LIF

Laboratoire d'Informatique Fondamentale
de Marseille

Unité Mixte de Recherche 6166
CNRS – Université de Provence – Université de la Méditerranée

From Automata to Semilinear Sets: a Solution for Polyhedra and Even More General Sets

Denis Lugiez

Rapport/Report 21-2004

20 april 2004

Les rapports du laboratoire sont téléchargeables à l'adresse suivante
Reports are downloadable at the following address

http://www.lif.univ-mrs.fr

From Automata to Semilinear Sets: a Solution for Polyhedra and Even More General Sets

Denis Lugiez

LIF – Laboratoire d'Informatique Fondamentale de Marseille
UMR 6166
CNRS – Université de Provence – Université de la Méditerranée

Laboratoire d'Informatique Fondamentale (LIF) de Marseille
Université de Provence – CMI
39, rue Joliot-Curie F-13453 Marseille Cedex 13

lugiez@cmi.univ-mrs.fr

Abstract/Résumé

Following Semenov, Muchnik [Muc03] solves the problem of deciding whether an automaton working on the representation of tuples of numbers in some basis is definable in Presburger arithmetic, or equivalently is a semilinear set. His solution yields a formula characterizing the semilinear sets, that can be decided in elementary time if dynamic programming techniques are used. But there is no easy way to extract from this formula the representation of semilinear sets with bases and periods, which is useful in applications to program verification. We give a solution to this problem when the semilinear sets corresponds to a conjunction of equalities, inequalities and moduli equations which corresponds to semilinears sets that have the same periods. Our approach relies on a logical characterization of these sets and yields doubly exponential bounds.

Keywords: Automata, Presburger Arithmetic, Semilinear Set

Après Semenov, Muchnik [Muc03] résoud le problème de décider si le langage reconnu par un automate qui fonctionne sur les représentations (dans une certaine base) de n-uplet d'entiers est définissable dans l'arithmétique de Presburger, ou de manière équivalente si c'est un ensemble semilinéaire. Sa solution donne une formule logique qui caractérise les ensembles semilinéaires, qui peut être décidée en temps élémentaire si on utilise la programmation dynamique. Mais il n'y a pas de technique simple permettant d'extraire de la formule la représentation des semilinéaires sous la forme base, périodes, ce qui est utile dans certaines applications pour la vérification de programmes. Nous donnons une solution à ce problème quand les semilinéaires correspondent à une conjonction d'égalités, d'inégalités et d'équations modulo, ce qui correspond à des semilinéaires qui ont les mêmes périodes. Notre solution utilise une caractérisation logique de ces semilinéaires et donne une borne doublement exponentielle.

Mots-clés : Automates, Arithmétique de Presburger, Ensembles Semilinéaires.

Relecteurs/Reviewers: Roberto Amadio and Silvano DalZilio.

1 Introduction

The connection between automata working on representations of integers (in some basis) Presburger arithmetic (the first-order theory of addition over the integers) and semilinear sets has been extensively studied by logicians. Since few years, computer scientists have realized that the representation of solutions of Presburger formula with automata deserves more investigations and can be used successfully, especially in the area of verification. This representation is now used in several model-checkers like LASH ([WB95]) or FAST ([BFLP03]) and has been extended in many ways (see [BW02] for a survey). Most of these works focuss on the automata representation and on the formula representation but few consider semilinear sets. However, this representation is also interesting and it has been used recently in the BRAIN model-checker (combined with formulas see [RV02]). This raises the problem of changing one representation for another one and of computing one representation from another one. This is easy in some cases (from a formula to an automaton, or from a semilinear set to a formula) but much more difficult in other cases. The most difficult problem is to go from an automaton to a formula or a semilinear set: the automata are more powerful than Presburger arithmetic, and the structure of the automaton and the structure of the formula seem to be weakly related. Muchnik ([Muc03]) achieved a breakthrough by characterizing the automata corresponding to Presburger formulas, but its solution suffers two weaknesses: the complexity is high and it doesn't give a straighforward way to recover a semilinear representation. Therefore some researchers have tried to improve his results. Two results have been obtained recently: one characterizes the automata accepting quantifier free formulas [Ler03] and the other one characterizes the automata accepting convex polyhedras [Lat04]. Both works study in detail the structure of the automaton and try to extract the Presburger formula from it. In this work we present a completely different approach that relies on a logical characterization of semilinear sets of a particular form. This means that, given an automaton, we solve a logical formula to decide if it corresponds to a finite union of linear sets with the same periods, and we compute at the same time an explicit representation of these sets. The class of semilinear sets that we accept is more general than the class defining convex polyhedra, since we can recognize semilinear sets that are defined by conjunction of equalities, disequalities, and moduli equations. On the other hand we can't get the class of unquantified formulas.

The paper is as follows: we recall the basic notions in section 2, and we briefly state the problem in section 3. Then we describe the expressive power of the semilinear sets that we consider in section 4 and we state by solving the linear case in section 5. The more complex case is explained in section 6.

2 Basic Notions

For more informations on automata, logic and Presburger arithmetic, the reader is refered to [BHMV94].

2.1 Semilinear Sets

We denote the set of natural numbers[1] by $\mathcal{N}$ and we consider subsets of $\mathcal{N}^p$. We use the vector notation $\vec{x}, \vec{y}, \ldots$ to distinguish tuples of integers from integers.

A set L of $\mathcal{N}^p$ is a linear set iff there exists $\vec{b} \in \mathcal{N}^p$ (the basis), a finite subset $\mathcal{P} = \{\vec{p}_1, \ldots, \vec{p}_n\} \subseteq \mathcal{N}^p$ (the periods) such that $L = \{\vec{x} \mid \vec{x} = \vec{b} + \Sigma_{i=1}^{i=k} \lambda_i \vec{p}_i, \lambda_i \in \mathcal{N}\}$ and we shall denote L by $L(\vec{b}, \mathcal{P})$ A semilinear set is a finite union of linear sets. Given a finite set $\mathcal{C} = \{\vec{c}_1, \ldots, \vec{c}_m\} \subseteq \mathcal{N}^p$, we denote by $L(\mathcal{C}, \mathcal{P})$ the semilinear set $L(\vec{c}_1, \mathcal{P}) \cup \ldots \cup L(\vec{c}_m, \mathcal{P})$. This paper will focus on such sets.

Presburger arithmetic is the first-order theory of $(\mathcal{N}, +, \geq)$ and it is well-known that semilinear sets are the models of Presburger arithmetic formulas. More precisely, for any formula $\phi(x_1, \ldots, x_p)$ of Presburger arithmetic the set $\{(x_1, \ldots, x_p) \mid \models \phi(x_1, \ldots, x_p)\}$ is a semilinear set of $\mathcal{N}^p$.

2.2 Automata

When we write natural numbers in some basis r (usually $r = 2$), a number is a word on the alphabet $\Sigma = \{0, \ldots, r-1\}$. Similarly a tuple of p numbers can be represented as a word on Σ^p. Finite state automata on this alphabet recognize sets of tuples of integers, and it is well-known that semilinear sets are recognizable (i.e. for each semilinear set L there is some automaton such that $L = L(\mathcal{A})$[2]). But there exist recognizable sets which are not semilinear (for instance the set of r^n for $n \geq 0$). To match naturals that have representation of different size, it is necessary to add trailing 0's in the representation. For instance the natural 4 in basis 2 is represented by 0010^* (to be read from left to right).

A more concise representation is to use automata such that the alphabet is $\{0, .., r-1\}$ (so-called *binary automata* for $r = 2$). In this case, the language that we consider is the set of words with length $l = kp$. It is straightforward to transform an automaton on tuples into an automaton on $\{0, \ldots, r-1\}$ (just unfold the automaton and add new states). The size of the alphabet for these automata doesn't depend on p the number of variables.

The size of an automaton $\mathcal{A}$, denoted by $|\mathcal{A}|$, is the size of the transition relation and depends on the number of states and of the size of the alphabet. This explains why the size of a minimal binary automaton associated to some set of tuples of numbers can be smaller than the size of an automaton defined on the alphabet $\{0,1\}^p$ accepting the same set of tuples.

Our results are independent of the kind of automata that we use, except when we are concerned with complexity issues (in this case, we precise which automata are considered).

2.3 Logic

To each formula of Presburger arithmetic, one can associate an automaton accepting the models of the formula. Moreover, if $L \subseteq \mathcal{N}^p$ is a set such that $L = L(\mathcal{A})$ for some automaton $\mathcal{A}$, the extension of Presburger arithmetic by $\vec{x} \in L$ is still decidable. For readability we introduce some abbreviations:

[1] therefore we consider positive numbers only, not negative ones

[2] actually, $L(\mathcal{A})$ is the set of *representations in the basis* r of elements of L

- We write $\overrightarrow{y} \leq \overrightarrow{x}$ iff $\bigwedge_{i=1}^{i=p} x_i \leq y_i$ (where $\overrightarrow{x} = (x_1, \ldots, x_p)$, $\overrightarrow{y} = (y_1, \ldots, y_p)$). Similarly, we write $\overrightarrow{x} \leq C$ for $\bigwedge_{i=1}^{i=p} x_i \leq C$ (where $C \in \mathcal{N}$) and we write $\overrightarrow{x} > C$ for $\neg(\overrightarrow{x} \leq C)$.

- $min(L)$ denotes the set of minimal elements of a set L. It can be expressed as
$$\overrightarrow{x} \in L \wedge \forall \overrightarrow{y} \ (\overrightarrow{y} \in L \implies \neg(\overrightarrow{y} \leq \overrightarrow{x}))$$
This set always exists and is always finite (even for sets that are not recognizable) by Dickson's lemma ([Dic13] that states that there is no infinite sequence of incomparable tuples of integers). We can express that there is only one minimal element by $\overrightarrow{x} \in L \wedge \forall \overrightarrow{y} \ (\overrightarrow{y} \in L \implies \overrightarrow{x} \leq \overrightarrow{y})$, that we denote by $\overrightarrow{x} = min(L)$.

- $finite(\phi)$ states that the set of $\overrightarrow{z} \in \mathcal{N}^p$ satisfying some formula ϕ of Presburger arithmetic is finite. It can be expressed as
$$\exists M \ \forall \overrightarrow{z} \ \ \phi(\overrightarrow{z}) \implies \overrightarrow{z} \leq M$$

3 The Problem

We are interested in the following question:

Problem: Given an automaton $\mathcal{A}$, decide if $L(\mathcal{A}) = L(\mathcal{C}, \mathcal{P})$ and in this case, compute $\mathcal{C}$ and $\mathcal{P}$.

Remark 3.1 *To get a Presburger formula equivalent to $L(\mathcal{C}, \mathcal{P})$ is obvious.*

4 Sets Represented by a Semilinear Set $L(\mathcal{C}, \mathcal{P})$

We show that many sets can be represented as a semilinear set of the form $L(\mathcal{C}, \mathcal{P})$. In particular all integer polyhedra belong to this class. Let $x_1, \ldots, x_p$ be variables ranging over $\mathcal{N}$, and let us consider the set of $\mathcal{N}^p$ defined by a conjunction (C) of inequalities
$$\Sigma_{i=1}^{i=p} a_{i,j} x_i \geq d_j \quad j \in J$$
and moduli equations
$$\Sigma_{i=1}^{i=p} b_{i,k} x_i \equiv c_k \ mod \ m_k \quad k \in K$$
where $a_{i,j}, b_{i,k}, d_j$ are integers (possibly negative ones), c_k, m_k are positive integers.

We don't mention equations since an equation can be replaced by the conjunction of two inequalities. We prove that these sets are representable by a semilinear set $L(\mathcal{C}, \mathcal{P})$. Firstly, we introduce new variables (also ranging over $\mathcal{N}$) to get the system of diophantine equations (S):
$$\bigwedge_{j \in J} \Sigma_{i=1}^{i=p} a_{i,j} x_i + y_j = d_j \wedge \bigwedge_{k \in K} \Sigma_{i=1}^{i=p} b_{i,k} x_i = c_k + m_k z_k$$

In what follows, *solution* stands for *positive solution* (i.e. all components of a solution are ≥ 0).

A minimal solution $\overrightarrow{s}_\mu$ of a system of diophantine equations (S) is a solution such that there is no solution $\overrightarrow{s} \neq \overrightarrow{s}_\mu$ of (S) such that $\overrightarrow{s} \leq \overrightarrow{s}_\mu$.

We use some well-known facts on solutions of diophantine equations: a solution $(x_1, \ldots, x_p, y_1, \ldots, y_m, z_1, \ldots, z_n)$ of (S) is the sum of a minimal solution of (S) (and there are finitely many of them) and of a solution of the homogeneous system (H)

$$\bigwedge_{j \in J} \Sigma_{i=1}^{i=p} a_{i,j} x_i + y_j = 0 \wedge \bigwedge_{k \in K} \Sigma_{i=1}^{i=p} b_i x_i - m_k z_k = 0$$

Moreover a solution of the homogeneous system is a linear combination of the minimal non-zero solutions[3] of the homogeneous system, again in finite number. If $\mathcal{C}_\mu$ denotes the finite set of the minimal solutions of (S) and $\mathcal{P}_\mu$ denote the minimal solutions of the homogeneous system, we get that $(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z})$ is a solution iff $(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z}) \in L(\mathcal{C}_\mu, \mathcal{P}_\mu)$.

Let π be the projection defined by $\pi((\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z})) = \overrightarrow{x}$.

Proposition 4.1 *$\overrightarrow{x}$ is a solution of (C) iff $\overrightarrow{x} \in L(\pi(\mathcal{C}_\mu), \pi(\mathcal{P}_\mu))$*

Proof: $\Rightarrow$ direction.
Assume $\overrightarrow{x}$ is a solution.

By definition there exists $\overrightarrow{y}, \overrightarrow{z}$ such that $(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z})$ is a solution of (S).
Therefore $(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z}) \in L(\mathcal{C}_\mu, \mathcal{P}_\mu)$ which proves that $\overrightarrow{x} \in \pi(L(\mathcal{C}_\mu, \mathcal{P}_\mu))$.
Since projection is a linear mapping, we get $\pi(L(\mathcal{C}_\mu, \mathcal{P}_\mu)) = L(\pi(\mathcal{C}_\mu), \pi(\mathcal{P}_\mu))$.
Therefore $\overrightarrow{x} \in L(\pi(\mathcal{C}_\mu), \pi(\mathcal{P}_\mu))$.

$\Leftarrow$ direction.
Assume that $\overrightarrow{x} \in L(\pi(\mathcal{C}_\mu), \pi(\mathcal{P}_\mu))$. By definition $\overrightarrow{x} = \overrightarrow{x}_\mu + \Sigma_{i \in I} \lambda_i \overrightarrow{x}_i$ where $\overrightarrow{x}_\mu \in \pi(\mathcal{C}_\mu)$, $\overrightarrow{x}_i \in \pi(\mathcal{P}_\mu)$ for $i \in I$.
By definition of projection, $\overrightarrow{x}_\mu = \pi((\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z})_\mu)$ for some $(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z})_\mu \in \mathcal{C}_\mu$ and and for each $i \in I$, $\overrightarrow{x}_i = \pi((\overrightarrow{x}_i, \overrightarrow{y}_i, \overrightarrow{z}_i)_\mu)$ for some $(\overrightarrow{x}_i, \overrightarrow{y}_i, \overrightarrow{z}_i)_\mu \in \mathcal{P}_\mu$.
Therefore, by linearity of projection, we get that $\overrightarrow{x} = \pi((\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z}))$ where $(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z}) \in L(\mathcal{C}_\mu, \mathcal{P}_\mu)$.

Therefore there exists $\overrightarrow{y}, \overrightarrow{z}$ such that $(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z})$ is a solution of (S) which yields that $\overrightarrow{x}$ is a solution of (C).

This proves that the set of solutions of (C) is exactly $L(\pi(\mathcal{C}_\mu), \pi(\mathcal{P}_\mu))$. $\square$

Remark 4.2 *The same proof works also for union of conjunctions that differ only in the constant terms (d_j's or c_j's): for instance, this allows sets that are union of a poyhedra and some translations of this polyhedra.*

5 Recovering a Linear Set from its Automaton

Firstly, we consider the particular case of a linear set and we start the process by providing a canonical way to describe linear sets.

[3] hence minimal means minimal in the set of solutions $\neq \overrightarrow{0}$

5.1 A Canonical Representation for Linear Sets.

Let $L \subseteq \mathcal{N}^p$, an element $\vec{x} \in L$ is *reducible* in L iff either $\vec{x} = 0$ or there exist $\vec{x}_1 \in L, \vec{x}_2 \in L$ such that $\vec{x} = \vec{x}_1 + \vec{x}_2$ and $\vec{x}_1, \vec{x}_2 \neq 0$. A element $\vec{x} \neq 0$ is *irreducible* iff it is not reducible. We denote by $\mathcal{I}r(L)$ the set of irreducible elements of L. This set can be infinite: if $L = \{2^i \mid i \in \mathcal{N}\}$, then $\mathcal{I}r(L) = L$, but for linear sets we have:

Proposition 5.1 *If L is a linear set then L has a unique minimal element $\vec{b}$, $\mathcal{I}r(L - \vec{b})$ is finite and $L = L(\vec{b}, \mathcal{I}r(L - \vec{b}))$ where $L - \vec{b} = \{\vec{x} \mid \vec{x} + \vec{b} \in L\}$.*

This proposition yields a canonical representation of a linear set: $\vec{b}$ and $\mathcal{I}r(L - \vec{b})$ are uniquely defined. Moreover any other representation of L in the form $L(\vec{b}, \mathcal{P})$ contains superfluous information. The reader should remark that the elements of $\mathcal{I}r(L - \vec{b})$ may be dependent (in the meaning of dependent vectors, with $\mathcal{Q}$ as the set of scalars), see $L = L(0, \{(2), (3)\})$ in $\mathcal{N}$.

Proof: By definition the basis of a linear set is its minimal element $\vec{b}$.

Let $M = L - \vec{b} = L(\vec{0}, \mathcal{P})$.

(i) We prove that $\mathcal{I}r(M) \subseteq \mathcal{P}$.

Let $\vec{x} \in \mathcal{I}r(M)$. By definition $\vec{x}$ is a linear combination of elements of $\mathcal{P}$. Since each element of $\mathcal{P}$ is in M, we get that the linear combination is necessarily reduced to only one element, therefore $\vec{x} \in \mathcal{P}$.

(ii) We prove that $M = L(\vec{0}, \mathcal{I}r(M))$.

Let $\vec{p} \in \mathcal{P}$ with $\vec{p} \notin \mathcal{I}r(M)$. We show that $M = L(\vec{0}, \mathcal{P} - \{\vec{p}\})$, that we can eliminate all reducible elements of $\mathcal{P}$. Combined with (i), this yields the desired result (since $\mathcal{P}$ is finite, we can eliminate all reducible elements of $\mathcal{P}$ to end with a set of irreducible elements).

By definition $\vec{p} = \vec{x}_1 + \vec{x}_2$ with $\vec{x}_1, \vec{x}_2 \neq \vec{0} \in M$.

Therefore each x_i is a linear combination (with positive coefficients) of elements of $\mathcal{P}$.

Moreover $\vec{p}$ can't occur in $\vec{x}_1$ nor $\vec{x}_2$ (otherwise $\vec{x}_1$ or $\vec{x}_2 = \vec{p}$).

Therefore $\vec{p}$ is a linear combination of elements of $\mathcal{P} - \{\vec{p}\})$ which proves that $M = L(\vec{0}, \mathcal{P} - \{\vec{p}\})$. □

Moreover the set of irreducible terms can be defined by the formula:

$$\vec{x} \in L \wedge \forall\, \vec{x}_1, \vec{x}_2\ \ (\vec{x} \neq 0 \wedge\ \vec{x} = \vec{x}_1 + \vec{x}_2 \wedge \vec{x}_1 \in L \wedge\ \vec{x}_2 \in L \implies \vec{x}_1 = \vec{x} \vee \vec{x}_2 = \vec{x})$$

We may have an exponential number of elements in $\mathcal{I}r(L - \vec{b})$ with respect to the size of the Presburger formula defining L: for a system of homogeneous linear diophantine equation, the number of minimal solutions can be exponential in the size of the system and the irreducible elements of the set of solutions are exactly these minimal solutions.

5.2 A Logical Characterization of Linear Sets.

Let us define:

(i) there is a unique minimal element $\vec{b}$ in L.

$$\vec{b} = min(L)$$

(ii) $\mathcal{I}r(L - \vec{b})$ is finite.

$$Finite(\mathcal{I}r(L - \vec{b}))$$

(iii) $\mathcal{I}r(L - \vec{b})$ is stable under addition:

$$\forall x, \vec{y}\ \vec{x} \in L - \vec{b}, \vec{y} \in L - \vec{b} \Longrightarrow \vec{x} + \vec{y} \in L - \vec{b}$$

The conjunction of these formulas (and the definition of $\mathcal{I}r(L - \vec{b})$) yields the formula $DEFLIN(L)$ which is true iff L is a linear set. Moreover, in this case, we have $L = L(\vec{b}, \mathcal{I}r(L - \vec{b}))$.

Proposition 5.2 *A set L satisfies $DEFLIN(L)$ iff L is linear.*

Proof: $\Rightarrow$ direction.
Assume that L satisfies $DEFLIN(L)$. We show that $L = L(\vec{b}, \mathcal{I}r(L - \vec{b}))$.

By (ii) $\mathcal{I}r(L - \vec{b})$ is finite and $\vec{b}$ is the unique minimal element of L by (i).

By (iii) $L - \vec{b}$ is stable under $+$ and $\mathcal{I}r(L - \vec{b}) \subseteq L - \vec{b}$ yields that $L(\vec{b}, \mathcal{I}r(L - \vec{b})) \subseteq L$.

To prove the reverse inclusion, we assume that $L \not\subseteq L(\vec{b}, \mathcal{I}r(L - \vec{b}))$.

Let $\vec{x}$ be the minimal element ($\neq 0$) of $L - \vec{b}$ which is not in $L(0, \mathcal{I}r(L - \vec{b}))$.

By definition $\vec{x}$ is reducible (otherwise $\vec{x} \in \mathcal{I}r(L - \vec{b})$), then $\vec{x} = \vec{x}_1 + \vec{x}_2$ with $\vec{x}_1, \vec{x}_2 \in L - \vec{b}$.

By minimality of $\vec{x}$, $\vec{x}_1, \vec{x}_2 \in L(0, \mathcal{I}r(L - \vec{b}))$, therefore $\vec{x} \in L(0, \mathcal{I}r(L - \vec{b}))$, which yields a contradiction.

$\Leftarrow$ direction.
Assume that L is linear.

By proposition 5.1, $L = L(\vec{b}, \mathcal{I}r(L - \vec{b}))$ with $\vec{b}$ the minimal element of L, and $IRR(L - \vec{b})$ the (finite) set of irreducible elements, therefore L satisfyies $(i), (ii), (iii)$.

$\square$

Remark 5.3 *There exists a semilinear set which is not linear but which has a with a unique minimal element and is stable under addition (hence condition (i) is necessary). Take L as $\{(0,0)\} \cup L((1,0), \{(1,1),(1,0)\})$ which has an infinite set of irreducible elements of the form $(1,0) + \lambda(1,1)$.*

5.3 Complexity Issues for Linear Sets

In that section we consider automata on alphabets of the form $\{0,1\}^p$.

Basics Results: Firstly, we recall some basic definitions and results.

An automaton is *complete* iff each word reaches at least one state. Determinization of an automaton $\mathcal{A}$ yields an automaton of size $0(2^{|\mathcal{A}|})$ and requires time $0(2^{|\mathcal{A}|})$. Given an automaton $\mathcal{A}$ accepting the models of $\phi(\overrightarrow{x}_1, \ldots, \overrightarrow{x}_m)$, automata $\mathcal{A}_1, \ldots, \mathcal{A}_m$ accepting the sets $L_1, \ldots, L_m$, one can construct an automaton accepting the models of $\phi(\overrightarrow{x}_1, \ldots, \overrightarrow{x}_m) \wedge \bigwedge_{i=1}^{i=m} \overrightarrow{x}_i \in L_i$ of size $0(|\mathcal{A}||\mathcal{A}_1| \ldots |\mathcal{A}_m|)$ and the construction is done in time $0(|\mathcal{A}||\mathcal{A}_1| \ldots |\mathcal{A}_m|)$. Moreover the automaton is complete and deterministic if the automata $\mathcal{A}, \mathcal{A}_1, \ldots, \mathcal{A}_m$ are complete and deterministic.

Given an automaton $\mathcal{A}$ accepting the models of $\phi(\overrightarrow{x}, \overrightarrow{y})$, one can construct a complete (but usually not deterministic) automaton accepting the models of $\exists \overrightarrow{x} \quad \phi(\overrightarrow{x}, \overrightarrow{y})$ of size $0(|\mathcal{A}|)$. Similarly, one can construct a complete and deterministic automaton accepting the models of $\forall \overrightarrow{x} \quad \phi(\overrightarrow{x}, \overrightarrow{y})$ of size $0(2^{|\mathcal{A}|})$ and the construction is done in time $0(2^{|\mathcal{A}|})$.

Given a deterministic complete automaton $\mathcal{A}$, (resp. $\mathcal{A}'$) accepting the models of $\phi(\overrightarrow{x})$, (resp. $\phi'(\overrightarrow{x})$), a complete deterministic for $\neg\phi(\overrightarrow{x})$ can be obtained in constant time, and complete deterministic automata for $\phi(\overrightarrow{x}) \wedge \phi'(\overrightarrow{x})$, $\phi(\overrightarrow{x}) \vee \phi'(\overrightarrow{x})$, $\phi(\overrightarrow{x}) \implies \phi'(\overrightarrow{x})$ can be computed in time $O(|\mathcal{A}||\mathcal{A}'|)$ and have size $O(|\mathcal{A}||\mathcal{A}'|)$.

The Complexity of Constructions. We assume that $L = L(\mathcal{A})$ with $\mathcal{A}$ a complete deterministic automaton. We assume that complete deterministic automata have been constructed for the following languages $\{(\overrightarrow{x}, \overrightarrow{y}) \mid \overrightarrow{x} \leq \overrightarrow{y}\}$, $\{(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z}) \mid \overrightarrow{z} = \overrightarrow{x} + \overrightarrow{y}\}$ and these automata are considered as constants.

Proposition 5.4 *The existence of a minimal element of L can be decided in time $O(|\mathcal{A}|2^{|\mathcal{A}|})$ and the size of the minimal element is $O(|\mathcal{A}|)$.*

Proof: The minimal element is defined by

$$\overrightarrow{b} \in Ł \wedge \forall \overrightarrow{x} \ (\overrightarrow{x} \in L \implies \overrightarrow{b} \leq \overrightarrow{x})$$

which yields an automaton of size $O(|\mathcal{A}|2^{|\mathcal{A}|})$ that is computable in time $O(|\mathcal{A}|2^{|\mathcal{A}|})$. The emptiness of this automaton is decided in linear time.

Since $\overrightarrow{b}$ is the minimal element, $\overrightarrow{b}$ is accepted by $\mathcal{A}$ with a cycle free path (non-cycle free paths are also accepted because of the trailing 0's used in the representation of natural numbers). This yields that $\overrightarrow{b}$ has size $O(|\mathcal{A}|)$.

Therefore we can get an automaton of size $O(|\mathcal{A}|^2)$ for $L - \overrightarrow{b}$.

Proposition 5.5 *Let L be accepted by a complete deterministic automaton $\mathcal{A}$, then there exists an automaton accepting $\mathcal{I}r(L)$ of size $O(2^{|\mathcal{A}|^3})$.*

Proof: The set $\{(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{z}) \mid \overrightarrow{x} \in L - \{\overrightarrow{0}\}, \overrightarrow{y} \in L, \overrightarrow{z} = \overrightarrow{x} + \overrightarrow{y} \in L\}$ is accepted by a deterministic automaton of size $O(|\mathcal{A}|^3)$. The set of reducible elements is the set $\{\overrightarrow{z} \mid \exists \overrightarrow{x}, \overrightarrow{y} \ \overrightarrow{x} \in L - \{\overrightarrow{0}\}, \overrightarrow{y} \in L, \overrightarrow{z} = \overrightarrow{x} + \overrightarrow{y} \in L\}$ and $\mathcal{I}r(L)$ is the complement of this set.

Proposition 5.6 *Given a set L accepted by a deterministic complete automaton $\mathcal{A}$, the finiteness of L can be tested in $O(|\mathcal{A}|^2)$.*

Here finiteness means finiteness of the set of tuples of naturals associated to L, which is not the same as finiteness of the language accepted by $\mathcal{A}$, since we add trailing 0's to representations of natural numbers.
Proof: To get rid of trailing 0's we intersect L with the language of words not ending by the letter $(0, \ldots, 0)$, which is done in constant time.

Therefore we have to test for finiteness the language accepted by an automaton of size $0(|\mathcal{A}|)$. For each state, we test if it is accessible which is done in $O(|\mathcal{A}|)$. Then for each accessible state, we test if there is a loop on this state, which can be done in $0(|\mathcal{A}|)$. Therefore finiteness can be tested in $O(|\mathcal{A}|^2)$ (we take $|\mathcal{A}|$ to bound the number of states, a more accurate bound is $O(|\mathcal{A}||\mathcal{Q}|)$ where $\mathcal{Q}$ is the set of states of $\mathcal{A}$). □

Combining all results together, we get a doubly exponential bound (in a polynomial of $|\mathcal{A}|$) to decide the linearity of $L = L(\mathcal{A})$.

In the next section we give a linear set which has exponentially many irreducible periods in the size of the *binary* automaton accepting this set. Therefore an exponential blowup can't be avoided if we want to compute explicitely the set of periods.

5.4 An Exponential Blowup between the Binary Automaton and the Linear Set

We give a formula such that the size of the corresponding *binary automaton* is polynomial but such that the set of solutions is a linear set with an exponential number of irreducible periods. Let us consider the formula

$$(*)\ \Sigma_{i=1}^{i=p} x_i = 0\ mod\ \alpha$$

where $\alpha = 2^m$ for some m, $p = \alpha$ and the x_i are variables ranging over $\mathcal{N}$. It is known that the set of solutions of the equation $\Sigma_{i=1}^{i=p} x_i = \alpha$ has $C_{p+\alpha-1}^{\alpha}$ solutions (see the appendix for a direct proof). Let S_α denote the set of solutions of $\Sigma_{i=1}^{i=p} x_i = \alpha$ different from $\overrightarrow{0}$.

For $\alpha = p$ we get $C_{p+\alpha-1}^{\alpha} = \frac{(2p-1)!}{p!(p-1)!} = \frac{1}{2}\frac{2p!}{p!p!}$. By Stirling formula $n! = O(\sqrt{2\pi n}(\frac{n}{e})^n)$, therefore $C_{2p-1}^{p} = 0(\frac{2^p}{\sqrt{p}})$ for p large enough.

Claim 1 *Each solution of $(*)$ is a linear combination of elements of S_α.*

Proof: Let $\overrightarrow{y}$ such that $\Sigma_{i=1}^{i=p} y_i = k\alpha$. The proof is by induction on k.

Base case: $k = 0$, the nul combination does the work.

Inductive case: assume $\Sigma_{i=1}^{i=p} y_i = k\alpha$. There exists $x_i \leq y_i$ such that $\Sigma_{i=1}^{i=p} x_i = \alpha$, and we have $\Sigma_{i=1}^{i=p}(y_i - x_i) = (k-1)\alpha$. The results follows by induction hypothesis. □

Claim 2 *The elements of S_α are irreducible elements of $L(\overrightarrow{0}, S_\alpha)$.*

Proof: Assume that $\Sigma_{i=1}^{i=p} x_i = \alpha$ and $\overrightarrow{x} = \overrightarrow{y} + \overrightarrow{z}$ with $\Sigma_{i=1}^{i=p} y_i = k_1\alpha$ and $\Sigma_{i=1}^{i=p} Z_i = k_2\alpha$. Then $\Sigma_{i=1}^{i=p} x_i = (k_1 + k_2)\alpha = \alpha$ which means that k_1 or k_2 is 0, i.e. $\overrightarrow{y}$ or $\overrightarrow{z}$ is $\overrightarrow{0}$. □

We give a deterministic automaton on the alphabet $\{0, 1\}$ accepting the solutions of $(*)$ which has a size polynomial in the size of $(*)$. A solution of $(*)$ instantiates a variable x_i by a number $b_0^i b_1^i b_2^i \ldots$ i.e. $x_i = b_0^i + 2^1 b^i + 2^2 b_2^i + \ldots$ (b_j^i is say to have rank j) and we simply need to check that the sum of these numbers is divisible by α. Since $\alpha = 2^m$, it is enough to check that the sum of the m first bits is divisible by α.

The idea underlying the construction of the automaton is to compute the sum $\Sigma_{i=1}^{i=p} x_i$ summing all bits of the same rank, keeping in memory the remainder modulo α (which implies that we also remember the rank). Since the automaton is binary, we must also remember how many bits we have added modulo p (the number of variables). Therefore a state of the automaton is:

$$(nbit, rank, modulus)$$

where

- $nbit \in \{0, \ldots, p\}$ is the number of bits of rank $rank$ that we have already seen,
- $rank \in \{0, \ldots, m\}$ is the rank of the bits that we are summing,
- $modulus \in \{0, \ldots, \alpha - 1\}$ is the value of the partial sum corresponding to $\Sigma_{i=1}^{i=p} x_i$ where the x_i's have been read up to rank $rank - 1$, and $nbit$ bits of rank $rank$ have been read.

The transition rules of the automaton are:

$$\begin{array}{lcl}
(nbit, rank, modulus) & \xrightarrow{0} & (nbit + 1, rank, modulus) \\
 & & if\ nbit < p, rank < m \\
(p, rank, modulus) & \xrightarrow{0} & (1, rank + 1, modulus) \\
 & & if\ rank < m \\
(nbit, rank, modulus) & \xrightarrow{1} & (nbit + 1, rank, modulus + 2^{rank}\ mod\ \alpha) \\
 & & if\ nbit < p, rank < m \\
(p, rank, modulus) & \xrightarrow{1} & (1, rank + 1, modulus + 2^{rank+1}\ mod\ \alpha) \\
 & & if\ rank < m \\
(p, m, modulus) & \xrightarrow{0,1} & (p, m, modulus)
\end{array}$$

The final states are the states $(p, rank, 0)$.

Proposition 5.7 *The automaton $\mathcal{A}$ has size $0(p^2 log(p))$ and $L(\mathcal{A})$ is the set of the solutions of* $(*)$.

Proof: The number of states of $\mathcal{A}$ is $O(p^2 m)$ with $m = log(p)$ and the size of the alphabet is 2, therefore the total size of $\mathcal{A}$ is $0(p^2 log(p))$.

Let $w \in \{0, 1\}^*$ be a word. We can write $w = w_1 w_2 \ldots w_k b_1 \ldots b_{nbit}$ where all w_i have length p.

We prove that w reaches the state $(nbit, rank, modulus)$ iff either $k = rank - 1$ and $\Sigma_{i=1}^{i=p} x_i = modulus$ when $k < m$, or else $k \geq m$ and $\Sigma_{i=1}^{i=p} x_i = modulus \ mod \ 2^m$ when $k \geq m$ where $x_1, \ldots, x_p$ is the tuple of integers such that the binary representation of x_i is the sequence of the i^{th} letters of the w_j's followed by b_i if $i \leq nbit$.

The proof is by structural induction on w. Let $w' = wb$ and let $(nbit, rank, modulus)$ be the state reached by w. Let $x_1, \ldots, x_p$ be the sequence associated to w.

- Assume $rank < m$, $nbit < p$ and $b = 0$. Then the state reached by wb is $(nbit+1, rank, modulus)$. The sequence $x'_1, \ldots, x'_p$ obtained from wb is such that $x'_{nbit} = x_{nbit}0$ and $x'_j = x_j$. Therefore $\Sigma_{i=1}^{i=p} x'_i = modulus$.
- Assume $rank < m$, $nbit = p$ and $b = 0$. Then the state reached by wb is $(1, rank + 1, modulus)$. The sequence $x'_1, \ldots, x'_p$ obtained from wb is $x'_1 = x_1 0$ and $x'_j = x_j$. Therefore $\Sigma_{i=1}^{i=p} x'_i = modulus$.
- Assume $rank < m$, $nbit < p$ and $b = 1$. Then the state reached by wb is $(nbit, rank + 1, modulus + 2^{rank})$. The sequence $x'_1, \ldots, x'_p$ obtained from wb is $x'_{nbit} = x_{nbit}1$ (and $|x_{nbit}| = rank$). Therefore $\Sigma_{i=1}^{i=p} x'_i = \Sigma_{i=1}^{i=p} x_i + 2^{rank} = modulus + 2^{rank}$.
- Assume $rank < m$, $nbit = p$ and $b = 1$. Then the state reached by wb is $(1, rank{+}1, modulus{+}2^{rank})$. The sequence $x'_1, \ldots, x'_p$ obtained from wb is $x'_{nbit} = x_{nbit}1$ (and $|x_{nbit}| = rank$). Therefore $\Sigma_{i=1}^{i=p} x'_i = \Sigma_{i=1}^{i=p} x_i + 2^{rank} = modulus + 2^{rank}$.
- Assume $rank \geq m$. By induction hypothesis $\Sigma_{i=1}^{i=p} x_i = modulus$ and $\Sigma_{i=1}^{i=p} x'_i = \Sigma_{i=1}^{i=p} x_i + b2^{m+l} = modulus \ mod \ 2^m$

Assuming $rank < m$, when we read the bit 0, we have read one more bit, the modulus doesn't change and the rank is increased only if we have already read p bits (and we have seen one bit of this rank). This is accounted by the first two transitions. When we read the bit 1, we increase the sum by 2^{rank} , so we must adjust the modulus accordingly, and go to next rank if $nbit = p$ (and we have seen one bit of this rank). The two next transitions account for these updates. Finally, when we have read more that m bits, we add only multiples of α therefore the modulus doesn't change. □

6 Recovering a Semilinear Set $L(\mathcal{C}, \mathcal{P})$ from its Automaton

We extend the previous result to a semilinear set $L = L(\mathcal{C}, \mathcal{P})$ with $\mathcal{C} = \{\vec{c}_1, \ldots, \vec{c}_l\}$.

6.1 A Logical Characterization of $L(\mathcal{C}, \mathcal{P})$

Let $Add(L)$ be the set $\{\vec{z} \mid \forall \vec{x} \in L, \vec{x} + \vec{z} \in L\}$. Note that we don't require now any membership property on $\vec{z}$. This set is infinite for (infinite) semilinear sets, but the set of irreducible elements of Add may be finite. The reader may easily check that all the following formulas are definable in Presburger arithmetic extended by the predicate $\vec{x} \in L$.

We recall that given $C \in \mathcal{N}$, $\vec{x} = (x_1, \ldots, x_p) \in \mathcal{N}^p$, we write $\vec{x} \leq C$ for $\bigwedge_{i=1}^{i=p} x_i \leq C$ and we write $\vec{x} > C$ for $\neg(\vec{x} \leq C)$ i.e. $\exists x_i \; x_i > C$.

Let us define:

(i) $\mathcal{I}r(Add(L))$ is finite.

(ii) $\exists C \in \mathcal{N}$ such that

$$\forall \vec{x} \;\; [\vec{x} > C \implies \exists \, \vec{z}, \vec{x}' \;\; (\vec{z} \in Add(L) \wedge \vec{x}' \in L \wedge \vec{x} = \vec{x}' + \vec{z})]$$

Proposition 6.1 *If L satisfies (i) and (ii) for some C, then $L = L(\mathcal{C}, \mathcal{I}r(Add(L)))$ for $\mathcal{C} = \{\vec{c} \in L \mid \vec{c} \leq C\}$.*

Proof: By definition $\mathcal{C}$ is finite. Firstly, we remark that if $\vec{z} \in Add(L)$ then $\vec{z}$ is a linear combination of elements of $\mathcal{I}r(Add(L))$. The proof is by induction on $|\vec{z}|$: either $\vec{z}$ is irreducible and we are done, or $\vec{z} = \vec{z}_1 + \vec{z}_2$ where $\vec{z}_1, \vec{z}_2 \neq \vec{0}$, and $\vec{z}_1, \vec{z}_2 \in Add(L)$. By induction hypothesis, each $\vec{z}_i$ is a linear combination of irreducible elements of $Add(L)$ and we are done.

We prove now both inclusions.

- $L \stackrel{?}{\subseteq} L(\mathcal{C}, \mathcal{I}r(Add(L)))$.

  Let $\vec{x}$ be a minimal element such that $\vec{x} \in L$ and $\vec{x} \notin L(\mathcal{C}, \mathcal{I}r(Add(L)))$ (i.e. there is no $\vec{y} \in L$ such that $\vec{y} \notin L(\mathcal{C}, \mathcal{I}r(Add(L)))$ and $\vec{x} = \vec{y} + \vec{u}$). We have:

  $\vec{x} > C$ (otherwise $\vec{x} \in \mathcal{C}$) therefore by (ii), we get $\vec{x} = \vec{z} + \vec{x}'$ with $\vec{z} \in Add(L)$ and $\vec{x}' \in L$.

  Moreover $\vec{z}$ is a linear combination of elements of $\mathcal{I}r(Add(L))$ by the preliminary remark.

  $\vec{x}' > C$ otherwise $\vec{x}' \in \mathcal{C}$ and $\vec{x} \in L(\mathcal{C}, \mathcal{I}r(Add(L)))$,

  $\vec{x}' \notin L(\mathcal{C}, \mathcal{I}r(Add(L)))$ otherwise $\vec{x} \in L(\mathcal{C}, \mathcal{I}r(Add(L)))$.

  But this contradicts the minimality of $\vec{x}$.

- $L(\mathcal{C}, \mathcal{I}r(Add(L))) \stackrel{?}{\subseteq} L$.

  This is obvious: $\mathcal{C} \subseteq L$, and L is stable under addition of any $\vec{z} \in \mathcal{I}r(Add(L))$.

$\square$

The converse is stated as:

Proposition 6.2 *if $L = L(\mathcal{C}, \mathcal{P})$ then L satisfies (i) and (ii) for some C.*

Proof: For (i) there is a short proof relying on Dickson's lemma [Dic13], but we give a slightly more complex proof because we want a more accurate estimation of the number of irreducible elements.

$L = L(\mathcal{C}, \mathcal{P})$ satisfy (i)?

We must show that $\mathcal{I}r(Add(L))$ is finite.
By definition $\mathcal{P} = \{\vec{p}_1, \vec{p}_2 \ldots\} \subseteq Add(L)$.
Let $\mathcal{C} = \{\vec{c}_1, \ldots, \vec{c}_m\}$ and let $\vec{z} \in \mathcal{I}r(Add(L))$. Since adding $\vec{z}$ to each $\vec{c}_i$ yields an element of L, we get that $\vec{z}$ satisfies the conjunction of equations

$$(E) \quad \vec{c}_i + \vec{z} = \vec{c}_{\sigma(i)} + \Sigma_{j=1}^{j=|\mathcal{P}|} \lambda_j^i \vec{p}_j \quad i = 1, \ldots, m$$

for some $\lambda_j^i \in \mathcal{N}$, where σ is a mapping from $1, ..m$ onto itself.

There are $|\mathcal{C}|^{|\mathcal{C}|}$ different possible systems (E) since there are $|\mathcal{C}|$ possible values for each $\vec{c}_{\sigma(i)}$ (actually $|\mathcal{C}| - 1$ since we can rule out $\vec{c}_{\sigma(i)} = \vec{c}_i$ which yields $\mathcal{P}$ as solutions).
Conversely, if $\vec{z}$ is a solution of (E), for any $\vec{c}_i$, for any linear combination $\Sigma_{j=1}^{j=|\mathcal{P}|} \rho_j \vec{p}_j$ we have

$$\Sigma_{j=1}^{j=|\mathcal{P}|} \rho_j \vec{p}_j + \vec{c}_i + \vec{z} = \vec{c}_{\sigma(i)} + \Sigma_{j=1}^{j=|\mathcal{P}|} (\lambda_j^i + \rho_j) \vec{p}_j$$

for $i = 1, \ldots, m$ which proves that $\vec{z} \in Add(L)$.
Let us consider (E) as a system of diophantine equations in the unknowns $\vec{z}, \vec{\lambda}$. A solution of (E) is the sum of a minimal solution $(\vec{z}, \vec{\lambda})_\mu$ of (E) and of a solution $(\vec{z}, \vec{\lambda})_h$ of the homogeneous system (H)

$$\vec{z} = \Sigma_{j=1}^{j=|\mathcal{P}|} \lambda_j^i \vec{p}_j \quad for\ i = 1, \ldots, m$$

By definition, any solution $(\vec{z}, \vec{\lambda})_h$ of (H) is such that $\vec{z}$ is a linear combination of elements of $\mathcal{P}$.
If π is the projection defined by $\pi(\vec{z}, \vec{\lambda}) = \vec{z}$, any solution $\vec{z}$ of (E) is some $\pi((\vec{z}', \vec{\lambda}')_\mu + (\vec{z}'', \vec{\lambda}'')_h)$.
Therefore $\vec{z} = \vec{z}' + \vec{z}''$ where $\vec{z}' \in Add(L)$ because it is a solution of (E) and $\vec{z}'' \in Add(L)$ because it is a linear combination of elements of $\mathcal{P}$.
By definition, if $\vec{z}$ is irreducible, then $\vec{z}'' = 0$, hence $\vec{z}$ is the projection of a minimal solution of (E).
Since there are finitely many minimal solutions of (E) we have that $\mathcal{I}r(Add(L))$ is finite.
Since the number of minimal solutions of a system of equations (E) is bounded by $0(2^{|E|})$ and we have at most $|\mathcal{C}|^{|\mathcal{C}|}$ different such systems, we

get a bound $0(2^{|\mathcal{P}|+p+|\mathcal{C}|+|\mathcal{C}|log(|\mathcal{C}|)})$ for the number of irreducible elements of $IRR(Add(L))$.

$L = L(\mathcal{C}, \mathcal{P})$ satisfy (ii)?

Let $C = Max\{c_i \mid \vec{c} = (c_1, \ldots, c_n) \in \mathcal{C}\}$ and let $\mathcal{C}' = \{\vec{x} \in L \mid \vec{x} \leq C\}$. Then we prove that $L = L(\mathcal{C}', \mathcal{I}r(Add(L)))\}$.

By construction $\mathcal{C} \subseteq \mathcal{C}'$.

Moreover $\mathcal{P} \subseteq Add(L)$ and the previous proof ensures that $\mathcal{I}r(Add(L)$ is finite.

Therefore for each $\vec{p} \in \mathcal{P}$, either $\vec{p} \in \mathcal{I}r(Add(L))$ or $\vec{p} = \Sigma_{i \in I} \alpha_i \vec{z}_i$ for $\vec{z}_i \in \mathcal{I}r(Add(L))$.

This yields $L(\mathcal{C}, \mathcal{P}) \subseteq L(\mathcal{C}', \mathcal{I}r(Add(L)))$.

Conversely by definition of $Add(L)$ and $\mathcal{C}'$, we have $L(\mathcal{C}', \mathcal{I}r(Add(L))) \subseteq L(\mathcal{C}, \mathcal{P})$ $\square$

From the proof, we get that the size of $\mathcal{I}r(Add(L))$ is bounded by an exponential in the size of the description of L with bases and periods.

The next theorem summarizes the two previous propositions.

Theorem 6.3 *$L = L(\mathcal{C}, \mathcal{P})$ for some finite $\mathcal{C}, \mathcal{P}$ iff L satisfies (i) and (ii).*

6.2 Complexity Issues for Semilinear Sets

In that section we consider automata on alphabets of the form $\{0,1\}^p$. We assume that $L = L(\mathcal{A})$ with $\mathcal{A}$ a complete deterministic automaton.

First we give a bound for $Add(L)$.

Proposition 6.4 *There exists an automaton accepting $Add(L)$ of size $O(2^{|\mathcal{A}|^2})$.*

Proof: The complement of $Add(L)$ is the set $\{\vec{z} \mid \exists \vec{x} \in L \wedge \vec{x} + \vec{z} \notin L\}$.

A deterministic complete automaton for $M = \{(\vec{x}, \vec{y}, \vec{z}) \mid \vec{y} = \vec{x} + \vec{z}$ *and* $\vec{x} \in L, \vec{y} \notin L\}$ of size $0(|\mathcal{A}|^2)$ can be computed in time $0(|\mathcal{A}|^2)$.

The complement of $Add(L)$ is $\{\exists \vec{x}, \vec{y} \mid (\vec{x}, \vec{y}, \vec{z}) \in M\}$. Therefore a complete deterministic for $Add(L)$ can be computed in $0(2^{|\mathcal{A}|^2})$. $\square$

Proposition 6.5 *To test if $L(\mathcal{A})$ is a semilinear set of the form $L(\mathcal{C}, \mathcal{P})$ (for some unknown $\mathcal{C}, \mathcal{P}$) can be done in $O(2^{2^{c|\mathcal{A}|^2}})$ for some $c > 3$.*

Proof: We compute the complexity of deciding (i) and (ii).

Deciding (i).

By proposition 6.4 an automaton for $Add(L)$ has size $0(2^{|\mathcal{A}|^2})$ and by proposition 5.5 an automaton accepting $\mathcal{I}r(Add(L))$ has size $O(2^{2^{3|\mathcal{A}|^2}})$ and is computed in time $O(2^{2^{3|\mathcal{A}|^2}})$.

By proposition 5.6, the finiteness of this language can be tested in quadratic time in the size of the automaton i.e. $O(2^{2^{3|\mathcal{A}|^2}+1})$ that we simplify into $O(2^{2^{c|\mathcal{A}|^2}})$ for some $c > 3$.

Deciding (ii).

The formula (ii) states that the set

$$\{\vec{x} \ \mid\ \vec{x} \in L \wedge \forall\ \vec{z} \in Add(L), \vec{y} \in L, \vec{x} \neq \vec{z} + \vec{y}\}$$

is finite. An automaton for

$$\{(\vec{x}, \vec{y}, \vec{z}) \mid\ \vec{z} \in Add(L), \vec{y} \in L, \vec{x} \neq \vec{z} + \vec{y}\}$$

has size $O(|\mathcal{A}|2^{|\mathcal{A}|^2})$. Therefore an automaton for

$$\{\vec{x} \ \mid\ \vec{x} \in L \wedge \forall\ \vec{z} \in Add(L), \vec{y} \in L, \vec{x} \neq \vec{z} + \vec{y}\}$$

has size $O(2^{|\mathcal{A}|2^{|\mathcal{A}|^2}})$.

Therefore the test for emptiness can be done in quadratic time in this size, i.e. $O(2^{2|\mathcal{A}|2^{|\mathcal{A}|^2}})$.

The main complexity arises from the computation of $\mathcal{I}r(Add(L))$ which yields $O(2^{2^{c|\mathcal{A}|^2}})$ for some $c > 3$. $\square$

7 Conclusion

We have presented a new method that allows to decide if an automaton accepting tuples of integers recognize a semilinear set of the form $L(\mathcal{C}, \mathcal{P})$. Moreover this method provides an effective construction of $\mathcal{C}$ and $\mathcal{P}$. We get doubly exponential bounds when we use tuples automata. The sets that are dealt with represent a class which is more general or independent from the other classes that have been considered until now by studying the structure of the automaton. Moreover the logical characterizations that we have exhibited for linear sets and semilinear sets of the form $L(\mathcal{C}, \mathcal{P})$ are interesting by themselves and can be relevant for other purposes. The most natural question is whether the method can be improved to get the whole class of semilinear sets. That would close the loop and provide another proof of Muchnik's result in the semilinear framework and it would give a complete picture of the relationships between the three formalisms. The complexity of the problem for our restricted class first, and then in the general case is also an issue.

References

[BFLP03] S. Bardin, A. Finkel, J. Leroux, and L. Petrucci. Fast acceleration of symbolic transition systems. In *Proc. 15th Int. Conf. on Computer Aided Verification*, volume 2725 of *Lecture Notes in Computer Science*, pages 118–121. Springer-Verlag, 2003.

[BHMV94] V. Bruyere, G. Hansel, C. Michaux, and R. Villemaire. Logic and p-recognizable sets of integers. *Bull. Bel. Math. Soc.*, 1:191–238, 1994.

[BW02] B. Boigelot and P. Wolper. Representing arithmetic constraints with finite automata: An overview. In P.J. Stuckey, editor, *Proc. ICLP*, number 2401 in Lecture Notes in Computer Science, pages 1–19. Springer-Verlag, 2002.

[Dic13] L. Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *Am. J. Math.*, 35::113122, 1913.

[Lat04] L. Latour. From automata to formulas: Convex integer polyhedra. In *Proc. 19th Symp. on Logic in Computer Science (LICS 2004)*. IEEE, 2004. to appear.

[Ler03] Jerôme Leroux. *Algorithmique de la vérification des systèmes à compteurs. Approximation et accélération. Implémentation de l'outil FAST.* PhD thesis, ENS-Cachan, December 2003. http://www.lsv.ens-cachan.fr/Publis/PAPERS/Leroux-these.ps.

[Muc03] A. Muchnik. The definable criterion for definability in Presburger arithmetic and its applications. *Theoretical Computer Science*, 290:1433–1444, 2003.

[RV02] T. Rybina and A. Voronkov. Using canonical representation of solutions to speed-up infinite-state model-checking. In *Computer Aided Verification, 14th International Conference, CAV*, volume 2404 of *Lecture Notes in Computer Science*, pages 386–400. Springer-Verlag, 2002.

[WB95] P. Wolper and B. Boigelot. An automata-theoretic approach to presburger arith- metic constraints. In *In Proceedings of SAS 95*, volume 983 of *Lecture Notes in Computer Science*, page 2132. Springer-Verlag, 1995.

Appendix

We give a proof that the equation $x_1 + \ldots + x_p = \alpha$ has $C^{\alpha}_{\alpha+p-1}$ solutions. The proof is by induction on p.

We decompose the solutions into disjoint sets of solutions such that $x_1 = 0, x_1 = 1, \ldots, x_1 = \alpha$. Each solution such that $x_1 = i$ is $(i, x_2, \ldots, x_p)$ where $(x_2, \ldots, x_p)$ is a solution of $x_2 + \ldots + x_p = \alpha - i$.

By induction hypothesis we get that the number of solutions is

$$C^{\alpha}_{\alpha+p-1-1} + C^{\alpha-1}_{\alpha+p-1-2} + \ldots + C^{0}_{\alpha+p-1-\alpha-1}$$

The identity $C^l_m = C^l_{m-1} + C^{l-1}_{m-1}$ for $0 \leq l < m$ is used repeatedly to prove that

$$C^l_m = C^l_{m-1} + C^{l-1}_{m-2} + C^{l-2}_{m-3} + \ldots + C^0_{m-l-1}$$

and we replace m, l by $\alpha + p - 1, \alpha$ to get the result.