pure Σ_0-unifier. Soundness of the transformation rules shows that the unique ancestor RBHOUP in level 2, S''_{inp}, has a pure Σ_0-unifier. Soundness of decomposition shows that the unique ancestor RBHOUP in level 1, S'_{inp}, has a pure Σ_0-unifier. Part 2 of Theorem 6.7 shows that (S_{inp}, b_{inp}) has a unifier.

Summing up, we have seen that (S_{inp}, b_{inp}) has a unifier iff $\mathcal{T}(S_{inp}, b_{inp})$ has a leaf that is labelled with a RBHOUP of type "xy". Since $\mathcal{T}(S_{inp}, b_{inp})$ is finite and can be effectively computed the decidability result follows. $\square$

Obviously, distinct strategies for traversing the search tree $\mathcal{T}(S_{inp}, b_{inp})$ can be realized.

8. A reduction rule

For the rest of the paper, for soundness and completeness issues we consider the situation where a finite background signature Σ_0 and a bound E for the exponent of periodicity have been fixed, as selected in Steps 1 and 2 of (Order-Reduction). Before we discuss the treatment of RBHOUPs of specific types we describe the application of the rule (reduce-bv), which is derived from the projection rule from higher-order unification. It represents one possible alternative in various situations and immediately leads to a reduced μ-measure of the resulting RBHOUP. We start with a remark on the possible values of variables under Σ_0-unifiers.

Remark 8.1. Consider the value $\sigma(x)$ of a II-variable x under a pure Σ_0-unifier σ of a decomposed RBHOUP S. As always, we assume $\sigma(x)$ to be in $\beta\overline{\eta}$-normal form. Since σ is pure, the following two cases represent an exhaustive subcase analysis.

(1) x has type $\iota \to \iota$ for an appropriate ι and $\sigma(x)$ has the form $\lambda y.y$ for some first-order variable y.
(2) $\sigma(x)$ has the form $\lambda y. f(t_1, \ldots, t_k)$ for some first-order variable y where $f(t_1, \ldots, t_k)$ is a first-order term over Σ_0 with at most one occurrence of y that does not have any occurrence of another variable.

Note that the second case includes the case $\lambda x.a$ for a constant a.
The following reduction rule refers to situation (1).

Definition 8.2 (*Reduce-bv*). The input is a decomposed RBHOUP S in $\beta\overline{\eta}$-normal form together with a II-variable $x \in FV(S)$.

If the type of x is not of the form $\iota \to \iota$, then fail. Otherwise transform the RBHOUP as follows.

(a) Instantiate x by $\lambda y.y$.
(b) Beta-reduce the terms until a $\beta\overline{\eta}$-normal form is reached.
(c) Decompose the resulting RBHOUP.

Lemma 8.3. (a) *The reduction rule (reduce-bv) is sound.*
(b) *The reduction rule (reduce-bv) either fails or leads to a decomposed RBHOUP S^* in $\beta\overline{\eta}$-normal form such that $\mu_1(S^*) < \mu_1(S)$.*

Proof. (a) Let σ^* be a pure Σ_0-unifier for the output system S^*. Soundness of decomposition shows that there exists a pure Σ_0-unifier σ' for the system S' reached before decomposition. Obviously, $\sigma(x) := \lambda z.z$ and $\sigma(y) := \sigma'(y)$ for all free variables $y \neq x$ of S defines a pure Σ_0-unifier for S.

(b) Assume that the final decomposition step in Part (c) does not lead to failure. Rule (reduce-bv) strictly reduces μ_1 since x is removed. Since decomposition does not destroy $\beta\overline{\eta}$-normal forms we are done. □

Lemma 8.4 (*Weak Completeness of Reduction*). *Let S be a decomposed RBHOUP in $\beta\overline{\eta}$-normal form with a pure Σ_0-unifier σ with exponent of periodicity $\leq E$. Let $M \neq \emptyset$ be a subset of the set of II-variables in $FV(S)$. Then either*

1. *it is possible to reach via application of rule (reduce-bv) to a II-variable $x \in M$ a decomposed RBHOUP S^* in $\beta\overline{\eta}$-normal form such that $\mu(S^*) < \mu(S)$ and (S^*) has a pure Σ_0-unifier σ^* with exponent of periodicity $\leq E$, or*
2. *each value $\sigma(x)$ for $x \in M$ has the form* (2) *described in Remark* 8.1.

Proof. If there is any variable $x \in M$ where $\sigma(x)$ has the form (1) described in Remark 8.1, then we apply (reduce-bv) using x. We define $\sigma'(y) := \sigma(y)$ for $x \neq y \in FV(S)$. It is trivial to verify that σ' is a pure Σ_0-unifier for the system reached after Step (b). Clearly the exponent of periodicity of σ' does not exceed the exponent of periodicity of σ. Using completeness of decomposition we are done. In the remaining case, every value $\sigma(x)$ for $x \in M$ is of the form (2) described in Remark 8.1. The result follows. □

9. Strategy for transforming RBHOUPs — termination

Informally, the strategy for transforming RBHOUPs and the termination arguments used in the following sections can be summarized as follows:

– Decomposition rules and (reduce-bv) clearly decrease $\mu(S)$.
– Properly applied to a non-"xy"-problem, imitation:
  - decreases $\mu(S)$ if S is of type "nocycle": either a cycle is created, or the number of surface positions in S is decreased;
  - decreases $\mu(S)$ if S is of type "amb": either a cycle of length $h > 1$ is transformed into a cycle of length $h - 1$, or the main depth of the relevant context of some non-path-unique cycle of length 1 is decreased;
  - gathers the contexts of a non-special path-unique cycle into a single context, so that a problem of type "unique" is progressively transformed into a "special path-unique" problem, where only one relevant context of the path-unique cycle is not trivial.
– In the case where S contains a special path-unique cycle, either the value of some II-variable can be guessed from the maximal exponent of periodicity of a minimal unifier of S (rule solve-special-cycle), or only a prefix of this value can be guessed, but the corresponding imitation and decomposition decreases the length of the cycle.

10. Rules for type "nocycle"

Let S denote a (decomposed) RBHOUP of type "nocycle", with a set of variables $\mathcal{V}_S := FV(S)$. Let the relations "$\sim_1$" and "$>_1$" on $\mathcal{V}_S$ be defined as follows: if there exists an equation $x\ s \doteq y\ t \in S \cup S^T$, or $x\ s \doteq y \in S \cup S^T$, then $x \sim_1 y$. If there exists an equation $x\ s \doteq t \in S \cup S^T$ or $x \doteq t \in S \cup S^T$ and t has some function symbol f as head and y is on the surface of t, then $x >_1 y$.

Let "$\sim$" denote the equivalence relation in $\mathcal{V}_S$ generated by $\sim_1$. Denote the equivalence class of a variable x by $[x]_\sim$. For equivalence classes D_1, D_2 of $\mathcal{V}_S/\sim$ define $D_1 \rhd_1 D_2$ if there exist $x_i \in D_i$ for $i = 1, 2$ such that $x_1 >_1 x_2$. Let "$\rhd$" denote the transitive closure of "$\rhd_1$".

Lemma 10.1. *If the decomposed RBHOUP S is of type "nocycle", then the relation "$\rhd$" is an irreflexive partial order on $\mathcal{V}_S/\sim$.*

Proof. Assume that "$\rhd$" is not irreflexive. Then there exists a sequence $r_1 \doteq t_1, \ldots, r_h \doteq t_h$ where $r_j = x_j$ or $r_j = x_j\ s_j$ for $j = 1, \ldots, h$ of length $h \geq 1$ of equations from $S \cup S^T$ such that x_i occurs on the surface of $t_{i-1 \ \mathtt{mod*}\ h}$ for $1 \leq i \leq h$. Moreover, there is at least one term t_i of the form $f(t_{i,1}, \ldots, t_{i,n})$. Since the sequence does not represent a cycle, all x_i have arity 0. But then the decomposition rule (occurs-check) would lead to failure, a contradiction. $\square$

Definition 10.2 (*Imitation*). Let S be a decomposed RBHOUP in $\beta\overline{\eta}$-normal form of type "nocycle". Select a $\rhd$-maximal $\sim$-equivalence class D and a function symbol f according to the following conditions: there must be an equation $z\ \cdots \doteq f \cdots$ in $S \cup S^T$ where $z \in D$ and z is a I-variable or a II-variable. Let $k := ar(f)$. Select one of the following three alternatives.

The first alternative is only possible if there is some II-variable in D. The second alternative is only possible if $k = 0$ and $f \in \Sigma_0$. The third alternative is only possible if $f \in \Sigma_0$ and f has arity $k \geq 1$ and if all arguments of f have elementary type, i.e. f is a first-order function symbol.
Fail, if no alternative can be selected.

1. Apply (reduce-bv) using a II-variable $x \in D$.
2. Apply the following steps:
   (a) For every II-variable $x \in D$, instantiate x by $\lambda y. f$. For every I-variable $x \in D$ instantiate x by f.
   (b) Use β-reduction to transform the terms into $\beta\overline{\eta}$-normal form.
   (c) Decompose the resulting RBHOUP.
3. Apply the following steps:
   (a) For every II-variable $x \in D$ select an index j_x with $1 \leq j_x \leq k$. Instantiate x by

   $$\lambda y. f(z_1, \ldots, z_{j_x-1}, (x'\ y), z_{j_x+1} \ldots, z_k)$$

   where the $z_i, i = 1, \ldots k, i \neq j_x$ are fresh I-variables and x' is a new II-variable of appropriate type.
   For every I-variable $x \in D$ instantiate x by

   $$f(z'_1, \ldots, z'_k)$$

   where the $z'_i, i = 1, \ldots k$ are fresh I-variables.
   (b) Use β-reduction to transform the terms into $\beta\overline{\eta}$-normal form.
   (c) Decompose the resulting RBHOUP.

Lemma 10.3. *Application of the rule (imitation) to a decomposed RBHOUP S of type "nocycle" either fails or results in a RBHOUP S^* such that $\mu(S^*) < \mu(S)$.*

Proof. A $\rhd$-maximal equivalence class with the required properties exists, since there are no cycles, there is no occurs-check failure, and the RBHOUP is decomposed and not of type "xy".
If alternative 1 is selected, then the result follows from Lemma 8.3.
If alternative 2 is selected, then no component of μ is increased, and μ_1 is strictly decreased, if there is a II-variable in D; otherwise μ_5 is strictly decreased.
Assume that alternative 3 is selected. Since II-variables x' occurring in S^* correspond to II-variables x occurring in S it follows that μ_1 is not modified. If S^* contains a cycle, then μ_2 is reduced. Hence we may assume that S^* does not have a cycle, which means that μ_2 is not modified. Furthermore, none of the systems reached before or during Step (c) has a cycle, by Corollary 7.17. Hence rule (decomp-repvt) is not applied. Since the lambda-binders introduced by the instantiations of II-variables are removed by the β-reduction in Step (b), and since decomposition cannot introduce new λ-bound variables, μ_3 is not modified. Note that all surface occurrences of variables $x \in D$ are the distinguished occurrences in equations $x\,r \doteq y\,s, x\,r \doteq y, x \doteq y\,s, x\,r \doteq f\,\overrightarrow{s}, x \doteq f\,\overrightarrow{s}$, or in symmetric versions. We consider the modifications of μ_4 that result from the treatment of each type of equation.

We first consider the equations of the form $x\ r \doteq y\ s$ in $S \cup S^T$. Here $x \in D$ implies $y \in D$. The instantiation of x, y, after β-reduction, yields equations

$$f(z_1, \ldots, x'\ r', \ldots, z_n) \doteq f(z'_1, \ldots, y'\ s', \ldots, z'_n).$$

Via decomposition both occurrences of f are removed. The decomposition of the successor equations only uses (repvv) since (decomp-repvt) is not applied. We do not obtain new function symbols on the surface. Replacements of variable occurrences in $x\ r \doteq y\ s$ that are not on the surface do not modify μ_4. Hence, after Step (c), we do not have any new contribution to measure component μ_4 from equations $x\ r \doteq y\ s$.

The same holds for the equations $x\ r \doteq y$, $x \doteq y\ s$ in $S \cup S^T$.

For equations $x\ r \doteq f\ \overrightarrow{s}$ in S the instantiation of x, after β-reduction, yields

$$f(z_1, \ldots, x'\ r', \ldots, z_n) \doteq f(s'_1, \ldots, s'_{i_0}, \ldots, s'_n).$$

Via decomposition both occurrences of f are removed. The rest is as above. Hence, if there is an equation $x\ r \doteq f\ \overrightarrow{s}$ with $x \in D$, then after Step (c), at least one surface occurrence of f is removed. The same holds for the equations $x \doteq f\ \overrightarrow{s}$ in S where $x \in D$.

Since there is at least one equation of the form $x \ldots \doteq f\ \overrightarrow{s}$ with $x \in D$ the measure μ_4 is strictly decreased. □

Lemma 10.4. *The rule (Imitation) is sound and complete.*

Proof. Let S be a RBHOUP before application of the rule.
Soundness. Assume there is a pure Σ_0-unifier σ^* of the RBHOUP S^* reached after the transformation. If alternative 1 is selected, then Lemma 8.3 shows that S has a pure Σ_0-unifier.

If alternative 2 is selected, soundness of decomposition shows that there exists a pure Σ_0-unifier σ' of the RBHOUP S' reached before the final decomposition. We now show that there is a pure Σ_0-unifier for S. Since the variables in D do not occur in S', we can define $\sigma(x) := \lambda y. f$ for every II-variable $x \in D$, and $\sigma(x) := f$ for every I-variable $x \in D$, and $\sigma(x) := \sigma'(x)$ otherwise. It is easy to see that σ is a pure Σ_0-unifier of S.

If alternative 3 is used, soundness of decomposition shows that there exists a pure Σ_0-unifier σ' of the RBHOUP S' reached before the final decomposition. We now show that there is a pure Σ_0-unifier for S.

For each II-variable $x \in D$ define $\sigma(x)$ as the $\beta\overline{\eta}$-normal form of the σ'-image of $\lambda y. f(z_1, \ldots, z_{j_x-1}, (x'\ y), z_{j_x+1} \ldots, z_k)$. For each I-variable $x \in D$ define $\sigma(x)$ as $\sigma'(f(z'_1, \ldots, z'_k))$. It is trivially seen that σ is a pure Σ_0-unifier for S. Since for $i \neq j_x$, $\sigma'(z_i)$ is always a ground first-order term and since all values $\sigma'(z'_i)$ are ground first-order terms (cf. Lemma 4.3) it follows that σ is a pure unifier. It follows that σ is a pure Σ_0-unifier for S.

Completeness. Let σ be a pure Σ_0-unifier of S. We use Lemma 8.4: first we treat the case where each II-variable $x \in D$ has a value $\sigma(x)$ of the form $\lambda y. f(t_1, \ldots, t_k)$ where $f \in \Sigma_0$, y is a first-order variable, and $f(t_1, \ldots, t_k)$ is a first-order term over Σ_0 with at most one occurrence of y that does not have any occurrence of another variable. Obviously f must be the function symbol mentioned in the rule (Imitation). We use alternative 2, if $ar(f) = 0$. For further arguments, see the proof below for alternative 3.

If $ar(f) \geq 1$, we use alternative 3. Let $x \in D$ and let $\sigma(x) = \lambda y. f(t_1, \ldots, t_n)$. In Step (a), if there exists a term t_i with an occurrence of y, then select $j_x := i$; in the other case the selection of j_x is arbitrary. We define $\sigma'(x') := \lambda y.t_i$ and $\sigma'(x) := \sigma(y)$ for $x \neq y$. Images of fresh first-order variables are obvious. The definition respects the kind of the variables.

For I-variables $x \in D$ it is sufficient to treat the case where $\sigma(x)$ is a ground first-order term. This is analogous to the case where x is a II-variable.

It is trivial to verify that σ' is a pure Σ_0-unifier for the system reached after Step (b). The exponent of periodicity of σ' does not exceed the exponent of periodicity of σ. Since decomposition is complete we are done. □

Lemma 10.5. *The rule (Imitation) either fails or transforms a decomposed RBHOUP S in $\beta\overline{\eta}$-normal form into a decomposed RBHOUP S' in $\beta\overline{\eta}$-normal form. Moreover, $subt(S') \subseteq subt(S)$.*

Proof. By inspecting the rule. □

11. Rules for type “amb”

Before we describe the treatment of RBHOUPs of type “amb” we introduce a rule that is used to replace surface occurrences of first-order variables by a term t, if the equation $x \doteq t$ is in the problem set. It is not used for decomposition, since for RBHOUPs of type “nocycle” it would in general increase the measure μ.

(repvt) $\dfrac{\{s_1 \doteq s_2\} \cup S_0}{\{s_1 \doteq s_2\} \cup S_1}$ $s_1 \doteq s_2$ is $x \doteq t$ or $t \doteq x$, where x is a first-order variable. The rule (occurs-check) must not be applicable. Then S_1 is constructed from S_0 by replacing all surface occurrences of x by t.

Now let S denote a problem of type “amb”. Recall that S is decomposed and has a ψ-minimal cycle L that is compressed and non-path-unique. We may assume that L has the form $x_1\ s_1 \ \doteq t_1, \ldots, x_h\ s_h \ \doteq t_h$. The cycle could as well be represented as $x_1\ s_1 \doteq C_1[t'_1], \ldots, x_h\ s_h \doteq C_h[t'_h]$, where C_i are the relevant contexts (see Definition 7.4). Note that all x_i are II-variables ($1 \leq i \leq h$).

Definition 11.1 (*Solve-ambiguous-cycle*). The input is the decomposed RBHOUP S in $\beta\overline{\eta}$-normal form of type “amb” with a ψ-minimal cycle L as described above. Select one of the following two alternatives.

1. Apply (reduce-bv) using a variable $x \in \{x_1, \ldots, x_h\}$.
2. Select an index j such that $x_j\ s_j \doteq t_j$ is an equation in L where $x_{(j+1 \ \mathtt{mod*}\ h)}$ occurs at least twice on the surface of $t_j = f(t_{j,1}, \ldots, t_{j,k})$ and the main depth of the relevant context C_j is minimal in L. If $f \notin \Sigma_0$ or f is not first-order, then fail.
   Now apply the following steps:
   (a) Select an index $r \in \{1, \ldots, k\}$. In the special situation where $h = 1$, the selection of r is subject to the following condition: all surface occurrences of x_1

in $f(t_{1,1}, \ldots, t_{1,k})$ have to be in $t_{1,r}$. If this is not possible since C_1 is trivial, then stop with fail.

(b) Instantiate x_j by $\lambda y. f(z_1, \ldots, z_{r-1}, x'_j\ y, z_{r+1}\ \ldots\ z_k)$ where the z_i are fresh I-variables ($1 \leq i \leq k, i \neq r$), and x'_j is a fresh II-variable.

(d) Use β-reduction until a $\beta\overline{\eta}$-normal form is reached for every term in the system.

(e) Apply rule (decomp) to the equation that is obtained from the equation $x_j\ s_j \doteq t_j$ in Step (d).

(f) Apply (repvt) for all the new equations $z_i \doteq t'_{j,i}$ ($1 \leq i \leq k, i \neq r$) that are obtained from the previous step.

(g) Then decompose the resulting RBHOUP.

Lemma 11.2. *Application of the rule (solve-ambiguous-cycle) to a RBHOUP S of type "amb" either fails or results in a RBHOUP S^*, such that $\mu(S^*) < \mu(S)$.*

Proof. If alternative 1 is selected, then the result follows from Lemma 8.3.

Assume that alternative 2 is selected. By Lemma 7.16 it suffices to show that the system S' reached after Step (f) satisfies $\mu(S') < \mu(S)$. Obviously Steps (a)–(f) do not affect the measure μ_1. Note that in (b) both x'_j and x_j are II-variables. We now show that μ_2 is reduced.

We first assume that $h > 1$ and consider the relevant equation, its predecessor and successor equation (for $h = 2$, the first and the third equation are identical).

$$\begin{aligned} x_{j-1}\ q &\doteq t_{j-1}\lfloor x_j \rfloor \\ x_j\ r &\doteq f(t_{j,1}, \ldots, t_{j,k})\lfloor x_{j+1} \rfloor \\ x_{j+1}\ s &\doteq t_{j+1} \end{aligned}$$

Instantiating x_j plus beta-reductions yields the equations

$$\begin{aligned} x_{j-1}\ q' &\doteq t'_{j-1}\lfloor f(z_1, \ldots, z_{r-1}, x'_j\ p, z_{r+1}\ \ldots\ z_k) \rfloor \\ f(z_1, \ldots, z_{r-1}, x'_j\ r', z_{r+1}\ \ldots\ z_k) &\doteq f(t'_{j,1}, \ldots, t'_{j,k})\lfloor x_{j+1} \rfloor \\ x_{j+1}\ s' &\doteq t'_{j+1}. \end{aligned}$$

Here primed terms are obtained from unprimed predecessors via instantiation. The arguments represented as "p" in a uniform manner depend on the arguments of the respective surface occurrences of x_j in t_{j-1}. Decomposition of the central equation in Step (e) yields

$$\begin{aligned} x_{j-1}\ q' &\doteq t'_{j-1}\lfloor f(z_1, \ldots, z_{r-1}, x'_j\ p, z_{r+1}\ \ldots\ z_k) \rfloor \\ x'_j\ r' &\doteq t'_{j,r} \\ x_{j+1}\ s' &\doteq t'_{j+1}. \end{aligned}$$

plus the equations $z_i \doteq t'_{j,i}$ ($i \neq r$). Replacing surface occurrences of the z_i ($i \neq r$) by $t'_{j,i}$ in Step (f) now yields

$$\begin{aligned} x_{j-1}\ q' &\doteq t''_{j-1}\lfloor f(t'_{j,1}, \ldots, t'_{j,r-1}, x'_j\ p, t'_{j,r+1}\ \ldots\ t'_{j,k}) \rfloor \\ x'_j\ r' &\doteq t''_{j,r} \\ x_{j+1}\ s' &\doteq t''_{j+1}. \end{aligned}$$

First assume that there exists at least one index $l \neq r$ such that $t_{j,l}$ has a surface occurrence of x_{j+1}. Since the cycle has minimal length and $h > 1$ we have $x_{j+1} \neq x_j$. It follows that $t'_{j,l}$ has a surface occurrence of x_{j+1}. This shows that the equations

$$\begin{aligned} x_{j-1}\, q' &\doteq t''_{j-1}\lfloor f(t'_{j,1}, \ldots, t'_{j,r-1}, x'_j\ p, t'_{j,r+1}\ \ldots\ t'_{j,k})\rfloor \\ x_{j+1}\, s' &\doteq t''_{j+1} \end{aligned}$$

together with the images of the remaining equations of L represent a cycle of length $h - 1$. Note that the conditions for a cycle are satisfied since $t''_{j-1}\lfloor f(t'_{j,1}, \ldots, t'_{j,r-1}, x'_j\ p, t'_{j,r+1}\ \ldots\ t'_{j,k})\rfloor$ contains a function symbol as head. Hence, after Steps (a)–(f) we reach a system with smaller ψ_1-measure.
Now assume that all surface occurrences of x_{j+1} belong to $t_{j,r}$. This means that x_{j+1} occurs at least twice on the surface of $t'_{j,r}$. Together with the images of the remaining equations of L, the equations

$$\begin{aligned} x_{j-1}\, q' &\doteq t''_{j-1}\lfloor f(t'_{j,1}, \ldots, t'_{j,r-1}, x'_j\ p, t'_{j,r+1}\ \ldots\ t'_{j,k})\rfloor \\ x'_j\, r' &\doteq t''_{j,r}\lfloor x_{j+1}\rfloor \\ x_{j+1}\, s' &\doteq t''_{j+1} \end{aligned}$$

represent a cycle of the system reached after Step (d). The main depth of the relevant context of the equation with index j is decreased. The new cycle is non-path-unique, and hence it has smaller ψ-measure than L. Hence after Steps (a)–(d) we reach a system with smaller ψ-measure.

It remains to consider the case $h = 1$. Let the equation be

$$x_1\, r \doteq f(t_1, \ldots, t_m)\lfloor x_1\rfloor.$$

Here x_1 has at least two surface occurrences in t_r. Instantiation and beta-reductions yield

$$\begin{aligned} & f(z_1, \ldots, z_{r-1}, (x'_1\ r'), z_{r+1}\ \ldots\ z_k) \\ \doteq\ & f(t'_1, \ldots, t'_{r-1}, t'_r\lfloor f(z_1, \ldots, z_{r-1}, (x'_1\ p), z_{r+1}\ \ldots\ z_m)\rfloor, t'_{r+1}, \ldots, t'_m). \end{aligned}$$

Decomposition gives

$$x'_1\, r' \doteq t'_r\lfloor f(z_1, \ldots, z_{r-1}, (x'_1\ p), z_{r+1}\ \ldots\ z_k)\rfloor.$$

We have a non-path-unique cycle where the depth of the main context is strictly smaller than before. As above it follows that we reach a system with smaller ψ-measure. □

Fig. 3 illustrates the first situation considered in the preceding proof where $h = 3$ and $j = 2$.

Lemma 11.3. *The rule (solve-ambiguous-cycle) is sound and complete.*

Proof. Let S be a RBHOUP before application of the rule.
Soundness. Assume there is a pure Σ_0-unifier σ^* of the RBHOUP S^* reached after the transformation. If alternative 1 is selected, then Lemma 8.3 shows that S has a pure Σ_0-unifier.

If alternative 2 is used, soundness of decomposition shows that there exists a pure Σ_0-unifier σ' of the RBHOUP S' reached before the final decomposition. Obviously,

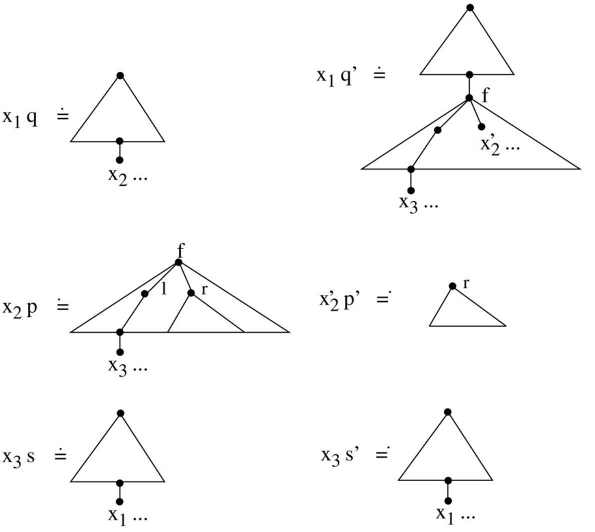

Fig. 3. Illustration for the proof of Lemma 11.2.

if $t = \lambda y.\, f(z_1, \ldots, z_{r-1}, x'_j\ y, z_{r+1}\ \ldots\ z_k)$ denotes the substitute for x_j as defined in Step (b) of alternative 2, then $\sigma(x_j) := \sigma'(t){\downarrow_{\beta\overline{\eta}}}$ and $\sigma(y) := \sigma'(y)$ for all free variables $y \neq x_j$ of S defines a Σ_0-unifier for S. To prove soundness it remains to show that σ is a pure unifier. We have

$$\sigma(x_j) = \lambda y.\, f(\sigma'(z_1), \ldots, \sigma'(z_{r-1}), (\sigma'(x'_j)\ y){\downarrow_{\beta\overline{\eta}}}, \sigma'(z_{r+1})\ \ldots\ \sigma'(z_k))$$

where $\sigma'(z_i)$ is a ground first-order term for $i \in \{1, \ldots, k\}, i \neq r$. The expression $(\sigma'(x'_j)\ y){\downarrow_{\beta\overline{\eta}}}$ is either a first-order ground term or of the form $A[y]$ where A is a first-order ground context. Hence σ is a pure Σ_0-unifier.

Completeness. Let σ be a pure Σ_0-unifier of S. It follows from Lemma 8.4 that it suffices to consider the case where each variable $x \in \{x_1, \ldots, x_h\}$ has a value $\sigma(x)$ of the form $\lambda y.\, f(t_1, \ldots, t_k)$ where y is a first-order variable and $f(t_1, \ldots, t_k)$ is a first-order term over Σ_0 with at most one occurrence of y which does not have any other occurrence of a variable. Here we use alternative 2. We select an index j such that $x_j\ s_j \doteq t_j$ is an equation in L where $x_{(j+1 \bmod * h)}$ occurs at least twice on the surface of $t_j = f(t_{j,1}, \ldots, t_{j,k})$ and the main depth of the relevant context C_j is minimal in L. Let $\sigma(x_j) = \lambda y.\, f(q_1, \ldots, q_k)$; let r be an index such that q_r has an occurrence of y. If $f(q_1, \ldots, q_k)$ is a ground term, then r is any index with $1 \leq r \leq k$. We select this index in Step (a).

We have to show that for $h = 1$ all surface occurrences of x_1 in $f(t_{1,1}, \ldots, t_{1,k})$ are in $t_{1,r}$. For index $i \neq r$, the subterms q_i of $\sigma(x_1) = \lambda y.\, f(q_1, \ldots, q_k)$ are ground first-order terms. Hence we have $\sigma(x_1\ s_1) = f(q_1, \ldots, q_{r-1}, q'_r, q_{r+1}, \ldots, q_k) = f(\sigma(t_{1,1}), \ldots, \sigma(t_{1,k}))$. Assume that for some $i \neq r$ the variable x_1 occurs on the surface of $t_{1,i}$. Since q_i is a proper subterm of $\sigma(x_1)$ and $\sigma(x_1)$ is a subterm of $\sigma(t_{1,i}) = q_i$ we obtain a contradiction.

Returning to the general case with j, r as above it is simple to see that $\sigma'(x'_j) := \lambda y.q_r$ and $\sigma'(z_i) := q_i$ for $i \neq r$, $1 \leq i \leq k$ defines a pure Σ_0-unifier for S'. It obviously has an exponent of periodicity that does not exceed the exponent of periodicity of σ. From completeness of decomposition rules it follows that we are done. $\square$

Lemma 11.4. *The rule (solve-ambiguous-cycle) either fails or transforms a decomposed RBHOUP S in $\beta\overline{\eta}$-normal form into a decomposed RBHOUP S' in $\beta\overline{\eta}$-normal form. Moreover, $subt(S') \subseteq subt(S)$.*

Proof. By inspecting the rule. $\square$

12. Rules for type "unique"

In this section we describe the rules for transforming RBHOUPs in $\beta\overline{\eta}$-normal form of type "unique", where the focus is on transforming path-unique cycles. We will first introduce a special form of path-unique cycles.

12.1. Special path-unique cycles

Recall that in RBHOUPs of type "unique" every minimal-length cycle is path-unique. Every path-unique cycle can be represented as

$$x_1\ s_1 \doteq C_1[x_2\ t_1], \ldots, x_h\ s_h \doteq C_h[x_1\ t_h].$$

Definition 12.1. Let S be a RBHOUP. A term t occurring in S is *b-constrained* iff either $t \in FV(S)$ is a I-variable or there exists an equation $z \doteq t$ in S where z is a I-variable. A context $f(t_1, \ldots, t_{r-1}, [\cdot], t_{r+1}, \ldots, t_k)$ appearing in S is called *b-constrained* iff f is a first-order function symbol with $f \in \Sigma_0$, and every subterm t_i ($i = 1, \ldots, r-1, r+1, \ldots, k$) is b-constrained. A non-empty context C is *b-constrained* iff every subcontext of main depth 1 is b-constrained.

Definition 12.2. A cycle L of length h is *special path-unique* if the following hold:

– it is path-unique,
– only the relevant context C_h of the last equation is non-trivial,
– the context C_h is b-constrained.

Lemma 12.3. *Let S be an RBHOUP and σ be a pure Σ_0-unifier of S. Then*

1. *$\sigma(t)$ is a first-order Σ_0-term for every b-constrained term t occurring in S.*
2. *$\sigma(C)$ is a first-order Σ_0-context for every b-constrained context C occurring in S.*

Proof. Obvious.

Lemma 12.4. *If a cycle L is* special path-unique*, then the type of the relevant context C_h is the same as the type of the hole of C_h.*

Proof. This follows by checking all the target types of top-level terms in the cycle, and of the expression in the hole, since the last equation is $x_h\ s_h \doteq C_h[x_1\ t_h]$. The observation is that by special path-uniqueness all these (elementary) target types of terms in equations of L are equal. □

12.2. *From unique cycles to special path-unique cycles*

The following rules (shuffle), (shuffle*) and (shuffle**) are intended to transform the RBHOUP into one with a minimal-length special path-unique cycle.

The rule (shuffle) does not necessarily decrease μ; it may increase the ψ_3-component of the measure component μ_2. It can be applied to a RBHOUP S of type "unique" (which must have a minimal-length path-unique cycle L) only in the context of a larger procedure where we will be able to decrease μ eventually.

The two alternatives of (shuffle) can be considered as special forms of projection and imitation in the terminology of higher-order unification.

Definition 12.5 (*Sub-rule (shuffle)*). The input is a decomposed RBHOUP S of type "unique" in $\beta\overline{\eta}$-normal form, a minimal-length (path-unique) cycle L of length h in S and an index j with $1 \leq j \leq h$.

Let L have the form $x_1\ s_1 \doteq C_1[x_2\ t_1], \ldots, x_h\ s_h \doteq C_h[x_1\ t_h]$ where the relevant context C_j is non-trivial. Here each variable x_i $(1 \leq i \leq h)$ is a II-variable, since L is compressed.

Select one of the following alternatives.

1. Apply (reduce-bv) using a II-variable $x \in \{x_1, \ldots, x_h\}$.
2. Let $C_j[x_{j+1}\ t_j]$ have the form $f(t_{j,1}, \ldots, t_{j,s-1}, t_{j,s}\lfloor x_{j+1} \rfloor, t_{j,s+1}, \ldots, t_{j,k})$, where $k = ar(f) \geq 1$.
   (a) Fail, if f is not a first-order function symbol, or if $f \notin \Sigma_0$.
   Select an index $1 \leq r \leq k$. In the special situation where $h = 1, r = s$ is the only permitted selection. (Note that for $h = 1$ all surface occurrences of x_1 in C_1 are in $t_{1,r} = t_{1,s}$ since L is path-unique.)
   (b) Instantiate x_j by $\lambda y. f(z_1, \ldots, z_{r-1}, x'_j\ y, z_{r+1}, \ldots, z_k)$, where the z_i $(1 \leq i \leq k, i \neq r)$ are fresh I-variables and x'_j is a fresh II-variable.
   (c) Use β-reduction to reach a $\beta\overline{\eta}$-normal form.
   (d) Apply (decomp) to the jth equation of L after the instantiation, i.e. to
   $$f(z_1, \ldots, z_{r-1}, x'_j\ s'_j, z_{r+1}, \ldots, z_k) \doteq f(t'_{j,1}, \ldots, t'_{j,s-1}, t'_{j,s}\lfloor x_{j+1} \rfloor, t'_{j,s+1}, \ldots, t'_{j,k}).$$
   (e) Apply (repvt) or (repvv) for all equations $z_i \doteq t'_{j,i}$ for $i \neq r$ added by the last step.
   (f) Then decompose the resulting RBHOUP.

Lemma 12.6. *The rule (shuffle) is sound and complete.*

Proof. The reader should note that the procedure in alternative 2 is almost the same as in alternative 2 of the rule (solve-ambiguous-cycle), modulo the irrelevant origin of the variable x_j. For (shuffle), the situation $h = 1$ is also the same as in the rule (solve-ambiguous-cycle). Hence soundness and completeness of (shuffle) can be shown exactly as in the proof of Lemma 11.3. □

In the following, let $\overline{\mu}$ be the measure with the components μ_1 and $\mu'_2 = min\{(\psi_1(L), \psi_2(L)) \mid L$ is a cycle in $S\}$. Note that $\overline{\mu}(S') < \overline{\mu}(S)$ implies that $\mu(S') < \mu(S)$, for arbitrary S and S'.

Lemma 12.7. *Let S be a decomposed RBHOUP of type "unique" in $\beta\overline{\eta}$-normal form with a minimal-length (path-unique) cycle L. Let S' be obtained from S by an application of (shuffle) using L. Then S' is decomposed, in $\beta\overline{\eta}$-normal form, and one of the following cases holds:*

1. $\overline{\mu}(S') < \overline{\mu}(S)$.
2. $\overline{\mu}(S') = \overline{\mu}(S)$, $h \geq 2$, *alternative 2 is selected,* $r = s$, *the system S' is decomposed and contains a (ψ_1, ψ_2)-minimal path-unique cycle L' of length h such that the relevant contexts $C'_1, \ldots, C'_h$ have main depth corresponding to $C_1, \ldots, C_h$ except for indices j and $j - 1 \mathtt{\ mod}{*}\ h$. We have $|C'_{j-1 \mathtt{\ mod}* \ h}| = |C_{j-1 \mathtt{\ mod}* \ h}| + 1$ and $|C'_j| = |C_j| - 1$. In $C'_{j-1 \mathtt{\ mod}* \ h}$, the suffix subcontext of main depth* 1 *is b-constrained.*
3. $\overline{\mu}(S') = \overline{\mu}(S)$, $h = 1$, *alternative 2 is selected,* $r = s$, *the system (S') is decomposed and contains a (ψ_1, ψ_2)-minimal path-unique cycle L' of length $h = 1$ such that for the relevant context C'_1, the equation $|C'_1| = |C_1|$ holds. In C'_1, the suffix subcontext of main depth* 1 *is b-constrained.*

Proof. If Selection 1 is used, then $\overline{\mu}(S') < \overline{\mu}(S)$ (cf. Lemma 8.3).

Assume that selection 2 is used. Then μ_1 is not affected since both x_j, x'_j are II-variables. First consider the case where $r \neq s$. In this case, we have $h \geq 2$. We have to show that the minimal (ψ_1, ψ_2)-measure of a cycle is decreased. Let the equations with indices $j - 1 \mathtt{\ mod}* \ h$, j and $j + 1 \mathtt{\ mod}* \ h$ be (in the case $h = 2$ the first and third equation are identical)

$$\begin{aligned} x_{j-1}\, r_{j-1} &\doteq C_{j-1}[x_j\ t_{j-1}] \\ x_j\, r_j &\doteq f(t_{j,1}, \ldots, t_{j,s-1}, t_{j,s}\lfloor x_{j+1}\rfloor, t_{j,s+1}, \ldots, t_{j,k}) \\ x_{j+1}\, r_{j+1} &\doteq t. \end{aligned}$$

Path uniqueness and length-minimality imply that L contains only the explicitly indicated surface occurrences of x_j. Instantiation (b) and β-reductions give successor equations of the form

$$\begin{aligned} x_{j-1}\, r'_{j-1} &\doteq C'_{j-1}[f(z_1, \ldots, z_{r-1}, (x'_j\ t'_{j-1}), z_{r+1}, \ldots, z_k)] \\ f(z_1, \ldots, z_{r-1}, (x'_j\ r'_j), z_{r+1}, \ldots, z_k) & \\ &\doteq f(t'_{j,1}, \ldots, t'_{j,s-1}, t'_{j,s}\lfloor x_{j+1}\rfloor, t'_{j,s+1}, \ldots, t'_{j,k}) \\ x_{j+1}\, r'_{j+1} &\doteq t'. \end{aligned}$$

Applying (decomp) to the middle equation yields (among others) the equation $z_s \doteq t'_{j,s}\lfloor x_{j+1}\rfloor$. Applying (repvt) or (repvv) we obtain a cycle L_1 of length $h - 1$:

$$\begin{aligned} x_{j-1}\, r'_{j-1} &\doteq C'_{j-1}[f(z_1, \ldots, z_{r-1}, (x'_j\ t'_{j-1}), z_{r+1}, \ldots, z_{s-1}, t'_{j,s}\lfloor x_{j+1}\rfloor, \\ &\quad z_{s+1}, \ldots, z_k)] \\ x_{j+1}\, r'_{j+1} &\doteq t'. \end{aligned}$$

After decomposition we have a successor cycle L' of length $h-1$ in the system S' that is reached (cf. Corollary 7.17). Hence $\overline{\mu}(S') < \overline{\mu}(S)$.

It remains to consider the case where $r = s$. Now $h = 1$ is also a possibility. First let $h \geq 2$. If $\overline{\mu}(S') < \overline{\mu}(S)$ we are done. Assume that $\overline{\mu}(S') \geq \overline{\mu}(S)$. Recall that μ_1 is not affected. Since $r = s$, instantiation (c) and β-reductions lead to

$$\begin{array}{rcl} x_{j-1}\, r'_{j-1} & \doteq & C'_{j-1}[f(z_1,\ldots,z_{r-1},(x'_j\; t'_{j-1}), z_{r+1},\ldots,z_k)] \\ \multicolumn{3}{l}{f(z_1,\ldots,z_{r-1},(x'_j\; r'_j), z_{r+1},\ldots,z_k)} \\ & \doteq & f(t'_{j,1},\ldots,t'_{j,r-1},t'_{j,r}\lfloor x_{j+1}\rfloor, t'_{j,r+1},\ldots,t'_{j,k}) \\ x_{j+1}\, r'_{j+1} & \doteq & t'. \end{array}$$

Applying (decomp) to the middle equation yields a variant L_1 of L of length h with equations

$$\begin{array}{rl} x_{j-1}\, r'_{j-1} \doteq & C'_{j-1}[f(z_1,\ldots,z_{r-1},(x'_j\; t'_{j-1}), z_{r+1},\ldots,z_k)] \\ x'_j\; r'_j \doteq & t'_{j,r}\lfloor x_{j+1}\rfloor \\ x_{j+1}\, r'_{j+1} \doteq & t'. \end{array}$$

We keep this as an intermediate result and consider the case $r = s$ and $h = 1$. Here the relevant equation has the form

$$x_1\, r_1 \doteq C_1[x_1\; t_1] = f(t_{1,1},\ldots,t_{1,r-1},t_{1,r}\lfloor x_1\; t_1\rfloor, t_{1,r+1},\ldots,t_{1,k}).$$

Instantiation (c) and β-reductions lead to

$$\begin{array}{l} \quad f(z_1,\ldots,z_{r-1},(x'_1\; r'_1), z_{r+1},\ldots,z_k) \\ \doteq f(t'_{1,1},\ldots,t'_{1,r-1},t'_{1,r}\lfloor f(z_1,\ldots,z_{r-1},(x'_1\; t'_1), z_{r+1},\ldots,z_k)\rfloor, t'_{1,r+1},\ldots,t'_{1,k}) \end{array}$$

Applying (decomp) yields a variant L_1 of L of length 1 with the equation

$$(x'_1\; r'_1) \doteq t'_{1,r}\lfloor f(z_1,\ldots,z_{r-1},(x'_1\; t'_1), z_{r+1},\ldots,z_k)\rfloor.$$

Let S' be the system reached after decomposition.

Now we can treat the case $h = 1$ and $h > 1$ together:

As we have seen in Corollary 7.17, S' contains a cycle L' of length h corresponding to L_1. Since $x_1,\ldots,x_h$ are II-variables, possible applications of decomposition rules (repvv) and (decomp-repvt) do not affect the length h of L'. Since by assumption $\overline{\mu}(S') \geq \overline{\mu}(S)$, the new system S' cannot have any cycle L'' such that $(\psi_1,\psi_2)(L'') < (\psi_1,\psi_2)(L)$. It follows that L' is again path-unique and (ψ_1,ψ_2)-minimal. Hence $\overline{\mu}(S') = \overline{\mu}(S)$. Obviously L' has the properties demanded in Situations 2, 3 above. Note that $f(z_1,\ldots,z_{r-1},[\cdot],z_{r+1},\ldots,z_k)$ is b-constrained. Decomposition does not affect this property. Hence the result follows. $\square$

We now introduce two complex procedures that use (shuffle) in an iterated manner. The first one, (shuffle*), is intended to operate on a minimal-length path-unique cycle L of length $h > 1$ with several non-trivial relevant contexts. The applications of (shuffle) shuffle subcontexts of main depth 1 of a non-trivial relevant context to another index, until two non-trivial relevant contexts are merged into one. Continuing in this way we eventually reach a cycle with one non-trivial relevant context only. The rule (shuffle**) operates in the situation where exactly one non-trivial relevant context in the focused cycle is left. This rule may also be used for a cycle of length $h = 1$. The operation shuffles the non-trivial context to the next index, or in the case $h = 1$ cyclically permutes the context C_1. The intention is to "clean" the context C_h, such that afterwards each subcontext of the relevant context is b-constrained.

Definition 12.8 (*Shuffle**). Let (S) be a decomposed RBHOUP of type "unique' in $\beta\overline{\eta}$-normal form and let L be a minimal-length path-unique cycle of length $h \geq 2$ with at least two non-trivial relevant contexts C_j and $C_{j'}$. Let $S = S_0$. Iterate (shuffle) as follows:

1. First select an index j in the cycle L such that C_j is non-trivial.
2. Apply (shuffle) for index j, yielding the RBHOUP S'.
3. If $\mu(S') < \mu(S_0)$, then return S'.
4. Otherwise, let L' be the path-unique minimal-length cycle obtained from L. If C'_j is nontrivial, then go to 2 using the same index.
   If C'_j is trivial, but still L' has at least two non-trivial relevant contexts, then go to 2, replacing S by S' and using the index $j - 1 \ \mathtt{mod}{*}\ h$ instead of j. Otherwise return S'.

Definition 12.9 (*Shuffle***). Let S be a decomposed RBHOUP of type "unique" in $\beta\overline{\eta}$-normal form, let L be a path-unique cycle of S of length h that contains exactly one non-trivial relevant context, say C_h. If L is not special path-unique, then iterate (shuffle) as follows:

1. Apply (shuffle) for index h, yielding the RBHOUP S'.
2. If $\mu(S') < \mu(S)$, then return S'.
3. Otherwise, let L' be the cycle obtained from L.
   If L' is not special path-unique, then go to 1 using the same index and the cycle L'.
   If L' is special path-unique, then return S'.

Lemma 12.10. *Given a decomposed RBHOUP S in $\beta\overline{\eta}$-normal form of type "unique" with a minimal-length cycle L that is not special path-unique, it is possible using (shuffle*) and (shuffle**) to either reach failure or a decomposed RBHOUP S' in $\beta\overline{\eta}$-normal form with $\mu(S') < \mu(S)$, or a decomposed RBHOUP S' in $\beta\overline{\eta}$-normal form of type "unique" with a minimal-length and special path-unique cycle L' where $\overline{\mu}(S') = \overline{\mu}(S)$. In both cases we have $subt(S') \subseteq subt(S)$. The whole transformation is sound and complete.*

Proof. If L has at least two non-empty relevant contexts we first apply (shuffle*). If there is no failure and measure μ is not strictly decreased, then we reach the RBHOUP S^* of type "unique" with a path-unique cycle L^* of the same length as L that has fewer non-empty relevant contexts. If L has only one non-empty relevant context, then let $S^* := S$ and $L^* := L$.
If L^* is already special path-unique, then we are ready. In the other case we apply (shuffle**). If there is no failure and measure μ is not strictly decreased, then we eventually reach the RBHOUP (S') of type "unique" with a special path-unique cycle L' of the same length as L. This holds, since every application of (shuffle) strictly increases the main depth of the b-constrained suffix of the relevant context C_{h-1} for $h > 1$, or C_1 for

$h = 1$; hence at most $|C_h|$ applications of (shuffle) are necessary. Thus after application of (shuffle**) the new relevant context has only b-constrained subcontexts, and is thus itself b-constrained. Soundness and completeness of the complete procedure directly follow from Lemma 12.6. $\square$

12.3. The rule for special path-unique cycles

Now we consider the case where there is a (ψ_1, ψ_2)-minimal special path-unique cycle. That is, there is a (ψ_1, ψ_2)-minimal cycle with exactly one non-trivial relevant context C_h of the form

$$x_1\ s_1 \doteq x_2\ t_1, \ldots, x_{h-1}\ s_{h-1} \doteq x_h\ t_{h-1}, x_h\ s_h \doteq C_h[x_1\ t_h],$$

where C_h is b-constrained.

Recall that E is the upper bound for the exponent of periodicity of unifiers fixed in the decision algorithm. Recall also that C^e for a context C and a positive integer e means the expanded form $\underbrace{C \ldots C}_{e}$.

Definition 12.11 (*Solve-special-cycle*). The input is a decomposed RBHOUP S of type "unique" in $\beta\overline{\eta}$-normal form with a minimal-length special path-unique cycle L of the form described above. Select one of the following alternatives.

1. Apply (reduce-bv) using a variable $x \in \{x_1, \ldots, x_h\}$.
2. (a) Select some $0 \le e \le E$ and some (possibly trivial) proper prefix $C_{h,1}$ of C_h. Let $C_h = C_{h,1}C_{h,2}$.
   (b) For $i = 1, \ldots, h$, replace x_i by $\lambda y_i.C_h^e C_{h,1}[x_i'\ y_i]$ where x_i' is a fresh II-variable.
   (c) Use β-reduction to transform the system into $\beta\overline{\eta}$-normal form.
   (d) Select an index $1 \le j \le h$ and apply (reduce-bv) for x_j'.
3. This selection is only applicable if $h > 1$.
   (a) Select $e \le E$ and some (possibly trivial) proper prefix $C_{h,1}$ of C_h, such that $C_h = C_{h,1}C_{h,2}$ and $C_{h,2}$ has a first-order top level function symbol $f \in \Sigma_0$ of arity $n > 1$. If this is not possible, then fail.
   (b) For $i = 1, \ldots, h$ select an index k_i with $1 \le k_i \le n$ and instantiate x_i by $\lambda y_i.C_h^e C_{h,1}[f(z_{i,1}, \ldots, z_{i,k_i-1}, x_i'\ y_i, z_{i,k_i+1}, \ldots, z_{i,n})]$ with new I-variables $z_{i,l}$. At least one index k_j should be different from *firstdpos*$(C_{h,2}C_{h,1})$. The variables x_i' for $i = 1, \ldots, h$ are fresh II-variables, and $z_{i,l}$ are new I-variables. Use β-reduction to reach a $\beta\overline{\eta}$-normal form of all terms in S.
   (c) Apply (decomp) to the equations obtained from the equations of L by instantiation.
   (d) Apply (repvt) or (repvv) to all the equations $z_{i,l} \doteq t_{i,l}$ obtained from repeated (decomp) in (c) for the first $h - 1$ equations of L.
   (e) Then decompose the resulting RBHOUP.

Note that since the context C_h is b-constrained, for every pure unifier σ the expression $\sigma(C_h)$ is a first-order context over Σ_0 by Lemma 12.3.

Lemma 12.12. *Application of the rule (solve-special-cycle) to a decomposed RBHOUP S of type "unique" in $\beta\overline{\eta}$-normal form with a minimal-length special path-unique cycle*

L either fails or results in a decomposed RBHOUP S^* in $\beta\overline{\eta}$-normal form such that $\overline{\mu}(S^*) < \overline{\mu}(S)$.

Proof. If alternative 1 is selected, then the result follows from Lemma 8.3.

First assume that alternative 2 is selected. Steps (a)–(c) do not affect the measure component μ_1. The application of (reduce-bv) in (d) decreases μ_1, and hence we are done.

Assume now that alternative 3 is selected. As above we see that the steps do not affect the measure component μ_1. From the cycle equations

$$x_1\ s_1 \doteq x_2\ t_1, \ldots, x_{h-1}\ s_{h-1} \doteq x_h\ t_{h-1}, x_h\ s_h \doteq C_h[x_1\ t_h]$$

we obtain after Step (b)

$$\begin{array}{ll} & C_h^e C_{h,1}[f(z_{1,1}, \ldots, z_{1,k_1-1}, x_1'\ s_1', z_{1,k_1+1}, \ldots, z_{1,n})] \\ \doteq & C_h^e C_{h,1}[f(z_{2,1}, \ldots, z_{2,k_2-1}, x_2'\ t_1', z_{2,k_2+1}, \ldots, z_{2,n})], \\ \ldots & \\ & C_h^e C_{h,1}[f(z_{h-1,1}, \ldots, z_{h-1,k_{h-1}-1}, x_{h-1}'\ \ s_{h-1}', z_{h-1,k_{h-1}+1}, \ldots, z_{h-1,n})] \\ \doteq & C_h^e C_{h,1}[f(z_{h,1}, \ldots, z_{h,k_h-1}, x_h'\ t_{h-1}', z_{h,k_h+1}, \ldots, z_{h,n})], \\ & C_h^e C_{h,1}[f(z_{h,1}, \ldots, z_{h,k_h-1}, x_h'\ s_h', z_{h,k_h+1}, \ldots, z_{h,n})] \\ \doteq & C_h^{e+1} C_{h,1}[f(z_{1,1}, \ldots, z_{1,k_1-1}, x_1'\ t_h', z_{1,k_1+1}, \ldots, z_{1,n})]. \end{array}$$

Decomposition yields (among others) the equations

$$\begin{array}{ll} & f(z_{1,1}, \ldots, z_{1,k_1-1}, x_1'\ s_1', z_{1,k_1+1}, \ldots, z_{1,n}) \\ \doteq & f(z_{2,1}, \ldots, z_{2,k_2-1}, x_2'\ t_1', z_{2,k_2+1}, \ldots, z_{2,n}), \\ \ldots & \\ & f(z_{h-1,1}, \ldots, z_{h-1,k_{h-1}-1}, x_{h-1}'\ s_{h-1}', z_{h-1,k_{h-1}+1}, \ldots, z_{h-1,n}) \\ \doteq & f(z_{h,1}, \ldots, z_{h,k_h-1}, x_h'\ t_{h-1}', z_{h,k_h+1}, \ldots, z_{h,n}), \\ & f(z_{h,1}, \ldots, z_{h,k_h-1}, x_h'\ s_h', z_{h,k_h+1}, \ldots, z_{h,n}) \\ \doteq & C_{h,2} C_{h,1}[f(z_{1,1}, \ldots, z_{1,k_1-1}, x_1'\ t_h', z_{1,k_1+1}, \ldots, z_{1,n})]. \end{array}$$

Let $k = \mathit{firstdpos}(C_{h,2}C_{h,1})$. Decompose the above equations and collect the equations that result from pairing terms at index k. Let $C_{h,2}C_{h,1} = C'C''$ where C' has main depth 1. Then in the interval $2 \leq j < h$ all pairs of consecutive equations have either the form

$$\cdots \doteq z_{j,k}, \quad z_{j,k} \doteq \cdots$$

or

$$\cdots \doteq x_j'\ t_{j-1}', \quad x_j'\ s_j' \doteq \cdots$$

The final equation is either

$$z_{h,k} \doteq C''[f(z_{1,1}, \ldots, z_{1,k_1-1}, x_1'\ t_h, z_{1,k_1+1}, \ldots, z_{1,n})]$$

or

$$x_h'\ s_h \doteq C''[f(z_{1,1}, \ldots, z_{1,k_1-1}, x_1'\ t_h, z_{1,k_1+1}, \ldots, z_{1,n})].$$

If there are only equations $z_{j,k} \doteq z_{j+1,k}$ for all $1 \leq j < h$, then there is a fail due to occurs-check. In the remaining case there occurs at least one pair

$$\cdots \doteq x'_j \ t'_{j-1}, \quad x'_j \ s'_j \doteq \cdots$$

for some $2 \leq j < h$. Then the series of equations represents a cycle of length h. Moreover, there is at least one pair

$$\cdots \doteq z_{j,k}, \quad z_{j,k} \doteq \cdots$$

in the cycle since for at least one index j we have $k_j \neq k$ (cf. (b)). Using (repvt) for indices $j < h$ this yields a cycle of length $h-1$. Then also the RBHOUP S^* reached after decomposition has a cycle, L^*, of length $h-1$. In any case we obtain $\mu(S^*) < \mu(S)$. $\square$

Lemma 12.13. *The rule (solve-special-cycle) is sound and complete.*

Proof. *Soundness.* Assume there is a pure Σ_0-unifier σ^* of the RBHOUP S^* reached after the transformation. If alternative 1 is selected, then Lemma 8.3 shows that S has a pure Σ_0-unifier. Assume that alternative 2 is selected. By Part (a) of Lemma 8.3 the RBHOUP S' reached after Step (c) has a pure Σ_0-unifier σ'. For each $x_i \in \{x_1, \ldots, x_h\}$ define $\sigma(x_i)$ as the $\beta\overline{\eta}$-normal form of $\lambda y.\sigma'(C_h^e C_{h,1})[\sigma'(x'_i)\ y]$. For the remaining free variables z of S let $\sigma(z) := \sigma'(z)$. It is simple to show that σ' is a Σ_0-unifier for S. Since the context $C_h^e C_{h,1}$ is b-constrained, the ground context $\sigma'(C_h^e C_{h,1})$ is a first-order context. Hence the body of $\lambda y.\sigma'(C_h^e C_{h,1})[\sigma'(x'_i)\ y]$ is either a first-order ground term, or $A[y]$, where A is a ground first-order context (see Lemma 12.3). Hence σ is a pure Σ_0-unifier of S. If alternative 3 is selected, the proof is analogous.

Completeness. Let σ be a pure Σ_0-unifier for S with exponent of periodicity not exceeding E. Looking at alternative 1 it follows from Lemma 8.4 that it suffices to consider the case where each variable $x_i \in \{x_1, \ldots, x_h\}$ has a value $\sigma(x_i)$ of the form $\lambda y_i.g(t_{i,1}, \ldots, t_{i,k_i})$ where y_i is a first-order variable and $g(t_{i,1}, \ldots, t_{i,k_i})$ is a first-order term over Σ_0 with at most one occurrence of y_i that does not have any occurrence of another variable, and where g is the top level function symbol of C_h with $ar(g) \geq 1$. This holds, since σ is a unifier of the cycle L. For $x_i \in \{x_1, \ldots, x_h\}$, let $\sigma(x_i)$ be of the form $\lambda y_i.g(t_{i,1}, \ldots, t_{i,k_i})$. We say that x_i is a *context variable* (w.r.t. σ) iff $g(t_{i,1}, \ldots, t_{i,k_i})$ contains exactly one occurrence of y_i.

We first claim that at least one variable $x_i \in \{x_1, \ldots, x_h\}$ is a context variable: as above, let L have the form

$$x_1\ s_1 \doteq x_2\ t_1, \ldots, x_{h-1}\ s_{h-1} \doteq x_h\ t_{h-1}, x_h\ s_h \doteq C_h[x_1\ t_h],$$

let

$$T_1 := \sigma(x_1\ s_1) = \sigma(x_2\ t_1), \ldots, T_h := \sigma(x_h\ s_h) = \sigma(C_h[x_1\ t_h]). \qquad (\dagger)$$

If none of the variables $x_i \in \{x_1, \ldots, x_h\}$ is a context variable, then $T_1 = \sigma(x_1) = \sigma(x_2) = T_2 = \cdots = \sigma(C_h[x_1])$, which is impossible since C_h is non-empty. Hence the claim follows.

For each context variable $x_i \in \{x_1, \ldots, x_h\}$, let $\sigma(x_i) = \lambda y_i.D_i[y_i]$. The contexts D_i are non-empty for all $1 \leq i \leq h$ where x_i is a context variable. Let $G := \sigma(C_h)$. Let D_0

denote the maximal common prefix of all contexts D_i (where x_i is a context variable) and of G^{E+1}. There exists a (possibly trivial) prefix G_1 of $G = G_1[G_2]$ such that D_0 has the form $G^e[G_1]$ for some $0 \leq e \leq E$. G_1 has the form $\sigma(C_{h,1})$ for some prefix $C_{h,1}$ of $C_h = C_{h,1}[C_{h,2}]$. Note that G_1 must be a proper prefix of G by our assumption on E.

If $D_0 = D_j$ for some context variable x_j, $1 \leq j \leq h$, then we select alternative 2. The number e and the prefix $C_{h,1}$ are determined by the above equations. The reduction in (d) uses variable x'_j. It follows from the choice of D_0 that for each context variable x_i, $\sigma(x_i)$ always has the form $\sigma(x_i) = \lambda y_i.D_0[s_i^{(0)}]$ for suitable $s_i^{(0)}$. The equations in (†) show that the same kind of representation is possible for all $\sigma(x_i)$, $1 \leq i \leq h$. Define $\sigma'(x'_i) := \lambda y_i.s_i^{(0)}$. It is trivial to verify that σ' is a pure Σ_0-unifier of the RBHOUP S' reached after Step (c). The exponent of periodicity of σ' does not exceed the exponent of periodicity of σ. Moreover, by choice of the D_i, for index j we know $\sigma'(x'_j) = \lambda y_j.y_j$. Hence, applying (reduce-bv), we reach the RBHOUP S^*. As in the proof of Lemma 8.4 it follows that S^* has a Σ_0-unifier σ^* such that the exponent of periodicity of σ^* does not exceed the exponent of periodicity of σ'.

In the remaining case, D_0 is a proper prefix of all D_i where x_i is a context variable, $1 \leq i \leq h$. Furthermore, D_0 is a proper prefix of G^{E+1}. We first verify that $h \neq 1$.

Assume that $h = 1$. Then x_1 is a context variable. Let D_0 be represented in the form $G^e[G_1]$ as above. The equation $\sigma(x_1\, s_1) = \sigma(C_h[x_1\, t_1])$ shows that there must be an index $k \neq \mathit{firstdpos}(G_2)$ such that $\sigma(x_1) = \lambda y_1.G^e G_1[f(r_1, \ldots, r_k\lfloor y_1 \rfloor, \ldots, r_n)]$. The equation $x_1\, s_1 \doteq C_1[x_1\, t_1]$, after applying σ plus beta-reductions, has the form

$$G^e G_1[f(r_1, \ldots, r_k\lfloor s_1{}' \rfloor, \ldots, r_n)] \doteq G G^e G_1[f(r_1, \ldots, r_k\lfloor t_1{}' \rfloor, \ldots, r_n)].$$

This implies that

$$f(r_1, \ldots, r_k\lfloor s_1{}' \rfloor, \ldots, r_n) \doteq G_2 G_1[f(r_1, \ldots, r_k\lfloor t_1{}' \rfloor, \ldots, r_n)].$$

By assumption we have $G_2 G_1 = f(r'_1, \ldots, r'_{j-1}, G'\lfloor[\cdot]\rfloor, r'_{j+1}, \ldots, r'_n)$ where $k \neq j$. Then by decomposition we get that r_j has a subterm $f(r_1, \ldots, r_k\lfloor t_1{}' \rfloor, \ldots, r_n)$ on the surface, which is impossible.

Now assume that $h \geq 2$. For a context variable $x_i \in \{x_1, \ldots, x_h\}$, let $\sigma(x_i) = \lambda y_i.D_0[D'_i[y_i]]$. Let π denote the first-order position of the hole of D_0. Let f be the symbol at first-order position π of G^{E+1}, which is the topmost symbol of G_2. Using the equations in (†) it now follows that f is the symbol at first-order position π, for all terms $\sigma(x_i)$, $1 \leq i \leq h$. In fact, from the last equation we know that f is the symbol at first-order position π of T_h. Once we know that f is the symbol at first-order position π of T_l for some $1 < l \leq h$ it follows from our assumption that D'_l is non-empty and hence f is the symbol at first-order position π of $\sigma(x_l)$, which shows that f is the symbol at first-order position π of T_{l-1}. It follows that $f \in \Sigma_0$ and each value $\sigma(x_i)$ can be represented in the form $\lambda y_i.D_0[f(r_{i,1}, \ldots, r_{i,k_i-1}, r_{i,k_i}\lfloor s_i^* \rfloor, r_{i,k_i+1}, \ldots, r_{i,n})]$. The choice of D_0 implies that $k_j \neq \mathit{firstdpos}(G_2)$ for at least one index k_j. Hence f has arity > 1. We select alternative 3. The choice of e and of $C_{h,1}$ is as in the previous case. The indices k_i in Step (b) are triggered by the above representation of $\sigma(x_i)$. The choice of s_i^* shows that the terms $r_{i,l}$ for $l \neq k_i$ are ground first-order terms. It is now straightforward to see that the RBHOUP S' reached after Step (d) has a pure Σ_0-unifier σ' where the exponent of periodicity does not exceed E. The rest is standard. $\square$

Lemma 12.14. *Every application of the rules (shuffle*), (shuffle**), or (solve-special-cycle) either fails or transforms a decomposed RBHOUP S in $\beta\overline{\eta}$-normal form into a decomposed RBHOUP S' in $\beta\overline{\eta}$-normal form. Moreover, subt$(S') \subseteq$ subt(S).*

Proof. By inspecting the rule. $\square$

Remark 12.15. The treatment of RBHOUPs S of type "unique" can be summarized as follows. We focus a (ψ_1, ψ_2)-minimal cycle L. Since S is of type unique, L is path-unique. First, using iterated applications of the sound and complete rule (shuffle), we proceed until we either reach a RBHOUP S^* with smaller μ-measure, or a RBHOUP S' with a (ψ_1, ψ_2)-minimal and special path-unique cycle L' corresponding to L where $\overline{\mu}(S') = \overline{\mu}(S)$ (Lemma 12.10). The final application of rule (solve-special-cycle) is sound and complete and reduces $\overline{\mu}$ (Lemma 12.12). Summing up, the above rule applications are sound and complete, they lead to a finite number of alternative RBHOUPs of the form S^* where $\overline{\mu}(S^*) < \overline{\mu}(S)$ and hence $\mu(S^*) < \mu(S)$.

13. Complexity of bounded higher-order unification

In this section we want to give a lower bound on the complexity of bounded higher-order unification. We use the ideas of the proof of a lower non-elementary worst-case complexity bound in Wierzbicki (1999) which in turn is based on Statman (1979).

We require the following lemma, which follows by standard arguments from the proof of Lemma 3.14.

Lemma 13.1. *The computation of the $\overline{\eta}$-normal form of a term t can be done in polynomial time and space.*

The idea is now to show that bounded higher-order unification can simulate the solution of the question whether $s =_{\alpha\beta\eta} t$, where s, t are closed, and t is in $\beta\overline{\eta}$-normal form. This decision problem is known to be not elementary recursive (see Statman, 1979). By Lemma 13.1 it is no restriction to assume that s is in $\overline{\eta}$-normal form, since the computation of the $\overline{\eta}$-normal form can be done in polynomial time.

The trivial encoding is to consider the equation $s =_{\alpha\beta\eta} t$ as a bounded higher-order unification problem. We are a bit more ambitious and want to show that unifiability of BHOUPs where all terms are in $\beta\overline{\eta}$-normal form is also not elementary recursive.

The bounded higher-order unification problem is constructed from the equation $s =_{\alpha\beta\eta} t$ by hiding redexes, i.e., if s contains a redex $C[s_1\ s_2]$, then s is replaced by $C[x\ s_1\ s_2]$ for a new variable x, and the equation $x \doteq \lambda y_1, y_2.y_1\ y_2$ of appropriate type is added. This can be done for all β-redexes, such that there are no more redexes in the left-hand sides of the constructed higher-order unification problem Γ. In order to bring all terms of the new system into $\beta\overline{\eta}$-normal form we apply $\overline{\eta}$-normalization, resulting in a system Γ_1 with an only polynomial size increase in *maxtypesize*(Γ) (see Lemma 3.14).

The problem Γ_1 has the required form. If Γ_1 is solvable we know the unique solution by construction. Following Lemma 3.14 we can select a bound $b(x)$ that is a linear function of *maxtypesize*(Γ) for all variables x. Now (Γ_1, b) has a solution as a BHOUP iff s β-reduces to t. Hence:

Proposition 13.2. *The decision problem of BHOUPs in $\beta\overline{\eta}$-normal form is not elementary recursive.*

This could be interpreted as follows: The problem class (the algorithm) BHOUP has two sources of non-elementary recursive complexity:

– Guessing the exponent of periodicity (see Lemma 5.7),
– The beta-reduction in the step (Order-Reduction).

This clarifies our remarks in the conclusion of Schmidt-Schauß and Schulz (2002a).

14. Results and corollaries

We summarize the decidability results and also describe some improvements and reformulations of the theorems.

14.1. Higher-order unification

The main goal of the paper is to prove Theorem 7.26:

Unifiability of BHOUPs is decidable.

The decision procedure BHOU is very careful with the origin of used function symbols. If unifiability of arbitrary input BHOUPs w.r.t. a fixed given signature is an issue, then we can specialize the claim:

Theorem 14.1. *Let $\Sigma_0 \subseteq \Sigma$ be a signature. Then Σ_0-unifiability is decidable for input BHOUPs S where all function symbols occurring in S are in Σ_0 and where Σ_0 contains an elementary constant a^ι for each target type in subt(S).*

Proof. For finite Σ_0 this follows from the fact that for each input BHOUP of the aforementioned form we may select Σ_0 in the (Order-Reduction) step. It is simple to see that finiteness of Σ_0 does not represent a restriction, cf. Lemma 5.1. $\square$

It might be interesting to restrict the signature Σ_0 of codomain terms of unifiers in the sense that only some of the symbols of the BHOUP S may be used. We conjecture that the corresponding bounded unification problem is also decidable as long as Σ_0 contains elementary constants for all target types occurring in S. In order to prove this result, a lemma on a bound for the exponent of periodicity of minimal Σ_0-solutions of BHOUPs over a super-signature of Σ_0 would be needed. Note that a minimal Σ_0-solution is not necessarily minimal w.r.t. the set of all solutions.

Remark 14.2. Whether it is possible to use a bounding function that only refers to the number of occurrences of bound variables (in contrast to the sum of the number of occurrences of bound variables and the number of occurrences of lambda-binders) is an open question. One obstacle is the term representation and the estimate for the representation size in Lemma 5.4, which uses first-order contexts. It is not obvious how to generalize this construction to a measure ignoring the number of lambdas, since a context of the form $\lambda x_1. f_1(\lambda x_2. f_2(\ldots\ (\lambda x_n. f_n(\ldots[\cdot]))))$ may be constructed. However, this context is not a first-order context, and moreover, it may be destroyed during reduction of terms $\sigma(s)$ to their normal form.

14.2. Higher-order matching

Currently, it is not known whether higher-order ($\alpha\beta\eta$-) matching is decidable, however there is some knowledge about decidability and complexity of special cases (Wolfram, 1993; Dowek, 1992, 1994; Comon and Jurski, 1997; Wierzbicki, 1999; Padovani, 2000; Schmidt-Schauß, 2003). Note that under $\alpha\beta$-convertibility higher-order matching is undecidable (Loader, 2003).

It is clear that bounded higher-order matching as a special case of bounded higher-order unification is decidable. The techniques in this paper permit to show that a variant of higher-order matching with a bound ignoring the lambdas (i.e. the same bound as for bounded second-order unification) is decidable:

Let S be a HOUP in $\beta\overline{\eta}$-normal form, such that in every equation $s \doteq t$ in S, the right-hand side t has no occurrences of free variables. Then S is called a higher-order matching problem. Let b be a function from free variables to $\mathbb{N}$. Then S is called a *bounded higher-order matching problem (BHOMP)*. A substitution σ in $\beta\overline{\eta}$-normal form is a *solution* of a (BHOMP), iff σ is a unifier of S, and furthermore, for every free variable x in S, the number of bound variables in $\sigma(x)$ is not greater than $b(x)$.

Theorem 14.3. *Bounded higher-order matching is decidable*

Proof. Using a similar technique as in the proof of soundness and completeness of (constantify) (see 7.14), it is easy to prove that in a minimal unifier σ the number of occurrences of function symbols is not greater than the number of occurrences of function symbols in the right-hand side of S. Lemma 5.1 shows that the types of subterms of terms in the codomain of σ are already in $\mathit{subt}(S)$. Hence, lambda-prefixes in the codomain are bounded by the maximal arity of types in $\mathit{subt}(S)$. Since codomain terms are in $\beta\overline{\eta}$-normal form, we conclude that the following holds: there is a constant $c(S)$, such that $\mathit{size}(\sigma(x)) \leq c(S) * b(x)$. In summary, decidability follows, since it is only necessary to test a finite number of potential unifiers, which are effectively enumerable. $\square$

This theorem is comparable with the result on the decidability of k-duplicating higher-order matching in Dougherty and Wierzbicki (2002).

14.3. Bounded second-order unification

The results in this paper are a generalization of the decidability result for bounded second-order unification (Schmidt-Schauß, 1999a, 2004). The specializations for second-order (which is treated in an untyped manner in Schmidt-Schauß (1999a, 2004)) are:

- There is exactly one elementary type ι.
- All function symbols in the signature have type of the form $\iota \to \cdots \to \iota$.

– In unification problems, every type of a subterm is either ι or a function type of the form $\iota \to \cdots \to \iota$. However, there are nor abstractions. In particular, every free variable has type ι or $\iota \to \cdots \to \iota$, which corresponds to the distinction between first-order variables and second-order variables. It follows also that every bound variable has type ι.

In the following, recall that in bounded second-order unification we only count the number of occurrences of bound variables in substitution terms, the number of lambda-binders is not taken into account. Hence we have to show that we may imitate this kind of bounding function with the formalism of this paper.

It is easy to see that second-order unifiers either instantiate variables by a ground first-order term, or by a term with a lambda-prefix and a first-order term as body. Hence a given bound $b_2(x)$ on the number of occurrences of bound variables in a codomain term in bounded second-order unification can be translated into an equivalent bound $b_h(x)$ for a higher-order unification problem by defining $b_h(x) = 0$ iff x is a first-order variable, and $b_h(x) = m + b_2(x)$ iff x is a second-order variable of arity m.
We obtain as corollary of Theorem 7.26.

Corollary 14.4. *Bounded second-order unification is decidable.*

Acknowledgements

The authors thank the referees of the *Journal of Symbolic Computation* for carefully reading the manuscript. Their comments helped to improve the presentation. We gratefully acknowledge a significant contribution from one referee, who suggested the use of the (Order-Reduction) rule as an initial step. This step simplified the presentation of the decision algorithm.

References

Andrews, P., 1986. An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof. Academic Press.

Andrews, P., 2001. Classical type theory. In: Robinson, A., Voronkov, A. (Eds.), Handbook of Automated Reasoning, vol. 2. North-Holland, pp. 965–1007 (Chapter 15).

Barendregt, H.P., 1984. The Lambda Calculus. Its Syntax and Semantics. North-Holland, Amsterdam, New York.

Barendregt, H.P., 1990. Functional programming and lambda calculus. In: van Leeuwen, J. (Ed.), Handbook of Theoretical Computer Science: Formal Models and Semantics, vol. B. Elsevier, pp. 321–363 (Chapter 7).

Beckmann, A., 2001. Exact bounds for lengths of reductions in typed λ-calculus. J. Symbolic Logic 66, 1277–1285.

Bird, R., 1998. Introduction to Functional Programming using Haskell. Prentice Hall.

Burstall, R., MacQueen, D., Sanella, D.T., 1980. Hope: an experimental applicative language. In: Proc. LISP Conference. pp. 136–143.

Baader, F., Siekmann, J., 1994. Unification theory. In: Gabbay, D.M., Hogger, C.J., Robinson, J.A. (Eds.), Handbook of Logic in Artificial Intelligence and Logic Programming. Oxford University Press, pp. 41–125.

Comon, H., Jurski, Y., 1997. Higher-order matching and tree automata. In: Proc. of CSL 97. In: LNCS, vol. 1414. pp. 157–176.

Comon, H., 1998. Completion of rewrite systems with membership constraints. Part I: Deduction rules. J. Symbolic Comput. 25 (4), 397–419.

Cervesato, I., Pfenning, F., 1997. Linear higher-order pre-unification. In: Proc. 12th LICS. pp. 422–433.

Dershowitz, N., Jouannaud, J.-P., 1990. Rewrite systems. In: van Leeuwen, J. (Ed.), Handbook of Theoretical Computer Science: Formal Models and Semantics, vol. B. Elsevier, pp. 243–320 (Chapter 6).

Dowek, G., 1992. Third order matching is decidable. In: Proceedings of the 7th Annual IEEE Symposium on Logic in Computer Science. pp. 2–10.

Dowek, G., 1994. Third order matching is decidable. Ann. Pure Appl. Logic 69 (2–3), 135–155.

Dowek, G., 2001. Higher-order unification and matching. In: Robinson, A., Voronkov, A. (Eds.), Handbook of Automated Reasoning, vol. 2. North-Holland, pp. 1009–1062 (Chapter 16).

Dougherty, D., Wierzbicki, T., 2002. A decidable variant of higher order matching. In: Proc. RTA'02. In: LNCS, vol. 2378. Springer, pp. 340–351.

Farmer, W.A., 1988. A unification algorithm for second order monadic terms. Ann. Pure Appl. Logic 39, 131–174.

Farmer, W.A., 1991. Simple second-order languages for which unification is undecidable. J. Theoret. Comput. Sci. 87, 173–214.

Gandy, R.O., 1980. Proofs of strong normalization. In: Seldin, J.P., Hindley, J.R. (Eds.), H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism. Academic Press, pp. 457–477.

Goubault-Larrecq, J., Mackie, I., 1997. Proof Theory and Automated Deduction. In: Applied Logic Series, vol. 6. Kluwer.

Goldfarb, W.D., 1981. The undecidability of the second-order unification problem. Theoret. Comput. Sci. 13, 225–230.

Gutierrez, C., 1998. Satisfiability of word equations with constants is in exponential space. In: Proceedings FOCS'98. IEEE Computer Society Press, Palo Alto, CA, pp. 112–119.

Hindley, J.R., 1997. Basic Simple Type Theory. In: Cambridge tracts in theoretical computer science. Cambridge University Press.

Hanus, M., Kuchen, H., Moreno-Navarro, J.J., 1995. Curry: A truly functional logic language. In: Proc. ILPS'95 Workshop on Visions for the Future of Logic Programming. pp. 95–107.

Hindley, J.R., Seldin, J.P., 1986. Introduction to Combinators and λ-calculus. Cambridge University Press.

Huet, G., 1975. A unification algorithm for typed λ-calculus. Theoret. Comput. Sci. 1, 27–57.

Huet, G., 1976. Résolution d'équations dans des langages d'ordre 1,2,... ω. Thèse de doctorat d'etat, Université Paris VII (in French).

Jensen, D., Pietrzykowski, T., 1976. Mechanizing ω-order type theory through unification. Theoret. Comput. Sci. 3 (2), 123–171.

Klop, J.W., 1992. Term rewriting systems. In: Abramsky, S., Gabbay, D.M., Maibaum, T.S.E. (Eds.), Handbook of Logic in Computer Science, vol. 2. Oxford University Press, pp. 2–116.

Kościelski, A., Pacholski, L., 1996. Complexity of Makanin's algorithms. J. Assoc. Comput. Machinery 43, 670–684.

Levy, J., 1996. Linear second order unification. In: Proceedings of the 7th International Conference on Rewriting Techniques and Applications. In: Lecture Notes in Computer Science, vol. 1103. pp. 332–346.

Loader, R., 2003. Higher order beta matching is undecidable. Logic J. IGPL 11 (1), 51–68.

Levy, J., Veanes, M., 2000. On the undecidability of second-order unification. Inform. and Comput. 159, 125–150.

Levy, J., Villaret, M., 2000. Linear second-order unification and context unification with tree-regular constraints. In: Proceedings of the 11th Int. Conf. on Rewriting Techniques and Applications. In: Lecture Notes in Computer Science, vol. 1833. pp. 156–171.

Makanin, G.S., 1977. The problem of solvability of equations in a free semigroup. Math. USSR Sbornik 32 (2), 129–198.

Miller, D., 1991. A logic programming language with lambda-abstraction, function variables and simple unification. J. Logic Comput. 1 (4), 497–536.

Narendran, P., 1990. Some remarks on second order unification. Technical Report, Inst. of Programming and Logics, Department of Computer Science, Univ. of NY at Albany.

Nipkow, T., 1991. Higher-order critical pairs, Proc. 6th IEEE Symp. LICS, pp. 342–349.

Niehren, J., Pinkal, M., Ruhrberg, P., 1997. On equality up-to constraints over finite trees, context unification, and one-step rewriting. In: Proceedings of the International Conference on Automated Deduction. In: Lecture Notes in Computer Science, vol. 1249. pp. 34–48.

Niehren, J., Tison, S., Treinen, R., 2000. On rewrite constraints and context unification. Inform. Process. Lett. 74, 35–40.

Padovani, V., 2000. Decidability of fourth-order matching. Math. Structures Comput. Sci. 10 (3), 361–372.

Paulson, L.C., 1991. ML for the Working Programmer. Cambridge University Press.

Paulson, L.C., 1994. Isabelle. In: Lecture Notes in Computer Science, vol. 828. Springer-Verlag.

Pfenning, F., 2001. Logical frameworks. In: Robinson, A., Voronkov, A. (Eds.), Handbook of Automated Reasoning, vol. 2. North-Holland, pp. 1063–1147 (Chapter 17).

Plandowski, W., 1999. Satisfiability of word equations with constants is in PSPACE. In: FOCS 99. pp. 495–500.

Schwichtenberg, H., 1982. Complexity of normalization in the pure typed λ-calculus. In: Troelstra, A.S., van Dalen, D. (Eds.), The L.E.J. Brouwer Centenary Symposium. Proceedings of the Conference hold in Noordwijkerhout. 8–13 June, 1981. In: Studies in Logic and the Foundations of Mathematics, vol. 110. North Holland, pp. 453–458.

Schulz, K.U., 1990. Makanin's algorithm — two improvements and a generalization. In: Proc. of IWWERT 1990. In: Lecture Notes in Computer Science, vol. 572. Springer-Verlag, pp. 85–150.

Schwichtenberg, H., 1991. An upper bound for reduction sequences in the typed λ-calculus. Arch. Math. Logic 30, 405–408. Dedicated to Kurt Schütte on the occasion of his 80th birthday.

Schulz, K.U., 1993. Word unification and transformation of generalized equations. J. Automat. Reason. 149–184.

Schmidt-Schauß, M., 1994. Unification of stratified second-order terms. Internal Report 12/94, Fachbereich Informatik, J.W. Goethe-Universität Frankfurt, Frankfurt, Germany.

Schmidt-Schauß, M., 1999a. Decidability of bounded second order unification. Frank Report 11, FB Informatik, J.W. Goethe-Universität Frankfurt am Main. http://www.ki.informatik.uni-frankfurt.de/papers/articles.html.

Schmidt-Schauß, M., 1999b. A decision algorithm for stratified context unification. Frank-Report 12, Fachbereich Informatik, J.W. Goethe-Universität Frankfurt, Frankfurt, Germany. Available at http://www.ki.informatik.uni-frankfurt.de/papers/articles.html.

Schmidt-Schauß, M., 2001. Stratified context unification is in PSPACE. In: Proceedings of CSL'01. In: LNCS, vol. 2142. pp. 498–512.

Schmidt-Schauß, M., 2002. A decision algorithm for stratified context unification. J. Logic Comput. 12 (6), 929–953.

Schmidt-Schauß, M., 2003. Decidability of arity-bounded higher-order matching. In: CADE-19. In: LNCS, 2741. Springer, pp. 488–502.

Schmidt-Schauß, M., 2004. Decidability of bounded second order unification. Inform. and Comput. 188 (2), 143–178.

Schmidt-Schauß, M., Schulz, K.U., 1998. On the exponent of periodicity of minimal solutions of context equations. In: Proceedings of the 9th Int. Conf. on Rewriting Techniques and Applications. In: Lecture Notes in Computer Science, vol. 1379. pp. 61–75.

Schmidt-Schauß, M., Schulz, K.U., 2002a. Decidability of bounded higher-order unification. In: CSL 2002. In: LNCS, vol. 2471. Springer-Verlag, pp. 522–536.

Schmidt-Schauß, M., Schulz, K.U., 2002b. Solvability of context equations with two context variables is decidable. J. Symbolic Comput. 33 (1), 77–122.

Statman, R., 1979. The typed λ-calculus is not elementary recursive. Theoret. Comput. Sci. 9, 73–81.

Turner, D.A., 1985. Miranda: A non-strict functional language with polymorphic types. In: Functional Programming Languages and Computer Architecture. In: Lecture Notes in Computer Science, vol. 201. Springer, pp. 1–16.

Vorobyov, S., 1998. $\forall\exists^*$-equational theory of context unification is Π_1^0-hard. In: MFCS 1998. In: Lecture Notes in Computer Science, vol. 1450. Springer-Verlag, pp. 597–606.

Wierzbicki, T., 1999. Complexity of the higher-order matching. In: Proc. 16th CADE. In: LNCS, vol. 1632. Springer-Verlag, pp. 82–96.

Wolfram, D.A., 1993. The Clausal Theory of Types. In: Cambridge Tracts in Theoretical Computer Science, vol. 21. Cambridge University Press.

Zhezherun, A.P., 1979. Decidability of the unification problem for second order languages with unary function symbols. Kibernetika (Kiev) 5, 120–125; Translated as 1980. Cybernetics 15 (5), 735–741.