

The Complexity of Presburger Arithmetic with Power or Powers

Michael Benedikt ✉ 

Department of Computer Science, University of Oxford, UK

Dmitry Chistikov ✉ 

Centre for Discrete Mathematics and its Applications (DIMAP) &
Department of Computer Science, University of Warwick, Coventry, UK

Alessio Mansutti ✉ 

IMDEA Software Institute, Madrid, Spain

Abstract

We investigate expansions of Presburger arithmetic ($\mathcal{P}\mathbf{a}$), i.e., the theory of the integers with addition and order, with additional structure related to exponentiation: either a function that takes a number to the power of 2, or a predicate $2^{\mathbb{N}}$ for the powers of 2. The latter theory, denoted $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$, was introduced by Büchi as a first attempt at characterizing the sets of tuples of numbers that can be expressed using finite automata; Büchi’s method does not give an elementary upper bound, and the complexity of this theory has been open. The former theory, denoted as $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$, was shown decidable by Semenov; while the decision procedure for this theory differs radically from the automata-based method proposed by Büchi, Semenov’s method is also non-elementary. And in fact, the theory with the power function has a non-elementary lower bound. In this paper, we show that while Semenov’s and Büchi’s approaches yield non-elementary blow-ups for $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$, the theory is in fact decidable in triply exponential time, similarly to the best known quantifier-elimination algorithm for $\mathcal{P}\mathbf{a}$. We also provide a NEXPTIME upper bound for the existential fragment of $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$, a step towards a finer-grained analysis of its complexity. Both these results are established by analyzing a single parameterized satisfiability algorithm for $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$, which can be specialized to either the setting of $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$ or the existential theory of $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$. Besides the new upper bounds for the existential theory of $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$ and $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$, we believe our algorithm provides new intuition for the decidability of these theories, and for the features that lead to non-elementary blow-ups.

2012 ACM Subject Classification Theory of computation → Logic and verification

Keywords and phrases arithmetic theories, exponentiation, decision procedures

Digital Object Identifier 10.4230/LIPIcs.ICALP.2023.112

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

Related Version *Extended Version*: <https://arxiv.org/abs/2305.03037>

Funding *Michael Benedikt*: funded in part by EPSRC grant EP/T022124/1.

Dmitry Chistikov: acknowledges the support of IMDEA Software Institute.

Alessio Mansutti: funded in part by ERC grant No. 852769 (ARIAT).

1 Introduction

This paper concerns decision problems involving first-order logic sentences over the integers. We are given a sentence in the logic, and want to know if it holds in a certain infinite structure over the integers – we refer to these as “satisfaction problems” below. If the sentence can mention “full arithmetic” – both addition and multiplication on the integers – then it is well-known that the satisfaction problem is undecidable [3]. On the other hand, if the sentence mentions only addition, inequality, and integer constants – *Presburger arithmetic* ($\mathcal{P}\mathbf{a}$) –



© Michael Benedikt, Dmitry Chistikov, and Alessio Mansutti;
licensed under Creative Commons License CC-BY 4.0

50th International Colloquium on Automata, Languages, and Programming (ICALP 2023).

Editors: Kousha Etessami, Uriel Feige, and Gabriele Puppis; Article No. 112; pp. 112:1–112:18

Leibniz International Proceedings in Informatics



Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



then the satisfaction problem is decidable [20]. Presburger arithmetic is by no means the maximal decidable arithmetic theory. For instance, adding a “bit predicate” to Presburger arithmetic – a binary predicate holding on (m, n) if m is the largest power of 2 dividing n – does not undermine decidability. This extension is known as *Büchi arithmetic*. A decision procedure for the satisfaction problem of this theory is based on translating each formula into a finite automaton over strings, representing the binary expansions of possible solutions to the formula [2]. Although both are decidable, there is a big difference between Presburger arithmetic and Büchi arithmetic: the satisfaction problem of the former can be decided in triply exponential time [17] and even in doubly exponential space [8], whereas the latter is known to have no elementary bound. See [1] and [25] for a finer-grained analysis of the complexity of Presburger arithmetic, in terms of alternating Turing machines.

Sitting in between Presburger arithmetic and Büchi arithmetic is the extension of $\mathcal{P}\mathbf{a}$ with a predicate for the powers of 2: we refer to this set of numbers as $2^{\mathbb{N}}$ and to the theory as $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$. This predicate is clearly definable in Büchi arithmetic, so the first-order theory of $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$ is again decidable with automata. An alternative decision procedure, avoiding automata, was developed by Semenov in [23]. It proceeds by eliminating quantifiers, arriving at a quantifier-free formula in an enhanced signature – including, for example, a predicate for the highest power of 2 below a given integer. Semenov’s procedure applies more broadly to extensions of Presburger arithmetic by a unary predicate satisfying a condition “effective sparseness”: thus it isolates combinatorial properties of $2^{\mathbb{N}}$ that underlie decidability, rather than automata-theoretic constructions. Semenov’s procedure has been refined and extended by Point; see, e.g., [19]. The complexity of the procedure and the complexity of $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$ has, to our knowledge, received no attention.

Instead of adding a unary predicate, one can add to $\mathcal{P}\mathbf{a}$ the function taking a number n to 2^n : the power function for short, rather than the powers predicate above. The theory, which we denote $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$, subsumes $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$. Semenov proved decidability of this theory as well [24]. But in this case a non-elementary lower bound follows from [6], see [4]. We are not aware of any finer-grained analysis of the complexity of the theory. Note that $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$ is incomparable to Büchi arithmetic in expressiveness: in fact the union of the two is undecidable [4].

In this paper we show that the complexity of $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$ is elementary, and is in fact contained in 3EXPTIME. In this sense $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$ is quite similar to Presburger arithmetic in complexity. We also show that the existential fragment of the theory $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$ has elementary complexity: its satisfaction problem is in NEXPTIME.

We show our results on extending $\mathcal{P}\mathbf{a}$ with powers or the power function using a single parameterized algorithm. The algorithm can be applied to decide satisfaction of a $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$ sentence φ in time tower of $|\varphi|$, matching the prior non-elementary complexity. But it can be specialized to the context of either a $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$ sentence or an existential $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$ sentence, giving in each case an elementary bound. The algorithm is based on eliminating quantifiers: it makes heavy use both of existing Presburger quantifier elimination algorithms [25] and the core of the method of Semenov, which involves removing “problematic occurrences” of a variable within a formula. Intuitively, an occurrence of a variable in an atomic formula is unproblematic if it occurs only outside of power functions, or if the atomic formula is just a comparison between two power terms. In the latter case, the exponentiation of the variable can be eliminated by taking logarithms. We factor this core Semenovian idea out into a self-contained subroutine. We give a short top-level procedure that interleaves calls to this subroutine with calls to a variant of Presburger quantifier elimination. The latter enables us to remove quantified variables completely. A meticulous complexity analysis that tracks several parameters of the input formula shows that the procedure achieves the desired

bounds in the special cases of $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$ and existential $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$.

Our work brings the following ideas:

- For $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$, we rewrite the formula by processing quantifier blocks inside out but do not eliminate quantifier alternation. The resulting formula is put in a nontrivial fragment of $\mathcal{P}\mathbf{a}$: integer octagon arithmetic ($\mathcal{O}\mathbf{ct}$), which we observe to be decidable in PSPACE.
- In eliminating problematic variables from formulae, our algorithm exploits a new substitution strategy: it tailors substitutions to individual inequalities, rather than applying them in the entire formula globally or in an individual disjunct in the DNF (à la Reddy and Loveland [22]). For both $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$ and $\exists\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$, this will be key to obtaining strong bounds on the number of homogeneous terms produced during the transformation, and bounds on these terms will give us bounds on the running times of the algorithms.

We believe that our procedure, in addition to providing the desired bounds, gives a good intuition for the decidability of $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$ and the sources of non-elementary blow-up within it. Note that in this work we deal for simplicity with powers of 2, but the same complexity results apply to any other base $k \in \mathbb{N}$, $k > 2$. Expansion with two bases is undecidable [12].

2 Preliminaries

The symbols \mathbb{Z} , \mathbb{N} and \mathbb{N}_+ denote the set of integers, natural numbers (including zero), and positive integers, respectively. We write $\#A$ for the cardinality of a finite set A . Given $n, m \in \mathbb{Z}$, we define $[n, m] := \{n, n+1, \dots, m\}$ and, if $n \in \mathbb{N}_+$, $[n] := [0, n-1]$. For two sets D and C , $[D \rightarrow C]$ stands for the set of all functions from D to C . We write $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ to denote the *floor* and *ceiling* functions, respectively, $|\cdot|$ to denote the *absolute value* function, and $\log(\cdot)$ to denote the *binary logarithm* function. All these functions take as input a real number. Note that $n \in \mathbb{N}$ can be represented in binary using $\lceil \log(n+1) \rceil$ many bits.

We sometimes apply standard set operations and predicates, such as for instance \in , \subseteq and \setminus , to vectors $\mathbf{v} = (v_1, \dots, v_d)$. In these cases, there is an implicit conversion of \mathbf{v} into the set $V = \{v_1, \dots, v_d\}$. As an example, $v \in \mathbf{v}$ and $\mathbf{v} \setminus A$ stand for $v \in V$ and $V \setminus A$, respectively, where A is a set (or another vector).

Presburger arithmetic with a power function. We consider the structure $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|}) := \langle \mathbb{Z}, 1, +, (a \cdot x)_{a \in \mathbb{Z}}, 2^{|x|}, (q \mid x)_{q \in \mathbb{N}_+}, < \rangle$, in which the classical signature of Presburger arithmetic ($\mathcal{P}\mathbf{a}$) is enriched with the unary *power of the absolute value* function $x \mapsto 2^{|x|}$. As usual, 1 is the constant (interpreted as) $1 \in \mathbb{Z}$, + and < stand for addition and strict ordering over \mathbb{Z} , respectively, $x \mapsto a \cdot x$ is the unary function multiplying its input by the constant $a \in \mathbb{Z}$, and $x \mapsto q \mid x$ is the unary relation that is true for integers divisible by $q \in \mathbb{N}_+$.

The *first-order formulae* $\Phi, \Psi, \varphi, \psi, \dots$ of $\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$ are generated from the grammar

$$\Phi, \Psi := \alpha \mid \top \mid \perp \mid \neg\Phi \mid \Phi \wedge \Psi \mid \exists x \Phi \mid \forall x \Phi \quad \alpha := t_1 < t_2 \mid (q \mid t_1),$$

where x is a first-order variable from an infinite countable set \mathbb{V} . The elements of α are the *atomic formulae* of the logic, i.e., they are *linear inequalities* $t_1 < t_2$ between terms t_1 and t_2 , or *divisibility constraints* $q \mid t_1$, where $q \in \mathbb{N}_+$. Instead of allowing arbitrary terms of the signature, we will deal with a simpler language where *terms* are expressions of the form $\sum_{i \in I} a_i \cdot 2^{|x_i|} + \sum_{j \in J} b_j \cdot x_j + c$ where $c \in \mathbb{Z}$ is the *constant* of the term, $a_i, b_j \in \mathbb{Z} \setminus \{0\}$ are the *coefficients* of the *power terms* $(2^{|x_i|})_{i \in I}$ and of the *linear variables* $(x_j)_{j \in J}$, and I, J are finite sets of indices (which might overlap). We also restrict to terms where no variable occurs linearly twice, or occurs exponentiated twice. It is easy to see that formulae of the full language can be converted to use this restricted term language, and our algorithms will

not take us out of this fragment. When we talk about equality of terms, we mean modulo associativity and commutativity of $+$. A term is said to be *homogeneous* if its constant is 0.

The Boolean connectives \vee , \rightarrow and \leftrightarrow , and the standard (in)equalities between terms \leq , $=$, \geq and $>$ are defined from \wedge , \neg and $<$, as usual. We use the absolute value of variables occurring linearly as a shortcut: e.g., $a \cdot |x| < t$ is equivalent to the formula $(x \geq 0 \rightarrow a \cdot x < t) \wedge (x < 0 \rightarrow -a \cdot x < t)$. We write $\Phi(x_1, \dots, x_d)$ or $\Phi(\mathbf{x})$ to highlight the fact that all free variables of the formula Φ are in \mathbf{x} . A formula without free variables is said to be a *sentence*. We write $\Phi \Leftrightarrow \Psi$ whenever Φ and Ψ are equivalent. A finite set of formulae $S := \{\Phi_i : i \in I\}$ is said to be a *cover for* (or *to cover*) a formula Ψ whenever $\Psi \Leftrightarrow \bigvee_{i \in I} \Phi_i$.

The *satisfaction problem* asks whether a given sentence is true.

Term and formula normalization. To simplify the exposition, we often bring terms and formulae to convenient (normal) form, without mentioning this explicitly every time. This normalization does not change our bounds on the asymptotic running time of the algorithms, nor their correctness.

- We assume inequalities have the form $t < 0$, where t is a term. Thus, we convert inequalities of the form $t_1 < t_2$ into $t_1 - t_2 < 0$. The construction of $t_1 - t_2$ follows the convention on terms described above. We will still sometimes refer to more general inequalities $t_1 < t_2$ for brevity, but these should be taken as abbreviations for inequalities of the above form.
- We rearrange terms following associativity and commutativity of $+$. We also evaluate arithmetic expressions, including, e.g., $2^{|a|}$ and $|a|$, where $a \in \mathbb{Z}$.
- In divisibility constraints of the form $q \mid \sum_{i \in I} a_i \cdot 2^{|x_i|} + \sum_{j \in J} b_j \cdot x_j + c$, we always assume $a_i, b_j, c \in [q]$.
- Inequalities $a < b$ and divisibility constraints $q \mid a$ on integers $a, b \in \mathbb{Z}$ and $q \in \mathbb{N}_+$ are evaluated to \top or \perp . So are divisibility constraints $1 \mid t$ or $q \mid 0$, where $q \in \mathbb{N}_+$ and t is a term (these are \top).
- Inequalities of the form $a \cdot x < b$ with $a, b \in \mathbb{Z}$ and $|a| \geq 2$ are rewritten into $x \leq \lfloor \frac{b-1}{a} \rfloor$ if $a > 0$, and to $x \geq \lceil \frac{b-1}{a} \rceil$ if $a < 0$. This normalization is required in the context of the quantifier elimination (q.e.) procedure for Presburger arithmetic applied to octagons (see *integer octagon arithmetic* below).
- Trivial inequalities involving power terms, where just the fact that $2^{|c|}$ is always positive suffices to evaluate the atomic formula, are also rewritten as \top or \perp . For instance, $a \cdot 2^{|x|} < c$ when a and c have different signs or when $c = 0$; or $a \cdot 2^{|x|} < b \cdot 2^{|y|}$ where a and b have different signs.

Beyond normalization, we need the following operations and notation for terms. We write $t(\mathbf{x})$ if all variables appearing in the term t are in \mathbf{x} . Let α_1 be a formula (resp., a term of the form x or $2^{|x|}$). Given a second formula (resp., term) α_2 , $\Phi[\alpha_2 / \alpha_1]$ stands for the formula obtained from Φ by replacing every occurrence of α_1 by α_2 . Additionally, when α_1 and α_2 are two terms t_1 and t_2 , and given $n \in \mathbb{N}_+$, we write $\Phi[\frac{t_2}{n} / t_1]$ for the formula obtained from Φ by replacing each inequality $a \cdot t_1 + t' < t''$ by $a \cdot t_2 + n \cdot t' < n \cdot t''$ and each divisibility constraint $q \mid a \cdot t_1 + t'$ by $n \cdot q \mid a \cdot t_2 + n \cdot t'$. This operation can be seen as scaling by n the atomic formulae where t_1 occurs linearly, relying on the equivalences $s_1 < s_2 \Leftrightarrow n \cdot s_1 < n \cdot s_2$ and $q \mid s \Leftrightarrow n \cdot q \mid n \cdot s$, followed by the substitution of each $n \cdot t_1$ by t_2 . We will restrict the use of term substitutions to the following cases: $\Phi[t / x]$ and $\Phi[\frac{t}{n} / x]$, where x is a variable only occurring linearly in Φ ; and $\Phi[t / 2^{|x|}]$ and $\Phi[\frac{t}{n} / 2^{|x|}]$. Note that in the last two cases, all linear occurrences of the variable x are left untouched. We extend the notion of substitution to multiple terms or formulae: $\Phi[\beta_i / \alpha_i : i \in [1, k]] := (\dots (\Phi[\beta_1 / \alpha_1])[\beta_2 / \alpha_2] \dots)[\beta_k / \alpha_k]$.

Parameters of formulae. As often done for \mathcal{Pa} , the complexity analysis of our procedure requires the introduction of several parameters for a formula. We define functions $lin(\cdot)$, $hom(\cdot)$, $heft(\cdot)$, $mod(\cdot)$, $\mathcal{B}(\cdot)$, and $alt(\cdot)$, to track various features of a formula Φ from $\mathcal{Pa}(\lambda x.2^{|x|})$:

- $lin(\Phi)$ is the set containing the terms 0 and 2 as well as all the terms t that appear in linear inequalities $t < 0$ of Φ (implicitly converting $t_1 < t_2$ into $t_1 - t_2 < 0$);
- $hom(\Phi)$ is the set of *homogeneous linear terms* obtained from the linear terms in $lin(\Phi)$ by eliminating their constant term c (alternatively, updating c to 0);
- $heft(\Phi)$ is the maximum number of variables in a term of Φ ;
- $mod(\Phi)$ is the least common multiple of all $q \in \mathbb{N}_+$ appearing in constraints $q \mid t$ of Φ (if the formula Φ has no divisibility constraints, then we postulate $mod(\Phi) = 1$);
- $\mathcal{B}(\Phi)$ denotes the number of occurrences of negations \neg and conjunctions \wedge in Φ (note that the syntax permits binary conjunction only);
- $alt(\Phi)$ is the *quantifier alternation rank* (number of quantifier blocks) of a formula Φ in prenex normal form.

Throughout the paper, we assume an encoding of terms where constants and coefficients are given in binary representation. By $len(\Phi)$ we denote the *length* of the formula Φ : the number of bits required to write it down. For simplicity, we assume it is always at least 2. We extend the notion of infinity norm to terms. The *infinity norm* $\|t\|$ of a linear term t is the maximum absolute value of a coefficient or constant appearing in t . For a finite set of terms T , we define $\|T\| := \max\{\|t\| : t \in T\}$. The *1-norm* of t , denoted $\|t\|_1$, is the sum of absolute values of all its coefficients and of its constant; this is always non-negative.

3 Summary of main results

This paper focuses on two fragments of $\mathcal{Pa}(\lambda x.2^{|x|})$:

- The first-order theory of $\mathcal{Pa}(2^{\mathbb{N}}(\cdot)) := \langle \mathbb{Z}, 1, +, (a \cdot x)_{a \in \mathbb{Z}}, 2^{\mathbb{N}}(x), (q \mid x)_{q \in \mathbb{N}_+}, < \rangle$, that is, the structure which enriches Presburger arithmetic with the unary relation $x \mapsto 2^{\mathbb{N}}(x)$ that is true for the powers of 2, i.e., $2^{\mathbb{N}}(x) = \top$ iff $x \in \{1, 2, 4, \dots\}$. Note that the relation $2^{\mathbb{N}}(x)$ can be expressed in $\mathcal{Pa}(\lambda x.2^{|x|})$, with the formula $\exists y. x = 2^{|y|}$.
- The existential fragment of $\mathcal{Pa}(\lambda x.2^{|x|})$, denoted by $\exists \mathcal{Pa}(\lambda x.2^{|x|})$. Formulae of this fragment are of the form $\exists x.\Phi$, where Φ is quantifier-free (q.f., in short).

The main results of this paper are summarized below:

► **Theorem 1.** *The satisfaction problem for $\exists \mathcal{Pa}(\lambda x.2^{|x|})$ is in NEXPTIME.*

► **Theorem 2.** *The satisfaction problem for $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$ is in 3EXPTIME.*

Theorems 1 and 2 are based on a common core procedure for $\mathcal{Pa}(\lambda x.2^{|x|})$ that we introduce in Section 4. The procedure manipulates the subformulae of an input formula so that they (eventually) enter the following fragments of $\mathcal{Pa}(\lambda x.2^{|x|})$:

- The *power comparisons fragment*, denoted by \mathcal{PowCmp} . In this fragment, inequalities are restricted to the form $a \cdot 2^{|x|} < b \cdot 2^{|y|}$ or $a \cdot 2^{|x|} < b$, where $a, b \in \mathbb{Z}$, and divisibility constraints are of the form $q \mid 2^{|x|} - r$, where $q \in \mathbb{N}_+$ and $r \in [q]$.
- *Integer octagon arithmetic*, denoted by \mathcal{Oct} (see e.g. [13, 16]), that is, the fragment of \mathcal{Pa} in which inequalities are restricted to the forms $\pm x \pm y < c$ and $\pm x < c$, where $c \in \mathbb{Z}$, and divisibility constraints are of the form $q \mid x - r$, where $q \in \mathbb{N}_+$ and $r \in [q]$.
- The fragment \mathcal{Sem} (short for *Semenov*, as this fragment is related to the one used in [23]). In formulae Φ of this fragment, each variable appears either always linearly or always in a power, and every *bound* variable x appears only in atomic formulae from \mathcal{PowCmp}

(hence, x is always in a power). Moreover, divisibility constraints in Φ are *simple*, i.e., they are of the form $q \mid 2^{|x|} - r$ or of the form $q \mid x - r$, where $q \in \mathbb{N}_+$ and $r \in [q]$. Notice that a sentence in this fragment must be in \mathcal{PowEmp} .

- The quantifier-free fragment of $\mathcal{Pa}(\lambda x.2^{|x|})$, denoted \mathcal{QF} , consisting of all q.f. formulae.

4 The core procedure

Overall organization. Our final decision procedures, which will be presented in Section 5, rely on a core procedure (Algorithm 1) that interleaves calls to what are essentially quantifier elimination subroutines à la Presburger [20] and Semenov [23], respectively, to be explained further below. The input of Algorithm 1 is a formula Φ of $\mathcal{Pa}(\lambda x.2^{|x|})$ in prenex normal form. The procedure can be run in two modes, taking an additional parameter \mathcal{F} accordingly. This parameter specifies the “target” fragment of the logic:

- For Φ obtained as a translation of a $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$ formula into $\mathcal{Pa}(\lambda x.2^{|x|})$, set $\mathcal{F} = \mathit{Sem}$.
- For general formulae of $\mathcal{Pa}(\lambda x.2^{|x|})$, and for the handling of existential $\mathcal{Pa}(\lambda x.2^{|x|})$ in non-deterministic exponential time, set $\mathcal{F} = \mathcal{QF}$.

The output of the algorithm is a simplified formula: more specifically, it is a formula of the form $\exists \mathbf{x}.\varphi$ or $\neg \exists \mathbf{x}.\varphi$, where φ is in \mathcal{F} : “*alternation-free modulo \mathcal{F}* ” below. If the input is a sentence, then the output has no leading quantifiers and is thus a sentence of \mathcal{F} .

The procedure processes blocks of quantifiers at a time, eliminating them one by one. Each block corresponds to one iteration of the outer **while** loop. In line 2 we split the quantifier prefix at the innermost existential block that takes us out of the fragment \mathcal{F} . There may be a choice as to whether \neg appears at the beginning of Π , but this introduces no ambiguity to the choice of \mathbf{u} , because $\forall v.\Psi$ is in \mathcal{F} iff $\exists v.\Psi$ is in \mathcal{F} . This follows because both fragments $\mathcal{F} = \mathit{Sem}$ and $\mathcal{F} = \mathcal{QF}$ are closed under negation.

The organization of the procedure maintains a DNF-like structure. The set Q acts as a worklist containing the formulae; intuitively, they are the conjunctions (although not necessarily of atomic formulae). The PresQE and SemCover subroutines embed Reddy and Loveland’s optimization for \mathcal{Pa} [22]: whenever a pair $(\mathbf{x}, \varphi_1 \vee \varphi_2)$ could be produced, it is split into two pairs (\mathbf{x}, φ_1) and (\mathbf{x}, φ_2) evolving independently for as long as possible. Thus, the DNF-like structure is maintained within each iteration of the outer **while** loop of the Master procedure:

$$\Phi \Leftrightarrow \Pi.\Pi'. \left[\left(\bigvee_{(\mathbf{x}, \varphi) \in Q} \exists \mathbf{x}.\varphi \right) \vee \bigvee_{\varphi \in D} \varphi \right].$$

For each $\varphi \in D \cup \{\varphi : (\mathbf{x}, \varphi) \in Q \text{ for some } \mathbf{x}\}$, we have $\varphi \in \mathcal{F}$. Pairs from Q are processed in the inner **while** loop one at a time. Formulae from D are “done” and will only be picked up again after leaving the current block: the algorithm will no longer process them within the current block. Thanks to the DNF-like structure, our analysis of the parameter growth for an individual pair (\mathbf{x}, φ) can ignore the complexity of the big disjunction (i.e., other pairs in Q and D).

Above we have presented Algorithm 1 deterministically: any deterministic choice can be made in line 5 when popping an element from Q , and in lines 7–9 when choosing an appropriate $x \in \mathbf{x}$. This implementation will be employed to obtain the claimed triply-exponential bound for $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$, but not the NEXPTIME bound for the existential fragment of $\mathcal{Pa}(\lambda x.2^{|x|})$. In the latter case, we will only perform the outer loop once. The prefix Π will always be empty, and thus the formula we are processing can always be considered

■ **Algorithm 1** Master procedure.

Input: fragment $\mathcal{F} \in \{\mathcal{QF}, \mathcal{Sem}\}$;
 formula $\Phi(\mathbf{y})$ in $\mathcal{Pa}(\lambda x. 2^{|x|})$ in prenex normal form with quantifier-free part from \mathcal{F}
 in which all divisibility constraints are simple

Output: an equivalent formula $\Phi'(\mathbf{y})$, alternation-free modulo \mathcal{F} ;
 if Φ is a sentence, Φ' is a sentence of \mathcal{F}

- 1: **while** true **do**
- 2: $\Pi \leftarrow$ the shortest quantifier prefix of Φ , possibly with \neg in front,
 such that $\Phi = \Pi.\exists \mathbf{u}.\Psi$ where Ψ is in \mathcal{F} (if necessary, rewrite $\forall \mathbf{u}$ as $\neg\exists \mathbf{u}.\neg$)
- 3: $Q \leftarrow \{(\mathbf{u}, \Psi)\}$; $D \leftarrow \emptyset$
- 4: $\Pi' \leftarrow$ empty string of quantifiers ▷ Π' is a **global** variable
- 5: **while** $(\mathbf{x}, \varphi) \leftarrow \text{pop}(Q)$ **do**
- 6: **if** \mathbf{x} is empty **then** add φ to D
- 7: **else if** some $x \in \mathbf{x}$ does not appear in φ **then** add pair $(\mathbf{x} \setminus \{x\}, \varphi)$ to Q
- 8: **else if** $\exists x.\varphi$ is in \mathcal{F} for some $x \in \mathbf{x}$ **then** add pair $(\mathbf{x} \setminus \{x\}, \exists x.\varphi)$ to Q
- 9: **else if** some $x \in \mathbf{x}$ appears only linearly in φ **then** add $\text{PresQE}(x, \mathbf{x}, \varphi)$ to Q
- 10: **else** add $\text{Linearize}(\text{SemCover}(\mathbf{x}, \varphi))$ to Q
- 11: $\Phi \leftarrow \Pi.\Pi'. \bigvee_{\varphi \in D} \varphi$
- 12: **if** Π contains no quantifiers **then return** Φ

■ **Algorithm 2** Function Linearize.

Input: a set S of pairs (\mathbf{x}, θ) , with \mathbf{x} a vector of variables and θ a formula

Output: if $\mathcal{F} = \mathcal{QF}$: for each (\mathbf{x}, θ) , a pair (\mathbf{x}, θ') where $\theta \Leftrightarrow \theta'$ and, for every $x \in \mathbf{x}$,
 if $2^{|x|}$ only occurs in constraints from \mathcal{PowCmp} in θ , then x only occurs linearly in θ'

- 1: **if** $\mathcal{F} = \mathcal{Sem}$ **then return** S ▷ do nothing unless $\mathcal{F} = \mathcal{QF}$
- 2: **for** $(\mathbf{x}, \theta) \in S$ **do**
- 3: $\mathbf{x}' \leftarrow$ vector of all $x \in \mathbf{x}$ s.t. $2^{|x|}$ only occurs in constraints from \mathcal{PowCmp} in θ
- 4: **for** $x \in \mathbf{x}'$ **do**
- 5: update θ by applying all of the following replacements:
- 6: $a \cdot 2^{|x|} < b \mapsto \begin{cases} |x| < \lceil \log_2(b/a) \rceil & \text{if } a > 0 \text{ and } b > 0 \\ |x| > \lfloor \log_2(b/a) \rfloor & \text{if } a < 0 \text{ and } b < 0 \end{cases}$
- 7: $a \cdot 2^{|x|} < b \cdot 2^{|y|} \mapsto \begin{cases} |x| < |y| + \lceil \log_2(b/a) \rceil & \text{if } a > 0 \text{ and } b > 0 \\ |x| > |y| + \lfloor \log_2(b/a) \rfloor & \text{if } a < 0 \text{ and } b < 0 \end{cases}$
- 8: $q \mid 2^{|x|} - r \mapsto \begin{cases} q' \mid |x| - r' & \text{if } r' = \min\{s \geq 0 : q \mid 2^s - r\}, \\ & q' = \min\{t > 0 : q \mid r \cdot (2^t - 1)\} \\ |x| = r' & \text{if } r' = \min\{s \geq 0 : q \mid 2^s - r\}, \\ & \{t > 0 : q \mid r \cdot (2^s - 1)\} = \emptyset \\ \perp & \text{otherwise, i.e., } \{s \geq 0 : q \mid 2^s - r\} = \emptyset \end{cases}$
- 9: ▷ in the replacements in line 8, search for $s, t \leq q - 1$ only
- 9: **return** S

Algorithm 3 Function PresQE.

Input: variable x ; vector of variables \mathbf{x} , where $x \in \mathbf{x}$;

 formula $\varphi(x, \mathbf{y})$ of \mathcal{F} where $\mathbf{x} \setminus \{x\} \subseteq \mathbf{y}$ and x appears only linearly in atomic formulae

Output: a set of pairs $(\mathbf{x}, \psi(\mathbf{y}))$ where $\psi \in \mathcal{F}$ and the set of all ψ is a cover for $\exists x.\varphi$

- 1: $T \leftarrow \{(a, -t(\mathbf{y})) : a > 0, a \cdot x + t \in \text{hom}(\varphi)\} \cup \{(-a, t(\mathbf{y})) : a < 0, a \cdot x + t \in \text{hom}(\varphi)\}$
 - 2: $g \leftarrow \prod\{a : (a, t) \in T \text{ for some } t\}$ ▷ product of all elements of non-empty set
 - 3: $\Gamma \leftarrow \{\varphi[\frac{t+k}{a}/x] \wedge (a \mid t+k) : (a, t) \in T, k \in [-r, r] \text{ where } r := a \cdot (2 \cdot \|\text{lin}(\varphi)\| + g \cdot \text{mod}(\varphi))\}$
 - 4: **return** $\{(\mathbf{x}, \psi) : \psi \in \text{SimplifyDiv}(\Gamma), \gamma \in \Gamma\}$
-

Algorithm 4 Function SemCover.

Input: vector \mathbf{x} of variables; formula $\varphi(\mathbf{x}, \mathbf{z})$ of \mathcal{F} , containing $2^{|\mathbf{x}|}$ for each $x \in \mathbf{x}$
Output: a set of pairs $(\mathbf{x}, \psi(\mathbf{x}, \mathbf{z}))$, where $\psi \in \mathcal{F}$ and the set of all $\Pi'. \exists \mathbf{x}.\psi$ covers $\Pi'. \exists \mathbf{x}.\varphi$;
 in every ψ some $2^{|\mathbf{x}|}$ ($x \in \mathbf{x}$) only occurs in constraints from \mathcal{PowCmp}
Side effect: update global variable Π' (string of quantifiers)

- 1: **for** $x \in \mathbf{x}$ **do**
 - 2: $I \leftarrow$ set of inequalities in φ outside \mathcal{PowCmp} in which x appears as a power
 - 3: $H \leftarrow \{(\eta, \sigma) : \eta(\mathbf{x}) + \sigma(\mathbf{z}) + c < 0 \text{ in } I; \eta \text{ and } \sigma \text{ homogeneous}\}$
 - 4: $\Gamma_x \leftarrow \{\varphi\}$
 - 5: **for** $(\eta, \sigma) \in H$ **do**
 - 6: $A \leftarrow$ subset of I with these η and σ (only c varies)
 - 7: $2^g \leftarrow 2^7 \cdot (\lambda(\|\eta\|_1 + \max\{c : (\eta + \sigma + c < 0) \in A\}))^2$
▷ factors up to 2^g are considered “small”
 - 8: $a \leftarrow$ coefficient at $2^{|\mathbf{x}|}$ in η
 - 9: $V \leftarrow$ variables in η except x
 - 10: $\beta \leftarrow 2^{|\mathbf{x}|} > 2^g \wedge (\bigwedge_{u \in V} 2^{|\mathbf{x}|} > 2^g \cdot 2^{|u|})$
 - 11: $\Gamma_x \leftarrow \{2^{|\mathbf{x}|} = 2^j \wedge \gamma[\alpha[2^j / 2^{|\mathbf{x}|}] / \alpha : \alpha \in A],$
 - 12: $2^{|\mathbf{x}|} > 2^g \wedge 2^{|\mathbf{x}|} = 2^j \cdot 2^{|\mathbf{v}|} \wedge \gamma[\alpha[2^j \cdot 2^{|\mathbf{v}|} / 2^{|\mathbf{x}|}] / \alpha : \alpha \in A],$
 - 13: $\beta \wedge \lambda(a) \cdot 2^{|\mathbf{x}|} < \lambda(\sigma) \wedge \sigma < 0 \wedge \gamma[\top / \alpha : \alpha \in A],$
 - 14: $\beta \wedge \lambda(a) \cdot 2^{|\mathbf{x}|} < \lambda(\sigma) \wedge \sigma \geq 0 \wedge \gamma[\perp / \alpha : \alpha \in A],$
 - 15: $\beta \wedge \lambda(a) \cdot 2^{|\mathbf{x}|} = \lambda(\sigma) \wedge \gamma[\alpha[\frac{\lambda(\sigma)}{\lambda(a)} / 2^{|\mathbf{x}|}] / \alpha : \alpha \in A],$
 - 16: $\beta \wedge \lambda(a) \cdot 2^{|\mathbf{x}|} = 2 \cdot \lambda(\sigma) \wedge \gamma[\alpha[\frac{2 \cdot \lambda(\sigma)}{\lambda(a)} / 2^{|\mathbf{x}|}] / \alpha : \alpha \in A],$
 - 17: $\beta \wedge \lambda(a) \cdot 2^{|\mathbf{x}|} > 2 \cdot \lambda(\sigma) \wedge a < 0 \wedge \gamma[\top / \alpha : \alpha \in A],$
 - 18: $\beta \wedge \lambda(a) \cdot 2^{|\mathbf{x}|} > 2 \cdot \lambda(\sigma) \wedge a > 0 \wedge \gamma[\perp / \alpha : \alpha \in A]$
 - 19: : $\gamma \in \Gamma_x, 0 \leq j \leq g, v \in V\}$
 - 20: $\Gamma \leftarrow \bigcup_{x \in \mathbf{x}} \{(\bigwedge_{y \in \mathbf{x}} 2^{|\mathbf{x}|} \geq 2^{|\mathbf{y}|}) \wedge \gamma : \gamma \in \Gamma_x\}$ ▷ we next remove all occurrences of λ
 - 21: $\Sigma \leftarrow \{\sigma : \lambda(\sigma) \text{ is a subterm of some } \gamma \in \Gamma\} \setminus \{0\}$
 - 22: **for** $\sigma \in \Sigma$ **do**
 - 23: **if** $\forall w_\sigma$ is not in Π' **then**
 - 24: $w_\sigma \leftarrow$ fresh variable; add $\forall w_\sigma$ to Π' ▷ update **global** Π'
 - 25: $\Theta \leftarrow \{(\sigma \neq 0 \wedge \neg(2^{|w_\sigma|} \leq |\sigma| < 2 \cdot 2^{|w_\sigma|})) : \sigma \in \Sigma\}$
 - 26: **for** each $\Sigma' \subseteq \Sigma$ and each $\gamma \in \Gamma$ **do**
 - 27: add to Θ the following formula:
 - 28: $(\bigwedge_{\sigma \in \Sigma'} 2^{|w_\sigma|} \leq |\sigma| < 2 \cdot 2^{|w_\sigma|}) \wedge (\bigwedge_{\sigma \in \Sigma \setminus \Sigma'} \sigma = 0) \wedge \gamma[2^{|w_\sigma|} / \lambda(\sigma) : \sigma \in \Sigma'] [0 / \lambda(\sigma) : \sigma \in \Sigma \setminus \Sigma']$
 - 29: **return** $\{(\mathbf{x}, \theta) : \theta \in \Theta\}$
-

Algorithm 5 Function SimplifyDiv.

Input: formula φ *almost* in \mathcal{F} : may contain non-simple divisibility constraints

Output: a cover for φ of formulae from \mathcal{F} , in which all divisibility constraints are simple

- 1: $G \leftarrow$ set of non-simple divisibilities in φ
 - 2: $d \leftarrow$ least common multiple of all divisors in G
 - 3: $\mathbf{t} \leftarrow$ all variables x and powers $2^{|y|}$ appearing in G
 - 4: $\Gamma \leftarrow \emptyset$
 - 5: **for** $r \in [\mathbf{t} \rightarrow [d]]$ **do**
 - 6: add $((\bigwedge_{t \in \mathbf{t}} d \mid t - r(t)) \wedge \varphi[r(\alpha) / \alpha : \alpha \in G])$ to Γ
 - 7: **where** $r(q \mid \sum_{i=1}^n a_i \cdot t_i + c) := q \mid \sum_{i=1}^n a_i \cdot r(t_i) + c$ \triangleright simplifies to \top or \perp
 - 8: **return** Γ
-

an existentially quantified DNF, or equivalently a disjunction of existentials. It suffices to guess one disjunct, corresponding to one element of Q that is satisfiable. Thus, we will replace a deterministic inner loop that maintains a set of pairs in Q with a non-deterministic algorithm that maintains a single pair from Q . In the deterministic interpretation, calls to the subroutines in lines 9 and 10 replace a single element of Q with a set of pairs. In the non-deterministic interpretation, we guess one pair in the output of the subroutine as the new element of Q .

Subroutines. We turn from the Master procedure to its subroutines. The core of the subroutine PresQE (Algorithm 3) corresponds to Weispfenning’s quantifier elimination for $\mathcal{P}a$ [25], while Linearize given in Algorithm 2 is a simple procedure taking $\mathcal{P}ow\mathcal{C}mp$ atomic formulae like $2^{|x|} < 2^{|y|}$ and transforming them to Presburger formulae $x < y$ by “taking logs”. We remark that all three types of divisibility replacements in line 8 of Linearize are possible: e.g., $(7 \mid 2^{|x|} - 4) \mapsto (3 \mid |x| - 2)$, and $(20 \mid 2^{|x|} - 2) \mapsto (|x| = 1)$, and $(6 \mid 2^{|x|} - 3) \mapsto \perp$.

The SemCover subroutine (Algorithm 4) is a variation of procedures dating back to Semenov’s [23]. This will be less familiar to most readers, and so we discuss it in detail here.

The purpose of subroutine SemCover is to ensure that, in each of the pairs (\mathbf{x}, ψ) in its output, for some variable $x \in \mathbf{x}$ every occurrence is either linear or in an atomic formula from $\mathcal{P}ow\mathcal{C}mp$. Across all outputs, the identity of the variable x may differ. Thus, the subroutine is essentially “ $\mathcal{P}ow\mathcal{C}mp$ -ifying” the formula. The significance of converting atomic formulae to $\mathcal{P}ow\mathcal{C}mp$ is that powers can then be eliminated by just “taking logarithms”, i.e., by invoking Linearize. And once a quantified variable is so heavily processed that it occurs only linearly (outside powers), then by applying standard Presburger arithmetic quantifier elimination, we can eliminate the variable completely using PresQE.

To be more precise about how SemCover assists the Master procedure, consider what happens when (\mathbf{x}, ψ) from the output of SemCover gets popped from Q in the Master procedure. Our actions depend on the chosen fragment (unless x is eliminated from ψ entirely, in which case line 7 takes care of it).

If $\mathcal{F} = \mathcal{S}em$: for some $x \in \mathbf{x}$, we can move $\exists x$ into ψ while still staying in the fragment, since x occurs only in power comparisons (line 8).

If $\mathcal{F} = \mathcal{Q}\mathcal{F}$: all occurrences of x became linear after the execution of Linearize on the output of SemCover, so the variable x can be eliminated by PresQE (line 9).

A look inside subroutine *SemCover*. Intuitively, the overall workflow of the Master procedure is repeated processing of atomic formulae lying within the scope of a particular block of quantifiers. The constraints we process will be those containing “problematic quantified variables”: those that appear in atomic formulae involving powers, but are outside the fragment *PowCmp*. We exhibit the idea using the following subformula:

$$\exists x. \exists y. \quad 3 \cdot 2^{|x|} - 5 \cdot 2^{|y|} - z < 0. \quad (1)$$

Here both x and y are problematic within the sole atomic subformula of the quantified formula. A major component of all prior procedures is to replace such a formula with a quantified DNF corresponding to a case analysis on the relative values of the problematic variables. These cases correspond to lines 11 to 19 of Algorithm 4 and are a cover for the formula under analysis, hence the name “Semenov cover” given to the algorithm. Each case is defined by a *PowCmp* “guard” and, under the assumption specified in a given case, we will be able to eliminate one problematic variable within a constraint, without introducing new problematic existentially-quantified variables. Thus, by applying the procedure repeatedly, we can expunge all problematic quantified variables.

The case analysis includes a guess as to which existentially quantified variable is the largest. In the example, one such guess is that $2^{|x|}$ is the largest. In all the subcases for this guess, we will make x unproblematic. The Semenov cover breaks up this guess into several subcases. One subcase is where $2^{|x|}$ is not much bigger than one of the other power terms, say $2^{|x|} = 4 \cdot 2^{|y|}$. In such cases we can substitute $2^{|x|}$ by a constant multiple of the other term, where the constant is itself a power of 2. Returning to the subcase mentioned just above, where $2^{|x|} = 4 \cdot 2^{|y|}$, we can replace $2^{|x|}$ by $4 \cdot 2^{|y|}$. The remaining case is where $2^{|x|}$ is significantly bigger than all other power terms like $2^{|y|}$; the threshold for “significantly” is set by line 7 of Algorithm 4. In this case we further analyze the most significant digit of the binary expansion for each term.

► **Definition.** For any integer N , let $\lambda(N)$ denote the *highest power of 2 below* $|N|$ ¹; we have $\lambda(0) = 0$ and $\lambda(N) \leq |N| < 2\lambda(N)$.

Algorithm 4 will make use of intermediate terms that contain λ 's – for example, $\lambda(\sigma)$ for σ an ordinary *Pa*($\lambda x. 2^{|x|}$) term. The semantics of such terms is the obvious one, which could be formalized by translation into *Pa*($\lambda x. 2^{|x|}$), where the function λ is definable.

Returning to the example, our “significantly bigger” hypothesis implies that

$$\lambda(3 \cdot 2^{|x|} - 5 \cdot 2^{|y|}) = \lambda(3 \cdot 2^{|x|}) = 2^{x+1}.$$

This equality in turn implies that, when $\lambda(3 \cdot 2^{|x|})$ is strictly below $\lambda(z)$, the corresponding inequality in Equation (1) is true: $\lambda(3 \cdot 2^{|x|} - 5 \cdot 2^{|y|})$ is strictly below $\lambda(z)$, and while each term can differ from the corresponding λ , the difference cannot be large enough to make the inequality go the other way. By a similar argument, in the subcase where $\lambda(3 \cdot 2^{|x|})$ is at least four times greater than $\lambda(z)$, the inequality must be false. Here we reason that if $\lambda(3 \cdot 2^{|x|} - 5 \cdot 2^{|y|})$ is at least four times greater than $\lambda(z)$, then the offset of each term from its λ value cannot change the inequality from true to false.

This leaves some subcases where $\lambda(3 \cdot 2^{|x|})$ is close to $\lambda(z)$, and in these cases we can substitute away $2^{|x|}$ as well. For example, in the subcase where $\lambda(3 \cdot 2^{|x|}) = \lambda(z)$, we note that $\lambda(3 \cdot 2^{|x|}) = 2 \cdot 2^{|x|}$, and thus we could replace $2^{|x|}$ with $\lambda(z)/2$. By multiplying through the inequality by 2, we can eliminate the division by 2.

¹ We will be mostly concerned with this function on positive integers, but using absolute values gives us the convenience of avoiding partial functions.

Using the output of SemCover. The procedure above removed x , but there are several caveats. Firstly, each case was associated with a condition, where the problematic variable x still appears! However, these conditions are in \mathcal{PowEmp} , and therefore all occurrences of x are now unproblematic. Secondly, in some of our substitutions to eliminate x , we introduced λ terms, which appear both in the condition describing the case and in the formula obtained by substituting (assuming the condition). One solution to this problem, applied in earlier procedures such as Point’s [19], is to extend the signature with several functions such as λ , and declare that such conditions are acceptable. In this way one can obtain quantifier elimination in the extended signature. In our SemCover subroutine, we proceed slightly differently, eliminating λ terms in favor of new variables that are bound by *definitional quantifiers*. That is, the new variables are associated with additional conditions which define them from the free variables. For example, $\lambda(z)$ can be replaced by 2^w , with additional conditions $2^w \leq |z| < 2 \cdot 2^w$. There is a unique such w for a given z , so the quantification over w can be thought of simultaneously as an existential conjoined with this condition and as a universal relativized to this condition. Such quantifications take us out of \mathcal{PowEmp} . But when considered as leading universal quantifiers, they will not increase the quantifier alternation of the global formula – they will add on variables to the next quantified block considered in the Master procedure. And since Algorithm 1 will process from inner quantifier blocks outward, the fattening of outer quantifier blocks does not jeopardize termination of our procedure. Note that at the end of processing quantifier blocks outward with Algorithm 1, we will have only an outermost block of definitional quantifiers; if there are no free variables in the top-level input formula Φ , the quantified variables will depend only on constants, and thus can be replaced by numbers, leading to a quantifier-free sentence.

Analysis of the core procedure. We analyze the procedure, showing in particular that each of Algorithms 1–5 correctly implements its specification. In the sequel we will also need the following facts.

► **Lemma 3.** *All divisibility constraints in $D \cup \{\varphi : (\mathbf{x}, \varphi) \in Q \text{ for some } \mathbf{x}\}$ are simple.*

► **Lemma 4.** *The Master procedure always terminates and, on a formula $\Phi(\mathbf{y})$, returns an equivalent formula $\Phi'(\mathbf{y})$ such that:*

- $\Phi'(\mathbf{y})$ is equal to either $\exists \mathbf{w}.\varphi'(\mathbf{w}, \mathbf{y})$ or $\neg \exists \mathbf{w}.\varphi'(\mathbf{w}, \mathbf{y})$, where $\varphi' \in \mathcal{F}$,
- if Φ is a sentence, then $\Phi' \in \mathcal{F}$ (in other words, if \mathbf{y} is empty, then \mathbf{w} is empty).

In fact, Φ' starts with \neg iff the outermost quantifier block of Φ is existential.

► **Lemma 5.** *Consider a prenex formula $\Pi.\Phi$, with Φ from \mathcal{F} , in which all variables from the quantifier prefix Π appear only linearly. When running the Master procedure on Φ , SemCover is never invoked. Moreover, if no variable in Φ occurs in a power term (i.e., it is a formula from \mathcal{Pa}), then the quantifier-free formula returned by the procedure is in \mathcal{Pa} .*

5 Decision procedures and their complexity

In this section, we provide our top-level decision procedures, which make use of the algorithms presented in Section 4. We then provide a complexity analysis that establishes Theorems 1 and 2. To simplify the exposition, the growth of the formulae returned by the procedure is described with the help of “parameter tables” having the following shape:

112:12 The Complexity of Presburger Arithmetic with Power or Powers

	$p_1(\cdot)$	$p_2(\cdot)$	\dots	$p_n(\cdot)$
φ	a_1	a_2	\dots	a_n
ψ_1	$f_{1,1}(a_1, \dots, a_n)$	$f_{1,2}(a_1, \dots, a_n)$	\dots	$f_{1,n}(a_1, \dots, a_n)$
\dots	\dots	\dots	\dots	\dots
ψ_m	$f_{m,1}(a_1, \dots, a_n)$	$f_{m,2}(a_1, \dots, a_n)$	\dots	$f_{m,n}(a_1, \dots, a_n)$

In this table, $\varphi, \psi_1, \dots, \psi_m$ are formulae, $p_1(\cdot), \dots, p_n(\cdot)$ are parameter functions from formulae to \mathbb{N} , $a_1, \dots, a_n \in \mathbb{N}_+$, and all $f_{j,k}$ are functions from \mathbb{N}^n to \mathbb{N} . The table states that

if $p_i(\varphi) \leq a_i$ for all $i \in [1, n]$, then $p_k(\psi_j) \leq f_{j,k}(a_1, \dots, a_n)$ for all $j \in [1, m]$ and $k \in [1, n]$.

We sometimes assume lower bounds on the values a_1, \dots, a_n (e.g., $h \geq 2$ and $a \geq 2$ in the table of Lemma 6) in order to simplify the definition of the functions $f_{j,k}$. Note that this does not change the semantics of the table. We sometimes write the ditto mark " inside a cell of the table. In that case, the ditto mark represents the value of the cell directly above it (e.g., the rightmost " appearing in the table of Lemma 6 is short for $b + 2 \cdot v + 1$).

Theorem 1: NExpTime upper bound for existential $\mathcal{P}a(\lambda x. 2^{|x|})$

Before arguing for a NEXPTime decision procedure for $\exists \mathcal{P}a(\lambda x. 2^{|x|})$, we analyze the growth of formulae resulting from calls to PresQE, SemCover and Linearize.

Our analysis of PresQE simply merges the analysis of Weispfenning's quantifier elimination for Presburger arithmetic from [25] (implemented in lines 1 to 3 of PresQE) with an analysis of SimplifyDiv. Here are the resulting bounds:

► **Lemma 6.** *On input $(x, \mathbf{x}, \varphi(\mathbf{x}, \mathbf{z}))$ where x only occurs linearly in φ , PresQE returns a set $\{(\mathbf{x}, \psi_1), \dots, (\mathbf{x}, \psi_k)\}$ whose formulae satisfy the parameter table below ($i \in [1, k]$):*

	#hom	heft	$\ \text{hom}(\cdot)\ $	$\ \text{lin}(\cdot)\ $	mod	\mathcal{B}
φ	$h \geq 2$	v	$a \geq 2$	c	m	b
ψ_i	h	$2 \cdot v$	$2 \cdot a^2$	$a^{h+2}(c + m)$	$a \cdot m$	$b + 2 \cdot v + 1$
$\bigvee_{j=1}^k \psi_j$	h^2	"	"	"	$a^h m$	$k(\text{"} + 1)$

and $k \leq h \cdot c \cdot m^{2v+1} \cdot a^{2v+h+4}$. The running time is in $(\text{len}(\varphi) \cdot c \cdot m)^{\text{poly}(h,v)}$.

A simple analysis of SemCover yields the following bounds.

► **Lemma 7.** *Let $\Theta = \{(\mathbf{x}, \theta_1), \dots, (\mathbf{x}, \theta_k)\}$ be the output of SemCover($\mathbf{x}, \varphi(\mathbf{x}, \mathbf{z})$), where $\mathbf{x} = (x_1, \dots, x_n)$ with $n \geq 1$. Then, the following parameter table holds, where $i \in [1, k]$:*

	#hom	heft	$\ \text{lin}(\cdot)\ $	mod	\mathcal{B}
φ	h	$v \geq 2$	$c \geq 2$	m	b
θ_i	$h \cdot (v + 10) + n$	v	$2^{11} \cdot v^2 \cdot c^4$	m	$b + h \cdot (v + 11) + n$
$\bigvee_{j=1}^k \theta_j$	$h \cdot 2^8 \cdot v^3 \log(c) + n^2$	"	"	"	$k \cdot (\text{"} + 1)$

where $k \leq (v + 1)^{8h} \cdot \log(c)^h \cdot n$. Moreover, (i) at most h universal quantifiers are added to the global variable Π' ; (ii) the running time is in $(\text{len}(\varphi) \cdot n)^{\text{poly}(h)}$; and (iii) for every $i \in [1, k]$ there are at most h terms $t \in \text{hom}(\theta_i)$ that contain some variable from \mathbf{x} and satisfy $(t + c' < 0) \in \text{lin}(\theta_i)$ with $t + c' < 0$ not in PowCmp , for some $c' \in \mathbb{Z}$.

Note that an estimate of $\|hom(\theta_i)\|$ is missing from Lemma 7. For SemCover, this parameter grows similarly to $\|lin(\theta_i)\|$, which by definition always bounds $\|hom(\theta_i)\|$. We also note that Lemma 7 gives an upper bound on the number of global variables added to Π' . This bound is later required to analyze $\mathcal{P}\mathbf{a}(2^{\mathbb{N}}(\cdot))$, but is not needed in the context of deciding sentences from $\exists\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$. Indeed, from Lemma 4, in this latter case Π' is empty.

In computing the upper bounds on $\|hom(\theta_i)\|$ and $\|hom(\bigvee_{j=1}^k \theta_j)\|$ keep in mind that in lines 11 to 18 of SemCover we perform “tailored substitutions”: we only replace $2^{|x|}$ in linear terms $\alpha \in A$ with either a constant, a unique (given α) expression $\lambda(\sigma)$, or a multiple of a power $2^{|y|}$, where y appears in α . Our analysis tracks the impact of iterating these types of replacements on the number of homogeneous terms.

We continue by analyzing Linearize. Here the bounds are quite simple, but one observation is in order: line 8 might require iterating through the $q - 1$ residue classes of q in order to find suitable q' and r' . Since q is encoded in binary, this yields an exponential running time for Linearize (see m below), as shown in the following lemma.

► **Lemma 8.** *Consider a set $S = \{(\mathbf{x}, \theta_1), \dots, (\mathbf{x}, \theta_k)\}$ where $\mathbf{x} = (x_1, \dots, x_n)$, and let r be the maximum number of variables appearing in some θ_i . On input S , Linearize returns a set $\{(\mathbf{x}, \theta'_1), \dots, (\mathbf{x}, \theta'_k)\}$ with bounds as in the following table, for all $j \in [1, k]$:*

	$\#hom$	$heft$	$\ hom(\cdot)\ $	$\ lin(\cdot)\ $	mod	\mathcal{B}
θ_j	h	v	a	$c \geq 2$	$m \geq 2$	b
θ'_j	$h + (6 \cdot r + 2) \cdot n$	v	a	c	m^2	$22 \cdot b$

The running time is in $\text{poly}(\max_{i=1}^k \text{len}(\theta_i), m, n, \#S)$.

We now complete the description of the non-deterministic algorithm deciding $\exists\mathcal{P}\mathbf{a}(\lambda x.2^{|x|})$ in NEXPTIME, completing the informal comments in Section 4. As a preliminary step, the algorithm runs SimplifyDiv on the matrix of the input existential sentence Φ , guessing a residue class for each variable and power, and obtaining an existential sentence where all the divisibilities are simple. The algorithm puts that sentence in prenex form. Afterwards, the algorithm follows Algorithm 1 (with $\mathcal{F} = \mathcal{Q}\mathcal{F}$), but replaces $\text{pop}(Q)$ in line 5, as well as other forms of iteration inside PresQE and SemCover, with non-deterministic guesses. More precisely, when SemCover is called, it guesses a variable $x \in \mathbf{x}$ in line 1, iterates (deterministically) over every $(\eta, \sigma) \in H$ in line 5, and guesses only one of the cases in lines 11 to 18. As a result, in the non-deterministic version of SemCover, the variable Γ in line 20 contains a single formula γ . Since Φ is an existential sentence, the various terms σ considered by SemCover are always 0. Hence, $\lambda(\sigma)$ is 0 and lines 21 to 28 have no effect on the subroutine, which simply returns a singleton set containing the pair (\mathbf{x}, γ) . For PresQE, non-deterministic guesses replace the iterations done in line 3, as well as the ones performed in line 5 of SimplifyDiv (as done in the aforementioned preliminary step of the algorithm).

The non-deterministic versions of PresQE and SemCover described above always return singleton sets containing a pair of the form (\mathbf{x}, θ) . Then, by the correctness of PresQE and SemCover, we conclude that on an input sentence Φ containing n quantified variables, the non-deterministic version of Algorithm 1 never calls each of the subroutines PresQE, SemCover and Linearize more than n times. By looking at the bounds on ψ_i , θ_i and θ'_j from Lemmas 6–8 we conclude that these $3n$ subroutine calls (non-deterministically) return a formula that never requires more than exponential space to be represented. Since Φ is a sentence and $\mathcal{F} = \mathcal{Q}\mathcal{F}$, the non-deterministic algorithm will eventually obtain a formula Ψ with no variables, only constants, which can be evaluated in exponential time. As usual, the algorithm returns true if one such (non-deterministically derived) formula Ψ is valid.

Theorem 2: 3ExpTime upper bound for $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$

We now move to $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$. Let Φ be obtained by translating a sentence of $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$ into a prenex sentence of $\mathcal{Pa}(\lambda x.2^{|x|})$ without divisibility constraints (i.e., replace each $2^{\mathbb{N}}(x)$ with $\exists y. x = 2^{|y|}$, each $q \mid t$ with $\exists z. t = q \cdot z$, and bring the resulting sentence in prenex form). Note that this translation is in polynomial time, and that each variable in Φ appears either always linearly or always in a power. The algorithm to decide Φ is described below:

- 1: $\Psi_1 \leftarrow$ run Algorithm 1 with $\mathcal{F} = \mathcal{Sem}$, on Φ \triangleright as Φ is a sentence, $\Psi_1 \in \mathcal{PowCmp}$.
- 2: $\Pi.\Psi_2(\mathbf{x}) \leftarrow$ prenex form of Ψ_1 $\triangleright \Psi_2$ q.f.; \mathbf{x} are the variables appearing in Π
- 3: $\{(\mathbf{x}, \Psi_3)\} \leftarrow$ Linearize($\{(\mathbf{x}, \Psi_2)\}$) $\triangleright \Psi_3(\mathbf{x})$ belongs to \mathcal{Oct}
- 4: $\Omega \leftarrow$ run Algorithm 1 with $\mathcal{F} = \mathcal{QF}$, on $\Pi.\Psi_3$ $\triangleright \Omega$ does not contain variables
- 5: evaluate truth of Ω

Above we highlight the fact that, after the first invocation to Algorithm 1, we obtain a formula from \mathcal{PowCmp} which is then manipulated by Linearize into a formula from integer octagon arithmetic (\mathcal{Oct}). Then, in order to estimate the running time of this algorithm, we need to study the running time of Algorithm 1 on inputs that either come from $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$ or are from \mathcal{Oct} . Let us discuss the latter case first.

Since \mathcal{Oct} is a fragment of Presburger arithmetic, line 4 above fundamentally runs Weispfenning's quantifier elimination procedure for Presburger arithmetic (see Lemma 5), plus calls to SimplifyDiv. It turns out that on formulae from \mathcal{Oct} , this procedure only runs in exponential time, as summarized in the following proposition.

► **Proposition 9.** *Let $\mathcal{F} = \mathcal{QF}$. Consider a formula $\Phi(\mathbf{z})$ from integer octagon arithmetic (\mathcal{Oct}) in prenex form and having $alt(\varphi) = \ell \geq 1$ quantifier blocks, each with $n \geq 1$ many variables. On input Φ , Algorithm 1 returns a formula Ψ with bounds:*

	#hom	heft	$\ hom(\cdot)\ $	$\ lin(\cdot)\ $	mod	\mathcal{B}
Φ	$h \geq 2$	2	1	$c \geq 2$	$m \geq 2$	b
Ψ	$4 \cdot \#z^2$	2	1	$4^{\ell \cdot n}(c + 2 \cdot m)$	m	$k(b + \ell \cdot n + 1)$

and $k \leq 2^{2^5 \ell^2 n^2} (\#z^2 \cdot c \cdot m)^{\ell \cdot n}$. The running time of the procedure is in $(len(\Phi) \cdot c \cdot m)^{\text{poly}(\ell, n)}$.

The proof of this proposition is by induction on $alt(\varphi)$, and essentially follows the standard arguments to bound the running time of the quantifier elimination procedure for Presburger arithmetic. The key ingredient that leads to the bounds above is that, for \mathcal{Oct} , the natural numbers a in line 3 and g in line 2 of PresQE are always 1. This has two effects. Firstly, it shows that \mathcal{Oct} admits quantifier elimination, i.e., while running the procedure no atomic formulae outside \mathcal{Oct} can arise. This is best witnessed by looking at line 3 in PresQE. There, the divisibility constraints $a \mid t + k$ are trivially satisfied, and we are replacing x with a term of the form $\pm y + c$ for some $c \in \mathbb{Z}$. From these substitutions, only constraints from \mathcal{Oct} or constraints of the form $\pm 2 \cdot y < b$ can arise, and the latter are normalized to $y \leq \lfloor \frac{b-1}{2} \rfloor$ or $y \geq \lceil \frac{1-b}{2} \rceil$ as explained in Section 2. The second effect is on the growth of the constants. The variable r in line 3 only depends on $mod(\Phi)$, which now does not grow during the procedure, and on $\|lin(\Phi)\|$, which grows only exponentially in the number of variables in Φ .

We now move to the running time of Algorithm 1 on inputs that come from $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$. The properties of this procedure are summarized in the next proposition.

► **Proposition 10.** *Let $\mathcal{F} = \mathit{Sem}$. Let $\Phi(\mathbf{y})$ be a formula from $\mathcal{Pa}(\lambda x.2^{|x|})$ in prenex normal form, with no divisibility constraints, and in which each quantified variable appears either only linearly or only in powers. Suppose Φ has $\mathit{alt}(\Phi) = B$ quantifier blocks, each with at most $L \geq 1$ variables occurring linearly and each block having at most $E \geq 1$ variables occurring in powers. On input Φ , Algorithm 1 returns a formula Ψ with bounds as in the following table:*

	$\#hom$	$heft$	$\ \mathit{lin}(\cdot)\ $	mod	\mathcal{B}
Φ	$h \geq 2$	$v \geq 2$	$c \geq 4$	$m \geq 4$	b
Ψ	$H := (E \cdot h \cdot \log(c \cdot m))^{(2 \cdot v)^{2^5 \cdot L \cdot B^3}}$	$2^{B \cdot L \cdot v}$	2^H	2^H	$b \cdot 2^H$

and the number of quantifiers added to Π' is at most H . The running time of the procedure is in $\mathit{len}(\Phi) \uparrow (E \cdot h \cdot \log(c \cdot m)) \uparrow v \uparrow \mathit{poly}(L, B)$.

In the above proposition, $a \uparrow b := a^b$ is the exponentiation function and, following Knuth's up-arrow notation, it is right-associative. In view of our bounds for one iteration of $\mathit{SemCover}$ given in Lemma 7, the bound on $\#hom(\Psi)$ should seem somewhat surprising. We know from the correctness of $\mathit{SemCover}$ that, after a call to $\mathit{SemCover}$, one of the variables appearing in powers will only occur in constraints from $\mathit{PowComp}$. Since all these variables need to have this property before moving to the next quantifier block, $\mathit{SemCover}$ must be chained at least E times within a block, E being the number of variables occurring in powers in the block. Then, from the bound $\#hom(\theta_j) \leq h \cdot (v + 10) + n$ in Lemma 7, one might expect $\#hom(\Psi)$ to be roughly $h \cdot v^E$, thus exponential in E even for a single block of quantifiers. Proposition 10, however, proves otherwise: $\#hom(\Psi)$ is only polynomial in E . For this, we need to sharpen the correctness statement for $\mathit{SemCover}$.

We already know that in each output formula some variable x is made unproblematic, and no new problematic occurrences of variables are created. We prove a stronger statement: namely, that every *rewriting* of a homogeneous term (by a substitution in lines 11–18) makes the number of problematic variables in that term decrease. This is possible thanks to the tailoring of these substitutions to individual inequalities, which allows us to track the evolution of each term independently of the rest of the formula. As a result:

- a formula that is placed into D at the end of processing a single quantifier block can be a result of E chained calls to $\mathit{SemCover}$, but
- the number of calls to $\mathit{SemCover}$ that rewrite an individual term during its evolution is bounded by the $heft$.

Therefore, we can iterate the above-mentioned bound $\#hom(\theta_j) \leq h \cdot (v + 10) + n$ just $2^L \cdot v$ times instead of E times. Thus $\#hom(\Psi)$ in Proposition 10 is found to be exponential in v and only polynomial in E . We remark that if $\#hom(\Psi)$ were instead found to be exponential in E , then the algorithm would not have any hope of running in elementary time. This is because, after a block of quantifiers is considered, E increases by the number of variables introduced in $\mathit{SemCover}$, which from Lemma 7 is roughly the number of homogeneous terms.

To prove Theorem 2 it suffices to chain the bounds and running times of Proposition 10, Lemma 8 and Proposition 9, according to the algorithm given at the beginning of the section.

Avoiding quadruply exponential numbers. It may not be immediately evident from the bounds in the various tables why we do not perform quantifier elimination eagerly and instead run Algorithm 1 without fully eliminating quantifiers first (in mode Sem), then call $\mathit{Linearize}$, and only afterwards eliminate the remaining quantifiers by running Algorithm 1 again (now in mode \mathcal{QF}). In fact, this sequence is fundamental for obtaining a 3EXPTIME procedure.

Consider the formula $\Psi := q \mid 2^{|x|} - r \wedge y \geq 2^{|x|}$. For specific values of q and r , the smallest $|x|$ satisfying Ψ might be $q - 1$. If Ψ is a subformula obtained during quantifier elimination, then, according to Proposition 10, q might have a triply exponential magnitude relative to the input size. This means that the smallest y satisfying Ψ might have a quadruply exponential magnitude. Eliminating x and y in this case would lead to a quadruply exponential blow-up in the number of disjuncts to be considered during quantifier elimination. Our strategy avoids this problem by delaying (if necessary) the elimination of x and y until we obtain a formula in \mathcal{PowEmp} . Calling `Linearize` reduces the reasoning to the exponents, which are triply exponential at worst.

An observation on $\mathcal{O}ct$. The bounds for integer octagon arithmetic presented in Proposition 9 reveal not only that this logic admits an exponential-time quantifier elimination procedure, but also that the satisfaction problem for this theory can be solved in PSPACE. Indeed, observe that the bound on $\|lin(\Psi)\|$ given in Proposition 9 implies that all constants and coefficients appearing in the output formula Ψ have polynomial bit-length. Then, one can apply the standard quantifier relativization algorithm from \mathcal{Pa} to obtain a PSPACE procedure for $\mathcal{O}ct$. Briefly, the quantifier relativization procedure for \mathcal{Pa} first replaces every quantifier $\exists x.\varphi$ in the input formula with a *bounded quantifier* $\exists x \in [-f(\varphi), f(\varphi)].\varphi$, where $f: \mathcal{Pa} \rightarrow \mathbb{N}$, and then iterates through all (finitely many) values the quantified variable can take, searching for a solution to the formula. The bound on $\|lin(\Psi)\|$ obtained for $\mathcal{O}ct$ implies that $f(\Psi)$ has bit-length that is at most polynomial in $\text{len}(\Psi)$. See [22] for more information on quantifier relativization.

On the non-elementary bound for $\mathcal{Pa}(\lambda x.2^{|x|})$. To conclude, we provide some insights on why our procedure runs in non-elementary time on formulae from the TOWER-complete logic $\mathcal{Pa}(\lambda x.2^{|x|})$. One of the ingredients that guarantee that our procedure for $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$ runs in 3EXPTIME is that we are able to postpone calls to `Linearize` to after Algorithm 1. In $\mathcal{Pa}(\lambda x.2^{|x|})$ this is not possible: since each variable can appear both linearly and in powers, `Linearize` must be invoked after each call to `SemCover`, in order to “linearize” a variable, and then eliminate it with `PresQE`. However, `SemCover` adds, in the worst case, a number of additional variables that is roughly the number h of homogeneous terms in the formula (see Lemma 7). When the next quantifier block is considered, these variables must all be linearized and eliminated with `PresQE`. As indicated in the table of Lemma 6 (leftmost column of the last row), in doing this, the number of homogeneous terms of the resulting formula becomes exponential in h . This “ h^h ” dependency makes the algorithm run in non-elementary time (in fact TOWER).

6 Conclusion

We have proven new elementary upper bounds for $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$, and for the existential fragment of $\mathcal{Pa}(\lambda x.2^{|x|})$. We believe this is a step towards understanding which decidable arithmetic theories have elementary bounds, and moreover that our method extends to provide elementary bounds for any prefix class of $\mathcal{Pa}(\lambda x.2^{|x|})$, but we leave this for future work. Our results open several research directions, which we now summarize.

Tighter bounds for $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$. It is well-known that, using the bounds on the formulae returned by quantifier elimination procedures for \mathcal{Pa} , one can derive a 2AEXPTIME(POLY) quantifier relativization procedure for \mathcal{Pa} [25]. Here 2AEXPTIME(POLY) is the class of

all problems that can be decided with an alternating Turing machine running in doubly exponential time and performing a polynomial number of alternations. In fact, \mathcal{Pa} is complete for this class under polynomial-time reductions [1]. Our 3EXPTIME procedure for $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$ shows that, in terms of deterministic time complexity, this theory is not harder to decide than \mathcal{Pa} . However, at this stage obtaining a $2\text{AEXPTIME}(\text{POLY})$ quantifier relativization algorithm from the bounds of our procedure seems not easy.

Automata-based decision procedures. As $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$ is a fragment of Büchi arithmetic, it also admits a representation by finite automata. It appears plausible that the automata-based procedure for \mathcal{Pa} [14, 7] could be adapted to $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$. However, having the procedure run in 3EXPTIME might be very challenging. This is due to the fact that, as observed above, $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$ formulae may require numbers of quadruply exponential magnitude; instead of triply exponential as in the case of \mathcal{Pa} .

Geometric decision procedures. A class of regular expressions corresponding to $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$ was already defined by Semenov [23, Theorem 5]. These expressions can be seen as an extension of semilinear sets [18, 9], so it is conceivable that there is an elementary decision procedure for $\mathcal{Pa}(2^{\mathbb{N}}(\cdot))$ which is based on geometry and manipulates these objects directly. However, similarly to the automata-based approach, making such a procedure run in 3EXPTIME , as the recent one for \mathcal{Pa} [5] does, appears challenging.

Tighter bounds for $\exists\mathcal{Pa}(\lambda x.2^{|x|})$. An obvious question is whether our upper bound for the existential fragment can be improved. For comparison, the existential fragment of Büchi arithmetic is known to be in NP [11]. While the same may be true for $\exists\mathcal{Pa}(\lambda x.2^{|x|})$, it would be very surprising if such a result were to be proved with a technique similar to the one in our paper. In our NEXPTIME algorithm for $\exists\mathcal{Pa}(\lambda x.2^{|x|})$, the main source of blow-up is the use of Weispfenning’s quantifier elimination procedure to eliminate linearly occurring variables. Quantifier elimination is known to be often non-optimal when it comes to deciding existential fragments of logics, and this is the case for $\exists\mathcal{Pa}$, the existential fragment of Presburger arithmetic. A possible avenue to improve the NEXPTIME upper bound would be to look at geometric procedures, which in the context of $\exists\mathcal{Pa}$ perform much better.

Improving the NP lower bound is also challenging. There are several extensions of $\exists\mathcal{Pa}$ that currently fall between NP and NEXPTIME . These include $\exists\mathcal{Pa}$ with pre-quadratic constraints [10, 21] and $\exists\mathcal{Pa}$ with divisibility constraints [15]. One idea is to exploit the ability of $\exists\mathcal{Pa}(\lambda x.2^{|x|})$ to express a pairing function, that is, an injection from $\mathbb{N} \times \mathbb{N}$ to \mathbb{N} , with, e.g., the formula $z = 2^{|2x|} + 2^{|2y+1|}$ [6, p. 55]. Pairing functions are known to lead to non-elementary lower bounds in the presence of quantifier alternation, and an interesting direction would be to study their effect on existential theories.

References

- 1 Leonard Berman. The complexity of logical theories. *Theor. Comput. Sci.*, 11(1):71–77, 1980.
- 2 Alexis Bès. A survey of arithmetic definability. *Soc. Math. Belgique*, pages 1–54, 2002.
- 3 Egon Börger, Erich Grädel, and Yuri Gurevich. *The Classical Decision Problem*. Springer, 1997.
- 4 Gregory Cherlin and Françoise Point. On extensions of Presburger arithmetic. In *4th Easter Conference on Model Theory*, volume 86 of *Humboldt-Univ. Berlin Seminarberichte*, pages 17–34, 1986. URL: https://webusers.imj-prg.fr/~francoise.point/papiers/cherlin_point86.pdf.

- 5 Dmitry Chistikov, Christoph Haase, and Alessio Mansutti. Geometric decision procedures and the VC dimension of linear arithmetic theories. In *LICS*, 2022.
- 6 Kevin J. Compton and C. Ward Henson. A uniform method for proving lower bounds on the computational complexity of logical theories. *APAL*, 48(1):1–79, 1990.
- 7 Antoine Durand-Gasselín and Peter Habermehl. On the use of non-deterministic automata for Presburger arithmetic. In *CONCUR*, 2010.
- 8 Jeanne Ferrante and Charles Rackoff. A decision procedure for the first order theory of real addition with order. *SIAM J. Comput.*, 4(1):69–76, 1975.
- 9 Seymour Ginsburg and Edwin H. Spanier. Semigroups, Presburger formulas, and languages. *Pacific Journal of Mathematics*, 16(2):285–296, 1966.
- 10 Robert Givan, David McAllester, Carl Witty, and Dexter Kozen. Tarskian set constraints. *Information and Computation*, 174(2):105–131, 2002.
- 11 Florent Guépin, Christoph Haase, and James Worrell. On the existential theories of Büchi arithmetic and linear p -adic fields. In *LICS*, 2019.
- 12 Philipp Hieronymi and Christian Schulz. A strong version of Cobham’s theorem. In *STOC*, 2022.
- 13 Deepak Kapur, Zhihai Zhang, Matthias Horbach, Hengjun Zhao, Qi Lu, and ThanhVu Nguyen. Geometric quantifier elimination heuristics for automatically generating octagonal and max-plus invariants. In *Automated Reasoning and Mathematics - Essays in Memory of William W. McCune*, volume 7788 of *LNCS*, pages 189–228. Springer, 2013.
- 14 Felix Klaedtke. Bounds on the automata size for Presburger arithmetic. *ACM Trans. Comput. Log.*, 9(2):11:1–11:34, 2008.
- 15 Antonia Lechner, Joël Ouaknine, and James Worrell. On the complexity of linear arithmetic with divisibility. In *LICS*, 2015.
- 16 Antoine Miné. The octagon abstract domain. *High. Order Symb. Comput.*, 19(1):31–100, 2006.
- 17 Derek C. Oppen. A $2^{2^{pn}}$ upper bound on the complexity of Presburger arithmetic. *JCSS*, 16(3):323–332, 1978.
- 18 Rohit J. Parikh. On context-free languages. *J. ACM*, 13(4):570–581, 1966.
- 19 Françoise Point. On decidable extensions of Presburger arithmetic: from A. Bertrand numeration systems to Pisot numbers. *JSL*, 65(3):1347–1374, 2000.
- 20 Mojżesz Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt, 1929. In *Comptes Rendus du I Congrès des Mathématiciens des Pays Slaves*, pages 92–101.
- 21 Rodrigo Raya, Jad Hamza, and Viktor Kunčák. On the complexity of convex and reverse convex prequadratic constraints. In *LPAR*, 2023.
- 22 C. R. Reddy and D. W. Loveland. Presburger arithmetic with bounded quantifier alternation. In *STOC*, 1978.
- 23 Aleksei L. Semenov. On certain extensions of the arithmetic of addition of natural numbers. *Math. USSR Izv.*, 15(2):401–418, 1980.
- 24 Aleksei L. Semenov. Logical theories of one-place functions on the set of natural numbers. *Math. USSR Izv.*, 22(3):587–618, 1984.
- 25 Volker Weispfenning. The Complexity of Almost Linear Diophantine Problems. *J. Symb. Comput.*, 10(5):395–404, 1990.