Quadratic Word Equations with Length Constraints, Counter Systems, and Presburger Arithmetic with Divisibility

Anthony W. Lin[1] and Rupak Majumdar[2]

[1] Oxford University, UK
[2] Max Planck Institute for Software Systems, Germany

Abstract. Word equations are a crucial element in the theoretical foundation of constraint solving over strings. A word equation relates two words over string variables and constants. Its solution amounts to a function mapping variables to constant strings that equate the left and right hand sides of the equation. While the problem of solving word equations is decidable, the decidability of the problem of solving a word equation with a length constraint (i.e., a constraint relating the lengths of words in the word equation) has remained a long-standing open problem. We focus on the subclass of quadratic word equations, i.e., in which each variable occurs at most twice. We first show that the length abstractions of solutions to quadratic word equations are in general not Presburger-definable. We then describe a class of counter systems with Presburger transition relations which capture the length abstraction of a quadratic word equation with regular constraints. We provide an encoding of the effect of a simple loop of the counter systems in the existential theory of Presburger Arithmetic with divisibility (PAD). Since PAD is decidable, we get a decision procedure for quadratic words equations with length constraints for which the associated counter system is *flat* (i.e., all nodes belong to at most one cycle). In particular, we show a decidability result (in fact, also an NP algorithm with a PAD oracle) for a recently proposed NP-complete fragment of word equations called regular-oriented word equations, when augmented with length constraints. Decidability holds when the constraints are extended with regular constraints with a 1-weak control structure.

1 Introduction

Reasoning about strings is a fundamental problem in computer science and mathematics. The first order theory over strings and concatenation is undecidable. A seminal result by Makanin [24] (see also [11, 15]) shows that the satisfiability problem for the *existential fragment* is decidable, by giving an algorithm for the satisfiability of *word equations*. A word equation $L = R$ consists of two words L and R over an alphabet of constants and variables. It is satisfiable if there is a mapping σ from the variables to strings over the constants such that $\sigma(L)$ and $\sigma(R)$ are syntactically identical.

An original motivation for studying word equations was to show undecidability of Hilbert's 10th problem (see, e.g., [26]). While Makanin's later result shows that word equations could not, by themselves, show undecidability, Matiyasevich in 1968 considered an extension of word equations with *length constraints* as a possible route to

showing undecidability of Hilbert's 10th problem [26]. A length constraint constrains the solution of a word equation by requiring a linear relation to hold on the lengths of words in a solution σ (e.g., $|x| = |y|$, where $|\cdot|$ denotes the string-length function). The decidability of word equations with length constraints remains open.

In recent years, reasoning about strings with length constraints has found renewed interest through applications in program verification and reasoning about security vulnerabilities. The focus of most research has been on developing practical string solvers (cf. [1, 5, 6, 14, 16, 21, 28, 31–33]). These solvers are sound but make no claims of completeness. Relatively few results are known about the decidability status of strings with length and other constraints (see [9] for an overview of the results in this area). The main idea in most existing decidability results is the encoding of length constraints into Presburger arithmetic [1, 9, 13, 22]. However, as we shall see in this paper, the length abstraction of a word equation (i.e. the set of possible lengths of variables in its solutions) need not be Presburger definable.

In this paper, we consider the case of *quadratic* word equations, in which each variable can appear at most twice [12, 19], together with length constraints and *regular constraints* (conjunctions $\bigwedge_{i=1}^{n} x \in L_i$ of assertions that the variable x must be assigned a string in the regular language L_i for each i). For quadratic word equations, there is a simpler decision procedure (called the Nielsen transform or Levi's method) based on a non-deterministic proof tree construction. The technique can be extended to handle regular constraints [12]. However, we show that already for this class (even for a simple equation like $xaby = yabx$, where x, y are variables and a, b are constants), the length abstraction need not be Presburger-definable. Thus, techniques based on Presburger encodings are not sufficient to prove decidability.

Our first observation in this paper is a connection between the problem of quadratic word equations with length constraints and a class of counter systems with Presburger transitions. Informally, the counter system has control states corresponding to the nodes of the proof tree constructed by Levi's method, and a counter standing for the length each word variable. Each step of Levi's method may decrease at most one counter. Thus, from any initial state, the counter system terminates. We show that the set of initial counter values which can lead to a successful leaf (i.e., one containing the trivial equation $\epsilon = \epsilon$) is precisely the length abstraction of the word equation.

Our second observation is that the reachability relation for a simple loop of the counter system can be encoded in the existential theory of Presburger arithmetic with divisibility ($\mathcal{PAD}$). The encoding is non-trivial in the presence of regular constraints, and depends on structural results on semilinear sets. As $\mathcal{PAD}$ is decidable [18, 23], we obtain a technique to symbolically represent the reachability relation for *flat* counter systems, in which each node belongs to at most one loop.

Moreover, the same encoding shows decidability for word equations with length constraints, provided the proof tree is associated with flat counter systems. In particular, we show that the class of *regular-oriented* word equations, introduced by [10], have flat proof trees. Thus, the satisfiability problem for quadratic regular-oriented word equations with length constraints is decidable (and in NEXP[3]).

[3] In fact, it is a NP algorithm with an oracle access to $\mathcal{PAD}$. The best complexity bound for the latter is NEXP and NP-hardness [18].

While our decidability result is for a simple subclass, this class is already non-trivial without length and regular constraints: satisfiability of regular-oriented word equations is NP-complete [10]. Our result generalizes previous decidability results [9]. Moreover, we believe that the techniques in this paper — the connection between acceleration and word equations, and the use of existential Presburger with divisibility — can pave the way to more sophisticated decision procedures based on counter system acceleration.

2 Preliminaries

General notation: Let $\mathbb{N} = \mathbb{Z}_{\geq 0}$ be the set of all natural numbers. For integers $i \leq j$, we use $[i, j]$ to denote the set $\{i, i+1, \ldots, j-1, j\}$ of integers. If $i \in \mathbb{N}$, let $[i]$ denote $[0, i]$. We use $\preceq$ to denote the component-wise ordering on $\mathbb{N}^k$, i.e., $(x_1, \ldots, x_k) \preceq (y_1, \ldots, y_k)$ iff $x_i \leq y_i$ for all $i \in [1, k]$. If $\bar{x} \preceq \bar{y}$ and $\bar{x} \neq \bar{y}$, we write $\bar{x} \prec \bar{y}$.

If S is a set, we use S^* to denote the set of all finite sequences, or *words*, $\gamma = s_1 \ldots s_n$ over S. The length $|\gamma|$ of γ is n. The empty sequence is denoted by ϵ. Notice that S^* forms a monoid with the concatenation operator $\cdot$. If γ' is a prefix of γ, we write $\gamma' \preceq \gamma$. Additionally, if $\gamma' \neq \gamma$ (i.e. a strict prefix of γ), we write $\gamma' \prec \gamma$. Note that the operator $\preceq$ is overloaded here, but the meaning should be clear from the context.

Words and automata: We assume basic familiarity with word combinatorics and automata theory. Fix a (finite) alphabet A. For each finite word $w := w_1 \ldots w_n \in A^*$, we write $w[i, j]$, where $1 \leq i \leq j \leq n$, to denote the segment $w_i \ldots w_j$.

Two words x and y are *conjugates* if there exist words u and v such that $x = uv$ and $y = vu$. Equivalently, $x = \mathrm{cyc}^k(y)$ for some k and for the *cyclic permutation* operation $\mathrm{cyc} : A^* \to A^*$, defined as $\mathrm{cyc}(\epsilon) = \epsilon$, and $\mathrm{cyc}(a \cdot w) = w \cdot a$ for $a \in A$ and $w \in A^*$.

Given a *nondeterministic finite automaton (NFA)* $\mathcal{A} := (A, Q, \Delta, q_0, q_F)$, a *run* of $\mathcal{A}$ on w is a function $\rho : \mathbb{N} \to Q$ with $\rho(0) = q_0$ that obeys the transition relation Δ. We may also denote the run ρ by the word $\rho(0) \cdots \rho(n)$ over the alphabet Q. The run ρ is said to be *accepting* if $\rho(n) = q_F$, in which case we say that the word w is *accepted* by $\mathcal{A}$. The language $\mathcal{L}(\mathcal{A})$ of $\mathcal{A}$ is the set of words in A^* accepted by $\mathcal{A}$. In the sequel, for $p, q \in Q$ we will write $\mathcal{A}_{p,q}$ to denote the NFA $\mathcal{A}$ with initial state replaced by p and final state replaced by q.

Word equations: Let A be a (finite) alphabet of constants and V a set of variables; we assume $A \cap V = \emptyset$. A *word equation* E is an expression of the form $L = R$, where $(L, R) \in (A \cup V)^* \times (A \cup V)^*$. A system of word equations is a nonempty set $\{L_1 = R_1, L_2 = R_2, \ldots, L_k = R_k\}$ of word equations. The length of a system of word equations is the length $\sum_{i=1}^{k} (|L_i| + |R_i|)$. A system is called *quadratic* if each variable occurs at most twice in all. A *solution* to a system of word equations is a homomorphism $\sigma : (A \cup V)^* \to A^*$ which maps each $a \in A$ to itself that equates the l.h.s. and r.h.s. of each equation, i.e., $\sigma(L_i) = \sigma(R_i)$ for each $i = 1, \ldots, k$.

For each variable $x \in V$, we shall use $|x|$ to denote a formal variable that stands for the length of variable x, i.e., for any solution σ, the formal variable $|x|$ takes the value $|\sigma(x)|$. Let L_V be the set $\{|x| \mid x \in V\}$. A *length constraint* is a formula in Presburger arithmetic whose free variables are in L_V.

A *solution* to a system of word equations with a length constraint $\Phi(|x_1|, \ldots, |x_n|)$ is a homomorphism $\sigma : (A \cup V)^* \to A^*$ which maps each $a \in A$ to itself such that

$\sigma(L_i) = \sigma(R_i)$ for each $i = 1, \ldots, k$ and moreover $\Phi(|\sigma(x_1)|, \ldots, |\sigma(x_n)|)$ holds. That is, the homomorphism maps each variable to a word in A^* such that each word equation is satisfied, and the lengths of these words satisfy the length constraint.

The *satisfiability problem* for word equations with length constraints asks, given a system of word equations and a length constraint, whether it has a solution.

We also consider the extension of the problem with regular constraints. For a system of word equations, a variable $x \in V$, and a regular language $\mathcal{L} \subseteq A^*$, a *regular constraint* $x \in \mathcal{L}$ imposes the additional restriction that any solution σ must satisfy $\sigma(x) \in \mathcal{L}$. Given a system of word equations, a length constraint, and a set of regular constraints, the satisfiability problem asks if there is a solution satisfying the word equation, the length constraints, as well as the regular constraints.

In the sequel, for clarity of exposition, we restrict our discussion to a system consisting of a single word equation.

Linear arithmetic with divisibility: Let $\mathcal{P}$ be a first-order language with equality, with binary relation symbol $\leq$, and with terms being linear polynomials with integer coefficients. We write $f(x)$, $g(x)$, etc., for terms in integer variables $x = x_1, \ldots, x_n$. Atomic formulas in Presburger arithmetic have the form $f(x) \leq g(x)$ or $f(x) = g(x)$. The language $\mathcal{PAD}$ of *Presburger arithmetic with divisibility* extends the language $\mathcal{P}$ with a binary relation $|$ (for divides). An atomic formula has the form $f(x) \leq g(x)$ or $f(x) = g(x)$ or $f(x)|g(x)$, where $f(x)$ and $g(x)$ are linear polynomials with integer coefficients. The full first order theory of $\mathcal{PAD}$ is undecidable, but the existential fragment is decidable [18, 23].

Note that the divisibility predicate $x|y$ is *not* expressible in Presburger arithmetic: a simple way to see this is that $\{(x, y) \in \mathbb{N}^2 \mid x|y\}$ is not a semi-linear set.

Counter systems: In this paper, we specifically use the term "counter systems" to mean counter systems with Presburger transition relations (e.g. see [3]). These more general transition relations can be simulated by standard Minsky's counter machines, but they are more useful for coming up with decidable subclasses of counter systems. A *counter system* $\mathcal{C}$ is a tuple (X, Q, Δ), where $X = \{x_1, \ldots, x_m\}$ is a finite set of counters, Q is a finite set of control states, and Δ is a finite set of transitions of the form $(q, \Phi(\bar{x}, \bar{x}'), q')$, where $q, q' \in Q$ and Φ is a Presburger formula with free variables $x_1, \ldots, x_m, x'_1, \ldots, x'_m$. A *configuration* of $\mathcal{C}$ is a tuple $(q, \mathbf{v}) \in Q \times \mathbb{N}^m$.

The semantics of counter systems is given as a transition system. A *transition system* is a tuple $\mathfrak{S} := \langle S; \rightarrow \rangle$, where S is a set of *configurations* and $\rightarrow \subseteq S \times S$ is a binary relation over S. A *path* in $\mathfrak{S}$ is a sequence $s_0 \rightarrow \cdots \rightarrow s_n$ of configurations $s_0, ..., s_n \in S$. If $S' \subseteq S$, let $pre^*(S')$ denote the set of $s \in S$ such that $s \rightarrow^* s'$ for some $s' \in S'$. We might write $pre^*_{\rightarrow}(S')$ to disambiguate the transition system.

A counter system $\mathcal{C}$ generates the transition system $\mathfrak{S}_{\mathcal{C}} = \langle S; \rightarrow \rangle$, where S is the set of all configurations of $\mathcal{C}$, and $(q, \mathbf{v}) \rightarrow (q', \mathbf{v}')$ if there exists a transition $(q, \Phi(\bar{x}, \bar{x}'), q') \in \Delta$ such that $\Phi(\mathbf{v}, \mathbf{v}')$ is true.

In the sequel, we will be needing the notion of flat counter systems [3, 4, 7, 20]. Given a counter system $\mathcal{C} = (X, Q, \Delta)$, the *control structure* of $\mathcal{C}$ is an edge-labeled directed graph $G = (V, E)$ with the set $V = Q$ of nodes and the set $E = \Delta$. The counter system $\mathcal{C}$ is *flat* if each node $v \in V$ is contained in at most one simple cycle.

3 Solving Quadratic Word Equations

We start by recalling a simple textbook recipe (Nielsen transformation, a.k.a., Levi's Method) [11, 19] for solving quadratic word equations, both for the cases with and without regular constraints. We then discuss the length abstractions of solutions to quadratic word equations, and provide a natural example that is not Presburger-definable.

3.1 Nielsen transformation

We will define a rewriting relation $E \Rightarrow E'$ between quadratic word equations E, E'. Let E be an equation of the form $\alpha w_1 = \beta w_2$ with $w_1, w_2 \in (A \cup V)^*$ and $\alpha, \beta \in A \cup V$. Then, there are several possible E':

- *Rules for erasing an empty prefix variable*. These rules can be applied if $\alpha \in V$ (symmetrically, $\beta \in V$). We nondeterministically guess that α be the empty word ϵ, i.e., E' is $(w_1 = \beta w_2)[\epsilon/\alpha]$. The symmetric case of $\beta \in V$ is similar.
- *Rules for removing a nonempty prefix*. These rules are applicable if each of α and β is either a constant or a variable that we nondeterministically guess to be a nonempty word. There are several cases:
  - **(P1)** $\alpha \equiv \beta$ (syntactic equality). In this case, E' is $w_1 = w_2$.
  - **(P2)** $\alpha \in A$ and $\beta \in V$. In this case, E' is $w_1[\alpha\beta/\beta] = \beta(w_2[\alpha\beta/\beta])$. In the sequel, to avoid notational clutter we will write $\beta w_2[\alpha\beta/\beta]$ instead of $\beta(w_2[\alpha\beta/\beta])$.
  - **(P3)** $\alpha \in V$ and $\beta \in A$. In this case, E' is $\alpha(w_1[\beta\alpha/\alpha]) = w_2[\beta\alpha/\alpha]$.
  - **(P4)** $\alpha, \beta \in V$. In this case, we nondeterministically guess if $\alpha \preceq \beta$ or $\beta \preceq \alpha$. In the former case, the equation E' is $w_1[\alpha\beta/\beta] = \beta(w_2[\alpha\beta/\beta])$. In the latter case, the equation E' is E' is $\alpha(w_1[\beta\alpha/\alpha]) = w_2[\beta\alpha/\alpha]$.

Note that the transformation keeps an equation quadratic.

Proposition 1. *E is solvable iff $E \Rightarrow^* (\epsilon = \epsilon)$. Furthermore, checking if E is solvable is in PSPACE.*

See [11] for a proof. Roughly speaking, the proof uses the fact that each step either decreases the size of the equation, or the length of a length-minimal solution. It runs in PSPACE because each rewriting does not increase the size of the equation.

3.2 Handling regular constraints

Nielsen transformation easily extends to quadratic word equations with regular constraints (e.g. see [12]). We assume that a regular constraint $x \in \mathcal{L}$ is given as an NFA $\mathcal{A}_{p,q}$ representing $\mathcal{L}$. If q_0 and q_F are the initial and final states (respectively) of an NFA $\mathcal{A}$, we can be more explicit and write $\mathcal{A}_{q_0,q_F}$ instead of $\mathcal{A}$.

Our rewriting relation $\Rightarrow$ now works over a pair consisting of an equation E and a set S of regular constraints over variables in E. Let E be an equation of the form $\alpha w_1 = \beta w_2$ with $w_1, w_2 \in (A \cup V)^*$ and $\alpha, \beta \in A \cup V$. We now define $(E, S) \Rightarrow (E', S')$ by extending the pervious definition of $\Rightarrow$ without regular constraints. Firstly, we make

sure that S is satisfiable by a standard automata-theoretic algorithm, which can be done in PSPACE. In particular, it has to be the case that $E \Rightarrow E'$ and additionally do the following:

- *Rules for erasing an empty prefix variable* α. When applied, ensure that each regular constraint $\alpha \in \mathcal{L}$ in S satisfies $\epsilon \in \mathcal{L}$. Define S' as S minus all regular constraints of the form $\alpha \in \mathcal{L}$.
- *Rules for removing a nonempty prefix*. For (P1), we set S' to be S minus all constraints of the form $\alpha \in \mathcal{L}$ if α is a variable. For (P2)–(P4), assume that E' is $w_1[\alpha\beta/\beta] = \beta(w_2[\alpha\beta/\beta])$; the other case is symmetric. For each regular constraint $\beta \in \mathcal{L}(\mathcal{A}_{p,q})$, we nondeterministically guess a state r, and add $\alpha \in \mathcal{L}(\mathcal{A}_{p,r})$ and $\beta \in \mathcal{L}(\mathcal{A}_{r,q})$ to S'. In the case when $\alpha \in A$, we could immediately perform the check $\alpha \in \mathcal{L}(\mathcal{A}_{p,r})$: a positive outcome implies removing this constraint from S', while on a negative outcome our algorithm simply fails on this branch. For any variable y that is distinct from β, we add all regular constraints $y \in L$ in S to S'. If α still occurs in E', add regular constraints $\alpha \in L$ in S to S'. If S' is unsatisfiable, fail on this this branch.

Proposition 2. *(E, S) is solvable iff $(E, S) \Rightarrow^* (\epsilon = \epsilon, \emptyset)$. Furthermore, checking if (E, S) is solvable is in PSPACE.*

Note that this is still a PSPACE algorithm because it never creates a new NFA or adds new states to existing NFA in the regular constraints, but rather adds a regular constraint $x \in \mathcal{L}(\mathcal{A}_{p,q})$ to a variable x, where $\mathcal{A}$ is an NFA that is already in the regular constraint.

3.3 Generating all solutions using Nielsen transformation

One result that we will need in this paper is that Nielsen transformation is able to *generate all solutions* of quadratic word equations with regular constraints. To clarify this, we extend the definition of $\Rightarrow$ so that each a configuration E or (E, S) in the graph of $\Rightarrow$ is also annotated by an assignment σ of the variables in E to concrete strings. We write $E_1[\sigma_1] \Rightarrow E_2[\sigma_2]$ if $E_1 \Rightarrow E_2$ and σ_2 is the modification from σ_1 according to the operation used to obtain E_2 from E_1. Observe that the domain of σ_2 is a subset of the domain of σ_1; in fact, some rules (e.g., erasing an empty prefix variable) could remove a variable in the prefix in E_1 from σ_1. The following example illustrates how $\Rightarrow$ works with this extra annotated assignment. Suppose that $\sigma_1(x) = ab$ and $\sigma_1(y) = abab$ and $E_1 := xy = yx$ and E_2 is obtained from E_1 using rule (P4), i.e., substitute xy for y. In this case, $\sigma_2(x) = \sigma_2(y) = \sigma_1(x) = ab$. Observe that $E_2[\sigma_2] \Rightarrow E_3[\sigma_3] \Rightarrow E_4[\sigma_4]$, where $E_3 := E_2$, $\sigma_3(x) = ab$, $\sigma_3(y) = \epsilon$, $E_4 := x = x$, and $\sigma_4(x) = ab$. The definition for the case with regular constraints is identical.

Proposition 3. *$(E, S)[\sigma] \rightarrow^* (\epsilon = \epsilon, \emptyset)[\sigma']$ where σ' has the empty domain iff σ is a solution of (E, S).*

This proposition immediately follows from the proof of correctness of Nielsen transformation for quadratic word equations (cf. [11]).

3.4 Length abstractions and semilinearity

Given a quadratic word equation E with constants A and variables $V = \{x_1, \ldots, x_k\}$, its *length abstraction* is defined as follows

$$\textsc{Len}(E) = \{(|\sigma(x_1)|, \ldots, |\sigma(x_k)|) : \sigma \text{ is a solution to } E\},$$

namely the set of tuples of numbers corresponding to lengths of solutions to E.

Example 1. Consider the quadratic equation $E := \ xaby = yz$, where $V = \{x, y, z\}$ and A contains at least two letters a and b. We will show that its length abstraction $\textsc{Len}(E)$ can be captured by the Presburger formula $|z| = |x| + 2$. Observe that each $(n_x, n_y, n_z) \in \textsc{Len}(E)$ must satisfy $n_z = n_x + 2$ by a length argument on E. Conversely, we will show that each triple $(n_x, n_y, n_z) \in \mathbb{N}^3$ satisfying $n_z = n_x + 2$ must be in $\textsc{Len}(E)$. To this end, we will define a solution σ to E such that $(|\sigma(x)|, |\sigma(y)|, |\sigma(z)|) = (n_x, n_y, n_z)$. Consider $\sigma(x) = a^{n_x}$. Then, for some $q \in \mathbb{N}$ and $r \in [n_x + 1]$, we have $n_y = q(n_x + 2) + r$. Let w be a prefix of $\sigma(x)ab$ of length r. Therefore, for some v, we have $wv = \sigma(x)ab$. Define $\sigma(y) = (\sigma(x)ab)^q w$. We then have $\sigma(x)ab\sigma(y) = \sigma(y)vw$. Thus, setting $\sigma(z) = vw$ gives us a satisfying assignment for E which satisfies the desired length constraint. □

However, it turns out that Presburger Arithmetic is not sufficient for capturing length abstractions of quadratic word equations.

Theorem 1. *There is a quadratic word equation whose length abstraction is not Presburger-definable.*

To this end, we show that the length abstraction of $xaby = yabx$, where $a, b \in A$ and $x, y \in V$, is not Presburger definable.

Lemma 1. *The length abstraction* $\textsc{Len}(xaby = yabx)$ *coincides with tuples* $(|x|, |y|)$ *of numbers satisfying the expression* $\varphi(|x|, |y|)$ *defined as:*

$$\begin{aligned} |x| = |y| \vee (|x| = 0 \wedge |y| \equiv 0 \pmod 2) \vee (|y| = 0 \wedge |x| \equiv 0 \pmod 2) \\ \vee (|x|, |y| > 0 \wedge \gcd(|x| + 2, |y| + 2) > 1) \end{aligned}$$

Observe that this would imply non-Presburger-definability: for otherwise, since the first three disjuncts are Presburger-definable, the last disjunct would also be Presburger-definable, which is not the case since the property that two numbers are relatively prime is not Presburger-definable. Let us prove this lemma. Let $S = \textsc{Len}(xaby = yabx)$. We first show that given any numbers n_x, n_y satisfying $\varphi(n_x, n_y)$, there are solutions σ to $xaby = yabx$ with $|\sigma(x)| = n_x$ and $|\sigma(y)| = n_y$. If they satisfy the first disjunct in φ (i.e., $n_x = n_y$), then set $\sigma(x) = \sigma(y)$ to an arbitrary word $w \in A^{n_x}$. If they satisfy the second disjunct, then $aby = yab$ and so set $\sigma(x) = \epsilon$ and $\sigma(y) \in (ab)^*$. The same goes with the third disjunct, symmetrically. For the fourth disjunct (assuming the first three disjuncts are false), let $d = \gcd(n_x + 2, n_y + 2)$. Define $\sigma(x), \sigma(y) \in (a^{d-1}b)^*(a^{d-2})$ so that $|\sigma(\alpha)| = n_\alpha$ for $\alpha \in V$. It follows that $\sigma(x)ab\sigma(y) = \sigma(y)ab\sigma(x)$.

We now prove the converse. So, we are given a solution σ to $xaby = yabx$ and let $u := \sigma(x)$, $v := \sigma(y)$. Assume to the contrary that $\varphi(|u|, |v|)$ is false and that u and v are the shortest such solutions. We have several cases to consider:

- $u = v$. Then, $|u| = |v|$, contradicting that $\varphi(|u|, |v|)$ is false.
- $u = \epsilon$. Then, $abv = vab$ and so $v \in (ab)^*$, which implies that $|v| \equiv 0 \pmod 2$. Contradicting that $\varphi(|u|, |v|)$ is false.
- $v = \epsilon$. Same as previous item and that $|u| \equiv 0 \pmod 2$.
- $|u| > |v| > 0$. Since $\varphi(|u|, |v|)$ is false, we have $\gcd(|u|+2, |v|+2) = 1$. It cannot be the case that $|u| = |v| + 1$ since then, comparing prefixes of $uabv = vabu$, the letter at position $|u|+2$ would be b on l.h.s. and a on r.h.s., which is a contradiction. Therefore $|u| \geq |v| + 2$. Let $u' = u[|v| + 3, |u|]$, i.e., u but with its prefix of length $|v| + 2$ removed. By Nielsen transformation, we have $u'abv = vabu'$. It cannot be the case that $u' = \epsilon$; for, otherwise, $abv = vab$ implies $v \in (ab)^*$ and so $u = vab$, implying that 2 divides both $|u| + 2$ and $|v| + 2$, contradicting that $\gcd(|u| + 2, |v| + 2) = 1$. Therefore, $|u'| > 0$. Since $\gcd(|u'| + 2, |v| + 2) = \gcd(|u|+2, |v|+2) = 1$, we have a shorter solution to $xaby = yabx$, contradicting minimality.
- $|v| > |u| > 0$. Same as previous item.

4 Reduction to Counter Systems

In this section, we will provide an algorithm for computing a counter system from (E, S), where E is a quadratic word equation and S is a set of regular constraints. We will first describe this algorithm for the case without regular constraints, after which we show the extension to the case with regular constraints.

Given the quadratic word equation E, we show how to compute a counter system $\mathcal{C}(E) = (X, Q, \Delta)$ such that the following theorem holds.

Theorem 2. *The length abstraction of E coincides with*

$$\{v \in \mathbb{N}^{|V|} \mid (E, v) \in pre^*_{\mathcal{C}(E)}(\{\epsilon = \epsilon\} \times \mathbb{N}^{|V|})\}$$

Before defining $\mathcal{C}(E)$, we define some notation. Define the following formulas:

- $\texttt{ID}(\bar{x}, \bar{x}') := \bigwedge_{x \in \bar{x}} x' = x$
- $\texttt{SUB}_{y,z}(\bar{x}, \bar{x}') := z \leq y \wedge y' = y - z \wedge \bigwedge_{x \in \bar{x}, x \neq y} x' = x$
- $\texttt{DEC}_y(\bar{x}, \bar{x}') := y > 0 \wedge y' = y - 1 \wedge \bigwedge_{x \in \bar{x}, x \neq y} x' = x$

Note that the $\neq$ symbol in the guard of $\bigwedge$ denotes syntactic equality (i.e. not equality in Preburger Arithmetic). We omit mention of the free variables $\bar{x}$ and $\bar{x}'$ when they are clear from the context.

We now define the counter system. Given a quadratic word equation E with constants A and variables V, we define a counter system $\mathcal{C}(E) = (X, Q, \Delta)$ as follows. The counters X will be precisely all variables that appear in E, i.e., $X := V$. The control states are precisely all equations E' that can be rewritten from E using Nielsen transformation, i.e., $Q := \{E' : E \Rightarrow^* E'\}$. The set Q is finite (at most exponential in $|E|$) as per our discussion in the previous section.

We now define the transition relation Δ. We use $\bar{x}$ to enumerate V in some order. Given $E_1 \Rightarrow E_2$ with $E_1, E_2 \in Q$, we then add the transition $(E_1, \Phi(\bar{x}, \bar{x}'), E_2)$, where Φ is defined as follows:

– If $E_1 \Rightarrow E_2$ applies a rule for erasing an empty prefix variable $y \in \bar{x}$, then $\Phi := y = 0 \wedge \mathtt{ID}$.
– If $E_1 \Rightarrow E_2$ applies a rule for removing a nonempty prefix:
  • If (P1) is applied, then $\Phi = \mathtt{ID}$.
  • If (P2) is applied, then $\Phi = \mathtt{DEC}_\beta$.
  • If (P3) is applied, then $\Phi = \mathtt{DEC}_\alpha$.
  • If (P4) is applied and $\alpha \preceq \beta$, then $\Phi = \mathtt{SUB}_{\beta,\alpha}$. If $\beta \preceq \alpha$, then $\Phi = \mathtt{SUB}_{\alpha,\beta}$.

Observe that if $(E_1, \mathbf{v}_1) \rightarrow (E_2, \mathbf{v}_2)$, then $|E_1| \leq |E_2|$ and $\mathbf{v}_1 \preceq \mathbf{v}_2$. In addition, if $\mathbf{v}_1 = \mathbf{v}_2$, then $|E_1| < |E_2|$. This implies the following lemma.

Lemma 2. *The counter system $\mathcal{C}(E)$ terminates from every configuration $(E_0, \mathbf{v}_0)$.*

The proof of Theorem 2 immediately follows from Proposition 3 that Nielsen transformation generates all solutions.

Extension to the case with regular constraints: In this extension, we will only need to assert that the counter values belong to the length abstractions of the regular constraints, which are effectively semilinear due to Parikh's Theorem [27]. Given a quadratic word equation E with a set S of regular constraints, we define the counter system $\mathcal{C}(E, S) = (X, Q, \Delta)$ as follows. Let $\mathcal{C}(E) = (X_1, Q_1, \Delta_1)$ be the counter system from the previous paragraph, obtained by ignoring the regular constraints. We define $X = X_1$. Let Q be the finite set of all configurations reachable from (E, S), i.e., $Q = \{(E', S') : (E, S) \Rightarrow^* (E', S')\}$. Given $(E_1, S_1) \Rightarrow (E_2, S_2)$, we add the transition $((E_1, S_2), \Phi(\bar{x}, \bar{x}'), (E_2, S_2))$ as follows. Suppose that $(E_1, \Phi'(\bar{x}, \bar{x}'), E_2)$ was added to Δ_1 by $E_1 \Rightarrow E_2$. Then,

$$\Phi := \Phi' \wedge \bigwedge_{x \in \bar{x}} \left(x \in \textsc{Len}(\bigcap_{(x \in L) \in S} L) \wedge x' \in \textsc{Len}(\bigcap_{(x \in L) \in S'} L) \right).$$

The size of the NFA for $\bigcap_{(x \in L) \in S} L$ is exponential in the number of constraints of the form $(x \in L)$ in S (of which there are polynomially many). The constraint $x \in \textsc{Len}(L)$ is well-known to be effectively semilinear [27]. In fact, using the algorithm of Chrobak-Martinez [8, 25, 29], we can compute in polynomial time two finite sets A, A' of integers and an integer b such that, for each $n \in \mathbb{N}$, $n \in U := A \cup (A' + b\mathbb{N})$ is true iff $n \in \textsc{Len}(L)$. Note that U is a finite union of arithmetic progressions (with period 0 and/or b). In fact, each number $a \in A \cup A'$ (resp. the number b) is at most quadratic in the size of the NFA, and so it is a polynomial [4] size even when they are written in unary. Therefore, treating U as an existential Presburger formula $\varphi(x)$ with one free variable (an existential quantifier is needed to guess the coefficient n such that $x = a_i + bn$ for some i), the resulting Φ' is a polynomial-sized existential Presburger formula.

Theorem 3. *The length abstraction of (E, S) coincides with*

$$\{v \in \mathbb{N}^{|V|} \mid ((E, S), v) \in pre^*_{\mathcal{C}(E,S)}(\{(\epsilon = \epsilon, \emptyset)\} \times \mathbb{N}^{|V|})\}$$

[4] Note that we mean polynomial in the size of the NFA, which can be exponential in $|S|$.

As for the case without regular constraints, the proof of Theorem 2 immediately follows from Proposition 3 that Nielsen transformation generates all solutions.

5 Decidability via Linear Arithmetic with Divisibility

5.1 Accelerating a 1-variable-reducing cycle

Consider a counter system $\mathcal{C} = (X, Q, \Delta)$ with $Q = \{q_0, \ldots, q_{n-1}\}$, such that for some $y \in X$ the transition relation Δ consists of precisely the following transition $(q_i, \Phi_i, q_{i+1 \pmod n})$, for each $i \in [n-1]$, and each Φ_i is either $\mathtt{SUB}_{y,z}$ (with z a variable distinct from y) or $\mathtt{DEC}_y$. Such a counter system is said to be a *1-variable-reducing cycle*.

Lemma 3. *There exists a polynomial-time algorithm which given a 1-variable-reducing cycle $\mathcal{C} = (X, Q, \Delta)$ and two states $p, q \in Q$ computes an formula $\varphi_{p,q}(\bar{x}, \bar{x}')$ in existential Presburger arithmetic with divisibility such that $(p, \mathbf{v}) \to_{\mathcal{C}}^* (q, \mathbf{w})$ iff $\varphi_{p,q}(\mathbf{v}, \mathbf{w})$ is satisfiable.*

This lemma can be seen as a special case of the acceleration lemma for flat parametric counter automata [7] (where all variables other than y are treated as parameters). However, its proof is in fact quite simple. Without loss of generality, we assume that $q = q_0$ and $p = q_i$, for some $i \in \mathbb{N}$. Any path $(q_0, \mathbf{v}) \to_{\mathcal{C}}^* (q_i, \mathbf{w})$ can be decomposed into the cycle $(q_0, \mathbf{v}) \to^* (q_0, \mathbf{v}')$ and the simple path $(q_0, \mathbf{w}_0) \to \cdots \to (q_i, \mathbf{w}_i)$ of length i. Therefore, the reachability relation $(q_0, \mathbf{x}) \to_{\mathcal{C}}^* (q_i, \mathbf{y})$ can be expressed as

$$\exists \mathbf{z}_0, \cdots, \mathbf{z}_{i-1} : \varphi_{q_0,q_0}(\mathbf{x}, \mathbf{z}_0) \wedge \Phi_0(\mathbf{z}_0, \mathbf{z}_1) \wedge \cdots \wedge \Phi_{i-1}(\mathbf{z}_{i-1}, \mathbf{y}).$$

Thus, it suffices to show that $\varphi_{q_0,q_0}(\mathbf{x}, \mathbf{x}')$ is expressible in $\mathcal{PAD}$. Consider a linear expression $M = a_0 + \sum_{x \in X \setminus \{y\}} a_x x$, where a_0 is the number of instructions i in the cycle such that $\Phi_i = \mathtt{DEC}_y$ and a_x is the number of instructions i such that $\Phi_i = \mathtt{SUB}_{y,x}$. Each time around the cycle, y decreases by M. Thus, for some $n \in \mathbb{N}$ we have $y' = y - nM$, or equivalently

$$nM = y - y'$$

The formula φ_{q_0,q_0} can be defined as follows:

$$\varphi_{q_0,q_0} := \; M \mid (y - y') \wedge y' \leq y \wedge \bigwedge_{x \in X \setminus \{y\}} x' = x.$$

Handling unary Presburger guards: Recalling our reduction for the case with regular constraints from Section 4 reveals that we also need unary Presburger guards on the counters. We will show how to extend Lemma 3 to handle such guards. As we will see shortly, we will need a bit of the theory of semilinear sets.

As before, our counter system $\mathcal{C} = (X, Q, \Delta)$ has $Q = \{q_0, \ldots, q_{n-1}\}$, and the control structure is a simple cycle of length n, i.e., the transitions in Δ are precisely $(q_i, \Phi_i, q_{i+1 \pmod n})$ for some Presburger formula $\Phi_i(\bar{x}, \bar{x}')$, for each $i \in [n-1]$. We say that $\mathcal{C}$ is *1-variable-reducing with unary Presburger guards* if there exists a counter

$y \in X$ such that each Φ_i is of the form $\theta_i \wedge \psi_i$, where θ_i is either $\mathtt{SUB}_{y,z}$ (with z a variable distinct from y) or $\mathtt{DEC}_y$, and ψ_i is a conjunction of formulas of the form $x \in A_i \cup (A'_i + b\mathbb{N})$, where both A_i and A'_i are finite sets of natural numbers and $x \in X$. For each counter $x \in X$, we use $\psi_{i,x}$ to denote the set of conjuncts in ψ_i that refers to the counter x.

Lemma 4. *There exists a polynomial-time algorithm which given a 1-variable-reducing cycle with unary Presburger guards $\mathcal{C} = (X, Q, \Delta)$ and two states $p, q \in Q$ computes an formula $\lambda_{p,q}(\bar{x}, \bar{x}')$ in existential Presburger arithmetic with divisibility such that $(p, \mathbf{v}) \rightarrow_{\mathcal{C}}^* (q, \mathbf{w})$ iff $\lambda_{p,q}(\mathbf{v}, \mathbf{w})$ is satisfiable.*

Unlike Lemma 3, this lemma does not immediately follow from the results of [7] on flat parametric counter automata. To prove this, let us first take the formula $\varphi_{p,q}(\bar{x}, \bar{x}')$ from Lemma 3 applied to $\mathcal{C}'$, which is obtained from $\mathcal{C}$ by first removing the unary Presburger guards. We can insert these unary Presburger guards to $\varphi_{p,q}$, but this is not enough because we need to make sure that all "intermediate" values of y have to also satisfy the Presburger guards corresponding to y on that control state. More precisely, let the counter decrement in θ_i be α_i (which can either be a variable x distinct from y or 1). Write $f(\bar{x}) = \sum_{i=0}^{n-1} \alpha_i$. Then, we can write

$$\lambda_{q_0,q_0} := \bar{x}' = \bar{x} \vee \left(\varphi_{q_0,q_0} \wedge \bigwedge_{i=0}^{n-1} \psi_i(\bar{x}) \wedge \eta_{q_0,q_0} \right)$$

$$\eta_{q_0,q_0} := \forall k : y' + (k+1)f(\bar{x}) \leq y \longrightarrow \left(\bigwedge_{i=0}^{n-1} \bigwedge_{(\alpha_i \in A \cup A' + b\mathbb{N}) \in \psi_{i,y}} y' + kf(\bar{x}) + \alpha_i \in A \cup (A' + b\mathbb{N}) \right)$$

Owing to the constraint φ_{q_0,q_0}, the premise $y' + (k+1)f(\bar{x}) \leq y$ in η_{q_0,q_0} could have been rewritten to $y' + (k+1)f(\bar{x}) = y$. As we shall soon see, the former will be more useful for completing our proof of Lemma 4. The formula λ_{q_0,q_0} is a correct expression that captures the reachability relation $(q_0, \mathbf{w}) \rightarrow_{\mathcal{C}}^* (q_0, \mathbf{w}')$, but the problem is that it has a universal quantifier and therefore is not a formula of existential Presburger arithmetic with divisibility. To fix this problem, we will need to exploit the semilinear structure of unary Presburger guards. To this end, we first notice that, by taking the big conjunction over i and the big conjuncton over α_i out, the formula η_{q_0,q_0} is equivalent to:

$$\eta_{q_0,q_0} \equiv \bigwedge_{i=0}^{n-1} \bigwedge_{(\alpha_i \in A \cup A' + b\mathbb{N}) \in \psi_{i,y}} \forall k : y' + (k+1)f(\bar{x}) \leq y \longrightarrow (y' + kf(\bar{x}) + \alpha_i \in A \cup (A' + b\mathbb{N}))$$

Therefore, it suffices to rewrite each conjunct $C(\bar{x}) := \forall k : y' + (k+1)f(\bar{x}) \leq y \longrightarrow (y' + kf(\bar{x}) + \alpha_i \in A \cup (A' + b\mathbb{N})$ as an existential Presburger formula, for each i and constraint $(\alpha_i \in A \cup A' + b\mathbb{N})$. To this end, let $a := \max A$ and let N denote $|A'|$. We

claim that φ_{q_0,q_0} entails

$$C(\bar{x}) \Leftrightarrow \bigwedge_{i=0}^{a} y' + (i+1)f(\bar{x}) \leq y \rightarrow y' + if(\bar{x}) + \alpha_i \in A \cup (A' + b\mathbb{N})$$
$$\wedge \bigwedge_{i=a+1}^{a+N+1} y' + (i+1)f(\bar{x}) \leq y \rightarrow y' + if(\bar{x}) + \alpha_i \in A' + b\mathbb{N}.$$

Simply put, we distinguish the cases when $y' + if(\bar{x}) + \alpha_i$ is "small" (i.e., less than the maximum threshold that can keep this number in an arithmetic progression with 0 period), and when this number is "big" (i.e. must be in an arithmetic progression with a nonzero period). To prove this equivalence, it suffices to show that if $y' + kf(\bar{x}) + \alpha_i \notin A \cup (A' + b\mathbb{N})$ with $k > a + N + 1$ and $y' + (k+1)f(\bar{x}) \leq y$ (i.e. $y' + kf(\bar{x}) + \alpha_i \leq y$ since $y' = y + hf(\bar{x})$ for some h because of φ_{q_0,q_0}), then we can find $k' \leq a + N + 1$ such that $y' + k'f(\bar{x}) + \alpha_i \notin A \cup (A' + b\mathbb{N})$. Suppose to the contrary that such k' does not exist. Then, since there are $N + 1$ numbers in between $a + 1$ and $a + N + 1$, by pigeonhole principle there is an arithmetic progression $a' + b\mathbb{N}$ and two different numbers $a + 1 \leq j_1 < j_2 \leq a + N + 1$ such that $y' + j_h f(\bar{x}) + \alpha_i \in a' + b\mathbb{N}$, for $h = 1, 2$. Let $d := (j_2 - j_1)$. Note that $df(\bar{x})$ denotes the difference between $y' + j_1 f(\bar{x}) + \alpha_i$ and $y' + j_2 f(\bar{x}) + \alpha_i$, and this difference is of the form mb, for some positive integer m. We now find a number $j \in [a + 1, a + N]$ with $j + qd = k$ for some positive integer q. Since $y' + jf(\bar{x}) + \alpha_i \in a'' + b\mathbb{N}$ for some $a'' \in A'$, it must be the case that $y' + (j + qd)f(\bar{x}) + \alpha_i \in a'' + b\mathbb{N}$ for $q \in \mathbb{N}$, contradicting that $y' + kf(\bar{x}) + \alpha_i \notin A \cup (A' + b\mathbb{N})$.

We have proven correctness, and what remains is to analyse the size of the formula λ_{q_0,q_0}. To this end, it suffices to show that each formula $C(\bar{x})$ is of polynomial size. This is in fact the case since there are at most polynomially many numbers in A and A' and that the size of all numbers in $A \cup A' \cup \{b\}$ are of polynomial size even when they are written in unary.

5.2 An extension to flat control structures and an acceleration scheme

The following generalisation to flat control structures is an easy corollary of Lemma 3 and 4.

Theorem 4. *There exists a polynomial-time algorithm which, given a flat Presburger counter system $\mathcal{C} = (X, Q, \Delta)$, each of whose simple cycle is 1-variable-reducing with unary Presburger guards and two states $p, q \in Q$, computes an formula $\lambda_{p,q}(\bar{x}, \bar{x}')$ in existential Presburger with divisibility such that $(p, \mathbf{v}) \rightarrow_{\mathcal{C}}^{*} (q, \mathbf{w})$ iff $\lambda_{p,q}(\mathbf{v}, \mathbf{w})$ is satisfiable.*

Indeed, to prove this theorem, we can simply use Lemma 4 to accelerate all cycles and the fact that transition relations expressed in existential Presburger with divisibility is closed under composition.

5.3 Application to word equations with length constraints

Theorem 4 gives rise to a simple and sound (but not complete) technique for solving quadratic word equations with length constraints: given a quadratic word equation (E, S) with regular constraints, if the counter system $\mathcal{C}(E, S)$ is flat, each of whose simple cycle is 1-variable-reducing with unary Presburger guards, then apply the decision procedure from Theorem 4. In this section, we show completeness of this method for the class of regular-oriented word equations recently defined in [10], which can be extended with regular constraints given as 1-weak NFA [2]. A word equation is *regular* if each variable $x \in V$ occurs at most once on each side of the equation. Observe that $xy = yx$ is regular, but $xxyy = zz$ is not. It is easy to see that a regular word equation is quadratic. A word equation $L = R$ is said to be *oriented* if there is a total ordering $<$ on V such that the occurrences of variables on each side of the equation preserve $<$, i.e., if $w = L$ or $w = R$ and $w = w_1\alpha w_2\beta w_3$ for some $w_1, w_2, w_3 \in (A \cup V)^*$ and $\alpha, \beta \in V$, then $\alpha < \beta$. Observe that $xy = yz$ (i.e. that x and z are conjugates) is oriented, but $xy = yx$ is not oriented. It was shown in [10] that the satisfiability for regular-oriented word equations is NP-hard. We show satisfiability for this class with length constraints is decidable.

Theorem 5. *The satisfiability problem of regular-oriented word equations with length constraints is decidable in nondeterministic exponential time.*

This decidability (in fact, an NP upper bound) for the *strictly regular-ordered* subcase, in which each variable occurs precisely once on each side, was proven in [9]. For this subcase, it was shown that Presburger Arithmetic is sufficient, but the decidability for the general class of regular-oriented word equations with length constraints remained open. Theorem 5 shows the problem is decidable.

We start with a simple lemma that $\Rightarrow$ preserves regular-orientedness. Its proof can be found in the full version.

Lemma 5. *If $E \Rightarrow E'$ and E is regular-oriented, then E' is also regular-oriented.*

Next, we show a bound on the lengths of cycles and paths of the counter system associated with a regular-oriented word equation.

Lemma 6. *Given a regular-oriented word equation E, the counter system $\mathcal{C}(E)$ is flat. Moreover, the length of each simple cycle (resp. path) in the control structure of $\mathcal{C}(E)$ is of length $O(|E|)$ (resp. $O(|E|^2)$).*

Let $E := L = R$. We first show that the length of a simple cycle in the control structure of $\mathcal{C}(E)$ is of length at most $N = \max\{|L|, |R|\} - 1$. Given a simple cycle $E_0 \Rightarrow E_1 \Rightarrow \cdots \Rightarrow E_n$ with $n > 0$ (i.e. $E_0 = E_n$ and $E_i \neq E_j$ for all $0 \leq i < j < n$), it has to be the case that each rewriting in this cycle applies one of the (P2)–(P4) rules since the other rules reduce the size of the equation. We have $|E_0| = |E_1| = \cdots = |E_n|$. Let $E_i := L_i = R_i$ with $L_i = \alpha_i w_i$ and $R_i = \beta_i w_i'$. Let us assume that E_1 be $w_0[\alpha_0\beta_0/\beta_0] = \beta_0 w_0'[\alpha_0\beta_0/\beta_0]$; the case with E_1 be $\alpha_0 w_0[\beta_0\alpha_0/\alpha_0] = w_0'[\beta_0\alpha_0/\alpha_0]$ will be easily seen to be symmetric. This assumption implies that β_0 is a variable y, and that $L_0 = uyv$ for some words $u, v \in (A \cup V)^*$ (for, otherwise, $|E_1| < |E_0|$

because of regularity of E). Furthermore, it follows that, for each $i \in [n-1]$, E_{i+1} is $w_i[\alpha_i y/] = y w_i'$ and $\beta_i = y$, i.e., the counter system $\mathcal{C}(E)$ applies either $\mathtt{SUB}_{y,x}$ (in the case when $x = \alpha_i$) or $\mathtt{DEC}_y$ (in the case when $\alpha_i \in A$). For, otherwise, taking a minimal $i \in [1, n-1]$ with E_{i+1} being $\alpha_i w_i[y\alpha_i/\alpha_i] = w_i'[y\alpha_i/\alpha_i]$ for some variable $x = \alpha_i$ shows that E_i is of the form $x...y... = y...x...$ (since $|E_{i+1}| = |E_i|$) contradicting that E_i is oriented. Consequently, we have

- $R_i = R_j$ for all i, j, and
- $L_i = \mathrm{cyc}^i(u)yv$ for all $i \in [n]$

implying that the length of the cycle is at most $|L_0| - 1 \leq |L| - 1$.

Consider the control structure $\mathcal{C}(E)$ as a dag of SCCs. In this dag, each edge from one SCC to the next is size-reducing. Therefore, the maximal length of a path in this dag is $|E|$. Therefore, since the maximal path of each SCC is N (from the above analysis), the maximal length of a simple path in the control structure is at most N^2.

Handling regular constraints: First, we note that the length abstraction of regular-oriented word equations with regular constraints is already not Presburger-definable in general (see full version for proof):

Proposition 4. *The regular-oriented word equation $xy = yz$ over the alphabet $\{a, b, \#\}$, together with regular constraints $x, y \in \#(a+b)^*$ has non-Presburger-definable length abstraction.*

It is difficult to extend Theorem 5 to the case with regular constraints because they may introduce nestings of cycles (which breaks the flat control structure) even for regular-oriented word equation. However, we can show that restricting to regular constraints given by *1-weak NFA* [2] (i.e. a dag of SCCs, each with at most one state) preserves the flat control structure. A 1-weak regular constraint is of the form $x \in L$ where L is accepted by a 1-weak NFA. The class of 1-weak automata is in fact quite powerful, e.g., when considered as recognisers of languages of ω-words, they capture the subclass of LTL with operators **F** and **G** [2]. They have also been used to obtain a decidable extension of infinite-state concurrent systems in term rewriting systems, e.g., see [17, 30]. Note that the regular constraint in Proposition 4 is accepted by a 1-weak NFA: the NFA has two states q_0 and q_1, and transitions $q_0 \xrightarrow{\#} q_1$ and $q_1 \xrightarrow{a,b} q_1$, where q_0 is an initial state and q_1 a final state.

Theorem 6. *The satisfiability problem of regular-oriented word equations with 1-weak regular constraints and length constraints is solvable in nondeterministic double exponential time (2NEXP).*

Let us prove this theorem. Suppose E is a regular-oriented word equation with the set S of 1-weak regular constraints. Let $\mathcal{C}(E, S) = (X, Q, \Delta)$ be the corresponding counter system. Let $M(S)$ denote the maximum number of states ranging over all NFA in S.

Lemma 7. *The counter system $\mathcal{C}(E, S)$ is flat. Moreover, the length of each simple cycle in the control structure of $\mathcal{C}(E, S)$ is of length $O(|E|)$, while the length of each simple path is of length $O(|E|^2|V||S|M(S)^3)$.*

By virtue of Theorem 4, this lemma implies decidability of Theorem 6, but it does NOT imply the nondeterministic exponential time upper bound since each unary Presburger guard in $\mathcal{C}(E)$ will be of the form $x \in \textsc{Len}(\bigcap_{(x \in L) \in S} L)$. Even though we know that $|S|$ is always of a polynomial size, their intersection requires performing a product automata construction, which will result in an NFA of an exponential size. Therefore, we obtain a nondeterministic double exponential time complexity upper bound (2NEXP), instead of NEXP as for the case without regular constraints. The proof of Lemma 7 can be found in the full version.

Remark 1. Our proof of Theorem 6 does not extend to the case when we allow *generalised flat* NFA (i.e. after mapping all the letters in A to a new symbol '?', the control structure of the NFA is flat) in the regular constraints. This is because a simple cycle involving two or more states will result in a counter system that is no longer flat.

6 Future Work

One research direction is to study extensions of our techniques to deal with the class of regular (but not necessarily oriented) word equations with length constraints. We believe that this is a key subproblem of the general class of quadratic word equations with length constraints. We also conjecture that the length abstractions of general quadratic word equations can be effectively captured by existential Presburger with divisibility.

Acknowledgment. We thank Jatin Arora, Dmitry Chistikov, Volker Diekert, Matthew Hague, Artur Jeż, Philipp Rümmer, and James Worrell for the helpful discussions. This research was partially funded by the ERC Starting Grant AV-SMP (grant agreement no. 759969) and the ERC Synergy Grant IMPACT (grant agreement no. 610150).

References

1. P. A. Abdulla, M. F. Atig, Y. Chen, L. Holík, A. Rezine, P. Rümmer, and J. Stenman. String constraints for verification. In *CAV*, pages 150–166, 2014.
2. T. Babiak, V. Rehák, and J. Strejcek. Almost linear Büchi automata. *Mathematical Structures in Computer Science*, 22(2):203–235, 2012.
3. S. Bardin, A. Finkel, J. Leroux, and L. Petrucci. FAST: acceleration from theory to practice. *STTT*, 10(5):401–424, 2008.
4. S. Bardin, A. Finkel, J. Leroux, and P. Schnoebelen. Flat acceleration in symbolic model checking. In *ATVA*, pages 474–488, 2005.
5. M. Berzish, V. Ganesh, and Y. Zheng. Z3str3: A string solver with theory-aware heuristics. In *FMCAD*, pages 55–59, 2017.
6. N. Bjørner, N. Tillmann, and A. Voronkov. Path feasibility analysis for string-manipulating programs. In *TACAS*, pages 307–321, 2009.
7. M. Bozga, R. Iosif, and Y. Lakhnech. Flat parametric counter automata. *Fundam. Inform.*, 91(2):275–303, 2009.
8. M. Chrobak. Finite automata and unary languages. *Theor. Comput. Sci.*, 47(3):149–158, 1986.
9. J. D. Day et al. The satisfiability of extended word equations: The boundary between decidability and undecidability. *CoRR*, abs/1802.00523, 2018.

10. J. D. Day, F. Manea, and D. Nowotka. The hardness of solving simple word equations. In *MFCS*, pages 18:1–18:14, 2017.
11. V. Diekert. Makanin's Algorithm. In M. Lothaire, editor, *Algebraic Combinatorics on Words*, volume 90 of *Encyclopedia of Mathematics and its Applications*, chapter 12, pages 387–442. Cambridge University Press, 2002.
12. V. Diekert and J. M. Robson. Quadratic word equations. In *Jewels are Forever, Contributions on Theoretical Computer Science in Honor of Arto Salomaa*, pages 314–326, 1999.
13. V. Ganesh et al. Word equations with length constraints: what's decidable? In *Hardware and Software: Verification and Testing*, pages 209–226. Springer, 2013.
14. L. Holík, P. Janku, A. W. Lin, P. Rümmer, and T. Vojnar. String constraints with concatenation and transducers solved efficiently. *PACMPL*, 2(POPL):4:1–4:32, 2018.
15. A. Jéz. Recompression: a simple and powerful technique for word equations. In *STACS 2013*, LIPIcs, Vol. 20, pages 233–244, 2013.
16. A. Kiezun et al. HAMPI: A solver for word equations over strings, regular expressions, and context-free grammars. *ACM Trans. Softw. Eng. Methodol.*, 21(4):25, 2012.
17. M. Kretínský, V. Rehák, and J. Strejcek. Reachability is decidable for weakly extended process rewrite systems. *Inf. Comput.*, 207(6):671–680, 2009.
18. A. Lechner, J. Ouaknine, and J. Worrell. On the complexity of linear arithmetic with divisibility. In *LICS 15: Logic in Computer Science*. IEEE, 2015.
19. A. Lentin. *Equations dans les Monoides Libres*. Gauthier-Villars, Paris, 1972.
20. J. Leroux and G. Sutre. Flat counter automata almost everywhere! In *Software Verification: Infinite-State Model Checking and Static Program Analysis, 19.02. - 24.02.2006*, 2006.
21. T. Liang, A. Reynolds, C. Tinelli, C. Barrett, and M. Deters. A DPLL(T) theory solver for a theory of strings and regular expressions. In *CAV*, pages 646–662, 2014.
22. A. W. Lin and P. Barceló. String solving with word equations and transducers: towards a logic for analysing mutation XSS. In *POPL*, pages 123–136, 2016.
23. L. Lipshitz. The Diophantine problem for addition and divisibility. *Transactions of the American Mathematical Society*, 235:271–283, 1976.
24. G. S. Makanin. The problem of solvability of equations in a free semigroup. *Sbornik: Mathematics*, 32(2):129–198, 1977.
25. A. Martinez. Efficient computation of regular expressions from unary NFAs. In *In DFCS*, pages 174–187, 2002.
26. Y. Matiyasevich. A connection between systems of words-and-lengths equations and Hilberts tenth problem. *Zapiski Nauchnykh Seminarov POMI*, 8:132–144, 1968.
27. R. Parikh. On context-free languages. *J. ACM*, 13(4):570–581, 1966.
28. P. Saxena, D. Akhawe, S. Hanna, F. Mao, S. McCamant, and D. Song. A symbolic execution framework for JavaScript. In *S&P*, pages 513–528, 2010.
29. A. W. To. Unary finite automata vs. arithmetic progressions. *Inf. Process. Lett.*, 109(17):1010–1014, 2009.
30. A. W. To and L. Libkin. Algorithmic metatheorems for decidable LTL model checking over infinite systems. In *FOSSACS*, 2010.
31. M. Trinh, D. Chu, and J. Jaffar. S3: A symbolic string solver for vulnerability detection in web applications. In *CCS*, pages 1232–1243, 2014.
32. H. Wang, T. Tsai, C. Lin, F. Yu, and J. R. Jiang. String analysis via automata manipulation with logic circuit representation. In *CAV*, pages 241–260, 2016.
33. F. Yu, M. Alkhalaf, T. Bultan, and O. H. Ibarra. Automata-based symbolic string analysis for vulnerability detection. *Formal Methods in System Design*, 44(1):44–70, 2014.