Satisfiability of Multivalued Circuits with Lists

Paweł M. Idziak
Jagiellonian University, Kraków, Poland

Piotr Kawałek
TU Wien, Austria

Jacek Krzaczkowski
University of Maria Curie-Skłodowska, Lublin, Poland

Armin Weiß
FMI, Universität Stuttgart, Germany

Abstract

We study the problem ListCSat of satisfiability of multivalued circuits, whose input values are restricted by lists. We obtain a clear description of quasipolynomial time cases of the problem, where the time complexity is determined by the set of gates which are allowed to build such circuits. That is, algebraic structures induced by gates which allow for a quasipolynomial time algorithm decompose into a direct product $\mathbf{L} \times \mathbf{N}$, where the polynomial clone of $\mathbf{L}$ is isomorphic to the clone of two element lattice and polynomial clone of $\mathbf{N}$ is isomorphic to the clone of some finite nilpotent Malcev algebra. If such a decomposition does not exist the ListCSat problem is NP-complete. The result applies to a general setting of finite algebras from congruence modular varieties. In our tractability proofs we assume certain lower bounds for the Boolean CC-circuits. We discuss the cases in which such an assumption can be avoided and polynomial/quasipolynomial time algorithms exist unconditionally.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Theory of computation → Design and analysis of algorithms; Theory of computation → Complexity classes

Keywords and phrases satisifiability, circuit satisfiability, solving equations, lists

Digital Object Identifier 10.4230/LIPIcs.MFCS.2026.80

Funding *Piotr Kawałek*: 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.

1 Introduction

Solving equations is one of the oldest problems in mathematics and, among other things, contributed to the emergence of algebra as a branch of mathematics. This problem is also interesting from the point of view of computer science. There are numerous results concerning the computational complexity of solving equations over a fixed structure. A somewhat unexpected phenomenon is that the computational complexity of solving equations for algebras with the same term or polynomial clone, i.e., algebras in which the same functions can be expressed, may differ. For example, the polynomial satisfiability problem for the alternating group $\mathbf{A}_4$ can be solved in polynomial time, whereas the same problem for the group $\mathbf{A}_4$ enriched with the commutator operation is NP-complete [10]. This phenomenon is

due to the exponential growth of the polynomial size when translating from one language to another. Since computational complexity is measured as a function of the input size, blowing up the size of the input may result in a smaller computational complexity of the problem. Hence, considering the polynomial satisfiability problem (PolSat) makes sense mainly for structures with a fixed language, such as groups, rings, lattices, etc. There are a number of such results, see for example [6], [23], [11], [9], and [4].

On the other hand, Idziak et al. proposed considering equations given by circuits over algebraic structures instead of polynomials, and defined the circuit satisfiability problem CSat($\mathbf{A}$), in which we ask whether a given equation of circuits over $\mathbf{A}$ has a solution [17]. One of the main advantages of this problem is that its complexity depends only on the polynomial clone of $\mathbf{A}$. Thus the unwanted phenomenon known from PolSat does not occur for CSat. The paper [17] initiated a series of works studying CSat, see e.g. [13], [18], [14], and [22]. However, we are still far from a characterization of the computational complexity of CSat. One of the difficulties is that the computational complexity of CSat for a quotient algebra can be greater than for the original algebra. It is somewhat counterintuitive that the problem for a simpler algebra can be harder, and this makes a precise characterization of algebras for which CSat is easy more difficult.

ListCSat is a natural generalization of CSat in which an instance consists of an equation together with lists of allowed values for the variables (a separate list for each variable). Note that since there is no bound on the size of the lists, hardness results obtained for CSat also apply to ListCSat. On the other hand, algorithms for CSat often do not extend to ListCSat. We consider algebras from congruence-modular varieties, a large class of algebras containing most well-known algebraic structures such as groups, rings, and lattices, with the notable exception of semigroups. Note that even for this class we do not have a characterization of the computational complexity of CSat. In contrast to this, we obtain the following conditional dichotomy.

▶ **Theorem 1.1.** *Under the assumption of ESH,* ListCSat *for a finite algebra from a congruence-modular variety is either* NP*-complete, or there exists a quasipolynomial-time algorithm solving* ListCSat*.*

ESH is a circuit complexity hypothesis, stated precisely in Conjecture 5.2, concerning the size of circuits computing AND built from MOD_m gates. Theorem 1.1 relies on Lemma 4.1 and Theorem 6.3.

Modular commutator theory [5] introduces generalizations of the well-known group-theoretic notion of the commutator, which, among other things, is used to define nilpotency in a general setting. This notion allows us to describe in a simple way the borderline in the dichotomy stated in Theorem 1.1. It turns out that ListCSat($\mathbf{A}$) for an algebra from a congruence-modular variety is NP-complete if $\mathbf{A}$ is not a homomorphic image of a direct product of an algebra polynomially equivalent to the two-element lattice and a nilpotent algebra. Otherwise, under the assumption of ESH, there exists a quasipolynomial-time algorithm solving ListCSat.

In cases in which we are able to prove ESH or a stronger version of it, our results can be stated unconditionally. By Corollary 6.5 we obtain that ListCSat for algebras that are direct products of an algebra polynomially equivalent to the two-element lattice and a 2-nilpotent algebra can be solved in quasipolynomial time. Supernilpotency is another generalization of group-theoretic nilpotency. Corollary 6.4 states that ListCSat for direct products of an algebra polynomially equivalent to the two-element lattice and a supernilpotent algebra can be solved in deterministic polynomial time.

2 Background materials

We use the standard notation of universal algebra as in, e.g., [3]. An algebra is a set, called the universe, together with a set of operations on this universe, called the basic operations of the algebra. We consider only finite algebras, i.e., algebras with a finite universe and a finite set of basic operations. We use boldface capital letters to denote algebras and the corresponding regular letters to denote their universes. Polynomial operations of an algebra are operations obtained by composing the basic operations of the algebra, possibly using constants. An algebra $\mathbf{A}$ is polynomially equivalent to an algebra $\mathbf{B}$ if it is isomorphic to an algebra with the same universe and the same set of polynomial operations as $\mathbf{B}$. For an algebra $\mathbf{A}$, the algebra induced on a subset $B \subseteq A$ (denoted $\mathbf{A}|_B$) is the algebra with universe B whose operations consist of all polynomial operations of $\mathbf{A}$ that map tuples from B to elements of B.

Congruences of an algebra $\mathbf{A}$ are equivalence relations preserved by all operations of the algebra; equivalently, they are equivalence relations that form subalgebras of $\mathbf{A} \times \mathbf{A}$. The set of congruences of $\mathbf{A}$, denoted $\mathsf{Con}\,\mathbf{A}$, ordered by inclusion, forms a lattice. For $\alpha, \beta \in \mathsf{Con}\,\mathbf{A}$ we write $\alpha \prec \beta$ to denote that β covers α, i.e., $\alpha < \beta$ and there is no $\gamma \in \mathsf{Con}\,\mathbf{A}$ such that $\alpha < \gamma < \beta$. For an algebra $\mathbf{A}$, $1_{\mathbf{A}}$ and $0_{\mathbf{A}}$ denote the largest (the universal relation) and the smallest (the equality relation) congruence of $\mathbf{A}$, respectively. Join-irreducible elements of the congruence lattice are called join-irreducible congruences. It is well known that for every join-irreducible congruence α there exists a unique congruence α^- such that $\alpha^- \prec \alpha$. For $a \in A$ and $\alpha \in \mathsf{Con}\,\mathbf{A}$ we write a/α to denote the α-coset (equivalence class) containing a. The notation $\Theta(a, b)$ denotes the smallest congruence in which a and b belong to the same coset.

We will use Tame Congruence Theory (TCT), an advanced theory describing the local behavior of finite algebras (see [8]). The central notion of TCT is the notion of a minimal set. For congruences $\alpha \prec \beta$ of $\mathbf{A}$, an (α, β)-minimal set is a minimal (with respect to inclusion) range of a unary polynomial operation of $\mathbf{A}$ that does not collapse β to α (that is, a polynomial $\mathbf{f}$ such that $\mathbf{f}(\beta) \not\subseteq \alpha$). The set of all (α, β)-minimal sets is denoted by $M_{\mathbf{A}}(\alpha, \beta)$. It is well known that for every minimal set U there exists a unary polynomial whose range is exactly U and which, when restricted to U, acts as the identity. We denote such a polynomial by $\mathbf{e}_U$. For an (α, β)-minimal set U and $a \in U$, the set $a/\beta \cap U$ is called a trace of U if $a/\beta \cap U \neq a/\alpha \cap U$. Minimal sets of algebras from congruence modular varieties are unions of their traces. It turns out that the algebras induced on traces of (α, β)-minimal sets are, modulo α, polynomially equivalent to one of the following types of algebras:

1. a finite set with a (unary) 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.

Algebras induced on traces of the same minimal set are polynomially equivalent. Hence it makes sense to speak about the above types **1**–**5** as types of minimal sets. Moreover, every (α, β)-minimal set for fixed congruences α, β has the same type, which we call the type of the quotient. For every pair of minimal sets corresponding to the same quotient there exist polynomials of the algebra that are bijections between these minimal sets (in both directions). We use the notation $\alpha \prec_{\mathbf{i}} \beta$ to denote that $\mathbf{i}$ is the type of the quotient $\alpha \prec \beta$. Every minimal set is a minimal set for some quotient $\alpha^- \prec \alpha$, where α is join-irreducible. Algebras from congruence modular varieties admit only types **2**, **3**, and **4**, and in the cases of types **3** and **4** the minimal sets are two-element sets and therefore consist of a single trace.

The second important theory of universal algebra used in this paper is Modular Commutator Theory (see [5]). This theory provides a generalized commutator defined as a binary operation on congruences. Using this commutator one can define nilpotency in a general setting. Every nilpotent algebra from a congruence modular variety has Malcev term, i.e., a term operation $\mathbf{d}$ such that $\mathbf{d}(y,x,x) = \mathbf{d}(x,x,y) = y$. Moreover, in the case of a nilpotent algebra, its Malcev term has the additional property that $\mathbf{d}(x,a,b)$ and $\mathbf{d}(a,b,x)$ are permutations for every choice of constants a and b. Another generalization of group nilpotency is supernilpotency (see, e.g., [1]). It is known that every finite supernilpotent algebra from a congruence modular variety is nilpotent.

3 Restricting lattice behavior by lists evaluation

We will use tools presented in Section 8 of [17] in which computational complexity of CSat for algebras from congruence modular varieties admitting only TCT type **4** is considered. First, after [17] we observe that some configurations lead to NP-completeness of considered problems. The original result is stated for CSat but it obviously holds also for ListCSat.

▶ **Lemma 3.1** ([17, Lemma 8.1]). *Let $\mathbf{A}$ be a finite algebra from a congruence modular variety such that* $\mathrm{typ}\{\mathbf{A}\} \subseteq \{\mathbf{4}\}$. *If one of the following configurations*

1. $\{0,1\}$ *is a minimal set and there are three different elements* $a,b,c \in A$ *and* $\mathbf{f}_a, \mathbf{f}_c \in \mathrm{Pol}_1\mathbf{A}$ *such that* $\mathbf{f}_a\binom{1}{0} = \binom{b}{a}$ *and* $\mathbf{f}_c\binom{1}{0} = \binom{c}{b}$,
2. $\{0,1\}$ *is the range of a polynomial* $\mathbf{p} \in \mathrm{Pol}_1\mathbf{A}$, $\{a,b\}$ *is a minimal set and there are polynomials* $\mathbf{f},\mathbf{g} \in \mathrm{Pol}_1\mathbf{A}$ *such that* $\mathbf{f}\binom{1}{0} = \binom{b}{a}$ *and* $\mathbf{g}\binom{1}{0} = \binom{a}{b}$,
3. $\{0,1\}$ *is a minimal set and one of the sets*

$$F_1 = \bigcap\{\mathbf{f}^{-1}(1) \;:\; \mathbf{f} \in \mathrm{Pol}_1\mathbf{A} \text{ and } \mathbf{f}(A) = \{0,1\}\},$$
$$F_0 = \bigcap\{\mathbf{f}^{-1}(0) \;:\; \mathbf{f} \in \mathrm{Pol}_1\mathbf{A} \text{ and } \mathbf{f}(A) = \{0,1\}\},$$

is empty

can be found in $\mathbf{A}$ *then* $\mathrm{CSat}(\mathbf{A})$ ($\mathrm{ListCSat}(\mathbf{A})$) *is* NP*-complete.*

Let α be a join irreducible congruence of $\mathbf{A}$ and $\{0,1\}$ a (α^-,α)-minimal set. Then for $a,b \in A$ we define

$a \leqslant_\alpha b$ iff there is a polynomial $\mathbf{f} \in \mathrm{Pol}_1\mathbf{A}$ with $\mathbf{f}\binom{1}{0} = \binom{b}{a}$.

It can be observed that $\leqslant_\alpha$ doesn't depend on choice of the (α^-,α)-minimal set. For simplicity, we will use the following notation:

- $a <_\alpha b$ iff $a \leqslant_\alpha b$ and $a \neq b$,
- $a <>_\alpha b$ iff $a <_\alpha b$ or $b <_\alpha a$.

The following basic facts about $\leqslant_\alpha$ were shown in [17]:

▶ **Lemma 3.2** ([17, Lemma 8.2]). *Let $\mathbf{A}$ be a finite algebra from a congruence modular variety with* $\mathrm{typ}\{\mathbf{A}\} \subseteq \{\mathbf{4}\}$ *and let α be a join irreducible congruence of $\mathbf{A}$. Then*

1. $a <>_\alpha b$, *whenever* $\{a,b\} \in M_{\mathbf{A}}(\delta,\delta')$ *for some* $\delta \prec \delta' \leqslant \alpha$,
2. *for every* $a \in A$ *the graph* $(a/\alpha, <>_\alpha)$ *is connected,*
3. *all unary polynomials of $\mathbf{A}$ preserve the relation* $\leqslant_\alpha$, *and all polynomials of $\mathbf{A}$ preserve the transitive closure of* $\leqslant_\alpha$.

Now, excluding configurations listed in Lemma 3.1 it can be proved (see [17]) that $\leqslant_\alpha$ fulfills even stronger conditions.

▶ **Lemma 3.3** ([17, Lemma 8.3]). *Let* $\mathbf{A}$ *be a finite algebra from a congruence modular variety,* α *be a join irreducible congruence of* $\mathbf{A}$ *and* $\mathrm{typ}\{\mathbf{A}\} \subseteq \{\mathbf{4}\}$*. Then either* $\mathrm{CSat}(\mathbf{A})$ ($\mathrm{ListCSat}(\mathbf{A})$) *is* NP*-complete or all the following hold:*

1. $\leqslant_\alpha$ *is a partial order on* A *without* 3*-element chains,*
2. $\leqslant_\alpha$ *is preserved by all polynomials of* $\mathbf{A}$*,*
3. *for every* $a \in A$ *the graph* $(a/\alpha, <>_\alpha)$ *is connected and acyclic,*
4. $a <>_\alpha b$ *if and only if* $\{a, b\} \in M_{\mathbf{A}}(\delta, \delta')$ *for some* $\delta \prec \delta' \leqslant \alpha$*.*

By [20] an algebra generates (is included in) a congruence modular variety iff it has so called directed Gumm terms, i.e. terms $\mathbf{D}_1(x,y,z), \ldots, \mathbf{D}_n(x,y,z), \mathbf{Q}(x,y,z)$ satisfying the following equalities:

$$\begin{array}{lcll} x & = & \mathbf{D}_i(x,y,x), & \text{for all } i = 1,\ldots,n, \\ x & = & \mathbf{D}_1(x,x,y), & \\ \mathbf{D}_i(x,y,y) & = & \mathbf{D}_{i+1}(x,x,y), & \text{for all } i = 1,\ldots,n-1, \\ \mathbf{D}_n(x,y,y) & = & \mathbf{Q}(x,y,y), & \\ & & \mathbf{Q}(x,x,y) = y. & \end{array}$$

Lemma 3.3 and the existence of the Gumm terms will be heavily used in the proof of main result of this section. Moreover, we will need the following Lemma that helps forcing NP-completeness when allowed to use lists.

▶ **Lemma 3.4.** *Let* $\mathbf{A}$ *be a finite algebra from a congruence modular variety such that* $\mathrm{typ}\{\mathbf{A}\} \subseteq \{\mathbf{4}\}$ *and sets* $\{a, b\}$ *as well as* $\{b, c\}$ *are minimal. If* $\mathbf{A}$ *has two unary polynomials* $\mathbf{f}_a, \mathbf{f}_c$ *such that* $\mathbf{f}_a\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} a \\ b \\ b \end{pmatrix}$ *and* $\mathbf{f}_c\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} b \\ b \\ c \end{pmatrix}$*, then* $\mathrm{ListCSat}(\mathbf{A})$ *is* NP*-complete.*

Proof. The minimal sets $\{a, b\}$ and $\{b, c\}$ are of type **4** so that they carry the lattice structure and, without loss of generality, we may assume that $a < b$ and $b > c$ inside these minimal sets.

As in the proof of [17, Lemma 8.1], we transform the system of two lattice equations:

$$\begin{cases} \bigwedge_{i=1}^m x_1^i \vee x_2^i \vee x_3^i = 1, \\ \bigvee_{i=1}^n y_1^i \wedge y_2^i \wedge y_3^i = 0 \end{cases}$$

into a single equation of the form

$$\bigvee_{i=1}^m \mathbf{f}_a(x_1^i) \wedge \mathbf{f}_a(x_2^i) \wedge \mathbf{f}_a(x_3^i) \quad = \quad \bigvee_{i=1}^n \mathbf{f}_c(y_1^i) \wedge \mathbf{f}_c(y_2^i) \wedge \mathbf{f}_c(y_3^i)$$

of the algebra $\mathbf{A}$, where meets and joins are performed in the minimal set $\{a < b\}$ on the left hand side of this equation, and in the minimal set $\{c < b\}$ on the right hand side of this equation. To make this work we restrict all the variables in the above equation to be evaluated in the list $\{a, c\}$. It should be clear that the only way to satisfy this equation is to make both sides equal to b. Moreover this equation is satisfiable iff the two lattice equations are. ◀

Now, we are ready to prove the main result of this section.

▶ **Theorem 3.5.** *Let* $\mathbf{A}$ *be a finite algebra from a congruence modular variety such that* $\mathrm{typ}\{\mathbf{A}\} \subseteq \{\mathbf{4}\}$*. If* $|A| > 2$ *then* $\mathrm{ListCSat}(\mathbf{A})$ *is* NP*-complete.*

Proof. Suppose that $|\mathbf{A}| \geqslant 3$ and that ListCSat($\mathbf{A}$) is not NP-complete. First we show the following property:

(3.1) Each coset of a join irreducible congruence of $\mathbf{A}$ has at most 2 elements.

Suppose that a join irreducible congruence α has a coset with at least 3 elements and note that by Items 1 and 3 of Lemma 3.3 this can be witnessed by some elements a, b, c satisfying either $a <_\alpha b >_\alpha c$ or $a >_\alpha b <_\alpha c$. We will focus on the first case, as in the second one the argument is the same (just by reversing the inequalities). Moreover Item 4 of Lemma 3.3 tells us that the sets $\{a, b\}$ and $\{b, c\}$ are minimal (with respect to some prime quotients below α).

We start with showing that

(3.2) $\mathbf{D}_i(a, b, c) \neq b$, for $i = 1, \ldots, n$.

Indeed, if for some i we would have $\mathbf{D}_i(a, b, c) = b$ then the polynomials $\mathbf{f}_a(x) = \mathbf{D}_i(a, b, x)$ and $\mathbf{f}_c(x) = \mathbf{D}_i(x, b, c)$ satisfy the assumptions of Lemma 3.4, as $\mathbf{f}_a(c) = b = \mathbf{f}_c(a)$ and, by monotonicity of the polynomials with respect to the order $\leqslant_\alpha$ $\mathbf{f}_a(b) = b = \mathbf{f}_c(b)$; the remaining properties $\mathbf{f}_a(a) = a$ and $\mathbf{f}_c(c) = c$ follow from the equality $\mathbf{D}_i(x, y, x) = x$ directed Gumm terms have to satisfy.

Now we induct on i to show that

(3.3) $\mathbf{D}_i(a, b, c) = a$ for $i = 1, \ldots, n$.

To start we note that $a = \mathbf{D}_1(a, a, c) \leqslant_\alpha \mathbf{D}_1(a, b, c) \leqslant_\alpha \mathbf{D}_1(b, b, c) = b$, so that $\mathbf{D}_1(a, b, c) \in \{a, b\}$ and in view of (3.2) we get $\mathbf{D}_1(a, b, c) = a$.

Using induction hypothesis $a = \mathbf{D}_i(a, b, c) \geqslant_\alpha \mathbf{D}_i(a, c, c)$ we get $a = \mathbf{D}_i(a, c, c) = \mathbf{D}_{i+1}(a, a, c) \leqslant_\alpha \mathbf{D}_{i+1}(a, b, c) \leqslant_\alpha \mathbf{D}_{i+1}(b, b, b) = b$, so that again $\mathbf{D}_{i+1}(a, b, c) \in \{a, b\}$ which together with (3.2) gives $\mathbf{D}_{i+1}(a, b, c) = a$, as required.

Now, $\mathbf{Q}(a, c, c) = \mathbf{D}_n(a, c, c) \leqslant_\alpha \mathbf{D}_n(a, b, c) = a$, while $\mathbf{Q}(a, c, c) \leqslant_\alpha \mathbf{Q}(b, b, c) = c$ which gives a contradiction as a and c are two different minimal elements in the coset a/α. This shows (3.1)

Finally, if $1_\mathbf{A}$ is join irreducible then we are done by (3.1). This means that $1_\mathbf{A} = \alpha_1 \vee \ldots \vee \alpha_k$ for some (maximal) join irreducible congruences $\alpha_1, \ldots, \alpha_k$ of $\mathbf{A}$ and that $k \geqslant 2$. With the help of (3.1) we can show that:

(3.4) There are $i \neq j$ and $b \in A$ such that $b/\alpha_i \nsubseteq b/\alpha_j \nsubseteq b/\alpha_i$.

Indeed, otherwise we would have that for all i, j the composition $\alpha_i \circ \alpha_j$ is contained in the union $\alpha_i \cup \alpha_j$, as $a \overset{\alpha_i}{\equiv} b \overset{\alpha_j}{\equiv} c$ would give $a = b$ or $b = c$. But having $\alpha_i \circ \alpha_j \subseteq \alpha_i \cup \alpha_j$ for all i, j implies that the congruence $1 = \alpha_1 \vee \ldots \vee \alpha_k \subseteq \alpha_1 \cup \ldots \cup \alpha_k$ has at most 2-element coset(s) contrary to our assumption that $|\mathbf{A}| \geqslant 3$. Finally, using the congruences α_i, α_j and $b \in A$ provided by (3.4) we can find $a, c \in A$ with $(a, b) \in \alpha_i - \alpha_j$ and $(b, c) \in \alpha_j - \alpha_i$. Using Items 3 and 4 of Lemma 3.3 we know that the sets $\{a, b\}$ and $\{b, c\}$ are minimal. Now we may finish our proof by applying Lemma 3.4 with $\mathbf{f}_a$ and $\mathbf{f}_c$ being the idempotent polynomials with the ranges $\{a, b\}$ and $\{b, c\}$, respectively. Obviously $\mathbf{f}_a(c) = b$ as $\mathbf{f}_a(c) \in \{a, b\}$ has to be α_j-congruent to $\mathbf{f}_a(b) = b$; the fact that also $\mathbf{f}_c(a) = b$ can be argued in the very same way. ◀

Note that an equation over a distributive lattice has a solution iff it has a solution in which all variables has the same value [23]. In case of 2-element lattice this algorithm trivially works also for ListCSat. Hence, we get the following corollary of Theorem 3.5.

▶ **Corollary 3.6.** *Let* $\mathbf{A}$ *be a finite algebra from a congruence modular variety such that* $\mathrm{typ}\{\mathbf{A}\} \subseteq \{\mathbf{4}\}$. *If* $|A| > 2$ *then* ListCSat($\mathbf{A}$) *is* NP*-complete. Otherwise* ListCSat($\mathbf{A}$) *is in* P.

4 Consequences of known results

The main goal of this section is to explain how to prove the following lemma, which characterizes algebras for which the problem may fail to be NP-complete.

▶ **Lemma 4.1.** *Let* $\mathbf{A}$ *be a finite algebra from a congruence modular variety. Then either* ListCSat($\mathbf{A}$) *is* NP*-complete, or* $\mathbf{A}$ *is one of the following:*

- *an algebra polynomially equivalent to the two-element lattice,*
- *a finite nilpotent algebra,*
- *a direct product of algebras mentioned above.*

As mentioned in the Introduction, hardness results for CSat also apply to ListCSat. In the remainder of this section we explain how results from [17] and [14], together with Corollary 3.6, imply Lemma 4.1.

(4.1) If an algebra $\mathbf{A}$ admits TCT type **3**, then ListCSat($\mathbf{A}$) is NP-complete.

This follows immediately from the fact that solving equations over $(\{0,1\}, \wedge, \neg)$ is NP-complete. If $\{0,1\}$ is a minimal set of type **3**, then in ListCSat we may restrict the lists of allowed values of variables to $\{0,1\}$. Consequently,

(4.2) If ListCSat($\mathbf{A}$) is not NP-complete for an algebra $\mathbf{A}$ from a congruence modular variety, then the typeset of $\mathbf{A}$ is contained in $\{\mathbf{2}, \mathbf{4}\}$.

Now [17, Theorems 6.4 and 6.5] imply that if α, β, γ are congruences of an algebra $\mathbf{A}$ from a congruence modular variety and either $\alpha \prec_4 \beta \prec_2 \gamma$ or $\alpha \prec_2 \beta \prec_4 \gamma$, then ListCSat($\mathbf{A}$) is NP-complete. Therefore the proof of [17, Corollary 6.5] yields the following statement.

(4.3) Let $\mathbf{A}$ be a finite algebra from a congruence modular variety. If ListCSat($\mathbf{A}$) is not NP-complete, then $\mathbf{A}$ is isomorphic to a direct product $\mathbf{A}_2 \times \mathbf{A}_4$, where $\mathrm{typ}\{\mathbf{A}_2\} \subseteq \{\mathbf{2}\}$ and $\mathrm{typ}\{\mathbf{A}_4\} \subseteq \{\mathbf{4}\}$.

The complexity of ListCSat for algebras from congruence modular varieties admitting only type **4** is characterized by Corollary 3.6. On the other hand, algebras from congruence modular varieties with typeset equal to $\{\mathbf{2}\}$ possess a Malcev term. Hardness results for such algebras can be found in [14, Theorem 1.1].

(4.4) Let $\mathbf{A}$ be a finite Malcev algebra. If $\mathbf{A}$ is not nilpotent, then CSat($\mathbf{A}$) (and hence also ListCSat($\mathbf{A}$)) is NP-complete.

However, Corollary 3.6, (4.3), and (4.4) do not immediately imply Lemma 4.1. In particular, it is not clear whether hardness of CSat($\mathbf{A}$) for $\mathbf{A} = \mathbf{A}_2 \times \mathbf{A}_4$ follows from hardness for $\mathbf{A}_2$ or $\mathbf{A}_4$. Although this is can not be true in general, in our situation we can show that if $\mathbf{A}_2$ is non-nilpotent or $\mathbf{A}_4$ is not polynomially equivalent to the two-element lattice, then ListCSat($\mathbf{A}$) is NP-complete. This is possible because all constructions used in the proofs of Corollary 3.6 and (4.4) take place inside minimal sets, and the typesets of $\mathbf{A}_2$ and $\mathbf{A}_4$ are disjoint. The first step to show it is the following observation.

(4.5) Let $\mathbf{A}$ be a finite algebra from a congruence modular variety and let $\alpha \prec_i \beta$, $\gamma \prec_j \delta$ be congruences of $\mathbf{A}$ such that $\{i,j\} = \{\mathbf{2}, \mathbf{4}\}$. Then for every (α, β)-minimal set U we have $\gamma|_U = \delta|_U$.

To prove (4.5), first observe that U cannot contain any (γ, δ)-minimal set. If $i = 4$, this is immediate since U has two elements and $i \neq j$. If $i = 2$, the claim follows from [8, Theorem 4.31]. Now suppose that for the polynomial $\mathbf{e}_U$ we have $\mathbf{e}_U(\delta) \not\subseteq \gamma$. Then $\mathbf{e}_U(A) = U$ would contain a (γ, δ)-minimal set, which is impossible. Hence $\mathbf{e}_U(\delta) \subseteq \gamma$, and since $\mathbf{e}_U$ acts as the identity on U, it follows that $\delta|_U \subseteq \gamma|_U$. The consequence of (4.5) is the following.

(4.6) Let $\mathbf{A} = \mathbf{A}_2 \times \mathbf{A}_4$ be an algebra from a congruence modular variety such that $\mathrm{typ}\{\mathbf{A}_2\} \subseteq \{\mathbf{2}\}$ and $\mathrm{typ}\{\mathbf{A}_4\} \subseteq \{\mathbf{4}\}$. Then every minimal set of $\mathbf{A}$ is of the form $U_2 \times \{a_4\}$ or $\{a_2\} \times V_4$, where $a_i \in A_i$ and U_2 and V_4 are minimal sets of $\mathbf{A}_2$ and $\mathbf{A}_4$, respectively.

Let $\alpha \prec_2 \beta$ be congruences of $\mathbf{A}$ and U be a (α, β)-minimal set. Suppose, towards a contradiction, that $(a_2, a_4), (b_2, b_4) \in U$ with $a_4 \neq b_4$. Let $\delta_4 = \Theta(a_4, b_4) \in \mathsf{Con}\,\mathbf{A_4}$ and choose $\gamma_4 \in \mathsf{Con}\,\mathbf{A_4}$ with $\gamma_4 \prec \delta_4$. Define $\gamma = 1_{\mathbf{A}_2} \times \gamma_4$ and $\delta = 1_{\mathbf{A}_2} \times \delta_4$. Clearly $\gamma, \delta \in \mathsf{Con}\,\mathbf{A}$ and $\gamma \prec_4 \delta$. Since $(a_4, b_4) \in \delta_4 - \gamma_4$, we have $((a_2, a_4), (b_2, b_4)) \in \delta - \gamma$. However, both pairs belong to U, and thus $\gamma|_U = \delta|_U$, a contradiction. Hence, $U = U_2 \times \{a_4\}$. Let α_4 and β_4 be projections of α and β on the first coordinate. It is clear that $\alpha_4, \beta_4 \in \mathsf{Con}\,\mathbf{A_4}$ and $\alpha_4 \prec \beta_4$. U_2 contains an (α_4, β_4)-minimal set as it is the range of the projection of $\mathbf{e}_U$, which is a polynomial of $\mathbf{A}_2$. On the other hand (α_4, β_4)-minimal set strictly contained in U_4 would imply (α, β)-minimal smaller then U. Hence, U_4 is an (α_4, β_4)-minimal set. The prove for minimal sets of type $\mathbf{4}$ is analogous.

Item (4.6) shows that the constructions used in the proofs of Corollary 3.6 and (4.4), which take place inside minimal sets, can be carried out not only in $\mathbf{A}_2$ and $\mathbf{A}_4$, but also in $\mathbf{A}$. Consequently, Lemma 4.1 follows.

5 CC circuits and programs

Lemma 4.1 can be seen as a variation of [17, Theorem 9.1], where a similar decomposition result was achieved for the CSat problem. Here we obtain a much clearer theorem for ListCSat. This is due to the fact that the lists nicely interfere with the algebraic structure in the hardness proofs.

In the nilpotent case quasipolynomial time algorithms were obtained by connecting the expressive power of circuits over nilpotent algebras to the expressive power of so-called CC-circuits [21, 13]. This connections led to a surprising finding of NP-intermediate problems in the realm of satisfiability of algebraic circuits [13]. Later even (randomized) polynomial time cases were quite well described by a more detailed exploration [15] through the notion of so-called programs over algebraic structures. Here we aim to extend these earlier ideas to be able to handle interaction between nilpotent and lattice behavior.

▶ **Definition 5.1.** *For an integer m a MOD_m-type gate is a Boolean gate MOD_m^S labeled with an accepting set $S \subseteq Z_m$. It sums the inputs modulo m and returns 1 iff the sum belongs to S. $CC_h[m]$ circuit is a height h circuit which has h layers of unbounded fan-in MOD_m-type gates. Different gates are allowed to admit different accepting sets.*

There is an important circuit complexity hypothesis that some authors refer to as Exponential Size Hypothesis (aka ESH) which might be viewed as a dual Hypothesis to the famous (and already proven) $2^{\Omega(n^c)}$ lower bound for AC^0 circuits computing PARITY [24, 7].

▶ **Conjecture 5.2** (ESH). *$CC_h[m]$ circuit computing boolean n-ary AND_n function requires size at least $2^{\Omega(n^c)}$, for some constant c depending on h and m.*

To connect expressive power of CC circuits to the expressive power of polynomials over nilpotent algebras we need a notion of a program.

▶ **Definition 5.3.** *Let* $\mathbf{A}$ *be a finite algebra. An* n*-ary program over* $\mathbf{A}$ *is a triple* $(\mathbf{p}, \iota, c)$*, where* $\mathbf{p}$ *is an* n*-ary polynomial over* $\mathbf{A}$*,* ι *is a collection of functions* $(\iota_i)_{i\in\{1..n\}}$ *of type* $\{0,1\} \longrightarrow A$ *and* $c \in A$ *is a constant. We will call each such* ι_i *an instruction. We say that a program* $(\mathbf{p}, \iota, c)$ *computes a Boolean function* $\{0,1\}^n \longrightarrow \{0,1\}$ *which is an indicator function of the condition* $\mathbf{p}(\iota_1(b_1), \iota_2(b_2), \ldots, \iota_n(b_n)) = c$*, i.e. the program returns* 1 *iff the condition is satisfied for a given* $\bar{b} \in \{0,1\}^n$*. We will write this condition as* $(\mathbf{p})[\iota]\,(\bar{b}) = c$ *for short.*

The following result from [19] explains the connection we will use.

▶ **Theorem 5.4.** *Let* $\mathbf{A}$ *be a finite solvable Malcev algebra, let* $(\mathbf{p}, \iota, c)$ *be a program of length* l *over* $\mathbf{A}$*. Then there exists* h, m *dependent only on* $\mathbf{A}$ *such that a characteristic function of an equation* $(\mathbf{p})[\iota]\,(\bar{b}) = c$ *can be computed by* $CC_h[m]$ *circuit of size* $O(poly(l))$*.*

We note that the proof from [19] considers only programs that use the same instruction for every variable (see the proof of Lemma 29 and Theorem 30 in [19]). However, the proof works perfectly well also in the case when the variables have possibly different instructions, one needs to apply only a trivial modification of replacing instruction ι locally by ι_i whenever ι was applied to the i-th variable. In fact the notion of programs [19] uses generalizes also to any expressions of the form $(\mathbf{p})[\iota]\,(\bar{b}) \in S$ where S is some accepting set. In our case we only consider sets S which only have one element, therefore they represent an equation.

To connect equations over nilpotent algebras to programs (and hence to CC circuits), we need the notion of indicator functions that was defined in [16], where the authors considered equations with lists in finite groups. A class of indicator functions is a class of functions with two closure properties and a notion of size associated to each function. Each of the functions has an associated arity k and is of type $S_1 \times ... \times S_k \longrightarrow \{0,1\}$, where $S_1, \ldots, S_k$ are some finite sets of size at least 2. The first closure property is that if we substitute a constant $a_i \in S_i$ for a variable x_i in such an indicator function $f(x_1, \ldots, x_k)$ we obtain another indicator function of arity $k-1$. The second property is, that if a list S_i is restricted to a subset $S_i' \subseteq S_i$ of size at least two we still obtain an indicator function, with smaller domain on the coordinate i. Moreover the size of the functions cannot increase while applying one of the two mentioned closure operators.

A *spike* is an indicator function such that $\left|f^{-1}(1)\right| = 1$. By $\gamma_C(n)$ we mean the smallest size for an n-ary spike in the class C of indicator functions. If such a spike does not exist for some n (which barely happens in our applications), the function is partial and undefined on n (note that it must be therefore undefined for all values bigger than n). By $\gamma_C^{-1}(m)$ we mean the maximal integer n such that $\gamma_C(n) \leqslant m$. Note that $\gamma(n)$ is nondecreasing in its domain and $\gamma_C^{-1}(n)$ is nondecreasing for all integer numbers as long as the class C contains any nonconstant function. In our proofs the most important indicator function family we consider is the characteristic functions for equations of the form $\mathbf{p}(x_1, .., x_n) = c$ with lists $S_1, \ldots, S_n$ of size at least 2, where the equation is in a finite nilpotent algebra and c is a constant from this algebra. The function indicates whether the equation is satisfied or not, and that is essentially the reason for the name. the second indicator function family is family of functions computed by programs $(\mathbf{p}, \iota, c)$ over nilpotent algebra, for which the instructions ι_i are nonconstant. In both cases the size is derived from the number of gates in the smallest possible circuit definition of $\mathbf{p}$, where we don't count constant nor variable gates on the input, and from now on when we write $|\mathbf{p}|$ we refer to this notion of size.

Let $\gamma_{List}(n)$, $\gamma_{Prog}(n)$ be the size of a spike in a specific indicator function family (where the algebra we are working with will be always clear from the context). Note that in fact $\gamma_{List}(n) = \gamma_{Prog}(n)$, as whenever we are given a list equation which has a singular solution we can narrow the lists to the size 2 while preserving the solution. Then we can use these lists to create program instructions for corresponding variables. Opposite is also true as we can use program instructions to create two element lists from their images. Hence we will not distinguish this two definitions of $\gamma(n)$ and we omit the subscript. This simple abstraction allows us to capture the density of a solution set to an equation in terms of a size of n-ary spikes.

▶ **Fact 5.5** (Proposition 9 in [16]). *Let f be an indicator function with the domain $S = S_1 \times \ldots \times S_k$ and k be a common upper bound for the size of all S_i. Then either $\left|f^{-1}(1)\right| = 0$ or $\left|f^{-1}(1)\right| / |S| \geqslant 1/|k|^{\gamma^{-1}(|f|)}$.*

Now we are ready to explain how the concepts introduced in this section connect to provide algorithms for ListCSat in a nilpotent case.

▶ **Lemma 5.6.** *Let $\mathbf{N}$ be a finite nilpotent Malcev algebra. Consider the equation $\mathbf{p}(\overline{x}) = \mathbf{q}(\overline{x})$ with nonempty lists $S_1, \ldots, S_n$. Let $(s_1, s_2, \ldots, s_n)$ be arbitrary element of $S_1 \times \ldots \times S_n$. Then*

1. *either the equation has no solution or at least $1/|N|^{\gamma^{-1}(|\mathbf{p}|+|\mathbf{q}|+1)}$ fraction of assignments is a solution.*
2. *if $n > \gamma^{-1}(|\mathbf{p}| + |\mathbf{q}| + 1)$ the equation has no solutions or at least two solutions.*
3. *either the equation has no solution or it has a solution $(s'_1, \ldots, s'_n)$ which is different from $(s_1, \ldots, s_n)$ on at most $\gamma^{-1}(|\mathbf{p}| + |\mathbf{q}| + 1)$ coordinates.*

Proof. Since N is nilpotent and Malcev it has a Malcev term $\mathbf{d}(x, y, z)$ which becomes a permutation of N after we fix the second and the third arguments. Hence the equation $\mathbf{p}(\overline{x}) = \mathbf{q}(\overline{x})$ is equivalent to the equation $\mathbf{d}(\mathbf{p}(\overline{x}), \mathbf{q}(\overline{x}), c) = c$, for arbitrary constant $c \in N$. Then Item 1 follows as an application of Fact 5.5. The only thing we have to take care of is the variables which have one element lists, but we can basically just substitute the unique value from the list and consider equation over smaller number of variables. Such substitution does not change the density of a solution set.

If Item 2 did not hold it would be a direct contradiction with the definition of γ, as then the polynomial $\mathbf{d}(\mathbf{p}(\overline{x}), \mathbf{q}(\overline{x}), c)$ of size $|p| + |q| + 1$ would be a witness for an n-ary spike.

For the Item 3, let us assume that a solution $(s'_1, \ldots, s'_n)$ differs from $(s_1, \ldots, s_n)$ on more than $\gamma^{-1}(|\mathbf{p}| + |\mathbf{q}| + 1)$ coordinates. Wlog assume that it differs on the first m coordinates. Let $\mathbf{p}'(\overline{x}) = \mathbf{d}(\mathbf{p}(\overline{x}), \mathbf{q}(\overline{x}), c)$. Then consider the equation $\mathbf{p}'(x_1, \ldots, x_m, s'_{m+1}, \ldots, s'_n) = c$ with lists $\{s_i, s'_i\}$ for each variable i. It must have at least two solutions or we would have $\gamma^{-1}(|\mathbf{p}'|) = \gamma^{-1}(|\mathbf{p}| + |\mathbf{q}| + 1) \geqslant m$. But if $s''_1, \ldots, s''_m$ is a different solution to this equation, then $(s''_1, \ldots, s''_m, s'_{m+1}, \ldots, s'_n)$ is a solution to the original equation that differs from $(s_1, \ldots, s_n)$ in fewer coordinates than $(s'_1, \ldots, s'_n)$. So by continuously improving the solution in such a way we will eventually end up with a solution which differs from $(s_1, \ldots, s_n)$ on at most $\gamma^{-1}(|\mathbf{p}| + |\mathbf{q}| + 1)$ coordinates. ◀

There are cases when the exact lower bounds for $\gamma(n)$ are known. For example, when $\mathbf{N}$ is not only nilpotent but also supernilpotent, γ is defined only on finitely many arguments. What follows $\gamma^{-1}(n)$ is bounded by a constant (see [18] and corollary to Theorem 6 in [2]). So the simple deterministic polynomial time algorithm for solving ListCSat($\mathbf{N}$) is to pick some $(s_1, s_2, \ldots, s_n) \in S_1 \times \ldots \times S_n$ and look for a solution satisfying list conditions that are in a

constant Hamming distance from $(s_1, s_2, \ldots, s_n)$. On the other hand in some cases when $\mathbf{N}$ is 2-nilpotent explicit lower bounds of the form $2^{\Omega(n)}$ are know for $\gamma(n)$ (that follow from [12] and [2]) which implies that $\gamma^{-1}(n)$ is $O(\log n)$. This leads to a deterministic algorithm of complexity $n^{O(\log n)}$. However Item 1 allows us just to probe $O(\mathrm{poly}(n))$ random assignments and if there exist solution to an equation we can find it with large probability by Markov inequality (see [15] for details). So there is also a probabilistic RP algorithm in this case. For general nilpotent case it is a very hard open problem to establish nontrivial lower bounds for $\gamma(n)$. However assuming ESH we obtain one useful such bound.

▶ **Fact 5.7.** *Let* $\mathbf{N}$ *be a finite nilpotent Malcev algebra. Then assuming ESH* $\gamma(n)$ *can be bounded from below by* $2^{\Omega(n^\epsilon)}$*. It follows that* $\gamma^{-1}(n)$ *can be bounded from above by* $O(poly(\log n))$*. In consequence* ListCSat($\mathbf{N}$) *has a quasipolynomial time algorithm.*

Proof. Let $(\mathbf{p}, \iota, c)$ be a program over $\mathbf{N}$ representing a spike. Then the equation $(\mathbf{p})[\iota](\mathbf{b}) = c$ has only one solution, without loss of generality assume it is $(1, 1, ..., 1)$ as we can always swap values for $\iota_i(0)$ and $\iota_i(1)$. But then by Theorem 5.4 the program $(\mathbf{p}, \iota, c)$ can be rewritten to $CC_h[m]$ circuit of size $O(\mathrm{poly}(|\mathbf{p}|))$ for some integers h, m depending only on $\mathbf{N}$. But then this $CC_h[m]$ circuit computes an AND_n function, so by ESH it has size at least $2^{\Omega(n^\epsilon)}$. This lower bound transfers to the size of $\mathbf{p}$, with possible change of the constant in Ω notation. Hence $\gamma(n) \geqslant 2^{\Omega(n^\epsilon)}$. This gives us the desired bound for γ^{-1} and the deterministic algorithm for ListCSat($\mathbf{N}$) by Item 3 in Lemma 5.6. ◀

6 Quasipolynomial time algorithm for products

Note that if $\mathbf{A}$ is decomposable like in Lemma 4.1 into a direct product $\mathbf{L} \times \mathbf{N}$, where $\mathbf{L}$ is polynomially equivalent to two element lattice and N is a nilpotent algebra, we can wlog write every element $a \in A$ as a tuple $(a^{\mathbf{L}}, a^{\mathbf{N}})$.

Note that we cannot directly apply the reasoning presented in the previous section to the product $\mathbf{L} \times \mathbf{N}$, because we can easily encode a spike function as a circuit over $\mathbf{L} \times \mathbf{N}$. Indeed, polynomials of $\mathbf{L}$ can encode a conjunction, and thus for at least some of these algebras $\mathbf{A}$ we have $\gamma_{\mathbf{A}}(n) \leqslant O(n)$ (and thus $\gamma_{\mathbf{A}}^{-1}(n) \geqslant 2^{\Omega(n)}$). We cannot solve the equation separately for the coordinates $\mathbf{L}, \mathbf{N}$ either (like one could for the CSat problem), because lists encode nontrivial interactions between coordinates. We will propose an algorithm that deals with such interactions.

Assume we are given some variables $x_1, \ldots, x_n$ with lists $S_1, \ldots, S_n \subseteq A$. Without loss of generality assume that the underlying subset of $\mathbf{L}$ is just $\{0, 1\}$. Then given lists we can define functions $\{0, 1\} \longrightarrow P(N)$ such that $T_i(b) = \{c \in \mathbf{N} : (b, c) \in S_i\}$. Note that having all the functions T_i we can reconstruct the initial lists S_i in an obvious way.

▶ **Definition 6.1.** *Let* $\mathbf{p}(x_1, \ldots, x_n) = \mathbf{q}(x_1, \ldots, x_n)$ *be an equation over* $\mathbf{A} = \mathbf{L} \times \mathbf{N}$ *with lists* $S_1, \ldots, S_n$*. A b-solution is a tuple* $\overline{a}$ *such that* $\mathbf{p}(a_1, \ldots, a_n) = \mathbf{q}(a_1, \ldots, a_n) = (b, c)$ *for some* $c \in N$*. We call a variable* x_i *determined whenever one of the sets* $T_i(0), T_i(1)$ *is empty. We say that variable* x_i *is choiceless whenever* $T_i(0) = T_i(1)$ *and* $|T_i(0)| = |T_i(1)| = 1$*. If the variable is neither choiceless nor determined, we call it choice-essential.*

The following Lemma is the key to our algorithm. From now $\gamma = \gamma_{\mathbf{N}}$ for the nilpotent part of a product $\mathbf{L} \times \mathbf{N}$.

▶ **Lemma 6.2.** *Let* $\mathbf{p}(x_1, \ldots, x_n) = \mathbf{q}(x_1, \ldots, x_n)$ *be an equation over* $\mathbf{A} = \mathbf{L} \times \mathbf{N}$ *of length* l *with lists* $S_1, \ldots, S_n$ *such that all its variables are choice-essential. Suppose it admits a b-solution for some* $b \in \{0, 1\} = L$*. Then it also has a (possibly different) b-solution such that at most* $\gamma^{-1}(|\mathbf{p}| + |\mathbf{q}| + 1)$ *variables* x_i *satisfy* $x_i^{\mathbf{L}} = 1 - b$*.*

Proof. First note that equation itself has two components $\mathbf{p}^{\mathbf{L}}(x_1^{\mathbf{L}},\dots,x_n^{\mathbf{L}}) = \mathbf{q}^{\mathbf{L}}(x_1^{\mathbf{L}},\dots,x_n^{\mathbf{L}})$ and $\mathbf{p}^{\mathbf{N}}(x_1^{\mathbf{N}},\dots,x_n^{\mathbf{N}}) = \mathbf{q}^{\mathbf{N}}(x_1^{\mathbf{N}},\dots,x_n^{\mathbf{N}})$.

Let $\overline{a}$ be a supposed b-solution, our goal is to improve it to have at most $\gamma^{-1}(|\mathbf{p}|+|\mathbf{q}|+1)$ indices i satisfying $a_i^{\mathbf{L}} = 1-b$. First, if there is some index i such that $a_i^{\mathbf{L}} = 1-b$ and the list S_i contains both tuples $(0,a_i^{\mathbf{N}}),(1,a_i^{\mathbf{N}})$ we can just flip the value $a_i^{\mathbf{L}}$ and preserve the list condition while making $x_i^{\mathbf{L}} = b$ satisfied for one more index i. The $\mathbf{L}$ component $\mathbf{p}^{\mathbf{L}}(x_1^{\mathbf{L}},\dots,x_n^{\mathbf{L}}) = \mathbf{q}^{\mathbf{L}}(x_1^{\mathbf{L}},\dots,x_n^{\mathbf{L}})$ of the original equation is still satisfied, since all polynomials of $\mathbf{L}$ compute only monotone functions, hence if both sides of the equation are already equal to b they will still be equal after such flip. So assume there is no such coordinate i in $(a_1,\dots,a_n)$. As we can freely permute variable indices in the equation, we can assume that the first s of the indices satisfy $a_i^{\mathbf{L}} = 1-b$ and the rest $a_i^{\mathbf{L}} = b$. Consider the equation $\mathbf{p}(x_1,\dots,x_s,a_{s+1},\dots,a_n) = \mathbf{q}(x_1,\dots,x_s,a_{s+1},\dots,a_n)$. Because for $i \in \{1..s\}$ variables x_i are not choiceless, we can pick a value $c_i \in N$ such that $(b,c_i) \in S_i$ and $c_i \neq a_i^{\mathbf{N}}$. Indeed the only way this would not be possible is $T_i(b) = \{a_i^{\mathbf{N}}\}$ which we already resolved. Now let ι_i be a function defined by $\iota_i(1-b) = a_i^{\mathbf{N}}, \iota_i(b) = c_i$. If the number s associated to the equation $\mathbf{p}(x_1,\dots,x_s,a_{s+1},\dots,a_n) = \mathbf{q}(x_1,\dots,x_s,a_{s+1},\dots,a_n)$ with lists $\{a_i^{\mathbf{N}},c_i\}$ is bigger than $\gamma^{-1}(|\mathbf{p}|+|\mathbf{q}|+1)$ it must have at least one solution other than $(a_1^{\mathbf{N}},\dots,a_s^{\mathbf{N}})$ say $(d_1,\dots,d_s)$. But then $((\iota_1^{-1}(d_1),d_1),\dots,(\iota_s^{-1}(d_s),d_s),a_{s+1},\dots,a_n)$ is a solution to the original equation $\mathbf{p}(x_1,\dots,x_n) = \mathbf{q}(x_1,\dots,x_n)$ which satisfies the list conditions $x_i \in \{a_i^{\mathbf{N}},c_i\}$ and has less coordinates i with $x_i^{\mathbf{L}} = 1-b$. Indeed it satisfies the $\mathbf{L}$ part of the equation by monotonicity and it clearly satisfies the $\mathbf{N}$ part. By repeatedly decreasing number of coordinates on which $x_i^{\mathbf{L}} = 1-b$ we arrive at a solution with $s \leqslant \gamma^{-1}(|\mathbf{p}|+|\mathbf{q}|+1)$ that we desired. ◀

Now we are finally ready to prove the main theorem of this section.

▶ **Theorem 6.3.** *Let* $\mathbf{A}$ *be of the form* $\mathbf{L}\times\mathbf{N}$ *where* $\mathbf{L}$ *is an algebra polynomially equivalent to a two element lattice and* $\mathbf{N}$ *is a finite nilpotent Malcev algebra. Then* ListCSat($\mathbf{A}$) *can be solved in time* $n^{O(\gamma^{-1}(l))}$ *where* n *is the number of variables of the input equation of length* l*. In particular, assuming ESH,* ListCSat($\mathbf{A}$) *is solvable in quasipolynomial time.*

Proof. First assume that the first k variables of the input equation are choiceless, afterwards there are r variables that are determined and last s variables are choice-essential, for if not we properly permute variables on the input. Grouping variables is an easy, syntactic task and can be done in polynomial (linear) time. Hence our equation is of the form $\mathbf{p}(x_1,\dots,x_k,y_1,\dots,y_r,z_1,\dots,z_s) = \mathbf{q}(x_1,\dots,x_k,y_1,\dots,y_r,z_1,\dots,z_s)$ with list S_x is given for each variable $x \in \{x_1,..,z_s\}$. We will assume that none of the lists S_x is one element set, as in this case we would immediately substitute constant for the variable and reduce to a problem with less variables.

The strategy is to try to solve the component equations for $\mathbf{N}$ and $\mathbf{L}$ separately while carefully treating interactions between components first. We try to find b-solution for either $b=0$ or $b=1$, so we reduce to two such cases immediately. Firstly we will eliminate variables $z_i^{\mathbf{L}}$ for $i \in \{1..s\}$. Note that if there exists some solution to the equation assigning value $a(x)$ to variables $x \in \{x_1,\dots,z_s\}$ then the equation $\mathbf{p}(a(x_1),\dots,a(x_k),a(y_1),\dots,a(y_r),z_1,\dots,z_s) = \mathbf{q}(a(x_1),\dots,a(x_k),a(y_1),\dots,a(y_r),z_1,\dots,z_s)$ with lists S_{z_i} corresponding to variables z_i has, by Lemma 6.2, a possibly different solution with at most $\gamma^{-1}(l)$ indices i satisfying $z_i = 1-b$ (here l can be a bit larger than $|p|+|q|+1$ due to encoding, but γ^{-1} is nondecreasing so it still works). There is $O(n^{\gamma^{-1}(l)})$ such assignments and we consider all of them. For every considered assignment we have to update lists, i.e. if we set variable $z_i^{\mathbf{L}}$ to value b_i we introduce the set $\{c : (b_i,c) \in S_{z_i}\}$ as a new list for variable $z_i^{\mathbf{N}}$. Clearly, if we don't find a solution in any of such partially assigned cases there is no solution for unrestricted $z_i^{\mathbf{L}}$ and our algorithm will return the answer *no solution*.

Now notice that we can also eliminate the $\mathbf{L}$ parts of the determined variables $x_1^{\mathbf{L}}, .., x_k^{\mathbf{L}}$ as there is simply only one value we can assign to them. Then if $x_i^{\mathbf{L}}$ was put to the value b_i we introduce a list $\{c : (b_i, c) \in S_{x_i}\}$ for the variable $x_i^{\mathbf{N}}$. Similarly for choiceless variables $y_1, .., y_r$ we can set their nilpotent part to the only value they can be set to and then the remaining variables $y_i^{\mathbf{L}}$ have trivial lists $\{0, 1\}$. Such elimination of variables is basically a one-to-one reduction and can be done in a straightforward way.

After this whole process we are left with two equations $\mathbf{t}^{\mathbf{L}}(y_1^{\mathbf{L}}, \dots, y_r^{\mathbf{L}}) = \mathbf{u}^{\mathbf{L}}(y_1^{\mathbf{L}}, \dots, y_r^{\mathbf{L}})$ and $\mathbf{t}^{\mathbf{N}}(x_1^{\mathbf{N}}, \dots, x_k^{\mathbf{N}}, z_1^{\mathbf{N}}, \dots, z_s^{\mathbf{N}}) = \mathbf{u}^{\mathbf{N}}(x_1^{\mathbf{N}}, \dots, x_k^{\mathbf{N}}, z_1^{\mathbf{N}}, \dots, z_s^{\mathbf{N}})$, where $\mathbf{t}^{\mathbf{L}}, \mathbf{u}^{\mathbf{L}}$ are polynomials of $\mathbf{L}$ and $\mathbf{t}^{\mathbf{N}}, \mathbf{u}^{\mathbf{N}}$ are polynomials of $\mathbf{N}$. But these are equations over totally disjoint sets of variables and lists do not introduce any dependencies anymore! Thus we can solve them completely independently. The equation for the $\mathbf{L}$ part is trivial because it is always preferred to assign value b to each variable due to monotonicity, as if there is any solution then $(b, .., b)$ is also a solution. We can also solve an equation $\mathbf{N}$ in an apropriate time by the application of Lemma 5.6. Overall we reduce to $n^{O(\gamma^{-1}(l))}$ recursive subinstances in which the most time consuming task is to solve equation with lists in the nilpotent case which can take time up to $n^{O(\gamma^{-1}(l))}$. Multiplying this two complexities we again get a function of the form $n^{O(\gamma^{-1}(l))}$. ◀

As mentioned in the previous section for supernilpotent algebras $\gamma(n)$ can be defined only on finitely many points. What follows $\gamma^{-1}(n)$ is bounded. This allows us to eliminate the assumption of ESH.

▶ **Corollary 6.4.** *Let* $\mathbf{A}$ *be of the form* $\mathbf{L} \times \mathbf{N}$ *where* $\mathbf{L}$ *is an algebra polynomialy equivalent to a two element lattice and* $\mathbf{N}$ *is a finite supernilpotent Malcev algebra. Then* ListCSat($\mathbf{A}$) $\in$ P.

The paper [12] studies satisfiability and equivalence problems for 2-nilpotent algebras, i.e. algebras which are isomorphic to a Freese-Mcckenzie product $\mathbf{M}_1 \otimes \mathbf{M}_2$ where $\mathbf{M}_1, \mathbf{M}_2$ are polynomially equivalent to finite modules. For the exact definition see Chapter VII in [5], it is worth noting that by using this product one can build arbitrary finite nilpotent Malcev algebra from module like components. It follows from [12] together with [2], that if $\mathbf{M}_1, \mathbf{M}_2$ are polynomially equivalent to one dimensional vector spaces of coprime characteristics then $\gamma^{-1}(n)$ is bounded by $O(\log n)$.

▶ **Corollary 6.5.** *Let* $\mathbf{A}$ *be of the form* $\mathbf{L} \times \mathbf{N}$ *where* $\mathbf{L}$ *is an algebra polynomially equivalent to a two element lattice and* $\mathbf{N} = \mathbf{M}_1 \otimes \mathbf{M}_2$*, where* $\mathbf{M}_1, \mathbf{M}_2$ *are polynomially equivalent to finite one dimensional vector spaces of coprime characteristics. Then* ListCSat($\mathbf{A}$) *can be solved in* $n^{O(\log n)}$ *steps.*

Interestingly even probabilistic polynomial time algorithm follows from Lemma 5.6 for ListCSat($\mathbf{N}$) in Corollary 6.5. It is a curious problem whether the product $\mathbf{L} \times \mathbf{N}$ also admits a (probabilistic) polynomial time algorithm, any naive approaches seem to be insufficient.

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, Howard Straubing, and Denis Thérien. Non-uniform automata over groups. *Inf. Comput.*, 89(2):109–132, 1990. doi:10.1016/0890-5401(90)90007-5.

3 Stanley N. Burris and H. P. Sankappanavar. A course in universal algebra, 2012. Millennium Edition, freely available PDF. URL: https://www.math.uwaterloo.ca/~snburris/htdocs/UALG/univ-algebra2012.pdf.

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

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

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

7 Johan Hastad. *Computational limitations for small depth circuits*. PhD thesis, Massachusetts Institute of Technology, 1986.

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

9 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.

10 Gábor Horváth and Csaba Szabó. Equivalence and equation solvability problems for the alternating group A_4. *J. Pure Appl. Algebra*, 216(10):2170–2176, 2012. doi:10.1016/j.jpaa.2012.02.007.

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

12 Pawel M. Idziak, Piotr Kawalek, and Jacek Krzaczkowski. Expressive power, satisfiability and equivalence of circuits over nilpotent algebras. In *43rd International Symposium on Mathematical Foundations of Computer Science (MFCS 2018)*, volume 117 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 17:1–17:15, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2018.17.

13 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, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3373718.3394780.

14 Paweł M. Idziak, Piotr Kawałek, and Jacek Krzaczkowski. Satisfiability of circuits and equations over finite Malcev algebras. In *39th International Symposium on Theoretical Aspects of Computer Science (STACS 2022)*, volume 219 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 37:1–37:14, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2022.37.

15 Paweł M. Idziak, Piotr Kawałek, and Jacek Krzaczkowski. Nonuniform deterministic finite automata over finite algebraic structures. In *52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025)*, volume 334 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 161:1–161:14, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2025.161.

16 Paweł M. Idziak, Piotr Kawałek, Jacek Krzaczkowski, and Armin Weiß. Satisfiability problems for finite groups. In *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022*, volume 229 of *LIPIcs*, pages 127:1–127:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ICALP.2022.127.

17 Paweł M. Idziak and Jacek Krzaczkowski. Satisfiability in multi-valued circuits. *SIAM J. Comp.*, 51(3):337–378, 2022. (Conference version at LICS'18, https://doi.org/10.1145/3209108.3209173). doi:10.1137/18M1220194.

18 Piotr Kawałek and Jacek Krzaczkowski. Even faster algorithms for CSAT over supernilpotent algebras. In *45th International Symposium on Mathematical Foundations of Computer Science (MFCS 2020)*, volume 170 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 55:1–55:13, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.MFCS.2020.55.

19 Piotr Kawałek and Jacek Krzaczkowski. Complexity classes arising from circuits over finite algebraic structures. In *41st Annual Symposium on Logic in Computer Science (LICS 2026)*, volume 380 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 70:1–70:26, 2026. doi:10.4230/LIPIcs.LICS.2026.70.

20 Alexandr Kazda, Marcin Kozik, Ralph McKenzie, and Matthew Moore. Absorption and directed jónsson terms. In *Don Pigozzi on Abstract Algebraic Logic, Universal Algebra, and Computer Science*, pages 203–220. Springer International Publishing, Cham, 2018. doi:10.1007/978-3-319-74772-9_7.

21 Michael Kompatscher. CC-circuits and the expressive power of nilpotent algebras. *Logical Methods in Computer Science*, Volume 18, Issue 2, May 2022. doi:10.46298/lmcs-18(2:12)2022.

22 Michael Kompatscher. CSAT and CEQV for nilpotent maltsev algebras of fitting length > 2, 2023. arXiv:2105.00689.

23 Bernhard Schwarz. The complexity of satisfiability problems over finite lattices. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 31–43. Springer, 2004. doi:10.1007/978-3-540-24749-4_4.

24 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles (preliminary version). In *26th Annual Symposium on Foundations of Computer Science, FOCS 1985*, pages 1–10. IEEE Computer Society, 1985. doi:10.1109/SFCS.1985.49.