ELSEVIER

Theoretical Computer Science 275 (2002) 463–479

Theoretical Computer Science

www.elsevier.com/locate/tcs

$\forall\exists^5$-equational theory of context unification is undecidable ☆

Sergei Vorobyov *

Computing Science Department, Uppsala University, Box 337, SE-751 05 Uppsala, Sweden

Received 15 February 2000; revised 15 March 2001; accepted 15 March 2001
Communicated by U. Montanari

Abstract

Context unification is a particular case of *second-order unification* in which all second-order variables are *unary* and only *linear* functions are sought for as solutions. Its decidability is an intriguing *open* problem, with only a very weak known NP-lower bound. We present the simplest (currently known) undecidable quantified fragment of the theory of *context unification* by showing that the set of $\forall\exists^5$-quantified context equations (i.e., sentences of the form $\forall W\,\exists\, U, V, S, G, H\; s = t$) is *undecidable* and, in fact, is *co-recursively enumerable hard* (i.e., every set with recursively enumerable complement is many-one reducible to it). © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Context unification problem; Second-order unification; Word unification; Equational theory; Quantified fragments; Undecidability; Operator algorithms

1. Introduction

The *Context Unification Problem* (CUP for short) is:

- A *generalization* of the celebrated Markov–Löb's problem of solvability of equations in a free semigroup proved *decidable* by Makanin [1]; CUP coincides with this

☆ This paper is a '*significantly*' revised version of a preliminary report, which appeared under the title'$\forall\exists^*$-Equational Theory of Context Unification is Π_1^0-Hard' in the Proceedings of the *23rd* International Symposium on Mathematical Foundations of Computer Science (MFCS'98, Brno, Czech Republic, August 1998), Lecture Notes in Computer Science, vol. 1450, 1998, pp. 597–606 (Vorobyov, 1998). The preliminary version contained a *weaker* result on undecidability of the $\forall\exists^8$-equational theory, obtained by a different method. A sketch of the proof presented here was reported at the *Constraints* in Computational Logic Workshop (CCL'98), Jerusalem, September 1998. Work partially done at the Max-Planck-Institut für Informatik, Saarbrücken, Germany.

* Tel.: +46-18-471-10-55; fax: +46-18-51-1925 WWW: http://www.csd.uu.se/~vorobyov.
E-mail address: vorobyov@csd.uu.se (S. Vorobyov).

0304-3975/02/$ - see front matter © 2002 Elsevier Science B.V. All rights reserved.
PII: S0304-3975(01)00190-6

problem for monadic signatures, consisting of function symbols of arity at most one.

- A *specialization* of the *Second-Order Unification Problem* (SOUP for short), known to be *undecidable* due to Goldfarb [2] and improved by Farmer [3]. CUP is almost SOUP, but with *only unary function variables* allowed and solutions required to be *linear*, i.e., of the form $\lambda x.t(x)$, where $t(x)$ contains *exactly one occurrence* of x.

Context unification is useful in different areas of Computer Science: term rewriting and constraint solving (satisfiability of one-step rewrite constraints), theorem proving, equational unification (decidability of distributive unification), computational linguistics (semantic underspecification of natural languages); (see [4–7]).

CUP is formally stated as follows:

Given a pair of terms t, t' built as usual from symbols of a signature Σ, first-order variables $\bar{w}$, and unary function variables $\bar{F}$, does there exist an assignment θ of terms to $\bar{w}$ and linear *second-order functions to $\bar{F}$ such that $t\theta = t'\theta$?*

Thus, CUP is a decidability problem for the existentially quantified equations ($\exists^*$-equational theory) of the form

$$\exists \bar{F} \exists \bar{w}\, t = t', \tag{1}$$

where the quantified context variables $\bar{F}$ range over *linear* functions.

Currently the decidability of CUP is an intriguing open problem [4–6]. Most researchers conjecture and hope that CUP is decidable. All the above papers provide some approximations: either prove decidability of particular cases, or settle undecidability of some generalizations, or provide technical results towards decidability of CUP. It is worth noting that only a very weak NP-hardness is currently known as a lower bound for the problem, with a very simple proof by reduction from 1-IN-3-SAT [6].

Presumably, CUP is very hard to settle, both in decidable and undecidable sense. This is because CUP lies between a technically difficult decidable case of equations in free semigroups (Markov-Löb's problem proved decidable by Makanin [1], and the undecidable case of SOUP settled by Goldfarb [2] and reinforced by Farmer [3]). Goldfarb–Farmer's results are also technically quite difficult.

Goldfarb [2] demonstrated that SOUP is undecidable for second-order languages containing at least one function constant of arity $\geqslant 2$ and finitely many *unary* and *ternary* function variables. Later Farmer [3] improved it by showing that SOUP remains undecidable in presence of *unary function variables only* (but, unlike CUP, substitutions looked for are *not required to be linear*; in fact, they are not linear in Farmer's proof). It follows from the result of Makanin [1] that SOUP is decidable when all variable and constant function symbols are unary. Farmer [8] improved this by showing that decidability is preserved if n-ary function variables are allowed in addition to constant function symbols of arity *at most one*. Recently, further refinements of the SOUP undecidability, with restrictions on the numbers of functional variables and their occurrences, were obtained by Levy and Veanes [9].

Thus, CUP represents the only unknown remaining difficult intermediate case between decidable word equations and undecidable SOUP (unary function variables, n-ary

function constants, linear solutions). This explains why the progress on CUP has been quite slow. Indeed, decidability of CUP would considerably improve Makanin's result, whereas undecidability would considerably improve Goldfarb–Farmer's undecidability of SOUP.

In this paper we show that adding just one outermost universal quantifier to context equations (1) leads to the Π_1^0-hard (w.r.t. many-one reducibility) class of formulas, where Π_1^0 is the class of all co-recursively enumerable sets.

For comparison, the following undecidability results were previously known about quantified fragments of context unification, and theories of free semigroups as a particular case. Quine [10] showed that the *full* first-order theory of free semigroups is undecidable (this corresponds to the CUP in unary signatures). Durnev [11] improved it to the undecidability of $\exists\forall\exists^3$-*positive* (without negation, but with $\wedge$ and $\vee$) theory of free semigroups. Marchenkov [12] improved it to the undecidability of $\forall\exists^4$-positive theory of free semigroups. Finally, Durnev [13] improved it even further to undecidability of $\forall\exists^3$-positive theory of free semigroups. Niehren et al. [5] showed undecidability of the $\exists^*\forall^*\exists^*$-theory (when all connectives $\vee$, $\wedge$, $\neg$ are allowed) of context unification. For comparison, we prove a stronger undecidability of the sentences of the form $\forall W\ \exists U, V, S, G, H\ s = t$, i.e., our undecidability result holds for the positive theory with a simpler quantifier prefix and not using any boolean connectives.

As to relation to the undecidability results of the *positive* $\exists\forall\exists^3$-, $\forall\exists^4$-, and $\forall\exists^3$-theories of free semigroups, it should be noticed that all known methods to transform a positive formula of the theory of free semigroups into just one equation require a considerable number of *auxiliary existentially quantified variables* (see, e.g., [14, 15]), depending on the number of disjunctions involved. Thus the undecidability results of Marchenkov [12], Durnev [15] for $\forall\exists^4$- and $\forall\exists^3$-positive theories of free semigroups yield only undecidability of the $\forall\exists^n$-equational theories of free semigroups with a *very large* number n of existentially quantified variables, see Durnev [15].

In this paper we show that the situation with context equations is quite different, and *just two* extra existentially quantified context variables suffice to eliminate all disjunctions. This, together with a direct reduction from the halting problem for operator algorithms to the $\forall\exists^3$-positive fragment, gives the undecidability of the $\forall\exists^5$-*equational theory* of context unification, with a reasonably simple quantifier prefix.

The main result of this paper may now be stated as follows.

Main Theorem. *For every finite signature consisting of* $\geqslant 1$ *binary*, $\geqslant 2$ *unary*, *and* $\geqslant 1$ *nullary function symbols one can construct a* context equation $\mathscr{E}(p, W, U, V, S, G, H)$ *with parameter* p *and context variables* W, U, V, S, G, H *such that the set of true sentences of the form*

$$\forall W\ \exists U, V, S, G, H\ \mathscr{E}(p, W, U, V, S, G, H) \tag{2}$$

is Π_1^0*-hard* (*i.e.*, *every co-r.e. set is many-one reducible to it*), *as* p *ranges over finite terms constructed from unary functions and constants* (*i.e.*, *finite words*).

It follows, in particular, that the $\forall\exists^5$-equational theory (without $\wedge$, $\vee$, $\neg$) of context unification is *undecidable*. This improves over the author's previous result on undecidability of the $\forall\exists^8$-equational theory of context unification [16, 17], as well as over the undecidability of the $\exists^*\forall^*\exists^*$-fragment due to Niehren et al. [5].

Remark. The restrictions on the signature in the statement of the Main Theorem are essential in the following sense. If all function symbols are at most unary, then we are in the case of free semigroups discussed above, and the best known result is due to Durnev [15] for $\forall\exists^n$-quantified equations with very large n. If, instead of a binary, one only has a $k>2$-ary symbol h, the existential quantifier prefix becomes more complicated. This is because one needs k variables to say 'X contains h' by $\exists U_1,\ldots,U_k\, h(U_1,\ldots,U_k)$; see Section 3. The requirement about presence of constants is technical: to speak about the validity of $\forall\exists^5$-sentences we need a model (class of models). To simplify matters we use the standard Herbrand universe of closed (ground) terms, and the presence of a constant is necessary for nonemptiness of this universe (we could have used the class of all freely generated models instead).

Warning: We would like to stress that in this paper we do not intend to improve the undecidability results of Marchenkov [12] and Durnev [13] for $\forall\exists^4$- and $\forall\exists^3$-*positive theories of free semigroups* as to *minimizing* the number of existential quantifiers. However, we do get an improvement (in a different framework of context unification, of course) as to simplicity of the existential prefix in the undecidable $\forall\exists^*$-*equational theory* of context unification. As we mentioned above, all known methods of eliminating disjunctions from positive formulas in free semigroups use a considerable number of auxiliary existentially quantified variables, proportional to the number of disjunctions to be eliminated [15]. We show that in context unification just two extra variables are enough. We do not claim that the quantifier prefix we obtain is minimal. We rather tried to keep proofs intuitive and transparent.

One more note on relation to word equations: As soon as CUP is closely related to Markov–Löb's problem on solvability of word equations, several further remarks on its complexity/decidability are in order. The only known nontrivial lower bound for word equations is NP. All known decision procedures for word equations (e.g. [18, 19]) are modifications and improvements of Makanin [1], one of the most complicated and subtle algorithms in Computer Science, with the key idea (bounding the exponent of periodicity) remaining always the same. Complexity analysis of Makanin's algorithm has known a series of improvements over the time: Makanin himself did not give any complexity bounds, Jaffar [18] gave a 4-DEXPTIME upper bound, Kościelski and Pacholski [20] improved it to 3-NEXPTIME. Recently Gutiérrez [21] obtained the EXPSPACE upper bound, and as the last achievement. Plandowski [22] improved it to the PSPACE upper bound (using an algorithm different from Makanin's). Recently Schmidt-Schauß and Schulz [6] tried to generalize Makanin's algorithm to context unification. They succeeded to implement the easier part of Makanin's program: to show that whenever a context equation has a solution, it has a solution of bounded exponent

of periodicity, computable from the size of equation. The second, more complicated, step would consist in developing *generalized context equations* and their transformations, similar to Makanin's, and demonstrating that the number of boundary equations imposes a lower bound on the exponent of periodicity. If successful, both ideas together will prove decidability of the CUP (again along Makanin's ideas). Currently it is unclear whether this program is achievable.

Paper outline. After Section 2 with preliminaries, in Section 3 we prove undecidability of the positive $\forall\exists^3$-theory of context unification. Section 4 contains a technical development preparing disjunction elimination in Section 5.

2. Preliminaries

Context unification: *Let* Σ be a fixed finite signature with each symbol assigned a fixed non-negative arity, *containing at least one constant* symbol ε. Let X be an infinite set of first-order variables and $\mathscr{F} = \{F, G, \ldots\}$ be an infinite set of *function variables of arity one*, also called *context variables.*

Definition 2.1 (*Terms*). The set $\mathscr{T}(\Sigma, X)$ of terms of signature Σ with variables from X is defined as usual: variables from X are terms and for $f \in \Sigma$ of arity $n \geqslant 0$ an expression $f(t_1, \ldots, t_n)$ is a term whenever the t_i's are terms.

We assume all the standard definitions and conventions concerning λ-notation, like β-reduction, normalization, substitutions, etc. Substitutions will be denoted by Greek letters Θ, Υ.

Definition 2.2 (*Contexts*). A *context* is an expression of the form $\lambda x.t(x)$, where $t(x) \in \mathscr{T}(\Sigma, \{x\})$ contains *exactly one* occurrence of the variable x. A context with β-normal form $\lambda x.x$ is called *empty*.

Remark. Note that the '*exactly one*' requirement, absent from the definition of SOUP, is a characteristic feature of CUP, distinguishing it from SOUP.

Definition 2.3 (*Context terms*). Context terms are defined inductively: if $\phi \in \Sigma \cup \mathscr{F}$ is n-ary and $t_1, \ldots, t_n$ are context terms, then $\phi(t_1, \ldots, t_n)$ is also a context term.

Notational conventions: To simplify notation, let us agree, in writing terms, contexts, and context terms with unary function symbols and unary function variables, to omit parentheses and the constant ε (sometimes).

Thus, for example, 01, $S0$, $0S$ abbreviate, respectively, $0(1(\varepsilon))$, $S(0(\varepsilon))$, $0(S(\varepsilon))$. As usual, 0^p and S^p for a natural p mean $\underbrace{0 \ldots 0}_{p \text{ times}}$ and $\underbrace{S \ldots S}_{p \text{ times}}$ respectively. Such abbreviations will be always transparent, improve readability, and the strict bracket structure can be

always unambiguously restored. Context variables will always be denoted by capital Latin letters. We reserve f to denote a binary function symbol.

Definition 2.4 (*CUP*, *context equations*). An instance of CUP, also called a *context equation* (CE for short), is an expression $\tau_1 \stackrel{?}{=} \tau_2$, where τ_1, τ_2 are context terms. A *solution* to a CE $\tau_1 \stackrel{?}{=} \tau_2$ is a substitution Θ of first-order terms for first-order variables and contexts for context variables such that $\tau_1\Theta \equiv_\beta \tau_2\Theta$ (equality of substitutional instances modulo the usual β-reduction).

Remark. The requirement of replacing functional variables with *contexts*, as opposed to arbitrary second-order λ-terms, distinguishes CUP from SOUP. When this requirement is dropped, the problem becomes *undecidable* [3].

3. Undecidability of the positive $\forall\exists^3$-theory of context unification

In this section we prove that the positive $\forall\exists^3$-theory of context unification is undecidable in signatures $\Sigma_1 = \{\varepsilon, 0(\_), 1(\_)\}$ and $\Sigma_2 = \{\varepsilon, 0(\_), 1(\_), f(\_,\_)\}$, where ε is nullary (denotes the empty word), $0(\_)$, $1(\_)$ are unary (denote letters 0 and 1), and $f(\_,\_)$ is binary. Our proof easily generalizes to other finite signatures containing Σ_1, but the number n of existentially quantified variables in the prefix $\forall\exists^n$ increases when dealing with function symbols of arity > 2.

The proof in this section follows the ideas and reductions from Marchenkov [12], Durnev [15]. However, our proof deals with a different framework of function symbols of arity $\geqslant 2$. Additionally, we use a simpler formula format, which will allow us to get rid of disjunctions more easily, by using just two auxiliary existentially quantified variables, and to prove undecidability of the *equational* $\forall\exists^5$-theory of context unification, i.e., of formulas of the form $\forall\exists^5 s = t$ in Section 5 (as contrasted with $\forall\exists^n$ for very large n undecidability for free semigroups; see Introduction).

Theorem 3.1. *The positive $\forall\exists^3$-theory of context unification is undecidable in signatures $\Sigma_1 = \{\varepsilon, 0(\_), 1(\_)\}$, $\Sigma_2 = \{\varepsilon, 0(\_), 1(\_), f(\_,\_)\}$, as well as in any finite signature extending Σ_1 with function symbols of arity $\leqslant 2$.*

More generally, let Σ be a finite extension of Σ_1 with function symbols of arity at most k. Then the positive $\forall\exists^{\max(3,k+1)}$-theory of context unification in Σ is undecidable.

Proof. For simplicity, we work with a finite signature Σ extending Σ_2 with finitely many unary symbols $a_2, \ldots, a_n$. It will be clear (and we will add necessary comments) how the proof extends to other signatures mentioned in the statement of the theorem. Proving the claim for the extension of Σ_1 with a binary function symbol is necessary for establishing the Main Theorem by eliminating disjunctions in Section 5.

Instead of the traditional scheme of proving undecidability by reduction from the halting problem for Turing machines, we will use a more convenient (for our purposes) reduction from the undecidable problem of (*non-*) *applicability of operator algorithms* defined shortly.

As in Marchenkov [12], Durnev [15], let A be an operator algorithm (*fixed* in the sequel) with a *nonrecursive*, even *r.e.-complete* or Σ_1^0-complete, domain $dom(A)$. The existence of such algorithms is proved, e.g., in Mal'cev [23], and is briefly sketched below. Recall that an *operator algorithm* consists of m commands numbered consecutively from 1 to m, with the unique *Stop* command numbered m. The other commands have one of the following three forms:

- $[\times 2]$ — multiply a given number by 2 and proceed to the next command,
- $[\times 3]$ — multiply a given number by 3 and proceed to the next command,
- $[:6; j]$ — if a given number is a multiple of 6, then divide it by 6 and proceed to the command numbered j $(1 \leqslant j \leqslant m)$; otherwise do not change the number and proceed to the next command.

The algorithm A is *applicable* to a given input natural number $x_0 \geqslant 0$ iff there exists a sequence, called a *correct run*

$$(x_0, i_0), (x_1, i_1), (x_2, i_2), \ldots, (x_t, i_t) \tag{3}$$

such that $i_0 = 1$, $i_t = m$ (the number of *Stop*), and for all $1 \leqslant s \leqslant t$ an application of the command number i_{s-1} to x_{s-1} gives x_s and next command number i_s. Additionally, $i_s \neq m$ for $1 \leqslant s < t$. Informally, A is applicable to x_0 if there exists a correct terminating run of A starting from x_0.

To make the paper self-contained, we sketch a representation of an arbitrary *two-register machine* (a well-known universal computational formalism; see Minsky [24]) by means of an operator algorithm. Recall that a two-register machine has two registers (left and right) and the commands *Stop*, AL, AR (adding one to the contents of the left or right register, respectively), $SL\,j$, $SR\,j$ (subtract one from the contents of the left/right register, if it is positive, and go to the command number j; otherwise proceed to the next command).

We represent the contents $\langle p, q \rangle$ of the two registers as the number $2^p \cdot 3^q$. The commands representing the AL and AR commands are, respectively $[\times 2]$ and $[\times 3]$. Representing the conditional register subtraction commands by commands of an operator algorithm is only slightly more complicated. A command $SL\,j$ is represented by a sequence of four operator commands

$$[\times 3],\ [:6, j'],\ [\times 2],\ [:6, k],$$

where j' is the number of operator command starting the representation of the two-register machine command numbered j (as we use more operator commands in modeling, the systematic renumbering is necessary), and k is the number of command following the sequence. It is easy to see that the sequence above transforms $2^p \cdot 3^q$ first to $2^p \cdot 3^{q+1}$, then to $2^{p-1} \cdot 3^q$ (if $p > 0$) and proceeds to j'; if, on the other hand, $p = 0$ then $2^p \cdot 3^{q+1}$ is transformed to $2^{p+1} \cdot 3^{q+1}$ and then to $2^p \cdot 3^q$ with control passing

to the next command numbered k following the sequence. As a result, we can adequately model the behavior of the conditional subtracting command and an arbitrary two-register machine. Therefore, all standard recursion-theoretic results (like existence of algorithms with complete r.e.-domains needed for our purposes) follow for operator algorithms. The choice of operator algorithms (in favor of two-register machines) for our proof is motivated by the fact that a state of an operator algorithm (just one number) is easier to represent and analyze (for divisibility) as a word (a term built of unary function symbols) than a state of a two-register machine (two numbers).

Let us represent a sequence (3) representing a correct run of the operator algorithm A (fixed with a complete r.e.-domain) by the following term [1]

$$1\ 0^{x_0+1}1^{i_0+\delta_0 m}0^{x_1+1}1^{i_1+\delta_1 m}0^{x_2+1}1^{i_2+\delta_2 m}\ \ldots\ 0^{x_t+1}1^{m+\delta_t m}, \tag{4}$$

where $\delta_i = 0$ if 6 divides x_i and $\delta_i = 1$ otherwise (for $i = 0,\ldots,t$). Thus (with the exception of the first 1) the term (4) represents the sequence of states of A as alternating groups of zeros (representing the counter contents) and ones (representing the command counter). The role of δ_i's consists in encoding whether the corresponding number is divisible by 6. If it is then the command counter i is encoded as 1^i, otherwise i is encoded as 1^{m+i}. This idea due to Durnev allows for writing easier formulas with fewer quantifiers to express *incorrectness* of every ground term representing a potential run of A; see below.

For a natural k, denote by $r \equiv r_k[W]$ the term with one context variable W: [2]

1. $r \equiv 1\ 0^{k+1}10W(\varepsilon)$, if k is divisible by 6,
2. $r \equiv 1\ 0^{k+1}1^{1+m}0W(\varepsilon)$, if k is not divisible by 6.

Informally, $r_k[W]$ top-level matches with any run (correct or incorrect) starting with the input number k.

Analogously to [15], let us construct a *positive* [3] $\forall\exists^3$-sentence of the theory of context unification

$$\forall W \exists U,V,S\ \Phi(r_k[W],U,V,S), \tag{5}$$

which asserts that the algorithm A is *inapplicable* to k. The latter means that there are no ground terms representing a correct run of A starting with k, or, equivalently, every ground term represents an *incorrect* run, violating some correctness conditions.

Consequently, our reduction proceeds from the complement of the r.e.-complete set $\overline{dom(A)}$, thus proving Π_1^0-hardness of the positive $\forall\exists^3$-theory of CU, and later in Section 5 of the equational $\forall\exists^3$-theory of CU, according to the following scheme:

$$\Pi_1^0 \equiv \overline{\Sigma_1^0} \leqslant_m \overline{dom(A)} \leqslant_m \text{ positive } \forall\exists^3\text{-theory of CU}$$

[1] Similar to the word from Durnev [15], but starting with 1; this allows us to write more simple, intuitive, and short formulas.

[2] We could have used a first-order variable w and write w instead of $W(\varepsilon)$.

[3] I.e., built of conjunctions, disjunctions, and equalities.

(Here: (1) $dom(A)$ is r.e.-complete, or Σ_1^0-complete, by assumption, (2) consequently, $\overline{dom(A)}$ is co-r.e.-complete, or Π_1^0-complete, (3) $\leqslant_m$ is the usual many-one reducibility; see Rogers [25], for details.)

Clearly, it is enough to construct the formula $\exists S, U, V\ \Phi(r[W], U, V, S)$ asserting that $r \equiv r_k[W]$ does not have form (4) for every possible *ground* substitution for W. Let Φ be the disjunction of the following formulas enumerated in 1–8 below.

1. r contains an occurrence of a unary function symbol a_i (for some $3 \leqslant i \leqslant n$)

   $$\bigvee_{i=2}^{n} r = Ua_iV.$$

   (This is unneeded in the case of signatures Σ_1, Σ_2 containing just two unary symbols 0 and 1.)
2. r contains an occurrence of f (unneeded in the case of Σ_1):

   $$r = U(f(V(\varepsilon), S(\varepsilon))).$$

   (Clearly, we would need *more variables*, hence a longer existential quantifier prefix, for f of *higher arity*. For example, if it were ternary, we would need to write $r = U(f(V(\varepsilon), S(\varepsilon), Q(\varepsilon)))$.)
3. r does not end with 01^m, nor with 01^{m+m}:

   $$\bigvee_{1 \leqslant i < m} r = U01^i \ \vee \bigvee_{1 \leqslant i < m} r = U01^{i+m} \ \vee \ r = U1^{2m+1}.$$

4. r contains 1^{2m+1} (note that it subsumes the last disjunct in the preceding formula, which therefore may be omitted)

   $$r = U1^{2m+1}V.$$

5. r contains 1^m or 1^{2m} not in the end:

   $$r = U01^m0V \vee r = U01^{2m}0V.$$

   Note: that if r *satisfies* none of the preceding formulas, then it has form

   $$r \equiv 1\,0^{x_0+1}1^{j_0}0^{x_1+1}1^{j_1}\ldots 0^{x_t+1}1^{j_t}$$

   for some natural $x_0, x_1, \ldots, x_t$, $j_0, j_1, \ldots, j_t$ such that $x_0 = k$, $1 \leqslant j_l \leqslant 2m$ $(0 \leqslant l \leqslant t)$, j_t is either m or $2m$, j_l is different from m and $2m$ for $l < t$, j_0 is either 1 (if 6 divides $k = x_0$) or $1 + m$ (if 6 does not divide $k = x_0$).
6. Recall that if 6 divides x_l, then $1 \leqslant j_l \leqslant m$; otherwise, $m + 1 \leqslant j_l \leqslant 2m$. This condition is violated by the following formula:[4]

   $$0S = S0 \ \wedge \ \Big(r = U10S^61^{m+m} \ \vee \ \bigvee_{1 \leqslant p \leqslant 5} r = U100^pS^61^m \ \vee$$

[4] Clearly, any solution for the auxiliary conjunct $0S = S0$ here is of the form $\lambda x.0^k(x)$. It is precisely the role of this conjunct to guarantee that S is substituted by a sequence of zeros.

$$\bigvee_{1\leqslant i<m} r = U10S^6 1^{i+m} 0V \ \vee \bigvee_{1\leqslant p\leqslant 5,\ 1\leqslant i<m} r = U100^p S^6 1^i 0V \Bigg).$$

7. For every i, which is a number of the command $[\times d]$, where $d = 2, 3$, write a formula saying that r contains an incorrect application of this command: either the result is incorrect, or the number of the next command is incorrect. Such a formula is written in the form $0S = S0 \wedge (\Psi_1 \vee \Psi_2)$, where Ψ_1 says that the result is incorrect, and Ψ_2 says the result is correct, but the next command number is computed incorrectly.
   The formula Ψ_1 is as follows:

$$\bigvee_{p=0,1;1\leqslant q<d} r = U01^{i+pm} 0S^d 0^q 1V \ \vee \ \bigvee_{p=0,1} r = U00S1^{i+pm} 0S^d 1V \ \vee$$
$$\bigvee_{p=0,1} r = U1S01^{i+pm} 0S^d 0V$$

   (the first $\bigvee$ says that the result is not divisible by d; the second that the result is too small; third that the result is too large).
   The formula Ψ_2 is as follows:

$$\bigvee_{p=0,1;q=0,1;1\leqslant j\leqslant m;j\neq i+1} \begin{pmatrix} r = U10S1^{i+pm} 0S^d 1^{j+qm} 0V \vee \\ r = U10S1^{i+pm} 0S^d 1^{j+qm} \end{pmatrix}.$$

8. For every i, which is a number of the command $[: 6; j]$, write a formula saying that r contains an incorrect application of this command: either the result is incorrect, or the number of the next command is incorrect.
   This formula has form $0S = S0 \wedge (\Psi_1 \vee \Psi_2 \vee \Psi_3 \vee \Psi_4)$, where:
   (a) Ψ_1 says that although the number is divisible by 6, the division result is computed incorrectly, too large or too small, respectively

$$r = U10S^6 1^i 00SV \vee r = U00S^6 1^i 0S1V;$$

   (b) Ψ_2 says that although the number is not divisible by 6[5] (and therefore should not be changed), it increases or decreases, respectively

$$r = U1S1^{i+m} 0SV \vee r = U0S1^{i+m} S1V;$$

   (c) Ψ_3 says that the next command number is incorrect (should be j when applied to a multiple of 6)

$$\bigvee_{q\neq j;\, p=0,1} (r = U01^i S1^{q+pm} \vee r = U01^i S1^{q+pm} 0V);$$

   (d) Ψ_4 says that the next command number is incorrect (should be $i+1$ when applied to a number not divisible by 6)

[5] Recall that non-divisibility is encoded into the command number representation 1^{i+m}, and S is composed of zeros.

$$\bigvee_{q\neq i+1}(r = U01^{i+m}S1^{q+m} \vee r = U01^{i+m}S1^{q+m}0V).$$

This finishes the construction of the sentence (5). The claim of Theorem 3.1 now follows from the assumption that the operator algorithm A has a complete r.e.-domain. Since the sentences (5) assert non-applicability of A to input natural numbers, the set of all true such sentences forms a co-r.e.-hard set.

4. Equalizing left-hand sides

The proof of Theorem 3.1 shows undecidability of the set of positive $\forall\exists^3$-sentences of the theory of context unification of the form

$$\forall W\exists U,V,S\left(\bigvee_{i=0}^{N'} r = t_i \vee \left[0(S(\varepsilon)) = S(0(\varepsilon)) \wedge \left(\bigvee_{i=N'+1}^{N} r = t_i\right)\right]\right),$$

where $r \equiv r[W] \equiv 10^{k+1}10W(\varepsilon)$ or $r \equiv r[W] \equiv 10^{k+1}1^{1+m}0W(\varepsilon)$ (depending on whether 6 divides k; see (4)), and the terms t_i are explicitly enumerated in the proof of Theorem 3.1.

Our aim in this section consists in transforming the last formula into an equivalent form (6) below, with *disjunctive* matrix and every equation containing the *same* term on the left-hand side. This will allow us to eliminate disjunctions in the next section more easily.

Rewrite the last formula in equivalent form conjoining the vacuously true $0(S(\varepsilon)) = 0(S(\varepsilon))$ to the first $N'+1$ disjuncts:

$$\forall W\exists U,V,S\left[\bigvee_{i=0}^{N'}(r = t_i \wedge 0(S(\varepsilon)) = 0(S(\varepsilon))) \vee \bigvee_{i=N'+1}^{N}(r = t_i \wedge 0(S(\varepsilon)) = S(0(\varepsilon)))\right].$$

Rewrite further the last formula in equivalent form, by using the simple fact that $s_1 = u_1 \wedge s_2 = u_2$ iff $f(s_1,s_2) = f(u_1,u_2)$, as follows:[6]

$$\forall W\exists U,V,S\left[\bigvee_{i=0}^{N'} f(r,0(S(\varepsilon))) = f(t_i,0(S(\varepsilon))) \vee \bigvee_{i=N'+1}^{N} f(r,0(S(\varepsilon))) = f(t_i,S(0(\varepsilon)))\right].$$

[6] Although we use a binary function symbol f here, it is *inessential*. In fact, there is a well known trick to do the same without binary symbols, since $s = s' \wedge t = t'$ is equivalent to $satsbt = s'at's'bt'$, where a, b are different function symbols of arity 1. However, the construction in the following section seems crucially dependent on the use of a function symbol of arity $\geqslant 2$.

Now, all left-hand sides in the equalities in disjunctions of the last formula are the *same*, as desired. We can rewrite it as

$$\forall W \exists U, V, S \left(\bigvee_{i=0}^{N} f(r, 0(S(\varepsilon))) = t'_i \right), \tag{6}$$

where t'_i are constructed (as shown in the preceding formula) from t_i explicitly enumerated in the proof of Theorem 3.1. It is important to note here (for the further development) that:

1. every t'_i contains *at least one occurrence* of S,
2. the number of disjuncts $N+1$ in (6) is *sufficiently large*; we will rely on the fact that $N \geqslant 1$.

These facts immediately follow from our construction and the proof of Theorem 3.1 (by inspection). They will simplify our aim of getting rid of disjunctions in the next section.

5. Eliminating disjunctions with two extra context variables

In this section we show that *just two* extra existentially quantified context variables suffice to eliminate all disjunctions from the formula (6), *independently* of their (finite) number, provided we have a binary function symbol. This contrasts with the case of free semigroups, where the number of existentially quantified variables grows proportionally (logarithmically at best) with the number of disjunctions needed to be eliminated; see Durnev [15]. In fact, Büchi and Senger [14] describe a method using about 40 auxiliary existentially quantified variables to eliminate a single disjunction. Durnev [15] improves this number to 4.

Our disjunction elimination method is described by the following lemma. Let $[\,]$ denote the empty list ε and $[x_0, x_1, \ldots, x_N] = f(x_0, [x_1, \ldots, x_N])$, where f is *binary* (our method easily generalizes for a symbol of any arity >2). Suppose for convenience that a is an extra *unary* function symbol. In fact, we may use an appropriate context, like $\lambda x.0(f(1(x), 1(\varepsilon)))$, constructed of ε, 0, 1, and f instead of $\lambda x.a(x)$. Recall that our operator algorithm A is fixed, and we make reference to the corresponding fixed terms t_i and t'_i describing its incorrect runs and enumerated in the proofs of the previous sections.

Lemma 5.1. *Let Θ be the substitution $\{t'_i/x_i\}_{i=0}^{N}$, where t'_i's are the terms in formula* (6).[7] *Consider the system of two context equations*:

[7] All starting with f. Recall that t'_i's are non-ground.

$$\begin{aligned} G(a(H(a))) = \quad & [a((\lambda x_0.[x_0,\ldots,x_N])a), \\ & \ldots \\ & a((\lambda x_i.[x_0,\ldots,x_N])a), \\ & \ldots \\ & a((\lambda x_N.[x_0,\ldots,x_N])a)]\Theta, \end{aligned} \tag{7}$$

$$H(f(r,0S)) = [x_0,\ldots,x_N]\Theta \equiv [t'_0,\ldots,t'_N]. \tag{8}$$

For every ground term r of signature Σ_2 the following claims are equivalent:

1. *the system* (7), (8) *has a solution*,
2. *r is an incorrect run of the operator algorithm A.*

Intuitive explanation. The term on the right of (7) is:

$$\begin{array}{l} [\ a\ [\ a,\ t'_1,\ \ldots\ t'_i,\ \ldots\ t'_N\ ], \\ \quad a\ [\ t'_0,\ a,\ \ldots,\ t'_i,\ \ldots,\ t'_N\ ], \\ \qquad\qquad \ldots \\ \quad a\ [\ t'_0,\ t'_1,\ \ldots,\ a,\ \ldots,\ t'_N\ ], \\ \qquad\qquad \ldots \\ \quad a\ [\ t'_0,\ t'_1,\ \ldots,\ t'_i,\ \ldots,\ a\ \ ]\ ]\ ] \end{array} \tag{9}$$

with a (being a shorthand for $a(\varepsilon)$) on the diagonal.

We use a special 'grid' structure of (9) to model disjunction (recall that all t'_i start with f).

Proof. The direction 1. $\Leftarrow$ 2. is straightforward. Assume a ground term r is an incorrect run of the operator algorithm A for the reason it satisfies the ith disjunct in (6), for $0 \leqslant i \leqslant N$. Let the substitution $\Upsilon \equiv \{G_i/G, H_i/H\}$, be defined by

$$\begin{aligned} G_i = \lambda u. \quad & [a((\lambda x_0.[x_0,\ldots,x_N])a), \\ & \ldots \\ & a((\lambda x_{i-1}.[x_0,\ldots,x_N])a), \\ & u, \\ & a((\lambda x_{i+1}.[x_0,\ldots,x_N])a), \\ & \ldots \\ & a((\lambda x_N.[x_0,\ldots,x_N])a)]\Theta, \end{aligned} \tag{10}$$

$$H_i = (\lambda y_i.[x_0,\ldots,x_{i-1},y_i,x_{i+1},\ldots,x_N])\Theta,$$

where u, y_i are fresh variables.

Clearly, applying this substitution Υ to (7) yields the identity. Moreover, by substituting H_i for H in (8) we obtain

$$[t'_0,\ldots,t'_{i-1},f(r,0S),t'_{i+1},\ldots,t'_N] = [t'_0,\ldots,t'_{i-1},t'_i,t'_{i+1},\ldots,t'_N],$$

which has a solution by our assumption that r is an incorrect run of the operator algorithm A, satisfying the ith disjunct of (6). Thus the system (7), (8) has a solution. □

For the opposite direction 1. $\Rightarrow$ 2. in Lemma 5.1 we prove the following (contrapositive)

Lemma 5.2. *Let r be a correct run of the operator algorithm A. Then the system* (7), (8) *has no solutions for context variables G,H,U,V,S.*

Proof. The system (7), (8) cannot have solutions of the form (10), because the opposite would mean that r is an incorrect run (see the previous proof).

Let us consider other 'possible' solutions Υ to (7), (8), i.e., substitutions satisfying simultaneously

$$\begin{aligned} G(\mathbf{a}(H(a)))\Upsilon = [\ & \mathbf{a}\ [\ a,\ t_1',\ \ldots\ t_i',\ \ldots\ t_N'\ ],\\ & \mathbf{a}\ [\ t_0',\ a,\ \ldots,\ t_i',\ \ldots,\ t_N'\ ],\\ & \qquad\ldots\\ & \mathbf{a}\ [\ t_0',\ t_1',\ \ldots,\ a,\ \ldots,\ t_N'\ ],\\ & \qquad\ldots\\ & \mathbf{a}\ [\ t_0',\ t_1',\ \ldots,\ t_i',\ \ldots,\ a\ ]\ ]\ \Upsilon,\\ H(f(r,0S))\Upsilon = \quad & [\ t_0',\ t_1',\ \ldots,\ t_i',\ \ldots,\ t_N'\ ]\quad \Upsilon. \end{aligned} \tag{11}$$

Such 'possible' solutions[8] Υ split into the following two remaining cases (besides the already considered case (10) corresponding to incorrect runs), depending on how the boldface and italic occurrences **a** and a on the left-hand side of the first equation match with right-hand side occurrences. Note that in the case (10) the boldface **a** on the left matches one of the **a**'s on the right, say in line i, and the italic a on the left matches the shown italic occurrence of a in the *same* line i.

Case 1: The boldface occurrence **a** from the left-hand side $G(\mathbf{a}(H(a)))$ of the first equation in (11) matches with one of the boldface occurrences **a** on the right-hand side (let it be the ith occurrence of boldface **a** on the right). At the same time, the italicized occurrence a from $G(\mathbf{a}(H(a)))$ does not match the corresponding a on the right shown in the same ith line, matching instead within jth term, where $j \neq i$ (otherwise we have a solution of the form (10) meaning that r is an incorrect run; see the previous proof).

Therefore, by analyzing the first equation of (11), a 'possible' solution for H is either

$$\lambda z.[\ldots \underbrace{a}_{i},\ldots,t_j'\,\Upsilon[z/a(\varepsilon)],\ldots] \quad \text{or} \quad \lambda z.[\ldots,t_j'\,\Upsilon[z/a(\varepsilon)],\ldots,\underbrace{a}_{i},\ldots]$$

with $i \neq j$, and $[z/a(\varepsilon)]$ (ambiguously) meaning the replacement of exactly one occurrence of $a(\varepsilon)$ with z.

It is easily seen that none of such solutions can satisfy the second equation of (11), because $t_i'\,\Upsilon$ (on the right of this equation) cannot be equal for any Υ to $a \equiv a(\varepsilon)$ in the ith position on the left, since all the terms t_i' start with function symbol f; see (6) and the formula preceding (6).

[8] We show below that such solutions are in fact impossible.

Case 2. The whole subterm $\mathbf{a}(H(a))$ on the left-hand side of the first equation in (11) matches *with some proper*[9] *subterm* of $t'_k\Upsilon$ on the right, for some $0 \leqslant k \leqslant N$. In other words, neither $\mathbf{a}$, nor a in $\mathbf{a}(H(a))$ on the left matches a visible $\mathbf{a}$ or a on the right of the first equation of (11).

By assumption that r is a correct run, the term $f(r, 0S)$ cannot match any of $t'_j\Upsilon$ in the second equation of (11), because the opposite would mean that the run r is incorrect containing some forbidden pattern enumerated in the proof of Theorem 3.1. Therefore, in order to satisfy the second equation in (11), H should be substituted by any 'possible' solution Υ with

$$\lambda x.[q_0, \ldots, q_l(x), \ldots, q_N]$$

for some $0 \leqslant l \leqslant N$, $q_i \equiv t'_i\Upsilon$, for $i \in \{1, \ldots, l-1, l+1, \ldots, N\}$, and $q_l(x) \not\equiv x$. Let p_i denote the number of occurrences of a in q_i.

Since the whole subterm $\mathbf{a}(H(a))$ on the left of the first equation in (11) matches *with some proper subterm* of $t'_k\Upsilon$ on the right (the case we analyze), the term $t'_k\Upsilon$ should contain $a[q_0, \ldots, q_l(a), \ldots, q_N]$ as a *proper* subterm. Let us show that this condition cannot yield a solution.

1. Suppose, $k \neq l$. Then $t'_k\Upsilon$ *properly contains* q_k, and therefore the second equation in (11) cannot be satisfied, because it requires $q_k = t'_k\Upsilon$.
2. If $k = l$, then $t'_k\Upsilon$ properly contains $aHa \equiv a[q_0, \ldots, q_k(a), \ldots, q_N]$. Thus $t'_k\Upsilon$ contains at least $\sum_{i=0}^{N} p_i + 2$ occurrences of a. On the other hand, to satisfy the second equation of (11) we should have

$$q_k(f(r, 0S\Upsilon)) = t'_k\Upsilon. \tag{12}$$

Note that $q_k(f(r, 0S\Upsilon))$ contains at most $p_k + s$ occurrences of a, where s is the number of occurrences of a in $S\Upsilon$, (compared with $\sum_{i=0}^{N} p_i + 2$ on the right), because r is a correct run containing no a's at all. Hence, the equality (12) cannot hold, because its left-hand side should contain fewer occurrences of a than the right-hand side. Indeed (1) if $s = 0$, clearly $p_k < \sum_{i=0}^{N} p_i + 2$; (2) if $s > 0$, then also $p_k + s < \sum_{i=0}^{N} p_i + 2$, because every $p_i \geqslant s$ (since each t'_i contains at least one occurrence of S by construction; see (6) and the formula preceding (6)), and $N \geqslant 1$ (see the concluding remarks in the end of Section 4).

Therefore, the system of context equations (7), (8) (equivalent, by using a binary function symbol f, to just one equation) has no solutions. This finishes the proof of Lemmas 5.1, 5.2, and concludes the proof of the main claim of this paper. □

6. Conclusions

In this paper we showed undecidability of the $\forall\exists^5$-equational theory of context unification. Decidability of the genuine CUP, i.e., existential equational theory of context

[9] Recall that all t'_i start with f.

unification, remains a fascinating open problem. The long-standing unresolved status and hardness of the latter problem may be explained by the fact that it lies between technically difficult: (1) decidable case of word equations (Makanin) and (2) undecidable case of second-order unification with unary variables (Goldfarb–Farmer).

Our result provides for the simplest currently known quantifier prefix for which context unification is undecidable. Compared with free semigroups, our undecidability result holds for the same $\forall\exists^3$-prefix when positive theories are concerned, but we get a considerable improvement of $\forall\exists^5$-for equational theories, due to the novel method of eliminating arbitrary many disjunctions by using just two extra existential context variables (in presence of a function symbol of arity $\geqslant 2$).

As problems for further research let us mention the following ones:

1. Improving (if possible) a weak NP-lower bound for the CUP inherited from word equations. This may justify the intuitive inherent computational difficulty of the problem, and, lack of the undecidability result may provide evidence for (provable) intractability of the problem.
2. Simplifying (if possible) the number of existential quantifiers in the $\forall\exists^5$-prefix, or else proving decidability for simpler prefixes.
3. Venturing to generalize Makanin's techniques of generalized equations and their transformations with bounding exponent of periodicity for context equations (as described in the end of the Introduction), or, if proved unsuccessful, inventing a radically new approach.
4. Trying to prove the *undecidability* of the context unification problem.
5. Attempting to carry out the disjunction elimination method from Section 5 *without a binary symbol*, with a *constant number of auxiliary existentially quantified variables*. This will improve the best undecidability result for free semigroups due to Durnev [15].

A substantial further effort seems necessary to achieve any of the above goals.

Acknowledgements

The author thank Harald Ganzinger, Manfred Schmidt-Schauß, Margus Veanes, and the anonymous referees for discussions, comments, and helpful suggestions for the improvement of the paper.

References

[1] G.S. Makanin, The problem of solvability of equations in a free semigroup, Math USSR Sbornik 32(2) (1977) 127–198.
[2] W.D. Goldfarb, The undecidability of the second-order unification problem, Theoret. Comput. Sci. 13 (1981) 225–230.
[3] W. Farmer, Simple second-order languages for which unification is undecidable, Theoret. Comput. Sci. 87 (1991) 25–41.
[4] M. Schmidt-Schauß. Unification of stratified second-order terms. Interner Bericht 12/94, University of Frankfurt am Main, 1994.

[5] J. Niehren, M. Pinkal, P. Ruhrberg, On equality up-to constraints over finite trees, context unification, and one-step rewriting., in: W. McCune (Ed.), CADE-14, Lecture Notes in Computer Science, Vol. 1249, Springer, Berlin, 1997, pp. 34–48.
[6] M. Schmidt-Schauß, K.U. Schulz, On the exponent of periodicity of minimal solutions of context equations, in: T. Nipkow (Ed.), Rewriting Techniques and Applications'98, Lecture Notes in Computer Sciences, Vol. 1379, Springer, Berlin, 1998, pp. 61–75.
[7] M. Schmidt-Schauß, A decision algorithm for distributive unification, Theoret. Comput. Sci. 208 (1998) 111–148.
[8] W. Farmer, A unification algorithm for second-order monadic terms, Annu. Pure Appl. Logic 39 (1988) 131–174.
[9] J. Levy, M. Veanes, On the undecidability of second-order unification, Inform. Comput. 159 (2000) 125–150.
[10] W.V. Quine, Concatenation as a basis for arithmetic, J. Symbolic Logic 11 (4) (1946) 105–114.
[11] V.G. Durnev, Positive theory of a free semigroup, Soviet Math. Doklady 211 (4) (1973) 772–774.
[12] S.S. Marchenkov, Undecidability of the positive $\forall\exists$-theory of a free semigroup, Siberian Math. J. 23 (1) (1982) 196–198.
[13] V. Durnev, Studying algorithmic problems for free semi-groups and groups, Logical; Foundations of Computer Science (LFCS'97), Lecture Notes in Computer Science, Vol. 1243, Springer, Berlin, 1997, pp. 88–101.
[14] J.R. Büchi, S. Senger, Coding in the existential theory of concatenation, Arch. Math. Logic 26 (1986) 101–106.
[15] V.G. Durnev, Undecidability of the positive $\forall\exists^3$-theory of a free semigroup. Siberian Math. J. 36 (5) (1995) 1067–1080 (in Russian, also exists in English translation).
[16] S. Vorobyov, $\forall\exists^*$-equational theory of context unification is Π_1^0-hard. Research Report MPI-I-98-2-008, Max-Planck Institut für Informatik, 1998a.
[17] S. Vorobyov, $\forall\exists^*$-equational theory of context unification is Π_1^0-hard. in: L. Brim, J. Gruska, J. Zlatuška (Eds.), Proc. 23rd Internat. Symp. on Mathematical Foundations of Computer Science (MFCS'98), Lecture Notes on Computer Science, Vol. 1450, Brno, Czech Republic, Springer, Berlin, August 24–28, 1998b, pp. 597–606.
[18] J. Jaffar, Minimal and complete word unification, J. ACM 37 (1) (1990) 47–85.
[19] K.U. Schulz, Word unification and transformation of generalized equations, J. Automat. Reasoning 11 (1993) 149–184.
[20] A. Kościelski, L. Pacholski, Complexity of Makanin's algorithm, J. ACM 43 (4) (1996) 670–684.
[21] C. Gutiérrez, Satisfiability of word equations with constants is in exponential space. Proc. 39th Annu. IEEE Symp. on Foundations of Computer Science'98, 1998, pp. 112–119.
[22] W. Plandowski, Satisfiability of word equations with constants is in PSPACE. 40th Annu. IEEE Symp. on Foundations of Computer Science'99, 1991.
[23] A.I. Mal'cev, Algorithms and Recursive Functions., Wolters-Noordhoff, Groningen, Holland, 1970.
[24] M. Minsky, Recursive unsolvability of Post's problem of 'tag' and other topics in the theory of Turing machines, Annu. Math. 74 (3) (1961) 437–455.
[25] H. Rogers, Theory of Recursive Functions and Effective Computability, McGraw-Hill, New York, 1967.