Nonuniform Deterministic Finite Automata over Finite Algebraic Structures

Paweł M. Idziak
Institute of Theoretical Computer Science, Jagiellonian University, Krakow, Poland

Piotr Kawałek
Institute of Discrete Mathematics and Geometry, TU Wien, Austria
Institute of Theoretical Computer Science, Jagiellonian University, Krakow, Poland

Jacek Krzaczkowski
Institute of Computer Science and Mathematics, University of Maria Curie-Skłodowska, Lublin, Poland

Abstract

Nonuniform deterministic finite automata (NUDFA) over monoids were invented by Barrington in [2] to study boundaries of nonuniform constant-memory computation. Later, results on these automata helped to identify interesting classes of groups for which equation satisfiability problem (PolSat) is solvable in (probabilistic) polynomial time [13, 26]. Based on these results, we present a full characterization of groups, for which the identity checking problem (called PolEqv) has a probabilistic polynomial-time algorithm. We also go beyond groups, and propose how to generalise the notion of NUDFA to arbitrary finite algebraic structures. We study satisfiability of these automata in this more general setting. As a consequence, we present a full description of finite algebras from congruence modular varieties for which testing circuit equivalence CEqv can be solved by a probabilistic polynomial-time procedure. In our proofs we use two computational complexity assumptions: randomized Expotential Time Hypothesis and Constant Degree Hypothesis.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Complexity classes

Keywords and phrases program satisfiability, circuit equivalence, identity checking

Digital Object Identifier 10.4230/LIPIcs.ICALP.2025.161

Category Track B: Automata, Logic, Semantics, and Theory of Programming

Related Version *Full Version*: https://arxiv.org/abs/2501.12260 [24]

Funding *Piotr Kawałek*: This research was funded in whole or in part by National Science Centre, Poland #2021/41/N/ST6/03907. For the purpose of Open Access, the author has applied a CC-BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission. Funded by the European Union (ERC, POCOCOP, 101071674). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.
Jacek Krzaczkowski: This research was funded in whole or in part by National Science Centre, Poland #2022/45/B/ST6/02229.
For the purpose of Open Access, the author has applied a CC-BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

Acknowledgements The second and third authors would like to express their deep gratitude to Paweł Idziak, who passed away during the preparation of this paper. It was thanks to him that we met and collaborated for many years on problems related to satisfiability and equivalence in finite algebraic structures. His contributions to Algebra, Logic, and Computer Science will be remembered by many. Rest in peace, Paweł.

52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025).
Editors: Keren Censor-Hillel, Fabrizio Grandoni, Joël Ouaknine, and Gabriele Puppis
Article No. 161; pp. 161:1–161:14
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

There are many interactions between mathematics and (theoretical) computer science. Many branches of these sciences influence each other, sometimes in quite surprising ways. A relatively recent example of such an influence is the so-called algebraic approach to Constraint Satisfaction Problem (CSP) which led to a complete classification of computational complexity of CSP [8, 36]. It is really impressive how in this case (universal) algebra, combinatorics, logic, computational complexity and algorithmic work together to give new results in each of these fields.

Another example of synergy between different fields of mathematics and theoretical computer science can be observed on the borderline of circuit complexity, automata theory and (universal) algebra. The most significant example here is the role of monoids played in automata theory and formal languages.

Usually a deterministic finite automaton (DFA) is determined by an alphabet Σ acting over a set Q of states by a function $\delta : \Sigma \times Q \ni (\sigma, q) \longmapsto \sigma \cdot q \in Q$. This action can be extended (in an obvious way) to the action of the free monoid Σ^*. To decide if a word $w \in \Sigma^*$ is accepted by a particular DFA we need to endow it with a starting state q_0 and a set $F \subseteq Q$ of accepting states. Then w gets accepted if $w \cdot q_0 \in F$. For our purposes we prefer, first to treat the set Q^Q of functions as the monoid with $f \cdot g = g \circ f$, and then to treat the action δ as a function $a : \Sigma \longrightarrow Q^Q$ given by $a(\sigma)(q) = \sigma \cdot q$. Now, the word $\sigma^1 \ldots \sigma^n$ gets accepted if $a(\sigma^1) \cdot \ldots \cdot a(\sigma^n) \in S$, where S consists of transitions determined by the words $w \in \Sigma^n$ satisfying $w \cdot q_0 \in F$.

Before restating Barrington's definition of Non-uniform Deterministic Finite Automata (NUDFA) over monoids [3] we note that $a(\sigma^1) \cdot \ldots \cdot a(\sigma^n)$ is nothing else but $\mathbf{t}_n(a(\sigma^1), \ldots, a(\sigma^n))$, where $\mathbf{t}_n(x_1, \ldots, x_n) = x_1 \cdot \ldots \cdot x_n$. In a NUDFA over the monoid $\mathbf{M}$ to accept the word from Σ^n we are going to relax the term $x_1 \cdot \ldots \cdot x_n$ to an arbitrary semigroup term $\mathbf{t}(x_1, \ldots, x_k)$ and replace one action $a : \Sigma \longrightarrow Q^Q$ by a bunch of functions $a^x : \Sigma \longrightarrow M$, one for each variable of $\mathbf{t}$. Now a $\mathbf{t}$-program (with inputs from Σ^n represented by an n-variable word $b_1 \ldots b_n$) consists of:

- a set of k-instructions, one for each variable x of $\mathbf{t}$, of the form $\iota(x) = (b^x, a^x)$, where b^x is one of the variables b_i,
- and a set $S \subseteq M$ of accepting values.

Finally, a NUDFA over $\mathbf{M}$ is a sequence (possibly even nonrecursive) of programs $(\mathbf{t}_n, n, \iota_n, S_n)_{n \in \mathbb{N}}$ with $\mathbf{t}_n(x_1, \ldots, x_{k_n})$ being some terms of $\mathbf{M}$, $S_n \subseteq M$ and ι_n being the instructions for the variables of $\mathbf{t}_n$. A word $b_1 \ldots b_n \in \Sigma^n$ gets accepted by such a NUDFA if $\mathbf{t}_n(a^{x_1}(b^{x_1}), \ldots, a^{x_{k_n}}(b^{x_{k_n}})) \in S_n$.

▶ **Example 1.** Let $\Sigma = \{a, b, c\}$, $M = (\mathbb{Z}_2, +) \times (\mathbb{Z}_2, +)$ and for every $n \in \mathbb{N}$ let

- $t_n(x_1, \ldots, x_{2n}) = x_1 + x_2 + \ldots + x_{2n}$,
- $\iota_n(x_i) = (b_{\lceil i/2 \rceil}, a^{x_i})$, where

$$a^{x_i}(b^{x_i}) = \begin{cases} (1,0) & i \text{ is even and } b^{x_i} \in \{a, b\}, \\ (0,1) & i \text{ is odd and } b^{x_i} \in \{b, c\}, \\ (0,0) & \text{otherwise}, \end{cases}$$

- $S_n = \{(0, 1)\}$.

Now, to determine the language accepted by the NUDFA consisting of the sequence of programs $(\mathbf{t}_n, n, \iota_n, S_n)_{n \in \mathbb{N}}$ note that $\mathbf{t}_n(a^{x_1}(b^{x_1}), \ldots, a^{x_{k_n}}(b^{x_{k_n}}))$ is the sum of:

- $n_{a,b}$ values $(1,0)$, where $n_{a,b}$ is the number of occurrences of letters 'a' and 'b' in the input word,
- $n_{b,c}$ values $(0,1)$, where $n_{b,c}$ is the number of occurrences of letters 'b' and 'c' in the input word,
- some number of values $(0,0)$.

This sum is equal $(0,1)$, and hence the word is accepted by the NUDFA, iff the following holds:

- parities of the numbers of occurrences of letters 'a' and 'b' in word $b_1 \ldots b_n$ are equal,
- parities of the numbers of occurrences of letters 'b' and 'c' in word $b_1 \ldots b_n$ are not equal.

Originally Barrington considered mainly the Boolean case where $\Sigma = \{0,1\}$ to study computational boundaries of non-uniform constant-memory computation. Such automata can be used to compute the Boolean functions of the form $\{0,1\}^n \longrightarrow \{0,1\}$. They also give an interesting algebraic insight into the internal structure of the class NC^1 [6]. Note that in the Boolean case $\Sigma = \{0,1\}$ the function a^x is given by the pair of values $a^x(0), a^x(1)$. In such a case, for simplicity we write $\iota(x) = (b^x, a^x(0), a^x(1))$.

A natural question that arises here is whether the language accepted by a particular NUDFA over $\mathbf{M}$ is nonempty. This problem reduces to:

ProgSat($\mathbf{M}$): Decide if a given program over $\mathbf{M}$ accepts at least one word.

The problem ProgSat proved itself to be extremely useful in studying groups (and monoids) for which determining if an equation has a solution (PolSat) is in P. In fact the proof of Goldmann and Russell in [13] that each nilpotent group $\mathbf{G}$ has tractable PolSat($\mathbf{G}$) modifies a polynomial time algorithm for ProgSat($\mathbf{G}$). A study of connections between PolSat and ProgSat for finite monoids is given in [5]. Recently a complete classification of finite groups $\mathbf{G}$ with ProgSat($\mathbf{G}$) $\in$ RP, together with its consequences for PolSat has been provided by Idziak, Kawałek and Krzaczkowski in [26].

Now the results of [26] allow us to complete the long term extensive investigations [9, 20, 17, 18, 11, 35, 25] on equivalence problem PolEqv of polynomials (i.e. terms with some variables already evaluated) over finite groups.

The lower bound in our characterization relies on the randomized version of the Exponential Time Hypothesis (rETH), while the upper bounds explore the so-called Constant Degree Hypothesis (CDH). This hypothesis, introduced in [6], can be rephrased to state that for a fixed integer d, a prime p and an integer m which is not just a power of p, any 3-level circuit of the form $\mathsf{AND}_d \circ \mathsf{MOD}_m \circ \mathsf{MOD}_p$ requires exponential size to compute AND_n of arbitrary large arity n (in which AND_d gates are in the input layer, MOD_m gates are in the middle layer, and there is one MOD_p gate in the output layer). The best known lower bound, due to Chattopadhyay et al. [10], for the size of $\mathsf{AND}_d \circ \mathsf{MOD}_m \circ \mathsf{MOD}_p$ computing AND_n is only superlinear. Much earlier CDH has been considered in many different contexts. Already in [6] the case $d = 1$, i.e. no AND_d layer, has been confirmed. Also restrictions put either on the number of AND_d used locally, or on the local structure of $\mathsf{AND}_d \circ \mathsf{MOD}_m$ fragments, allowed Grolmusz and Tardos [15, 14] to confirm CDH. Very recently Kawałek and Weiß [30] confirmed CDH for symmetric circuits.

In this paper, under these two complexity hypotheses (i.e. rETH and CDH), the characterization of finite groups with tractable ProgSat from [26] is applied to obtain an unexpectedly different characterization for tractable PolEqv.

▶ **Theorem 1.1.** *Let $\mathbf{G}$ be a finite group. Assuming rETH and CDH the problem* PolEqv($\mathbf{G}$) *is in coRP if and only if $\mathbf{G}$ is solvable and has a nilpotent normal subgroup $\mathbf{H}$ with the quotient $\mathbf{G}/\mathbf{H}$ being also nilpotent.*

A surprising part of these investigations is that such characterization can not only be done, but that it can be stated, in terms of algebraic structure of the groups. In fact this reveals another connection between (universal) algebra and circuit complexity theory.

The goal of this paper is to leave the group realm and generalize Theorem 1.1 to a much broader setting of algebras. As we will see shortly this generalization is two-fold. First we generalize the concept of NUDFAs to cover automata working over arbitrary finite algebraic structure $\mathbf{A}$. This requires to define a program over $\mathbf{A}$ and can be done by simply replacing the monoid $\mathbf{M}$ by the algebra $\mathbf{A}$ and assume that this time $\mathbf{t}$ is a term of the algebra $\mathbf{A}$.

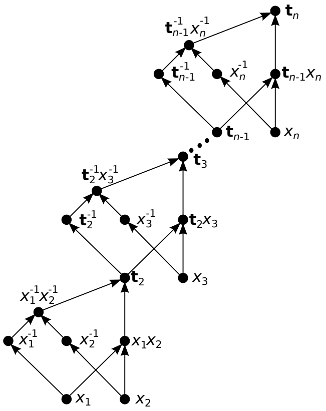

Figure 1 Compressing the size of $\mathbf{t}_n$. ©Idziak, Krzaczkowski [27]

Second, in general algebraic context it is not clear which operations are to be chosen to be the basic ones. And this choice may be extremely important from computational point of view. Recall after [27], that adding the binary commutator operation $[x, y] = x^{-1}y^{-1}xy$ to the language of a group may exponentially shorten the size of the input. Indeed the term $\mathbf{t}_n(x_1, \dots, x_n) = [\dots[[x_1, x_2], x_3]\dots, x_n]$ has linear size if the commutator operation is allowed, while after writing this term in the pure group language (of multiplication and the inverse) we see that the size $|\mathbf{t}_n|$ of $\mathbf{t}_n$ is $2\,|\mathbf{t}_{n-1}|+2$, as $\mathbf{t}_n(x_1, \dots, x_n) = \mathbf{t}_{n-1}(x_1, \dots, x_{n-1})^{-1} \cdot x_n^{-1} \cdot \mathbf{t}_{n-1}(x_1, \dots, x_{n-1}) \cdot x_n$, so that $|\mathbf{t}_n|$ is exponential in n. Actually for the alternating group $\mathbf{A}_4$ it is shown in [19] that PolSat($\mathbf{A}_4$) is in P, while after endowing $\mathbf{A}_4$ with the commutator operation (and therefore shortening the size of inputs) the problem becomes NP-complete. A solution to this phenomenon has been proposed by Idziak and Krzaczkowski in [27] by presenting a term by the algebraic circuit that computes it. For example $\mathbf{t}_n(x_1, \dots, x_n)$ can be computed by the circuit of size $6n - 5$ as shown by Figure 1.

Representing polynomials of an algebra with algebraic circuits leads to a modified version of ProgSat which we explore in this paper.

ProgCSat($\mathbf{A}$): Decide if a given program $(\mathbf{t}, n, \iota, S)$ over $\mathbf{A}$ accepts at least one word, where $\mathbf{t}$ is given by a circuit over $\mathbf{A}$.

Measuring the size of the input, i.e. the expression of the form $\mathbf{p} = \mathbf{q}$ by the lengths of the polynomials $\mathbf{p}$ and $\mathbf{q}$ or by the sizes of their algebraic circuits used to compute $\mathbf{p}$ and $\mathbf{q}$ give rise to either PolSat or CSat in the satisfiability setting or to PolEqv or CEqv in the equivalence setting. Note here that [27] argues that the complexity of CSat($\mathbf{A}$) and CEqv($\mathbf{A}$) is independent of which term operations of the algebra $\mathbf{A}$ are chosen to be the basic ones. This independence gives a hope for a characterization of algebras $\mathbf{A}$ with tractable CSat($\mathbf{A}$) or CEqv($\mathbf{A}$) in terms of algebraic structure of $\mathbf{A}$. Actually already a series of papers [21, 35, 23, 26, 27, 33] enforces a bunch of such necessary algebraic conditions for an algebra to have CSat or CEqv tractable. Not surprisingly solvability and nilpotency are among these conditions.

To define these two notions of solvability and nilpotency outside the group realm we need a notion of a commutator. However we need to work with a commutator $[\alpha, \beta]$ of two congruences α, β (that in the group setting correspond to normal subgroups) instead of a commutator of elements of an algebra. For more details on the definition and the properties of commutator, we refer to the book [12] and Section 2. Here we only note that this concepts of commutator of congruences works smoothly only in some restricted setting of the so-called congruence modular varieties, i.e. equationally definable classes of algebras with modular congruence lattices. Fortunately this setting includes groups, rings, quasigroups, loops, Boolean algebras, Heyting algebras, lattices and almost all algebras related to logic. In groups, rings or Boolean/Heyting algebras the congruences are determined by normal subgroups, ideals or filters respectively. Obviously this new concept of commutator of normal subgroups coincides with the old one. The commutator of two ideals I, J of a commutative ring is simply their algebraic product $I \cdot J$, while the commutator of filters in a Boolean/Heyting algebra is their intersection. Now we can say that a congruence α is abelian, nilpotent or solvable if $[\alpha, \alpha] = 0_{\mathbf{A}}$, $[\ldots[[\alpha, \alpha], \alpha], \ldots \alpha] = 0_{\mathbf{A}}$ or $[[[\alpha, \alpha], [\alpha, \alpha]], \ldots [[\alpha, \alpha], [\alpha, \alpha]]] = 0_{\mathbf{A}}$ respectively (for some number of nested commutators). Here, $0_{\mathbf{A}}$ is the identity relation/congruence of $\mathbf{A}$. The algebra $\mathbf{A}$ itself is said to be abelian, nilpotent or solvable if the total congruence $1_{\mathbf{A}}$ collapsing everything is abelian, nilpotent or solvable, respectively.

Note that, for nilpotent groups, Boolean programs (of NUDFAs) compute AND functions only of bounded arity, i.e. for each nilpotent group $\mathbf{G}$ there is a constant k such that AND_k is computable by no program over $\mathbf{G}$ [6]. This nonexpressibility phenomenon does not transfer to nilpotent algebras in general congruence modular context. The most natural example here is the algebra $(\mathbb{Z}_6; +, \%2)$, i.e. the group $(\mathbb{Z}_6; +)$ endowed with the unary operation $\%2$ computing the parity. In this algebra all the circuits of the form $\mathsf{MOD}_2 \circ \mathsf{MOD}_3$ can be modeled so that AND_n can be expressed for all n (however by exponential size of the circuits). This action of the prime 2 acting over prime 3 cannot occur in nilpotent groups. Indeed, due to the Sylow theorem, each finite nilpotent group is a product of p-groups. This decomposition prevents interaction between different primes, as they occur on different stalks/coordinates. And the lack of such interactions is crucial in bounding the arity of expressible AND_n's. Also in our considerations the finite nilpotent algebras that decompose into a product of algebras of prime power order occur naturally. They are known as supernilpotent algebras [7, 1]. We will return to this concept of supernilpotent algebras and its relativization to supernilpotent congruences in Section 2.

Now we are ready to state the other two main results of the paper.

▶ **Theorem 1.2.** *Let $\mathbf{A}$ be a finite algebra from a congruence modular variety. Assuming rETH and CDH the problem* ProgCSat($\mathbf{A}$) *is in RP if and only if $\mathbf{A}$ is nilpotent and has a supernilpotent congruence σ with supernilpotent quotient $\mathbf{A}/\sigma$ and such that all cosets of σ have sizes that are powers of the same prime number p.*

▶ **Theorem 1.3.** *Let $\mathbf{A}$ be a finite algebra from a congruence modular variety. Assuming rETH and CDH the problem* CEqv($\mathbf{A}$) *is in coRP if and only if $\mathbf{A}$ is nilpotent and has a supernilpotent congruence σ with supernilpotent quotient $\mathbf{A}/\sigma$.*

Results from [26] that we use in the proof of Theorem 1.1 heavily rely on a method of representing terms/polynomials of finite solvable group $\mathbf{G}$ by bounded-depth circuits that use only modular gates. Besides the obvious requirement that the circuit representing a group-polynomial $\mathbf{p}$ has to compute the very same function as $\mathbf{p}$ does, we also want to control the size of the circuit to be polynomial in terms of the size (length) of $\mathbf{p}$.

A very similar approach can be found in [32], where M. Kompatscher considers generalizations of finite nilpotent groups, i.e. nilpotent algebras from the congruence modular varieties. He provides a method to rewrite circuits over such algebras to constant-depth circuits which, again, use only modulo-counting gates. These modular circuits, which appear in both of the mentioned cases, are known as CC-circuits. However, since [32] does not use the notion of a program/NUDFA, the author formulates his results for circuits over **A** representing only some specific functions. For those functions, it is natural how to interpret Boolean values 0/1 in the non-Boolean algebra **A**. In our paper, the notion of a program/NUDFA provides us with a formal framework which helps to relate the expressive power of algebraic structures to some standard circuit complexity classes. Here, we present a very precise characterization of functions computable by programs over algebras corresponding to polynomial-time cases of ProgCSat.

▶ **Theorem 1.4.** *Let* $\mathbf{A}$ *be a finite nilpotent algebra from a congruence modular variety with supernilpotent congruence* σ *of* $\mathbf{A}$ *such that cosets of* α *are of prime power size* p^k *and* $\mathbf{A}/\sigma$ *is supernilpotent. Then the function computable by a Boolean program of size* ℓ *over the algebra* $\mathbf{A}$ *can be also computed by an* $\mathsf{AND}_d \circ \mathsf{MOD}_m \circ \mathsf{MOD}_p$*-circuits of size* $O(\ell^c)$ *with* d, m, c *being natural numbers depending only on* $\mathbf{A}$*, and* m *being relatively prime to* p*.*

This theorem not only is an interesting result on its own, but also is crucial in proving Theorem 1.2 and 1.3. Quite a technical proof of Theorem 1.4 can be found in the full version of the paper on arXiv [24]. It combines advanced tools of commutator theory [12], as well as many combinatorial lemmas on how to compose / compress circuits using gates MOD_m. This is in line (and in some parts extends) with some of the earlier works on these circuits (see for instance [14, 15, 22]).

2 Algebraic preliminaries

An algebra is a set called a universe together with a finite set of operations acting on it called basic operations of the algebra. We usually use boldface letter to denote the algebra and the very same letter, but with a regular font, to denote its universe. Pol **A** is the polynomial clone of **A**, that is the set of all polynomial operations of **A**. An algebra **A** is polynomially equivalent to an algebra **B** if it is isomorphic to an algebra which has the same set of polynomial operations as **B**. An induced algebra $\mathbf{A}|_S$ is a set S with all polynomial operations of **A** closed on S (or in other word polynomial operations that produce values in S when applied to arguments from S). An idempotent function is a function f such that $f(f(x)) = f(x)$.

In proofs of intractability of ProgCSat and CEqv for algebras from congruence modular varieties the crucial role is played by Tame Congruence Theory (see [16] for details). This is a deep algebraic tool describing local behavior of finite algebras. TCT shows that locally finite algebras behave in one of following five ways:

1. a finite set with a group action on it,
2. a finite vector space over a finite field,
3. a two-element Boolean algebra,
4. a two-element lattice,
5. a two-element semilattice.

By $\mathrm{typ}\{\mathbf{A}\}$, let us denote a subset of $\{\mathbf{1}, .., \mathbf{5}\}$ which describes the local behaviours we can find in an algebra **A**. Note that in a case of algebra **A** from a congruence modular variety only three types can appear in $\mathrm{typ}\{\mathbf{A}\}$, that is types **2**, **3** and **4**. In case of types **3** and **4** there has to be a two-element set U (whose elements we call 0 and 1) such that

- there is a polynomial $\mathbf{e}$ of $\mathbf{A}$ fulfilling $\mathbf{e}(A) = U$,
- there are polynomials of $\mathbf{A}$ which behaves on U like $\wedge$ and $\vee$,
- in case of type $\mathbf{3}$ there is also unary polynomial of $\mathbf{A}$ which is a negation on U.

If we can find type $\mathbf{3}$ or $\mathbf{4}$ in $\operatorname{typ}\{\mathbf{A}\}$, the complexity of both ProgCSat and CEqv is relatively easy to determine, as we shall see in forthcoming chapters. For this reason the most of the volume of the paper is devoted to algebras with $\operatorname{typ}\{\mathbf{A}\} = \{\mathbf{2}\}$. In the congruence modular variety those are precisely the solvable algebras [12, 16]. In fact, some of the earlier papers already dealt with algebras that are solvable but not nilpotent, so we will be mostly concerned with the notion of nilpotency and its properties.

Every solvable (so in particular - nilpotent) algebra in the congruence modular variety is Malcev, i.e. it possesses a polynomial operation $\mathbf{d}$ satisfying the following identity: $\mathbf{d}(y, x, x) = \mathbf{d}(x, x, y) = y$. A standard example of Malcev algebras are groups with a Malcev term of the form $x \cdot y^{-1} \cdot z$. Unlike for groups, nilpotent algebras do not necessarily decompose into a direct product of algebras of prime power order. However, the technique contained in this paper splits an algebra into slices on which such nice decomposition can be observed.

To define this slicing properly, we need to consider congruences. Recall that congruence σ of an algebra $\mathbf{A}$ is an equivalence relation which is preserved by the operations of $\mathbf{A}$. Such relations are naturally associated with surjective homomorphisms from $\mathbf{A}$ to $\mathbf{A}/\sigma$ mapping x to $[x]_\sigma$ (equivalence class of x in σ), so congruences are essentially generalization of normal subgroups of a group. Similarly as normal subgroups, congruences of an algebra form a lattice. From now on, we write $\mathsf{Con}\,\mathbf{A}$ for the set of all congruences of $\mathbf{A}$. Every element of a finite lattice can be written as a meet (join) of meet-irreducible (join-irreducible) elements, i.e. elements which cannot be written as a meet (join) of any other two elements of the lattice. These special elements, generating $\mathsf{Con}\,\mathbf{A}$, will play a significant role in our analysis.

For $\alpha, \beta \in \mathsf{Con}\,\mathbf{A}$, by $I[\alpha, \beta]$ we mean a set of congruences γ such that $\alpha \leqslant \gamma \leqslant \beta$. In case when $I[\alpha, \beta] = \{\alpha, \beta\}$, i.e. there are no congruences between α and β, we call β a cover of α, we call α a subcover of β, and we call α, β a covering pair. To highlight such a situation we write $\alpha \prec \beta$ for short. Whenever α is meet-irreducible (join-irreducible) congruence, then there is a unique congruence α^+ (α^-) such that $\alpha \prec \alpha^+$ ($\alpha^- \prec \alpha$). For a nilpotent algebra $\mathbf{A}$ from a congruence modular variety whenever its congruences α, β satisfy $\alpha \prec \beta$, the cosets (congruence classes) of β/α in $\mathbf{A}/\alpha$ have equal sizes, being a power of some prime (this is a consequence of the results contained in the fifth and seventh chapters of [12]). We later denote this prime by $\mathsf{char}(\alpha, \beta)$ and call it a characteristic of a congruence cover $\alpha \prec \beta$. Moreover for arbitrary $\alpha < \beta$, we write $\mathsf{char}\{\alpha, \beta\}$ for the set of all possible prime characteristics of covering pairs, which are fully contained in $I[\alpha, \beta]$. We say that a pair of congruences $\alpha < \beta$ of a nilpotent algebra $\mathbf{A}$ forms a Prime Uniform Product Interval (PUPI), if there are congruences $\alpha < \alpha_1, \ldots, \alpha_k \leqslant \beta$ such that

- $\bigvee \alpha_i = \beta$
- $\alpha_i \wedge (\bigvee_{j \neq i} \alpha_j) = \alpha$, for $i, j \in \{1..k\}$,
- For every i we have $|\mathsf{char}\{\alpha, \alpha_i\}| = 1$.

We call a congruence β supernilpotent whenever it is nilpotent and the interval $I[0_{\mathbf{A}}, \beta]$ is a PUPI, and we call an algebra $\mathbf{A}$ supernilpotent, whenever $1_{\mathbf{A}}$ is supernilpotent. Each supernilpotent algebra is isomorphic to a direct product of nilpotent algebras of prime power size. In this sense supernilpotence generalizes nilpotence for groups. Note that this definition is equivalent to a standard definition of supernilpotent algebras in congruence modular varieties [34].

We say that a nilpotent algebra $\mathbf{A}$ has supernilpotent rank k whenever k is the smallest number for which we can find sequence of congruences $0_{\mathbf{A}} = \alpha_0 < \alpha_1 < \ldots < \alpha_k = 1_{\mathbf{A}}$ such that interval $I[\alpha_i, \alpha_{i+1}]$ is a PUPI for each $0 \leqslant i < k$. Note that for a finite nilpotent algebra

$\mathbf{A}$ we can always find such a finite sequence, since each covering pair forms a PUPI. This notion of supernilpotent rank of an algebra, later denoted by $\mathsf{sr}(\mathbf{A})$, proved to be extremely useful in some very recent results on the computation complexity of circuit satisfiability problem [21, 33]. In fact, results for ProgCSat / CEqv we present in this paper, are essentially about nilpotent algebras with $\mathsf{sr}(A) = 2$.

3 Polynomial equivalence

Our characterization of polynomial time cases of PolEqv is achieved through a reduction to some special instances of PolSat. It was already noticed in [13] that PolSat($\mathbf{G}$) for finite group $\mathbf{G}$ reduces in polynomial time to ProgSat($\mathbf{G}$). Very recently, the full characterization of groups for which ProgSat can be solved in randomized polynomial time was shown under the assumptions of rETH and CDH in [26].

▶ **Theorem 3.1.** *Let $\mathbf{G}$ be a finite group. Assuming rETH and CDH the problem* ProgSat($\mathbf{G}$) *is in RP if and only if $\mathbf{G}/\mathbf{G}_p$ is nilpotent for some normal p-subgroup $\mathbf{G}_p$ of $\mathbf{G}$ (with p being prime).*

Theorem 3.1 together with a result from [25] provides us with enough information to prove the Theorem 1.1.

Proof. We start with recalling that [25, Theorem 1] tells us that under ETH, PolEqv($\mathbf{G}$) $\in \mathsf{P}$ forces $\mathbf{G}$ to have a normal nilpotent subgroup $\mathbf{H}$ with nilpotent quotient $\mathbf{G}/\mathbf{H}$. The very same proof actually gives the same structure of $\mathbf{G}$ under rETH and PolEqv($\mathbf{G}$) $\in \mathsf{coRP}$.

For the converse we start by observing that the nilpotent normal subgroup $\mathbf{H}$ of $\mathbf{G}$ is a product of its Sylow subgroups, say $\mathbf{H}_1, \ldots, \mathbf{H}_s$ with $\mathbf{H}_j$ being a p_j-group. Each such subgroup $\mathbf{H}_j$ is isomorphic to the quotient $\mathbf{H}/\mathbf{H}'_j$ for the normal subgroup $\mathbf{H}'_j$ of $\mathbf{H}$ consisting of all elements in H with the order not divisible by p_j. In particular $H'_1 \cap \ldots \cap H'_s = \{1\}$. Moreover for an inner automorphism h of $\mathbf{G}$ we have not only $h(H) = H$ but also $h(H'_j) = H'_j$, as h has to preserve the order of elements. This means that $\mathbf{H}'_j$ is normal also in $\mathbf{G}$.

Now we know that the quotient $\mathbf{G}/\mathbf{H}'_j$ has a nilpotent normal subgroup $\mathbf{H}/\mathbf{H}'_j$ with a nilpotent quotient $(\mathbf{G}/\mathbf{H}'_j)/(\mathbf{H}/\mathbf{H}'_j) = \mathbf{G}/\mathbf{H}$. Thus CDH and Theorem 3.1 give us that ProgSat$\left(\mathbf{G}/\mathbf{H}'_j\right)$ and consequently PolSat$\left(\mathbf{G}/\mathbf{H}'_j\right)$ are in RP. Consequently PolEqv$\left(\mathbf{G}/\mathbf{H}'_j\right) \in \mathsf{coRP}$ as deciding whether $\mathbf{t} = \mathbf{s}$ holds in $\mathbf{G}/\mathbf{H}'_j$ reduces to check if none of the $\left|G/H'_j\right| - 1$ equations of the form $\mathbf{t}\mathbf{s}^{-1} = a$, with $a \in G/H'_j - \{1\}$ has a solution.

Finally, $H'_1 \cap \ldots \cap H'_s = \{1\}$ tells us that an equation holds in $\mathbf{G}$ iff it holds in all the quotients $\mathbf{G}/\mathbf{H}'_j$, so that PolEqv($\mathbf{G}$) $\in \mathsf{coRP}$. ◀

4 Program satisfiability

The goal of this section is to prove Theorem 1.2. First we observe in Fact 4.1 that if a nilpotent algebra has a Malcev term then CSat and CEqv for this algebra reduce to ProgCSat. Thus, intractability of CSat or CEqv implies intractability of ProgCSat and conversely if ProgCSat is in RP, then CSat is in RP and CEqv is in coRP.

▶ **Fact 4.1.** *For a finite nilpotent Malcev algebra $\mathbf{A}$ the problems* CSat($\mathbf{A}$) *and* CEqv($\mathbf{A}$) *are many-to-one reducible to* ProgCSat($\mathbf{A}$) *and the complement of* ProgCSat($\mathbf{A}$) *respectively.*

Proof. To prove the fact we start with fixing $a_0 \in A$ and enumerating $A - \{a_0\} = \{a_1, \ldots, a_k\}$ to form $A_j = \{a_0, a_j\}$. Using Malcev term $\mathbf{d}$ we define the k-ary polynomial $\mathbf{f}$ by putting

$$\mathbf{f}(x_1, \ldots, x_k) = \mathbf{d}(\ldots \mathbf{d}(\mathbf{d}(x_1, a_0, x_2), a_0, x_3) \ldots, a_0, x_k).$$

Obviously $\mathbf{f}(a_0, \ldots, a_0) = a_0$. To see that the other a_j's are in $\mathbf{f}(A_1, \ldots, A_k)$ simply evaluate x_j by $a_j \in A_j$ and the rest of the x_i's by a_0.

Observe that if $\mathbf{A}$ is a nilpotent algebra with a Malcev term $\mathbf{d}(x, y, z)$ and $e, a, b \in A$ then, by [12, Lemma 7.3] we have $a = b$ iff $\mathbf{d}(a, b, e) = e$. This immediately shows that in nilpotent Malcev algebras only equations of the form $\mathbf{t}(\overline{x}) = e$ (i.e. equations in which one side is a constant polynomial) need to be considered in satisfiability or equivalence problems.

Thus we start with an instance of CSat [or CEqv] $\mathbf{t}(\overline{x}) = e$ and define kn-ary polynomial

$$\mathbf{t}'(x_1^1, \ldots, x_1^k, \ldots, x_n^1, \ldots, x_n^k) = \mathbf{t}(\mathbf{f}(x_1^1, \ldots, x_1^k), \ldots, \mathbf{f}(x_n^1, \ldots, x_n^k)).$$

Due to $A = \mathbf{f}(A_1, \ldots, A_k)$ we easily get that $\mathbf{t} = e$ has a solution in $\mathbf{A}$ [holds identically in $\mathbf{A}$] iff $\mathbf{t}' = \mathbf{e}$ has a solution with x_i^j restricted to be taken from A_j [or holds for all 2^{kn} evaluations of the x_i^j's in A_j, respectively].

We define the reduction which for a given instance of CSat($\mathbf{A}$) [CEqv($\mathbf{A}$)] in the form $\mathbf{t}(\overline{x}) = e$ returns nk-ary Boolean $\mathbf{t}'$-program (over the variables $b_1^1, \ldots, b_1^k, \ldots, b_n^1, \ldots, b_n^k$) with the instructions $\iota(x_i^j) = (b_i^j, a_0, a_j)$. One can easily see that these reductions work, after setting the accepting values to be $S = \{e\}$ in case of CSat($\mathbf{A}$), and $S = A - \{e\}$ for CEqv($\mathbf{A}$). ◀

In the proof of Theorem 1.2 we will use Fact 4.1 to show that, under our assumptions, nilpotent Malcev algebra with tractable ProgCSat has supernilpotent rank equal at most 2. We use advanced tools of Tame Congruence Theory and Commutator Theory to prove the following lemma which shows that not for every algebra with supernilpotent rank equal 2 ProgCSat is tractable. The proof of the lemma can be found in the full version of the paper on arXiv [24].

▶ **Lemma 4.2.** *Let $\mathbf{A}$ be a finite nilpotent algebra from a congruence modular variety with* $\mathsf{sr}(\mathbf{A}) = 2$ *and tractable* ProgCSat($\mathbf{A}$).

Then $\mathbf{A}$ has a supernilpotent congruence α with cosets of prime power order such that quotient algebra $\mathbf{A}/\alpha$ is also supernilpotent, or rETH fails.

The important ingredient of the proof of above lemma is an idea of Barrington et al [4] heavily explored in [22] and [26] resulting in the following lemma.

▶ **Lemma 4.3.** *Let p be a prime number and $\nu \geqslant 1$ be an integer. Then for each 3-CNF formula $\Phi(\overline{x})$ with n variables there is a polynomial $w_p^\Phi(\overline{x})$ over $GF(p)$ of degree at most $O(p^\nu)$ such that for all $\overline{b} \in \{0,1\}^n$ we have*

$$w_p^\Phi(\overline{b}) = \begin{cases} 0, & \textit{if the number of unsatisfied (by } \overline{b}\textit{) clauses in } \Phi \\ & \textit{is divisible by } p^\nu \\ 1, & \textit{otherwise.} \end{cases}$$

Moreover, computing w_p^Φ from Φ can be done in $2^{O(p^\nu(\log n + \log p))}$ steps.

The power of Lemma 4.3 can be observed when we use it simultaneously for two different primes, say p_1 and p_2. Then if for a given 3-CNF formula Φ with m clauses we will choose positive integers ν_1, ν_2 such that $p_i^{\nu_i - 1} \leqslant \sqrt{m} < p_i^{\nu_i}$, we will get, by Chinese Remainder

Theorem, that Φ is satisfied by $\overline{b} \in \{0,1\}$ iff $w^{\Phi}_{p_1}(\overline{b}) = w^{\Phi}_{p_2}(\overline{b}) = 0$. Moreover, the lengths of $w^{\Phi}_{p_1}(\overline{b})$ and $w^{\Phi}_{p_2}(\overline{b})$ are subexponential in the size of Φ. The core of the proof of Lemma 4.2 is showing (with heavilly use of Tame Congruence Theory and Commutator Theory) that if nilpotent Malcev algebra $\mathbf{A}$ with supernilpotent rank 2 has supernilpotent congruence α which cosets are not of prime power size and such that $\mathbf{A}/\alpha$ is supernilpotent then we can simulate by programs over $\mathbf{A}$ systems of equations in the form:

$$w^{\Phi}_{p_1}(\overline{b}) = 0,$$

$$w^{\Phi}_{p_2}(\overline{b}) = 0.$$

Now we are ready to prove Theorem 1.2

Proof of Theorem 1.2: We start with the following observation:

(4.1) ProgCSat$((\{0,1\};\wedge,\vee))$ is NP-complete.

To see that we start with an n-ary CNF-formula $\Phi(b_1,\ldots,b_n)$, treat it as a function $\{0,1\}^n \longrightarrow \{0,1\}$, and convert to n-ary program over the lattice $(\{0,1\};\wedge,\vee)$. First, by introducing the variables $x_1,\ldots,x_n$ and $x'_1,\ldots,x'_n$ we produce a new $2n$-ary formula $\Phi'(x_1,\ldots,x_n,x'_1,\ldots,x'_n)$ by simply replacing each positive literal b_i by x_i and negative literal $\neg b_i$ by x'_i. This leads to a function $\Phi' : \{0,1\}^{2n} \longrightarrow \{0,1\}$ and allows us to transform the formula Φ to the program $(\Phi', n, \iota, \{1\})$ by putting $\iota(x_i) = (b_i, 0, 1)$ and $\iota(x'_i) = (b_i, 1, 0)$.

Using (4.1) we can exclude TCT types **3** and **4** from the typeset typ$\{\mathbf{A}\}$ so that:

(4.2) Either ProgCSat$(\mathbf{A})$ is NP-complete or $\mathbf{A}$ is solvable.

Actually we can force $\mathbf{A}$ to be nilpotent. Indeed, Lemma 2.2 of [23] supplies us with an element $e \in A$ and a partition of A into two nonempty disjoint subsets $A = A_0 \cup A_1$ which allows to associate (in linear time $O(m)$) with a 3-CNF-formula Φ (with m clauses and n variables) a $3m$-ary circuit(polynomial) $\mathbf{sat}_\Phi$ of $\mathbf{A}$ such that for $b^1_1, b^1_2, b^1_3, \ldots, b^m_1, b^m_2, b^m_3 \in \{0,1\}$ and $x^1_1, x^1_2, x^1_3, \ldots, x^m_1, x^m_2, x^m_3 \in A$ with $x^j_i \in A_{b^j_i}$ we have

$$\Phi(b^1_1, b^1_2, b^1_3, \ldots, b^m_1, b^m_2, b^m_3) = 1 \quad \text{iff} \quad \mathbf{sat}_\Phi(x^1_1, x^1_2, x^1_3, \ldots, x^m_1, x^m_2, x^m_3) = e.$$

Thus fixing $a_0 \in A_0$ and $a_1 \in A_1$ we end up with a program $(\mathbf{sat}_\Phi, 3m, \iota, \{e\})$ over $\mathbf{A}$, where $\iota(x^j_i) = (b^j_i, a_0, a_1)$. This reduction from 3-CNF-SAT to ProgCSat$(\mathbf{A})$ shows that:

(4.3) Either ProgCSat$(\mathbf{A})$ is NP-complete or $\mathbf{A}$ is nilpotent.

To enforce that tractability of ProgCSat$(\mathbf{A})$ enforces $\mathbf{A}$ to have supernilpotent rank 2 we refer to [21], where $\mathsf{sr}(\mathbf{A}) \geqslant 3$ gives a chain $p_1 \neq p_2 \neq p_3$ of primes occurring as characteristics in consecutive PUPI in Con $\mathbf{A}$. Moreover [21] shows how to use one alternation of characteristics to represent n-ary AND by a polynomial/circuit of size $2^{O(n)}$, and further how to use two alternations of characteristics to compose such created $\sqrt{n}$-ary AND functions to represent n-ary AND by a polynomial/circuit of size $2^{O(\sqrt{n})}$. From this one expects that under (r)ETH CSat$(\mathbf{A})$ is not in (R)P if $\mathsf{sr}(\mathbf{A}) \geqslant 3$. In fact [21] provides examples of such algebras, while [33] contains a nice proof of this expectation. This, together with Fact 4.1, gives us that:

(4.4) If ProgCSat$(\mathbf{A}) \in$ (R)P then $\mathsf{sr}(\mathbf{A}) \leqslant 2$, or (r)ETH fails.

Now, the immediate consequence of (4.4) and Lemma 4.2 is that:

(4.5) If ProgCSat$(\mathbf{A}) \in$ (R)P then there exists supernilpotent congruence α of $\mathbf{A}$ with cosets of prime power size and such that $\mathbf{A}/\alpha$ is supernilpotent, or (r)ETH fails.

Finally, let $\mathbf{A}$ be a nilpotent algebra of supernilpotent rank 2 having supernilpotent congruence α of $\mathbf{A}$ with costes of prime power size and such that $\mathbf{A}/\alpha$ is supernilpotent. In such the case programs over $\mathbf{A}$ computing n-ary AND functions have size at least $2^{\Omega(n)}$, or (r)ETH and CDH fails. This is an immediate consequence of Theorem 1.4, for if not, the AND_n function computed by a program of size $\ell(n)$ can be also computed by a $\mathsf{AND}_d \circ \mathsf{MOD}_m \circ \mathsf{MOD}_p$-circuit of size $O(\ell(n)^c)$ for the constants c, d, m, p depending only of the algebra $\mathbf{A}$. But then CDH tells us that $O(\ell(n)^c)$, and therefore $\ell(n)$ itself, has to dominate $2^{\Omega(n)}$. Now, to prove that $\textsc{ProgCSat}(\mathbf{A})$ is in RP recall that the second part of [26, Proposition 9] tells that a set of binary words accepted by a program $(\mathbf{p}, n, \iota, S)$ of size ℓ is either empty or has the size at least $2^n/\ell^c$, for some constant c. Thus checking at least ℓ^c random Boolean words of length n finds a word accepted by the program (if there is one at all) with probability at least 1/2. This obviously puts $\textsc{ProgCSat}(\mathbf{A})$ into RP. ◀

5 Circuit equivalence

In this section we will prove Theorem 1.3. To do it we will show that solving $\textsc{CEqv}$ for an algebra $\mathbf{A}$ can be reduced to solving the very same problem for quotients of $\mathbf{A}$ by a meet-irreducible congruences. Obviously, every quotient algebra by a meet-irreducible congruence has the smallest congruence bigger than the identity relation. This observation plays a crucial role in the proof of Theorem 1.3.

Proof of Theorem 1.3. First suppose that $\textsc{CEqv}(\mathbf{A})$ is tractable to eliminate types $\mathbf{3}$ and $\mathbf{4}$ from the $\text{typ}\{\mathbf{A}\}$. It should be obvious how to do it for type $\mathbf{3}$. For the lattice type $\mathbf{4}$ we first note that the existence of a solution (in the two element lattice) to the systems of two equations of the form $\mathbf{m}(\overline{x}) = 1 \;\&\; \mathbf{j}(\overline{x}) = 0$ is NP-complete, where $\mathbf{m}(\overline{x})$ is in CNF and $\mathbf{j}(\overline{x})$ is in DNF, both with only positive literals. But obviously this system has no solutions iff $\mathbf{m}(\overline{x}) \vee \mathbf{j}(\overline{x}) = \mathbf{j}(\overline{x})$ holds identically in $(\{0,1\}; \wedge, \vee)$. To put this into the minimal set U of type $\mathbf{4}$ simply replace each variable x by $\mathbf{e}_U(x)$ (where $\mathbf{e}_U$ is an idempotent polynomial of $\mathbf{A}$ with the range U) and the operations $\wedge, \vee$ by the corresponding polynomials of $\mathbf{A}$ turning $\mathbf{A}|_U$ into the two element lattice. This shows that $\text{typ}\{\mathbf{A}\} \subseteq \{\mathbf{2}\}$ (i.e. $\mathbf{A}$ is solvable) or $\textsc{CEqv}(\mathbf{A})$ is co-NP-complete.

Now the discussion made in [23, Section 4] between Problems 2 and 3 shows that (under (r)ETH) in fact $\mathbf{A}$ has to be nilpotent and that $\mathsf{sr}\,(\mathbf{A}) \leqslant 2$. This shows the “only if” direction of our theorem.

To prove the converse note that an identity holds in an algebra $\mathbf{A}$ iff it holds in all quotients of $\mathbf{A}$ by meet-irreducible congruences (as the intersection of all meet-irreducible congruences is the identity relation $\mathbf{0_A}$). Let α be a meet-irreducible congruence of $\mathbf{A}$. To complete the proof it suffices to show that (under CDH) $\textsc{CEqv}(\mathbf{A}/\alpha) \in \mathsf{coRP}$.

Observe that if $\mathbf{A}$ is nilpotent and $\mathsf{sr}\,(\mathbf{A}) \leqslant 2$ then each quotient of $\mathbf{A}$ has these two properties as well. Moreover, identity relation in $\mathbf{A}/\alpha$ (i.e. $0_{\mathbf{A}/\alpha}$) has the unique cover. Since congruence classes for covering pairs have equal sizes [12, Colloraly 7.5], it is clear from definition of supernilpotency that every supernilpotent congruences of $\mathbf{A}/\alpha$ has cosets of prime power sizes (as for every $\beta_1, \beta_2 \in \mathsf{Con}\,\mathbf{A}/\alpha$, if $\beta_1, \beta_2 > 0_{\mathbf{A}/\alpha}$, then $\beta_1 \wedge \beta_2 > 0_{\mathbf{A}/\alpha}$). Therefore by Theorem 1.2, $\textsc{ProgCSat}(\mathbf{A}/\alpha) \in \mathsf{RP}$. Finally Fact 4.1 gives us $\textsc{CEqv}(\mathbf{A}) \in \mathsf{coRP}$, as required. ◀

6 Final Remarks

The Constant Degree Hypothesis plays a crucial role in our proofs of the existence of randomized polynomial-time algorithms. One can ask if this assumption is really needed. In fact, there are some unconditional results. For example very recent paper [29] shows that CEqv($\mathbf{A}$) is in P whenever $\mathbf{A}$ from a congruence modular variety is 2-nilpotent, i.e. it has abelian congruence with abelian quotient. Also supernilpotent algebras admit (unconditional) polynomial-time algorithm for CSat/CEqv [1, 31, 27], and if we allow random bits the time complexity drops down to linear [28]. Unfortunately, it is not hard to construct for a given d, m, p an algebra $\mathbf{A}$ with supernilpotent rank equal 2 such that functions computable by $\mathsf{AND}_d \circ \mathsf{MOD}_m \circ \mathsf{MOD}_p$-circuit are exactly functions computable by, not too long, programs over $\mathbf{A}$. This, together with our results, shows that (under ETH) showing unconditional algorithms solving ProgCSat for algebras from congruence modular variety with supernilpotent rank equal 2 is equivalent to proving CDH. Hence, the natural question is if CDH holds.

▶ **Problem 1.** *Prove or disprove the Constant Degree Hyphothesis.*

The natural next step in our investigations is to go outside of the congruence modular realm. The first problem in such a case is that Tame Congruence Theory and Commutator Theory (our heavily used tools) for arbitrary algebras do not work as well as for algebras from congruence modular varieties. Moreover [27, Example 2.8] shows that there is an algebra $\mathbf{A}$ (not contained in congruence modular variety) and its congruence σ such that CSat($\mathbf{A}$) is in P, while CSat($\mathbf{A}/\sigma$) is NP-complete. This suggests that we cannot expect a nice characterization of polynomial-time cases for CSat outside congruence modular setting. On the other hand, we are not aware of any such examples for ProgCSat and CEqv. In fact we saw that the hardness of ProgCSat($\mathbf{A}/\sigma$) implies the hardness for ProgCSat($\mathbf{A}$). This gives hope for characterization of tractable cases of ProgCSat for general finite algebras.

▶ **Problem 2.** *Characterize finite algebras with tractable* ProgCSat/CEqv.

References

1 Erhard Aichinger and Nebojša Mudrinski. Some applications of higher commutators in Mal'cev algebras. *Algebra universalis*, 63(4):367–403, 2010. doi:10.1007/s00012-010-0084-1.

2 David A. Mix Barrington. Width-3 permutation branching programs. Technical Report TM-293, MIT Laboratory for Computer Science, 1985.

3 David A. Mix Barrington. Bounded-Width Polynomial-Size Branching Programs Recognize Exactly Those Languages in NC^1. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing (STOC'86)*, pages 1–5, 1986. doi:10.1145/12130.12131.

4 David A. Mix Barrington, Richard Beigel, and Steven Rudich. Representing Boolean Functions as Polynomials Modulo Composite Numbers. *Computational Complexity*, 4:367–382, 1994. doi:10.1007/BF01263424.

5 David A. Mix Barrington, Pierre McKenzie, Cristopher Moore, Pascal Tesson, and Denis Thérien. Equation Satisfiability and Program Satisfiability for Finite Monoids. In *Proceedings of the 25th International Symposium on Mathematical Foundations of Computer Science (MFCS'2000)*, pages 172–181, 2000. doi:10.1007/3-540-44612-5_13.

6 David A. Mix Barrington, Howard Straubing, and Denis Thérien. Non-Uniform Automata Over Groups. *Information and Computation*, 89(2):109–132, 1990. doi:10.1016/0890-5401(90)90007-5.

7 Andrei Bulatov. On the number of finite Mal'tsev algebras. *Contributions to general algebra*, 13:41–54, 2000.

8 Andrei A. Bulatov. A Dichotomy Theorem for Nonuniform CSPs. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS'17)*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.

9 Stanley Burris and John Lawrence. Results on the equivalence problem for finite groups. *Algebra Universalis*, 52(4):495–500, 2005. doi:10.1007/s00012-004-1895-8.

10 Arkadev Chattopadhyay, Navin Goyal, Pavel Pudlak, and Denis Thérien. Lower bounds for circuits with MOD_m gates. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 709–718, 2006. doi:10.1109/FOCS.2006.46.

11 Attila Földvári and Gábor Horváth. The complexity of the equation solvability and equivalence problems over finite groups. *International Journal of Algebra and Computation*, 30(03):607–623, 2020. doi:10.1142/S0218196720500137.

12 Ralph Freese and Ralph McKenzie. *Commutator Theory for Congruence Modular Varieties*. London Mathematical Society Lecture Notes, No. 125. Cambridge University Press, 1987.

13 Mikael Goldmann and Alexander Russell. The complexity of solving equations over finite groups. *Information and Computation*, 178(1):253–262, 2002. doi:10.1006/inco.2002.3173.

14 Vince Grolmusz. A degree-decreasing lemma for (MOD_p-MOD_m) circuits. *Discrete Mathematics & Theoretical Computer Science*, 4(2):247–254, 2001. doi:10.46298/dmtcs.289.

15 Vince Grolmusz and Gábor Tardos. Lower bounds for (MOD_p-MOD_m) circuits. *SIAM Journal on Computing*, 29(4):1209–1222, 2000. doi:10.1137/S0097539798340850.

16 David Hobby and Ralph McKenzie. *Structure of Finite Algebras*. Contemporary Mathematics vol. 76. American Mathematical Society, 1988. doi:10.1090/conm/076.

17 Gábor Horváth. The complexity of the equivalence and equation solvability problems over nilpotent rings and groups. *Algebra Universalis*, 66(4):391–403, 2011. doi:10.1007/s00012-011-0163-y.

18 Gábor Horváth. The complexity of the equivalence and equation solvability problems over meta-Abelian groups. *Journal of Algebra*, 433:208–230, 2015. doi:10.1016/j.jalgebra.2015.03.015.

19 Gábor Horváth and Csaba Szabó. Equivalence and equation solvability problems for the alternating group $\mathbf{A}_4$. *Journal of Pure and Applied Algebra*, 216(10):2170–2176, 2012. doi:10.1016/j.jpaa.2012.02.007.

20 Gábor Horváth and Csaba A. Szabó. The complexity of checking identities over finite groups. *International Journal of Algebra and Computation*, 16(5):931–940, 2006. doi:10.1142/S0218196706003256.

21 Paweł M. Idziak, Piotr Kawałek, and Jacek Krzaczkowski. Intermediate Problems in Modular Circuits Satisfiability. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'20)*, pages 578–590, 2020. doi:10.1145/3373718.3394780.

22 Pawel M. Idziak, Piotr Kawałek, and Jacek Krzaczkowski. Complexity of Modular Circuits. In *Proceedings of the 37th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'22)*, pages 32:1–32:11, 2022. doi:10.1145/3531130.3533350.

23 Pawel M. Idziak, Piotr Kawałek, and Jacek Krzaczkowski. Satisfiability of circuits and equations over finite Malcev algebras. In *Proceedings of the 39th International Symposium on Theoretical Aspects of Computer Science (STACS'22)*, pages 37:1–37:14, 2022. doi:10.4230/LIPIcs.STACS.2022.37.

24 Paweł M Idziak, Piotr Kawałek, and Jacek Krzaczkowski. Nonuniform deterministic finite automata over finite algebraic structures. *arXiv eprint*, 2025. arXiv:2501.12260. doi:10.48550/arXiv.2501.12260.

25 Pawel M. Idziak, Piotr Kawałek, Jacek Krzaczkowski, and Armin Weiß. Equation satisfiability in solvable groups. *Theory of Computing Systems*, 68:740–757, 2022.

26 Paweł M. Idziak, Piotr Kawałek, Jacek Krzaczkowski, and Armin Weiß. Satisfiability Problems for Finite Groups. In *Proceedings of the 49th International Colloquium on Automata, Languages, and Programming (ICALP'22)*, volume 229, pages 127:1–127:20, 2022. doi:10.4230/LIPIcs.ICALP.2022.127.

27 Paweł M. Idziak and Jacek Krzaczkowski. Satisfiability in MultiValued Circuits. *SIAM Journal on Computing*, 51(3):337–378, 2022. doi:10.1137/18M1220194.

28 Piotr Kawałek and Jacek Krzaczkowski. Even Faster Algorithms for CSAT Over supernilpotent Algebras. In *Proceedings of the 45th International Symposium on Mathematical Foundations of Computer Science (MFCS'20)*, volume 170, pages 55:1–55:13, 2020. doi:10.4230/LIPIcs.MFCS.2020.55.

29 Piotr Kawałek, Michael Kompatscher, and Jacek Krzaczkowski. Circuit equivalence in 2-nilpotent algebras. In *Proceedings of the 41st International Symposium on Theoretical Aspects of Computer Science (STACS'24)*, volume 289, pages 45:1–45:17, 2024. doi:10.4230/LIPIcs.STACS.2024.45.

30 Piotr Kawałek and Armin Weiß. Violating Constant Degree Hypothesis Requires Breaking Symmetry. In *Proceedings of the 42nd International Symposium on Theoretical Aspects of Computer Science (STACS'25)*, volume 327, pages 58:1–58:21, 2025. doi:10.4230/LIPIcs.STACS.2025.58.

31 Michael Kompatscher. The equation solvability problem over supernilpotent algebras with Mal'cev term. *International Journal of Algebra and Computation*, 28:1005–1015, 2017. doi:10.1142/S0218196718500443.

32 Michael Kompatscher. CC-circuits and the expressive power of nilpotent algebras. *Logical Methods in Computer Science*, 18(2):12:1–12:15, 2022. doi:10.46298/lmcs-18(2:12)2022.

33 Michael Kompatscher. CSAT and CEQV for nilpotent Maltsev algebras of Fitting length > 2. *arXiv eprint*, 2023. arXiv:2105.00689.

34 Peter Mayr and Ágnes Szendrei. Algebras from congruences. *Algebra universalis*, 82(55), 2021. doi:10.1007/s00012-021-00740-7.

35 Armin Weiß. Hardness of equations over finite solvable groups under the exponential time hypothesis. In *Proceedings of the 47th International Colloquium on Automata, Languages, and Programming (ICALP'20)*, pages 102:1–102:19, 2020. doi:10.4230/LIPIcs.ICALP.2020.102.

36 Dmitriy Zhuk. A Proof of the CSP Dichotomy Conjecture. *Journal of the ACM*, 67(5), 2020. doi:10.1145/3402029.