Constructing Small Monadic Decompositions in Presburger Arithmetic

Moses Ganardi
University of Kaiserslautern-Landau, Kaiserslautern, Germany

Marin Ricros
University of Kaiserslautern-Landau, Kaiserslautern, Germany

Abstract

A monadic decomposition of a formula over a first-order theory is an equivalent Boolean combination of atomic formulas, each containing only one variable. Monadic decomposition is a generic simplification technique that has found applications in various settings such as quantifier elimination, string solving, and constraint databases. Previous work has mostly focused on the decision problem of whether a formula admits a monadic decomposition. However, much less is known on how to efficiently produce a small monadic decomposition, which is required for any application.

We study this question for the quantifier-free fragment of Presburger arithmetic. Here, monadic decomposability is known to be coNP-complete, and monadic decompositions can be computed in exponential time. An exponential size lower bound was only known for monadic decompositions in disjunctive or conjunctive normal form (DNF or CNF). In this work, we extend this exponential lower bound to general monadic decompositions. Guided by this lower bound, we present fragments that admit polynomially-sized monadic decompositions, which, in many cases, can be constructed efficiently. A surprising key ingredient in our proof is a family of small-depth circuits for arithmetic operations in Chinese remainder representation.

2012 ACM Subject Classification Theory of computation → Logic and verification

Keywords and phrases Monadic decomposition, variable independence, Presburger arithmetic

Digital Object Identifier 10.4230/LIPIcs.LICS.2026.47

Acknowledgements The authors thank Dmitry Chistikov, Markus Lohrey, Philipp Werner, and Georg Zetzsche for inspiring discussions. Also, the authors thank Mikhail Starchak, Roland Guttenberg, and Om Swostik Mishra for discussions on quantifier elimination.

1 Introduction

A first-order formula in a logical theory is *monadically decomposable* if it is equivalent to a finite Boolean combination of atomic formulas, each containing at most one variable. For example, the formula $x \cdot y > 0$ over the real field can be expressed equivalently as $(x > 0 \wedge y > 0) \vee (x < 0 \wedge y < 0)$. The latter formula is called a *monadic decomposition* of $x \cdot y > 0$. On the other hand, $x = y$ is not monadically decomposable if there exist infinitely many elements. While such a decomposition might increase the formula size, a monadic formula is usually easier to handle since all variable dependencies are made explicit in a purely propositional way.

Monadic decompositions were originally studied by Libkin [19] under the name of *variable independence*, and then revisited by Veanes, Bjørner, Nachmanson, and Bereg [23]. In various domains, monadic decomposition has turned out to be a useful preprocessing and simplification technique that either enables more efficient subsequent algorithms or bypasses undecidability barriers. Application domains include quantifier elimination [23, 6], string analysis [23, 16], symbolic transducers [9], and query optimization in constraint databases [18, 13]. As a concrete example from [16], consider the problem of word equations with length constraints

whose decidability is a longstanding open problem [11]. In the practically relevant case, where the length constraints are in fact monadically decomposable, the problem becomes decidable since monadic length constraints can be rephrased as regular constraints [20, Chapter 12].

The general approach of using monadic decompositions requires two steps: *Check* whether the input formula Φ is monadically decomposable, and if so, *compute* a monadic decomposition Ψ for Φ. While the problem of checking whether Φ is monadically decomposable is now well-studied for various logical theories (usually solved by high-performance SMT solvers), much less is known on how to actually compute a monadic decomposition Ψ. Since the decomposition Ψ will be used in subsequent algorithms, we are naturally interested in *efficient* algorithms that compute *small* monadic decompositions. Veanes et al. [23] presented a generic semidecision procedure that computes a monadic decomposition if it exists. The procedure works for any decidable theory under mild conditions, but does not possess worst-case guarantees on the decomposition size or the time complexity.

Presburger arithmetic. In this work, we study the problem of constructing small monadic decompositions in *Presburger arithmetic*, also known as *integer linear arithmetic*, which is the first-order theory of the structure $(\mathbb{N}, +, \leq, 0, 1, (\equiv_q)_{q\in\mathbb{N}})$ where $\equiv_q$ is congruence modulo q. As an illustrative example, consider the Presburger formula

$$\Phi(x, y) = (2x = y \vee x < 5) \wedge (x + y \equiv_3 1).$$

While not immediately clear at first sight, Φ admits a monadic decomposition. Indeed: the equality $2x = y$ implies that $x + y$ is divisible by 3, and thus contradicts the congruence $x + y \equiv_3 1$. Therefore we can safely remove $2x = y$. The remaining formula is monadically decomposable since $x < 5$ is already monadic and the congruence $(x + y \equiv_3 1)$ has the monadic decomposition $(x \equiv_3 0 \wedge y \equiv_3 1) \vee (x \equiv_3 1 \wedge y \equiv_3 0) \vee (x \equiv_3 2 \wedge y \equiv_3 2)$. In fact, it is easy to see that any congruence is monadically decomposable. This simple example illustrates how variable dependencies in the form of (in)equalities in a formula can be "cancelled" by constraints appearing in other subformulas, resulting in monadic decomposability.

Checking whether a Presburger formula is monadically decomposable (or equivalently, defines a *recognizable* relation) was shown to be decidable already in the 60's by Ginsburg and Spanier [12], see also [19, Corollary 7] and [23, Proposition 5.1]. The precise complexity for quantifier-free formulas was obtained by Hague et al. [16] who showed that the problem is coNP-complete. Furthermore, they provided an exponential time algorithm that computes a monadic decomposition in DNF of exponential size, assuming it exists. Later Chistikov et al. [6] gave a smaller exponential size bound, reducing the dependency on particular formula parameters. On the other hand, Hague et al. had complemented their result with the following lower bound: Any monadic decomposition of the formula $0 \leq x + y \leq 2^n$ in DNF or CNF must be exponentially large in n. The authors left open whether this result can be extended to general monadic decompositions, i.e., arbitrary Boolean combinations of monadic atoms. Intuitively, one might assume that modulo constraints (e.g. x is even) and monadic inequalities (e.g. $x \leq 7$ or $y \geq 10$) are insufficient to verify whether $x + y \leq 2^n$ holds, unless we enumerate all (exponentially many) cases. We will see, however, that this intuition is a fallacy.

Our contribution. In this work, we present a detailed picture of when Presburger formulas admit small monadic decompositions and provide sufficient conditions for when such decompositions can be computed efficiently. As in previous works, we will focus on the quantifier-free fragment of Presburger arithmetic in this work.

(1) Our first result is an exponential lower bound for the size of monadic decompositions in Presburger arithmetic (Theorem 2). Concretely, the formula $x \equiv y \pmod{2^n}$ only admits monadic decompositions of size $2^{\Omega(n)}$. In fact, our result is a lower bound holds on the number of atoms in such a decomposition. This means it even applies to more compact formula representations, for example with shared common subformulas.

(2) Guided by this lower bound, we identify syntactic and semantic fragments $\mathcal{F}$ of quantifier-free Presburger arithmetic which admit *efficient monadic decomposition*: There exists a polynomial-time algorithm that computes a monadic decomposition for a given formula $\Phi \in \mathcal{F}$, *under the promise* that the formula indeed is monadically decomposable. This includes formulas defining finite relations (Theorem 10), Boolean combinations of linear (in)equalities (Theorem 14), and more generally formulas whose maximal moduli are polynomially bounded (Theorem 15). In particular, the formula $0 \leq x + y \leq 2^n$ considered in [16] has a monadic decomposition of polynomial size. We emphasize that, if the input formula is not monadically decomposable, then the produced decomposition will be meaningless. In particular, the existence of such a polynomial-time decomposition algorithm does not contradict coNP-hardness of monadic decomposability. Still, we find it surprising that decompositions can be computed purely syntactically without any logical deduction in the theory. This contrasts with the algorithm *mondec* developed in [23], which relies on satisfiability checks.

(3) As an intermediate step towards (2) we consider a relaxation, called *weakly* monadic decompositions, i.e., Boolean combinations of monadic inequalities and congruences, the latter possibly involving multiple variables (Section 5.3). We show that, given an arbitrary monadically decomposable formula, one can compute in polynomial-time an equivalent weakly monadic decomposition (Theorem 11). For example, $(x < 5) \wedge (x + y \equiv 1 \pmod 3)$ is a weakly monadic decomposition of the initial example formula Φ. Since congruences are always monadically decomposable, weakly monadic decomposition provide an efficient syntax for monadically decomposable formulas, circumventing the aforementioned exponential lower bound.

(4) To quantify the attainability of small decompositions, independently of computational complexity, we define the *modular index*, a novel parameter of Presburger formulas Φ (Section 6). This index characterizes the minimal size of a monadic decomposition of Φ up to polynomial factors. Specifically, the modular index provides a tight lower bound for the size of monadic decompositions. Furthermore, we prove that, if Φ is monadically decomposable, it admits a decomposition bounded polynomially by the formula size and its modular index. This result resolves the characterization of formulas that permit small monadic decompositions.

Chinese remainder representations. The key ingredient in our upper bound proofs is the existence of P-uniform NC^1 circuits for integer division ($\lfloor x/y \rfloor$) and comparison ($x < y$), when the numbers are represented in *Chinese remainder representation (CRR)*, i.e., modulo small prime numbers. Notice that Presburger arithmetic can define the CRR of a number x, in terms of modulo constraints $x \equiv r \pmod q$. NC^1 is the class of languages computed by bounded-fan in circuits (or equivalently, formulas) of polynomial size and logarithmic depth. P-uniformity is the property that the n-th circuit in the family can be computed in time polynomial in n. The precise complexity of integer division on *binary encoded* numbers has been a longstanding open problem (even logspace was not known until 2001 [7]), until Hesse, Allender, and Barrington proved integer division to be complete for DLOGTIME-uniform TC^0. We refer to [1] for an excellent survey on the "division breakthrough". The main approach of all integer division algorithms, introduced in [2], is to convert between binary encodings and CRRs.

We are not aware of any other algorithmic results that exploit the connection between Presburger arithmetic and small-depth circuits for arithmetic in CRR.

Applications and limitations. Our algorithms can be used as a generic preprocessing technique, converting a monadically decomposable input formula – for large fragments – into explicit monadic form. However, the surprising succinctness of general monadic Presburger formulas comes at a cost. Most applications that we are aware of require formulas in DNF, and a transformation into DNF might increase the formula size up to exponentially. Previous works computed monadic decompositions without any theoretical guarantee [23], or decompositions that are always exponentially sized [16, 6], agnostic of the input formula. We leave it for future work to investigate whether our approach could produce smaller monadic DNFs in practice.

A precise application of our lower bound results concerns quantifier elimination, where an existential Presburger formula $\exists \boldsymbol{x}\, \Phi(\boldsymbol{x}, \boldsymbol{y})$ is to be converted into an equivalent quantifier-free formula $\Psi(\boldsymbol{y})$. Only very recently, it was shown that a block of existential quantifiers can be eliminated in exponential time [15], improving on the double exponential complexity of previous algorithms. Our lower bound implies that eliminating a *single* existential quantifier in a Presburger formula necessarily leads to formulas of exponential size in the worst-case. To the best of our knowledge, such an exponential lower bound has only been known for the elimination of a block of multiple existential quantifiers [15, Section 6].

We also provide an application of periodicity properties of monadically decomposable Presburger formulas. We determine the precise complexity of deciding *monadic Presburger arithmetic*, containing true quantified Presburger sentences $Q_1 x_1 \cdots Q_k x_k\, \Phi$ where the formula Φ is monadically decomposable. In fact, this logic is interreducible with quantified Boolean formulas (QBF) in a strong sense: Full monadic Presburger arithmetic is PSPACE-complete, and deciding its Σ_i-fragment is Σ_i^{P}-complete.

Related work. In [4], Bergsträßer et al. solved the problem of monadic decomposability in Presburger arithmetic in coNP using the framework of Ramsey quantifiers [3]. In its basic form, a Ramsey quantifier expresses the existence of an infinite clique. In particular, a Ramsey quantifier can express that a formula is *not* monadically decomposable, by describing "infinite dependencies" in a relation. While such an infinite clique describes a witness for non-decomposability, this approach does not provide a monadic decomposition in the positive case. Markgraf et al. [21] studied the problem of learning finite unions of rectilinear hypercubes in Angluin's exact learning framework with membership and equivalence queries, obtaining a learning-based monadic decomposition algorithm over $(\mathbb{Z}, +, \leq)$, i.e., the quantifier-free fragment of Presburger arithmetic without modulo constraints. Chistikov et al. [6] studied the complexity of Presburger arithmetic extended with unary counting quantifiers $\exists^{=x} y$, and showed that its complexity drops significantly when restricting to so called monadically-guarded formulas.

2 Preliminaries

Presburger arithmetic. We use the following syntax of *Presburger arithmetic* (PA) [14]. Fix a countable set of first-order variables $\{x, y, x_1, x_2, \dots, \}$. A *term* is an expression of the form $t(\boldsymbol{x}) = \lambda_1 x_1 + \cdots + \lambda_n x_n + c$ where each x_i is a variable and $\lambda_1, \dots, \lambda_n, c \in \mathbb{Z}$. Atomic formulas of Presburger arithmetic are *linear inequalities* and *congruences* (or *modulo constraints*)

$$s(\boldsymbol{x}) \leq t(\boldsymbol{x}) \text{ and } s(\boldsymbol{x}) \equiv t(\boldsymbol{x}) \pmod{m}$$

where $s(\boldsymbol{x}), t(\boldsymbol{x})$ are terms, and $m \in \mathbb{N} \setminus \{0\}$ is a *modulus*. We can always ensure that inequalities are of the form $t(\boldsymbol{x}) \geq 0$ and congruences are of the form $t(\boldsymbol{x}) \equiv 0 \pmod{m}$. Unless mentioned otherwise, all numbers are encoded in binary. If Φ_1, Φ_2 are formulas and x is a variable, then $(\Phi_1 \vee \Phi_2)$, $\neg\Phi_1$, and $\exists x\, \Phi_1$ are formulas. We almost exclusively work in the *quantifier-free fragment* of Presburger arithmetic, unless mentioned otherwise. We use the standard abbreviations for the truth values $\top = (0 = 0)$, $\bot = \neg\top$, Boolean operators $\wedge, \rightarrow, \leftrightarrow$, and the universal quantifier $\forall x$. The semantics of Presburger arithmetic is defined as expected. A variable assignment $\boldsymbol{a}$ is formally a partial function that maps variables to natural numbers. We usually write $\boldsymbol{a}$ as an ordered tuple, assuming an order on the variables. Given a formula $\Phi(\boldsymbol{x})$ and an assignment $\boldsymbol{a}$ on the free variables $\boldsymbol{x}$ of Φ, we write $\Phi(\boldsymbol{a}) \in \{\top, \bot\}$ for the truth value of Φ under $\boldsymbol{a}$.

If Φ is a formula, we denote by $|\Phi|$ its length, i.e., the number of bits required to write down Φ, where numbers are encoded in binary. By $\|\Phi\|$ we denote the maximal coefficient, constant, or modulus appearing in Φ. The set of all moduli occurring in Φ is denoted by $\mathrm{mod}(\Phi)$.

Circuit complexity. A key ingredient in this work are logarithmic depth circuits for arithmetic operations on natural numbers encoded in binary encoding and Chinese remainder representation. We only give the necessary definitions and refer to [24] for a detailed introduction into circuit complexity. A *Boolean circuit* C is a finite directed acyclic graph in which every node, called *gate*, is either labeled by one of the Boolean operations $\neg, \wedge, \vee$, or is an *input gate* labeled by some input variable x_i. Its *size* is the total number of gates, its *depth* is the length of the longest path from an input gate. The *fan-in* (*fan-out*) of a gate is the number of incoming (outgoing) edges. Each $\neg$-gate has fan-in 1, and input gates have fan-in 0. A circuit whose fan-out is bounded by 1 is simply a propositional formula. Finally, some gates are designated as the *output gates*, and are further labeled by some output variable y_i. A circuit with input variables $x_1, \ldots, x_n$ and output variables $y_1, \ldots, y_m$ naturally computes a Boolean function $f_C \colon \{0,1\}^n \to \{0,1\}^m$.

An NC^1 *circuit family* $\{C_n\}_{n\in\mathbb{N}}$ consists of circuits C_n such that for some polynomial $p(n)$ and some constant $c > 0$:

- the circuit C_n has size at most $p(n)$,
- C_n has input variables $x_1, \ldots, x_n$,
- C_n has depth at most $c \cdot \log(n)$,
- every gate of C_n has fan-in at most 2.

Additionally, we tacitly assume that all circuit families $\{C_n\}_{n\in\mathbb{N}}$ are P-*uniform*, i.e. there exists a polynomial-time algorithm which computes from 1^n a description of C_n. We note that stronger notions of uniformity are commonly used, such as $\mathsf{DLOGTIME}$-uniformity. A partial function $f \colon \{0,1\}^* \to \{0,1\}^*$ is *computable in* NC^1 if there exists a NC^1 circuit family $\{C_n\}_{n\in\mathbb{N}}$ such that for every $\boldsymbol{u} \in \mathsf{dom}(f)$ if the circuit $C_{|\boldsymbol{u}|}$ on input $\boldsymbol{u}$ computes $f(\boldsymbol{u})$. Since an NC^1-circuit of depth $d = O(\log n)$ can be unfolded in polynomial-time into a formula of size $2^d = n^{O(1)}$, P-uniform NC^1 can be alternatively defined using families of polynomial-size Boolean formulas. Throughout this paper we repeatedly make use of this fact.

We also mention the subclasses AC^0 and TC^0 of NC^1. An AC^0 circuit family $\{C_n\}_{n\in\mathbb{N}}$ consists of circuits C_n whose depth is bounded by a constant but the fan-in is unbounded. A TC^0 circuit family is defined similarly to AC^0, but circuits additionally may contain *threshold gates* that evaluate to true if and only if at least half of its incoming edges evaluate to true. It is known that $\mathsf{AC}^0 \subsetneq \mathsf{TC}^0 \subseteq \mathsf{NC}^1$. In this work, we use the fact that TC^0 contains iterated addition, i.e., the sum of n binary encoded numbers, and multiplication of two binary encoded numbers [5]. In fact, iterated multiplication is also contained in $\mathsf{DLOGTIME}$-uniform TC^0, which is a remarkable result by Hesse, Allender, and Barrington [17].

3 Monadic decompositions

A formula is *monadic* if it is a Boolean combination of atomic formulas each having at most one variable. A *monadic decomposition* (or simply *decomposition*) of a formula Φ is a monadic formula equivalent to Φ. If such a monadic decomposition exists, we say that Φ is *monadically decomposable*. We say that a monadic formula is *normalized* if all atomic formulas are of the form $x \leq c$, $x \geq c$, or $x \equiv c \pmod{p^k}$ where p is a prime. It is easy to see that every monadic formula can be normalized with only a constant factor size increase, see Lemma 22.

An n-ary relation $R \subseteq \mathbb{N}^n$ over the naturals is *recognizable* if it is a finite union $R = \bigcup_{i=1}^{k} R_i$ of Cartesian products $R_i = \prod_{j=1}^{n} A_{i,j}$ for Presburger-definable subsets $A_{i,j} \subseteq \mathbb{N}$. A Presburger formula defines a recognizable relation if and only if it is monadically decomposable [23, Proposition 3.1].

Periodicity of monadic Presburger relations. Our work is heavily based on the main argument of [16, 6] to compute monadic decompositions in Presburger arithmetic. In essence, it says that monadic Presburger relations are ultimately periodic along every component, i.e., if a tuple $\boldsymbol{a}$ contains a sufficiently large component a_i then a_i can be pumped, without changing membership to the relation. The case of $n = 2$ is illustrated in Figure 1, where any two regions of the same color are "equal".

More precisely, given a threshold $B \in \mathbb{N}$ and a period $M \in \mathbb{N} \setminus \{0\}$ we define the equivalence relation $\simeq_B^M$ on $\mathbb{N}$ by

$$a \simeq_B^M b \iff (a, b \geq B \text{ and } a \equiv b \pmod{M}) \text{ or } (a, b < B \text{ and } a = b).$$

We extend $\simeq_B^M$ to $\mathbb{N}^n$ by defining $\boldsymbol{a} \simeq_B^M \boldsymbol{b}$ iff $a_i \simeq_B^M b_i$ for all $i \leq n$.

▶ **Threshold lemma.** *[[6, Proposition 3]] Let $\Phi(\boldsymbol{x})$ be a monadically decomposable quantifier-free formula, let $|\boldsymbol{x}| = n$ and $M = \operatorname{lcm}(\operatorname{mod}(\Phi))$. Then there exists a threshold $B = 2^{O(n^2)}(M \cdot \|\Phi\|)^{O(n)}$ such that $\boldsymbol{a} \simeq_B^M \boldsymbol{b}$ implies $\Phi(\boldsymbol{a}) \Leftrightarrow \Phi(\boldsymbol{b})$, for all $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{N}^n$.*

The Threshold lemma implies that the following formula is a monadic decomposition of Φ, if Φ is indeed monadically decomposable:

$$\bigvee_{\boldsymbol{a} \in [0, B+M-1]^n} (\boldsymbol{x} \simeq_B^M \boldsymbol{a} \wedge \Phi(\boldsymbol{a}))$$

Notice that it is exponentially large in the number of variables n, and polynomial in $B + M$, which could be exponentially large in the formula size.

We observe on the complexity of *monadic Presburger arithmetic*, containing all true quantified Presburger sentences $Q_1 x_1 \cdots Q_k x_k \, \Phi$, where $Q_i \in \{\exists, \forall\}$ for $i \in [1, k]$ and Φ is monadically decomposable. By the Threshold lemma each variable in a monadically decomposable formulas only takes exponentially few possible types. This allows us to obtain a tight connection to propositional logic:

▶ **Theorem 1.** *Monadic Presburger arithmetic is polynomial-time interreducible with propositional QBF. Both reductions preserve the quantifier alternation. Therefore, monadic Presburger arithmetic is* PSPACE*-complete and deciding its Σ_i-fragment is Σ_i^{P}-complete.*

Proof. A QBF can be easily simulated in monadic Presburger arithmetic, e.g., by replacing each propositional variable x by the inequality $x > 0$. Conversely, consider a Presburger formula

$$\Psi = Q_1 x_1 \ldots Q_n x_n \colon \Phi(x_1, \ldots, x_n)$$

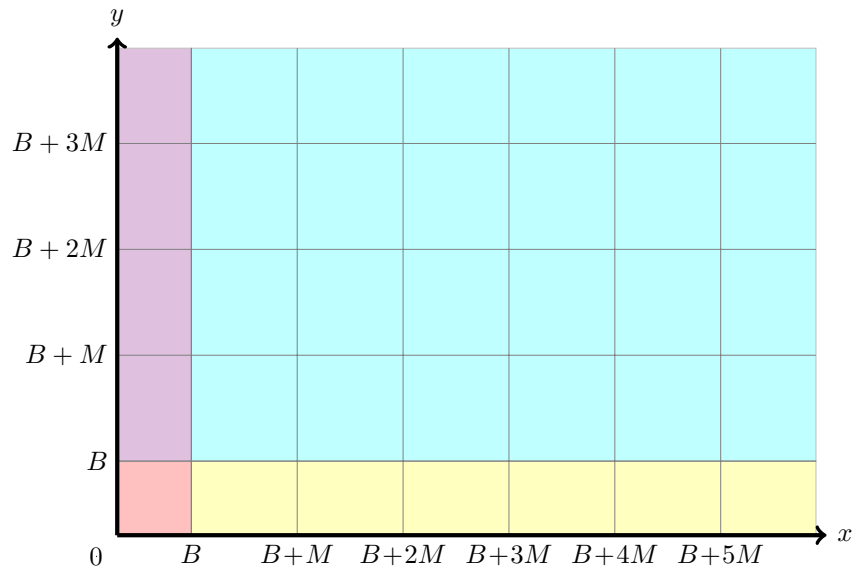

Figure 1 Any two boxes with the same color look the same.

where each $Q_i \in \{\exists, \forall\}$ and Φ is monadically decomposable. By the Threshold lemma we can restrict each quantified variable x_i in Ψ to $[0, B + M - 1]$, without changing the truth value of Ψ. Such a bounded variable x_i can be represented by $\ell = \lceil \log_2(B + M - 1) \rceil$ propositional variables $x_{i,0}, \ldots, x_{i,\ell-1}$ encoding the binary expansion $x_i = \sum_{j<\ell} x_{i,j} 2^j$ of x_i. Notice that ℓ is polynomially bounded in $|\Phi|$. It remains to replace every atomic formula φ in Φ by a propositional formula on the bit variables $x_{i,j}$ that simulates the φ. For inequalities φ this is possible since iterated addition, multiplication, and comparison of binary encoded numbers is in $\mathsf{TC}^0 \subseteq \mathsf{NC}^1$. For congruences, we can use that division in in $\mathsf{TC}^0 \subseteq \mathsf{NC}^1$. ◀

Let us remark that the above proof could have alternatively used Chinese remainder representations instead of binary encodings, which will become necessary in Section 5.

4 Exponential lower bound

In this section we will prove the following lower bound.

▶ **Theorem 2.** *Every monadic decomposition of the formula* $x \equiv y \pmod{2^n}$ *has size* $2^{\Omega(n)}$.

Let Φ be a monadic decomposition of $x \equiv y \pmod{2^n}$. The proof proceeds in four steps that successively simplify the monadic decomposition. In the first step we normalize Φ, i.e., we can assume that Φ is a Boolean combination of formulas $z \leq c$, $z \geq c$, or $z \equiv c \pmod{p^k}$ where p is a prime and z is either x or y. In the second step, we eliminate all inequalities in Φ. Intuitively, any inequality atom does not really contribute to describing $x \equiv y \pmod{2^n}$.

▶ **Lemma 3.** $x \equiv y \pmod{2^n}$ *has a decomposition of size* $O(|\Phi|)$ *whose atomic formulas are only congruences.*

Proof. Each monadic inequality is replaced by true or false by intuitively substituting the free variable by ∞. Formally, we replace

$$\begin{aligned} &\text{each inequality } z \leq c \text{ by } \bot, \text{and} \\ &\text{each inequality } c \leq z \text{ by } \top, \end{aligned} \tag{1}$$

where $z \in \{x, y\}$. Let $\Phi'(x, y)$ be the new formula. From the definition, it is clear that the atomic formulas in Φ' are all congruences, and $|\Phi'| \leq |\Phi|$. It remains to prove that Φ and Φ' are equivalent. For this, let $C \stackrel{\text{def}}{=} \|\Phi\| + 1$ and let $M \stackrel{\text{def}}{=} \operatorname{lcm}(\operatorname{mod}(\Phi))$. We prove that $\Phi'(x, y)$ correctly computes Φ on the values $(x + 2^n CM, y + 2^n CM)$, by induction on the subformulas of Φ. For any subformula φ of Φ obtain φ' by applying the substitution (1) to φ. Then:

$$\text{for all } a, b \in \mathbb{N}\colon\ \varphi(a + 2^n CM, b + 2^n CM) \Leftrightarrow \varphi'(a, b)$$

- Let φ be $z \leq c$. Then φ' is defined as $\bot$. For any $a \in \mathbb{N}$, $a + 2^n CM \geq a + c + 1 > c$ by definition of C. Therefore for any $a, b \in \mathbb{N}$ we get $\varphi(a+2^n CM, b+2^n CM) \Leftrightarrow \bot = \varphi'(a, b)$.
- Let φ be $c \leq z$. Then φ' is defined as $\top$. For any $a \in \mathbb{N}$, $a + 2^n CM \geq a + c + 1 \geq c$ by definition of C. Therefore for any $a, b \in \mathbb{N}$ we get $\varphi(a+2^n CM, b+2^n CM) \Leftrightarrow \top = \varphi'(a, b)$.
- Let φ be $z \equiv c \pmod{p^k}$. Then $\varphi' = \varphi$. For any $a \in \mathbb{N}$, $a + 2^n CM \equiv a \pmod{p^k}$ by definition of M. Therefore for any $a, b \in \mathbb{N}$ we get $\varphi(a + 2^n CM, b + 2^n CM) \Leftrightarrow \varphi(a, b) = \varphi'(a, b)$.
- The case of Boolean combinations follows naturally.

To conclude, we notice that for any $a, b \in \mathbb{N}$,

$$\begin{aligned}\Phi(a, b) &\Leftrightarrow (a \equiv b \pmod{2^n}) \\ &\Leftrightarrow (a + 2^n CM \equiv b + 2^n CM \pmod{2^n}) \\ &\Leftrightarrow \Phi(a + 2^n CM, b + 2^n CM)\end{aligned}$$

which is equivalent to $\Phi'(a, b)$ by the previous induction. ◀

Next, we eliminate congruence atoms whose moduli are not powers of two. Again, the elimination procedure is almost trivial, since these congruences do not contribute to the description of $x \equiv y \pmod{2^n}$.

▶ **Lemma 4.** *$x \equiv y \pmod{2^n}$ has a decomposition of size $O(|\Phi|)$ whose atomic formulas are only congruences with moduli that are powers of two.*

Proof. We assume that Φ is a Boolean combination of monadic congruences by Lemma 3. Now we replace each congruence $z \equiv c \pmod{p^k}$ in Φ where $p \neq 2$ by its evaluation on 0. Let $\Phi'(x, y)$ be the new formula. From the definition, it is again clear that the atomic formulas in Φ' are all congruences with moduli that are powers of two, and $|\Phi'| \leq |\Phi|$. It remains to prove that Φ and Φ' are equivalent.

Let $C \in \mathbb{N}$ be the maximum of 2^n and of all moduli in Φ which are powers of two. Let $M \in \mathbb{N}$ be the least common multiple of all the other moduli appearing in a congruence in Φ. Since C and M are co-prime, Bézout's theorem tells us there exist $u, v \in \mathbb{N}$ such that $1 + u \cdot C = v \cdot M$, and thus for any $a \in \mathbb{N}$ we have $a + auC = avM$. We prove that $\Phi'(x, y)$ correctly computes Φ on the values (xvM, yvM), by induction on the subformulas of Φ. For every subformula φ of Φ obtain φ' by substituting each congruence $z \equiv c \pmod{p^k}$ where $p \neq 2$ by its evaluation on 0. Then:

$$\varphi(avM, bvM) \Leftrightarrow \varphi'(a, b) \qquad \text{for all } a, b \in \mathbb{N}.$$

- Let φ be $z \equiv c \pmod{p^k}$ with $p \neq 2$. Then φ' is defined as $\varphi(0)$. For any $a \in \mathbb{N}$, $avM \equiv 0 \pmod{p^k}$ since p^k divides M. Therefore for any $a, b \in \mathbb{N}$ we get $\varphi(avM, bvM) \Leftrightarrow (0 \equiv c \pmod{p^k}) = \varphi'(a, b)$.
- Let φ be $z \equiv c \pmod{2^k}$. Then $\varphi' = \varphi$. For any $a \in \mathbb{N}$, $avM = a + auC \equiv a \pmod{2^k}$ since 2^k divides C. Therefore for any $a, b \in \mathbb{N}$ we get $\varphi(avM, bvM) \Leftrightarrow \varphi(a, b) = \varphi'(a, b)$.

- The case of Boolean combinations follows naturally.

To conclude, we notice that for any $a, b \in \mathbb{N}$, and by definition of C,

$$\begin{aligned}\Phi(a,b) &\Leftrightarrow (a \equiv b \pmod{2^n}) \\ &\Leftrightarrow (a + auC \equiv b + buC \pmod{2^n}) \\ &\Leftrightarrow (avM \equiv bvM \pmod{2^n}) \\ &\Leftrightarrow \Phi(avM, bvM)\end{aligned}$$

which is equivalent to $\Phi'(a,b)$ by the previous induction. ◀

To finish the proof of Theorem 2, it remains to prove that monadic decompositions of the relation $x \equiv y \pmod{2^n}$ which only use congruences with powers of two as moduli must be exponentially large.

▶ **Lemma 5.** *If Φ is a monadic decomposition of the relation $x \equiv y \pmod{2^n}$ whose atomic formulas are all congruences modulo powers of two, then Φ contains at least 2^{n+1} many literals.*

Proof. Consider a monadic decomposition $\Phi(x,y)$ and turn it into disjunctive normal form, preserving its set of literals. Assume that $\Phi = \bigvee_{i \in I}(\varphi_i(x) \wedge \psi_i(y))$ where φ and ψ are conjunctions of monadic literals. For any $i \in I$ the formula $\varphi_i(x) \wedge \psi_i(y)$ defines a rectangle

$$A \times B \subseteq \{(a,b) \in \mathbb{N}^2 \mid a \equiv b \pmod{2^n}\}. \tag{2}$$

Consider any number $c \in \mathbb{N}$. Since $\Phi(c,c)$ holds, there exists $i \in I$ such that $\varphi_i(c)$, and furthermore $\neg\varphi_i(c + 2^{n-1})$ holds by (2). This implies that every literal in φ_i is true under $x = c$ but one of the literals becomes false under $x = c + 2^{n-1}$. Recall that all atomic formulas are congruences modulo powers of two. Observe that the only congruences modulo a power of two that are true under $x = c$ and false under $x = c + 2^{n-1}$ are

$$L_c = \{x \equiv c \pmod{2^k} \mid k \geq n\} \cup \{x \not\equiv c + 2^{n-1} \pmod{2^k} \mid k \geq n\}.$$

In summary, the set of literals of Φ must intersect L_c for every $c \in \mathbb{N}$. Furthermore, the sets L_c for $c \in [0, 2^n - 1]$ are pairwise disjoint. Hence, the set of literals of Φ on variable x intersects 2^n disjoint sets, and must therefore be of size $\geq 2^n$. Repeating these arguments for variable y give us 2^n extra necessary literals, which concludes the proof. ◀

Application: Quantifier elimination

Theorem 2 implies another interesting result. It has been known that eliminating a block of existential quantifiers in Presburger arithmetic can incur an exponential size blow-up, see [25] and the discussion in [15, Section 6]. To the best of our knowledge, it has been open whether an exponential lower bound also holds for eliminating a single quantifier.

▶ **Theorem 6.** *For any $n \in \mathbb{N}$ there exists a formula $\exists y\, \Phi(x,y)$ of size $O(n)$ such that any equivalent quantifier-free formula $\Psi(x)$ has size $2^{\Omega(n)}$. The formula Φ is monadically decomposable.*

Proof. Consider the formula $\Phi(x,y) = x \equiv_{2^n} y \wedge y < 2^{n-1}$, and let $\Psi(x)$ be a quantifier-free formula equivalent to $\exists y\, \Phi(x,y)$. Then $\Psi(x)$ is automatically a monadic formula, and observe that $\Psi(x)$ holds if and only if the n-th bit of x is 0. Furthermore $\Psi(2^i \cdot x)$ holds if and only if the $(n-i)$-th bit of x is 0, for any $i \in [n-1]$. Therefore, we can express $x \equiv_{2^n} y$ monadically by

$$\bigwedge_{i=0}^{n-1} \Psi(2^i \cdot x) \leftrightarrow \Psi(2^i \cdot y),$$

stating that the n least significant bits of x and y agree. Since any monadic decomposition of $x \equiv_{2^n} y$ has size $2^{\Omega(n)}$ by Theorem 2, the same lower bound applies to the size of $\Psi(x)$. ◀

Let us remark that the techniques used to prove Theorem 2 can also be applied directly to the above formula to give a tighter lower bound: $\Psi(x)$ must contain at least 2^{n-1} literals. This matches the naïve expansion with one congruence for each constant $c < 2^{n-1}$ describing the n least significant bits of variable x.

5 Constructing small decompositions

Guided by the lower bound for congruences, in the following we will present relaxations that enable efficient decomposition algorithms. On the one hand, we identify fragments of Presburger arithmetic that admit efficient monadic decomposition, namely formulas for finite relations and formulas with congruences of polynomially-bounded moduli. On the other hand, we prove that in any monadically decomposable formula we can at least efficiently decompose all inequalities while leaving the congruences non-monadic. We start by presenting Chinese remainder representations, the main ingredient which allows the use of results from circuit complexity.

5.1 Chinese remainder representations

In the following let p_i denote the i-th prime number, and P_d be the product of the first d primes. The *Chinese remainder representation (CRR)* of a natural number $a \in \mathbb{N}$ *in dimension* d is a Boolean sequence $\mathrm{crr}_d(a)$ that encodes each remainder $a \bmod p_i$ in unary as follows:

$$\mathrm{crr}_d(a) \stackrel{\text{def}}{=} (a^{(p_i,r)})_{i\in[d],\, r\in[0,p_i-1]}$$

$$\text{where } a^{(p_i,r)} \stackrel{\text{def}}{=} \begin{cases} 1, & \text{if } a \equiv r \pmod{p_i}, \\ 0, & \text{otherwise.} \end{cases}$$

By the Chinese remainder theorem, every number between 0 and $P_d - 1$ is uniquely represented by its CRR in dimension d, implying a number written in binary using n bits can be written in CRR using $O(n^2 \log n) = n^{O(1)}$ bits. For tuples $\boldsymbol{a} \in \mathbb{N}^n$, we naturally define $\mathrm{crr}_d(\boldsymbol{a}) \stackrel{\text{def}}{=} (\mathrm{crr}_d(a_1), \ldots, \mathrm{crr}_d(a_n))$.

Now let f be a partial operation on natural numbers, mapping tuples $\boldsymbol{a} \in \mathbb{N}^n$ of variable length n to a number $f(\boldsymbol{a}) \in \mathbb{N}$. Given a variable x we denote by $\mathrm{crr}_d(x)$ the tuple of fresh variables $(x^{(p_i,r)})_{i\in[d],\, r\in[0,p_i-1]}$. Similarly, for a variable tuple $\boldsymbol{x}$ we define $\mathrm{crr}_d(\boldsymbol{x})$. Observe that f is NC^1-computable on CRR numbers if and only if there exists a circuit family $\{C_{n,d}\}_{n,d\in\mathbb{N}}$ such that

- $C_{n,d}$ has size at most polynomial in $n + d$ and depth $O(\log(n + d))$,
- $C_{n,d}$ has input variables $x_j^{(p_i,r)}$ and output variables $y^{(p_i,r)}$ for $i \in [d]$, $r \in [0, p_i - 1]$, and $j \in [n]$,
- for all $\boldsymbol{a} \in [0, P_d - 1]^n$ such that $f(\boldsymbol{a})$ is defined and $f(\boldsymbol{a}) < P_d$, the circuit $C_{n,d}$ computes on input $\mathrm{crr}_d(\boldsymbol{a})$ the tuple $\mathrm{crr}_d(f(\boldsymbol{a}))$.

The following theorem was proved in [2, Corollary 5.4] and [10, Theorem 4.6]. A better bound of $\mathsf{DLOGTIME}$-uniform TC^0 is given in [17].

▶ **Theorem 7.** *Integer division and comparison of CRR numbers are computable in* P*-uniform* NC^1*.*

Throughout this paper we also need the simple observation that linear term functions are uniformly computable on CRR numbers in NC^1.

▶ **Lemma 8.** *The following function on CRR numbers is computable in* NC^1*: Given tuples* $\boldsymbol{a}, \boldsymbol{\lambda} \in \mathbb{N}^n$ *of arbitrary length and* $b \in \mathbb{N}$*, compute* $\sum_{i=1}^{n} \lambda_i a_i + b$*.*

The following lemma converts Boolean NC^1-circuits for CRR computations into monadic Presburger formulas, showing Presburger arithmetic can be simulated monadically over a bounded domain.

▶ **Lemma 9.** *Given an atomic formula* $\alpha(\boldsymbol{x})$ *and a binary encoded number* B*, one can compute in polynomial time a monadic formula* $\alpha_B(\boldsymbol{x})$ *which is equivalent to* α *over* $[0, B-1]^n$*.*

Proof. We choose $d \in \mathbb{N}$ minimal such that $P_d > (n \cdot (B-1)+1) \cdot \|\alpha\|$. The sequence of primes $p_1, \ldots, p_d$ can be computed in polynomial time and yields polynomial-sized CRRs.

We can assume that the atomic formula $\alpha(\boldsymbol{x})$ is either an inequality $s(\boldsymbol{x}) \leq t(\boldsymbol{x})$ or a modulo constraint $t(\boldsymbol{x}) \equiv c \pmod m$, where $s(\boldsymbol{x}), t(\boldsymbol{x})$ are terms with *nonnegative* coefficients. Using Lemma 8 we construct for any term $t(\boldsymbol{x})$ in α in polynomial time a circuit C_t which computes the function $\boldsymbol{a} \mapsto t(\boldsymbol{a})$ on numbers encoded in CRR in dimension d. The circuit C_t has polynomial size and logarithmic depth measured in $|\alpha|$ and $\log B$. For any such term t in α, we can bound

$$0 \leq t(\boldsymbol{a}) \leq (n \cdot \|\boldsymbol{a}\| + 1) \cdot \|\alpha\| < P_d \text{ for all } \boldsymbol{a} \in [0, B-1]^n.$$

Therefore C_t computes the term function t correctly on all inputs $\boldsymbol{a} \in [0, B-1]^n$.

We can now construct in polynomial time a bounded fan-in Boolean circuit C_α of logarithmic depth with input variables $\mathrm{crr}_d(\boldsymbol{x})$ such that

$$C_\alpha(\mathrm{crr}_d(\boldsymbol{a})) = \alpha(\boldsymbol{a}) \text{ for all } \boldsymbol{a} \in [0, B-1]^n.$$

If α is an inequality $s(\boldsymbol{x}) \leq t(\boldsymbol{x})$ we use the NC^1-circuit for CRR comparison from Theorem 7, which takes as inputs the outputs of the circuits C_s and C_t. If α is a modulo constraint $t(\boldsymbol{x}) \equiv 0 \pmod m$, we test whether $\lfloor t(\boldsymbol{x})/m \rfloor \cdot m = t(\boldsymbol{x})$. This can be done in NC^1 using the NC^1-circuits for CRR integer division from Theorem 7 and CRR multiplication from Lemma 8, using as inputs the output of C_t and the hardcoded $\mathrm{crr}_d(m)$.

In both cases we unfold C_α into a polynomial-size Boolean formula and replace each variable $x^{(p,r)}$ by the atomic Presburger formula $x \equiv r \pmod p$. ◀

5.2 Finite relations

We can now prove our first upper bound result.

▶ **Theorem 10.** *Given a formula* Φ *defining a finite relation, one can compute a monadic decomposition of* Φ *in polynomial time.*

Proof. Let $\Phi(\boldsymbol{x})$ be a quantifier-free formula that defines a *finite* relation $R \subseteq \mathbb{N}^n$. Let B be a threshold and M a period from the Threshold lemma such that $\boldsymbol{a} \simeq_B^M \boldsymbol{b}$ implies $\Phi(\boldsymbol{a}) \Leftrightarrow \Phi(\boldsymbol{b})$. Observe that R must be contained in $[0, B-1]^n$: If R would contain a tuple $\boldsymbol{a}$ with $a_i \geq B$ for some $i \in [n]$ then we could add any multiple of M to the i-th component of $\boldsymbol{a}$ and obtain infinitely many elements in R, contradicting the assumption that R is finite.

Let $\Psi(\boldsymbol{x})$ be obtained from $\Phi(\boldsymbol{x})$ by replacing each atomic formula by a monadic formula using Lemma 9. Then $\Phi(\boldsymbol{x})$ and $\Psi(\boldsymbol{x})$ are equivalent over $[0, B-1]^n$, and the desired monadic decomposition of Φ is

$$\Phi'(\boldsymbol{x}) \wedge \bigwedge_{i\in[n]} x_i < B.$$

This concludes the proof. ◀

An intriguing open problem is to improve the *formula complexity* of arithmetic operations on CRR numbers, as this would immediately entail smaller monadic decompositions. [10, Theorem 4.6] proves that testing $x \leq y$ for numbers in CRR using n bits has bounded fan-in *circuits* of size $O(n^2)$ and $O(\log n)$ depth. Unfolding such a circuit into a formula, creates a formula of polynomial size whose degree depends on the O-constant of the depth bound.

5.3 Decomposing comparisons

The lower bound from Section 4 suggests that congruences with large modulus are hard to decompose. Our second upper bound result shows that comparison atoms can always be decomposed, assuming that the entire formula is decomposable. This motivates the definition of weakly monadic decompositions.

A Presburger formula is *weakly monadic* if it is a Boolean combination of *monadic* linear inequalities and congruences, the latter possibly involving *multiple* variables. A *weakly monadic decomposition* of a formula Φ is a weakly monadic formula equivalent to Φ. Since congruences are always monadically decomposable, weakly monadic formulas are an alternative syntax for monadic Presburger relations, with exponential succinctness in some cases as witnessed by Theorem 2. It turns out that this syntax is effective and indeed efficient:

▶ **Theorem 11.** *Given a decomposable formula Φ, one can compute a weakly monadic decomposition of Φ in polynomial time.*

In the following we will prove Theorem 11. Consider a monadically decomposable formula Φ, fix a threshold B and a period M from the Threshold lemma. Let us first present a weakly monadic decomposition whose size is exponential in the number of variables, and later improve it to a polynomial dependency. We have to monadically decompose the inequalities of Φ. The big picture of the proof is to look at every case of which variables are above the threshold B and which are below. Our goal is to write a monadic decomposition of Φ of the form

$$\Phi_1 \stackrel{\text{def}}{=} \bigvee_{U\subseteq[n]} \left(\Phi_U(\boldsymbol{x}) \wedge \bigwedge_{i\in U} x_i \geq B \wedge \bigwedge_{i\notin U} x_i < B \right) \tag{3}$$

where Φ_U is a formula to be defined that is equivalent to Φ on the subdomain $\mathrm{dom}(U) \stackrel{\text{def}}{=} \{\boldsymbol{a} \in \mathbb{N}^n \mid a_i \geq B \text{ iff } i \in U\}$.

Let us fix $U \subseteq [n]$ the subset of currently considered "unbounded" variables. We obtain Φ_U from Φ by replacing each inequality atom α by a monadic formula α_U. In this section we consider such atoms to have the shape

$$\alpha(\boldsymbol{x}) = (t(\boldsymbol{x}) \geq 0) \quad \text{where } t = \lambda_1 x_1 + \cdots + \lambda_n x_n + c.$$

where the coefficients λ_i and the constant c are possibly negative integers. We distinguish two cases:

1. If $\{i \in U \mid \lambda_i \neq 0\}$ is empty, then all free variables in α are bounded by B, and $\alpha(\boldsymbol{x})$ actually defines a finite relation when restricted to $\mathrm{dom}(U)$. We say that α is *bounded under* U. Then Lemma 9 gives a monadic formula α_B which is equivalent to α on $\mathrm{dom}(U)$.
2. If $\{i \in U \mid \lambda_i \neq 0\}$ is not empty, let j be its smallest element[1]. Then x_j can be considered "unbounded" and can be pumped arbitrarily large. On the one hand, according to the Threshold lemma, this pumping of x_j does not change the truth value of Φ at the global level. On the other hand, if we think of x_j as being "infinite" the truth value of α is fixed, independent of the valuation of the other variables. We replace α by $\alpha_U = (\lambda_j > 0)$, which is either $\alpha_U = \top$ or $\alpha_U = \bot$.

In summary, Φ_U is obtained from Φ by replacing each atom α by

$$\alpha_U(\boldsymbol{x}) \stackrel{\text{def}}{=} \begin{cases} \alpha_B, & \text{if } \lambda_i = 0 \text{ for all } i \in U, \\ (\lambda_j > 0), & \text{if } j = \min\{i \in U \mid \lambda_i \neq 0\}. \end{cases} \tag{4}$$

▶ **Lemma 12.** *Φ_1 is a monadic decomposition of Φ.*

Proof. Clearly, Φ_1 is monadic. Then fix $U \subseteq [n]$, we need to prove that Φ and Φ_U are equivalent on $\mathrm{dom}(U)$. To this end, we assign to each $\boldsymbol{a} \in \mathrm{dom}(U)$ the pumped tuple $\boldsymbol{a}^{(U)} = (a_1^U, \dots, a_n^U)$ where a_i^U is defined inductively:

$$a_i^U = \begin{cases} a_i, & \text{if } i \notin U \\ a_i + n\|\Phi\|(\max(B, a_{i+1}^U, \dots, a_n^U) + 1) \cdot M, & \text{if } i \in U. \end{cases}$$

The pumped tuple is defined to artificially satisfy the priority order $x_1 \gg \dots \gg x_n \gg B$, applied to variables appearing in U only. Moreover, it satisfies $\boldsymbol{a}^{(U)} \simeq_B^M \boldsymbol{a}$, and therefore $\Phi(\boldsymbol{a}) = \Phi(\boldsymbol{a}^{(U)})$ by the Threshold lemma. It remains to show that every inequality atom α in Φ satisfies $\alpha(\boldsymbol{a}^{(U)}) = \alpha_U(\boldsymbol{a})$ for all $\boldsymbol{a} \in \mathrm{dom}(U)$, which will imply that Φ_U is indeed equivalent to Φ on $\mathrm{dom}(U)$, eventually yielding $\Phi_1(\boldsymbol{x}) \Leftrightarrow \Phi(\boldsymbol{x})$.

▷ Claim. $\alpha(\boldsymbol{a}^{(U)}) = \alpha_U(\boldsymbol{a})$ for all $\boldsymbol{a} \in \mathrm{dom}(U)$.

Proof of claim. If α is bounded under U, then the free variables of α are unchanged after pumping. The claim follows directly from Lemma 9. Otherwise, let $j \in U$ be the minimal index with $\lambda_j \neq 0$. We show that a_j^U dominates the term $t(\boldsymbol{a}^{(U)})$. From the bounds

$$\begin{aligned} |\lambda_i|, |c| &\leq \|\Phi\| && \text{for all } i \in [n], \\ a_i^U &< B && \text{for all } i < j, \\ a_i^U &\leq \max(a_{j+1}^{(U)}, \dots, a_n^{(U)}) && \text{for all } i > j, \end{aligned}$$

we obtain

$$\left| \sum_{i \neq j} \lambda_i a_i^U + c \right| \leq n\|\Phi\| \max(B, a_{j+1}^U, \dots, a_n^U) < a_j^U .$$

Therefore the sign of

$$t(\boldsymbol{a}^{(U)}) = \lambda_j a_j^U + \left(\sum_{i \neq j} \lambda_i a_i^U + c \right)$$

[1] Here, we used the priority order $x_1 \gg x_2 \gg \dots \gg x_n$ on variables. This is an arbitrary choice, any other variable order would also work.

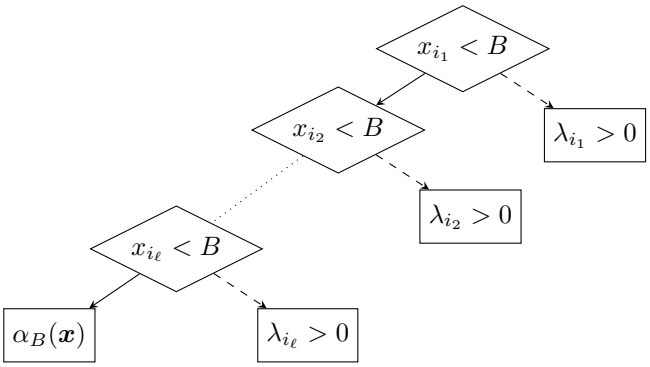

Figure 2 Binary decision diagram representing α_3.

is determined by λ_j, precisely $t(\boldsymbol{x}) \geq 0 \Leftrightarrow \lambda_j > 0$. This concludes the proof of the claim and of the lemma. ◁

◀

To reduce the size of Φ_1 we first push the exponentially large union over all subsets $U \subseteq [n]$ to the level of the formulas $\alpha_U(\boldsymbol{x})$. In other words, obtain Φ_2 from Φ by replacing each inequality atom α by

$$\alpha_2(\boldsymbol{x}) \stackrel{\text{def}}{=} \bigvee_{U \subseteq [n]} \left(\alpha_U(\boldsymbol{x}) \wedge \bigwedge_{i \in U} x_i \geq B \wedge \bigwedge_{i \notin U} x_i < B \right).$$

Since $\Phi \Leftrightarrow \Phi_1$ and each assignment of $\boldsymbol{x}$ induces exactly one set U of unbounded variables, we also have $\Phi \Leftrightarrow \Phi_2$.

Finally, to further compress the atoms α_2, for every inequality atom α in Φ of the form

$$\alpha : \lambda_{i_1} x_{i_1} + \ldots + \lambda_{i_\ell} x_{i_\ell} + c \geq 0 \quad \text{where } i_1 < \cdots < i_\ell \text{ and } \lambda_{i_1}, \ldots, \lambda_{i_\ell} \neq 0$$

define the monadic formula α_3 according to the binary decision diagram in Figure 2. The idea of α_3 is that, at the atomic level, we do not need to consider all exponentially many cases for U anymore. Instead, one goes through the free variables of α one-by-one following the priority order $x_1 \gg \ldots \gg x_n$, and concludes as soon as one unbounded value is found, by virtually pumping it.

▶ **Lemma 13.** *α_2 is equivalent to α_3.*

Proof. ($\Rightarrow$): If $\boldsymbol{a} \in \mathbb{N}^n$ satisfies the disjunction in α_2 then let U be such that $\boldsymbol{a} \in \mathrm{dom}(U)$, and we have $\alpha_U(\boldsymbol{a})$.

- If $\lambda_i = 0$ for all $i \in U$, then $\alpha_U = \alpha_B$ by (4), and $\alpha_2(\boldsymbol{a}) \Leftrightarrow \alpha_B(\boldsymbol{a})$. Similarly, following the decisions for $\alpha_3(\boldsymbol{a})$ leads to $\alpha_B(\boldsymbol{a})$, proving $\alpha_2(\boldsymbol{a}) \Leftrightarrow \alpha_3(\boldsymbol{a})$.
- Else, let $j \in U$ be the minimal index with $\lambda_j \neq 0$, then $\alpha_U(\boldsymbol{x})$ is defined as $\lambda_j > 0$, and following the decisions for α_3 leads us to $\lambda_j > 0$, proving $\alpha_2(\boldsymbol{a}) \Leftrightarrow \alpha_3(\boldsymbol{a})$.

($\Leftarrow$): Conversely, let $\boldsymbol{a}$ satisfy α_3, and let $U \subseteq [n]$ such that $\boldsymbol{a} \in \mathrm{dom}(U)$, i.e., $\bigwedge_{i \in U} a_i \geq B \wedge \bigwedge_{i \notin U} a_i < B$.

- if $\lambda_i = 0$ for all $i \in U$, then $\alpha_3(\boldsymbol{a}) \Leftrightarrow \alpha_B(\boldsymbol{a}) \Leftrightarrow \alpha_2(\boldsymbol{a})$.
- else, let $j \in U$ be the minimal index with $\lambda_j \neq 0$, then $\alpha_3(\boldsymbol{a}) \Leftrightarrow \lambda_j > 0 \Leftrightarrow \alpha_2(\boldsymbol{a})$. ◀

To conclude the proof of Theorem 11, the desired monadic decomposition of Φ is the formula Φ_3 obtained from Φ by replacing every inequality atom α in Φ by α_3.

▶ Remark. The weakly monadic decomposition computed in Theorem 11 has the following restricted shape: The new monadic comparisons are of the form $z < B$ or $z \geq B$, and the newly introduced modulo constraints are monadic, with polynomially small moduli, coming from the formulas α_B.

5.4 Formulas with small moduli

Let us first apply directly Theorem 11 to an input formula which is a Boolean combination of linear (in)equalities. Since the construction from Theorem 11 does not introduce non-monadic congruences, the obtained decomposition is in fact monadic.

▶ **Theorem 14.** *Given a monadically decomposable Boolean combination of linear (in)equalities, one can compute a monadic decomposition of* Φ *in polynomial-time.*

We extend this result to formulas with polynomial moduli, or alternatively moduli written in unary.

▶ **Theorem 15.** *Given a monadically decomposable formula* Φ*, one can compute a monadic decomposition of* Φ *in time polynomial in* $|\Phi|$ *and* $\max(\mathrm{mod}(\Phi))$.

Proof. By Theorem 11, we first compute a weakly monadic decomposition Φ_1 of Φ. Since the transformation $\Phi \mapsto \Phi_1$ does not introduce non-monadic congruences, it remains to decompose congruences in Φ_1 that were already present in Φ. Consider a congruence $t(\boldsymbol{x}) \equiv 0 \pmod{m}$. We represent a number a in unary modulo m, i.e., by the m-tuple of bits $(a \equiv r \pmod{m})_{r\in[0,m-1]}$. Then the term $t(\boldsymbol{x})$ can be computed in NC^1 in unary modulo m representation. From such an NC^1-circuit we obtain a polynomial-sized monadic formula equivalent to $t(\boldsymbol{x}) \equiv 0 \pmod{m}$. ◀

6 Characterization of formulas with small decompositions

In this section we want to characterize which Presburger formulas admit small monadic decompositions, independently of whether the decomposition can be computed efficiently. In light of Theorem 11, we need to understand how many monadic modulo constraints a given formula encodes. Since the dependency of the input formula size is exponential in the worst-case, we introduce a formula parameter, called *modular index*, for a more fine-grained analysis. We will prove that the modular index is both a lower bound and yields an upper bound (up to polynomial terms) to the minimal size of a monadic decomposition.

6.1 The modular index

The most straightforward parameter for measuring decomposability is the *index* of a formula, based on the following Myhill-Nerode style equivalence relation. Fix a quantifier-free Presburger formula $\Phi(\boldsymbol{x})$ in n free variables. For every variable x, define the equivalence relation $\sim^x$ on $\mathbb{N}$ by:

$$a_1 \sim^x a_2 \overset{\text{def}}{\Leftrightarrow} \forall \boldsymbol{b} \in \mathbb{N}^{n-1} \colon \Phi(a_1, \boldsymbol{b}) \Leftrightarrow \Phi(a_2, \boldsymbol{b}).$$

Here, and in the following, we denote by $\Phi(a, \boldsymbol{b})$ the value of Φ under the assignment mapping the distinguished variable x to a, and the other $n-1$ variables to $\boldsymbol{b}$. Then Φ is monadically decomposable if and only if for every variable x, the equivalence relation $\sim^x$ has finite index

(number of equivalence classes) [19, Lemma 4]. We call the maximal index of relations $\sim^x$ for different variables x the *index* of Φ. In fact, if the index is finite then there exists a monadic decomposition of size $O(\mathrm{idx}(\Phi)^{n+1} \cdot n \cdot |\Phi|)$, based on monadic formulas that define each equivalence class, see Lemma 23.

However, the index of Φ can be exponentially larger in the worst-case than its smallest monadic decomposition. For example, if P_d is the product of the first d primes $p_1, \ldots, p_d$, then the formula $x \equiv_{P_d} y$ has index P_d, but the following monadic decomposition has polynomial size in d:

$$\bigwedge_{i=1}^{d} \bigvee_{r \in [0, p_i - 1]} (x \equiv_{p_i} r \wedge y \equiv_{p_i} r)$$

Another example for this exponential gap is given by the formula $0 \leq x \leq y < 2^n$. While the formula has index 2^n, it has a polynomially-sized monadic decomposition by Theorem 10. Guided by these two examples, we will adapt the definition of the $\sim^x$-equivalence so that it looks at the *limit behavior* of Φ through congruence classes *modulo a prime power*.

From now on, we suppose that Φ is monadically decomposable. Let B be a threshold of Φ and $M = \mathrm{lcm}(\mathrm{mod}(\Phi))$ a period, according to the Threshold lemma. For a fixed variable x and a maximal prime power $q \mid M$, we define the *q-equivalence* of Φ with respect to x as the relation defined by $u \sim_q^x v$ if and only if

$$\forall u', v' \geq B, \begin{cases} u' \equiv v' \pmod{M/q} \\ u' \equiv u \pmod{q} \\ v' \equiv v \pmod{q} \end{cases} \implies u' \sim^x v'$$

Notice that u and v only appear in congruences $\bmod q$ in this definition, such that the q-equivalence is refined by congruence modulo q. In other words, $\sim_q^x$ can truly be understood as an equivalence relation over $\mathbb{Z}/q\mathbb{Z}$. The Chinese remainder theorem together with the Threshold lemma guarantee reflexivity of $\sim_q^x$. We define the *q-index* of Φ with respect to x as the index of $\sim_q^x$. The q-index is naturally bounded by q, and captures how many different q congruence classes have different behavior for variable x under Φ. The *modular index* of Φ is the maximal q-index where q ranges over all maximal prime power q dividing M, with respect to all variables. For example, the modular index of the example formula $x \equiv_{P_d} y$ from above is p_d.

▶ Remark. Let us remark that the modular index is in fact a semantic parameter, that only depends on the described relation. Although the definition uses the threshold B and a period M, it can be derived from the Threshold lemma that obtaining different B and M from an other formula describing the same relation would define the same modular index.

6.2 Lower bound

▶ **Proposition 16.** *Let q be a maximal prime power dividing M, let Ψ be a normalized monadic decomposition of Φ. If Ψ contains at most s distinct congruence atoms in x whose modulus divide q, then Φ has q-index at most $s + 1$ wrt. x.*

Proof. Let C be the set of congruences in Ψ in variable x, and $C_q \subseteq C$ be the congruences whose modulus divides q. Define $u \approx_q^x v$ if u and v satisfy the same set of congruences in C_q. We claim that $\approx_q^x$ refines $\sim_q^x$: Suppose that $u \approx_q^x v$, and let $u', v' \geq B$ such that $u' \equiv v' \pmod{M/q}$, $u' \equiv u \pmod{q}$, and $v' \equiv v \pmod{q}$. We need to prove that $u' \sim^x v'$. First,

u' and v' satisfy the same congruences in C_q by hypothesis. Moreover, $u' \equiv v' \pmod{M/q}$ implies that u' and v' satisfy the same congruences in $C \setminus C_q$. Furthermore, since $u', v' \geq B$, all inequalities on variable x evaluate to the same values under u' and v' ($x \leq c$ becomes false, $x \geq c$ becomes true). This establishes $\Psi(u') = \Psi(v')$.

Therefore, to bound the index of $\sim_q^x$, we instead bound the number of $\approx_q^x$-classes. Consider two congruences $\alpha, \beta \in C_q$ of the form $\alpha = (x \equiv c \pmod{p^i})$ and $\beta = (x \equiv d \pmod{p^j})$.

- If $i \geq j$ and $c \equiv d \pmod{p^j}$ then α implies β.
- If $i \leq j$ and $c \equiv d \pmod{p^i}$ then β implies α.
- Otherwise $c \not\equiv d \pmod{p^{\max(i,j)}}$, which implies that $\alpha \wedge \beta$ is unsatisfiable.

In other words, for any $a \in \mathbb{N}$ the set of congruences $C_q(a) = \{\alpha \in C_q \mid \alpha(a)\}$ satisfied under $x = a$ is linearly ordered by implication. Such a subset of consistent congruences can be specified by its minimal element, or $C_q(a) = \emptyset$. Therefore, the number of $\approx_q^x$-classes is bounded by $|C_q| + 1$. ◀

▶ **Corollary 17.** *The modular index of Φ is a lower bound on the size of any monadic decomposition of Φ.*

From Corollary 17 we obtain an alternative proof of Theorem 2. Consider the formula $\Phi = (x \equiv y \pmod{p^k})$ for a prime p. Then the p^k-index of Φ with respect to x is p^k. Therefore, any monadic decomposition of Φ has size $\Omega(p^k)$.

6.3 Upper bound

Finally, we complement the lower bound with a matching upper bound.

▶ **Theorem 18.** *Let Φ be a monadically decomposable quantifier-free formula. Then Φ has a monadic decomposition of size polynomial in $|\Phi|$ and its modular index.*

Fix a monadically decomposable formula Φ, a threshold B and a period $M = \mathrm{lcm}(\mathrm{mod}(\Phi))$. By Theorem 11 we can assume that Φ is weakly monadic. We also assume that $B \geq \|\Phi\|$. Let Q be the set of maximal prime powers that divide M. We assume each modulus in Φ divides a prime power in Q, by replacing each congruence by a system of congruences modulo prime powers. The proof idea of Theorem 18 is to determine, for each variable x, the $\sim_q^x$-class for every $q \in Q$, and to define a representative of the class in CRR. This allow us to evaluate every congruence in Φ.

First, let us observe that knowing the $\sim_q^x$-class of an element for all prime powers q already determines its $\sim^x$-class, if the element is larger than B.

▶ **Lemma 19.** *If $u, v \geq B$ and $u \sim_q^x v$ for all maximal prime powers q dividing M, then $u \sim^x v$.*

Proof. Let $M = q_1 \cdots q_k$ be the factorization into maximal prime powers q_i. We define inductively numbers $a_0, a_1, \ldots, a_k \geq B$ with the property that for all $i \in [0, k]$ and $j \in [1, k]$ we have

$$u \sim_{q_j}^x a_i \tag{5}$$

$$v \sim^x a_i \tag{6}$$

$$u \equiv a_i \pmod{q_1 \cdots q_i} \tag{7}$$

We start with $a_0 \stackrel{\text{def}}{=} v$, which satisfies (5) by assumption, and (6) and (7) trivially. If a_{i-1} is already defined, we choose $a_i \geq B$ such that $u \equiv a_i \pmod{q_i}$ and $a_{i-1} \equiv a_i \pmod{M/q_i}$, which exists since q_i and M/q_i are co-prime. Let us verify that a_i satisfies the three conditions.

Condition (7) is satisfied because $u \equiv a_{i-1} \pmod{q_1 \cdots q_{i-1}}$ by induction hypothesis and $a_{i-1} \equiv a_i \pmod{q_1 \cdots q_{i-1}}$ by choice of a_i. For condition (5), we distinguish whether $j = i$ or $j \neq i$. Clearly, $u \sim^x_{q_i} a_i$ since u and a_i are congruent modulo q_i, and $\sim^x_{q_i}$ refines congruence modulo q_i. Moreover, since a_{i-1} and a_i are congruent modulo q_j, for all $j \neq i$ we have $a_i \sim^x_{q_j} a_{i-1} \sim^x_{q_j} u$. Finally, for condition (6), we show that $a_{i-1} \sim^x a_i$. We use the definition of $a_{i-1} \sim^x_{q_i} a_i$: Since $a_{i-1}, a_i \geq B$ and $a_{i-1} \equiv a_i \pmod{M/q_i}$, we get $a_{i-1} \sim^x a_i$.

To conclude, the number $a_k \geq B$ satisfies $v \sim^x a_k$, and $u \equiv a_k \pmod{M}$. By the Threshold lemma we obtain $u \sim^x v$. ◀

For each variable x and $q \in Q$ define $\pi^x_q \colon \mathbb{N} \to [0, q-1]$ to be any function that maps each number to a unique representative from its class, i.e., $\pi^x(a) \sim^x_q a$ and $a_1 \sim^x_q a_2$ implies $\pi^x_q(a_1) = \pi^x_q(a_2)$. Since every $\sim^x_q$-class is a union of residue classes modulo q, such a function must exist. We show that the CRR of $\pi^x_q(x)$ can be defined by small formulas.

▶ **Lemma 20.** *Each bit in the CRR of $\pi^x_q(x)$ is definable by a Presburger formula of polynomial size.*

Proof. For $r \in \mathbb{N}$ obtain the formula Φ_r from Φ by substituting $x = r$ in any congruence whose modulus does not divide q. We call a pair $(r, \boldsymbol{b}) \in [0, M/q - 1] \times \mathbb{N}^{n-1}$ a *hint that* $u \not\sim^x_q v$ if $\Phi_r(u, \boldsymbol{b}) \not\Leftrightarrow \Phi_r(v, \boldsymbol{b})$. In fact, any inequivalence $u \not\sim^x_q v$ is witnessed by a hint, since by definition there exist $u', v' \geq B$ and $\boldsymbol{b} \in \mathbb{N}^{n-1}$ satisfying $u' \equiv v' \pmod{M/q}$, $u' \equiv u \pmod{q}$, $v' \equiv v \pmod{q}$ and $\Phi(u', \boldsymbol{b}) \neq \Phi(v', \boldsymbol{b})$. Choosing $r \stackrel{\text{def}}{=} u' \bmod M/q$, yields $\Phi_r(u, \boldsymbol{b}) = \Phi(u', \boldsymbol{b})$ and $\Phi_r(v, \boldsymbol{b}) = \Phi(v', \boldsymbol{b})$, and therefore $\Phi_r(u, \boldsymbol{b}) \neq \Phi_r(v, \boldsymbol{b})$.

Now, let s be the q-index of Φ with respect to variable x, and let $a_1, \ldots, a_s$ be the representatives of the $\sim^x_q$-classes of Φ. Let H be a set of $\binom{s}{2}$ hints for all pairs $a_i \neq a_j$; in fact, only $s - 1$ hints suffice, see [22]. Then, the $\sim^x_q$-class of a number $a \geq B$ is determined by the sequence of bits $(\Phi_r(a, \boldsymbol{b}))_{(r,\boldsymbol{b}) \in H}$. Given these input bits, one can compute each bit in the CRR representation of $\pi^x_q(a)$ in a hardcoded fashion by a DNF of size $O(s^2)$. Note that, even though $\boldsymbol{b}$ contains unrestricted natural numbers, we only hardcode them in monadic inequalities or in congruences. This prevents any blowup when writing down formulas $\Phi_r(x, \boldsymbol{b})$. ◀

We are now ready to prove Theorem 18. We naturally combine the functions π^x_q for all variables x into a single function π_q which take vectors $\boldsymbol{a} \in \mathbb{N}^n$ as inputs. For every congruence atom α in Φ whose modulus q' divides $q \in Q$, we build a monadic formula α^* of polynomial size such that

$$\alpha^*(\boldsymbol{a}) \Leftrightarrow \alpha(\pi_q(\boldsymbol{a})) \quad \text{for all } \boldsymbol{a} \in \mathbb{N}^n, \tag{8}$$

by combining Lemma 20 and the Boolean circuit from Lemma 9. Let Φ^* be obtained from Φ by replacing each congruence atom α by α^*, which has clearly polynomial size.

▶ **Lemma 21.** *Φ^* is equivalent to Φ.*

Proof. Let $\pi^x \colon \mathbb{N} \to [0, B + M - 1]$ be the unique function with

- If $a < B$ then $\pi^x(a) = a$.
- If $a \geq B$ then $\pi^x(a) \in [B, B + M - 1]$ satisfies $\pi^x(a) \equiv \pi^x_q(a) \pmod{q}$ for all $q \in Q$.

The system of congruences in the second point has a solution by the Chinese remainder theorem, since all moduli $q \in Q$ are co-prime. Moreover there exists a unique one in the interval $[B, B + M - 1]$ of length $M = \operatorname{lcm}(Q)$, confirming that π^x is well-defined. Let π be the function on $\mathbb{N}^n$ obtained by combining all π^x for variables x.

Let $\boldsymbol{a} \in \mathbb{N}^n$. From (8) we obtain

$$\alpha^*(\boldsymbol{a}) \Leftrightarrow \alpha(\pi(\boldsymbol{a})), \quad \text{for every congruence atom } \alpha \tag{9}$$

Intuitively, $\pi(\boldsymbol{a})$ can be seen as a simplified version of $\boldsymbol{x}$, on which we can evaluate atomic formulas efficiently using only monadic formulas, as described in α^*. Additionally, we will see in the following that $\pi(\boldsymbol{a})$ and $\boldsymbol{a}$ have the same truth value under Φ.

First, observe that $\Phi^*(\boldsymbol{a}) \Leftrightarrow \Phi(\pi(\boldsymbol{a}))$: This holds for congruence atoms by (9). The other atoms are inequality atoms, comparing single variables to constants smaller than B (recall our assumption that $B \geq \|\Phi\|$). By the first point in the definition of π^x, these inequalities are preserved under π^x. The property naturally carries across all subformulas of Φ and yields $\Phi^*(\boldsymbol{a}) \Leftrightarrow \Phi(\pi(\boldsymbol{a}))$. For any $a \in \mathbb{N}$, $q \in Q$, and any variable x, we have $a \sim_q^x \pi_q^x(a)$ by construction, and $\pi_q^x(a) \sim_q^x \pi^x(a)$ by the second point in the definition of π^x, as congruence modulo q refines $\sim_q^x$. Therefore, $a \sim_q^x \pi^x(a)$. By Lemma 19, we obtain $a \sim^x \pi^x(a)$, yielding $\Phi(\boldsymbol{a}) \Leftrightarrow \Phi(\pi(\boldsymbol{a}))$ by definition of $\sim^x$. ◀

Let us comment on why the constructions in this section can likely not be performed efficiently. A straightforward reduction from SAT shows that the modular index cannot be computed in polynomial-time, unless P = NP. The first step of our construction is already to decompose each congruence, by factorizing its modulus into its prime power divisors, which is not known to be computable in deterministic polynomial-time. Then, Lemma 20 requires computing a list of representatives and corresponding "hints", which is at least as hard as determining the modular index itself.

7 Future work

A generalization of monadic decomposition is called *variadic decomposition*: Given a formula $\Phi(\boldsymbol{x})$ and a partition P of the variables $\boldsymbol{x}$, can one rewrite Φ into a Boolean combination of atomic formulas whose free variables are contained in a single partition class of P? We would like to investigate whether the obtained results can be generalized to variadic decompositions.

Another generalization is the problem of monadic separability: Given two quantifier-free Presburger formulas φ, ψ, is there a monadic formula σ that *separates* φ from ψ, i.e., φ implies σ, and ψ implies $\neg\sigma$. Observe that φ is monadically decomposable if and only if φ and $\neg\varphi$ are separable by a monadic formula. The decision problem is coNP-complete [8], but not much is known on how to compute small separator formulas σ.

The decision problem of monadic decomposability has been studied for other logical theories, including real linear arithmetic, Tarski arithmetic, the theory of equality logic with uninterpreted functions (EUF), and over automatic structures. In all of these theories, the question on how to efficiently compute small decompositions has not been investigated yet.

References

1 Eric Allender. The division breakthroughs. *Bull. EATCS*, 74:61–77, 2001.

2 Paul Beame, Stephen A. Cook, and H. James Hoover. Log depth circuits for division and related problems. *SIAM J. Comput.*, 15(4):994–1003, 1986. doi:10.1137/0215070.

3 Pascal Bergsträßer, Moses Ganardi, Anthony W. Lin, and Georg Zetzsche. Ramsey quantifiers over automatic structures: Complexity and applications to verification. In Christel Baier and Dana Fisman, editors, *LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 – 5, 2022*, pages 28:1–28:14. ACM, 2022. doi:10.1145/3531130.3533346.

4 Pascal Bergsträßer, Moses Ganardi, Anthony W. Lin, and Georg Zetzsche. Ramsey quantifiers in linear arithmetics. *Proc. ACM Program. Lang.*, 8(POPL):1–32, 2024. doi:10.1145/3632843.

5 Ashok K. Chandra, Larry J. Stockmeyer, and Uzi Vishkin. Constant depth reducibility. *SIAM J. Comput.*, 13(2):423–439, 1984. doi:10.1137/0213028.

6 Dmitry Chistikov, Christoph Haase, and Alessio Mansutti. Quantifier elimination for counting extensions of presburger arithmetic. In Patricia Bouyer and Lutz Schröder, editors, *Foundations of Software Science and Computation Structures – 25th International Conference, FOSSACS 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings*, volume 13242 of *Lecture Notes in Computer Science*, pages 225–243. Springer, 2022. doi:10.1007/978-3-030-99253-8_12.

7 Andrew Chiu, George I. Davida, and Bruce E. Litow. Division in logspace-uniform NC^1. *RAIRO Theor. Informatics Appl.*, 35(3):259–275, 2001. doi:10.1051/ITA:2001119.

8 Elias Rojas Collins, Chris Köcher, and Georg Zetzsche. The complexity of separability for semilinear sets and parikh automata. In Pawel Gawrychowski, Filip Mazowiecki, and Michal Skrzypczak, editors, *50th International Symposium on Mathematical Foundations of Computer Science, MFCS 2025, Warsaw, Poland, August 25-29, 2025*, volume 345 of *LIPIcs*, pages 38:1–38:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.MFCS.2025.38.

9 Loris D'Antoni and Margus Veanes. The power of symbolic automata and transducers. In Rupak Majumdar and Viktor Kuncak, editors, *Computer Aided Verification – 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, volume 10426 of *Lecture Notes in Computer Science*, pages 47–67. Springer, 2017. doi:10.1007/978-3-319-63387-9_3.

10 George I. Davida and Bruce E. Litow. Fast parallel arithmetic via modular representation. *SIAM J. Comput.*, 20(4):756–765, 1991. doi:10.1137/0220048.

11 Vijay Ganesh, Mia Minnes, Armando Solar-Lezama, and Martin C. Rinard. Word equations with length constraints: What's decidable? In Armin Biere, Amir Nahir, and Tanja E. J. Vos, editors, *Hardware and Software: Verification and Testing – 8th International Haifa Verification Conference, HVC 2012, Haifa, Israel, November 6-8, 2012. Revised Selected Papers*, volume 7857 of *Lecture Notes in Computer Science*, pages 209–226. Springer, 2012. doi:10.1007/978-3-642-39611-3_21.

12 Seymour Ginsburg and Edwin H Spanier. Bounded regular sets. *Proceedings of the American Mathematical Society*, 17(5):1043–1049, 1966.

13 Stéphane Grumbach, Philippe Rigaux, and Luc Segoufin. Spatio-temporal data handling with constraints. *GeoInformatica*, 5(1):95–115, 2001. doi:10.1023/A:1011464022461.

14 Christoph Haase. A survival guide to presburger arithmetic. *ACM SIGLOG News*, 5(3):67–82, 2018. doi:10.1145/3242953.3242964.

15 Christoph Haase, Shankara Narayanan Krishna, Khushraj Madnani, Om Swostik Mishra, and Georg Zetzsche. An efficient quantifier elimination procedure for presburger arithmetic. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, Tallinn, Estonia, July 8-12, 2024*, volume 297 of *LIPIcs*, pages 142:1–142:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ICALP.2024.142.

16 Matthew Hague, Anthony W. Lin, Philipp Rümmer, and Zhilin Wu. Monadic decomposition in integer linear arithmetic. In Nicolas Peltier and Viorica Sofronie-Stokkermans, editors, *Automated Reasoning – 10th International Joint Conference, IJCAR 2020, Paris, France, July 1-4, 2020, Proceedings, Part I*, volume 12166 of *Lecture Notes in Computer Science*, pages 122–140. Springer, 2020. doi:10.1007/978-3-030-51074-9_8.

17 William Hesse, Eric Allender, and David A. Mix Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *J. Comput. Syst. Sci.*, 65(4):695–716, 2002. doi:10.1016/S0022-0000(02)00025-9.

18 Gabriel M. Kuper, Leonid Libkin, and Jan Paredaens, editors. *Constraint Databases*. Springer, 2000. doi:10.1007/978-3-662-04031-7.

19 Leonid Libkin. Variable independence for first-order definable constraints. *ACM Trans. Comput. Log.*, 4(4):431–451, 2003. doi:10.1145/937555.937557.

20 M. Lothaire. *Makanin's Algorithm*, pages 387–442. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2002.

21 Oliver Markgraf, Daniel Stan, and Anthony W. Lin. Learning union of integer hypercubes with queries – (with applications to monadic decomposition). In Alexandra Silva and K. Rustan M. Leino, editors, *Computer Aided Verification – 33rd International Conference, CAV 2021, Virtual Event, July 20-23, 2021, Proceedings, Part II*, volume 12760 of *Lecture Notes in Computer Science*, pages 243–265. Springer, 2021. doi:10.1007/978-3-030-81688-9_12.

22 Franco Parlamento, Alberto Policriti, and KPSB Rao. Witnessing differences without redundancies. *Proceedings of the American Mathematical Society*, 125(2):587–594, 1997.

23 Margus Veanes, Nikolaj S. Bjørner, Lev Nachmanson, and Sergey Bereg. Monadic decomposition. *J. ACM*, 64(2):14:1–14:28, 2017. doi:10.1145/3040488.

24 Heribert Vollmer. *Introduction to Circuit Complexity – A Uniform Approach*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 1999. doi:10.1007/978-3-662-03927-4.

25 Volker Weispfenning. Complexity and uniformity of elimination in presburger arithmetic. In Bruce W. Char, Paul S. Wang, and Wolfgang Küchlin, editors, *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation, ISSAC 1997, Maui, Hawaii, USA, July 21-23, 1997*, pages 48–53. ACM, 1997. doi:10.1145/258726.258746.

A Missing proofs

▶ **Lemma 22.** *For every monadic formula Φ there exists an equivalent normalized formula of size $O(|\Phi|)$.*

Proof. Clearly, every monadic inequality can be written as $x \leq a$ or $a \leq x$ for some constant $a \in \mathbb{N}$. Furthermore, by the Chinese remainder theorem we can decompose each congruence atom $\lambda x \equiv a \pmod m$ into

$$\lambda_i x \equiv a_i \pmod{q_i} \text{ for } i \in \{1, \dots, \ell\},$$

where $m = q_1 \cdots q_\ell$ is the factorization of m into pairwise co-prime prime powers q_i, with $\lambda_i = \lambda \bmod q_i$ and $a_i = a \bmod q_i$. Since each congruence $\lambda_i x \equiv c_i \pmod{q_i}$ has size $O(\log q_i)$, the congruence system has total size $\sum_{i=1}^{\ell} O(\log q_i) = O(\ell + \log m) = O(|\alpha|)$.

Now consider a monadic congruence $\lambda x \equiv a \pmod{p^k}$ where $\lambda, a \in [0, p^k - 1]$, $\lambda \neq 0$. Write $\lambda = p^{k_1} \cdot \mu$ and $a = p^{k_2} \cdot b$ with $p \nmid \mu, b$ and $k_1, k_2 < k$. Since μ and p^k are coprime, μ has an inverse in $\mathbb{Z}/p^k\mathbb{Z}$. Write $c = \mu^{-1} b$, and the congruence can be rewritten as $p^{k_1} \cdot x \equiv p^{k_2} \cdot c \pmod{p^k}$. Two cases:

- If $k_1 > k_2$, then this congruence has no solution and can be replaced by $\bot$.
- Else the congruence is equivalent to $x \equiv p^{k_2 - k_1} \cdot c \pmod{p^{k - k_1}}$

This concludes the proof. ◀

▶ **Lemma 8.** *The following function on CRR numbers is computable in NC^1: Given tuples $\boldsymbol{a}, \boldsymbol{\lambda} \in \mathbb{N}^n$ of arbitrary length and $b \in \mathbb{N}$, compute $\sum_{i=1}^{n} \lambda_i a_i + b$.*

Proof. We perform the computation modulo each prime p in parallel. It is known that multiplication of two unary encoded numbers is in AC^0, and that addition of multiple unary encoded numbers is in TC^0, see [5]. Furthermore, computing $a \bmod p$ for two unary encoded numbers a, p is in AC^0. Therefore, we can compute $\lambda_i a_i$ for all $i \in [n]$ in parallel, then compute the sum $\sum_{i=1}^{n} \lambda_i a_i + b$, and finally reduce the sum modulo p in $\mathsf{TC}^0 \subseteq \mathsf{NC}^1$. ◀

▶ **Lemma 23.** *Let Φ be a quantifier-free Presburger formula. If the index of Φ is finite then there exists a monadic decomposition of size $O(idx(\Phi)^{n+1} \cdot n \cdot |\Phi|)$.*

Proof. For each x, let R_x be a set of $\sim^x$-representatives, i.e. it contains one element from each $\sim^x$-class. Observe that Φ is equivalent to

$$\bigvee_{\boldsymbol{a} \in R} \bigwedge_{i=1}^{n} (x_i \sim^{x_i} a_i). \tag{10}$$

where $R = \{\boldsymbol{a} \in \prod_x R_x \mid \Phi(a)\}$.

We claim that each equivalence class $[a]_{\sim^x}$ is definable by a formula of size $O(|\Phi| \cdot |R_x|)$. Recall that inequivalence $a_1 \not\sim^x a_2$ is witnessed by the existence of a tuple $\boldsymbol{b} \in \mathbb{N}^{n-1}$ with $\Phi(a_1, \boldsymbol{b}) \not\Leftrightarrow \Phi(a_2, \boldsymbol{b})$. Choose a set S_x such that, for any pair $a_1, a_2 \in \mathbb{N}$ with $a_1 \not\sim^x a_2$ there exists $\boldsymbol{b} \in S_x$ with $\Phi(a_1, \boldsymbol{b}) \not\Leftrightarrow \Phi(a_2, \boldsymbol{b})$. Clearly, we can find such a set with $|S_x| \leq |R_x|^2$; in fact, a simple argument shows that $|S_x| \leq |R_x| - 1$ can be ensured [22]. By evaluating $\Phi(x, \boldsymbol{b})$ for all $\boldsymbol{b} \in S_x$, we can thereby determine whether $x \sim^x a$, using a formula of size $O(|\Phi| \cdot |R_x|)$.

To conclude, we can express Equation (10) by a monadic formula of size $O(\mathrm{idx}(\Phi)^{n+1} \cdot n \cdot |\Phi|)$ ◀