Decidability of Bounded Second Order Unification

Manfred Schmidt-Schauß

Fachbereich Informatik, Johann Wolfgang Goethe-Universität, Postfach 11 19 32,
D-60054 Frankfurt,Germany
Tel: (+49)69-798-28597, Fax: (+49)69-798-28919,
E-mail: schauss@ki.informatik.uni-frankfurt.de
URL: www.ki.informatik.uni-frankfurt.de/

Abstract. It is well-known that first order unification is decidable, whereas second order (and higher-order) unification is undecidable. Bounded second order unification (BSOU) is second order unification under the restriction that only a bounded number of holes in the instantiating terms for second order variables is permitted, however, the size of the instantiation is not restricted. In this paper, a decision algorithm for bounded second order unification is described.
This is the first non-trivial decidability result for second order unification, where the (finite) signature is not restricted and there are no restrictions on the occurrences of variables.
We show that the monadic second order unification (MSOU), a specialization of BSOU is in Σ_2^p. Since MSOU is related to word unification, this is compares favourably to the best known upper bound NEXPTIME (and also to the announced upper bound PSPACE) for word unification. This supports the claim that bounded second order unification is easier than context unification, whose decidability is currently an open question.

Keywords: unification, second order unification, context unification, automated deduction, rewriting, logics in artificial intelligence.

1 Introduction

Unification is solving equations. In Automated Deduction systems based on first order predicate logic, in particular in Logic Programming, (first order) unification is a central operation. It is the question whether two terms formed from function symbols and variables can be made equal by instantiating the variables by terms. It is well-known that first order unification is decidable in linear time (see the overview article [BS94]).

Second Order Unification (SOU) generalizes first order unification by extending terms with second order variables that may be instantiated with partial terms. Second order unification is a specialization of higher-order unification [Hue75,SG89,Wol93].

Goldfarb [Gol81] has shown that second order unification is undecidable. This result was sharpened in [Far91] for a restricted signature and in [Lev98,LV98] for severe restrictions on occurrences of second order variables.

Monadic second order unification (MSOU) (see [Far88]) is second order unification where the signature has only monadic function symbols, i.e., there are no function symbols of arity 2 or more. Its decidability follows from decidability of string unification [Mak77]. Recently obtained upper bounds for the complexity of word unification are EXPSPACE ([Gut98]), which was improved to NEXPTIME [Pla99a] and recently to even PSPACE [Pla99b].

Another kind of restriction which has recently attracted some interest is to permit as instantiations for the second order variables only terms where the number of bound variables is in a prescribed set. If the number is exactly one, then this is called context unification. It is currently an open question, whether (general) context unification is decidable.

If context unification is restricted, then there are some results on decidability: If the number of context variables is at most two [SSS98a,SSS98b,SSS99]; If the nesting of second order variables has a certain form [SS94,SS98]; or if every variable and every context variable occurs at most twice [Lev96].

In this paper we consider the problem of *bounded second order unification* (BSOU) where the number of holes in the instantiation of second order variables is bounded, but zero is also permitted; and also the specialization of monadic second order unification (MSOU). The new results in this paper are: bounded second order unification is decidable; as a lower complexity bound, it is shown that BSOU and MSOU are $\mathcal{NP}$-hard. For MSOU we show that it is in Σ_2^p

The result on decidability shows that a slight restriction makes second order unification decidable, which has a potential usage in implementations. A semi-decision procedure can easily be built upon such a decision algorithm by increasing the bound on the number of holes.

The relation to the open problem of decidability of context unification is as follows. Decidability of context unification would imply decidability of the bounded second order unification problem. Nevertheless, the algorithm given in this paper is of interest, since it is independent of [Mak77], whereas a (hypothetical) decision algorithm for context unification must also solve the string unification problem.

The decision algorithm for bounded SOU cannot be turned into a decision algorithm for context unification, since a final (pre-solved) unification problem wrt bounded unification may be unsolvable wrt context unification, and moreover it is completely unclear how to decide whether a final problem is unifiable wrt context unification.

Thus, second order unification with a restriction on the number of holes splits into two apparently different problems: One is bounded second order unification, the other is context unification. The main difference is whether zero holes are permitted or not.

The paper is structured as follows: After explaining the notation in section 2 in section 3 bounded second order unification is shown to be equivalent to Z-

context unification. The decidability of Z-context unification is shown in sections 4,5, 6. Section 7 contains a proof that bounded and monadic second order uification is $\mathcal{NP}$-hard. Section 8 contains the argumentation that monadic secodn order unification is low in the polynomial hierarchy: it is in Σ_2^p. The appendix contains proofs of the key lemmas of section 6.

2 Preliminaries

Let Σ be a finite signature containing function symbols and at least one constant. Let $\mathcal{V}_1$ be the (infinite) set of first order variables, let $\mathcal{V}_{2,i}$ be the (infinite) set of second order variables of arity i, and let $\mathcal{V}_2 := \bigcup \mathcal{V}_{2,i}$. First order variables are denoted by letters x, y, z, second order variables by letters X, Y, Z, and if we mean first or second order, then we use $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$. The arity of a symbol X or f is denoted as $ar(X)$ or $ar(f)$, respectively. A term in $\mathcal{T}(\Sigma, \mathcal{V}_1, \mathcal{V}_2)$ is either a first order variable, or $X(t_1, \ldots, t_n)$, or $f(t_1, \ldots, t_n)$, where t_i are terms and $ar(X) = n$ or $(ar(f) = n)$; X (or f) is called the *head* of $X(t_1, \ldots, t_n)$, (or $f(t_1, \ldots, t_n)$), respectively. In the following we will use an (untyped) lambda-notation for term functions, where the bound variable is always a first order variable, and the body is a term. A ground substitution is a function that replaces first order variables by ground terms, and second order variables by term functions of the form $\lambda x_1, \ldots, x_m.t$, where x_i are the only variables that occur in t. The application of a ground substitution to a term can be simplified into a unique ground term. A *context* is a lambda expression $\lambda x.t$, where x occurs exactly once in t, i.e., a term with exactly one hole. If there are no other variables in t, then this is a *ground context*. The path from the root of a context C to the hole is called *main path* and its length is called *main depth* of C, denoted as $|C|$. For two contexts C, D, we mean by the concatenation $C \cdot D$ the term function $\lambda x.C(D(x))$. Usually we omit the $\cdot$ and write CD for $C \cdot D$ and $CD(s)$ for the expression $(C \cdot D)(s)$. If k is a natural number and C is a context, C^k means the k-fold application, i.e., $C^1 := C, C^{i+1} := CC^i$. D is a *prefix* of a context C, iff there is some context C' such that $C = DC'$.

We use the notations Id for the instantiation $\lambda x.x$, and $K\ s$ for an instantiation $\lambda x.s$, where s does not contain x (K is the constant combinator $\lambda x\ .\ \lambda y\ .\ x$). We usually assume that an instantiation Id or $K\ s$ for a second order variable in a term is immediately simplified: $Id(t)$ is replaced by t, and $(K\ s)(t)$ by s. In order to simplify index notation for certain cyclic equations, we shall use j `mod*` n, and mean the number that is in the interval $[1, n]$.

Definition 2.1. *A* second order unification problem (SOUP) *is a set of (symmetric) equations* $\{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$ *where* s_i, t_i *are terms. A substitution* σ *such that for all* i $\sigma(s_i), \sigma(t_i)$ *are equal ground terms is called a* unifier *of* $\{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$.

In the following we consider restricted second order problems, where an upper bound on the number of occurrences of the lambda-bound variable in instantiations of the second order variables is given.

Definition 2.2. *let S be a SOUP. Let there be a function $b : \mathcal{V}_2 \to \mathbb{N}$. Then the pair (S, b) is called a bounded SOUP (BSOUP). A substitution σ is a* unifier *of a BSOUP (S, b), iff σ is a unifier of S and for every variable X where $\sigma(X) = \lambda y_1, \ldots, y_k.t_X$, the number of all occurrences of y_i in t_X is not greater than $b(X)$.*

Definition 2.3. *Let (S, b) be a BSOUP. A second order variable X is called a* Z-context, *iff it has arity 1 and $b(X) = 1$. If all second order variables in (S, b) are Z-contexts, then (S, b) is called a* Z-context unification problem (ZCUP). *In this case we usually write only S instead of (S, b).*

3 Bounded second order unification

In this section we argue that decidability of Z-context-unification is equivalent to decidability of bounded second order unification.

Definition 3.1. *The following rule is used to non-deterministically translate a BSOUP (S, b) into (S', b').*

Select a second order variable X that is not a Z-context. Select one of the following possibilities:

1. *This selection is applicable only if $b(X) > 1$. Define $b'(X) := b(X) - 1$ and $b'(Y) := b(Y)$ for second order variables $Y \neq X$.*
2. *This selection is applicable only if $b(X) \geq 1$ and $ar(X) > 1$. Let X' be a new second order variable with $ar(X') = ar(X) - 1$. Replace every occurrence of $X(t_1, \ldots, t_i, \ldots, t_n)$ by $X'(t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n)$, and define $b'(X') = b(X)$, and $b'(Y) = b(Y)$ for $Y \neq X'$.*
3. *This selection is applicable only if $b(X) > 1$. We guess the topmost branching node of an instantiation for X. The guess is of the form $X_0(f(r_1, \ldots, r_n))$, where at least two r_i contain holes of the instantiation of X.*
   More formally: Select a symbol f in the signature, of arity $n \geq 2$. Let $I_1 \cup I_2$ be a partition of $\{1, \ldots, n\}$ with $|I_2| \geq 2$. Let $z_i, i \in I_1$ be new first order variables, and $X_i, i \in \{0\} \cup I_2$ be new second order variables. The following conditions should hold: $ar(X_0) = 1$, $ar(X_i) = ar(X)$. Select a function b' such that: $b'(X_0) = 1$, $b(X) = \sum_{i \in I_2} b'(X_i)$, and $b'(X_j) \neq 0$ for all $j \in I_2$, $b'(Y) = b(Y)$ for $Y \notin \{X\} \cup \{X_i \mid i \in I_2\}$. Generate S' from S by replacing the subterms $X(s_1, \ldots, s_m)$ by $X_0(f(r_1, \ldots, r_n))$, where $r_i = z_i$ for $i \in I_1$ and $r_i = X_i(s_1, \ldots, s_m)$ for $i \in I_2$.

Proposition 3.2. *b-bounded second order unification is decidable iff Z-context unification is decidable.*

Proof. Soundness and completeness can be seen by standard methods. Termination is shown using the multiset $\{(b(X), ar(X)) \mid X \text{ occurs in } S\}$, where the pairs are ordered lexicographically, and the multiset is ordered by the induced multiset ordering. □

4 Decidability of Z-context Unification: Survey

In this section we start with a survey of the decision procedure for BSOU. Sections 5, and 6 are devoted to describing its parts.

Theorem 4.1. *Solvability of ZCUPs is decidable.*

Corollary 4.2. *Bounded second order unification is decidable.*

A *minimal* unifier of a ZCUP S is a unifier such that the sum of the sizes of the ground terms/contexts assigned to the variables in the problem is minimal with respect to all unifiers of the problem. The *exponent of periodicity* (see also [SSS98a]) of a unifier σ of S is the maximal number n such that for some variable x (resp. X) occurring in the problem S, its value $\sigma(x)$ (resp. $\sigma(X)$) contains a non-empty subcontext of the form C^n, where C is a ground context.

Note that a unifier σ', where $\sigma'(x)(\sigma'(X)$, respectively), is a subterm/context of the respective $\sigma(x)(\sigma(X)$, always has an exponent of periodicity not greater than that of σ.

Lemma 4.3. *There are constants c, d, such that for every unifiable Z-context problem S the exponent of periodicity of a minimal unifier of S is less than $c * (2^{2.14*d*N})$, where N is the size of S.*

Proof. Follows from [SSS98a] using standard arguments. □

Given a ZCUP S_0 as input, the decision algorithm determines the existence of a solution of S where the exponent of periodicity does not exceed the bound E given in Lemma 4.3.

Definition 4.4. *A non-deterministic transformation rule $\mathcal{T}$ that transforms a ZCUP S into other ZCUPs is called*

- sound, *if whenever S is transformed by $\mathcal{T}$ into S', and S' is solvable, then S is solvable.*
- complete, *iff the following holds: If S has a solution with exponent of periodicity not greater than E, then $\mathcal{T}$ can transform S into a ZCUP S' that has a solution with exponent of periodicity not greater than E.*

In the following section we introduce four particular types of ZCUPs (types 0–3), where type 0 is trivially solvable (it is pre-unified), and a well-founded measure μ for ZCUPs.

The algorithm starts with an initial step where S_0 is transformed into a ZCUP of type 0–3. In addition, we describe (non-deterministic) transformation rules that transform ZCUPs of type 1–3 into ZCUPs of type 0–3, such that the transformation is sound and complete, and if S is transformed into S', then $\mu(S') < \mu(S)$ (see Lemmas 6.2, 6.4, 6.9, 6.11).

Since μ is well-founded, and every problem of type 1–3 can be transformed, it follows that the non-deterministic search terminates either with fail, or with a problem of type 0. Soundness and completeness imply that the input problem S_0 has a solution iff some ZCUP of type 0 is reached. Theorem 4.1 is an immediate consequence.

(decomp) $\dfrac{f(s_1,\ldots,s_n) \doteq f(t_1,\ldots,t_n) \wedge S}{s_1 \doteq t_1 \wedge \ldots \wedge s_n \doteq t_n \wedge S}$

(repvv) $\dfrac{x \doteq y \wedge S}{S'}$ where y is a first order variable, S' is constructed from S by replacing all occurrences of x by y.

(repvt) $\dfrac{x \doteq t \wedge S}{x \doteq t \wedge S'}$ where $t \neq x$ is not a variable, $x \doteq t$ is part of a Z-cycle, and S' is constructed from S by replacing all surface occurrences of x by t.

Table 1. The decomposition rules

5 Preparing Definitions

In order to describe the main reduction techniques, the following notions play a central role. Note that we use $1 \leq i \texttt{ mod* } n \leq n$.

Definition 5.1. *A position p in t is* on the surface *of t, iff there is no proper prefix p' of p such that $t|_{p'}$ is of the form $Y(s)$. The depth of a surface position p is the length of p.*

A variable x (a Z-context variable X, a function symbol f, respectively) occurs on the surface *of t, iff x (a term $X(s)$ or $f(t_1,\ldots,t_n)$, for some s or t_i, respectively) occurs on a surface position of t.*

We use the notation $t[s]$ or $t[X]$ to indicate that t has a surface occurrence of s, or X, respectively.

Definition 5.2. *Let S be a ZCUP. A* Z-cycle *is a sequence $s_1 \doteq t_1,\ldots,s_h \doteq t_h$ of length $h \geq 1$ of equations from S, such that for all $1 \leq i \leq h$: Either s_i is a variable that occurs on the surface of $t_{i-1 \texttt{ mod* } h}$, or $s_i \equiv X_i(\ldots)$, and X_i occurs on the surface of $t_{i-1 \texttt{ mod* } h}$. Moreover, there should be 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 $X_i(s_{i,1})$. A Z-cycle is* path-unique *if for every $1 \leq i \leq h$ there is only one occurrence of x_i (resp. X_i) on the surface of $t_{(i-1) \texttt{ mod* } h}$.*

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

Definition 5.3. *The decomposition rules are defined in table 1. Decomposition fails, if there is an equation $f(\ldots) \doteq g(\ldots)$ with $f \neq g$ (clash), or if there is a chain of equations $x_1 \doteq t_1 \wedge \ldots \wedge x_n \doteq t_n$, such that for all $i = 1,\ldots,n$: t_i is not a variable, $head(t_i)$ is a function symbol in the signature, and x_i occurs on*

the surface of $t_{(i-1) \bmod^* n}$. *(occurs-check).*
A ZCUP S *is* decomposed *if no decomposition rule and no failure rule is applicable.*

We assume that failure rules have highest priority. In general the order of application of the remaining rules is arbitrary.

In a decomposed unification problem the Z-cycles of minimal length are in a compressed form, i.e., all equations in a shortest Z-cycle are of the form $X(s) \doteq t$.

Definition 5.4. *Let* S *be a decomposed unification problem and* L *be a shortest Z-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* r_i *be the smallest subterm of* t_i, *such that all surface occurrences of* X_{i+1} *from* t_i *are also contained in* t_i. *The* relevant context C_i *of equation* i *is uniquely determined by* $t_i = C_i[r_i]$.

Example 5.5. Let $X_i(s) \doteq g(f(X_{i+1}(t_1), X_{i+1}(t_2)))$ be part of a Z-cycle, then the relevant context is $C_i = g(.)$.

Definition 5.6. *The lexicographic measure* $\psi(L) = (\psi_1(L), \psi_2(L), \psi_3(L))$ *of a Z-cycle* L *of a decomposed ZCUP* S *has the following three components:*

1. *The length* h *of* L.
2. 0, *if* L *is non-path-unique, and* 1, *if* L *is path-unique.*
3. – *If* L *is non-path-unique, then the minimal main depth of a relevant context* C_j *of* L *where* t_j *contains at least two different surface-occurrences of* $X_{(j+1) \bmod^* h}$.
   – *If* L *is path-unique, then the number of indices* $1 \leq i \leq h$ *such that* C_i *is not trivial.*

Definition 5.7. *The measure* μ *on a ZCUP* S *is a lexicographic one with the following components* μ_1, μ_2, μ_3:

1. *The number of Z-context variables.*
2. *The minimal* $\psi(L)$ *for all Z-cycles. If there is no Z-cycle, then* ∞.
3. *The number of occurrences of function symbols in* S *on surface positions.*

Lemma 5.8. *The decomposition rules are sound, complete and terminate.*

If S' *is the result of exhaustingly decomposing* S, *then* S *has a Z-cycle implies that* S' *has a Z-cycle. Moreover, the number of Z-context variables is unchanged, and if* S *contains a Z-cycle, the the minimal* ψ_1 *is not increased.*

Proof. It is easy to see that soundness and completeness holds. Termination follows, since the following lexicographic measure is properly decreased in every step: 1) The number of first order variables on the surface, 2) the size of S.

Since (repvt) can only be applied for equations $x \doteq t$ in Z-cycles, it is clear that if S contains a Z-cycle, then S' contains a Z-cycle, too.

The rules are not able to modify the number of Z-context variables, hence μ_1 is unchanged. The length of Z-cycles can only be shortened by (repvv) or (repvt). □

Note that the decomposition rules may turn a path-unique Z-cycle into a non-path-unique one.

The following example shows that an uncontrolled variable replacement may transform a system with a Z-cycle into a system without Z-cycles.

Example 5.9. The system $x \doteq f(y), y \doteq X(s_1), X(s_2) \doteq f(x), x \doteq f(a)$ has a non-trivial Z-cycle. After instantiating it with $x \to f(a)$, there is no Z-cycle anymore: $f(a) \doteq f(y), y \doteq X(s_1), X(s_2) \doteq f(f(a)), x \doteq f(a)$.

Definition 5.10. *A ZCUP S is called* flat, *iff the depth of all surface position is ≤ 1. I.e., all terms are of the following forms: $x, a, f(x_1, \ldots, x_n), X(t)$.*

The strategy for reducing a given ZCUP S depends on the question if S contains a Z-cycle or not. For systems without Z-cycles, the termination arguments become more transparent if we move to flat systems. Note that (repvt) is not applicable to ZCUPs without any Z-cycles.

Definition 5.11. (flatten) *Given a ZCUP without Z-cycles, proper surface positions p are replaced as follows: The equation $s_1[t]_p \doteq s_2$ is replaced by $s_1[x]_p \doteq s_2, x \doteq t$, where t is not a variable. This is done until this replacement is no longer applicable.*

Definition 5.12. (standardization) *A ZCUP is standardized as follows: First use decomposition exhaustively. If the system does not have a Z-cycle after the decomposition, then use flatten exhaustively.*

We now introduce the following four types of ZCUPs.

Definition 5.13. *A ZCUP S is of*

- type 0 *if S is standardized, flat, does not have any Z-cycles, and if there is no function symbol f on the surface of S,*
- type 1 *if S is standardized, flat, does not have any Z-cycles, and if there exists a function symbol f on the surface of S,*
- type 2 *if S is decomposed, contains a Z-cycle and if all ψ-minimal Z-cycles are non-path-unique,*
- type 3 *if S is decomposed, contains a Z-cycle and if all ψ-minimal Z-cycles are path-unique.*

Lemma 5.14. *Standardization is sound and complete. After standardization, the system has one of the types 0–3. If S is the system before and S' the system after standardization, then S has a Z-cycle iff S' has a Z-cycle. Moreover, the measure (μ_1, μ_2) is not increased.*

Usually, decomposition and standardization are used in connection with other operations. At the places where they are used the measure μ will be properly decreased for other reasons.

As described in the previous section, the decision procedure starts, given an arbitrary ZCUP S_0 as input, with a *initial step* which consists only in standardization.

6 Main reduction

In this section we describe the reduction of ZCUPs of type 1–3. ZCUPs of type 0 are always solvable by a substitution that replaces every Z-context variable by $K\ a$ and every first order variable x by a, hence no further treatment is required.

6.1 Reduction of problems of type 1

Let S denote a problem of type 1, with set of variables $\mathcal{V}_S$. Let the relations "$\sim_1$" and "$>_1$" on $\mathcal{V}_S$ be defined as follows: If $x \doteq y \in S$, then $x \sim_1 y$; if $x \doteq Y(s) \in S$, then $x \sim_1 Y$; if $X(s) \doteq Y(t) \in S$, then $X \sim_1 Y$. If $x \doteq f(y_1, \ldots, y_n) \in S$, then $x >_1 y_i$, and if $X(s) \doteq f(y_1, \ldots, y_n) \in S$, then $X >_1 y_i$.

Let "$\sim$" denote the equivalence relation in $\mathcal{V}_S$ generated by "$\sim_1$". Denote the equivalence class of a variable x, (X) by $[x]_\sim$ or $[X]_\sim$, respectively. For equivalence classes D_1, D_2 of $\mathcal{V}_S/\sim$ define $D_1 \rhd_1 D_2$ if there exist $\mathcal{X}_i \in D_i$ such that $\mathcal{X}_1 >_1 \mathcal{X}_2$. Let "$\rhd$" denote the transitive closure of "$\rhd_1$". Note that the relation "$\rhd$" is an irreflexive partial order on $\mathcal{V}_S/\sim$, if the problem is of type 1.

Definition 6.1. Procedure (Imitation) *Let S be a ZCUP of type 1. Let $x \doteq f(\ldots)$ or $X(s) \doteq f(\ldots)$ be an equation of S where $D := [x]_\sim$ (resp. $D := [X]_\sim$) is maximal wrt "$\rhd$". Then select one of the following possibilities:*

1. *Select one Z-context variable $X' \in D$ and instantiate X' by Id or by $K\ y$ for some new first order variable y. Then standardize the system.*
2. *(a) For every variable $y \in D$ instantiate all occurrences of y by $f(y_1, \ldots, y_n)$, where y_i are new variables. For every Z-context variable $Y \in D$ select some index i and instantiate all occurrences of Y by $f(y_1, \ldots, y_{i-1}, Y', y_{i+1}, \ldots, y_n)$, where y_j and Y' are new.*
   (b) Then standardize the system.

Lemma 6.2. *The imitation rule is sound and complete. If S is a unifiable problem of type 1, then using imitation results in a ZCUP S', such that $\mu(S') < \mu(S)$.*

Proof. A $\rhd$-maximal element with the required property exists, since there are no Z-cycles, there is no occurs-check failure, and the ZCUP is not of type 0. For completeness note that the exponent of periodicity of a given unifier is not increased after adapting it to the new problem. The measure is properly decreased if a Z-context variable is eliminated. In the second case of the rule, after instantiation and decomposition, the number of surface occurrences of the function symbol f is strictly reduced, which is easy to see, since S is flat. If a Z-cycle is introduced, then the second component of the measure is decreased. □

6.2 Reduction of problems of type 2

Let S denote a problem of type 2. Recall that S is decomposed and has a ψ-minimal Z-cycle L that is non-path-unique. We may assume that L has the form $X_1(s_1) \doteq t_1, \ldots, X_h(s_h) \doteq t_h$.

As in Section 5, C_i denotes the relevant subcontext of t_i for $X_{(i+1) \bmod^* h}$.

Definition 6.3. Procedure (solve-ambiguous-Z-cycle). *The input is the ZCUP S of type 2 with a ψ-minimal Z-cycle L as described above. Select one of the following possibilities:*

1. *Select one of the variables X_i, for some $1 \leq i < h$, and instantiate X_i either with Id or with $K\ x$ where x is new. Then standardize the resulting ZCUP.*
2. *Let $X_j(s_j) \doteq t_j$ be an equation in L such that X_{j+1} occurs at least twice on the surface of $t_j = f(t_{j,1}, \ldots, t_{j,m})$ and the main depth of the relevant context C_j is chosen minimal in L. Now apply the following steps:*
   (a) *Select an index $r \in \{1, \ldots, m\}$. In the special situation where $h = 1$, the selection of r is subject to the following condition: all surface occurrences of X_1 in $f(t_{1,1}, \ldots, t_{1,m})$ 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 $f(x_1, \ldots, x_{r-1}, X'_j(.), x_{r+1}, \ldots, x_m)$, where x_i and X'_j are fresh.*
   (c) *Apply rule (decomp) to the equation that is obtained from the equation $X_j(s_j) \doteq t_j$ by step (2b).*
   (d) *Apply (repvv) or (repvt) to the new equations $x_i \doteq t_{j,i}$ $(1 \leq i \leq m, i \neq r)$ that are obtained from the previous step.*
   (e) *Then standardize the resulting ZCUP.*

Note that the steps (2.2c) and (2.2d) are necessary to control the decomposition to assure in the case where there is a Z-cycle of length > 1, that after application of the rule, there is a shorter Z-cycle.

Lemma 6.4. *Let S be as above. Application of the procedure (solve-ambiguous-Z-cycle) is sound and complete, its output is a ZCUP S' of type 0–3, and $\mu(S') < \mu(S)$.*

Proof. See Appendix A.1.

6.3 Reduction of problems of type 3

Now we consider the case where S is decomposed, contains a Z-cycle, and where each ψ-minimal Z-cycle is path-unique. Let L denote a ψ-minimal Z-cycle of S. L can be represented in the form $X_1(s_1) \doteq C_1(X_2(t_1)), \ldots, X_h(s_h) \doteq C_h(X_1(t_h))$. We first describe the reduction in the situation where L contains at least two non-trivial contexts C_j and $C_{j'}$. In particular we have $h \geq 2$. We consider the following sub-procedure, which can be applied to a system only through the procedure (shuffle*).

Definition 6.5. Subprocedure (shuffle) *Let L be a ψ-minimal path-unique Z-cycle of length $h \geq 2$. Let j be an index such that C_j is not trivial. Select one of the following possibilities.*

1. *Select some i and replace X_i by Id or by $K\ x$, where x is new. Then standardize the resulting ZCUP.*

2. *Let $C_j(t_j)$ have the form $f(t_{j,1}, \ldots, t_{j,k-1}, t_{j,k}[X_{j+1}], t_{j,k+1}, \ldots, t_{j,m})$.*
   (a) *Select an index $1 \leq r \leq m$.*
   (b) *Instantiate X_j by $f(x_1, \ldots, x_{r-1}, X'_j(.), x_{r+1}, \ldots, x_m)$, where the x_i, X'_j are new.*
   (c) *Apply (decomp) to the instantiation instance of equation j of L.*
   (d) *Apply (repvt) or (repvv) to all equations added by the last step.*
   (e) *Then standardize the resulting ZCUP.*

Lemma 6.6. *The rule (shuffle) is sound and complete*

Lemma 6.7. *Let S be a ZCOUP and L be a path-unique, ψ-minimal Z-cycle of length $h > 1$. Let S' be obtained from S by an application of the procedure (shuffle). Either we have $\mu(S') < \mu(S)$, or S' is decomposed, contains the same number of Z-context variables, and contains a path-unique Z-cycle L' of length h such that the relevant contexts $C'_0, \ldots, C'_h$ have main depths corresponding to the contexts $C_0, \ldots, C_h$ of L up to positions j and $j-1$. We have $|C'_{j-1}| = |C_{j-1}|+1$ and $|C'_j| = |C_j| - 1$.*

Proof. See Appendix A.2.

Definition 6.8. Procedure (shuffle*) *Let L be a ψ-minimal path-unique Z-cycle of length h with at least two non-trivial relevant contexts C_j and $C_{j'}$. Let k be the number of non-trivial relevant contexts in L. Let $\bar{\mu}$ be μ before (shuffle*) starts.*

Then iterate (shuffle) as follows:

1. *First select an index j in the Z-cycle, such that C_j is non-trivial.*
2. *Apply (shuffle) for index j.*
3. *If μ is strictly smaller than $\bar{\mu}$, then stop.*
4. *Otherwise, let L' be the Z-cycle obtained from L. If C'_j is nontrivial, then go to 2 using the same index.*
   If C'_j is trivial, then go to 2 using the index $j - 1$ $\texttt{mod}*$ h.

Note that μ may be temporarily increased by a step within (shuffle*), if there are no relevant contexts with neighbouring indices. The goal of (shuffle*) is to decrease the number of relevant contexts in the smallest Z-cycle.

Lemma 6.9. *Let S be a problem of type 3 that contains a ψ-minimal, path-unique Z-cycle L of length h with at least two non-trivial relevant contexts C_j, $C_{j'}$. Then applying shuffle* is sound, complete, terminates and results in a ZCUP S' of type 0–3 such that $\mu(S') < \mu(S)$.*

Proof. Obviously we may reach, by an iterated application of the procedure (shuffle), a system S' that contains less Z-context variables, a smaller ψ-minimal Z-cycle, or a ψ-minimal path-unique Z-cycle of the same length where the number of non-trivial context C_j is strictly reduced. □

Finally we describe the reduction of problems L of type 3 in the situation where the ψ-minimal path-unique Z-cycle L contains just one non-trivial relevant context C_j. We may assume that $j = h$ and L has 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 nontrivial.

In order to avoid instantiating a Z-context variable X_i by a context that contains X_i, the following construction is defined:
Let C be a context. The *skeleton context* B of C is constructed as follows: If $C = f(t_1, \ldots, t_{i-1}, C', t_{i+1}, \ldots, t_m)$, then $B = f(x_1, \ldots, x_{i-1}, B', x_{i+1}, \ldots, x_m)$, where B' is the skeleton context of C' and the variables x_i are fresh ones. For the empty context, the skeleton is the empty context. For example, the skeleton context of $f(g(a,b), g(b,[\cdot],c))$ is $f(x, g(y,[\cdot],z))$.

Definition 6.10. Procedure (solve-compressed-cycle) *Input is a ZCUP of type 3, such that a ψ-minimal path-unique Z-cycle L has exactly one non-trivial relevant context C_h. Select one of the following possibilities.*

1. *Select some Z-context variable X_i and replace X_i by Id or by $K\ x$, where x is new. Then standardize the resulting ZCUP.*
2. *Construct the skeleton context B_h from C_h. Select some $e \leq E$ and some (possibly trivial) prefix C of B_h, i.e. $B_h = CC'$ for some C'. Replace each X_i either by $B_h^e C(X_i')$, or by $B_h^e C$, where X_i' is new. For at least one index i, the second case should be selected. Then standardize the resulting ZCUP.*
3. *This selection is only applicable if $h > 1$. Construct the skeleton context B_h from C_h.*
   (a) *Select $e \leq E$ and some (possibly trivial) prefix C of B_h, such that $B_h = CC'$ and C' has a top level function symbol f of arity $n > 1$.*
   (b) *Select for every j with $1 \leq j \leq h$ an index k_j and replace X_j by $B_h^e C f(x_{j,1}, \ldots, x_{j,k_j-1}, X_j'(\cdot), x_{j,k_j+1}, \ldots, x_{j,n})$ with new variables. At least one index k_j should be different from the direction of the hole in B_h.*
   (c) *Decompose the equations obtained from the equations of L by instantiation.*
   (d) *Apply (repvt) or (repvv) to the equations obtained from (c),*
   (e) *Then standardize the resulting ZCUP.*

Lemma 6.11. *The procedure (solve-compressed-cycle) is sound, complete and either strictly reduces the measure μ, or the resulting ZCUP leads to failure by occur-check.*

Proof. see Appendix A.3

Example 6.12. We demonstrate the algorithm for the equation

$$X(Y(a)) \doteq f(Z(X(b)), X(c))$$

The main part of (solve-compressed-cycle) instantiates X by $f(x,\cdot)^e$. This gives for $e = 2$:

$$f(x, f(x, (Y(a)))) \doteq f(Z((f(x,\cdot)^e)(b)), (f(x, f(x, c))))$$

Decomposition yields:

$$x \doteq Z((f(x,\cdot)^e)(b)), Y(a) \doteq f(x,c)$$

Now there are two possibilities for the second equation. We choose $Y := f(Y',c)$. This results in:

$$x \doteq Z((f(x,\cdot)^e)(b)), Y'(a) \doteq x$$

Now the system is of type 0, and hence unifiable by constant term functions.

Example 6.13. For instance, consider the following (connected) Z-cycles:

$$\begin{aligned} X_1(t_1) &\doteq X_2(s_1) \\ X_2(t_2) &\doteq f(X_1(s_2), X_3(s_3)) \\ X_3(t_3) &\doteq X_2(s_4) \end{aligned}$$

Then there are two shortest Z-cycles. The algorithm will select one of them, say the X_1, X_2-cycle, and apply (solve-compressed-cycle). We demonstrate the nontrivial possibilities:

1. Instantiation deviating from the cycle:

   $$X_1 = f(x_1, X_1'(\cdot)), X_2 = f(x_2, X_2'(\cdot))$$

   Result:

   $$\begin{aligned} x_1 &\doteq x_2 \\ X_1'(t_1) &\doteq X_2'(s_1) \\ x_2 &\doteq f(x_1, X_1'(s_2)) \\ X_2'(t_2) &\doteq X_3(s_3) \\ &\dots \end{aligned}$$

   After (repvt)-applications this results in a occurs-check.
2. Another Instantiation deviating from the cycle:

   $$X_1 = f(X_1'(\cdot), x_1), X_2 = f(x_2, X_2'(\cdot))$$

   Result:

   $$\begin{aligned} X_1'(t_1) &\doteq x_2 \\ x_1 &\doteq X_2'(s_1) \\ x_2 &\doteq f(X_1'(s_2), x_1) \\ X_2'(t_2) &\doteq X_3(s_3) \\ &\dots \end{aligned}$$

   After one (repvt), this gives the shorter Z-cycle:

   $$X_1'(t_1) \doteq f(X_1'(s_2), x_1)$$

3. Instantiations following the cycle would usually result in CMSOUPS that are not smaller w.r.t,. μ:

   $$X_1 = f(X_1'(\cdot), x_1), X_2 = f(X_2'(\cdot), x_2)$$

Result:

$$\begin{aligned} X_1'(t_1) &\doteq X_2'(s_1) \\ x_1 &\doteq x_2 \\ X_2'(t_2) &\doteq f(X_1'(s_2), x_1) \\ x_2 &\doteq X_3(s_3) \\ &\dots \end{aligned}$$

However, the rule (solve-compressed-cycle) permits in this case only instantiations that either strictly decrease the number of Z-cycles, or instantiations that simulate a limited number of instanting around the Z-cycle and then deviating, which results in failure or a shorter Z-cycle.

7 Bounded Second Order Unification is $\mathcal{NP}$-hard

Given an instance of the 1-IN-3-SAT problem $p_{\varphi(i,1)} \vee p_{\varphi(i,2)} \vee p_{\varphi(i,3)}, i = 1, \dots, n$, where p_j are the propositional variables, we construct the following bounded second order unification problem:

Let $X_{\varphi(i,j)}, i = 1, \dots, n, j = 1, 2, 3$ be Z-context variables, let g be a unary function symbol and a be a constant. The equations are $X_j(g(a)) \doteq g(X_j(a))$ for every Z-context variable X_j, and $X_{\varphi(i,1)}(X_{\varphi(i,2)}(X_{\varphi(i,3)}(a))) \doteq g(a)$ for $i = 1, \dots, n$. The translation of truth is as follows: p_j is true if the unifier instantiates X_j by $g(\cdot)$, and false otherwise.

The first equation implies that the unifiers perform the instantiations $X_j = g^{m_j}(\cdot)$. The second equation implies that $m_j \in \{0, 1\}$. Moreover, the second equation implies that for every i exactly one of the variables $X_{\varphi(i,1)}, X_{\varphi(i,2)}, X_{\varphi(i,3)}$ is instantiated by $g(\cdot)$. Now it is easy to verify that the 1-IN-3-SAT problem is solvable iff the constructed Z-context unification problem has a unifier.

Theorem 7.1. *Bounded second order unification is $\mathcal{NP}$-hard.*

For monadic second order unification (MSOU) which is second order unification where the signature has only monadic function symbols the same encoding can be used, hence the following is immediate.

Corollary 7.2. *Monadic second order unification is $\mathcal{NP}$-hard.*

8 Monadic SOU is in Σ_2^p

Monadic SOU is second order unification where the signature is restricted to symbols of order 0 or 1. This immediately implies that monadic SOU-problems are also bounded second order unification problem, in particular Z-context unification problems, since there could be at most one hole in the instantiating expressions.

In this section we show that monadic SOU is in Σ_2^p. This is achieved by carefully modifying the procedure for bounded SOU by exploiting sharing and a compression technique for terms. Thus the exponential notational length of words of the form w^n can be avoided.

8.1 Unification Algorithm for Monadic SOU (MSOU)

It is obvious that the following non-deterministic step to remove the first order variables is sound, complete and efficient:

Definition 8.1. Initial step *removing all first order variables:*

> *Replace all first order variables x by $X(a)$, where X is a fresh Z-context variable and a is some constant occurring in the problem.*

A *m-word* may be: $f \mid X \mid w_1 w_2 \mid (\texttt{power}\ w\ n) \mid (\texttt{prefix}\ n\ w) \mid (\texttt{suffix}\ n\ w)$, where w, w_1, w_2 are m-words, f a unary function symbol from the signature, and n is a number.
A *term* is an expression of the form $w(a)$, where w is an m-word, and a is a constant from the signature.

Thus every equation in a monadic second order unification problem (MSOUP) has the form $w_1(s_1) \doteq w_2(s_2)$, where w_i are m-words, and s_i are constants.
Note that it is not possible to remove the constants, since the Z-context variables may also be instantiated by a constant term function.
The datastructure for the compressed MSOU problem (CMSOUP) consists of:

1. term equations $t_1 \doteq t_2$, also called *active equations*,
2. sharing equations of the form $U := w$, where U is a Z-context variable, and
3. m-word equations $w_1 = w_2$.

A substitution σ is a *unifier* of S, iff $\sigma(s) = \sigma(t)$ for all kinds of equations in S.
We omit the prefix and suffix S, if the CMSOUP S is clear from the context.
Given a CMSOUP S, and an m-word w, the expanded word $\texttt{expand}(w)$ has its obvious definition using power, suffix, prefix, and replacement using the sharing equations.
If for an m-word w in S, $\texttt{expand}(w)$ is ground, then w is called a *ground m-word*.
Non-ground Z-context variables are called *unsolved*.

A CMSOUP S should fulfill the following:

i.) There is no cycle in the sharing equations.
ii.) If $w_1 = w_2$ is a m-word equation, then w_1, w_2 are ground m-words.
iii.) In a sharing equation, the right hand side w should be a ground m-word.

These conditions are always satisfied by the CMSOUPs generated in the algorithm (8.3).

Definition 8.2. *Given a CMSOUP S and a ground m-word w, two algorithms are defined.*

- *The* `length` *of a ground m-word w. This algorithm has to memorize the already computed lengths of ground second order variables also during the computation, for example by labeling the datastructure, or by a table.*

– *Extraction (*`extract`*) of the function symbol at index i in a ground word. We assume for simplicity that $1 \leq i \leq$* `length`(w).

$$
\begin{array}{lll}
\texttt{extract}(1, f) & = f & \\
\texttt{extract}(i, w_1 w_2) & = \texttt{extract}(i, w_1) & \text{if } \texttt{length}(w_1) \geq i \\
\texttt{extract}(i, w_1 w_2) & = \texttt{extract}(i - \texttt{length}(w_1), w_2) & \text{if } \texttt{length}(w_1) < i \\
\texttt{extract}(i, w^n) & = \texttt{extract}(i \texttt{ mod* } (\texttt{length}(w)), w) & \\
\texttt{extract}(i, \texttt{suffix } n\ w) & = \texttt{extract}(i + n, w) & \\
\texttt{extract}(i, \texttt{prefix } n\ w) & = \texttt{extract}(i, w) &
\end{array}
$$

Now we can define the specialized unification algorithm for CMSOUP. Since we have to maximize sharing, the (repvt)-rule is not permitted. Z-cycles are defined as for ZCUPs, but only active equations are considered, and the sharing variables are considered as ground.
There are three different types of problems: 0,1,3.

– Type 0: Every term in an active equation starts with an unsolved Z-context variable.
– Type 1: There is no Z-cycle in S, and S is not of type 0.
– Type 2: There is a Z-cycle in S.

Type 2 is not possible, since there are no function symbols f with $ar(f) \geq 2$. Since sharing equations are only formed for ground Z-contexts, every m-word w_i in an active equation can be separated (in linear time) into a ground prefix $w_{i,1}$ and an m-word $r_{i,1}$ that is either empty or starts with an unsolved Z-context variable. In the following rules we assume this separation wr for every term in active equations.

Definition 8.3. *Let the input CMSOUP S be of type 1 or 3. The adapted rules are as follows:*

– *(MSOU-decomposition) The input is $w_1 r_1(a_1) \doteq w_2 r_2(a_2)$. Compute $n_i =$* `length`$(w_i), i = 1, 2$. *We may assume that $0 < n_1 \leq n_2$.*
If $n_1 < n_2$, then replace the input equation by
$r_1(a_1) \doteq (\texttt{suffix } n_1\ w_2) r_2(a_2), w_1 = (\texttt{prefix } n_1\ w_2)$.
If $n_1 = n_2$, then replace the input equation by
$r_1(a_1) \doteq r_2(a_2), w_1 = w_2$.
– *(MSOU-cc-decomposition) The input is $a \doteq a$, where a is a constant. Remove this equation.*
– *(MSOU-guess-trivial) Let X be an unsolved Z-context variable. Then select one of the following two possibilities:*
  - *(MSOU-guess-trivial-Id) Replace X by Id, i.e. remove X from the CMSOUP.*
  - *(MSOU-guess-trivial-Const) This selection is only possible, if X was not generated by a previous (MSOU-guess-trivial-Const).*
  Replace X by $K\ (X'(a))$ where X' is a new Z-context variable and a some first order constant. I.e., replace every m-word $wXw'(a_1)$ by $wX'(a)$, where w is assumed to not contain X.

- *(MSOU-instantiate) Let $\sim, \rhd$ be the relations defined in 6.1 on unsolved Z-context variables active equations. Let S be Z-cycle free and D be a $\sim$-equivalence class maximal wrt $\rhd$.*
  Select an equivalence class D, let $X_{j_i} t_i(s_{i,1}) \doteq w_i r_i(s_{i,2})$, $i = 1, \ldots, n$ be all the equations with $X_{j_i} \in D$, such that there is at least one equationwith a non-trivial w_i.
  Then compute $n_k = \mathtt{length}(w_k)$ and select the shortest w_k among the non-trivial ones.
  Let $I = \{i \mid 1 \leq i \leq n, n_i > 0\}$ and let Z_i be new context variables. Add the sharing equations $Z_i := w_i, i \in I$ and the m-word equations $Z_k = (\mathtt{prefix}\ n_k\ Z_i), i \in I$.
  Replace all $X_i \in D$ at non-surface positions by $Z_k X'_i$, where X'_i is new. Replace the equations $X_{j_i} t_i(s_{i,1}) \doteq w_i r_i(s_{i,2})$ by $X'_{j_i} t_i(s_{i,1}) \doteq (\mathtt{suffix}\ n_k\ Z_i) r_i(s_{i,2})$ for all i with $n_i > n_k$, and by $X'_{j_i} t_i(s_{i,1}) \doteq r_i(s_{i,2})$ for all i with $n_i = n_k$.
- *(MSOU-Z-cycle) Select a Z-cycle of minimal length h. Every equation in the Z-cycle is of the form $X_i t_{i,1}(s_{i,1}) = w_i X_{i+1 \bmod^* h} t_{i,2}(s_{i,2})$, $i = 1, \ldots, h$. Compute $n_i = \mathtt{length}(w_i)$ and let $I = \{i \mid 1 \leq i \leq n, n_i > 0\}$.*
  1. *Select $1 \leq j \leq h$.*
  2. *For all $i \in I$, add equations $Z_i := w_i$ and replace the prefixes w_i in the the Z-cycle equations by Z_i.*
  3. *Let w_z be the m-word $Z_1 \ldots Z_n$, where only for the indices in I, a Z_i is in w_z. Select a number $n \leq E$, and a number $m < \sum(n_i)$ and add $X_1 := (\mathtt{power}\ w_z\ n)(\mathtt{prefix}\ m\ w_z)$.*
- *failure rules: If there is an active equation of the form $a \doteq b$ or $a \doteq f(\ldots)$, where $a \neq b$, then fail.*

Remark 8.4.

- Note that the equation $a \doteq X(b)$ is recognized as solvable by a step (MSOU-guess-trivial-Const), followed by (MSOU-guess-trivial-Id).
- Since first order variables are removed, the occurs-check appears in a different form. For example in the equation $f(X(a)) \doteq X(a)$. This is transformed using (MSOU-Z-cycle), and subsequent (MSOU-decomposition). The failure is detected if the equation $f(\ldots) \doteq a$ is present.

Definition 8.5. *(MSOU-final) If the CMSOUP S is of type 0, i.e., every term in an active equation is of the form $X(\ldots)$, where X is unsolved, then check the word equations. If they hold, then a unifier is found, otherwise fail.*

8.2 Correctness and Complexity of the unification algorithm CMSOU.

Soundness and Completeness of the algorithm follows from the results in section 6.

Lemma 8.6. *Let N be the size of the CMSOUP S. Then* `length` *and* `extract` *can be computed in polynomial time depending on N.*

Proof. Follows from Lemma 4.3 and an analysis of the algorithm `length`. □

Now let's count the maximal number of steps of the CMSOU-unification algorithm: The first observations are that the number of active equations is never increased, that the number of unsolved Z-context variables is not increased, and that the number of occurrences of unsolved Z-context variables is not increased.

Let N_0 be the initial size of the problem and let $N_Z \leq N_0$ be the initial number of Z-context variables. N denotes the current size of S.

The rule (MSOU-Z-cycle) can be applied at most N_Z times, since in every step, the number of unsolved Z-context variables is strictly decreased. The rule can be applied in polynomial time. The size-increase depends only on the number of active equations, and the computed numbers m, n, hence this polynomial. The numbers m, n can be represented in linear space.
The number of (MSOU-guess-trivial) steps is at most N.

The number of (MSOU-instantiate)-steps without intermediate (MSOU-Z-cycle) is at most N_Z, since the number of unsolved Z-context variables not in a maximal $\triangleright$-equivalence class is properly decreased.

We have to estimate the size-increase: The size-increase resulting from adding non-active equations is polynomial in the number of active equations. The size-increase resulting from the replacement of X_i is polynomial in the number of occurrences of unsolved Z-contet variables, which is not increased. The numbers resulting from computing `length`s can be represented in linear space.

The problem S is solved after applying (MSOU-final). The check for equal ground m-words is in co-$\mathcal{NP}$, since we can guess a word equation and an index in the two words, which point to different symbols. The representation of the number is linear in the size of the final problem, since the exponents are all representable as numbers in linear space (see Lemma 4.3 and [SSS98a]). The extraction of the function symbol at this position can be done in polynomial time (Lemma 8.6).

The first part of the algorithm is non-deterministic and can be performed in polynomial time, the last part is a universal check that can be done in polynomial time, too. Hence:

Theorem 8.7. *Monadic SOU is in* Σ_2^p.

9 Conclusion

This paper shows that restricting the number of holes in instantiations of second order variables makes second order unification decidable. This result is consistent with the hypothesis that context unification is decidable. However, the methods and the results seem not helpful in solving the context unification problem. The procedure could be adapted to transform context unification problems into a type-0 form, but there is no decision method known for context unification problems of type 0.

Unfortunately, the decision algorithm is rather complex, it requires at least exponential space, since the procedure (solve-compressed-cycle) may generate an exponential number of copies of a context.

We have given an upper bound Σ_2^p on the complexity of the specialization MSOU. An estimation of the complexity of BSOU is left as future work.

Comparing the upper bound Σ_2^p of monadic SOU with the recently obtained upper bounds NEXPTIME [Pla99a] and even PSPACE [Pla99b] for the word unification problem, this is a hint that monadic SOU is an easier problem than word unification. Since monadic SOU is to Z-context unification as word unification to context unification, this is also a hint that bounded second order unification may be an easier problem than context unification.

10 Acknowledgements

I thank Klaus Schulz for his numerous comments. Thanks also to referees of the paper and to Jordy Levy and Mattieu Villaret.

References

[BS94] Franz Baader and Jörg Siekmann. Unification theory. In D.M. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, pages 41–125. Oxford University Press, 1994.

[Far88] W.A. Farmer. A unification algorithm for second order monadic terms. *Annals of Pure and Applied Logic*, 39:131–174, 1988.

[Far91] W.A. Farmer. Simple second-order languages for which unification is undecidable. *J. Theoretical Computer Science*, 87:173–214, 1991.

[Gol81] W.D. Goldfarb. The undecidability of the second-order unification problem. *Theoretical Computer Science*, 13:225–230, 1981.

[Gut98] C. Gutierrez. Satisfiability of word equations with constants is in exponential space. In *Proceedings FOCS'98*, Palo Alto, California, 1998. IEEE Computer Society Press.

[Hue75] Gerard Huet. A unification algorithm for typed λ-calculus. *Theoretical Computer Science*, 1:27–57, 1975.

[Lev96] Jordi Levy. Linear second order unification. In *Proceedings of the 7th International Conference on Rewriting Techniques and Applications*, volume 1103 of *Lecture Notes in Computer Science*, pages 332–346, 1996.

[Lev98] Jordi Levy. Decidable and undecidable second order unification problems. In *Proceedings of the 9th Int. Conf. on Rewriting Techniques and Applications*, volume 1379 of *Lecture Notes in Computer Science*, pages 47–60, 1998.

[LV98] Jordi Levy and Margus Veanes. On unification problems in restricted second order languages, 1998. to appear in CSL' 98.

[Mak77] G.S. Makanin. The problem of solvability of equations in a free semigroup. *Math. USSR Sbornik*, 32(2):129–198, 1977.

[Pla99a] W. Plandowski. Satisfiability of word equations with constants is in NEXPTIME. In T. Leighton, editor, *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing (STOC'99)*, Atlanta, Georgia, 1999. ACM Press. To appear.

[Pla99b] W. Plandowski. Satisfiability of word equations with constants is in PSPACE, 1999. To appear.

[SG89] Wayne Snyder and Jean Gallier. Higher-order unification revisited: Complete sets of transformations. *J. Symbolic Computation*, 8:101–140, 1989.

[SS94] Manfred Schmidt-Schauß. Unification of stratified second-order terms. Internal Report 12/94, Fachbereich Informatik, J.W. Goethe-Universität Frankfurt, Frankfurt, Germany, 1994.

[SS98] Manfred Schmidt-Schauß. A decision algorithm for distributive unification. *Theoretical Computer Science*, 208:111–148, 1998.

[SSS98a] Manfred Schmidt-Schauß and Klaus U. Schulz. On the exponent of periodicity of minimal solutions of context equations. In *Proceedings of the 9th Int. Conf. on Rewriting Techniques and Applications*, volume 1379 of *Lecture Notes in Computer Science*, pages 61–75, 1998.

[SSS98b] Manfred Schmidt-Schauß and Klaus U. Schulz. Solvability of context equations with two variables is decidable. Technical Report CIS-Report 98-114, Centrum für Informations- und Sprachverarbeitung (CIS), Universität München, 1998.

[SSS99] Manfred Schmidt-Schauß and Klaus U. Schulz. Solvability of context equations with two context variables is decidable. In *Proceedings of the International Conference on Automated Deduction*, Lecture Notes in Computer Science, pages 67–81, 1999.

[Wol93] D.A. Wolfram. *The clausal theories of types*. Number 21 in Cambridge tracts in theoretical computer science. Cambridge University Press, 1993.

Appendix

This appendix contains proofs of soundness, completeness and termination in terms of μ of the main reduction rules.

A.1 Proof of Lemma 6.4

Clearly the output problem is of type 0–3.

To verify completeness, let σ be a unifier of S. If $\sigma(X_i)$ is the identity or a constant term function, for some $1 \leq i \leq h$, then use selection 1. Let us now assume that none of the expressions $\sigma(X_i)$ is the identity or a constant term function $(1 \leq i \leq h)$.

First assume that $h > 1$. Let r denote the first index of the path of $\sigma(X_1)$ to the hole. Obviously r represents a possible choice in selection 2 above, and the system S' that is reached after instantiation and decomposition has a solution σ' with exponent of periodicity $\leq E$.

Now consider the special situation where $h = 1$. Note that $\sigma(X_1)$ is assumed to be non-trivial. Here L has the form $X_1(s_1) \doteq f(t_{1,1}, \ldots, t_{1,m})[X_1]$. We claim that all surface occurrences of X_1 in $f(t_{1,1}, \ldots, t_{1,m})$ are in the unique subterm $t_{1,r}$ where the index r is given by the first index of the path of $\sigma(X_1)$ to the hole. In fact, assume that there exists an index $1 \leq k \leq m, k \neq r$ such that $t_{1,k}$ has a surface occurrence of X_1. The definition of r and the form of L show that $\sigma(t_{1,k})$ is a proper subterm of $\sigma(X_1)$. On the other hand, $\sigma(t_{1,k})$ contains an occurrence of $\sigma(X_1)$. This is a contradiction. We have seen that the index r represents a (the only) possible selection. Obviously the system S' that is reached after instantiation and decomposition has a solution σ' with exponent of periodicity $\leq E$.

Let us now verify $\mu(S') < \mu(S)$. This is obvious if the first selection is used since the number of Z-context variables is properly reduced.

Now assume that the second selection has been chosen. Note that length-minimality implies that the Z-cycle contains only the explicitly indicated surface occurrences of X_i. First consider the case where $h > 1$. Then L contains a subsequence

$$\begin{aligned} X_{j-1}(s_{j-1}) &\doteq t_{j-1}[X_j], \\ X_j(s_j) &\doteq f(t_{j,1}, \ldots, t_{j,m})[X_{j+1}], \\ X_{j+1}(s_{j+1}) &\doteq t_{j+1} \end{aligned}$$

where X_{j+1} has at least two surface occurrences in $t_j = f(t_{j,1}, \ldots, t_{j,m})$ (the first and the third equation may be identical). Instantiation (b) leads to the equations

$$\begin{aligned} X_{j-1}(s'_{j-1}) &\doteq t'_{j-1}[f(x_1, \ldots, x_{r-1}, X'_j(\cdot), x_{r+1}, \ldots, x_m)], \\ f(x_1, \ldots, x_{r-1}, X'_j(s'_j), x_{r+1}, \ldots, x_m) &\doteq f(t'_{j,1}, \ldots, t'_{j,m})[X_{j+1}], \\ X_{j+1}(s'_{j+1}) &\doteq t'_{j+1} \end{aligned}$$

(where terms t' are obtained from t by instantiation). Application of rule (decomp) in Step (c) yields

$$\begin{aligned} X_{j-1}(s'_{j-1}) &\doteq t'_{j-1}[f(x_1,\ldots,x_{r-1},X'_j(\cdot),x_{r+1},\ldots,x_m)],\\ X'_j(s'_j) &\doteq t'_{j,r},\\ X_{j+1}(s'_{j+1}) &\doteq t'_{j+1} \end{aligned}$$

plus the equations $x_i \doteq t'_{j,i}$ for $i \neq r$. Application of (repvt) or (repvv) in Step (d) leads to the problem S' with the three equations

$$\begin{aligned} X_{j-1}(s'_{j-1}) &\doteq t'_{j-1}[f(t'_{j,1},\ldots,t'_{j,r-1},X'_j(\cdot),t'_{j,r+1},\ldots,t'_{j,m})],\\ X'_j(s'_j) &\doteq t'_{j,r},\\ X_{j+1}(s'_{j+1}) &\doteq t'_{j+1} \end{aligned}$$

First assume that there is at least one index $k \neq r$ such that $t_{j,k}$ has a surface occurrence of X_{j+1}. Since $X_{j+1} \neq X_j$ also $t'_{j,k}$ has a surface occurrence of X_{j+1}. This shows that the equations

$$\begin{aligned} X_{j-1}(s'_{j-1}) &\doteq t'_{j-1}[f(t'_{j,1},\ldots,t'_{j,r-1},X'_j(\cdot),t'_{j,r+1},\ldots,t'_{j,m})],\\ X_{j+1}(s'_{j+1}) &\doteq t'_{j+1}. \end{aligned}$$

together with the images of the equations of L with indices $\notin \{j, j+1, j-1\}$ represent a Z-cycle L' of length $h-1$. Note that the conditions for a Z-cycle are satisfied, since $t'_{j-1}[f(t'_{j,1},\ldots,t'_{j,r-1},X'_j(\cdot),t'_{j,r+1},\ldots,t'_{j,m})]$ contains at least one function symbol at the top. When we now fully decompose, we may assume that no failure rule is applied. Decomposition does not increase the ψ-measure of the Z-cycle, and the resulting system S'' contains a Z-cycle L'' such that $\psi(L'') = \psi(L') < \psi(L)$, which shows that $\mu(S'') < \mu(S)$.

In the other case, $t_{j,r}$ – and hence $t'_{j,r}$ – contains at least two surface occurrence of X_{j+1}. The three equations above can be combined with the images of the equations of L with indices $\notin \{j, j+1, j-1\}$ to a new Z-cycle L'. When moving from L to L', the main depth of the relevant context of equation j is decreased. Since $t'_{j,r}$ contains at least two surface occurrence of X_{j+1} the new Z-cycle L' is non-path-unique and this main depth is relevant for the ψ-measure.

It follows that $\psi(L') < \psi(L)$. As in the previous case it follows now that a system S'' is reached such that $\mu(S'') < \mu(S)$.

Consider the special situation where $h = 1$. Here L has the form $X_1(s_1) \doteq f(t_{1,1},\ldots,t_{1,m})[X_1]$. There are at least two surface occurrences of X_1 in $t_1 = f(t_{1,1},\ldots,t_{1,m})$, and all these surface occurrences are in $t_{1,r}$. Instantiation (b) yields $f(x_1,\ldots,x_{r-1},X'_1(s'_1),x_{r+1},\ldots,x_m) \doteq f(t'_{1,1},\ldots,t'_{1,m})[f(x_1,\ldots,x_{r-1},X'_1,x_{r+1},\ldots,x_m)]$. Step (c) yields the new Z-cycle L' with the equation $X'_1(s'_1) \doteq t'_{1,r}$. The main depth of the relevant context of L' is smaller than the main depth of the relevant context for L. Hence $\psi(L') < \psi(L)$. As in the previous cases we see that after Step (d) a problem S'' is reached such that $\mu(S'') < \mu(S)$. □

A.2 Proof of Lemma 6.7

If we use selection 1, then the number of Z-context variables is decreased. Assume now that we choose selection 2.

First consider the case where $r \neq k$. Let the equations with indices $j-1, j, j+1$ of L have the form

$$\begin{aligned} X_{j-1}(s_{j-1}) &\doteq C_{j-1}(X_j(t_{j-1})), \\ X_j(s_j) &\doteq f(t_{j,1}, \ldots, t_{j,k-1}, t_{j,k}[X_{j+1}], t_{j,k+1}, \ldots, t_{j,m}), \\ X_{j+1}(s_{j+1}) &\doteq t \end{aligned}$$

Note that path-uniqueness and length-minimality imply that the Z-cycle contains only the explicitly indicated surface occurrences of X_i. Instantiation (b) leads to

$$\begin{aligned} X_{j-1}(s'_{j-1}) &\doteq C'_{j-1}(f(x_1, \ldots, x_{r-1}, X'_j(t'_{j-1}), x_{r+1}, \ldots, x_m)), \\ f(x_1, \ldots, x_{r-1}, X'_j(s'_j), x_{r+1}, \ldots, x_m) &\doteq f(t'_{j,1}, \ldots, t'_{j,k-1}, t'_{j,k}[X_{j+1}], t'_{j,k+1}, \ldots, t'_{j,m}), \\ X_{j+1}(s'_{j+1}) &\doteq t' \end{aligned}$$

Decomposition (c) and replacement steps (d) lead to a system with a Z-cycle L' of length $h-1$ with the equations

$$\begin{aligned} X_{j-1}(s'_{j-1}) &\doteq C'_{j-1}(f(t'_{j,1}, \ldots, X'_j(t'_{j-1}), \ldots, t'_{j,k}[X_{j+1}], \ldots, t'_{j,m})), \\ X_{j+1}(s'_{j+1}) &\doteq t' \end{aligned}$$

After decomposition we still have a path-unique Z-cycle of length $h-1$ in the system S' that is reached. Hence $\mu(S') < \mu(S)$.

Now consider the selection $r = k$. In this case, instantiation (b) leads to

$$\begin{aligned} X_{j-1}(s'_{j-1}) &\doteq C'_{j-1}(f(x_1, \ldots, x_{k-1}, X'_j(t'_{j-1}), x_{k+1}, \ldots, x_m)), \\ f(x_1, \ldots, x_{k-1}, X'_j(s'_j), x_{k+1}, \ldots, x_m) &\doteq f(t'_{j,1}, \ldots, t'_{j,k-1}, t'_{j,k}[X_{j+1}], t'_{j,k+1}, \ldots, t'_{j,m}), \\ X_{j+1}(s'_{j+1}) &\doteq t' \end{aligned}$$

Decomposition (c) and replacement steps (d) lead to a system with a variant L' of the Z-cycle L, of length h, where the equations with indices $j-1, j, j+1$ have the form

$$\begin{aligned} X_{j-1}(s'_{j-1}) &\doteq C'_{j-1}(f(\ldots, X'_j(t'_{j-1}), \ldots)), \\ X'_j(s'_j) &\doteq t'_{j,k}[X_{j+1}], \\ X_{j+1}(s'_{j+1}) &\doteq t' \end{aligned}$$

The new Z-cycle L' is path-unique, and it is easy to see that it satisfies the properties mentioned in the lemma.□

A.3 Proof of Lemma 6.11

Soundness is obvious.

For completeness let σ be a unifier of S with an exponent of periodicity that is not greater than E. If σ instantiates some Z-context variable in the Z-cycle L as Id or by a constant function, then use selection 1.

Otherwise, let C_0 be the common prefix of the instantiations $\sigma(X_j), j = 1, \ldots, h$, and of $\sigma(C_h)^E$. If there is some X_j, such that $\sigma(X_j) = C_0$, then the second case can be selected, since C_0 is a prefix of $\sigma(C_h)^E$. Note that for an extension σ' (on new variables) of σ, $\sigma'(B_h) = \sigma(C_h)$.

The remaining case is that C_0 is a proper prefix of all $\sigma(X_j)$.
First we show that this is not possible for $h = 1$: Assume otherwise. Let $C_1 = C_{11}C_{12}$, such that $\sigma(C_1^e C_{11} = C_0$. Let k be an index not in the direction of the hole of $C_{12}C_{11}$. Then $\sigma(X_1) = \sigma(C_1)^e\sigma(C_{11})f(r_1, \ldots, r_k[\cdot], \ldots, r_n)$, and the equation $X_1(s_1) \doteq C_1(X_1(t_1))$ after instantiation looks like

$$\sigma(C_1)^e\sigma(C_{11})f(r_1, \ldots, r_k[\sigma(s_1)], \ldots, r_n) = \sigma(C_1)\sigma(C_1)^e\sigma(C_{11})f(r_1, \ldots, r_k[\sigma(s_1)], \ldots, r_n)$$

This implies that the following equation holds:

$$f(r_1, \ldots, r_k[\sigma(s_1)], \ldots, r_n) = \sigma(C_{11}C_{12})f(r_1, \ldots, r_k[\sigma(s_1)], \ldots, r_n)$$

Let $\sigma(C_{11}C_{12}) = f(a_1, \ldots, a_j[\cdot], \ldots, a_n)$, where $j \neq k$ by assumption. Then by decomposition, we get that r_j is a superterm of $a_j[f(r_1, \ldots, r_k[\sigma(s_1)], \ldots, r_n)]$, which is impossible.

Now we can assume that $h \geq 2$. There is a non-unary function symbol f, such that $\sigma(X_j) = C_0D_j$, and f is the top level function symbol of $D_j, j = 1, \ldots, h$. Then the third part can be selected, where in (b) we choose k_j to be the first number of the path to the hole of D_j. Obviously, at least one k_j is different from the direction of the hole of C_h. The exponent of periodicity of a new constructed unifier is not greater than E.
The measure μ is strictly decreased:

This is obvious in the first two cases, since the number of Z-context variables is strictly decreased. If selection 3 is used we show that either the number of Z-context variables is reduced or a new Z-cycle is generated whose length is strictly shorter than h. Note that we can assume that $h \geq 2$. Moreover, note that the contexts B_h, C, C' do not contain the Z-context variables X_i.

L has the form

$$X_1(s_1) \doteq X_2(t_1), \ldots, X_h(s_h) \doteq C_h(X_1(t_h)).$$

Instantiation (b) yields the equations

$$\begin{aligned}
B_h^e C f(x_{1,1}, \ldots, \underbrace{X_1'(s_1')}_{k_1}, \ldots, x_{1,n}) &\doteq B_h^e C f(x_{2,1}, \ldots, \underbrace{X_2'(t_1')}_{k_2}, \ldots, x_{2,n}) \\
&\ldots \doteq \ldots \\
B_h^e C f(x_{h,1}, \ldots, \underbrace{X_h'(s_h')}_{k_h}, \ldots, x_{h,n}) &\doteq C_h B_h^e C f(x_{1,1}, \ldots, \underbrace{X_1'(t_h')}_{k_1}, \ldots, x_{1,n})
\end{aligned}$$

Decomposition of equations yields (among others) the equations:

$$f(x_{1,1},\ldots,\underbrace{X'_1(s'_1)}_{k_1},\ldots,x_{1,n}) \doteq f(x_{2,1},\ldots,\underbrace{X'_2(t'_1)}_{k_2},\ldots,x_{2,n})$$
$$\ldots \doteq \ldots$$
$$f(x_{h,1},\ldots,\underbrace{X'_h(s'_h)}_{k_h},\ldots,x_{h,n}) \doteq C'Cf(x_{1,1},\ldots,\underbrace{X'_1(t'_h)}_{k_1},\ldots,x_{1,n})$$

Let k be the first number of the main path of C'. Then consider all the equations that result from decomposition of the equations, where only the result at index k is considered. Let C'' be the suffix of $C'C$ in direction k. The following pairs of equations can result: Every equation for $2 \leq j < h$ is of the form

$$s''_{j-1,k} \doteq t''_{j-1,k}$$
$$s''_{j,k} \doteq t''_{j,k},$$

where either $t''_{j-1,k} \equiv s''_{j,k} \equiv x_{j,k}$, or $t''_{j-1,k} \equiv X'_j(t'_{j-1}), s''_{j,k} \equiv X'_j(s'_j)$.

The equation for the index h has two possibilities: either

$$x_{h,k} \doteq C''(f(x_{1,1},\ldots,\underbrace{X'_1(t'_h)}_{k_1},\ldots,x_{1,n}))$$

or

$$X'_h(s'_h) \doteq C''(f(x_{1,1},\ldots,\underbrace{X'_1(t'_h)}_{k_1},\ldots,x_{1,n}))$$

There are two possibilities: Either there are only variable equations. In this case there is a fail due to occurs-check for the equations at index k.

The other case is that there is at least one equation with $X'_j(\ldots)$. Then at index k, the chain of equations is a Z-cycle of length h. Moreover, there is at least one variable in the chain, since there is some j, such that $k_j \neq k$. Using (repvt), this results in a Z-cycle of length $\leq h-1$.

□