Solving One Variable Word Equations in the Free Group in Cubic Time

Robert Ferens
Institute of Computer Science, University of Wrocław, Poland

Artur Jeż
Institute of Computer Science, University of Wrocław, Poland

Abstract

A word equation with one variable in a free group is given as $U = V$, where both U and V are words over the alphabet of generators of the free group and X, X^{-1}, for a fixed variable X. An element of the free group is a solution when substituting it for X yields a true equality (interpreted in the free group) of left- and right-hand sides. It is known that the set of all solutions of a given word equation with one variable is a finite union of sets of the form $\{\alpha w^i \beta : i \in \mathbb{Z}\}$, where α, w, β are reduced words over the alphabet of generators, and a polynomial-time algorithm (of a high degree) computing this set is known. We provide a cubic time algorithm for this problem, which also shows that the set of solutions consists of at most a quadratic number of the above-mentioned sets. The algorithm uses only simple tools of word combinatorics and group theory and is simple to state. Its analysis is involved and focuses on the combinatorics of occurrences of powers of a word within a larger word.

2012 ACM Subject Classification Mathematics of computing → Combinatorics on words; Theory of computation → Formalisms; Computing methodologies → Equation and inequality solving algorithms

Keywords and phrases Word equations, free group, one-variable equations

Digital Object Identifier 10.4230/LIPIcs.STACS.2021.30

Related Version *Full Version*: http://arxiv.org/abs/2101.06201

Funding This work was supported under National Science Centre (NCN), Poland project number 2017/26/E/ST6/00191.

1 Introduction

Word equations in the free group. A word equation is a formal equation $U = V$ in which both U, V contain letters from a fixed set (called alphabet) Σ and variables; a solution is a substitution of variables by words over Σ such that this formal equation is turned into an equality. We consider such equations in a free group, so the aforementioned equality is interpreted as the equality in the free group generated by Σ; naturally, we allow the usage of inverses of variables and generators in the equations. The satisfiability problem (of word equation over the free group) is to decide, whether the input equation has a solution. By solving the equation we mean to return an (explicit or effective) representation of all solutions.

The first algorithm for the satisfiability problem was given by Makanin [27] and it is an involved generalization of Makanin's algorithm for the satisfiability of word equation in the free monoid [26]; Razborov generalized the algorithm so that it solves word equations in the free group [32]; the description is infinite and is known as Makanin-Razborov diagrams. Makanin's algorithm is very involved and known to be not primitively recursive [21], the same applies to Razborov's generalisation, which was the first step of solving Tarski's conjectures (on elementary equivalence and decidability of the theory of free groups) [20, 33]. A different approach based on Plandowski's algorithm for the free monoid case [31] was later

proposed [8], and an even simpler approach, which gives also a finite description of the solution set, was given by Diekert, Plandowski and Jeż [9], it extends Jeż's algorithm for the free monoid case [16].

The problem of word equations in the free group was first investigated by Lyndon [25], who considered the restricted variant of one-variable equations. He showed that the solution set is a finite union of sets of the form

$$\{w_0 w_1^{i_1} w_2 w_3^{i_2} \cdots w_{2k-1}^{i_k} w_{2k} \ : \ i_1, \ldots, i_k \in \mathbb{Z}\} \ , \tag{1}$$

where $w_0, \ldots, w_{2k}$ are words over the generators of the free group, we call such sets k-parametric. In fact, it was first shown using combinatorial arguments that a superset of all solutions is of this form, and using algebraic methods the superset of all solutions is transformed into the actual set of all solutions. As a result, k depends on the equation and is a by-product of the algorithm rather than an explicitly given number. By using a more refined, though purely combinatorial, argument Appel [1] showed that there exists a superset of solutions that is a finite union of 1-parametric sets and that one can test for which values such words are indeed solutions. In principle, the proof can be readily used as an algorithm, but no reasonable bounds can be derived from it. Unfortunately, Appel's proof contains an error (see [5] for a discussion). A similar characterization was announced by Lorentz [23], but the proof was not supplied. Chiswell and Remeslennikov [5] used a different approach, based on geometric group theory, to show that the solution is a finite union of 1-parametric sets. However, their argument does not give any algorithm for solving an equation. Gilman and Myasnikov [12] gave a proof that the solution set is 4-parametric; their proof is based on formal language theory and is considerably simpler and shorter than the other known ones, however, it yields no algorithm.

A polynomial-time algorithm solving the one-variable word equations (in the free group) was given by Bormotov, Gilman and Myasnikov [3]. In principle, their argument is similar to Appel, though simpler (and without errors), and extra care is taken to guarantee that testing takes polynomial time. The running time is high, though little effort was made to lower the exponent, we believe that simple improvements and better analysis should yield $\mathcal{O}(n^5)$ running time of their algorithm.

It is known that already two-variable word equations (in the free group) do not always have a parametrizable solution set [2], here a parametrizable set is a generalization of parametric sets (1) in which the exponents using integer parameters can be nested and one exponent may depend on different parameters. Moreover, no polynomial-time algorithm for two-variable equations is known. Other restricted cases were also investigated, say the famous Lyndon-Schützenberger Theorem was originally shown for the free group [24] and satisfiability of quadratic word equations is known to be NP-complete [19] in the case of free group.

Our results and proof outline. We present an $\mathcal{O}(n^2 m)$ algorithm for solving equations with one variable in a free group, where n is the length of the equation and m the number of occurrences of the variable in it.

▶ **Theorem 1.** *Given a word equation with one variable in a free group, with length n and m occurrences of the variable, we can compute the set of all its solutions in time $\mathcal{O}(n^2 m)$. The set of solutions is a union of $\mathcal{O}(n^2)$ sets of the form $\{\alpha w^k \beta \ : \ k \in \mathbb{Z}\}$, where α, w, β are words over the generators of the given free group.*

The running time is achieved in the RAM model, more specifically we require that operations on $\log n$-bits long integers (and byte-arrays) can be performed in $\mathcal{O}(1)$ time. If this is not the case, then the running time increases by a multiplicative $\mathcal{O}(\log n)$ factor. Note that in Theorem 1 we allow $w = \varepsilon$, i.e. the set $\{\alpha w^k \beta \ : \ k \in \mathbb{Z}\}$ from Theorem 1 may consist of a single string.

The $\mathcal{O}(n^2 m)$ running time seems hard to improve: all known characterization of solution set include $\Omega(n^2)$ individual words that should be tested as solutions and natural testing of a single solution is done in $\Theta(m)$ time, note that this does not take into account the 1-parametric sets that do depend on the parameter, which seem to be harder to be tested.

We use a previous characterization of the solution superset [3], from which it follows that the main task is to compute, given words α, u, v, β, for which $i, j \in \mathbb{Z}$ the word $\alpha u^i v^j \beta$ is a solution. Roughly speaking, the previous approaches [1, 3] argued that if $\alpha u^i v^j \beta$ is a solution for a "large enough" i then $\alpha u^{i'} v^j \beta$ is a solution for each $i' \in \mathbb{Z}$; thus one has to check some "small" is and one "large enough"; for each fixed i we substitute its value and similarly argue that if j is "large enough" then each j' yields a solution (the actual argument is more subtle and symmetric in terms of i and j). We refine this approach: previously the tested values of i and j did not depend on the actual equation, but only on its length. We identify a small set of candidate pairs (i, j) based on the actual equation. To this end, we substitute $\alpha u^I v^J \beta$ to the equation, where I, J are integer variables, and intend to verify, for which values (i, j) of variables (I, J) it is a solution. Such parametric candidates cannot be tested as solutions (in particular because it could be that only for some values of I and J they indeed are solutions), however, some operations can be performed on u^I (or v^J), regardless of the actual value substituted for I: say $u^I u^I u^{-1} u^{-I}$ is equal to u^{I-1} (in a free group). After performing all such possible operations we obtain a word with "parametric powers" of u, v, i.e. powers, whose exponents depend on parameters I, J, note that the parameters are the same for all powers in the parametric word, but the actual exponents in different powers may be different. If there are only powers of u (or only powers of v) then using known tools one can show that one of those exponents is (almost) 0. This yields a linear set of possible is that should be tested. Ideally, we would like to say that a similar claim holds also when parametric powers of both u and v are present. However, those powers can interact and such an approach does not work directly. Instead, if $I = i, J = j$ yields a solution, then substituting $I = i$ (as a mental experiment) either reduces the whole word to ε, in which case each $J = j$ yields a solution, or leaves only powers of v, in which case we can reiterate the same approach, this time for powers of v. The former case gives a set of candidates for I, the latter for J, technically those depend on the substituted i, but this dependency can be removed by further analysis. A similar analysis can be made for substitution $J = j$, together yielding a superset of all possible solutions, which are then individually tested.

Additional analysis is needed to bound the number of candidates that is obtained in this way. To this end, we analyze the set of possible exponents of powers of u and v. In particular, we show that initially all such exponents are of the form $\pm I + c$ and $\pm J + c$, which allows for much better estimations: for the candidate solution to be different, the constants in those expressions need to be different and to have a factor u^{I+c} some $c|u|$ letters from the equation are "consumed" and easy calculations show that there are only $\mathcal{O}(\sqrt{n})$ different possible constants, which leads to $\mathcal{O}(\sqrt{n})$ different candidates. One has to take special care of α, β, as their introduction can yield a quadratic-size equation. To avoid this, we analyze how powers of u in concatenations of words can be obtained.

In most cases, we reduce the problem in the free group to the problem in the free monoid (with involution) and use standard tools of word combinatorics. However, this requires some additional properties of words α, u, v, β. Those cannot be inferred from known characterizations, and so known proofs are reproved and the additional claims are shown.

Connection to word equation in the free monoid. The connection between word equations in the free group and free monoid is not perfectly clear. On one hand, the satisfiability of the former can be reduced to the satisfiability of word equations over the free monoid (with involution), this was implicitly done by Makanin [27] and explicitly by Diekert et al. [8] and so generalizations of algorithms for the monoid case are used for the group case. However, there is an intuition that the additional group structure should make the equations somehow easier. This manifests for instance for quadratic equations (so the case when each variable is used at most twice), for which an NP algorithm was given for the free-group case [19] and no such result is known for the free monoid case. Furthermore, the whole first-order theory of equations over the free group is decidable [20], while already one alternation of the quantifiers make a similar theory for monoid undecidable (see [6] for an in-depth discussion of undecidable and decidable fragments).

On the other hand, such general reductions increase the number of variables and so are not suitable in the bounded number of variables case. In particular, a polynomial time algorithm for the satisfiability of two-variable equations for the free monoid is known [10], in contrast to the case of the free group (the set of solutions is still not parametrisable [13], as in the case of the free group.).

Word equations in free monoid with restricted number of variables. Word equations in the free monoid with restricted number variables were also considered. For one variable a cubic-time algorithm is trivial and can be easily improved to quadratic-running time [7]. Eyono Obono, Goralcik and Maksimenko gave a first non-trivial algorithm running in time $\mathcal{O}(n \log n)$ [30]. This was improved by Dąbrowski and Plandowski [11] to $\mathcal{O}(n + m \log n)$, where m is the number of occurrences of the variable in the equation, and to $\mathcal{O}(n)$ by Jeż [15]; the last two algorithms work in the RAM model, i.e. they assume that operations on the $\log n$-bits long numbers can be performed in constant time. The properties of the solution set were also investigated: all above algorithms essentially use the fact that the solution set consists of at most one 1-parametric set and $\mathcal{O}(\log n)$ other solutions [30]. Plandowski and Laine showed that the solution set is either exactly a 1-parametric set or of size $\mathcal{O}(\log m)$ [22] and conjectured that in the latter case there are at most 3 solutions. This conjecture was recently proved by Saarela and Nowotka [29] using novel techniques.

Word equations in the free monoid with two variables were also investigated. it was shown by Hmelevskiĭ [13] that there are equations whose solution set is not parametrizable. The first polynomial-time algorithm (of a rather high degree) for satisfiability of such equations was given by Charatonik and Pacholski [4], this was improved to $\mathcal{O}(n^6)$ by Ille and Plandowski [14] and later to $\mathcal{O}(n^5)$ by Dąbrowski and Plandowski [10], the latter algorithm also returns a description of all solutions. The computational complexity of word equations with three variables is unknown, similarly, the computational complexity of satisfiability in the general case of word equations in the free monoid remains unknown (it is NP-hard and in PSPACE).

2 Definitions and preliminaries

2.1 Notions

Monoids, monoids with involution. By Σ we denote an alphabet, which is endowed with involution $\bar{\cdot} : \Sigma \to \Sigma$, i.e. a function such that $\overline{a} \neq a = \overline{\overline{a}}$. The free monoid Σ^* with involution consists of all finite words over Σ and the involution uniquely extended from Σ to Σ^* by requiring that $\overline{(uv)} = \overline{v}\,\overline{u}$, i.e. we think of it as of inverse in a group. We denote the empty word by ε. Given a word uvw: u is its prefix, w suffix and v its subword; for a

word w often w' and w'' will denote the prefix and suffix of w, this will be always written explicitly. A word $w = a_1 \cdots a_k$, where $a_1, \ldots, a_k \in \Sigma$, has length $|w| = k$ and $w[i\,..\,j]$ denotes a subword $a_i \cdots a_j$. For $k \geq 0$ a word u^k is a k-th *power* of u (or simply u-power), by convention u^{-k} denotes $\overline{u}^k$. A u-power prefix (suffix) of v is the longest u-power that is prefix (suffix, respectively) of v, note that this may b a positive or negative power, or ε. A single-step reduction replaces $wa\overline{a}v$ with wv, a reduction is a sequence of single-step reductions. A word in a free monoid Σ^* with involution is *reduced* if no reduction can be performed on it. It is folklore knowledge that for w there exists exactly one reduced v such that w reduces to v; we call such a v the *normal form* of w and denote it by $\mathrm{nf}(w)$; we write $w \approx v$ when $\mathrm{nf}(w) = \mathrm{nf}(v)$. We write $u \sim v$ to denote that $u = u'v'$ and $v = v'u'$ or $\overline{v} = v'u'$ for some u', v'. A reduced word w is *cyclically reduced* if it is not of the form $w = av\overline{a}$ for any $a \in \Sigma$ and w is *primitive* if there is no word v, such that $w = v^k$ for some natural number $k > 1$.

Free group. Formally, the free group (over generators Σ) consists of all reduced words over Σ with the operation $w \cdot v = \mathrm{nf}(wv)$. We use all elements of Σ^* to denote elements of the free group, with w simply denoting $\mathrm{nf}(w)$. Note that in such a setting $\approx$ corresponds to equality in free group. Note that the inverse w^{-1} of w is $\overline{w}$ and we will use this notation, as most of the arguments are given for the monoid and not the free group.

Any equation in the free group is equivalent to an equation in which the right-hand side is ε, as $u \approx v$ is equivalent to $uv^{-1} \approx \varepsilon$, thus in the following we consider only equations in such a form. Moreover, $uv \approx \varepsilon$ is equivalent to $vu \approx \varepsilon$, which can be seen by multiplying by v from the left and v^{-1} from the right; hence we can assume that the equation begins with a variable. Let us fix the equation

$$X^{p_1} u_1 X^{p_2} u_2 \cdots u_{m-1} X^{p_m} u_m \approx \varepsilon \tag{2}$$

for the rest of the paper, each u_i is a reduced word in Σ^*, every p_i is 1 or -1 and there are no expressions $X\varepsilon\overline{X}$ nor $\overline{X}\varepsilon X$ in the equation. Clearly, m is the number of occurrences of the variable X in the equation, let $n = m + \sum_{i=1}^{m} |u_i|$ be the length of the equation. A reduced word $x \in \Sigma^*$ is a solution when $x^{p_1} u_1 x^{p_2} \cdots u_{m-1} x^{p_m} u_m \approx \varepsilon$.

Integer expressions, parametric words. Let us fix two integer variables I, J for the remainder of the paper. An integer expression is of the form $n_I I + n_J J + n_c$, where $n_I, n_J, n_c \in \mathbb{Z}$ are integers; an expression is *constant* when $n_I = n_J = 0$ and *non-constant* otherwise. We denote integer expressions with letters ϕ, ψ, note that all expressions that we consider are in the same two variables I, J. A value $\phi(i, j)$ is defined in a natural way; we also use this notation for substitutions of variables, say $\phi(I, k - I)$, which is defined in a natural way. The integer expression ϕ depends on the variable I (J) if $n_I \neq 0$ ($n_J \neq 0$) and it depends on $I + J$ if $n_I = n_J \neq 0$. If ϕ depends on exactly one variable then we write $\phi(i)$ to denote its value.

An s-parametric power is of the form s^{ϕ}, where ϕ is an integer expression and s a word; then $s(i, j)$ denotes $s^{\phi(i,j)}$, this can be interpreted both as an element in the monoid and in the free group. Unless explicitly stated, we consider only non-constant expressions ϕ as exponents in parametric powers, this should remove the ambiguity that an s-power is also an s-parametric power. A parametric word is of the form $w = t_0 s_1^{\phi_1} t_1 \cdots t_{k-1} s_k^{\phi_k} t_k$ (all arithmetic expressions $\phi_1, \ldots, \phi_k$ are in the same two variables I, J) and $w(i, j)$ denotes $t_0 s_1^{\phi_1(i,j)} t_1 \cdots t_{k-1} s_k^{\phi_k(i,j)} t_k$. In most cases, we consider very simple parametric words, where

$k \leq 2$ and both expressions depend on one variable only. We sometimes talk about equality of parametric words (in a free group), formally $w \approx w'$ if for each $(i,j) \in \mathbb{Z}^2$ it holds that $w(i,j) \approx w'(i,j)$. We will use those only in very simple cases, say $u^{I+1}u^{-I+1} \approx u^2$.

As we process sets of integer expressions (as well as parametric powers), we will often represent them as sorted lists (with duplicates removed): we can use any linear order, say for integer expressions the lexicographic order on triples (n_I, n_J, n_c) and for parametric powers the lexicographic order on tuples (s, n_I, n_J, n_c), where tuple (s, n_I, n_J, n_c) corresponds to a parametric power $u^{n_I I + n_J J + n_c}$.

2.2 Data structure

Words appearing naturally in our proofs and algorithms are concatenations of a constant number of subwords (or their involutions) of the input equation. We say that a word w is *k-represented*, if w is given as $w = (U\overline{U})[b_1 \,..\, e_1] \cdots (U\overline{U})[b_k \,..\, e_k]$, where $U = u_1 \cdots u_m$ is the concatenation of all words from the equation (2). A parametric word $s_0 t_1^{\phi_1} s_1 \cdots s_{\ell-1} t_\ell^{\phi_\ell} s_\ell$ is k-represented, when $s_0, t_1, s_1, \ldots, t_\ell, s_\ell$ are $k_0, \ldots, k_{2\ell}$ represented and $k = \sum_{i=0}^{2\ell} k_i$.

We use standard data structures, like suffix arrays [17] and structures for answering longest common prefix queries on them [18]. As a result, we can answer all basic queries (like normal form, longest common prefix, power prefix, etc.) about words in the equation in $\mathcal{O}(1)$ time; note that this is the place in which we essentially use that we can perform operations on $\mathcal{O}(\log n)$-size numbers in $\mathcal{O}(1)$ time. As an example of usage, we can test whether a word is a solution in $\mathcal{O}(m)$ time:

▶ **Lemma 2.** *Given a word $\alpha u^i v^j \beta$, where α, β, u, v are $\mathcal{O}(1)$-represented, α, β are reduced and u, v are cyclically reduced and primitive and i, j are a pair of integer numbers, we can test whether $\alpha u^i v^j \beta$ is a solution of (2) in $\mathcal{O}(m)$ time.*

2.3 Superset of solutions

The previous characterization [3] essentially showed that a solution is either a $\mathcal{O}(1)$-represented word or of the form $u^i u' v'' v^j$ for some $i, j \in \mathbb{Z}$ and u' is a prefix of u and v'' is a suffix of v for some well defined u, v. As we intend to analyze those solutions using word combinatorics, it is useful to assume that u, v are cyclically reduced and primitive. Unfortunately, this cannot be extracted directly from the previous characterization, so we repeat the previous arguments taking some extra care.

▶ **Lemma 3** (cf. [12, Lemma 15])**.** *For a given equation (2), in $\mathcal{O}(n^2)$ time one can compute a superset of solutions of the form*

$$S \cup \bigcup_{(\alpha u^I v^J \beta) \in W} \bigcup_{i,j \in \mathbb{Z}} \{\alpha u^{I(i)} v^{J(j)} \beta\}$$

where S is a set of $\mathcal{O}(1)$-represented words with $|S| = \mathcal{O}(n^2)$ and for each $0 \leq i \leq m-1$ there are numbers $\ell_i, \ell'_i \leq |u_i| + |u_{i+1}|$ such that W contains exactly $\ell_i \cdot \ell'_i$ parametric words satisfying

- *α, β, are $\mathcal{O}(1)$-represented, reduced and $|\alpha|, |\beta| \leq |u_i| + |u_{i+1}|$;*
- *u, v are 2-represented, cyclically reduced, primitive and $|u| = \ell_i$ and $|v| = \ell'_i$.*

2.4 Maximal powers

We say that a word s^p is a *maximal power* in a word t, if it is a subword of t and there is no s nor $\overline{s}$ to its left and right in t; note that t need not to be reduced. For instance a^3, a^2 and $(ab)^2$ are maximal powers in *aaababaa*. To streamline the analysis, we assume that s^0 (called the *trivial power*) is a maximal power in any word t, even the empty one.

If s^p is a maximal power in a normal form of concatenation of several words $\mathrm{nf}(w_1 \cdots w_\ell)$, then clearly s^p can be partitioned into ℓ subwords such that the i-th of them comes from w_i. However, we show more: we can identify such a maximal power in each w_i, that s^p is (almost) the normal form of concatenation of those maximal powers. This is beneficial: the number of different maximal powers in a word is much smaller than the number of different powers that are subwords.

▶ **Lemma 4.** *Let $w_1, w_2, \ldots, w_\ell$ be reduced and s be cyclically reduced. If s^k is a maximal power in $\mathrm{nf}(w_1 \cdots w_\ell)$ then for each $1 \leq h \leq \ell$ there exists such a maximal power s^{k_h} in w_h that $|\sum_{h=1}^{\ell} k_h - k| < \ell$. Moreover, if s^k is the s-power prefix (suffix) of $\mathrm{nf}(w_1 \cdots w_\ell)$ then we can choose s^{k_1} as the s-power prefix of w_1 or a trivial power (s^{k_ℓ} as the s-power suffix of w_ℓ or a trivial power, respectively); if $s^k = \mathrm{nf}(w_1 \cdots w_\ell)$ then both conditions hold simultaneously.*

The proof of Lemma 4 in case of $\ell \leq 2$ is a simple case distinction. For larger ℓ, we let $w_{1,2} = \mathrm{nf}(w_1 w_2)$ and apply the induction assumption to $w_{1,2} w_3, \ldots, w_\ell$, the proof again follows by simple combinatorics on words.

There cannot be too many different maximal powers of the same word s in a given word w: different maximal powers $s^{k_1}, \ldots, s^{k_p}$ use together $|s|k_1 + \cdots + |s|k_p$ letters in w and when $k_1, \ldots, k_p$ are pairwise different then this sum is $\Omega(p^2|s|)$ and so $p = \mathcal{O}(\sqrt{|w|/|s|})$; this can be naturally generalized to a set of words W instead of a single word w.

▶ **Lemma 5.** *Let s be cyclically reduced word. Let W be a set of words and $k = \sum_{w \in W} |w|$. Suppose that $s^{k_1}, \ldots, s^{k_p}$ are pairwise disjoint subwords of words in W and that $k_1, \ldots, k_p$ are pairwise different integers. Then $p \leq \sqrt{4k/|s| + 1}$ and if additionally $k \geq |s|$ then $p \leq \sqrt{5k/|s|}$.*

3 Restricting the superset of solutions

By Lemma 3, we know the form of possible solutions, and by Lemma 2 we can test a single candidate solution in $\mathcal{O}(m)$ time. In particular, all solutions from the set S in Lemma 3 can be tested in $\mathcal{O}(n^2 m)$ time, as desired. The other solutions are instances of parametric words the form $\alpha u^I v^J \beta$ for well-defined α, u, v, β. The next step is to bound, for fixed α, u, v, β, the set of values (i, j) such that $\alpha u^I v^J \beta(i, j)$ could be a solution; this is the main result of the paper.

Idea. Suppose we want to find out which words of the form u^i are a solution of (2). We substitute u^I to the equation and treat its left-hand side as a parametric word w depending on I. If substituting $I = i$ leads to a trivial word, then it is known that some u-power cancels within the neighboring u-powers (actually, a variant of this fact was used to characterize the superset of solutions [25, 1, 3], and it is attributed already to Nielsen [28]), more formally:

▶ **Lemma 6** (cf. [3, Lemma 3]). *Let $\varepsilon \approx s_0 u_1 s_1 u_2 \cdots s_{k-1} u_k s_k$. Then there is u_i which reduces within $u_{i-1} s_{i-1} u_i s_i u_{i+1}$.*

We want to use Lemma 6 to claim that some u-parametric powers need to reduce, however, as there can be powers of u as constants, this makes the analysis problematic: as an example, consider an equation $a u^I u^\ell \overline{a} \approx \varepsilon$, if $I = i$ is a solution and we set $s_0 = a, u_1 = u^i, s_1 = u^\ell \overline{a}$ (so that u_1 corresponds to u^I) then Lemma 6 guarantees that u^i cancels within u^ℓ, i.e. $0 \geq i \geq -\ell$, even though $I = -\ell$ is the only solution. This is caused by u-powers next to

u-parametric power, which makes our application of the Lemma 6 nearly useless. To fix this, in $au^iu^\ell\overline{a}$ we set $s_0 = a, u_1 = u^{i+\ell}, s_1 = \overline{a}$, and then Lemma 6 yields $i = -\ell$. On the level of the parametric word this corresponds to considering $au^{I+\ell}\overline{a} \approx au^Iu^\ell\overline{a}$, i.e. we include u-powers into the u-parametric power next to them.

This is formalized as follows: A parametric word w is *u-reduced* when u is cyclically reduced, primitive and w does not have a subword of the form:

- u^ϕ for a constant integer expression ϕ;
- $a\overline{a}$ for some letter a (so w is reduced);
- $u^\phi u^\psi$ for some (non-constant) integer expressions ϕ, ψ;
- $uu^\phi, \overline{u}u^\phi, u^\phi u, u^\phi\overline{u}$ for some (non-constant) integer expression ϕ.

Note that we do not forbid subwords that are powers of u, we forbid parametric subwords that are in fact subwords, i.e. have constant exponents.

Given a parametric word w we can u-reduce it to obtain a parametric word that is equal (in the free group) and u-reduced by a simple greedy procedure, i.e. replacing a parametric power with a constant integer expression as exponent with a power or reduction or joining two u-powers into one (the running time for specific applications is analyzed separately at appropriate places). When we replace, say uu^ϕ with $u^{\phi+1}$, then we say that letters in u were u-reduced to $u^{\phi+1}$. Note that there are different u-reduced equivalent parametric words, so the output of u-reduction is not unique, this has no effect on the algorithm, though.

If a parametric word w (with all exponents depending on one variable) is u-reduced then from Lemma 6 we infer that $w(i) \approx \varepsilon$ implies $|\phi(i)| \leq 3$ for some parametric power u^ϕ in w:

▶ **Lemma 7.** *Let $w = w_0u^{\phi_1}w_1 \cdots u^{\phi_k}w_k$ be a u-reduced parametric word, where $w_0, \ldots, w_k$ are words and $\phi_1, \ldots, \phi_k$ are integer expressions, all depending on exactly one and same variable. If $w(i) \approx \varepsilon$ then there is ϕ_ℓ such that $|\phi_\ell(i)| \leq 3$. In particular, $w(i) \approx \varepsilon$ for each i if and only if $w = \varepsilon$.*

As ϕ_ℓ in Lemma 7 is a non-constant integer expression then there are at most 7 values of i such that $|\phi_\ell(i)| \leq 3$. Hence it is enough to find appropriate i values. Clearly, there are at most m integer expressions in w (as this is the number of variables). We can give better estimations, though: if the expression is not of the form kI then it "used" at least $|u|$ letters from the equation. So there are $n/|u|$ different expressions and the ones of the form kI; as $|ki| \leq 3$ implies $|i| \leq 3$, there are $7(1 + n/|u|)$ candidates for i in total. Lastly, when the solution depends on two variables, it can be shown that all obtained parametric powers have coefficient ± 1, which allow even better estimations: a parametric power $I + c$ uses at least $c|u|$ letters from the equation and so it can be shown that at most $\mathcal{O}(\sqrt{n/|u|})$ different integer expressions can be formed in such a case.

The actual solution is of the form $\alpha u^I v^J \beta$. Firstly, the presence of α, β make estimations harder, as their letters can also be used in the u- and v-reductions. Secondly, there are two parameters, which makes a simple usage of Lemma 7 impossible. However, if $w(i, j) \approx \varepsilon$ then $w(I, j) \approx \varepsilon$ depends on one variable, so Lemma 7 is applicable to it. The analysis yields that we can restrict the possible value of i or j or (i, j); note that this is non-obvious, as there are infinitely many $w(I, j)$s. A similar analysis can be made for $w(i, J)$, and combining those two yields a set of pairs to be tested as well as $\mathcal{O}(1)$ individual is and js that should be tested separately. But for a fixed i (j) we can substitute it to the equation and use Lemma 7 for J (I, respectively).

3.1 Restricting the set of (i, j)

Fix some $0 \leq i_0 \leq m-1$ and the corresponding u_{i_0}, u_{i_0+1} in the equation (2). Using Lemma 3 we construct a parametric word $\alpha u^I v^J \beta$, with α, u, v, β depending on u_{i_0}, u_{i_0+1} as well as exponents $p_{i_0}, p_{i_0+1}, p_{i_0+2}$. We substitute $X = \alpha u^I v^J \beta$ to the equation (2), obtaining a parametric word on the left-hand side. We are to find values $(i, j) \in \mathbb{Z}^2$ for which the value of the obtained parametric word is equivalent to ε, thus we call such an (i, j) a solution. We want to find a suitable set of pairs (i, j) and test each one individually, using Lemma 2.

The analysis depends on the relation between u and v: i.e. whether $u \in \{v, \overline{v}\}$, $u \not\sim v$ or $u \sim v$. We analyze particulate cases in Sections 3.1.1–3.1.3. The idea is the same in each case, but technical details differ.

3.1.1 $u \not\sim v$

Due to symmetry, we consider the case when $|v| \geq |u|$, note that it could be that $|u| = |v|$. We rotate the left-hand side of the equation so that it begins and ends with a parametric power: we rotate $\alpha u^I v^J \beta w = \varepsilon$ to $v^J \beta w \alpha u^I = \varepsilon$ or $\overline{\beta} \overline{v}^J \overline{u}^I \overline{\alpha} w = \varepsilon$ to $\overline{u}^I \overline{\alpha} w \overline{\beta} \overline{v}^J = \varepsilon$, depending on the form of the equation. The equation after the rotation is equisatisfiable to the previous one.

We call each parametric word beginning with v^J or $\overline{u}^I$ and ending with u^I or $\overline{v}^J$ and no parametric power inside a *fragment*. The parametric word after the rotation is a concatenation of m fragments. We use the name h-th fragment to refer to the one corresponding to u_h (so h-th from the left); let f_h denote the word that is left from h-th fragment after removing the leading and ending parametric power; note that f_h is of one of the forms $\beta u_h \alpha$, $\beta u_h \overline{\beta}$, $\overline{\alpha} u_h \alpha$, $\overline{\alpha} u_h \overline{\beta}$. For u^I we call the preceding α the associated word, the same name is used to β succeeding v^J, $\overline{\alpha}$ succeeding $\overline{u}^I$ and $\overline{\beta}$ preceding $\overline{v}^J$. To simplify, we will call it a word associated with the parametric power.

We now preprocess the equation, by replacing the left-hand side with an equivalent parametric word (i.e. equal according to $\approx$). As a first step, we replace each f_h with $\mathrm{nf}(f_h)$. Next, observe that if w is the power of u then $\overline{u}^I w u^I \approx w$ and similarly $v^J w' \overline{v}^J \approx w'$ for w' being a power of v. In the second step we check each fragment separately, and if possible, replace it as described above. For fragments that remained unchanged in the second step, we use previous names, i.e. if h-th fragment $v^J \mathrm{nf}(f_h) u^I$ was not replaced then we still write it as $v^J \mathrm{nf}(f_h) u^I$ and call it h-th fragment. A *trivial fragment* is a maximal subword obtained as concatenations of words obtained due to replacements in the second step.

We now perform the u-reduction (note that the v^J is not touched) and afterwards the v-reduction. Let the obtained equation be of the form

$$W \approx \varepsilon \ , \tag{3}$$

where W is a parametric word.

▶ **Lemma 8.** *For $u \not\sim v$ we can perform the u-reduction and v-reduction after the preprocessing in $\mathcal{O}(m)$ time; the obtained parametric word is u-reduced. No two parametric powers are replaced by one during the u-reduction and v-reduction, in particular, for a given parametric power u^ϕ (v^ψ) in (3) the ϕ (ψ) has a coefficient of the variable equal to ± 1 and the only letters that are u-reduced (v-reduced) to this power come either from the associated fragment of u^I or $\overline{u}^I$ (v^J or $\overline{v}^J$) and the letters from the adjacent trivial fragment (assuming that there is an adjacent trivial fragment).*

Note that the claim that no two parametric powers are replaced by one is not obvious – in principle, it could be that after the preprocessing a trivial fragment is a power of u (or v) and then it is wholly u-reduced, which can lead to two adjacent parametric powers of u, which are then replaced with one. However, this cannot happen, as such a trivial fragment is of the form $u^{k_1}v^{k_2}\cdots$ for some $0 < |k_1|, |k_2|, \ldots$ and such a word cannot be a power of u nor v when $u \not\sim v$, as the subgroup generated by u, v is a free group.

We now estimate, how many different u-parametric expressions are there after the reductions. When we want to distinguish between occurrences of parametric powers with the same exponent (say, two occurrences of u^{I+1} counted separately) then we write about parametric powers and when we want to treat it as one, then we talk about exponents. We provide two estimations, one focuses on parametric powers and the other on exponents.

▶ **Lemma 9.** *There is a set S of $\mathcal{O}(1)$ size of integer expressions such that there are $\mathcal{O}(n/|u|)$ occurrences of u-parametric powers in W from* (3) *whose exponents are not in S and $\mathcal{O}(n/|v|)$ occurrences of v-parametric powers whose exponents are not in S. The set S can be computed and the parametric powers identified in $\mathcal{O}(m + n/|u|)$ time.*

The Lemma considers, whether the parametric power used some letters from the trivial fragment or its associated fragment had u_h of length at least $|u|$. If so, then it is in the $\mathcal{O}(n/|u|)$ parametric powers, as one such power uses at least $|u|$ letters of the input equation (this requires some argument for the trivial fragments) and otherwise is can be shown that there are only $\mathcal{O}(1)$ possible exponents: say, when we consider the longest suffix of $\mathrm{nf}(\beta u_h \alpha)$ that is a u-power, where $|u_h| < |u|$, then there is a constant number of possibilities how this suffix is formed (fully within α, within $\mathrm{nf}(u_h\alpha)$, uses some letters of β) and in each case the fact that $|u_h| < |u|$ means that there are only $\mathcal{O}(1)$ different u_hs that can be used; note that we need the primitivity of u here. Concerning the algorithm, note that we can distinguish between these two cases during the preprocessing and mark the appropriate powers.

The next lemma provides a better estimation for the number of different exponents, it essentially uses the fact that all exponents have coefficients at variables ± 1: as there are only two possible coefficients, we can focus on the constants. Now, to have a constant $|c|$, we have to use a power u^c from W and to have k *different* constants one has to use k different powers and so from Lemma 5 we conclude that $k = \mathcal{O}(|W|/|u|)$. In general, W can be of quadratic length, as we introduce m copies of α and β into it; the resulting bound is too weak for our purposes. To improve the bound, we employ Lemma 4: consider that when the u-power suffix of, say, $\beta u_h \alpha$, is u^k then by Lemma 4 there are k_α, k_u, k_β such that $|k - k_\alpha - k_u - k_\beta| \leq 2$ and u^{k_u}, u^{k_β} are maximal u-powers in u_h, β and u^{k_α} is the u-power suffix of α. Using Lemma 5, this yields that there are $\mathcal{O}(\sqrt{n/|u|})$ different possible values of k_u (over all u_h), $\mathcal{O}(\sqrt{|\beta|/|u|}) = \mathcal{O}(\sqrt{|u_{i_0}u_{i_0+1}|/|u|})$ of k_β and k_α is fixed, so there are at most $\mathcal{O}(\sqrt{n/|u|} \cdot \sqrt{|u_{i_0}u_{i_0+1}|/|u|}) = \mathcal{O}(\sqrt{n|u_{i_0}u_{i_0+1}|}/|u|)$ possible values of k.

The actual argument is more involved, as it is also possible that the u-parametric power includes letters from the trivial fragments, which requires some extra arguments, nevertheless the general approach is similar.

▶ **Lemma 10.** *After the u-reduction and v-reduction there are $\mathcal{O}(\sqrt{n|u_{i_0}u_{i_0+1}|}/|u|)$ different integer expressions as exponents in parametric powers of u and $\mathcal{O}(\sqrt{n|u_{i_0}u_{i_0+1}|}/|v|)$ of v in the equation. The (sorted) lists of such expressions can be computed in $\mathcal{O}(m + n/|u|)$ and $\mathcal{O}(m + n/|v|)$ time, respectively.*

Concerning the algorithm and its running time, it is enough to list all exponents, remove duplicates, and sort them.

We can use Lemma 6 together with bounds on the number of different exponents in parametric powers from Lemma 10 to limit the possible candidates (i, j) for a solution. However, these bounds are either on i or on j. And as soon as we fix, say, $J = j$ and substitute it to W, the obtained parametric word $W(I, j)$ (or $W(i, J)$) is more complex than W, in particular, we do not have the bounds of Lemma 10 for it, so the set of possible candidates for i for a given $W(I, j)$ is linear, which is too much for the desired running time.

Instead, we analyze (as a mental experiment) $W(I, j)$: Fix $j \in \mathbb{Z}$ such that $W(i, j) \approx \varepsilon$ for some i. Compute $W(I, j)$, u-reduce it, call the resulting parametric word $W_{J=j}$. If $W_{J=j} = \varepsilon$, then clearly for each i the (i, j) is a solution of (3) (and vice-versa, see Lemma 7). It can be shown that in this case for some v^ψ in $W_{J=j}$ it holds that $|\psi(j)| < 6$: at least some two u-parametric powers in W should be merged in $W_{J=j}$, in W they are separated by a v-parametric power, say v^ψ. All letters of $v^{\psi(j)}$ are u-reduced, then standard arguments using periodicity show that $|\psi(j)| < 6$ so we can compute all candidates for such js and test for each one whether indeed $W_{J=j} = \varepsilon$, this is formally stated in Lemma 12.

If $W_{J=j}$ depends on I then from Lemma 7 for some of the (new) u-parametric powers u^ϕ it holds that $|\phi(i)| < 6$. Consider, how this ϕ was created. It could be that it is (almost) unaffected by the second u-reduction and so it is (almost) one of the u-parametric powers in W, see Lemma 13 for precise formulation and sketch of proof, in which case we can use Lemma 10. Intuitively, u^ϕ is affected if the whole two parametric powers in W were used to create u^ϕ. Then it can be shown that some v-parametric power v^ψ from W turned into v-power $v^{\psi(j)}$ satisfies $|\psi(j)| < 6$ and is u-reduced to u^ϕ, the argument is as before, when $W_{J=j} \approx \varepsilon$. Moreover, this occurrence of v^ψ also determines u^ϕ; hence the choice of ψ determines $\mathcal{O}(1)$ candidates for j, uniquely identifies ϕ and i satisfies $|\phi(i)| < 6$, i.e. there are $\mathcal{O}(1)$ candidates for (i, j). Then Lemma 9 is applied to this v^ψ: if it is one of $n/|v|$ occurrences of v-parametric powers then we get $\mathcal{O}(1)$ candidates for (i, j) (for this ψ), so $\mathcal{O}(n/|v|)$ in total, over all choices of such ψ. Otherwise, ψ it is one of $\mathcal{O}(1)$ integer expressions (Lemma 9) and so j is from $\mathcal{O}(1)$-size set and we can compute and consider $W_{J=j}$ for each one of them separately.

A similar analysis applies also to $i \in \mathbb{Z}$ substituted for I. The results are formalized in the Lemma 11 below, its proof is spread across a couple of Lemmata.

▶ **Lemma 11.** *Given equation* (3) *we can compute in* $\mathcal{O}(mn/|u|)$ *time sets* $S_I, S_J, S_{\mathbb{Z},J} \subseteq \mathbb{Z}$ *and* $S_{I,J} \subseteq \mathbb{Z}^2$*, where* $|S_I| = \mathcal{O}(\sqrt{n|u_{i_0}u_{i_0+1}|}/|u|)$*,* $|S_J| = \mathcal{O}(1)$*,* $|S_{\mathbb{Z},J}|, |S_{I,J}| = \mathcal{O}(n/|u|)$*, such that: if* (i, j) *is a solution of* (3) *then at least one of the following holds:*

- $i \in S_I$ *or*
- $j \in S_J$ *or*
- $j \in S_{\mathbb{Z},J}$ *and for each* i' *the* (i', j) *is a solution or*
- $(i, j) \in S_{I,J}$*.*

Similarly, given equation (3) *we can compute in* $\mathcal{O}(mn/|v|)$ *time sets* $S'_I, S'_J, S'_{I,\mathbb{Z}} \subseteq \mathbb{Z}$ *and* $S'_{I,J} \subseteq \mathbb{Z}^2$*, where* $|S'_I| = \mathcal{O}(1)$*,* $|S'_J| = \mathcal{O}(\sqrt{n|u_{i_0}u_{i_0+1}|}/|v|)$ $|S'_{I,\mathbb{Z}}|, |S'_{I,J}| = \mathcal{O}(n/|v|)$ *such that at if* (i, j) *is a solution of* (3) *then least one of the following holds:*

- $i \in S'_I$ *or;*
- $i \in S'_{I,\mathbb{Z}}$ *and for each* $j' \in \mathbb{Z}$ *the* (i, j') *is a solution or;*
- $j \in S'_J$ *or;*
- $(i, j) \in S'_{I,J}$*.*

As noted above, the main distinction is whether the u^ϕ in $W_{J=j}$ was "affected" or not during the second u-reduction. Let us formalize this. Given an occurrence of a parametric power u^ϕ in $W_{J=j}$ consider the largest subword w of W such that each letter in $w(I, j)$ is

either reduced or u-reduced to this u^ϕ; note that this may depend on the order of reductions, we fix an arbitrary order. We say that parametric powers in w are *merged* to u^ϕ. We extend this notion also to the case when $W_{J=j} = \varepsilon$, in which case $W = w$ and every parametric power is merged to the same parametric power u^0. A similar notion is defined also for parametric powers of v. Note that a parametric power is not merged to two different parametric powers u^ϕ and $u^{\phi'}$.

We say that a u-parametric power u^ϕ in $W_{J=j}$ *was affected* by substitution $J = j$ if

- more than one parametric power was merged to u^ϕ *or*
- for the unique u-parametric power $u^{\phi'}$ merged to u^ϕ there is a v-parametric power $v^{\psi'}$ such that $|\psi'(j)| < 6$ and there is no u-parametric power between $u^{\phi'}$ and $v^{\psi'}$.

The intuition behind the first condition is that when we merge two u-powers then we create a completely new parametric power, for the second condition, when $|\psi'(j)| < 6$ then $v^{\psi'(j)}$ no longer behaves like $v^{\psi'}$ and can either be wholly merged to a u-power or be canceled by a trivial fragment, which can also lead to a large modification of the neighbouring u-parametric power. Note that the second condition could be made more restrictive, but the current formulation is good enough for our purposes.

We first investigate the case, when the parametric power was affected by a substitution.

▶ **Lemma 12.** *In $\mathcal{O}(mn/|v|)$ time we can compute and sort sets $S_J, S_{E,J}$, where $|S_J| = \mathcal{O}(1)$ and $|S_{E,J}| = \mathcal{O}(n/|v|)$, such that for each occurrence of a u-parametric power u^ϕ in $W_{J=j}$ affected by the substitution $J = j$ either $j \in S_J$ or $(\phi, j) \in S_{E,J}$.*

Similarly, in time $\mathcal{O}(mn/|u|)$ we can compute and sort sets $S'_I, S_{I,E}$, where $|S'_I| = \mathcal{O}(1)$ and $|S_{I,E}| = \mathcal{O}(n/|u|)$, such that for each occurrence of a v-parametric power v^ψ in $W_{I=i}$ affected by the substitution $I = i$ either $i \in S'_I$ or $(i, \psi) \in S_{I,E}$.

The sketch of the argument was given above Lemma 11. Concerning the running time, the appropriate exponents are identified during the u-reduction and v-reduction, which are performed in given times using the data structure.

We now consider the case when u^ϕ was not affected. Essentially, we claim that u^ϕ is almost the same as some $u^{\phi'}$ in W. The difference is that it can u-reduce letters from v-parametric powers that become v-powers. However, as such v-power is not wholly merged (as it is not affected), only its proper suffix or prefix can be u-reduced and by primitivity and by case assumption $u \not\sim v$ and $|v| \geq |u|$, this suffix is of length at most $|v| + |u|$. Thus, while in principle there are infinitely many possibilities for $v^\psi(j)$ when $j \in \mathbb{Z}$, it is enough to consider a constant number of different candidates (roughly: $\overline{v}^2, \overline{v}, \varepsilon, v, v^2$) and we can procure all of them so that an analysis similar to the one in Lemma 10 can be carried out: essentially we replace a fragment $v^J f_h u^I$ with 5 "fragments" $v^c f_h u^I$ for $c \in \{-2, -1, 0, 1, 2\}$. In this argument, we used the assumption that $|v| \geq |u|$ (the u-reduction is of length at most $|v| + |u| \leq 2|v|$), but it turns out that in the case v-parametric powers the argument is even simpler: the v-reduced prefix of u-parametric power is of length at most $2|v|$, so the v-parametric power is modified by an additive $\mathcal{O}(1)$ summand.

▶ **Lemma 13.** *We can compute and sort in $\mathcal{O}(m + n/|u|)$ time a set of $\mathcal{O}(\sqrt{n|u_{i_0}u_{i_0+1}|}/|u|)$ integer expressions E such that for every j if u^ϕ is a parametric power in $W_{J=j}$ not affected by substitution $J = j$ then $\phi \in E$.*

A similar set of $\mathcal{O}(\sqrt{n|u_{i_0}u_{i_0+1}|}/|v|)$ integer expressions can be computed for the not affected v-parametric powers after the second v-reduction in $\mathcal{O}(m + n/|v|)$ time.

The algorithm works by simple grouping of the parametric powers after the u- and v-reductions, the running times are obtained by appropriate usage of the data structure.

Lemmata 9, 10, 12 and 13 are enough to prove Lemma 11, by a simple case distinction, as described in text preceding Lemma 9.

What is left to show is how to compute candidate solutions, when one of I, J, say J, is already fixed, as in the claim of Lemma 11. The analysis is similar as in the case of two parameters, however, we cannot guarantee that after the u-reduction the coefficient at the u-parametric powers are ± 1. On the positive side, as there is only one integer variable, we can apply Lemma 7 directly. The additional logarithmic in the running time is due to sorting, which now cannot be done using counting sort, as the involved numbers may be large.

▶ **Lemma 14.** *For any given j in $\mathcal{O}(m)$ time we can decide, whether for each $i \in \mathbb{Z}$ the $\alpha u^i v^j \beta$ is a solution of (2) and if not then in $\mathcal{O}(m + n \log m/|u|)$ time compute a superset (of size $\mathcal{O}(n/|u|)$) of is such that $\alpha u^i v^j \beta$ is a solution.*

A similar claim holds for any fixed i (with superset size $\mathcal{O}(n/|v|)$ and running time $\mathcal{O}(m + n \log m/|v|)$).

3.1.2 $u \in \{v, \overline{v}\}$

When $u \in \{v, \overline{v}\}$ then $u^I v^J \approx u^{I+J}$ or $u^I v^J \approx u^{I-J}$ and we can replace the parameter $I + J$ (or $I - J$) with a single I. This case is subsumed by the case when we fix one of the parameters (i.e. I or J), see Lemma 14.

3.1.3 $u \sim v$

In this case either $u = u'u''$ and $v = u''u'$ or $v = \overline{u'}\,\overline{u''}$, for some u', u''. By substituting $v = \overline{v}$ we reduce the latter case to the former. We consider the parametric solution $\alpha u^I v^J \beta$, note that $v \approx u''u\overline{u''}$ and so $\alpha u^I v^J \beta \approx \alpha u^I u'' u^J \overline{u''} \beta$. From now on the approach is similar as when $u \not\sim v$. Most of the arguments are simpler, however, the extra technicality is that after the u-reduction we can have u-parametric power of the form $u^{\pm(I+J)+c}$. As a result, we consider not only substitutions $I = i$ and $J = j$, but also $I + J = k$, i.e. we substitute ϕ with $\phi(I, k - I)$, which depends only on I. This requires some additional cases to consider and makes some formulations longer, but everything follows in a similar way.

Overall, the main characterization is

▶ **Lemma 15** (cf. Lemma 11). *Given the equation (3) we can compute in $\mathcal{O}(mn/|u|)$ time sets $S_I, S_J, S_{I+J}, S_{I+J,\mathbb{Z}} \subseteq \mathbb{Z}$ and $S_{I,J} \subseteq \mathbb{Z}^2$, where $|S_I|, |S_J| = \mathcal{O}(\sqrt{n|u_{i_0}u_{i_0+1}|}/|u|)$, $|S_{I+J}| = \mathcal{O}(1)$, $|S_{I+J,\mathbb{Z}}|, |S_{I,J}| = \mathcal{O}(n/|u|)$ such that if (i, j) is a solution of (3) then at least one of the following holds:*

- *$i \in S_I$ or*
- *$j \in S_J$ or*
- *$i + j \in S_{I+J}$ or*
- *$i + j \in S_{I+J,\mathbb{Z}}$ and for each i' the $(i', (i + j) - i')$ is a solution or*
- *$(i, j) \in S_{I,J}$.*

Similar sets corresponding to substitutions $I = i$ and $J = j$ can be computed in the same time bounds.

The fourth possibility in Lemma 15 means that $W(I, k - I) \approx \varepsilon$, which would yield an infinite family of solutions $\{\alpha u^i v^{k-i} \beta : i \in \mathbb{Z}\}$. Additional combinatorial analysis yields that this cannot happen (and we know this from the earlier characterisation of the solution set).

▶ **Lemma 16.** *Consider a parametric word $\alpha u^I v^J \beta$ for $u \sim v$ and the corresponding $W \neq \varepsilon$ obtained after the substitution of $X = \alpha u^I v^J \beta$, as in (3). Then for every k it holds that $W(I, k - I) \not\approx \varepsilon$.*

3.2 Algorithm and running time

By Lemma 2 one solution of the form $\alpha u^i v^j \beta$, for fixed $\alpha, u, i, v, j, \beta$ which are $\mathcal{O}(1)$-represented, can be tested in $\mathcal{O}(m)$ time. So it is enough to show that there are at most $\mathcal{O}(n^2)$ different candidates tested (we estimate other computation times as well). Lemma 3 yields that there are $\mathcal{O}(n^2)$ candidate solutions (from the set S). Other solutions are obtained in the following way: for two consecutive words u_{i_0}, u_{i_0+1} from the equation we have a family of $\ell_{i_0} \cdot \ell'_{i_0}$ candidates of the form $\alpha u^I v^J \beta$, see Lemma 3, where $\ell_{i_0} = |u|, \ell'_{i_0} = |v|$ and $\ell_{i_0}, \ell'_{i_0} \leq |u_{i_0}| + |u_{i_0+1}|$; by Lemma 3 the total time, over all i_0, spent on computing words α, β, u, v is $\mathcal{O}(n^2)$ time. We will often use the estimation (a similar one hold for ℓ'_{i_0}):

$$\sum_{i_0=1}^{m} \ell_{i_0} \leq \sum_{i_0=1}^{m} |u_{i_0}| + |u_{i_0+1}| \leq 2n \ . \tag{4}$$

Suppose first that $u \not\sim v$, then by Lemma 11 we can compute in time $\mathcal{O}(mn/\ell_{i_0})$ sets S_I, S_J, $S_{J,\mathbb{Z}}$, $S_{I,J}$, where $|S_I| = \mathcal{O}\left(\sqrt{n|u_{i_0}u_{i_0+1}|}/l_{i_0}\right)$, $|S_J| = \mathcal{O}(1)$, $|S_{\mathbb{Z},J}|, |S_{I,J}| = \mathcal{O}(n/\ell_{i_0})$, such that for each solution (i, j) at least one of the following holds:

i1. $i \in S_I$ or
i2. $j \in S_J$ or
i3. $j \in S_{\mathbb{Z},J}$ and (i', j) is a solution for each i' or
i4. $(i, j) \in S_{I,J}$.

and in time $\mathcal{O}(mn/\ell'_{i_0})$ sets S'_J, S'_I, $S'_{I,\mathbb{Z}}$, $S'_{I,J}$, where $|S'_I| = \mathcal{O}(1)$, $|S'_J| = \mathcal{O}(\sqrt{n|u_{i_0}u_{i_0+1}|}/\ell'_{i_0})$ $|S'_{I,\mathbb{Z}}|, |S'_{I,J}| = \mathcal{O}(n/\ell'_{i_0})$, such that for each solution (i, j) at least one of the following holds:

j1. $j \in S'_J$ or
j2. $i \in S'_I$ or
j3. $i \in S_{I,\mathbb{Z}}$ and (i, j') is a solution for each j' or
j4. $(i, j) \in S'_{I,J}$.

As both of those characterization hold, we should describe how do we treat each of the 16 cases. Fortunately, for most of the cases the further action and analysis depends on one of the cases alone.

If we are in the case i4 or j4 then we test each pair $(i, j) \in S_{I,J} \cup S'_{I,J}$ separately. There are (over all $0 \leq i_0 \leq m-1$) at most (note that some of those solutions have $u \sim v$, we will estimate their running time separately, so now we overestimate the running time)

$$\sum_{i_0=0}^{m-1} \ell_{i_0}\ell'_{i_0}\left(\frac{n}{\ell_{i_0}} + \frac{n}{\ell'_{i_0}}\right) = n\sum_{i_0=0}^{m-1} \ell'_{i_0} + \ell_{i_0} \leq 4n^2 \qquad \text{by (4)} \tag{5}$$

such solutions.

Concerning the time of establishing those sets, the largest is from Lemma 12 and it is $\mathcal{O}(mn/\ell_{i_0})$ (for $S_{I,J}$) or $\mathcal{O}(mn/\ell'_{i_0})$ (for $S'_{I,J}$). So up to a constant it is:

$$\sum_{i_0=0}^{m-1} \ell_{i_0}\ell'_{i_0}\left(\frac{mn}{\ell_{i_0}} + \frac{mn}{\ell'_{i_0}}\right) = mn\sum_{i_0=0}^{m-1}(\ell_{i_0} + \ell'_{i_0}) \leq 2mn^2 \qquad \text{by (4)} \ .$$

If we are in the case i3 then for each $j \in S_{J,\mathbb{Z}}$ we substitute $J = j$ and test, whether $W(I, j) \approx \varepsilon$; by Lemma 7 this is equivalent to (i', j) being a solution for each $i' \in \mathbb{Z}$. Each such j yields a family of solutions of the required form $\{\underbrace{\alpha}_{\text{fixed}} u^i \underbrace{v^j\beta}_{\text{fixed}} : i \in \mathbb{Z}\}$ and there are at most $|S_{J,\mathbb{Z}}| = \mathcal{O}(n/\ell_{i_0})$ such families. Over all $0 \leq i_0 \leq m-1$ this yields at most (up to a constant)

$$\sum_{i_0=0}^{m-1} \ell_{i_0}\ell'_{i_0}\frac{n}{\ell'_{i_0}} = n\sum_{i_0=0}^{m-1} \ell_{i_0} \leq 2n^2 \qquad \text{by (4)} \ .$$

Concerning the running time, note that testing whether $W(I,j) \approx \varepsilon$ takes $\mathcal{O}(m)$ time, see Lemma 14, so it is enough to show that we test $\mathcal{O}(n^2)$ such js. As $|S_{\mathbb{Z},J}| = \mathcal{O}(n/\ell_{i_0})$, the calculations are as in (5). A similar analysis applies to $S_{I,\mathbb{Z}}$, i.e. case j3.

If we are in case i2 then for each $j \in S_J$ we can compute, by Lemma 14, in time $\mathcal{O}(m)$ whether each (i',j) is a solution, note that the set is of the required form, as in the case of $j \in S_{\mathbb{Z},J}$, moreover the estimation on the number of such solution sets is not larger than in the case of $j \in S_{\mathbb{Z},J}$, as $|S_J| = \mathcal{O}(1)$ and $|S_{\mathbb{Z},J}| = \mathcal{O}(n/\ell_{i_0})$. Otherwise, again by Lemma 14, we compute in time $\mathcal{O}(m + n/\ell_{i_0} \log m))$ a set S of size $|S| = \mathcal{O}(n/\ell_{i_0})$ such that if (i,j) is a solution then $i \in S$. This yields $|S_J| \times |S| = \mathcal{O}(n/\ell_{i_0})$ candidate pairs, which are individually tested, so the running time is $\mathcal{O}(mn/\ell_i)$, note that this dominates $\mathcal{O}(m + n \log m/\ell_{i_0})$ from Lemma 14. The estimation in (5) yields that there are at most $\mathcal{O}(n^2)$ such candidate pairs and the whole running time is $\mathcal{O}(mn^2)$. A similar analysis applies to $i \in S'_I$.

The only remaining option is that we are simultaneously in case i1 and j1, i.e. $i \in S_I$ and $j \in S'_J$. As $|S_I| = \mathcal{O}\left(\sqrt{n|u_{i_0}u_{i_0+1}|}/\ell_i\right)$ and $|S'_J| = \mathcal{O}\left(\sqrt{n|u_{i_0}u_{i_0+1}|}/\ell'_i\right)$ there are (over all i_0 and up to a constant) at most

$$\sum_{i_0=1}^{m} \ell_{i_0}\ell'_{i_0} \frac{\sqrt{n|u_{i_0}u_{i_0+1}|}}{\ell_{i_0}} \cdot \frac{\sqrt{n|u_{i_0}u_{i_0+1}|}}{\ell'_{i_0}} = \sum_{i_0=1}^{m} n|u_{i_0}u_{i_0+1}| \leq 2n^2$$

such solutions tested.

The cases of $u = v$ or $u = \overline{v}$ are done using Lemma 14, the bounds are the same as in case of $u \not\sim v$.

The case of $u \sim v$ is a bit more involved, let $\ell_{i_0} = |u|$. By Lemma 15 we can compute in time $\mathcal{O}(mn/|u|)$ sets S_I, S_J, S_{I+J}, $S_{I+J,\mathbb{Z}}$, $S_{I,J}$ and such that for each solution (i,j) either

1. $i \in S_I$ or
2. $j \in S_J$ or
3. $i + j \in S_{I+J}$ or
4. $i + j \in S_{I+J,\mathbb{Z}}$ and for each i' the $(i', (i+j) - i')$ is a solution or
5. $(i,j) \in S_{I,J}$.

The third case is dealt with as previously (for each $i+j$ we make a substitution $I + J = i + j$, check, whether the obtained equation is trivial and solve the corresponding equation), similarly fourth (for each $i + j$ we substitute $I + J = i + j$ and check whether the obtained word is ε; note that it can be shown that this never holds, see Lemma 16) and fifth (we substitute $I = i, J = j$ and test). So we are left only with the first two cases. Moreover, Lemma 15 also gives us a similar characterization resulting from a substitution $I = i$, again there are 5 cases and the last three of them are dealt with similarly, the first two give that there are sets S'_J, S'_{I+J} such that

1. $j \in S'_J$ or
2. $i + j \in S'_{I+J}$

and applied to substitution $J = j$ again gives 5 cases, the last three of which are dealt with and the first two yield that there are sets S''_I, S''_{I+J} such that

1. $i \in S''_I$ or
2. $i + j \in S''_{I+J}$.

There are in total 8 cases (we choose one of two options for three substitutions), in each such a case from the three choices some two (though not each two) allow to give $\mathcal{O}\left(n|u_{i_0}u_{i_0+1}|/\ell^2_{i_0}\right)$ candidates for (i,j) : say if $i \in S_I$, $j \in S'_J$ and $i + j \in S''_{I+J}$ then any two determine (i,j) and when $j \in S_J$, $j \in S'_J$ and $i + j \in S''_{I+J}$ then $j \in S_J \cap S'_J$ and $i + j \in S''_{I+J}$. The rest of the calculations is the same as in the case of $u \not\sim v$.

References

1 Kenneth I. Appel. One-variable equations in free groups. *Proceedings of the American Mathematical Society*, 19:912–918, 1968.

2 Kenneth I. Appel. On two variable equations in free groups. *Proceedings of the American Mathematical Society*, 21:179–184, 1969.

3 Dimitri Bormotov, Robert Gilman, and Alexei Myasnikov. Solving one-variable equations in free groups. *Journal of Group Theory*, 12:317–330, 2009. doi:10.1515/JGT.2008.080.

4 Witold Charatonik and Leszek Pacholski. Word equations with two variables. In *IWWERT*, pages 43–56, 1991. doi:10.1007/3-540-56730-5_30.

5 Ian M. Chiswell and Vladimir N. Remeslennikov. Equations in free groups with one variable. I. *Journal of Group Theory*, 3(4), 2000. doi:10.1515/jgth.2000.035.

6 Joel D. Day, Vijay Ganesh, Paul He, Florin Manea, and Dirk Nowotka. The satisfiability of word equations: Decidable and undecidable theories. In Igor Potapov and Pierre-Alain Reynier, editors, *Reachability Problems - 12th International Conference, RP 2018, Marseille, France, September 24-26, 2018, Proceedings*, volume 11123 of *Lecture Notes in Computer Science*, pages 15–29. Springer, 2018. doi:10.1007/978-3-030-00250-3_2.

7 Volker Diekert. Makanin's algorithm. In M. Lothaire, editor, *Algebraic Combinatorics on Words*, chapter 12, pages 342–390. Cambridge University Press, 2002.

8 Volker Diekert, Claudio Gutiérrez, and Christian Hagenah. The existential theory of equations with rational constraints in free groups is PSPACE-complete. *Inf. Comput.*, 202(2):105–140, 2005. doi:10.1016/j.ic.2005.04.002.

9 Volker Diekert, Artur Jeż, and Wojciech Plandowski. Finding all solutions of equations in free groups and monoids with involution. *Inf. Comput.*, 251:263–286, 2016. doi:10.1016/j.ic.2016.09.009.

10 Robert Dąbrowski and Wojciech Plandowski. Solving two-variable word equations. In *ICALP*, pages 408–419, 2004. doi:10.1007/978-3-540-27836-8_36.

11 Robert Dąbrowski and Wojciech Plandowski. On word equations in one variable. *Algorithmica*, 60(4):819–828, 2011. doi:10.1007/s00453-009-9375-3.

12 Robert H. Gilman and Alexei G. Myasnikov. One variable equations in free groups via context free languages. *Contemporary Mathematics*, 349:83–88, 2004.

13 Yu. I. Hmelevskiĭ. *Equations in Free Semigroups*. Number 107 in Proceedings Steklov Institute of Mathematics. American Mathematical Society, 1976. Translated from the Russian original: Trudy Mat. Inst. Steklov. 107, 1971.

14 Lucian Ilie and Wojciech Plandowski. Two-variable word equations. *RAIRO Theor. Informatics Appl.*, 34(6):467–501, 2000. doi:10.1051/ita:2000126.

15 Artur Jeż. One-variable word equations in linear time. *Algorithmica*, 74:1–48, 2016. doi:10.1007/s00453-014-9931-3.

16 Artur Jeż. Recompression: a simple and powerful technique for word equations. *J. ACM*, 63(1):4:1–4:51, March 2016. doi:10.1145/2743014.

17 Juha Kärkkäinen, Peter Sanders, and Stefan Burkhardt. Linear work suffix array construction. *J. ACM*, 53(6):918–936, 2006. doi:10.1145/1217856.1217858.

18 Toru Kasai, Gunho Lee, Hiroki Arimura, Setsuo Arikawa, and Kunsoo Park. Linear-time longest-common-prefix computation in suffix arrays and its applications. In *CPM*, pages 181–192, 2001. doi:10.1007/3-540-48194-X_17.

19 Olga Kharlampovich, Igor G. Lysënok, Alexei G. Myasnikov, and Nicholas W. M. Touikan. The solvability problem for quadratic equations over free groups is NP-complete. *Theory of Computing Systems*, 47(1):250–258, 2010. doi:10.1007/s00224-008-9153-7.

20 Olga Kharlampovich and Alexei Myasnikov. Elementary theory of free non-abelian groups. *Journal of Algebra*, 302:451–552, 2006.

21 Antoni Kościelski and Leszek Pacholski. Makanin's algorithm is not primitive recursive. *Theor. Comput. Sci.*, 191(1-2):145–156, 1998. doi:10.1016/S0304-3975(96)00321-0.

22 Markku Laine and Wojciech Plandowski. Word equations with one unknown. *Int. J. Found. Comput. Sci.*, 22(2):345–375, 2011. doi:10.1142/S0129054111008088.

23 A. A. Lorents. Representations of sets of solutions of systems of equations with one unknown in a free group. *Dokl. Akad. Nauk. SSSR*, 178:290–292, 1968.

24 Roger C. Lyndon and Marcel-Paul Schützenberger. The equation $a^M = b^N c^P$ in a free group. *Michigan Mathematical Journal*, 9(4):289–298, 1962.

25 Roger Conant Lyndon. Equations in free groups. *Transansaction of American Mathematical Society*, 96:445–457, 1960. doi:10.1090/S0002-9947-1960-0151503-8.

26 Gennadiĭ Makanin. The problem of solvability of equations in a free semigroup. *Matematicheskii Sbornik*, 2(103):147–236, 1977. (in Russian).

27 Gennadiĭ Makanin. Equations in a free group. *Izv. Akad. Nauk SSR*, Ser. Math. 46:1199–1273, 1983. English transl. in Math. USSR Izv. 21 (1983).

28 Jakob Nielsen. Über die Isomorphismen unendlicher Gruppen ohne Relation. *Mathematische Annalen*, 79:269–272, 1918.

29 Dirk Nowotka and Aleksi Saarela. An optimal bound on the solution sets of one-variable word equations and its consequences. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 136:1–136:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.136.

30 Seraphin D. Eyono Obono, Pavel Goralčik, and Marianne Maksimenko. Efficient solving of the word equations in one variable. In *MFCS*, pages 336–341, 1994. doi:10.1007/3-540-58338-6_80.

31 Wojciech Plandowski. Satisfiability of word equations with constants is in PSPACE. *J. ACM*, 51(3):483–496, 2004. doi:10.1145/990308.990312.

32 Alexander A. Razborov. *On Systems of Equations in Free Groups*. PhD thesis, Steklov Institute of Mathematics, 1987. In Russian.

33 Zlil Sela. Diophantine geometry over groups VI: the elementary theory of a free group. *Geometric & Functional Analysis*, 16:707–730, 2006. doi:10.1007/s00039-006-0565-8.