Complexity Classes Arising from Circuits over Finite Algebraic Structures

Piotr Kawałek
Institute of Discrete Mathematics and Geometry, TU Wien, Austria

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

Abstract

Most classical results in circuit complexity theory concern circuits over the Boolean domain. Besides their simplicity and the ease of comparing different languages, the actual architecture of computers is also an important motivating factor. On the other hand, by restricting attention to Boolean circuits, we lose sight of the much richer landscape of circuits over larger domains. Our goal is to bridge these two worlds: to use deep algebraic tools to obtain results in computational complexity theory, including circuit complexity, and to apply results from computational complexity to gain a better understanding of the structure of finite algebras.

In this paper, we propose a unifying algebraic framework which we believe will help achieve this goal. Our work is inspired by branching programs and nonuniform deterministic automata introduced by Barrington, as well as by their generalization proposed by Idziak et al. We begin our investigation by studying the languages recognized by natural classes of algebraic structures. In particular, we characterize language classes recognized by circuits over simple algebras and over algebras from congruence modular varieties.

2012 ACM Subject Classification Theory of computation → Circuit complexity; Theory of computation → Complexity classes

Keywords and phrases Circuit Complexity, Universal Algebra, algebraic branching programs

Digital Object Identifier 10.4230/LIPIcs.LICS.2026.61

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. 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.
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.

1 Introduction

It is a fact that algebraic insights can lead to breakthroughs in computational complexity theory. Let us only mention a few examples: polynomials over finite fields and their usefulness in establishing IP = PSPACE [23, 35]; modelling program verification with a Gröbner basis problems [34, 31]; the group-theoretic proof of Barrington that languages from NC^1 can be recognized by bounded-width branching programs [4]; currently explored machine learning algorithms heavily based on linear algebraic concepts [22, 40]. One could enumerate so many more brilliant applications of algebra. There seems to be only a thin boundary between the world of Turing Machines and the world of Algebra.

Perhaps the most demonstrative example of that in the past decade was the P vs. NP-complete dichotomy theorem for finite Constraint Satisfaction Problems (CSPs) [10, 41], which model a large class of natural algorithmic problems. In the theory of CSP the key idea was to study not only the combinatorial nature of the constraints which define the problem but rather the interaction between the combinatorics and the symmetries of the constraints. These symmetries, called polymorphisms, are multiary functions which are closed under compositions and preserve the constraints. They form a structure called a termal clone. The field of Universal Algebra, which has studied termal clones for about a century now, is what enabled this big CSP research project.

The other direction also holds and the computational complexity notions can help to understand some of the algebraic concepts deeper. A prime example of that are Nonuniform Deterministic Finite Automata studied by Barrington, Straubing and Thérien [6], where the expressive power of polynomial-size expressions over finite groups is linked to circuit complexity classes. Here DFA appear because, due to associativity, one may interpret these nonuniform expressions as acting on the state set of the finite automaton. For instance, one can solve arbitrary problems in nonuniform NC^1 (including all the problems with very effective parallel algorithms) with polynomial-size expressions over any nonsolvable group. On the contrary, solvable groups seem to have much smaller computational power as expressions over such groups can only solve problems that are also solvable by bounded-depth circuits (in the class CC^0). Nilpotent groups provably have very small expressive power, as they essentially can only evaluate bounded-degree polynomials. Similar research was done for monoids, where expressions over solvable monoid seem to have more power than the ones of solvable groups [7] (they solve problems in the class ACC^0).

Later it was spotted that the insights from [6] can be used to obtain polynomial or quasipolynomial time algorithms for the problem of solving equations in solvable groups/-monoids [5, 14]. The problems of solving equations was then considered in a much more general context of finite algebras [19], which means finite universes with arbitrary finite sets of operations. Note that the class of these problems is very rich as it generalizes class of all finite CSPs (see [19]). The research was especially successful for the algebras from so-called congruence modular varieties. These algebras generalize most of the structures studied in classical algebra, such as groups, rings, Boolean Algebra, Heyting Algebras, Lie Algebras, lattices, latin matrices and many more. Recently some assumptions about circuit lower bounds have proven their usefulness in some variants of satisfiability problems for groups [18] as well as general algebras [17] to provide randomized polynomial time algorithms.

In this paper we want to continue to study the connection between expressive power of circuits over finite algebras and classical Boolean circuit complexity classes. To define this correspondence correctly, we define how these multivalued circuits will be used to recognize languages. Say we have some finite universe A and some set of operations $f_1, \dots, f_k$ over A that we will use as gates while building our circuit. This is our algebraic structure $\mathbf{A}$. Variables as well as constants from A can appear on the input level of such circuits. This connects our consideration to the so-called polynomial clone of $\mathbf{A}$, that is composition closed set of operations that contains all constants from the universe and all projections. Let us say we have a language $L \subseteq \Sigma^*$ we want to recognize. Then we need a way to project the alphabet Σ to the universe of our algebra, we need a way to interpret the result (we need the accepting set $S \subseteq A$) and because there are many possible input lengths we need not only one circuit, but a sequence of circuits over $\mathbf{A}$ (that is why this model of computation is called nonuniform).

▶ **Definition 1.** *Non-uniform Deterministic Finite Automaton (NuDFA) over a finite algebra* $\mathbf{A}$ *and a finite alphabet* Σ *consists of:*

- *a sequence* $\{\mathbf{t}_i\}_{i=1}^{\infty}$ *of circuits over* $\mathbf{A}$ *such that circuit* $\mathbf{t}_n$ *is* n*-ary.*
- *function* $\iota : \Sigma \mapsto A$*,*
- *set* $S \subseteq A$*,*

A word $b_1 \dots b_n \in \Sigma^n$ gets accepted by such a NuDFA if $\mathbf{t}_n(\iota(b_1), \dots, \iota(b_n)) \in S$. A triple $(\mathbf{t}, \iota, S)$ where $\mathbf{t}$ is an n-ary circuit is called an n-ary program (in correspondence to straight-line programs). We will write $\mathbf{t}[\iota](\bar{b})$ to denote $\mathbf{t}(\iota(b_1), \dots, \iota(b_n))$ and $\mathbf{t}[\iota, S](\bar{b})$ as a characteristic function of language recognized by program (t, ι, S). We say that a NuDFA $(\{\mathbf{t}_i\}_{i=1}^{\infty}, \iota, S)$ is of polynomial size if there is a polynomial w such that the size $|\mathbf{t}_i| \leq w(i)$ for every i. Here by size we mean the number of nodes in the definition of $\mathbf{t}_i$. The set of languages over the binary alphabet $\{0, 1\}$ recognized by polynomial size NuDFA's over $\mathbf{A}$ we denote as $nu\mathsf{P}_{\mathbf{A}}$. Note that the class $nu\mathsf{P}_{\mathbf{A}}$ is closed under taking complement (as we can take the complement of the accepting set S) and bitwise complement (as we can take $\iota'(b) = \iota(\neg b)$).

Note that in [17] the word NuDFA was used only in reference to Barrington, Straubing and Thériens' work, and in fact one should think of NuDFA over algebraic structures as of nonuniform circuit family rather than nonuniform DFA. Also, unlike [17], we fix ι, S for the circuit family. It is much simpler, more natural, and it allows us to capture certain complexity classes which would not appear otherwise. In particular monotone circuits can appear in our theorems only due to this change, and they better correspond to the structure of considered algebras.

Our notation $nu\mathsf{P}_{\mathbf{A}}$ refers to nonuniform models of computation such as $\mathsf{P}/poly$, known as nonuniform P, which in our notation is equal to $nu\mathsf{P}_{\mathbf{A}}$ for $\mathbf{A} = (\{0, 1\}, \wedge, \vee, \neg)$. $\mathsf{P}/poly$ is defined to be class of languages recognized by a family of polynomial size Boolean circuits using standard $\wedge, \vee, \neg$ operations. It is the highest possible class $nu\mathsf{P}_{\mathbf{A}}$ can reach, as the $\wedge, \vee, \neg$ can simulate all the other operations on finite set. It was widely hoped in 70s and 80s that studying the class $\mathsf{P}/poly$ was the right way to approach a P vs. NP problem, as if one can prove $\mathsf{NP} \not\subseteq \mathsf{P}/poly$ it implies $\mathsf{P} \neq \mathsf{NP}$. As a natural proof barrier appeared on the scene [30] and the progress slowed down, enthusiasm was a bit toned down, but still $\mathsf{P}/poly$ is considered one of the most important classes in the complexity zoo.

Even though lower bounds for $\mathsf{P}/poly$ seem to be far out of reach for the current techniques, in some restricted settings full success has been achieved.

▶ **Example 2.** Let $\mathbf{A} = (\{0, 1\}, \wedge, \vee)$. Languages recognized by NuDFAs over $\mathbf{A}$ fulfill one of the two following properties:

1. If $(a_1, \dots, a_n) \in L$, then for every $(b_1, \dots, b_n)$ such that $a_i \leq b_i$ holds, also $(b_1, \dots, b_n) \in L$ holds.
2. If $(a_1, \dots, a_n) \in L$, then for every $(b_1, \dots, b_n)$ such that $a_i \geq b_i$ holds, also $(b_1, \dots, b_n) \in L$ holds.

$nu\mathsf{P}_{(2,\wedge,\vee)}$ is equal to the set of languages whose characteristic function or its negation can be described by the sequence of polynomial size circuits built of gates $\wedge$ and $\vee$.

Note that the study of monotone (in the sense of Item 1) polynomial-size circuits has a big history with applications of some of the techniques in Proof Complexity [21, 29, 9, 27]. Importantly, there are NP-complete languages that are monotone. With the right encoding CLIQUE is such a problem because adding an edge can produce only bigger cliques. By the classical result of Razborov CLIQUE requires superpolynomial-size monotone circuits [28], which was later improved to show that the required size is actually exponential [2]. Finally

it turns out that some monotone problems in P including perfect matching problem require superpolynomial-size circuits [28, 36], and the lower bound for matching was recently claimed to be lifted to exponential [11]. So the negation really does a great job at making some of our algorithmic problems easier. All these results also apply to $nu\mathsf{P}_{\mathbf{A}}$ from Example 2.

From Post's theory [26] of algebraic clones on the two element domain one can see that there are only few possible polynomial clones, i.e. Boolean clone, lattice clone, clone generated by xor, semilattice clones, unary clones. In contrast, there are uncountably many of polynomial clones on 3 element domain [1]. Fortunately, we will only deal with countably many of them in this paper, as we consider algebras with finitely many generating operations.

In this vastness of structures, we can expect to find variety of complexity classes arising from them. In this paper we explore this space, on the one hand we study algebras from the congruence modular varieties, for which the tools of Universal Algebra such as Tame Congruence Theory and Commutator Theory work particularly well. We effectively propose a representation theory for them, which allows us to achieve the characterizations of $nu\mathsf{P}_{\mathbf{A}}$. On the other hand, we also study simple algebras for which we achieve almost full description with some interesting open cases (see Section 10). For finite algebras there is no theory of representation similar to the theory of groups or to Krohn-Rhodes Theory for monoids. Some results in this direction can be found in [39]. Our research suggests that such representations might be possible to achieve, perhaps for bigger classes of structures than expected before.

We will also see that many natural classes studied in computer science appear as $nu\mathsf{P}_{\mathbf{A}}$ or as a limit of $nu\mathsf{P}_{\mathbf{A}}$ for natural classes of structures. This provides an algebraic common view on the well-known complexity classes and provides yet another perspective on some of the open problems in computational complexity theory.

2 Algebraic Closure Properties

In order to use algebraic tools in our research, the properties of the languages recognized by the automata must be related to the algebraic properties of the algebras used in the automata definition. We will not go deeply into Universal Algebra here, but it is useful, also for the future proofs, to establish some basic facts here.

Subalgebra is a subset of an algebra closed under operations of this algebra together with the inherited operations. The set of languages recognized by programs over a subalgebra is contained in the set of languages recognized by programs over the whole algebra.

▶ **Fact 3.** *Let* $\mathbf{A}$ *be a finite algebra and let* $\mathbf{B}$ *be a subalgebra of* $\mathbf{A}$*.*
Then, $nu\mathsf{P}_{\mathbf{B}} \subseteq nu\mathsf{P}_{\mathbf{A}}$*.*

Normal subgroups and quotient groups are well known notions of group theory. Congruences and quotient algebras are their analogues in universal algebra. Formally, congruences are equivalence relations preserved by operation of the algebra. They can be described as all possible kernels for homomorphisms of $\mathbf{A}$. As in the case of normal subgroups congruences form a lattice. Languages recognized by the quotient algebra are contained in the set of languages recognized by NuDFAs over the algebra.

▶ **Fact 4.** *Let* $\mathbf{A}$ *be a finite algebra and let* α *be a congruence of* $\mathbf{A}$*.*
Then, $nu\mathsf{P}_{\mathbf{A}/\alpha} \subseteq nu\mathsf{P}_{\mathbf{A}}$*.*

A little bit more complicated is situation in case of product of two algebras. Note that computing value of a circuit over a product $\mathbf{A} \times \mathbf{B}$ of two algebras $\mathbf{A}$ and $\mathbf{B}$ is like computing values for both algebras $\mathbf{A}$ and $\mathbf{B}$ separately. In case of programs the additional complication is that the accepting set in NuDFAs over $\mathbf{A} \times \mathbf{B}$ need not be the product of subsets of A and B.

▶ **Fact 5.** *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be finite algebras.*
Then,

- $nu\mathsf{P}_{\mathbf{A}} \subseteq nu\mathsf{P}_{\mathbf{A}\times\mathbf{B}}$.
- $nu\mathsf{P}_{\mathbf{B}} \subseteq nu\mathsf{P}_{\mathbf{A}\times\mathbf{B}}$.
- *if* $L \in nu\mathsf{P}_{\mathbf{A}\times\mathbf{B}}$, *then there exist* $L_1^{\mathbf{A}}, \dots, L_k^{\mathbf{A}} \in nu\mathsf{P}_{\mathbf{A}}$ *and* $L_1^{\mathbf{B}}, \dots, L_k^{\mathbf{B}} \in nu\mathsf{P}_{\mathbf{B}}$ *such that* $L = \bigcup_{i=1}^{k} L_i^{\mathbf{A}} \cap L_i^{\mathbf{B}}$.
- *if* $L_1 \in nu\mathsf{P}_{\mathbf{A}}$ *and* $L_2 \in nu\mathsf{P}_{\mathbf{B}}$
  - $L_1 \cap L_2 \in nu\mathsf{P}_{\mathbf{A}\times\mathbf{B}}$
  - $L_1 \cup L_2 \in nu\mathsf{P}_{\mathbf{A}\times\mathbf{B}}$

The above fact could suggest that the definition of $nu\mathsf{P}_{\mathbf{A}}$ does not work well with the products. However, language classes that we obtain behave well under taking products, despite the fact that the definition of a program seems to be not designed to work well for them.

3 Results

We will use the same name for the class of languages and the family of circuits that it defines. The meaning will be always clear from the context. It turns out that for many natural classes of finite algebras programs over them recognize natural classes of languages considered in circuit complexity. The widest class, we consider in this paper, is $\mathsf{P}/poly$, the class of languages recognized by boolean circuits of polynomial size. NC^2 is a class of languages recognized by circuits of depth $O(log^2 n)$ and with polynomial number of gates, i.e. 2-ary $\wedge, \vee$ and unary $\neg$. The last popular class we consider is CC^0, the class of languages recognized by polynomial size and constant depth circuits consisting of MOD_m gates with unbounded fan-in.

We call a language MONOTONE if it is recognized by polynomial size Boolean monotone circuits. All such languages fulfil Item 1 in Example 2. If we fix an algebra such that all its operations preserve fixed total order, accepting sets S being an upperset with respect to the same order and monotone mapping ι then polynomial size NuDFAs of the form $(\{\mathbf{t}_i\}_{i=1}^{\infty}, \iota, S)$ recognize languages from MONOTONE. BAR_c is a class of boolean circuits with number of inputs bounded by c. We define MULTIMONOTONE as a class of languages defined by the circuits of the form $\mathrm{BAR}_c \circ \mathrm{MONOTONE}$. Note that the class MULTIMONOTONE is closed under taking bitwise complements. It follows from the fact that monotone circuit with negated input is equivalent by De Morgan's laws to negation of another monotone circuit but with non-negated inputs.

Note that one can show that the language $CLIQUE$ requires superpolynomial size MULTIMONOTONE circuits. It is because we can view these circuits as having only bounded number of negations close to the output (hidden in BAR_c). Hence we can apply some classical results [3] to obtain separation. As we later characterize some of our classes $nu\mathsf{P}_{\mathbf{A}}$ as having MULTIMONOTONE circuits, this explicit $CLIQUE$ lower bound transfers to some of our algebras $\mathbf{A}$.

It turns out that for every finite algebra $\mathbf{A}$ from congruence modular variety $nu\mathsf{P}_{\mathbf{A}}$ is equal to $\mathsf{P}/poly$ or is contained in some, perhaps smaller, class of languages.

▶ **Theorem 6.** *Let* $\mathbf{A}$ *be a finite algebra from a congruence modular variety.*
Then, $nu\mathsf{P}_{\mathbf{A}} = \mathsf{P}/poly$ *or* $nu\mathsf{P}_{\mathbf{A}} \subseteq \mathsf{NC}^2 \circ MONOTONE$

In fact, we have a much more precise result than Theorem 6 but we need some preparation to state the main theorem of the paper. To characterize languages recognized by different algebras from congruence modular varieties we will use two important theories of universal

algebra: modular commutator theory (see [13]) and tame congruence theory ([15]). Commutator theory generalizes notion of commutator to the universal algebraic realm, which enables generalization of such group-theoretical notions as abelianity, nilpotency and solvability for arbitrary algebraic structures. Tame congruence theory describes a local behaviour of finite algebras. Such a behaviour is connected with covering pairs of congruences $\alpha \prec \beta$ and so called (α, β)-minimal sets. It turns out that such a minimal set can behave in one of five ways:

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.

Different prime quotients of the same algebra can behave in different ways. The typeset of an algebra $\mathbf{A}$ i.e the set of prime quotients' types which can be found in $\mathbf{A}$ is denoted by $\mathrm{typ}\{\mathbf{A}\}$. Note that typesets of algebras from congruence modular varieties are contained in a set $\{\mathbf{2}, \mathbf{3}, \mathbf{4}\}$ [15, Theorem 8.5]. Moreover in case of congruence modular varieties minimals sets of types $\mathbf{3}$ and $\mathbf{4}$ consist of exactly 2 elements. In case of other types and other varieties the situation is more involved and will be clarified later.

In a congruence modular variety, whenever all prime quotients of $\mathbf{A}$ are of type $\mathbf{2}$ the algebra $\mathbf{A}$ is solvable and also all prime quotients of solvable algebras have only $\mathbf{2}$ quotients [15, 13]. In a congruence modular variety all solvable algebras are Malcev, i.e. they have a ternary term $\mathbf{d}$ satisfying $\mathbf{d}(x, y, y) = \mathbf{d}(y, y, x) = x$.

We say that congruence α of $\mathbf{A}$ is join irreducible if it is a join irreducible element of congruence lattice. It is well known that every join irreducible congruence α has its unique subcover α^-. We define type of join irreducible congruence α as a type of quotient (α^-, α).

Subdirect product of algebras $\mathbf{A}_1, \ldots, \mathbf{A}_k$ is a subalgebra of their direct product such that projection on i-th coordinate is equal to A_i for all i. An algebra is subdirectly irreducible if it cannot be presented as a subdirect product of at least two nontrivial algebras. Every finite algebra $\mathbf{A}$ is isomorphic to a subdirect product of a finite number of subdirectly irreducible finite algebras (which are homomorphic images of $\mathbf{A}$ or in other words which are quotient algebras of $\mathbf{A}$).

We say that an algebra $\mathbf{A}$ is totally ordered if there is total order preserved by all polynomial operations of $\mathbf{A}$.

Now we can state the theorem.

▶ **Theorem 7.** *Let $\mathbf{A}$ be a finite algebra from congruence modular variety. Then:*

1. *If $\mathbf{A}$ is nilpotent then $nu\mathsf{P}_{\mathbf{A}} \subseteq \mathsf{CC}^0$.*
2. *If $\mathrm{typ}\{\mathbf{A}\} = \mathbf{2}$, then $nu\mathsf{P}_{\mathbf{A}} \subseteq \mathsf{NC}^2$.*
3. *If $\mathbf{3} \in \mathrm{typ}\{\mathbf{A}\}$ then $nu\mathsf{P}_{\mathbf{A}} = \mathsf{P}/poly$.*
4. *If $\mathbf{A}$ is a subdirect product of totally ordered algebras then $nu\mathsf{P}_{\mathbf{A}} = MULTIMONOTONE$.*
5. *If there is a congruence α of $\mathbf{A}$ such that $\mathrm{typ}\{\mathbf{A}/\alpha\} = \{\mathbf{4}\}$ and $\mathbf{A}/\alpha$ is a subdirectly irreducible algebra which is not totally ordered then $nu\mathsf{P}_{\mathbf{A}} = \mathsf{P}/poly$.*
6. *If $\mathbf{A}$ has join irreducible congruences $\alpha < \beta$ such that type of α is $\mathbf{4}$ and type of β is $\mathbf{2}$, then $nu\mathsf{P}_{\mathbf{A}} = \mathsf{P}/poly$.*
7. *If none of Items 1-6 holds, then $nu\mathsf{P}_{\mathbf{A}} \subseteq \mathsf{NC}^2 \circ MONOTONE$*

In fact, wherever NC^2 appears in the above theorem we can capture the real complexity class better, by a less natural class $\mathsf{nultDet}$ of languages recognized by evaluating iterated determinant. We will explain the class later in the proper context.

Now we explain how to combine the results of the next sections

Proof of Theorem 7. If $\mathrm{typ}\{A\} = \{\mathbf{2}\}$ by Theorem 27 we have $nu\mathsf{P}_{\mathbf{A}} \subseteq NC^2$. If $\mathbf{A}$ is additionally nilpotent $nu\mathsf{P}_{\mathbf{A}} = CC^0$ by Theorem 31.

If $\mathbf{3} \in \mathrm{typ}\{\mathbf{A}\}$ there is a two element set $U \subseteq A$ on which the algebra $\mathbf{A}$ behaves like a Boolean algebra. So there are operations $\wedge_U, \vee_U, \neg_U$ in the polynomial clone of $\mathbf{A}$ such that $(U, \wedge_U, \vee_U, \neg_U)$ is isomorphic to the two element Boolean algebra. Thus we can use ι to project to this two elements and encode arbitrary Boolean circuits by composing $\wedge_U, \vee_U, \neg_U$. Hence $nu\mathsf{P}_{\mathbf{A}} = \mathsf{P}/poly$.

If $\mathbf{A}$ is a subdirect product of totally ordered algebras, call them $\mathbf{B}_1, \ldots, \mathbf{B}_h$, every program $(\mathbf{p})[\iota, S]$ over $\mathbf{A}$ can be simulated by a system of constant number of programs of the form $\left(\mathbf{p}^{\mathbf{B}_i}\right)\left[\iota^{\mathbf{B}_i}, \{c^{\mathbf{B}_i}\}\right]$, $\iota^{\mathbf{B}_i}$ is just ι projected to the coordinate i and $c^{\mathbf{B}_i} \in B_i$. But by Lemma 15 the function computed by a program over each $\mathbf{B}_i$ can be also computed by MULTIMONOTONE circuit of size polynomial in the size of the program. There is a bounded arity function $\mathbf{d}$ which takes these programs from coordinates and reconstructs the answer for $(\mathbf{p})[\iota, S]$. Since MULTIMONOTONE is closed under composing with bounded arity functions from the outside we see that $nu\mathsf{P}_{\mathbf{A}} = \mathrm{MULTIMONOTONE}$.

If there is a congruence α such as in (5) by Lemma 16 we have $nu\mathsf{P}_{\mathbf{A}} = \mathsf{P}/poly$. If $\mathbf{A}$ has congruences α, β as in (6) then by Theorem 34 we have $nu\mathsf{P}_{\mathbf{A}} = \mathsf{P}/poly$.

If none of the items 1-6 hold, then $\mathrm{typ}\{A\} = \{\mathbf{2}, \mathbf{4}\}$. Since Item 6 does not hold by Lemma 35 there is a congruence β such that $\mathrm{typ}\{\mathbf{A}/\beta\} = \{\mathbf{4}\}$ and all prime quotients below β are of type $\mathbf{2}$. Now $\mathbf{A}/\beta$ has to decompose into a subdirect product of totally ordered algebras, otherwise Item 5 would hold for some subdirectly irreducible quotient $(\mathbf{A}/\beta)/\alpha$ of $\mathbf{A}/\beta$ which is also a quotient of $\mathbf{A}$. Then by Theorem 36 $nu\mathsf{P}_{\mathbf{A}} \subseteq \mathsf{NC}^2 \circ \mathrm{MONOTONE}$ which finishes the proof. ◀

4 Background material

4.1 Universal algebra

We use a standard universal algebraic notation (see [33]). Moreover, we use two important theories of universal algebra and their notions: tame congruence theory (TCT, see [15]) and modular commutator theory (see [13]).

We use boldface capital letters to denote algebras and the same letters but using regular font to denote their universes. Congruences of an algebra $\mathbf{A}$ form a complete lattice $(\mathsf{Con}\,\mathbf{A})$ with the smallest element $0_{\mathbf{A}}$ and the biggest element $1_{\mathbf{A}}$, where $0_{\mathbf{A}}$ is the equality relation and $1_{\mathbf{A}}$ is an equivalence relation with one equivalence class. An algebra which has no non-trivial congrunces, i.e. different than $0_{\mathbf{A}}$ and $1_{\mathbf{A}}$, is called simple. Minimal non-trivial congruences i.e. congruences which cover $0_{\mathbf{A}}$ are called atoms. It is well known that an algebra is subdirectly irreducible iff it has exactly one atom. This unique atom is called monolith. By polynomials over an algebra we mean every proper expression built of basic operations of the algebra, constants and variables. Polynomials define polynomial operations in an obvious way. Two algebras are polynomially equivalent if they have the same set of polynomial operations (or more precisely if one algebra is isomorphic to an algebra which has the same set of polynomial operations as the second algebra). For a subset $S \subseteq A$ an algebra induced on S is the algebra $(\mathbf{A}|_S)$ with universe S and the set of basic operations consisting of all polynomial operations of $\mathbf{A}$ which for values from S return elements from S.

We say that β covers α, and write $\alpha \prec \beta$, if $\alpha < \beta$ and there is no $\gamma \in \mathrm{Con}\,\mathbf{A}$ such that $\alpha < \gamma < \beta$. For a given covering pair of congruences $\alpha \prec \beta$ of $\mathbf{A}$, an (α, β)-minimal set is a minimal with respect to inclusion, image of unary polynomial which do not collapse β

to α. It is known that algebras induced on every two (α, β)-minimal sets are polynomially equivalent. For a (α, β)-minimal set U and $a \in U$, $N = a/\beta \cap U$ such that $N \neq a/\alpha \cap U$ is called a trace. The union of all traces of U is a body and a tail of U consists of all elements of U which are not contained in body i.e. $a \in U$ such that $a/\beta \cap U = a/\alpha \cap U$. For every minimal set U there is polynomial $\mathbf{e}_U$ such that $\mathbf{e}_U(U) = U$ and $\mathbf{e}_U(\mathbf{e}_U(x)) = \mathbf{e}_U(x)$ for all $x \in A$. One of the nice properties of algebras from congruence modular variety is the fact that its minimal sets have no tails [15, Theorem 8.5]. Every trace of (α, β)-minimal set is, modulo α, polynomially equivalent to the one of five types of local behaviour mentioned in Section 3. Every trace of the one minimal set is of the same type, and hence it makes sense to talk about the minimal set type. Minimal sets of types **3**, **4** end **5** have exactly one trace. In case of type **2** traces are polynomially equivalent to one-dimensional vector space and their sizes are of prime power order. This prime, the same for every trace in the one minimal set, we call the characteristic of the minimal set of type **2**.

Let $k \in \mathbb{N}$ and $\mathbf{M}$ be an algebra. Then the matrix power $\mathbf{M}^{[k]}$ is defined in a standard way, i.e. the domain is just M^k and operations on each component are just operations of $\mathbf{M}$ but taking all variables from all components. That is if $m \in M^k$, and m^i denotes the i-th component of m, the algebra $\mathbf{M}^k$ contains all operations of the form $\mathbf{p}(x_1, \ldots, x_n) = (\mathbf{p}_1(x_1^1, \ldots, x_n^k), \ldots, \mathbf{p}_k(x_1^1, \ldots, x_n^k))$ where $\mathbf{p}_1, \ldots, \mathbf{p}_k$ are some $n \cdot k$-ary polynomials of $\mathbf{M}$. Matrix powers of minimal sets play central role in our investigation. Actually, our results rely on the fact that every finite algebra from a congruence modular variety can be expressed as some type of a product of its minimal sets matrix powers. To be able to state this fact properly we need to introduce the new type of a product.

▶ **Definition 8.** *Let $\mathbf{A}, \mathbf{B}_1, \ldots, \mathbf{B}_h$ be finite algebras. We say that an algebra $\mathbf{A}$ is an action product $(\mathbf{B}_1 \boxtimes (\mathbf{B}_2 \boxtimes (\ldots \boxtimes \mathbf{B}_h) \ldots)$ whenever the domain $A = B_1 \times B_2 \times \ldots \times B_h$ and for each n-ary $\mathbf{f} \in \operatorname{Pol} \mathbf{A}$ there are $\mathbf{f}_1 \ldots, \mathbf{f}_h$, such that $\mathbf{f}_i : (B_i \times \ldots \times B_h)^n \longrightarrow B_i$ and*

$$\mathbf{f}(x) = (\mathbf{f}_1(x^{(1)}), \ldots, \mathbf{f}_h(x^{(h)})),$$

where $x^{(i)} = (x_1^i, \ldots, x_n^i, \ldots, x_1^h, \ldots, x_n^h) \in (B_i \times \ldots \times B_h)^n$. Moreover whenever we assign a constant value $c \in (B_{i+1} \times \ldots \times B_h)^n$ to $x^{(i+1)}$, the function

$$\mathbf{f}_i(x_1^i, \ldots, x_n^i, c_1^{i+1}, \ldots, c_n^{i+1}, \ldots, c_1^h, \ldots, c_n^h)$$

is an n-ary polynomial of the algebra $\mathbf{B}_i$.

The above definition may seem to be complicated but the main idea is quite simple. Algebras appearing in the product later act on their predecessor as it is in semidirect product of groups.

4.2 Algebraic Branching Programs

To understand the behaviour of circuits over solvable algebras, we need the notion of *algebraic branching program*. We take our basic definitions from [12]. An algebraic branching program is a way to represent a formal polynomial over a field by an edge-labeled directed acyclic graph in which multiple edges between vertices are allowed. There is always one source vertex, and one sink vertex in the graph, and edges are labeled by either a variable or a constant of the field $\mathbb{F}$. We will consider only ABPs over finite fields $\mathsf{GF}(p)$ where p is a prime. Each path from source to sink represents a monomial, created by multiplying variables/constants occurring on its edges. Then we say that ABP represents/computes a polynomial $\mathbf{w}$ equal to the sum of monomials on all such paths.

The size of such ABP is defined as the number of its vertices plus the number of the edges (but edges linearly dominate the size). Another way to define the size which is more useful in our context is by the following process, that we later call ABP-process. We start with a single identity 1 polynomial. Then we create new polynomials by applying one of the two steps:

1. take an already created polynomial and multiply it by a constant or a variable
2. add n already created polynomials.

In this way we can define the size as the minimal number of steps needed to create a polynomial. One can see that these two notions of size are linearly-dependent on each other, i.e. if some ABP represents a polynomial $\mathbf{f}$ using l edges then there is $O(l)$ step process defining $\mathbf{w}$, and also from each l-step process we get ABP of size $O(l)$ in an obvious way. Indeed, the multiplication step corresponds to adding edge to a DAG and addition corresponds to introducing new vertex in the DAG. As we are only interested in asymptotic sizes, we use this two notions of size interchangeably. If a polynomial has ABP of size $O(l)$, we will sometimes say that $\mathbf{w}$ is of ABP complexity $O(l)$.

Determinantal complexity of a polynomial $\mathbf{w}$ over field $\mathbb{F}$ is the smallest size of a matrix filled with variables and constants from $\mathbb{F}$ such that $\mathbf{w}$ is (as a polynomial) equal to determinant of the matrix. It is known, that every polynomial can be represented as determinant of such matrix. Moreover the determinantal complexity of polynomials represented by small arithmetic formulas is also relatively small [38]. We will use the following deep result.

▶ **Theorem 9** ([38, 25, 8, 32, 24]). *Determinantal complexity of a polynomial and the smallest size of ABP representing it are polynomially related.*

We will also use the fact that we can make affine substitutions for variables in a given ABP and it does not change its size too much. We note that many authors define ABP so that in the DAG defining it one can put affine combinations on edges instead of just variables and constants, but for our definition the following fact allows us to manage affine substitutions.

▶ **Fact 10.** *Let $\mathbf{w}$ be a polynomial over n variables represented by an ABP of size smaller than l. Let $A_1, A_2, \ldots, A_n$ be affine combinations of these n variables. Then the polynomial $\mathbf{w}(A_1, \ldots, A_n)$ has ABP of size $O(nl)$. Additionally if A_i depend only on one variable the size is $O(l)$.*

Proof. After such affine substitution label on each edge becomes an affine combination of variables $\alpha_1 \cdot x_1 + \ldots + \alpha_n x_n + c$. Its enough to replace this edge by an ABP computing this affine combination, i.e. put a source of the affine ABP in the beginning of the edge and sink of the ABP in the end of the edge. This ABP has size bounded by $O(n)$ and its $O(1)$ when such affine combination has one variable. ◀

Another fact we will use is that we can multiply polynomials represented by ABPs without expanding the size of the representation too much.

▶ **Fact 11.** *Let $\mathbf{w}_1, \mathbf{w}_2$ be two polynomials represented by ABPs of size at most l_1 and l_2 respectively. Then the polynomial $\mathbf{w}_1 \cdot \mathbf{w}_2$ has ABP of size at most $l_1 + l_2$.*

Proof. In the DAG definition of ABP, we indentify the source of ABP for $\mathbf{w}_2$ with a sink of ABP for $\mathbf{w}_1$ and we get an ABP for $\mathbf{w}_1 \cdot \mathbf{w}_2$. ◀

4.3 Circuit Combinatorics

In this paper we will consider bounded depth Boolean circuit families with gates of unbounded fan-in. As we consider only decision problems, we will always have only one output gate. As opposed to usual notions in circuit complexity, gates of our circuits will have different sizes, as they will sometimes represent complex computational objects. By default gates have size one unless specified directly. We measure size of the circuit as the total number of its edges plus the sum of the sizes of its gates.

A gate of type MOD_m is labelled with an accepting set $S \subseteq \mathbb{Z}_m$. It takes an arbitrary number of Boolean inputs and sums them modulo m. The gate returns 1 iff the sum belongs to S. This gate is always of size one. The $CC_h[m]$ circuit is a circuit which has h layers with MOD_m gate on each layer.

A gate AND_d computes arity d conjunction of its input wires. We will mention the complexity class $\mathsf{AC}_h[m]$. These are languages recognized by circuits of height h using gates AND, OR, MOD_m of unbounded fan-in and the $\neg$ gate.

A gate of type $\mathrm{PROG}_{\mathbf{A}}$ is labeled with an n-ary program $(\mathbf{p}, \iota, S)$ over $\mathbf{A}$. It takes boolean inputs $b_1, \ldots, b_n$ and returns $(\mathbf{p})[\iota, S]\,(b_1, \ldots, b_n)$. The size of such gate is the size of the circuit defining $\mathbf{p}$ in $\mathbf{A}$. A gate of type GABP_p is labelled with n-variate ABP over $GF(p)$ and the accepting set $S \subseteq GF(p)$. It takes n Boolean inputs and returns one iff evaluation of the polynomial corresponding to this ABP on these inputs belongs to set S. The size of such gate is just the size of the ABP.

We will do inductive proofs on both the structure of the algebra $\mathbf{A}$ as well as the definition of the circuits/polynomials defined over $\mathbf{A}$, for this reason we introduce the following notation.

▶ **Definition 12.** *For a circuit $\mathbf{p}$ over an algebra $\mathbf{A}$ and its gate G we write $\mathbf{p}_G$ for a unique subcircuit of $\mathbf{p}$ which consists output gate G and all its ancestors in $\mathbf{p}$. By $U(G)$ we mean the set of all ancestors of G in circuit $\mathbf{p}$.*

We write $\mathbf{f}_G$ for the basic operation computed by the gate G.

5 Local lattice behaviour

In this section we consider languages recognized by NuDFAs over algebras with typeset equal to $\{\mathbf{4}\}$. First we define binary relation which plays a central role in our proofs. Similar ideas can be found in [15].

▶ **Definition 13.** *Let $\mathbf{A}$ be a finite algebra, let $\alpha \prec \beta$ be congruences of $\mathbf{A}$, and let $U = \{0_U, 1_U\}$ be a (α, β)-minimal set of type $\mathbf{4}$ with no tail. Let $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k$ be all unary polynomial functions of $\mathbf{A}$ whose ranges are equal to U.*

$$\leq_U \stackrel{\text{def}}{=} \{(a, b) \in \beta : \mathbf{e}_i(a) \leq \mathbf{e}_i(b) \text{ for } i = 1, \ldots, k\}.$$

It turns out that $\leq_U$ is a preorder preserved by all operations of the algebra. Moreover, if U is minimal with respect to $(0_{\mathbf{A}}, \alpha)$, for some atom α, then $\leq_U$ is an order on α-classes. Following a standard convention in universal algebra, we use the term *order* when referring to *partial order* (in contrast to total/linear order).

▶ **Lemma 14.** *Let $\mathbf{A}$ be a finite algebra, let $\alpha \prec \beta$ be congruences of $\mathbf{A}$, and let U be a (α, β)-minimal set of type $\mathbf{4}$ with no tails. Then:*

1. *$\leq_U$ is a preorder;*
2. *$\leq_U / \alpha$ when restricted to any β/α-coset of $\mathbf{A}/\alpha$ is an order;*
3. *every polynomial of $\mathbf{A}$ preserves $\leq_U$.*

Proof. Let $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k$ be the unary polynomials of $\mathbf{A}$ with range equal to U. By definition, $\leq_U$ inherits reflexivity and transitivity from the coordinatewise order on 2^k, and hence $\leq_U$ is a preorder.

By [15, Theorem 2.8(4)], for every pair $(a, b) \in \beta \setminus \alpha$ there exists a unary polynomial $\mathbf{f}$ such that $\mathbf{f}(A) = U$ and $(\mathbf{f}(a), \mathbf{f}(b)) \in \beta \setminus \alpha$. Hence the mapping

$$x \mapsto (\mathbf{e}_1(x), \ldots, \mathbf{e}_k(x)),$$

for fixed a, is an embedding of the β/α-coset of a/α into 2^k. This proves Item 2.

To prove Item 3, assume for contradiction that there exists an n-ary polynomial $\mathbf{p}$ of $\mathbf{A}$ and tuples $\overline{a}, \overline{b} \in A^n$ such that $a_i \leq_U b_i$ for all i, but

$$\mathbf{p}(a_1, \ldots, a_n) \not\leq_U \mathbf{p}(b_1, \ldots, b_n).$$

Replacing, step by step, a_i with b_i, we obtain at some point $\overline{a'} = (b_1, \ldots, b_{i-1}, a_i, a_{i+1} \ldots, a_n)$ and $\overline{b'} = (b_1, \ldots, b_{i-1}, b_i, a_{i+1} \ldots, a_n)$ such that $\mathbf{p}(\overline{a'}) \not\leq_U \mathbf{p}(\overline{b'})$. Hence, for the unary polynomial $\mathbf{g}(x) = \mathbf{p}(b_1, \ldots, b_{i-1}, x, a_{i+1} \ldots, a_n)$ we have that $a_i \leq_U b_i$ while $\mathbf{g}(a_i) \not\leq_U \mathbf{g}(b_i)$. Then there exists j such that

$$\mathbf{e}_j(\mathbf{g}(a_i)) > \mathbf{e}_j(\mathbf{g}(b_i)).$$

Since $\mathbf{e}_j \circ \mathbf{g}$ is a polynomial with range U, there exists l such that $\mathbf{e}_l = e_j \circ \mathbf{g}$. Consequently,

$$\mathbf{e}_l(a_i) > \mathbf{e}_l(b_i),$$

contradicting $a_i \leq_U b_i$. ◀

The next lemma demonstrates that a total order preserved by all operations of $\mathbf{A}$ restricts languages that can be recognized by circuits over $\mathbf{A}$. In particular the lemma can be applied whenever $\leq_U$ is a total order for some minimal set U. Note that application of Lemma 15 is not restricted to algebras of type $\{\mathbf{4}\}$ (see Example 38).

▶ **Lemma 15.** *Let $\mathbf{A}$ be a totally ordered finite algebra. Then,*

$$\mathsf{nuP}_{\mathbf{A}} \subseteq \mathit{MULTIMONOTONE}.$$

Proof. Assume without loss of generality that $A \subseteq \{0,1\}^k$ for some k and that total order on A is just the coordinatewise order on $\{0,1\}^k$ (i.e. A is identified with a totally ordered subset of $\{0,1\}^k$). Let π_i be a function which project element of A onto its i-th coordinate. Note that since $\mathbf{A}$ is totally ordered, for every basic operation of $\mathbf{A}$ its projection on any coordinate can be expressed as a circuit over $(2, \wedge, \vee)$ which takes arguments from coordinates as inputs. Hence, projection of every polynomial size circuit of $\mathbf{A}$ can be expressed as a polynomial size circuit over two element lattice. The same is true for characteristic function of every upper set of A except the size which, in this case, is bounded by a constant.

First, we will show that polynomial size NuDFA $(\{\mathbf{t}_i\}_{i=0}^{\infty}, \iota, S)$ with monotone ι and S being upper set of A recognize language contained in MONOTONE. In such a case $\iota(0) \leq \iota(1)$ and hence $\pi_i(\iota(0)) \leq \pi_i(\iota(1))$ for all i. Hence, for all i either $\pi_i(\iota(0)) = \pi_i(\iota(1))$ or $\pi_i(\iota(0)) = 0$ and $\pi_i(\iota(1)) = 1$. Therefore, for every i, $\pi_i(\mathbf{t}_n(\iota(b_1), \ldots, \iota(b_n)))$ can be expressed as a polynomial size circuit over two element lattice on inputs b_i. As a consequence, $\mathbf{t}_n[\iota, S](\overline{b})$ is a composition of monotone circuits computing $\pi_i(\mathbf{t}_n(\iota(b_1), \ldots, \iota(b_n)))$, for every i, and the characteristic function of S.

Now, we will show that ever language recognized by NuDFA over $\mathbf{A}$ is contained in MULTIMONOTONE. Note, that for ever $S \subseteq A$ it holds that $S = \bigcup_{s\in S}(s\uparrow \setminus \bigcup_{a>s} a\uparrow)$, where $s\uparrow$ is the upper set $s\uparrow = \{a \in A : a \geq s\}$. Hence, since every set is a result of finitely many set operations on upper sets, the characteristic function of a language recognized by polynomial size NuDFA over $\mathbf{A}$ with monotone function ι and arbitrary set S can be computed by composition of monotone circuits and bounded arity Boolean circuit i.e $\mathrm{BAR}_c \circ \mathrm{MONOTONE} \subseteq \mathrm{MULTIMONOTONE}$. Finally, observe that if ι is not monotone, then $\iota'(x) = \iota(\neg x)$ is, and polynomial size NuDFA $\mathbf{D}' = (\{\mathbf{t}\}_{i=1}^{\infty}, \iota', S)$ recognizes language contained in MULTIMONOTONE. Since, language recognized by $\mathbf{D} = (\{\mathbf{t}\}_{i=1}^{\infty}, \iota', S)$ is just a bitwise complement of language recognized by $\mathbf{D}'$ and MULTIMONOTONE is closed under taking bitwise complement, it follows that language recognized by $\mathbf{D}$ is contained in MULTIMONOTONE. This observation completes the proof. ◀

The following lemma proves Item (5) in Theorem 7.

▶ **Lemma 16.** *Let $\mathbf{A}$ be a subdirectly irreducible finite algebra with $\mathrm{typ}\{\mathbf{A}\} = \{\mathbf{4}\}$ with monolith μ such that $(0_{\mathbf{A}}, \mu)$-minimal set U has no tail. If $\leq_U$ is not a total order then $\mathsf{nuP}_{\mathbf{A}} = P/poly$.*

Proof. Let $U = \{0_U, 1_U\}$ and let $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k$ be the unary polynomials of $\mathbf{A}$ with range equal to U. Assume without loss of generality that $\mathbf{e}_1$ is an idempotent operation. First note that for every $a, b \in A$ we have $\mu \subseteq \Theta(a, b)$. Hence there exists a sequence $\{\mathbf{f}_1, \ldots, \mathbf{f}_l\} \subseteq \mathrm{Pol}_1 \mathbf{A}$ such that

$$0_U \in \{\mathbf{f}_1(a), \mathbf{f}_1(b)\}, \quad 1_U \in \{\mathbf{f}_l(a), \mathbf{f}_l(b)\},$$

and

$$\{\mathbf{f}_i(a), \mathbf{f}_i(b)\} \cap \{\mathbf{f}_{i+1}(a), \mathbf{f}_{i+1}(b)\} \neq \emptyset$$

for all $i = 1, \ldots, l-1$.

Thus there exists i such that $\mathbf{e}_1(\mathbf{f}_i(a)) \neq \mathbf{e}_1(\mathbf{f}_i(b))$, and consequently there exists j such that $\mathbf{e}_j = \mathbf{e}_1 \circ \mathbf{f}_i$. It follows that for all distinct $a, b \in A$,

$$(\mathbf{e}_1(a), \ldots, \mathbf{e}_k(a)) \neq (\mathbf{e}_1(b), \ldots, \mathbf{e}_k(b)),$$

which gives us a natural embedding $A \mapsto \{0_U, 1_U\}^k$ and $\leq_U$ is just a coordinate-wise order inherited from $\{0_U, 1_U\}^k$.

Assume now that there exist $a, b \in A$ such that neither $a \leq_U b$ nor $b \leq_U a$. Then there exist i, j such that

$$\mathbf{e}_i(a) < \mathbf{e}_i(b) \quad \text{and} \quad \mathbf{e}_j(a) > \mathbf{e}_j(b).$$

Let $C(b_1, \ldots, b_n)$ be a Boolean circuit. By De Morgan's laws, we may assume that all negations occur only at the input level. Replace the Boolean operations $\wedge, \vee$ by the operations $\wedge_U, \vee_U$ on U to obtain a polynomial $\mathbf{p}$ of $\mathbf{A}$. If the m-th input is negated, replace it by $\mathbf{e}_j(x_m)$; otherwise replace it by $\mathbf{e}_i(x_m)$. Let the projection ι be such that $\iota(0) = a$ and $\iota(1) = b$, and let the accepting set be $S = \{1_U\}$.

The resulting program computes the same Boolean function as C, and its size is linear in the size of C. Hence $nu\mathsf{P}_{\mathbf{A}} = \mathsf{P}/poly$. ◀

Combining Lemma 15 and Lemma 16 we get the following.

▶ **Corollary 17.** *Let* $\mathbf{A}$ *be a subdirectly irreducible finite algebra with* $\mathrm{typ}\{\mathbf{A}\} = \{\mathbf{4}\}$ *and monolith* μ *such that* $(0_{\mathbf{A}}, \mu)$*-minimal sets have no tails. If there exists a total order preserved by all operations of* $\mathbf{A}$*, then*

$$\mathsf{nuP}_{\mathbf{A}} \subseteq \mathit{MULTIMONOTONE}.$$

Otherwise,

$$\mathsf{nuP}_{\mathbf{A}} = \mathit{P/poly}.$$

6 The structure of finite algebras

In this section we show a structural result giving description of algebras from congruence modular variety. We strongly believe that these results are of independent interest and can be generalised to work for arbitrary finite algebras.

▶ **Lemma 18.** *Let* $\mathbf{A}$ *be a finite algebra from a congruence modular variety, let* $\alpha \in \mathsf{Con}\,\mathbf{A}$ *be an atom, and let* U *be a* $(0_{\mathbf{A}}, \alpha)$*-minimal set.*

Then there exists k *such that* $\mathbf{A}$ *is isomorphic to a subreduct of* $U^{[k]} \boxtimes \mathbf{A}/\alpha$*.*

Proof. Note that for every $(a, b) \in \alpha$, $a \neq b$ there exists $f \in \mathrm{Pol}_1\,\mathbf{A}$ such that $f(\mathbf{A}) \subseteq U$ and $f(a) \neq f(b)$ ([15, Theorem 2.8(4)]). Let $\{\mathbf{e}_1, \dots, \mathbf{e}_k\}$ be the set of all such unary polynomials. It is easy to see that for $a, b \in A$ we have

$$a = b \iff (\mathbf{e}_1(a), \dots, \mathbf{e}_k(a), a/\alpha) = (\mathbf{e}_1(b), \dots, \mathbf{e}_k(b), b/\alpha).$$

We will show that $\mathbf{A}$ is isomorphic to a subreduct of $U^{[k]} \boxtimes \mathbf{A}/\alpha$ via the mapping

$$h(x) = (\mathbf{e}_1(x), \dots, \mathbf{e}_k(x), x/\alpha).$$

It suffices to show that for every basic n-ary operation $\mathbf{f}^{\mathbf{A}}$ of $\mathbf{A}$ there exists an operation $\mathbf{f}^{U^{[k]} \boxtimes \mathbf{A}/\alpha}$ such that

$$h(\mathbf{f}^{\mathbf{A}}(x_1, \dots, x_n)) = \mathbf{f}^{U^{[k]} \boxtimes \mathbf{A}/\alpha}(h(x_1), \dots, h(x_n)).$$

Fix $a_1, \dots, a_n \in A$. Directly from the definition of matrix power $U^{[k]} \boxtimes \mathbf{A}/\alpha$, it is enough to show that for each i there exists a polynomial $\mathbf{p}_i$ of U such that

$$\mathbf{e}_i(\mathbf{f}(x_1, \dots, x_n)) = \mathbf{p}_i(\mathbf{e}_1(x_1), \dots, \mathbf{e}_1(x_n), \dots, \mathbf{e}_k(x_1), \dots, \mathbf{e}_k(x_n))$$

for all $x_1 \in a_1/\alpha, \dots, x_n \in a_n/\alpha$.

We split the rest of the proof according to the TCT type of U. Since $\mathbf{A}$ belongs to a congruence modular variety, the only possible types of U are **2**, **3**, and **4**.

Type 2. In this case α is abelian. Hence a difference term d defines ternary abelian group operations on α-cosets (see [13, Lemma 5.6]) and commutes with every polynomial of $\mathbf{A}$ on α-related arguments (see [13, Proposition 5.7]). Moreover, the difference term induces a group operation on the traces of U, since

$$\mathbf{d}(\mathbf{e}_U(x), \mathbf{e}_U(y), \mathbf{e}_U(z)) = \mathbf{e}_U(\mathbf{d}(x, y, z))$$

for all $x, y, z \in U$.

Let $\{0_{a/\alpha} : a \in A\}$ be a transversal of α-classes such that $0_{u/\alpha} \in U$ for every $u \in U$. For $x, y \in a/\alpha$ define $x + y = \mathbf{d}(x, 0_{a/\alpha}, y)$. Let $\mathbf{f}$ be an n-ary basic operation of $\mathbf{A}$, fix $a_1, \dots, a_n \in A$, and define

$$\mathbf{g}_i(x_1, \dots, x_n) = \mathbf{e}_i(\mathbf{f}(x_1, \dots, x_n)) - \mathbf{e}_i(\mathbf{f}(0_{a_1/\alpha}, \dots, 0_{a_n/\alpha})).$$

Let $0 = 0_{f(a_1,\dots,a_n)/\alpha}$. Then $g_i(0_{a_1/\alpha}, \dots, 0_{a_n/\alpha}) = 0$.

Observe that for $x_1 \in a_1/\alpha$, …, $x_n \in a_n/\alpha$

$$\begin{aligned}
&\mathbf{g}_i(x_1, x_2, \dots, x_n) = \\
&= \mathbf{g}_i(\mathbf{d}(x_1, 0_{a_1/\alpha}, 0_{a_1/\alpha}), \mathbf{d}(0_{a_2/\alpha}, 0_{a_2/\alpha}, x_2), \dots, \mathbf{d}(0_{a_n/\alpha}, 0_{a_n/\alpha}, x_n)) = \\
&= \mathbf{d}(\mathbf{g}_i(x_1, 0_{a_2/\alpha}, \dots, 0_{a_n/\alpha}), \mathbf{g}_i(0_{a_1/\alpha}, 0_{a_2/\alpha}, \dots, 0_{a_n/\alpha}), \\
&\qquad \mathbf{g}_i(0_{a_1/\alpha}, x_2, \dots, x_n)) = \\
&= \mathbf{g}_i(x_1, 0_{a_2/\alpha}, \dots, 0_{a_n/\alpha}) + \mathbf{g}_i(0_{a_1/\alpha}, x_2, \dots, x_n).
\end{aligned}$$

In this way we obtain that for $x_j \in a_j/\alpha$

$$\mathbf{g}_i(x_1, \dots, x_n) = \sum_{j=1}^{n} \mathbf{g}_i(0_{a_1/\alpha}, \dots, 0_{a_{j-1}/\alpha}, x_j, 0_{a_{j+1}/\alpha}, \dots, 0_{a_n/\alpha}).$$

Each summand is either constant or a unary polynomial of $\mathbf{A}$ with image contained in U, and hence equals $\mathbf{e}_{s_j}(x_j)$ for some index s_j. Consequently,

$$\mathbf{g}_i(x_1, \dots, x_n) = \sum_{j \in J} \mathbf{e}_{s_j}(x_j),$$

and therefore

$$\mathbf{e}_i(\mathbf{f}(x_1, \dots, x_n)) = \mathbf{p}_i(e_1(x_1), \dots, e_k(x_n))$$

for some polynomial $\mathbf{p}_i$ of U.

Type 3. In this case the claim is immediate, since every function on a two-element set is a polynomial of U.

Type 4. Note that, since α is an atom, $\leq_U$, by Lemma 14, is an order on α-cosets. Moreover, for any n-ary polynomial $\mathbf{f}$ of $\mathbf{A}$ and fixed $a_1, \dots, a_n \in A$, the restriction

$$\mathbf{f}|_{a_1/\alpha \times \dots \times a_n/\alpha}$$

is monotone with respect to $\leq_U$. Since, every monotone Boolean function is a lattice polynomial, each function $\mathbf{e}_i \circ \mathbf{f}$ restricted to these cosets can be expressed as a lattice polynomial in the variables $\mathbf{e}_j(x_l)$, for $j = 1, \dots, k$ and $l = 1, \dots, n$. This completes the proof. ◀

In the case of atom of type **2** we can give even more precise description.

▶ **Lemma 19.** *Let $\mathbf{A}$ be a finite algebra from a congruence modular variety, let $\alpha \in \mathsf{Con}\,\mathbf{A}$ be an atom, and let U be a $(0_{\mathbf{A}}, \alpha)$-minimal set of type* **2**.

Then there exists k such that $\mathbf{A}$ is isomorphic to a subreduct of $\mathbb{Z}_p^{[k]} \boxtimes \mathbf{A}/\alpha$, where p is a prime divisor of $|U|$.

We omit a straightforward proof of Lemma 19. Let us only remark that the proof can be divided into two steps. First we can show that instead of projections into minimal set, used in the proof of Lemma 18, we can project elements of the algebra into one of the minimal set traces. Then, it is enough to show that every trace is subreduct of matrix power of $\mathbb{Z}_p$.

7 Solvable algebras

Now we focus on finite algebras $\mathbf{A}$ which are solvable, i.e. they satisfy $\mathrm{typ}\{\mathbf{A}\} \subseteq \{2\}$. We are going to characterize $nu\mathsf{P}_{\mathbf{A}}$ with the use of algebraic branching programs. It is quite unexpected, that Barrington used nonsolvable groups to demonstrate the power of (Boolean) branching programs, and here we use algebraic branching programs to bound the power of solvable algebras and their circuits.

The following lemma follows from analysis of the proof of Lemma 9.6 in [17].

▶ **Lemma 20.** *Let $\mathbf{A}$ be a finite simple algebra of affine type of characteristic p. Let $(\mathbf{p})[\iota, S]$ be a program over $\mathbf{A}$ of length l. Then there is an integer k depending only on $\mathbf{A}$ and a $BAR_k \circ MOD_p$ circuit of size $O(l)$ computing the same function as $\mathbf{p}$ does.*

To precisely characterize $nu\mathsf{P}_{\mathbf{A}}$ for nonsimple $\mathbf{A}$ we introduce the following complexity classes.

▶ **Definition 21.** *We say that a language L is in the class $\mathsf{nuDet}_{p_1} \circ \ldots \circ \mathsf{nuDet}_{p_h}$ whenever it is recognized by a sequence of polynomial size $GABP_{p_1} \circ \ldots \circ GABP_{p_h}$ circuits. In particular nuDet_p is a set of languages recognized by $GABP_p$ gates of the size polynomial in the input legth.*

$\mathsf{nultDet}$ is the set sum of classes $\mathsf{nuDet}_{p_1} \circ \ldots \mathsf{nuDet}_{p_h}$ for all sequences of primes $p_1, \ldots, p_h$.

Notice that for a polynomial $\mathbf{w}$ of ABP complexity l from Theorem 9 we can replace evaluation of expression $\mathbf{w}(\bar{b}) \in S$ by computing determinant of a matrix of size $O(\mathrm{poly}(l))$ and checking if a result is in S. It justifies the name of the class nuDet_p as the GABP_p gates just evaluate such polynomials. Moreover, computing such a determinant can be famously done uniformly in the class NC^2 [8]. Since we have different gates for different input lengths the resulting class is nonuniform.

▶ **Fact 22.** $\mathsf{nuDet}_p \subseteq$ *(nonuniform) NC^2.*

On the other hand, every n-ary formula in NC^1 of size $\mathrm{poly}(n)$ can be converted to arithmetic formula over $GF(p)$ of height $O(\log n)$, then expanded to arithmetic expression of size $O(\mathrm{poly}(n))$ and then written as a determinant of a $O(\mathrm{poly}(n))$ size matrix by [38]. Hence, by Theorem 9 $NC^1 \subseteq \mathsf{nuDet}_p$.

In our proofs we will use the following

▶ **Fact 23.** *Let p be a prime and k be a fixed constant. Let $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_k$ be n-ary polynomials over the field $GF(p)$ which can be represented by ABPs of size $O(l)$. Let $\mathbf{d}$ be a k-ary function of type $GF(p)^k \longrightarrow GF(p)$. Then the function $GF(p)^n \longrightarrow GF(p)$ induced by expression $\mathbf{d}(\mathbf{w}_1, \ldots, \mathbf{w}_k)$ can be represented by a polynomial that has ABP of size $O(l)$.*

Proof. Function $\mathbf{d}$ is of bounded arity, so it can be represented by a sparse polynomial over $GF(p)$ which in its definition requires some constant number of additions and multiplications. So by consecutively applying Fact 11 to every multiplication in this polynomial, the composition $\mathbf{d}(\mathbf{w}_1, \ldots, \mathbf{w}_n)$ can be represented by a polynomial of size $O(l)$. ◀

It turns out that the class nuDet_p can be exactly captured as $nu\mathsf{P}_A$ for some solvable algebra $\mathbf{A}$.

▶ **Theorem 24.** *For each prime p there is a solvable Malcev algebra $\mathbf{A}$ such that polynomial size programs over $\mathbf{A}$ recognize exactly languages from nuDet_p.*

Proof sketch. Let the underlying set of $\mathbf{A}$ be $Z_p \times Z_p$ and let $\mathbf{A}$ have 5 operations.

1. binary abelian + opearion of a group $\mathbb{Z}_p \times \mathbb{Z}_p$
2. unary projections π_1, π_2 satisfying
   $\pi_1(x,y) = (x,0), \pi_2(x,y) = (0,y)$
3. unary $u(x,y) = (y,0)$
4. binary quasiprojection $v((x_1,y_1),(x_2,y_2)) = (x_1 \cdot y_2, 0)$

It is automatic to check with the definitions that this algebra is Malcev and solvable. Moreover one can check by a simple inductive argument that every polynomial $\mathbf{p}(x_1,\ldots,x_n)$ of $\mathbf{A}$ of length l is of the form $(\mathbf{w}(x_1^1,\ldots,x_n^1,x_1^2,\ldots,x_n^2), \mathbf{a}(x_1^2,\ldots,x_n^2))$, where $\mathbf{w}(x_1^1,\ldots,x_n^1,x_1^2,\ldots,x_n^2)$ is a polynomial over $GF(p)$ with ABP complexity $O(l)$ and $\mathbf{a}$ is an affine function over variables x_i^2. Indeed all what the basic operations on the first coordinate allow is just addition and multiplication by an affine function from the second coordinate. The multiplication by an affine function can be simulated by $O(n)$ ABP steps. On the other hand any polynomial $\mathbf{w}(x_1^2,\ldots,x_n^2)$ over $GF(p)$ with some bounded ABP complexity can be obtained on the first coordinate by applying polynomially many basic operations of $\mathbf{A}$. This two facts together would give us $nu\mathsf{P}_{\mathbf{A}} = \mathsf{nuDet}_p$, the problem is that while rewriting program to polynomial over $GF(p)$ we have to deal with ι, S. To handle/eliminate ι we interpolate it with an affine function and apply Fact 10. To emulate S we apply Fact 23. Another issue is that $\mathbf{A}$ has two coordinates, but we can also conjunct them by an application of Fact 23. Rewriting polynomial over $GF(p)$ to a proper program of our algebra is straightforward. ◀

Now we show how to inductively upperbound the complexity class $nu\mathsf{P}_A$, when the induction is on the height of congruence lattice of $\mathbf{A}$.

▶ **Lemma 25.** *Let $\mathbf{A}$ be a finite algebra from a congruence modular variety and α be an atom of $\mathbf{A}$ of type 2 and characteristic p. Then any function computed by a program over $\mathbf{A}$ of size l can be also computed by circuit of type $GABP_p \circ PROG_{\mathbf{A}/\alpha}$ of size $O(poly(l))$.*

Proof. Let $\mathbf{C} = \mathbf{A}/\alpha$. By Lemma 19 the algebra $\mathbf{A}$ is isomorphic to a subreduct of $\mathbb{Z}_p^{[k]} \boxtimes \mathbf{C}$. Without loss of generality we assume that $\mathbf{A}$ is a reduct of $\mathbb{Z}_p^{[k]} \boxtimes \mathbf{C}$ as taking subalgebras decreases the power of the programs. For every $a \in A$ we write $a^{\mathbf{C}}$ for its coordinate corresponding to $\mathbf{C}$, and for $s \in \{1,\ldots,k\}$ we write a^s for its coordinate corresponding to the s-th component of $\mathbb{Z}_p^k$.

Let $\mathbf{f}(x_1,\ldots,x_r)$ be a basic operation of $\mathbf{A}$. Then by the definition of matrix product $\mathbf{f}(x_1,\ldots,x_r)^s$ depends on the $r + r \cdot k$ values

$$x_1^{\mathbf{C}},\ldots,x_r^{\mathbf{C}},x_1^1,\ldots,x_r^k$$

in such a way that after fixing $x_1^{\mathbf{C}},\ldots,x_r^{\mathbf{C}}$ it becomes a polynomial of $(Z_p,+)$, that is an affine function. It means that

$$(\mathbf{f}(x_1,\ldots,x_r))^s = \sum_{c_1,..,c_r \in \mathbf{C}} \chi_{c_1}(x_1^{\mathbf{C}}) \cdot \ldots \cdot \chi_{c_r}(x_r^{\mathbf{C}}) \cdot A_{\overline{c}}(x_1^1,..,x_r^k)$$

where $\chi_c(x)$ is a unary characteristic function $C \longrightarrow Z_p$ which is equal to 1 when $x = c$ and is equal to 0 otherwise, and each $A_{\overline{c}}$ is an affine function coming from substituting $c_1,..,c_r$ for the respective variables $x_1^C,\ldots,x_r^C$. But then we can regroup formula according to the variables x_i^j to get

$$(\mathbf{f}(x_1,\dots,x_r))^s = \sum_{i=1..r}\sum_{j=1..k} x_i^j \cdot \mathbf{w}_{i,j}(\chi_{c_1}(x_1^{\mathbf{C}}),\chi_{c_2}(x_1^{\mathbf{C}}),..,\chi_{c_m}(x_r^{\mathbf{C}})) + \mathbf{w}(\chi_{c_1}(x_1^{\mathbf{C}}),\chi_{c_2}(x_1^{\mathbf{C}}),..,\chi_{c_m}(x_r^{\mathbf{C}}))$$

where $c_1,\dots c_m$ enumerate all elements of $\mathbf{C}$ and each $\mathbf{w}_{i,j}$ as well as $\mathbf{w}$ is a rm-ary degree r polynomials over $GF(p)$.

Now take arbitrary n-ary polynomial/circuit $\mathbf{p}(x_1,\dots,x_n)$ of $\mathbf{A}$ which is built of l gates. From now, without loss of generality, we will treat l as the size of $\mathbf{p}$, because it is smaller than the actual size of $\mathbf{p}$ (it does not account for the edges). We will prove by induction on the definition of $\mathbf{p}$ that there exists $nk+ml$-ary polynomial $\mathbf{v}^s_{\mathbf{p}}$ over $GF(p)$ which has ABP complexity $O(l)$ such that $\mathbf{p}(x_1,\dots,x_n)^s$ is equal to

$$\mathbf{v}^s_{\mathbf{p}}(x_1^1,..,x_n^k,\chi_{c_1}(\mathbf{p}_1^{\mathbf{C}}(\overline{x}^{\mathbf{C}})),\chi_{c_2}(\mathbf{p}_1^{\mathbf{C}}(\overline{x}^{\mathbf{C}})),..,\chi_{c_m}(\mathbf{p}_l^{\mathbf{C}}(\overline{x}^{\mathbf{C}})))$$

Where $\mathbf{p}_1,\dots,\mathbf{p}_l$ enumerate all subcircuits of $\mathbf{p}$ (each is of the form $\mathbf{p}_G$ for some gate G in the definition of $\mathbf{p}$), $\mathbf{p}_i^{\mathbf{C}}$ is the polynomial $\mathbf{p}_i$ interpreted in the quotient $\mathbf{C}$ and $\overline{x}^{\mathbf{C}}$ is a notation for $x_1^{\mathbf{C}},\dots,x_n^{\mathbf{C}}$. We want to prove that there is an ABP process defining $\mathbf{v}^s_{\mathbf{p}}$ which also defines in its intermediate steps all $\mathbf{v}^s_{\mathbf{q}}$ corresponding to all subcircuits $\mathbf{q}$ of $\mathbf{p}$ from the set $\{\mathbf{p}_1,\dots,\mathbf{p}_l\}$. We do it inductively going from the sources (constants or variables) to the output gate of $\mathbf{p}$. Whenever $\mathbf{q}$ is a variable x_j we can see that $\mathbf{v}^s_{\mathbf{q}}$ is equal x_j^s so the claim holds. Whenever $\mathbf{q}$ is a constant c then $\mathbf{v}^s_{\mathbf{q}}$ is just c^s and again, the claim holds.

Essential case of the induction is when $\mathbf{q}(x_1,\dots,x_n)$ is created by composing basic operation $\mathbf{f}(x_1,\dots,x_r)$ of $\mathbf{A}$ with shorter polynomials $\mathbf{q}_1(x_1,\dots,x_n),\dots,\mathbf{q}_r(x_1,\dots,x_n)$. Note that $\{\mathbf{q}_1,\dots,\mathbf{q}_r\}\subseteq\{\mathbf{p}_1,\dots,\mathbf{p}_l\}$ so we can treat the expressions $\chi_{c_i}(\mathbf{q}_j(\overline{x})^{\mathbf{C}})$ as variables when defining $\mathbf{v}^s_{\mathbf{q}}$. Using the formula for $\mathbf{f}(x_1,\dots,x_r)^s$ we can see that

$$\mathbf{q}(x_1,\dots,x_n)^s = \sum_{i=1..r}\sum_{j=1..k} \mathbf{q}_i(\overline{x})^j\cdot\mathbf{w}_{i,j}(\chi_{c_1}(\mathbf{q}_1^{\mathbf{C}}(\overline{x}^{\mathbf{C}})),\chi_{c_2}(\mathbf{q}_1^{\mathbf{C}}(\overline{x}^{\mathbf{C}})),..,\chi_{c_m}(\mathbf{q}_r^{\mathbf{C}}(\overline{x}^{\mathbf{C}})) + \mathbf{w}(\chi_{c_1}(\mathbf{q}_1^{\mathbf{C}}(\overline{x}^{\mathbf{C}})),\chi_{c_2}(\mathbf{q}_2^{\mathbf{C}}(\overline{x}^{\mathbf{C}})),..,\chi_{c_m}(\mathbf{q}_r^{\mathbf{C}}(\overline{x}^{\mathbf{C}})))$$

Note that $\mathbf{w}_{i,j}$ and $\mathbf{w}$ in the above formula are degree r polynomials. So they can be written as a linear combination of $(rm)^r$ monomials of degree r. So these polynomials have a constant $O(1)$ ABP complexity, as r and m are constants depending on $\mathbf{A}$ and α. So it follows from Fact 11 (and its short proof) that ABP for $\mathbf{v}^s_{\mathbf{q}}$ can be created from ABPs for $(\mathbf{v}^j_{\mathbf{q}_i})_{i\in\{1..r\},j\in\{1..k\}}$ in $O(1)$ steps. In total the polynomial $\mathbf{v}^s_{\mathbf{p}}$ has ABP of size at most $O(l)$, as each vertex of $\mathbf{p}$ introduces only constant number of steps in the overall ABP-process defining $\mathbf{v}^s_{\mathbf{p}}$.

Now consider an arbitrary program $(\mathbf{p})[\iota,T]$ over $\mathbf{A}$. Let ι' be an evaluation of ι in the quotient $\mathbf{C}$, so if a_1,a_2 are two chosen elements in the image of ι we translate them to $a_1^{\mathbf{C}},a_2^{\mathbf{C}}$ in ι'. We established that $(\mathbf{p})[\iota]\,(b_1,\dots,b_n)^s$ is equal to

$$\mathbf{v}^s_{\mathbf{p}}(\iota(b_1)^1,..,\iota(b_n)^k,\chi_{c_1}(\mathbf{p}_1^{\mathbf{C}}(\iota'(\overline{b}))),\chi_{c_2}(\mathbf{p}_1^{\mathbf{C}}(\iota'(\overline{b})),..,\chi_{c_m}(\mathbf{p}_l^{\mathbf{C}}(\iota'(\overline{b})))$$

Where $\mathbf{v}^s_{\mathbf{p}}$ can be represented by ABP of size $O(l)$. Note that $\chi_c(\mathbf{p}_j^{\mathbf{C}}(\iota'(\overline{b}))$ behaves exactly like the evaluation of the program $\left(\mathbf{p}_j^{\mathbf{C}}\right)[\iota',\{c\}]\,(\overline{b})$ in $\mathbf{C}$. And each $\iota(b_i)$ can be affinely interpolated in $GF(p)$. So there is a polynomial $\mathbf{t}_s$ over $GF(p)$ of arity $n+kl$ with ABP of size $O(l)$ such that $(\mathbf{p})[\iota]\,(b_1,\dots,b_n)^s$ is equal to

$$\mathbf{t}_s(b_1,\dots,b_n,(\left(\mathbf{p}_j^{\mathbf{C}}\right)[\iota',\{c\}]\,(b_1,\dots,b_n))_{j\in\{1..l\},c\in C}) \tag{1}$$

Now notice that the value of the program $(\mathbf{p})[\iota, T]$ is determined by the set of values $(\mathbf{p})[\iota]\,(b_1, \ldots, b_n)^s$ for $s \in \{1..k\}$ and the values $\left(\mathbf{p}^{\mathbf{C}}\right)[\iota', \{t'\}]\,(\bar{b})$ for $t' \in C$. This means there is bounded arity function $\mathbf{d} : GF(p)^{k+|C|} \longrightarrow GF(p)$ such that

$$\mathbf{d}\Big(((\mathbf{p})[\iota]\,(b_1, \ldots, b_n)^s)_{s\in\{1..k\}}, \left(\left(\mathbf{p}^{\mathbf{C}}\right)[\iota', \{t'\}]\,(b_1, .., b_n)\right)_{t'\in C}\Big) = 1 \tag{2}$$

iff $(\mathbf{p})[\iota, T]\,(\bar{b}) = 1$. Each argument of $\mathbf{d}$ in the formula above is the composition of the polynomial over $GF(p)$ of ABP complexity $O(l)$ with some number of programs over $\mathbf{C}$. Indeed we can see from Equation (1) that all arguments are programs, except $b_1, \ldots b_n$, which can be replaced by trivial identity programs over $\mathbf{C}$. And the programs $\left(\mathbf{p}^{\mathbf{C}}\right)[\iota', \{t'\}]\,(b_1, .., b_n)$ are itself a program over $\mathbf{C}$ so we can just compose them on the outside with a trivial identity x polynomial of ABP complexity 1. In effect we can apply Fact 23 to formula Equation (2) and show that function computed by $(\mathbf{p})[\iota, T]$ can be also computed by composition of a polynomial which has ABP representation of size $O(l)$ with bunch of programs over $\mathbf{C}$. Size of these programs is upper bounded by the size of $\mathbf{p}$, which finishes the proof. ◀

Now we are ready to characterize $nu\mathsf{P}_{\mathbf{A}}$ for solvable algebras $\mathbf{A}$.

▶ **Theorem 26.** *Let $\mathbf{A}$ be a finite solvable Malcev algebra. Then there exists sequence of primes $p_1, \ldots, p_h$ such that*

$$nu\mathsf{P}_{\mathbf{A}} \subseteq \mathsf{nuDet}_{p_1} \circ \ldots \circ \mathsf{nuDet}_{p_h}$$

Proof. Let $0_A = \alpha_0 \prec \alpha_1 \prec \ldots \prec \alpha_h = 1_A$ be a tight chain of congruences of $\mathbf{A}$. All the prime quotients $\alpha_{i-1} \prec \alpha_i$ have type 2, so they have some prime characteristic p_i corresponding to their traces.

We can see that $\mathbf{C} = \mathbf{A}/\alpha_{h-1}$ is a simple algebra so by Lemma 20 any function computed by a program over $\mathbf{C}$ of length l can be also computed by $\mathrm{BAR}_k \circ \mathrm{MOD}_{p_h}$ circuit of size $O(\mathrm{poly}(l))$, where k is some fixed constant depending on $\mathbf{A}$ and α_{h-1}. Now we apply the Lemma 25 to the algebra $\mathbf{A}/\alpha_{h-1-i}$ and its atom α_{h-i} (for $i \in \{1, .., h-1\}$) to see that function computed by a program over $\mathbf{A}/\alpha_{i-1-h}$ of size l can be also computed by $\mathrm{GABP}_{p_{h-i}} \circ \mathrm{PROG}_{\mathbf{A}/\alpha_{h-1-i}}$ circuit of size $O(\mathrm{poly}(l))$. By applying an induction along the chain $0_A = \alpha_0 \prec \alpha_1 \prec \ldots \prec \alpha_h = 1_A$ we can see that any function computed by a program $\mathbf{q}$ of $\mathbf{A}$ can be also computed by a circuit $C_{\mathbf{q}}$ of the form

$$\mathrm{GABP}_{p_1} \circ \mathrm{GABP}_{p_2} \circ \ldots \mathrm{GABP}_{p_{h-1}} \circ \mathrm{BAR}_k \circ \mathrm{MOD}_{p_h}$$

Note that every $\mathrm{BAR}_k \circ \mathrm{MOD}_p$ subcircuit of size l can be replaced by one nuDet_{p_h} gate of size $O(\mathrm{poly}(l))$, because $\mathrm{BAR}_k \circ \mathrm{MOD}_{p_h}$ can be evaluated by a bounded degree polynomial over $GF(p_h)$, and n-ary bounded degree d polynomials have ABP complexity at most n^d. This finishes the proof. ◀

From Fact 22 and from the fact that NC^2 is closed under taking fixed height compositions we have

▶ **Theorem 27.** *Let $\mathbf{A}$ be a solvable Malcev algebra. Then $nu\mathsf{P}_A \subseteq NC^2$.*

To fully characterize circuit complexity classes over solvable algebras we now prove the opposite of Theorem 26.

▶ **Theorem 28.** *For each sequence of primes $p_1, \ldots, p_h$ there exists a finite solvable Malcev algebra $\mathbf{A}$ such that.*

$$\mathsf{nuDet}_{p_1} \circ \ldots \circ \mathsf{nuDet}_{p_h} \subseteq nu\mathsf{P}_{\mathbf{A}}$$

Proof. We use the the construction used in Theorem 24. Let the underlying set of A be $(Z_{p_1})^2 \times \ldots \times (Z_{p_h})^2$. The idea is that on every pair of coordinates using the same prime p we emulate all the ABPs over $GF(p)$ and then pass them to lower coordinates. So for the prime p_i we introduce operations $(+)^{p_i}, (\pi_1)^{p_i}, (\pi_2)^{p_i}, (u)^{p_i}, (v)^{p_i}$ like in the Theorem 24, which act only on coordinates corresponding to p_i and put all the other coordinates in the image to 0. Additionally we introduce global coordinatewise $+$ on $(Z_{p_1})^2 \times \ldots \times (Z_{p_h})^2$ just to ensure the resulting algebra is Malcev. Then we also introduce unary polynomials $\chi_{i,S}$ for every $i \in \{2, .., h\}$ and $S \subseteq Z_{p_i}$ that look at the left-most coordinate corresponding to p_i and and return 1 to right-most coordinate corresponding to p_{i-1} iff the argument is in the accepting set S, otherwise they return 0. One can check with the definition that the resulting algebra is solvable and Malcev. In fact, similar constructions have been considered in [39, 16]. In this way we can build a $\mathsf{nuDet}_{p_1} \circ \ldots \circ \mathsf{nuDet}_{p_h}$ circuit by building an appropriate ABPs with $(+)^p, (\pi_1)^p, (\pi_2)^p, u^p, v^p$ and using $\chi_{i,S}$ to simulate accepting sets and for passing values to the coordinates that are more to the left. Instruction ι should identify 0 with $(0, \ldots, 0)$ and 1 with $(0, \ldots, 0, 1)$. One can see that in this way we can simulate arbitrary $\mathrm{GABP}_{p_1} \circ \ldots \circ \mathrm{GABP}_{p_h}$ circuit of size $O(l)$ as program of size $O(\mathrm{poly}(l))$ which complete the proof. ◀

If C is a class of algebras by $nu\mathsf{P}_C$ we mean $\bigcup_{\mathbf{A} \in C} nu\mathsf{P}_{\mathbf{A}}$. What follows is the following.

▶ **Corollary 29.** *Let $SOLVABLE$ be the class of all finite solvable Malcev algebras. Then $nu\mathsf{P}_{SOLVABLE} = \mathsf{nultDet}$.*

It is a common saying in the arithmetic circuit complexity papers that arithmetic formulas are determinants in disguise. Here we see that circuits over solvable algebras are iterated determinants in disguise.

8 Nilpotent algebras

In this section we show that circuits over nilpotent algebras recognize exactly languages from the class CC^0. These algebras were historically very important in search for tractable cases in the field of equational satisfiability problems for finite algebras. It was suggested already in [20] that expressive power of circuits over nilpotent algebras is tied to class CC^0. The next Lemma follows from the analysis of the proof in [17] (Theorem 9.7, Claim 9.3).

▶ **Lemma 30.** *Let $\mathbf{A}$ be a finite nilpotent Malcev algebra and α be an atom of characteristic p. Then there exists a constant d such that any function computed by a program $(p)[\iota, S]$ over $\mathbf{A}$ of length l can be also computed by the $MOD_p \circ AND_d \circ PROG_{\mathbf{A}/\alpha}$ circuit of size $O(poly(l))$.*

▶ **Theorem 31.** *Let $\mathbf{A}$ be a finite nilpotent Malcev algebra. Then $nu\mathsf{P}_{\mathbf{A}} \subseteq CC^0$.*

Proof. Take an algebra $\mathbf{A}$ and a tight chain of congruences $0_A = \alpha_0 \prec \alpha_1 \prec \ldots \prec \alpha_h = 1_A$. Define a quotient algebra $\mathbf{A}_j = \mathbf{A}/\alpha_{h-j}$. By Lemma 30 we can see that a program of $\mathbf{A}_j$ of size l can be represented by $\mathrm{poly}(l)$-size $\mathrm{MOD}_{p_j} \circ \mathrm{AND}_d \circ \mathrm{PROG}_{\mathbf{A}_{j-1}}$ circuit, where p_j is a characteristic of $\alpha_{j-1} \prec \alpha_j$ prime quotient. Thus by induction on j we can see that $(\mathbf{p})[\iota, S]$ itself can be computed by $\mathrm{poly}(l)$ size circuit of type

$$\mathrm{MOD}_{p_1} \circ \mathrm{AND}_d \circ \mathrm{MOD}_{p_2} \circ \mathrm{AND}_d \circ \ldots \circ \mathrm{BAR}_k \circ \mathrm{MOD}_{p_h}$$

Here $\mathrm{BAR}_k \circ \mathrm{MOD}_{p_h}$ part comes from application of Lemma 20 at the top. We can replace every AND_d and BAR_k gate by $\mathrm{MOD}_2 \circ \mathrm{MOD}_3$ circuit of constant size (as every Boolean function has such $\mathrm{MOD}_2 \circ \mathrm{MOD}_3$ circuit [6]), and it finishes the proof of theorem (as we can simulate each MOD_p gate by MOD_m gate for m being the lcm of primes involved). ◀

The reverse is also true and $CC_h[m]$ circuits can be simulated by some nilpotent Malcev algebra.

▶ **Example 32.** Let $\mathbf{A}$ be such that $A = (\mathbb{Z}_m)^h$ and $\mathbf{A}$ has the following operations.

1. abelian binary $+$ taken from $(\mathbb{Z}_m, +)^h$,
2. for each $i \in \{1, \dots, h\}$ a unary projection $\mathbf{e}_i$ such that $(\mathbf{e}_i(x))_j = x_i$ iff $i = j$ and $(\mathbf{e}_i(x))_j = 0$ otherwise,
3. for each $i \in \{2, \dots, h\}$ and $S \subseteq \mathbb{Z}_m$ unary $\chi_{S,i}$ such that $\chi_{S,i}(x)_j = 1$ iff $j = i-1 \wedge x_i \in S$ and $(\chi_{S,i}(x))_j = 0$ otherwise.

It is easy to check that one can rewrite any $CC_h[m]$ circuit into a program of this algebra in an efficient way. Indeed just replace the gate MOD_m with an accepting set S on the i-th level of the circuit using projection $\mathbf{e}_i$ and $+$ operation and then pass the resulting value with $\chi_{S,i}$ to the next level. It can be easily checked with the definitions that this algebra is nilpotent and Malcev. Algebras with similar structure and their algebraic properties were already considered in [16][Chapter 3].

▶ **Corollary 33.** *Let $NILPOTENT$ be class of all finite nilpotent Malcev algebras. Then $nu\mathsf{P}_{NILPOTENT} = CC^0$.*

9 Products of solvable and totally ordered algebras

Now we are ready to prove the two last points of Theorem 7.

▶ **Theorem 34.** *Let $\mathbf{A}$ be a finite algebra from a congruence modular variety. If there exist join irreducible congruences $\alpha, \beta \in \mathsf{Con}\,\mathbf{A}$ such that $\beta \leq \alpha$, (α^-, α)-minimal sets are of type 2 and (β^-, β)-minimal sets are of type 4, then $nu\mathsf{P}_{\mathbf{A}} = \mathsf{P}/poly$.*

Proof. Let U be (α^-, α)-minimal, and V be (β^-, β)-minimal set. Let e_U, e_V be some projections to the respective minimal set. Pick arbitrary two distinct elements in U which are equal modulo α but not modulo α^-, call them $0_U, 1_U$, and let $0_V, 1_V$ be the two elements of V. From [15, Lemma 4.20] there is a ternary operation $\mathbf{d}(x, y, z)$ such that $\mathbf{d}(y, y, x) = \mathbf{d}(x, y, y) = x$ for all $x, y \in U$.

First we claim that there exists unary polynomial $\mathbf{f}$ in $\mathrm{Pol}\,\mathbf{A}$ such that $\mathbf{f}(0_U) = (0_V)$ and $\mathbf{f}(1_U) = 1_V$. Since α is join irreducible the pair $(0_U, 1_U)$ closed under unary polynomials of $\mathbf{A}$ must connect all the pairs in congruence α, in particular $0_V, 1_V$. Hence the undirected graph which vertices are elements of $\mathbf{A}$ and edges are of the form $\{\mathbf{g}(0_U), g(1_U)\}$ for all unary polynomials $\mathbf{g}$, must connect 0_V with 1_V. But since $\mathbf{e}_V(0_V) = 0_V$ and $\mathbf{e}_V(1_V) = 1_V$, if we replace each edge $\{\mathbf{g}(0_U), \mathbf{g}(1_U)\}$ by $\{\mathbf{e}_V(\mathbf{g}(0_U)), \mathbf{e}_V(\mathbf{g}(1_U))\}$ the resulting graph still connects 0_V with 1_V. But the image of $\mathbf{e}_V$ is equal to $\{0_V, 1_V\}$, hence there must be some unary $\mathbf{f}$ such that $\mathbf{f}(\{0_U, 1_U\}) = \{0_V, 1_V\}$. Then, $\mathbf{f}$ or $\mathbf{f}(\mathbf{d}(0_U, x, 1_U))$ does the job.

Now take arbitrary Boolean circuit $C(b_1, \dots, b_n)$. We may assume by De Morgans' laws that negations are only in leafs and all the other gates are $\vee, \wedge$. To simulate circuit C by the program of $\mathbf{A}$ pick $\iota(0) = 0_U, \iota(1) = 1_U$ and the accepting set $S = \{1_V\}$. Circuit $\mathbf{p}$ over $\mathbf{A}$ is created by replacing $\wedge, \vee$ in C by $\wedge_V, \vee_V$ and whenever variable b_i is nonnegated we just replace it by $\mathbf{f}(x_i)$ and when its negated we use $\mathbf{f}(\mathbf{d}(0_U, x_i, 1_U))$ instead. One can see that the program $(\mathbf{p})[\iota, S]$ has linear size in terms of the size of C and computes the exact same function. ◀

To prove the last point we need the following algebraic lemma, which allows us to separate lattice behaviour from the affine one.

▶ **Lemma 35.** *Let* $\mathbf{A}$ *be a finite algebra from a congruence modular variety such that* $\mathrm{typ}\{\mathbf{A}\} \subseteq \{2,4\}$*. If there are no join irreducible congruences* $\alpha, \beta \in \mathsf{Con}\,\mathbf{A}$ *such that* $\beta \leq \alpha$*,* (α^-, α)*-minimal sets are of type* 2 *and* (β^-, β)*-minimal sets are of type* 4*, then there exists a congruence* $\theta \in \mathsf{Con}\,\mathbf{A}$ *such that* $\mathrm{typ}\{\mathbf{A}/\theta\} = \{4\}$ *and all prime quotients below* θ *are of type* 2*.*

Proof. Valeriote in [37] introduced so called transfer principle. We say that a finite algebra satisfies the (i,j)-transfer principle if whenever there are prime quotients $\alpha \prec \beta$ of type i and $\beta \prec \gamma$ of type j there exists β' such that $\alpha \prec \beta'$ is a prime quotient of type j and $\beta' \leq \gamma$. If $\mathbf{A}$ satisfies $(4,2)$-transfer principle then we can transform any path from $0_{\mathbf{A}}$ to $1_{\mathbf{A}}$ in congruence lattice of $\mathbf{A}$ in such a way that it is of the form $0_{\mathbf{A}} = \alpha_1 \prec \alpha_2 \dots \alpha_{h-1} \prec \alpha_h = 1_{\mathbf{A}}$ and there exists k such that $\alpha_i \prec \alpha_{i+1}$ is of type 2 for $i < k$ and of type **4** for $i \geq k$. In such a case $\theta = \alpha_k$ does the job. If $\mathbf{A}$ does not satisfy $(4,2)$-transfer principle then by [19, Lemma 6.2] there are congruences $\alpha \prec \beta \prec \gamma$ such that $\alpha \prec \beta$ is a quotient of type **4**, $\beta \prec \gamma$ is a quotient of type **2** and γ is join irreducible. Contradiction, since $\alpha \prec \beta$ can be transposed down to quotient $\alpha' \prec \beta'$ of type **4** with β' being join irreducible. ◀

This leads us to the following.

▶ **Theorem 36.** *Let* $\mathbf{A}$ *be a finite algebra from a congruence modular variety such that* $\mathrm{typ}\{\mathbf{A}\} \subseteq \{2,4\}$ *and there exists a congruence* $\beta \in \mathsf{Con}\,\mathbf{A}$ *such that* $\mathrm{typ}\{\mathbf{A}/\beta\} = \{4\}$ *and all prime quotients below* β *are of type* 2*. If* $\mathbf{A}/\beta$ *is a subdirect product of totally ordered algebras*

$$nu\mathsf{P}_{\mathbf{A}} \subseteq \mathsf{nultDet} \circ MONOTONE \quad (\subseteq NC^2 \circ MONOTONE)$$

Proof. We can see that if $\beta = 1_{\mathbf{A}}$ the statement follows from Theorem 27. If β is nontrivial, we can choose a sequence of congruences $0_A = \alpha_0 \prec \alpha_1 \prec \ldots \prec \alpha_h = \beta$. Each prime quotient $\alpha_{i-1} \prec \alpha_i$ is of type 2 and has some prime characteristic p_i. Therefore by Lemma 15 and Lemma 25 iteratively applied along a chain α_i we can see that

$$nu\mathsf{P}_A \subseteq \mathsf{nuDet}_{p_1} \circ \ldots \circ \mathsf{nuDet}_{p_h} \circ \mathrm{MULTIMONOTONE}$$

But now any circuit of the class MULTIMONOTONE is a composition of bounded arity function with some number of monotone circuits. Any bounded arity Boolean function is computable with the polynomial of constant ABP complexity over any finite field with accepting set $\{1\}$ so

$$nu\mathsf{P}_A \subseteq \mathsf{nultDet} \circ \mathrm{MONOTONE}$$ ◀

10 Simple algebras

In this section we will consider simple algebras (not necessarily from congruence modular varieties). In such a case $0 \prec 1$ is an only prime quotient and hence every two minimal sets are polynomially equivalent. Hence, it makes sense to call a type of minimal sets a type of algebra. Moreover, every minimal set has exactly one trace and no tail. Most of our results can be summarized in the following theorem.

▶ **Theorem 37.** *Let* $\mathbf{A}$ *be simple finite algebra then if its type is equal:*

1 *then* $nu\mathsf{P}_{\mathbf{A}} = BAR_c$*, for some constant* c*.*
2 *then* $nu\mathsf{P}_{\mathbf{A}} = BAR_c \circ MOD_p$*, for some constant* c *and prime number* p*.*
3 *then* $nu\mathsf{P}_{\mathbf{A}} = \mathsf{P}/poly$*.*
4 *then* $MONOTONE \subseteq nu\mathsf{P}_{\mathbf{A}} \subseteq MULTIMONOTONE$ *for totally ordered* $\mathbf{A}$ *and* $nu\mathsf{P}_A = \mathsf{P}/poly$ *otherwise.*

Before we will prove Theorem 37 let us note that simple algebras of type **5** are not mentioned. It is because of surprisingly rich structure of these algebras and the resulting problem in simply characterizing the languages they recognize. It turns out that for many of considered languages there is a finite simple algebra of type 5 recognizing languages from this class. The following example can give us a taste of what might happen in case of algebras of type **5**.

▶ **Example 38.** Let $A = \{\bot, 0, 1\}$ $\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$, $\neg$, $\wedge$, $\vee$, $+$ be operations on A such that
- $\mathbf{e}_1(\bot) = \mathbf{e}_1(0) = \bot$ and $\mathbf{e}_1(1) = 0$,
- $\mathbf{e}_2(\bot) = \bot$ and $\mathbf{e}_2(0) = \mathbf{e}_2(1) = 1$,
- $\mathbf{e}_3(\bot) = \bot$ and $\mathbf{e}_3(0) = \mathbf{e}_3(1) = 0$,
- $\neg\bot = \bot$, $\neg 0 = 1$ and $\neg 1 = 0$,
- $\bot \wedge x$
- $\bot \wedge x = x \wedge \bot = \bot$ for $x \in A$, $0 \wedge y = y \wedge 0 = 0$ for $y \in \{0, 1\}$ and $1 \wedge 1 = 1$,
- $\bot \vee x = x \vee \bot = \bot$ for $x \in A$, $0 \vee 0 = 0$ and $1 \vee y = y \vee 1 = 1$ for $y \in \{0, 1\}$,
- $\bot + x = x + \bot = \bot$ for $x \in A$, $0 + 1 = 1 + 0 = 1$, $0 + 0 = 1 + 1 = 0$.

Then for algebras
- $\mathbf{A}_1 = (a, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \wedge)$,
- $\mathbf{A}_2 = (a, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \wedge, \vee)$,
- $\mathbf{A}_3 = (a, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \wedge, \neg)$,
- $\mathbf{A}_4 = (a, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, +)$,

it holds that
- MONOTONE $\not\subseteq nu\mathsf{P}_{\mathbf{A}_1} \subseteq$ MULTIMONOTONE,
- MONOTONE $\subseteq nu\mathsf{P}_{\mathbf{A}_2} \subseteq$ MULTIMONOTONE,
- $nu\mathsf{P}_{\mathbf{A}_3} = \mathsf{P}/poly$,
- $P_{\mathbf{A}_4} \subseteq \mathsf{AC}_k[2]$ for some small constant k.

From here until the end of the section we will prove Theorem 37. It follows from [15, Theorem 13.3] that for every finite simple algebra $\mathbf{A}$ of type 1 (unary type) there is a uniform bound $k_{\mathbf{A}}$ for the essential arity of its polynomial operations. Hence, also programs over $\mathbf{A}$ can express only essentialy-constant arity operations. Although this derivation is straightforward we include it here for reader's convenience. As already the expressive power of programs over such algebras is narrow, studying their corresponding circuit complexity is pointless.

Similarly as in the previous sections we will use representations of finite algebras that are inspired by the [15, Chapter 13]. In particular Lemma 13.1 provides representations for type 1 and type 2 simple algebras.

▶ **Lemma 39.** *Let $\mathbf{N}$ be a trace of a simple algebra $\mathbf{A}$ of type 1 (unary type) or type 2 (affine type). Then $\mathbf{A}$ is a subreduct of $\mathbf{N}^{[k]}$ for some integer k.*

What follows is the following

▶ **Observation 40.** *Let $\mathbf{A}$ be a finite simple algebra of type 1 (unary type). Then there is a constant $k_{\mathbf{A}}$ such that for every $f \in \operatorname{Pol}\mathbf{A}$ the essential arity of f is bounded by $k_{\mathbf{A}}$.*

Proof. Without loss of generality we assume that $\mathbf{A}$ is a subreduct of $\mathbf{N}^{[k]}$ for a trace N of A. We will prove that $k_{\mathbf{A}} := k$ does the job.

From the definition of matrix power we have that

$$f(x_1, \ldots, x_n) = (f_1(\overline{x}), \ldots, f_n(\overline{x}))$$

where each f_i is a polynomial of $\mathbf{N}$ over variables $(x_1)_1, \ldots, (x_n)_k$. But every polynomial f_i (as a polynomial of $\mathbf{N}$) depends only on one variable. Thus f depends on at most k variables, each one corresponding to the variable on which f_i depends. ◀

▶ **Corollary 41.** *Let $\mathbf{A}$ be a finite simple algebra of type 1 (unary type). Let $k_{\mathbf{A}}$ be a universal bound on the essential arity of polynomials of $\mathbf{A}$. Then $k_{\mathbf{A}}$ bounds the essential arity of programs over $\mathbf{A}$.*

Proof. If polynomial $\mathbf{p}(\overline{x})$ of $\mathbf{A}$ depends only on variables $x_{i_1}, \ldots, x_{i_k}$ then any program $(p)[\iota, S](\overline{b})$ depends only on variables $b_{i_1}, \ldots, b_{i_k}$. ◀

The first point of Theorem 37 is a straightforward consequence of Corollary 41.

The affine type 2 is similar to the unary type 1 in the sense that it also provides a very narrow expressive power when it comes to polynomials/programs. It follows from Lemma 39 that every simple algebra of type 2 is a subreduct of a matrix power of its trace and hence is subreduct of a module. Then, by Lemma 20 we have that NUDFAs over simple algebra $\mathbf{A}$ of affine type recognize languages contained in $nu\mathsf{P}_{\mathbf{A}} = \mathrm{BAR}_c \circ \mathrm{MOD}_p$ for come constant c and prime p.

NuDFAs over type 3 simple algebras can recognize exactly languages from $\mathsf{P}/poly$ as we can choose ι which maps $\{0, 1\}$ onto minimal set of type **3** which is polynomially equivalent with Boolean algebra. Characterization for simple algebras of type 4 is an easy consequence of Corollary 17.

11 Final remarks and open problems

We explored complexity classes $nu\mathsf{P}_{\mathbf{A}}$ in large, but still restricted class of structures. It is not clear how to characterize computational classes induced by simple algebras of type 5. It is clearly visible that just fundamental tools of Tame Congruence Theory are not enough for the analysis. Understanding this case better could open the door to study general finite algebraic structures in more depth, possibly allowing for some strong representation theorems.

It is also quite easy to see that using type 5 allows us to capture more of the standard classes such as $AC^0, AC[p], ACC^0$ studied in the computational complexity theory. Indeed one can use semilattices as layers, similarly to the proof of Theorem 28 or Example 32, and possibly mix them with some addition modulo layers (depending whether we want to achieve AC^0, $AC^0[p]$ or ACC). Then projections from the upper levels to the lower levels would allow for negation, which enables us to define $AC^0, AC[p], ACC^0$ as limits of such constructions. We can see that $nu\mathsf{P}_C$ for some easy to define classes/varieties of algebras C can capture truly a lot of fundamental computational classes.

It is quite interesting to study whether some other natural classes like TC^0 can be captured by some natural class of algebras. One can not only use the mysterious behaviours of type 5 algebras, but also try to define some solvable subvarieties with the desired properties.

▶ **Open Problem 1.** *Does there exist a class C of finite algebras such that $nu\mathsf{P}_C = TC^0$?*

In view of Theorem 6 the following open problem seems to be important, however probably out of reach for the current techniques.

▶ **Open Problem 2.** *Is $\mathsf{NC}^2 \circ MONOTONE \subsetneq \mathsf{P}/poly$?*

References

1 István Ágoston, János Demetrovics, and László Hannák. On the number of clones containing all constants (a problem of R. McKenzie). In *Lectures in Universal Algebra*, Colloquia Mathematica Societatis Janos Bolyai, pages 21–25. North-Holland, 1986. doi:10.1016/B978-0-444-87759-8.50006-7.

2 Noga Alon and Ravi B. Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7(1):1–22, 1987. doi:10.1007/BF02579196.

3 Kazuyuki Amano and Akira Maruoka. A superpolynomial lower bound for a circuit computing the clique function with at most (1/6) log log n negation gates. *SIAM Journal on Computing*, 35(1):201–216, 2005. doi:10.1137/S0097539701396959.

4 David A. Mix Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing (STOC 1986)*, pages 1–5, 1986. doi:10.1016/0022-0000(89)90037-8.

5 David A. Mix Barrington, Pierre McKenzie, Cris Moore, Pascal Tesson, and Denis Thérien. Equation satisfiability and program satisfiability for finite monoids. In *International Symposium on Mathematical Foundations of Computer Science*, pages 172–181. Springer, 2000. doi:10.1007/3-540-44612.

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 David A. Mix Barrington and Denis Therien. Finite monoids and the fine structure of NC. *Journal of the ACM (JACM)*, 35(4):941–952, 1988. doi:10.1145/48014.63138.

8 Stuart J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Information processing letters*, 18(3):147–150, 1984. doi:10.1016/0020-0190(84)90018-8.

9 Maria Bonet, Toniann Pitassi, and Ran Raz. Lower bounds for cutting planes proofs with small coefficients. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing (STOC 1995)*, pages 575–584, 1995. doi:10.1145/225058.225275.

10 Andrei A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *Proceedings of the 58th IEEE Annual Symposium on Foundations of Computer Science (FOCS 2017)*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.

11 Bruno Cavalar, Mika Göös, Artur Riazanov, Anastasia Sofronova, and Dmitry Sokolov. Monotone circuit complexity of matching. *arXiv:2507.16105*, 2025. doi:10.48550/arXiv.2507.16105.

12 Zeev Dvir, Guillaume Malod, Sylvain Perifel, and Amir Yehudayoff. Separating multilinear branching programs and formulas. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing (STOC 2012)*, pages 615–624, 2012. doi:10.1145/2213977.2214034.

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

14 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.

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

16 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 2020)*, pages 578–590, 2020. doi:10.1145/3373718.3394780.

17 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, 2025. doi:10.4230/LIPIcs.ICALP.2025.161.

18 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 *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 127:1–127:20, 2022. doi:10.4230/LIPIcs.ICALP.2022.127.

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

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

21 Jan Krajíček. Lower bounds to the size of constant-depth propositional proofs. *The Journal of Symbolic Logic*, 59(1):73–86, 1994. doi:10.2307/2275250.

22 Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. doi:10.1038/nature14539.

23 Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM (JACM)*, 39(4):859–868, 1992. doi:10.1145/146585.146605.

24 Meena Mahajan and V. Vinay. Determinant: Combinatorics, algorithms, and complexity. *Chicago Journal of Theoretical Computer Science*, 1997(5), 1997. URL: http://cjtcs.cs.uchicago.edu/articles/1997/5/contents.html.

25 Guillaume Malod and Natacha Portier. Characterizing Valiant's algebraic complexity classes. *Journal of complexity*, 24(1):16–38, 2008. doi:10.1016/j.jco.2006.09.006.

26 Emil L. Post. *The two-valued iterative systems of mathematical logic*. Number 5 in Annals of Mathematics studies. Princeton University Press, Princeton, 1941.

27 Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *The Journal of Symbolic Logic*, 62(3):981–998, 1997. doi:10.2307/2275583.

28 Alexander A. Razborov. Lower bounds on the monotone complexity of some boolean functions. *Dokl. Akad. Nauk SSSR*, 281(4):798–801, 1985. Translation in Soviet Math. Dokl. 31:354–357.

29 Alexander A. Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izvestiya: mathematics*, 59(1):205, 1995. doi:10.1070/IM1995v059n01ABEH000009.

30 Alexander A. Razborov and Steven Rudich. Natural proofs. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing (STOC 1994)*, pages 204–213, 1994. doi:10.1145/195058.195134.

31 Enric Rodríguez-Carbonell and Deepak Kapur. Automatic generation of polynomial loop invariants: Algebraic foundations. In *Proceedings of the 2004 international symposium on Symbolic and algebraic computation*, pages 266–273, 2004. doi:10.1145/1005285.1005324.

32 Paul A. Samuelson. A method of determining explicitly the coefficients of the characteristic equation. *The Annals of Mathematical Statistics*, 13(4):424–429, 1942. doi:10.1214/aoms/1177731540.

33 Hanamantagouda P. Sankappanavar and Stanley Burris. *A course in universal algebra*, volume 78. Springer, 1981.

34 Sriram Sankaranarayanan, Henny B. Sipma, and Zohar Manna. Non-linear loop invariant generation using Gröbner bases. In *Proceedings of the 31st ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 318–329, 2004. doi:10.1145/964001.964028.

35 Adi Shamir. IP= PSPACE. *Journal of the ACM (JACM)*, 39(4):869–877, 1992. doi:10.1145/146585.146609.

36 Éva Tardos. The gap between monotone and non-monotone circuit complexity is exponential. *Combinatorica*, 8(1):141–142, 1988. doi:10.1007/BF02122563.

37 Matthew A. Valeriote. *On Decidable Locally Finite Varieties*. Ph.D. thesis, University of California, Berkeley, 1986.

38 Leslie G. Valiant. Completeness classes in algebra. In *Proceedings of the eleventh annual ACM symposium on Theory of computing (STOC 1979)*, pages 249–261, 1979. doi:10.1145/800135.804419.

39 Joel Jamshid VanderWerf. *Wreath decompositions of algebras.* Ph.D. thesis, University of California, Berkeley, 1994.

40 Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

41 Dmitriy Zhuk. A proof of the CSP dichotomy conjecture. *Journal of the ACM (JACM)*, 67(5):1–78, 2020. doi:10.1145/3402029.