$\forall\exists^*$-Equational Theory of Context Unification is Π_1^0-Hard

Sergei Vorobyov

Max-Planck Institut für Informatik, Im Stadtwald, D-66123, Saarbrücken, Germany, sv@mpi-sb.mpg.de, http://www.mpi-sb.mpg.de/~sv

Abstract. Context unification is a particular case of second-order unification, where all second-order variables are *unary* and only *linear* functions are sought for as solutions. Its decidability is an open problem. We present the simplest (currently known) undecidable quantified fragment of the theory of *context unification* by showing that for every signature containing a ≥ 2-ary symbol one can construct a *context equation* $\mathcal{E}(p, r, \overline{F}, \overline{w})$ with parameter p, first-order variables r, $\overline{w}$, and context variables $\overline{F}$ such that the set of true sentences of the form

$$\forall r\, \exists\, \overline{F}\, \exists\, \overline{w}\;\; \mathcal{E}(p, r, \overline{F}, \overline{w})$$

is Π_1^0-hard (i.e., every co-r.e. set is many-one reducible to it), as p ranges over finite words of a binary alphabet. Moreover, the existential prefix above contains just 5 context and 3 first-order variables.

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 [7]; CUP coincides with this problem for monadic signatures.

– A specialization of the *Second-Order Unification* (SOU), known to be *undecidable* due to Goldfarb [6,5]. CUP is almost SOU, 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, theorem proving, equational unification, constraint solving, computational linguistics, software engineering [11,9,12]. CUP is stated as follows:

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

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

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

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

Currently the decidability of CUP is an open problem [11,9,12]. 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.

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 [7]), and the undecidable case of SOU settled by Goldfarb [6] and reinforced by Farmer [5]. Farmer's result is also technically quite difficult.

Goldfarb [6] demonstrated that SOU is undecidable for second-order languages containing at least one ≥ 2-ary function constant and finitely many *unary* and *ternary* function variables. Later Farmer [5] improved it by showing that SOU 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 Makanin's result [7] that SOU is decidable when all variable and constant function symbols are unary. Farmer [4] 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.*

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

In this paper we show that adding just one outermost universal quantifier to a context equation (1) leads to the Π^0_1-hard class of formulas, where Π^0_1 is the class of all co-recursively enumerable sets.

For comparison, the following undecidability results are known about quantified fragments of context unification. Quine [10] showed that the full first-order theory of free semigroups is undecidable (this corresponds to context unification in unary signatures). Durnev [3] improved it to the undecidability of $\exists\forall\exists^3$-positive (without negation, but with $\wedge$ and $\vee$) theory of free semigroups. Marchenkov [8] improved it to the undecidability of $\forall\exists^4$-positive theory of free semigroups. Durnev [2] improved it to undecidability of $\forall\exists^3$-positive theory of free semigroups. Niehren, Pinkal, and Ruhrberg [9] claimed undecidability of the $\exists^*\forall^*\exists^*$-theory of context unification.

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., [1]), depending on the number of disjunctions involved. Thus the above undecidability results 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.

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 Turing machines, gives the undecidability of the $\forall\exists^8$-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 signature containing a* ≥ 2*-ary symbol one can construct a* context equation $\mathcal{E}(p, r, C, F, F', G, H, x, y, z)$ *with parameter* p*, first-order variables* r, x, y, z*, and context variables* C, F, F', G, H *such that the set of true sentences of the form*

$$\forall r \; \exists\, C, F, F', G, H \; \exists\, x, y, z \;\; \mathcal{E}\,(p, r, C, F, F', G, H, x, y, z) \tag{2}$$

is Π^0_1*-hard (i.e., every co-r.e. set is many-one reducible to it), as* p *ranges over finite words of a binary alphabet.* □

It follows, in particular, that the $\forall\exists^8$-equational theory (without $\wedge$, $\vee$, $\neg$) of context unification is *undecidable.*

Warning. We would like to stress that in this paper we do not intend to improve the undecidability results of Marchenkov and Durnev [8,2] 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 over [8,2] (in a different framework of context unification) 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. 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. We believe that applying the methods of disjunction elimination with a constant number of extra variables (Section 5) directly to the reductions of [8,2] (the proofs absent from [2] are unavailable to the author) will yield the undecidability of the $\forall\exists^5$-equational theory of context unification.

Paper Outline. After Section 2 with preliminaries, in Sections 3, 4 we give the main construction of an $\forall\exists^*$-sentence (expressing non-applicability) from a DTM with a complete Σ^0_1 domain. In Section 5 we eliminate disjunctions.

2 Preliminaries

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

Definition 1 (Terms). *The set* $\mathcal{T}(\Sigma, X)$ *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* $n \geq 0$ *an expression* $f(t_1, \ldots, t_n)$ *is a term whenever* t_i*'s are terms.* □

We assume all the standard definitions and conventions concerning λ-notation, like β-reduction, normalization, etc.

Definition 2 (Contexts). *A* context *is an expression of the form $\lambda x.t(x)$, where $t(x) \in \mathcal{T}(\Sigma, \{x\})$ contains* exactly one *occurrence of the variable x. A context with β-normal form $\lambda x.x$ is called* empty. $\square$

Remark 1. Note that the '*exactly one*' requirement, absent from the definition of SOU, is a characteristic feature of CUP, distinguishing it from SOU. $\square$

Definition 3 (Context Terms). *Are defined inductively: if $\phi \in \Sigma \cup \mathcal{F}$ is n-ary and $t_1, \ldots, t_n$ are context terms, then $\phi(t_1, \ldots, t_n)$ is a context term.* $\square$

Convention. Writing terms and context terms with unary function symbols and function variables we usually drop parentheses to improve readability.

Definition 4 (CUP, Context Equations). *An instance of CUP, also called a* context equation *(CE for short), is an expression $\tau_1 \overset{?}{=} \tau_2$, where $\tau_{1,2}$ are context terms. A* solution *to a CE $\tau_1 \overset{?}{=} \tau_2$ is a substitution θ of contexts for context variables such that $\theta(\tau_1) \equiv_\beta \theta(\tau_2)$ (equality modulo the usual β-reduction).* $\square$

Remark 2. The requirement of replacing functional variables with *contexts*, as opposed to arbitrary second-order λ-terms, distinguishes CUP from SOU. When this requirement is dropped, the problem becomes *undecidable* [5]. $\square$

Turing Machine with Complete R.E. Domain. A Σ_1^0-set is a recursively enumerable set. A Π_1^0-set is a complement of a Σ_1^0-set. An Σ_1^0-set (resp., Π_1^0-set) A is called *complete* iff every Σ_1^0-set (resp., Π_1^0-set) B is *many-one reducible* to A, i.e., there exists a total recursive function f such that $x \in B \Leftrightarrow f(x) \in A$.

It is well known that there exists a DTM M whose domain is a complete Σ_1^0-set, and, therefore, the set of elements it does not accept is a complete Π_1^0-set. We may assume, without loss of generality, that the tape alphabet B of M consists of two symbols, 0 and 1, the tape of M is infinite to the right, M never tries to move left from its leftmost tape cell, M may only extend its tape by writing new symbols on the right end, that the states of M are $Q = \{q_0, \ldots, q_f\}$, q_0 is the unique initial state, q_f is the unique final state of M, M always starts by observing its leftmost tape cell, and M immediately stops entering state q_f. Further we assume that M is a fixed such DTM.

The DTM M applies to a nonempty word $b_1 \ldots b_m \in B^+$ iff there exists a finite sequence of IDs (instantaneous descriptions) $id_0, \ldots, id_F$ of M such that $id_0 \equiv q_0 b_1 \ldots b_m$, $id_F \equiv \gamma q_f \delta$ for some $\gamma, \delta \in B^*$, and such that for every pair of IDs id_i, id_{i+1} in the sequence id_{i+1} is obtained from id_i by application of some command of M. We omit the well-known definitions.

We will represent IDs of M by terms constructed from unary function symbols, starting from the constant ε for the empty word. We have a unary function symbol for every symbol in $B \cup Q$, and represent a word $q_0 10101010$ as a term $q_0(1(0(1(0(1(0(1(0(\varepsilon)))))))))$, which for simplicity will always be written as $q_0 10101010$ (i.e., with ( ) and ε omitted), if it does not lead to ambiguity.

We will represent a sequence $id_0, \ldots, id_F$ of IDs as a right-flattened list

$$f(id_0, f(id_1, f \ldots, f(id_{F-1}, f(id_F, \varepsilon)) \ldots)), \qquad (3)$$

where id_i's are term representations of words using unary functional symbols as described above. Thus the DTM M applies to an input iff such a term (3) exists.

Signature. Formally, we will use the following signature Σ: 1) ε - constant, for the empty word and list, 2) 0, 1 - unary for the tape alphabet B, 3) $q_0, \ldots, q_f$ - unary for M's states in Q, 4) f - binary list constructor, 5) a - unary, auxiliary.

Convention. If not stated otherwise, everywhere below s denotes a symbol from $B \cup Q$, b denotes a symbol from B, q denotes a symbol from Q.

3 Sentence Expressing Inapplicability

To prove the main claim of this paper we will write a sentence of the following form (with context variables C, F, F', G, H, and ordinary variables r, x, y, z):

$$\forall r \, \exists C, F, F' \, \exists x, y, z \, \exists G, H \;\; \Phi\Big(f(q_0 b_{i_1} \ldots b_{i_p}, r), t_1, \ldots, t_m\Big), \qquad (4)$$

which expresses the fact that the DTM M *does not apply* to the input string $\bar{s} \equiv b_{i_1} \ldots b_{i_p}$, because for every r, the term $t \equiv f(q_0 b_{i_1} \ldots b_{i_p}, r)$ *is not* a correct run (3) of M, i.e., $\bar{s} \equiv b_{i_1} \ldots b_{i_p}$ is in the complement of the r.e.-complete domain of M. This implies the main claim of the paper.

Technically, expressing that a term $t \equiv f(q_0 b_{i_1} \ldots b_{i_p}, r)$ *is not* a correct run amounts to saying that 'something goes wrong' in t. For example, t contains 'senseless' subterms like $f(f(\ldots, \ldots), \ldots)$, or $a(\ldots)$, etc. It turns out that this can be expressed by saying 'contains one of the *finitely* many wrong patterns' $t_1, \ldots, t_m$, which can be done by saying $t = Ct_1 \vee \ldots \vee t = Ct_m$, where C is a context variable used to say 'there exists a subterm of t'. All such patterns are expressed by using 2 context variables F, F' and 3 first-order variables x, y, z. In Section 5 we show how to get rid of disjunction by using just two extra existentially quantified context variables (G and H in (4)), independently of the number m of patterns. Thus just one context equation with 5 context and 3 first-order existentially quantified variables as in (2), (4) suffices for undecidability.

4 Forbidden Patterns

We enumerate all forbidden patterns preventing a term to be a correct run.

Forbidden Structural Patterns. A term containing one of the following subterms cannot represent a correct run (for some context F and terms x, y, z):

1. $f(F(a(x)))$. **Reason:** a is an auxiliary symbol and cannot occur in a run.
2. $f(f(x, y), z)$. **Reason:** a run is a right-flattened list of IDs, f cannot occur in the first argument position to itself.

3. $f(x, g(y))$ {for every function symbol $g \in B \cup Q$}. **Reason:** a run is a right-flattened list of IDs.
4. $q(F(q'(x)))$ {for all $q, q' \in Q$}. **Reason:** at most one state symbol per ID.
5. $g(f(x, y))$ {for all $g \in B \cup Q$}. **Reason:** IDs cannot contain f.
6. $f(F(q_f(x)), f(y, z))$. **Reason:** entering q_f DTM stops.
7. $q(s(x))$ {for all $q \in Q \backslash \{q_f\}$ and $s \in B$ such that M has no commands $(q, s \rightarrow \ldots)$}.
8. $q(\varepsilon)$ {for all $q \in Q \backslash \{q_f\}$ such that there are no commands to extend the tape on the right end in state q}.

Patterns for Incorrect ID Transitions. Now we enumerate patterns that cannot occur in a correct run of the DTM M for the reason a term matching one of these patterns contains a 'senseless' ID transition. This can be expressed by existence of solutions to a *finite* number of 'local' context equations. As an easy and representative example, note that an ID α cannot be transformed into an ID β, if for some context F, state $q \in Q$, tape symbols $s_{1,2,3,4,5} \in B$, and x, y one has simultaneously $\alpha \equiv F s_1 q s_2 x$ and $\beta \equiv F s_3 s_4 s_5 y$. This is because each TM's transition moves head either left or right, so in a correct transition either s_3 or s_5 should belong to Q (be a state). This illustrates the main idea. By routinely enumerating all possibilities we can assure that a transition is correct iff it *does not match any* of the patterns enlisted.

The reader is invited to check that all these patterns use only three first-order variables x, y, z, and two context variables F, F'. Recall that we use these patterns P_i disjunctively in $t = CP_1 \vee \ldots \vee t = CP_m$ to say that t contains a subterm matching one of the patterns $P_1, \ldots, P_m$ (where the context variable C is used to say 'there exists a subterm'). Since $\exists$ distributes over $\vee$, the variables may be reused and we need just three context C, F, F', and three first-order variables x, y, z. The two extra context variables G, H will be needed to transform a disjunction into an equation in Section 5.

List of the patterns for incorrect transitions.

$f(Fsx, f(Fs'y, z))$ {for $s, s' \in B$, $s \neq s'$}. **Reason:** in two succeeding IDs the leftmost position in which they differ should necessarily contain a state from Q. For example, for a right shift command $(q, s \rightarrow q', s', R)$ we have, in a correct ID transition (the first disagreement is $q - s'$):

$$\begin{aligned} ID_i &= s_1 \ldots s_n \; q \; s \; s_{n+1} \ldots s_{n+m} \\ ID_{i+1} &= s_1 \ldots s_n \; s' \; q' \; s_{n+1} \ldots s_{n+m} \end{aligned}$$

For a left shift command $(q, s \rightarrow q', s', L)$ we have, in a correct ID transition:

$$\begin{aligned} ID_i &= s_1 \ldots s_{n-1} \; s_n \; q \; s \; s_{n+1} \ldots s_{n+m} \\ ID_{i+1} &= s_1 \ldots s_{n-1} \; q' \; s_n \; s' \; s_{n+1} \ldots s_{n+m} \end{aligned}$$

(the first disagreement is $s_n - q'$). For an 'extension on the right' command $(q, \varepsilon \rightarrow q', s')$ we have, in a correct ID transition (first disagreement $q - q'$):

$$\begin{aligned} ID_i &= s_1 \ldots s_n \; q \\ ID_{i+1} &= s_1 \ldots s_n \; q' \; s' \end{aligned}$$

$f(Fsx, f(F(\varepsilon), z))$ $\{$for all $s \in B \cup Q\}$. **Reason:** ID_{i+1} cannot be shorter ID_i.
$f(qs_1x, f(s_2s_3y, z))$ $\{$for all $s_3 \in B\}$. **Reason:** if the first symbol of the ID_i is a state, then the second symbol in the ID_{i+1} should be a state.
$f(Fs_1qs_2x, f(Fs_3s_4s_5y, z))$ $\{$for all $s_3, s_5 \in B\}$. **Reason:** if the k-th symbol in ID_i is a state then either $k+1$-th or $k-1$-th in ID_{i+1} should be a state.
$f(Fqsx, f(Fs_1s_2y, z))$ $\{$for every command $(q, s \to q', s', R)$ and every pair of symbols $s_1, s_2 \in B \cup Q$ such that either $s_1 \neq s'$ or $s_2 \neq q'\}$. **Reason:** an ID $Fqsx$ should yield $Fs'q'x$, thus each pair of consecutive IDs $Fqsx$, Fs_1s_2y is incorrect.
$f(Fqsx, f(Fs_1(\varepsilon), z))$ $\{$for every $s_1 \in B \cup Q\}$. **Reason:** ID_{i+1} abruptly ends. Similar patterns must be written for the case of left shift commands below.
$f(Fbqsx, f(Fs_1s_2s_3y, z))$ $\{$for every command $(q, s \to q', s', L)$, every $b \in B$, and every triple of symbols $s_1, s_2, s_3 \in B \cup Q$ such that either $s_1 \neq q'$, or $s_2 \neq b$, or $s_3 \neq s'\}$. **Reason:** an ID $Fbqsx$ should yield $Fq'bs'x$, thus each pair of consecutive IDs $Fbqsx$, $Fs_1s_2s_3y$ is incorrect.
... Similar patterns should be spelled out for the commands extending the tape on the right. We leave it as a straightforward exercise.
$f(FqsF's_1x, f(Fs'q'F's_2y, z))$ $\{$for every command $(q, s \to q', s', R)$ and every pair of different symbols $s_1, s_2 \in B\}$. **Reason:** an ID transformation is 'almost' correct, but some symbol to the right of the head is copied erroneously.
... Similar pattern for an 'almost correct' left shift command (exercise).

Let us add a brief explanation of how the above patterns work. In a correct run no ID can contain two states (since we have a pattern $q(F(q'(x)))$. Can a correct run contain and ID with no state symbols at all? Let us show that this is impossible. For, if such a situation was possible, there would exist a pair of neighboring IDs with the least index i such that id_i contains a state symbol and id_{i+1} does not. Then it is easy to see that one of the patterns responsible for the transition correctness would match, thus guaranteeing the run candidate incorrectness.

We thus constructed a finite number of patterns $t_1, \ldots, t_m$ such that the DTM M *does not apply* to the word $\bar{s} = b_{i_1} \ldots b_{i_p}$ if and only if

$$\forall r\ \exists C, F, F'\ \exists x, y, z\ \Big(f(q_0 b_{i_1} \ldots b_{i_p}, r) = Ct_1 \vee \ldots \vee f(q_0 b_{i_1} \ldots b_{i_p}, r) = Ct_m\Big).$$

This already proves that the positive $\forall\exists^6$-theory of context unification is Π_1^0-hard (by the choice of M with a complete r.e.- or Σ_1^0-set). In the next section we proceed to eliminating disjunctions from the sentence above.

5 Trading Disjunctions for Equations

To finish the proof of the main claim we show how to represent a disjunction of context equations of the form $t = t_1 \vee \ldots \vee t = t_m$ by *just one* context equation, using only two auxiliary (existential) context variables, independently of the number m. Conjunctions do not cause any problems, because in presence of function symbols of arity ≥ 2, a conjunction of context equations can be easily expressed by just one equation ($s = s' \wedge t = t'$ iff $f(s,t) = f(s',t')$; similarly for larger arities). When all function symbols are unary, there exists a well-known trick to represent $s = s' \wedge t = t'$ as $satsbt = s'at's'bt'$, where a, b are different unary function symbols. Disjunction elimination is based on the following

Lemma 1. *Let Θ be the substitution $\left\{Ct_i/x_i\right\}_{i=1}^{m}$, where t_i's are the forbidden patterns enumerated in Section 4. Consider the system of two equations (where $[\,]$ denotes the empty list ε and $[e_0, \ldots] = f(e_0, [\ldots])$):*

$$\begin{aligned} G(a(H(a))) = \Big[\, & a\Big((\lambda x_1.[x_1, \ldots, x_m])a\Big), \\ & \ldots \\ & a\Big((\lambda x_i.[x_1, \ldots, x_m])a\Big), \\ & \ldots \\ & a\Big((\lambda x_m.[x_1, \ldots, x_m])a\Big)\Big]\ \Theta, \end{aligned} \tag{5}$$

$$Hf(q_0\bar{s}, r) = [x_1, \ldots, x_m]\ \Theta. \tag{6}$$

For every ground term r one has the following equivalence: the system (5), (6) has a solution if and only if ($\Leftrightarrow$) $f(q_0\bar{s}, r)$ is an incorrect run.

Brief Explanation. The term on the right of (5) is (with $a(\varepsilon)$ on the diagonal):

$$\begin{array}{l} \Big[\, a\,[\ a,\ Ct_2,\ \ldots\ Ct_i,\ \ldots\ Ct_m\ ], \\ \ \ a\,[\ Ct_1,\ a,\ \ldots, Ct_i, \ldots, Ct_m\ ], \\ \qquad\qquad \ldots \\ \ \ a\,[\ Ct_1,\ Ct_2\ \ldots,\ a,\ \ldots, Ct_m\ ], \\ \qquad\qquad \ldots \\ \ \ a\,[\ Ct_1, Ct_2, \ldots, Ct_i, \ldots,\ a\ \ ]\,\Big] \end{array} \tag{7}$$

Proof. ($\Leftarrow$) is straightforward. Suppose for a ground r the term $f(q_0\bar{s}, r)$ is an incorrect run for the reason it contains one of the forbidden patterns t_i. Let

$$\begin{aligned} G_i = \lambda u.\ \Big[& a\Big((\lambda x_1.[x_1, \ldots, x_m])a\Big), \\ & \ldots \\ & a\Big((\lambda x_{i-1}.[x_1, \ldots, x_m])a\Big), \\ & u, \\ & a\Big((\lambda x_{i+1}.[x_1, \ldots, x_m])a\Big), \\ & \ldots \\ & a\Big((\lambda x_m.[x_1, \ldots, x_m])a\Big)\Big]\ \Theta, \end{aligned} \tag{8}$$

$$H_i \ = \ \Big(\lambda x_i.[x_1, \ldots, x_i, \ldots, x_m]\Big)\ \Theta. \tag{9}$$

Clearly, substituting such G_i, H_i in (5) yields the identity. Moreover, by substituting H_i in (6) we obtain

$[Ct_1 \dots Ct_{i-1}, f(q_0\bar{s}, r), Ct_{i+1} \dots Ct_m] = [Ct_1 \dots Ct_{i-1}, Ct_i, Ct_{i+1} \dots Ct_m]$, which, obviously, has a solution for C, since $f(q_0\bar{s}, r)$ contains a forbidden pattern t_i by assumption. Thus the system (5), (6) has a solution. □

For the opposite direction $\Rightarrow$ in Lemma 1 we prove (the contrapositive)

Lemma 2. *Let $f(q_0\bar{s}, r)$ be a correct run. Then the system (5), (6) has no solutions.*

Proof. (5), (6) cannot have solutions of the form (8), (9), because the opposite would mean that $f(q_0\bar{s}, r)$ is incorrect (see the proof of the previous lemma). The other 'possible' solutions split into the following two cases.

Case 1. The outermost a on the left of (5) matches one of the outermost a's on the right, but the innermost a on the left does not match the corresponding a on the right. In other words, for some substitution Θ' either

$$\begin{array}{ll} H = \lambda z.\,[\,\dots, \quad a, \quad \dots, Ct_j\Theta'[z/a(\varepsilon)], \dots\,], & \text{or} \\ H = \lambda z.\,[\,\dots, Ct_j\Theta'[z/a(\varepsilon)], \dots, \quad a, \quad \dots\,], & \end{array}$$

with a in the i-th ($i \neq j$) place in the list. It is readily seen that none of such 'solutions' can satisfy (6), because $Ct_i\Theta'$ on the right cannot be equal (for any C, Θ') to $a \equiv a(\varepsilon)$ on the left, since all the forbidden patterns t_i enumerated in Section 4 contain function symbols different from a.

Case 2. The whole $a(H(a))$ on the left of (5) matches with some subterm of $Ct_k\Theta'$, $1 \leq k \leq m$, for some substitution Θ' (i.e., neither of a's in $a(H(a))$ on the left matches a visible a on the right of (5); see also (7)).

Since a correct ID $q_0\bar{s}$ cannot match any of Ct_j, in order to satisfy (6), H should be substituted with $\lambda x.[r_1, \dots, r_l(x), \dots, r_m]$ for some terms r_i. Consequently, $Ct_k\Theta'$ contains $a[r_1, \dots, r_l(a), \dots, r_m]$. Let us show that this cannot yield a solution.

Suppose, $k \neq l$. Then $Ct_k\Theta'$ *properly contains* r_k, and therefore (6) cannot be satisfied, because it requires $r_k = Ct_k\Theta'$.

Suppose, $k = l$. We have $Ct_k\Theta'$ contains $a[r_1, \dots, r_k(a), \dots, r_m]$. Thus $Ct_k\Theta'$ contains $p+2$ occurrences of a. On the other hand, to satisfy (6) we should have $r_k(f(q_0\bar{s}, r)) = Ct_k\Theta'$. Note that $r_k(f(q_0\bar{s}, r))$ contains at most p occurrences of a (compared with $p+2$ on the right). Hence the above equality cannot hold.

Thus, the system (5), (6) has no solutions. This finishes the proof of Lemmas 1, 2, and the proof of the main claim of his paper. □

6 Conclusions

By reduction from the non-acceptance problem for a deterministic Turing machine with a complete r.e. domain we demonstrated that for a one-parametric set of context equations $\mathcal{E}_p$ the set of true sentences of the form $\forall^1\exists^8\, \mathcal{E}_p$ is co-r.e.-hard as p runs over binary words. It follows that the $\forall^1\exists^8$-equational theory of context unification is undecidable. This gives the simplest currently known undecidable quantified fragment of context unification. The decidability of the CUP itself remains open. Quite surprisingly, the only known lower bound for the CUP is NP-hardness [12]. Thus, in the absence of the decidability proof it would be very interesting to obtain nontrivial lower complexity bounds for CUP.

Technically, we described an *ad hoc* method to eliminate disjunctions from the positive formulas (of some particular structure) of context unification, which is completely sufficient for the purposes of this paper. It is clear that the method can be generalized so as to apply to *arbitrary* positive formulas. We conjecture that combining techniques presented with ideas of [8,2] will lead to a more tight undecidability results for CUP, with fewer quantifiers.

Thanks to Margus Veanes and anonymous referees for insightful remarks.

References

1. J. R. Büchi and S. Senger. Coding in the existential theory of concatenation. *Archive of Mathematical Logic*, 26:101–106, 1986.
2. V. Durnev. Studying algorithmic problems for free semi-groups and groups. In *Logical; Foundations of Computer Science (LFCS'97)*, volume 1234 of *Lect. Notes Comput. Sci.*, pages 88–101. Springer-Verlag, 1997.
3. V. G. Durnev. Positive theory of a free semigroup. *Soviet Math. Doklady*, 211(4):772–774, 1973.
4. W. Farmer. A unification algorithm for second-order monadic terms. *Annals Pure Appl. Logic*, 39:131–174, 1988.
5. W. Farmer. Simple second-order languages for which unification is undecidable. *Theor. Comput. Sci.*, 87:25–41, 1991.
6. W. D. Goldfarb. The undecidability of the second-order unification problem. *Theor. Comput. Sci.*, 13:225–230, 1981.
7. G. S. Makanin. The problem of solvability of equations in a free semigroup. *Math USSR Sbornik*, 32(2):127–198, 1977.
8. S. S. Marchenkov. Undecidability of the positive ∀∃-theory of a free semigroup. *Siberian Math. Journal*, 23(1):196–198, 1982.
9. J. Niehren, M. Pinkal, and P. Ruhrberg. On equality up-to constraints over finite trees, context unification, and one-step rewriting. In W. McCune, editor, *CADE-14*, volume 1249 of *Lect. Notes Comput. Sci.*, pages 34–48. Springer-Verlag, 1997.
10. W. V. Quine. Concatenation as a basis for arithmetic. *J. Symb. Logic*, 11(4):105–114, 1946.
11. M. Schmidt-Schauß. Unification of stratified second-order terms. Interner Bericht 12/94, University of Frankfurt am Main, 1994.
12. M. Schmidt-Schauß and K. U. Schulz. On the exponent of periodicity of minimal solutions of context equations. In *Rewriting Techniques and Applications'98*, volume 1379 of *Lect. Notes Comput. Sci.*, pages 61–75. Springer-Verlag, 1998.