On Linear Arithmetic with Stars

Ruzica Piskac and Viktor Kuncak

LARA - I&C - EPFL

Abstract. We consider an extension of integer linear arithmetic with a star operator that takes closure under vector addition of the set of solutions of linear arithmetic subformula. We show that the satisfiability problem for this language is in NP (and therefore NP-complete). Our proof uses a generalization of a recent result on sparse solutions of integer linear programming problems. We present two consequences of our result. The first one is an optimal decision procedure for a logic of sets, multisets, and cardinalities that has applications in verification, interactive theorem proving, and description logics. The second is NP-completeness of the reachability problem for a class of "homogeneous" transition systems whose transitions are defined using integer linear arithmetic formulas.

1 Introduction

Decision procedures and automated theorem provers [7, 3, 19, 12, 9] are among key techniques that enable automated verification of infinite state systems, as, for example, in software model checkers [4, 14, 18]. These techniques are also increasingly used to raise the level of automation in interactive theorem provers [27, 10, 8, 20, 2]. We believe that an important step towards making such theorem provers even more effective is the development of decision procedures for new classes of formulas that go beyond the traditionally considered uninterpreted function symbols, arrays, free data structures, and linear arithmetic. In this paper we present a decision procedure for one such class, which introduces certain *unbounded* sums into linear arithmetic. Specifically, our decision procedure solves the satisfiability problem

$$\exists \vec{v}, \vec{u}.\ \ P(\vec{v}, \vec{u}) \wedge \exists N \geq 0. \exists \vec{x}_1, \ldots, \vec{x}_N. \vec{u} = \sum_{i=1}^{N} \vec{x}_i \wedge \bigwedge F(\vec{x}_i) \tag{1}$$

where P and F are arbitrary quantifier-free Presburger arithmetic formulas (QFPA) and $\vec{v}, \vec{u}, \vec{x}_i$ denote vectors of non-negative integers. The fact that N is not known means that (1) is not immediately in Presburger arithmetic. We denote (1) by $P(\vec{v}, \vec{u}) \wedge \vec{u} \in \{\vec{x} \mid F(\vec{x})\}^*$, using notation $A^* = \{\vec{a}_1 + \ldots + \vec{a}_n \mid \vec{a}_1, \ldots, \vec{a}_n \in A\}$ for closure of a set of vectors under addition. The class of formulas that contain such $*$ operator is a natural extension of the QFPA, and our results show that it preserves NP-completeness of the satisfiability problem.

Our decision procedure enables reasoning about collections of objects (sets and multisets) and their cardinalities, which was our original motivation for

Our class of formulas: $F \wedge \vec{u} \in \{\vec{x} \mid F\}^*$
QFPA formulas:
$F ::= A \mid F \wedge F \mid \neg F$
$A ::= T \leq T \mid T = T$
$T ::= k \mid C \mid T + T \mid C \cdot T \mid \mathsf{ite}(F, T, T)$
terminals:
k - integer variable; C - integer constant

Fig. 1. QFPA and Our Class of Formulas

introducing it in [24], where we showed that it can be solved in PSPACE, but left the exact complexity open. This paper settles the question of the exact complexity by presenting a polynomial-time satisfiability-preserving reduction of (1) to QFPA, which is known to be in NP [23].

Moreover, in this paper we identify another application of our constraints. We consider infinite-state transitions systems whose state consists of finite control and a finite number of integer counters, and whose transitions increment counters by a solution vector of a linear arithmetic formula given by the finite control. We show that the reachability problem for a class of such systems reduces to (1) and is therefore solvable in NP in the size of system description.

Contributions. We summarize the contributions of our paper as follows:

- We present a polynomial-time algorithm (Section 3) for reducing (1) to satisfiability of quantifier-free Presburger arithmetic formulas, showing NP-completeness of (1);
- We present a new application of this satisfiability problem to reachability of a class of symbolically represented transition systems (Section 4).

2 From Multisets to Linear Arithmetic with Star

We have identified constraints (1) when considering the satisfiability problem for constraints on multisets in the presence of a cardinality operator. We here summarize briefly the relevance of such constraints as well as the basic idea of their reduction to (1). For more details, see [24].

Uses of multiset constraints. Many modern programs perform extensive manipulation of collections of objects implemented either as built in data structures or as collection libraries. Languages such as SETL [26] directly support sets as primitives, whereas the Gamma parallel programming paradigm [6, Page 103] is based on multiset transformations. Sets and multisets would directly arise in verification conditions for proving properties of such programs. In programming languages such as Java, data abstraction can be used to show that data structures satisfy set specifications, and then techniques based on sets become applicable for verifying data structure clients [16, 15, 21]. Multisets and sets also arise in other domains, as witnessed by their use in libraries of interactive provers Isabelle [22] and KIV [5], and their presence in the Sparql query language for

semantic web [1]. Decision procedures for sets and multisets are therefore useful for automated reasoning about such domains.

Our notions of multisets and sets. We represent sets multisets (also called *bags*) with their characteristic functions. A multiset m is a function $E \to \mathbb{N}$, where E is the universe and $\mathbb{N}$ is the set of non-negative integers. The value $m(e)$ is multiplicity (number of occurrences) of element e in multiset m. We assume that the domain E is fixed and finite but of unknown size. We represent sets within our formulas as special multisets m for which $m(e) = 0 \vee m(e) = 1$ for all elements e.

Operations on multisets. We consider a natural class of operations and relations on multisets that are given pointwise by linear arithmetic formulas. Given a relation $R(x_1, \ldots, x_k)$ on non-negative integers, we define the corresponding relation on multisets $m_1, \ldots, m_k$ by $\forall e.\, R(m_1(e), \ldots, m_k(e))$. For example, we define subset $m_1 \subseteq m_2$ by $\forall e. m_1(e) \leq m_2$, union $m_1 = m_2 \cup m_3$ by $\forall e. m_1(e) = \max(m_2(e), m_3(e))$, multiset sum $m_1 = m_2 \uplus m_3$ by $\forall e. m_1(e) = m_2(e) + m_3(e)$. We permit the use of if-then-else operator $\mathsf{ite}(F, t_1, t_2)$ in QFPA, which denotes t_1 when F holds and t_2 otherwise. For example, we can define multiset difference $m_1 = m_2 \setminus m_3$ by $\forall e. m_1(e) = \mathsf{ite}(m_2(e) \leq m_3(e), 0, m_3(e) - m_2(e))$.

Cardinality operator and sums. We also permit the cardinality operator $|m|$ on multisets, given by $|m| = \sum_{e \in E} m(e)$. This operator turns a multiset expression into an integer expression, so we allow arbitrary QFPA operators on it. Because we can define multiset m pointwise in terms of other multisets, we obtain the ability to express conditions such as $\sum_{e \in E} t(e)$ where $t(e)$ is any QFPA term containing addition, multiplication by constants, and $\mathsf{ite}(F, t_1, t_2)$ expressions for F arbitrary QFPA formula. It turns out to be convenient to generalize the summation operator and admit in our language summations $\sum_{e \in E}(t_1(e), \ldots, t_k(e))$ over vectors of QFPA terms.

Reduction to QFPA with star. Not only is the cardinality operator expressible in terms of $\sum_{e \in E} t(e)$, but any pointwise relation $\forall e. F(m_1(e), \ldots, m_k(e)$ where F is a QFPA formula is expressible by $0 = \sum_{e \in E} \mathsf{ite}(F, 0, 1)$. Moreover, conjunction of two summations $\vec{a} = \sum_{e \in E} \vec{t}(e)$ and $\vec{b} = \sum_{e \in E} \vec{s}(e)$ is equivalent to one $(\vec{a}, \vec{b}) = \sum_{e \in E}(\vec{t}(e), \vec{s}(e))$. We thus obtained [24, Theorem 1] that formulas in a natural and expressive language on sets, multisets, and cardinality constraints can be reduced to normal form

$$P(\vec{a}, \vec{b}) \wedge \vec{b} = \sum_{e \in E} \vec{t}(\vec{m}(e)) \tag{2}$$

where P is a QFPA formula, $\vec{t}$ is a vector of QFPA terms, and $\vec{m}(e)$ is the vector of multiset variables occurring in the original problem. Because E is of arbitrary finite size and because the terms are evaluated using same rule $\vec{t}$ for each $e \in E$, it is immediate to show [24, Theorem 2] that (2) is equisatisfiable with (1). Moreover, this reduction is polynomial-time, so the main difficulty in a decision procedure for multisets and cardinalities is in solving (1).

3 Linear Arithmetic with Star Operator is in NP

In this section we show that the problem $P \wedge u \in \{\vec{v} \mid F(v)\}^*$, stated as (1), is NP-complete. Because P is a QFPA formula containing arbitrary propositional operators, the problem is clearly NP-hard. The non-trivial part of NP-completeness is therefore showing membership in NP. Figure 2 shows our NP-algorithm. The algorithm takes as input $P, F, \vec{u}$. It produces, in polynomial time, a QFPA formula ϕ whose satisfiability is equivalent to (1), and then tests the satisfiability of ϕ.

The idea behind the construction of ϕ is the following. Because F is a QFPA formula, its solution set $\{\vec{v} \mid F(\vec{v})\}$ is semilinear set [13]. This implies the existence of finitely many *generating vectors* $\vec{a}_i$, $\vec{b}_{ij}$ whose linear combination is $\{\vec{v} \mid F(\vec{v})\}$. However, in worst case the number of generating vectors is exponential, so we avoid explicitly constructing them. We instead apply [25] to compute an upper bound on the size of generating vectors. This gives us bounds on coefficients in an *exponentially large* QFPA formula equisatisfiable with (1). We then combine two constructions to find a polynomially large equisatisfiable formula.

1. We apply a small model theorem for QFPA that follows from [23]. Because the exponential QFPA formula has only polynomially many atomic formulas, we obtain a polynomial bound on the number of bits needed for $\vec{u}$ in the smallest solution of (1).
2. We apply twice a theorem on the size of minimal generator of integer cone [11] to prove that only polynomially many generating vectors are sufficient.

Finally, we show that we can group linear combinations of generating vectors into linear combination of polynomially many variables denoting solution vectors of F. Despite the multiplication of variables, we can express such linear combination as a QFPA formula because coefficients in linear combination are bounded by the bound on $\vec{u}$.

We proceed to describe our construction in more detail, including concrete bounds needed to implement our algorithm.

3.1 Estimating Coefficient Bounds of Disjunctive Form

The results on which we rely are usually expressed for integer linear programming problems, so we review in Figure 3 how to compute dimensions and coefficient bounds for integer linear programming problems arising from QFPA formula.

Definition 1. *Let F be a QFPA formula and let F be converted into the disjunction of systems $\bigvee_{i=1}^{l} A_i\vec{x} = \vec{b}_i$ such that $F(\vec{v}) \Leftrightarrow \bigvee_{i=1}^{l} A_i\vec{v} = \vec{b}_i$. Let m_i be a number of rows in A_i and let n_i be a number of columns in A_i. Let $a_k = \max\{|a_{ij}^k|, |b_j^k|\}$. For the given F with m_F, n_F and a_F we denote an upper bound for m_i, n_i and a_i respectively.*

INPUT: A QFPA formula P with free variables $u_1, \ldots, u_p$, a vector $\vec{u} \in \mathbb{N}^d$, whose components are u_i variables and a QFPA formula F which does not share any variable with P and $\vec{u}$
OUTPUT: true iff $P \wedge \vec{u} \in \{\vec{v} \mid F(\vec{v})\}^*$ is satisfiable

procedure checkSat(F, P : QFPA; $\vec{u}$: vector)
// procedures for estimating values m, n and a are given in Figure 3
$m_F :=$ EstimateM(F); $n_F :=$ EstimateN(F); $a_F :=$ EstimateA(F);
$k := 4d(\log(4d) + 2m_F \log(2 + (n_F + 1)a_F))$;
// procedure for estimating the value $||u||_\infty$ is given in Figure 4
$r' := upperBound(u, P, F)$;
$r = \lceil \log r' \rceil$;
let $\vec{x}_1, \ldots, \vec{x}_k \in \mathbb{N}^d$ be fresh vector variables, let $\lambda_1, \ldots, \lambda_k \in \mathbb{N}$ be fresh variables of the size r and let $\lambda_i = \sum_{j=0}^{r} \lambda_j^i 2^j$ be a binary representation of λ_i;
form a new formula ϕ:

$$\phi \equiv P \wedge \vec{u} = \lambda_1 \otimes \vec{x}_1 + \ldots + \lambda_k \otimes \vec{x}_k \wedge \bigwedge_{i=1}^{k} F(\vec{x}_i),$$

where $\lambda_i \otimes \vec{x}_i = \sum_{j=0}^{r} 2^j \mathsf{ite}(\lambda_j^i, \vec{x}_i, 0)$
ϕ is a QFPA formula and equisatisfiable to $P \wedge \vec{u} \in \{\vec{v} \mid F(\vec{v})\}^*$
return true iff ϕ is satisfiable

Fig. 2. NP Algorithm for testing whether $P \wedge \vec{u} \in \{\vec{v} \mid F(\vec{v})\}^*$ is satisfiable

INPUT: A QFPA formula F
OUTPUT: m_F, n_F and a_F

let $s = \#$ simple subformulas of F
let $i = \#$ite operator in F
let $ine = \#$ inequations in F
let $e = \#$ite operator in F, where equality appears as a boolean condition

procedure EstimateM(F : QFPA)
return $s + 2 * i$
procedure EstimateN(F : QFPA)
return $=\#$ variables that occur in F $+ ine + e$
procedure EstimateA(F : QFPA)
return maximal absolute value of all integers occurring in F $+ 1$

Fig. 3. Upper Bounds on Value of m, n and a

Lemma 1. *Let F be a QFPA formula and let F be converted into the disjunction of systems $\bigvee_{i=1}^{l} A_i \vec{x} = \vec{b}_i$ such that $F(v) \Leftrightarrow \bigvee_{i=1}^{l} A_i \vec{v} = \vec{b}_i$. The values returned by the algorithm shown in Figure 3 are correct, i.e. they are upper bounds for the number of rows and columns and a maximal value appearing in every individual matrix A_i.*

Proof. While constructing a matrix A, we can assume that every simple formula will occur in A. However, if there is a subformula in F containing ite expression, for example $\vec{z} = \mathsf{ite}(C, \vec{x}, \vec{y})$, the matrix A can get two new rows: either C and $\vec{z} = \vec{x}$, or $\neg C$ and $\vec{z} = \vec{y}$. If booelan condition C is an equality $t_1 = t_2$, then its negation will produce further disjunctions, but however the matrix A will still get at most two new rows: either $t_1 < t_2$ and $z = y$, or $t_2 < t_1$ and $\vec{z} = \vec{y}$. This shows that a matrix A can have at most $s + 2 * i$ rows.

The number of columns in a matrix A is the number of variables that will occur in the system $A\vec{x} = \vec{b}$. Evidently, it depends on the number of variables occurring in F. Moreover there might be some fresh variables that will appear while converting inequations into equations. Therefore, one also needs to count the number of inequations. As shown before, every equation appearing as a boolean condition within ite operator will arise in some disjunct as an inequation. This proves an upper bound on the number of columns of a matrix A.

To derive an upper bound on a maximal value that might occur in a matrix A, it it sufficient to inspect absolute values of all integers that are present in F. However, there might also be strict inequations. Given a a strict inequation $\vec{x} < c$, it is equivalent to the equation: $\vec{x} + \vec{\varepsilon} = \vec{c} - \vec{1}$, where $\vec{\varepsilon}$ is a fresh variable, $\vec{\varepsilon} \geq 0$. Therefore, for integers occurring in inequations, we consider their maximal absolute value increased for 1. To the same group of integers belong integers appearing in conditional expressions within the ite operator. ∎

3.2 Existence and Size of Solution Set Generators

This section describes the solution of a QFPA formula F using generating vectors and provides bounds on these vectors.

Definition 2. *Given a finite set of vectors of non-negative integers $S \subseteq \mathbb{N}^n$ and a vector of non-negative integers $\vec{a} \in \mathbb{N}^n$, a linear set $L(a; S)$ is defined as $L(a; S) = \{\vec{a} + \vec{x}_1 + \ldots + \vec{x}_n \mid \vec{x}_i \in S \wedge n \geq 0\}$. A vector $\vec{a}$ is called* base *vector, while elements of S are called* step *vectors.*

A semilinear set is defined as a finite union of linear sets. Formally, a semilinear set Z is defined as $Z = \cup_{i=1}^{n} LS(a_i; S_i)$.

We call elements of $S(Z) = \bigcup_{i=1}^{n} (\{\vec{a}_i\} \cup S_i)$ generators of the semilinear set Z; there are always finitely many generators of a semilinear set.

[13] shows that a solution of a Presburger arithmetic formula is a semilinear set. Furthermore, [25] computes bounds on generators of this semilinear set. Combining these results with Lemma 1 and the definition of a semilinear set, we obtain the following Lemma 2.

Lemma 2. *Given a quantifier-free Presburger arithmetic formula F, there exists a semilinear set Z defined with vectors a_i and b_{ij} such that the folowing formula holds:*

$$\vec{u} \in \{\vec{v} \mid F(\vec{v})\} \quad \Leftrightarrow \quad \exists \nu_{ij}. \bigvee_{i=1}^{q} (\vec{u} = \vec{a}_i + \sum_{j=1}^{q_i} \nu_{ij} \vec{b}_{ij}) \tag{3}$$

Moreover, $||\vec{u}||_\infty \leq (2+(n+1)a)^{2m}$*, for every generator* $\vec{u} \in S(Z)$*, and previously introduced* m, n *and* a*.*

Given generators $\vec{a}_i, \vec{b}_{ij} \in S(Z)$ we can express membership in the set $\{\vec{v} \mid F(\vec{v})\}^*$ in Presburger arithmetic.

Lemma 3. *Let* F *by a QFPA formula with* $\vec{a}_i, \vec{b}_{ij}$ *its generators. The condition* $\vec{u} \in \{\vec{v} \mid F(\vec{v})\}^*$ *is then equivalent to*

$$\exists \mu_i, \nu_{ij}.\ \vec{u} = \sum_{i=1}^{q}(\mu_i \vec{a}_i + \sum_{j=1}^{q_i} \nu_{ij}\vec{b}_{ij}) \ \wedge \ \bigwedge_{i=1}^{q}(\mu_i = 0 \implies \sum_{j=1}^{q_i} \nu_{ij} = 0) \qquad (4)$$

3.3 Selecting Polynomially Many Generators

In this section we establish bounds on the number of generators needed to generate any particular solution vector $\vec{u}$: if $\vec{u}$ is a linear combination of generators, then it is also a linear combination of a polynomial subset of generators that form a smaller semilinear set. We prove this fact using a theorem about sparse solutions of integer linear programming problems. Given a set of vectors X and a vector $\vec{b} \in X^*$, the following theorem determines the bound on the number of vectors sufficient for representing $\vec{b}$ as a linear combination of vectors from X.

Theorem 1 (Theorem 1 (ii) in [11]). *Let* $X \subseteq \mathbb{Z}^d$ *be a finite set of integer vectors and let* $\vec{b} \in X^*$*. Then there exists a subset* $\tilde{X}$ *such that* $\vec{b} \in \tilde{X}^*$ *and* $|\tilde{X}| \leq 2d\log(4dM)$*, where* $M = \max_{x \in X} ||x||_\infty$*.*

Theorem 1 has been applied in [17] in order to establishing membership in NP for quantifier-free Boolean algebra with Presburger arithmetic. However, in the case of linear arithmetic with stars, it is not sufficient to apply the theorem, as we need to maintain semilinearity of a set. Therefore, we apply the theorem two times in particular order and we maintain semilinearity.

Theorem 2. *Let* F *be QFPA formula,* $Z = \{\vec{v} \mid F(\vec{v})\} \subseteq \mathbb{N}^d$ *its solution set,* $Z = \cup_{i=1}^{q} L(\vec{a}_i; \{\vec{b}_{i1}, \ldots, \vec{b}_{iq_i}\})$ *the semilinear set representation of the solution set, and* $\vec{u} \in Z^*$*. Then there exists a semilinear set* $\tilde{Z}$ *with* $S(\tilde{Z}) \subseteq S(Z)$*,* $\vec{u} \in \tilde{Z}$ *and*

$$|S(\tilde{Z})| \leq 6d(\log 4d + 2m\log(2 + (n+1)a))$$

and m, n, a *are computed according to Figure 3.*

Proof. Let $\vec{u} \in Z^*$. Then there exist vectors $\vec{a}_i, \vec{b}_{ij} \in S(Z)$ and non-negative integers μ_i and ν_{ij} such that $\vec{u} = \sum_{i=1}^{q}(\mu_i\vec{a}_i + \sum_{j=1}^{q_i} \nu_{ij}\vec{b}_{ij})$ and $\bigwedge_{i=1}^{q}(\mu_i = 0 \implies \sum_{j=1}^{q_i} \nu_{ij} = 0)$. Let vectors $\vec{a}$ and $\vec{b}$ be defined by $\vec{a} = \sum_i \mu_i\vec{a}_i$ and $\vec{b} = \sum_{ij} \nu_{ij}\vec{b}_{ij}$. Then $\vec{u} = \vec{a} + \vec{b}$. Let A be a multiset containing all the base vectors, $A = \uplus\{\vec{a}_i\}$ and let B be a multiset containing all the step vectors, $B = \uplus\{\vec{b}_{ij}\}$.

Note that $\vec{b} \in B^*$. According to Theorem 1, there exists $B_1 \subseteq B$ such that $\vec{b} \in {B_1}^*$ and $|B_1| \leq 2d\log(4dM)$ for $M = \max_{b \in B} ||\vec{v}||_\infty$. Using Lemma 2 to estimate M, we conclude $|B_1| \leq 2d(\log(4d) + 2m\log(2 + (n+1)a))$.

To construct a smaller semilinear set, every step vector needs the corresponding base vector. Let A_1 denote corresponding base vectors, $A_1 = \{\vec{a}_i \mid \exists \vec{b} \in B_1.\vec{b} \in S_i\}$. Then $|A_1| \leq |B_1|$ so also $|A_1| \leq 2d(\log(4d) + 2m\log(2 + (n+1)a))$. The base vectors A_1 and the steps vectors B_1 define the new semilinear set Z_1.

Let $\vec{a}_1 = \sum_{a \in A_1} \mu_i \vec{a}_i$. If vector $\vec{a}_i$ occurs multiple times in A_1, it is also added multiple times to the sum. Note that $\vec{a}_1 + \vec{b} \in Z_1^*$ and $|Z_1| \leq 4d(\log 4d + 2m\log(2 + (n+1)a))$.

Let us now consider the remaining base vectors and let $A_2 = A \setminus A_1$ (where $\setminus$ is the multiset difference) and let $\vec{a}_2 = \vec{a} - \vec{a}_1$. Then $\vec{a}_2 \in {A_2}^*$. By applying Theorem 1 once again to A_2 and a_2 we conclude that there exists a set A_3 such that $\vec{a}_2 \in {A_3}^*$ and $|A_3| \leq 2d(\log(4d) + 2m\log(2 + (n+1)a))$. This way we construct another semilinear set Z_2, which contains only base vectors, and $Z_2 = A_3$. Note that the vector $\vec{a}_2 \in Z_2^*$.

We finally let $\tilde{Z} = Z_1 \cup Z_2$. By construction, $\tilde{Z}$ is a semilinear set. Because $\vec{a}_2 \in Z_2^*$ and $\vec{a}_1 + \vec{b} \in Z_1^*$, and $\vec{u} = \vec{a}_2 + \vec{a}_1 + \vec{b}$, we conclude $\vec{u} \in \tilde{Z}^*$. By bounds on $|S(Z_1)|$ and $|S(Z_2)|$, we have $|S(\tilde{Z})| \leq 6d(\log 4d + 2m\log(2 + (n+1)a))$. ∎

3.4 Grouping Generators into Solutions

In previous two sections we have shown that if $\vec{u} \in \{\vec{v} \mid F(\vec{v})\}^*$, then $\vec{u}$ is a particular linear combination of polynomially many generating vectors $\vec{a}_i$, $\vec{b}_{ij}$ that are themselves polynomially bounded. This suggests the idea of guessing polynomially many bounded vectors, checking whether they are generators, and then checking whether $\vec{u}$ is their linear combination. We next show that we can avoid the problem of checking whether a vector is a generator and reduce the problem to checking whether a vector is solution of F.

Theorem 3. *Let F be a QFPA formula and $\vec{u} \in \{\vec{v} \mid F(\vec{v})\}^*$. Then there exist k vectors $\vec{c}_1, \ldots, c_k$ such that $\wedge_{i=1}^k F(\vec{c}_i)$, $u = \sum_{i=1}^k \lambda_i \vec{c}_i$ for some non-negative integers λ_i, where $k \leq 8d(\log(4d) + 2m\log(2 + (n+1)a))$.*

Proof. Let $Z = \{\vec{v} \mid F(\vec{v})\}$ and $\vec{u} \in Z^*$. By Theorem 2 there are polynomially many vectors $\vec{a}_i$ and $\vec{b}_{ij}$ that generate Z such that $\vec{u} = \sum_{i=1}^q (\mu_i \vec{a}_i + \sum_{j=1}^{q_i} \nu_{ij} \vec{b}_{ij})$ and if $\mu_i = 0$ then $\nu_{ij} = 0$ for all j. We can therefore assume that $\mu_i \geq 1$ and rewrite the sum as

$$\vec{u} = \sum_{i=1}^{q} ((\mu_i - 1)\vec{a}_i + \vec{a}_i + \sum_{j=1}^{q_i} \nu_{ij} \vec{b}_{ij})$$

Let $\vec{v}_i = \vec{a}_i + \sum_{j=1}^{q_i} \nu_{ij} \vec{b}_{ij}$. Clearly $\vec{a}_i \in Z$ and $\vec{v}_i \in Z$ because they are a linear combination of generators for Z. Therefore $F(a_i)$ and $F(v_i)$. We conclude $\vec{u} = \sum_{i=1}^k \lambda_i \vec{c}_i$ and $\bigwedge_{i=1}^k F(\vec{c}_i)$ where $k = 2q$ and q is bounded by the number of a_i vectors, which by the proof of Theorem 2 is $4d(\log(4d) + 2m\log(2 + (n+1)a))$. We thus obtain the desired bound. ∎

3.5 Multiplication by Bounded Bit Vectors

To express terms $\lambda_i \vec{c}_i$ from Theorem 3 as QFPA term, we use the fact that the smallest solution $\vec{u}$, if exists, is bounded. Figure 4 describes an algorithm that calculates an upper bound on $\vec{u}$. Theorem 4 proves its correctness.

Theorem 4. *Given a QFPA formula P with free variables $u_1, \ldots, u_p$, a vector $\vec{u} \in \mathbb{N}^d$, whose components are u_i variables and a QFPA formula F which does not share any variable with P and $\vec{u}$, if $P \wedge u \in \{v \mid F(v)\}^*$ is satisfiable, then there exist a non-negative vector $\vec{u}$ such that $||\vec{u}||_\infty \leq r'$ and r' is defined as in Figure 4.*

Proof. Theorem 2 implies that for some polynomially many polynomially bounded a_i, b_{ij} the original problem (1) is equisatisfiable with

$$P \wedge \vec{u} = \sum_i^q (\mu_i \vec{a}_i + \sum_j \nu_{ij} \vec{b}_{ij}) \wedge \bigwedge_i (\mu_i = 0 \implies \bigwedge_j \nu_{ij} = 0)$$

By doing case analysis on which $\mu_i = 0$, we can rewrite this formula as

$$P \wedge \bigvee_{I \subseteq \{1,\ldots,q\}} \vec{u} = \sum_{i \in I} ((1 + \mu_i)\vec{a}_i + \sum_j \nu_{ij} \vec{b}_{ij})$$

The disjunctive normal form of this formula represented as disjunction of integer linear programming problems has variables $\vec{v}, \vec{u}, \mu_i, \nu_{ij}$, where the number of μ_i, ν_{ij} follows from Theorem 2. This determines the number of columns of the matrix. The number of rows is the sum of the number of rows in P and the dimension d. The estimate in Figure 4 then follows from [23]. ∎

From $||\vec{u}||_{\leq} r'$ we conclude that $\lambda_i \leq r'$ in $\vec{u} = \sum_i \lambda_i c_i$. Each λ_i can therefore be represented as a bit-vector of the size r, where $r = \lceil \log r' \rceil$. Let $\lambda_i = \overline{\lambda_r^i \ldots \lambda_1^i \lambda_0^i} = \sum_{j=0}^r \lambda_j^i 2^j$. Then

$$\lambda_i \vec{c}_i = (\sum_{j=0}^r \lambda_j^i 2^j)\vec{c}_i = \sum_{j=0}^r 2^j (\lambda_j^i \vec{c}_i) = \sum_{j=0}^r 2^j \mathsf{ite}(\lambda_j^i, \vec{c}_i, 0) =$$
$$\mathsf{ite}(\lambda_0^i, \vec{c}_i, 0) + 2(\mathsf{ite}(\lambda_1^i, \vec{c}_i, 0) + 2(\mathsf{ite}(\lambda_2^i, \vec{c}_i, 0) + \ldots))$$

This completes the proof of correctness of the algorithm in Figure 2.

4 Reachability in a Class of Transition Systems

We next consider a problem of reachability in certain transition systems given by vector addition where sets of vectors are determined by QFPA formulas.

Let $\mathcal{F}_d$ be the set of all QFPA formulas with the set of free variables $v_1, \ldots, v_d$. If $F \in \mathcal{F}_d$ is such a formula and $a_1, \ldots, a_d \in \mathbb{Z}$, we write $(a_1, \ldots, a_d) \models F$ to denote the fact that F is true when v_i has value a_i for $1 \leq i \leq d$.

Consider a transition system described by a tuple (d, Q, E, T) where

INPUT: A QFPA formula P with free variables $u_1, \ldots, u_p$, a vector $\vec{u} \in \mathbb{N}^d$, whose components are u_i variables and a QFPA formula F which does not share any variable with P and $\vec{u}$
OUTPUT: non-negative integer r' such that, assuming that $P \wedge u \in \{v \mid F(v)\}^*$ is satisfiable, $||u||_\infty \leq r'$

procedure upperBound($\vec{u}$: vector; F, P : QFPA)
// procedures for estimating values m, n and a are given in Figure 3
$m_P :=$ EstimateM(P); $n_P :=$ EstimateN(P); $a_P :=$ EstimateA(P);
$m_F :=$ EstimateM(F); $n_F :=$ EstimateN(F); $a_F :=$ EstimateA(F);
$m := d + m_P$;
$n := n_P + 6d(\log(4d) + 2m_F \log(2 + (n_F + 1)a_F))$;
$a := \max\{a_P, (2 + (n + 1)a_F)^{2m_F}\}$;
return $n(ma)^{2m+1}$

Fig. 4. Algorithm that applies [23] to estimate bound on $\vec{u}$

1. d is a non-negative integer, denoting the number of integer variables in the state;
2. Q is a finite set, denoting program counter values;
3. $E \subseteq Q \times Q$, denoting control-flow graph edges;
4. $T : E \to \mathcal{F}_d$, specifies possible changes of counters for each control-flow graph edge.

Given (d, Q, E, T) we consider the set of states $S \subseteq Q \times \mathbb{Z}^d$ and define the transition relation $R \subseteq S \times S$ such that

$$(q, \vec{a}), (q', \vec{a}') \in R \iff (q, q') \in E \ \wedge \ (\vec{a}' - \vec{a}) \models T(q, q') \tag{5}$$

Note that, unlike in Turing-complete transition systems with integer counters, the set of possible counter changes is given by formula $T(q, q')$ and does not depend on the values of integer counters, but only on control-flow edge (q, q'), whose number is bounded.

We are interested in the question of reachability in the transition systems given by relation R. Consider first the case $Q = \{q\}$, $E = \{(q, q)\}$, $T(q, q) = F$. Our definitions then imply that $(q, \vec{a})$ reaches $(q, \vec{a}')$ precisely when the condition $(\vec{a}' - \vec{a}) \in \{\vec{v} \mid F\}^*$ holds, where $*$ denotes our vector addition closure operator. Therefore, the reachability problems that test QFPA relationship between initial $\vec{a}$ and final $\vec{a}'$ state in such systems reduces to (1).

We note that the assumption that variables denote *non-negative* integers in (1) is not a restriction because we can express arbitrary integer variables as differences of non-negative integer variables.

Now consider arbitrary (d, Q, E, T) and two states $q, q' \in Q$. Let r be a regular expression over the alphabet E describing the set of all paths from q to q' in graph (Q, E). We can assume that r exists and is polynomial in the number of elements of Q. Define set addition by $A + B = \{\vec{a} + \vec{b} \mid \vec{a} \in A, \vec{b} \in B\}$. We map r into a simpler regular expression with set addition acting as

commutative concatenation operator and Kleene star with vector set closure, using the following function h:

$$\begin{aligned} h((q_1, q_2)) &= \{x \mid T(q_1, q_2)\} \\ h(r_1 r_2) &= h(r_1) + h(r_2) \\ h(r_1 \cup r_2) &= h(r_1) \cup h(r_2) \\ h(r*) &= h(r)^* \end{aligned}$$

Due to commutativity of set addition and its consequence $A^* + B^* = (A \cup B)^*$, we can rewrite r in polynomial time to normal form stratified according to star height (the number of nested applications of $*$ operator). We call $\{v \mid T(q_1, q_2)\}$ atomic expressions and denote them a_{kij}. Then each commutative regular expression of star height $k > 0$ has the form $r_k = \cup_{i=1}^{p}(a_{ki1} + ... + a_{kin_i} + r_{i,k-1}^*)$ where $r_{i,k-1}$ are expressions of star height $k-1$. If $r = r_1$ i.e. r has no nested stars, then the reachability problem immediately reduces to (1) and is solvable in NP using our algorithm.

More generally, we show that linear arithmetic with regular expressions over solution sets of linear arithmetic formulas is in NP. Namely, it is not difficult to see that condition $\vec{u} \in r_k$ is equivalent to

$$\begin{aligned} &\exists(\vec{v}_{kij} \in a_{kij})_{kij}.\exists(\lambda_{kij})_{kij}. \\ &\quad \vec{x} = \sum_{k,i,j} \lambda_{kij}\vec{v}_{kij} \wedge \bigwedge_{\substack{k>1 \\ i,j}} (\lambda_{kij} = 0 \Rightarrow \bigwedge_{i',j'} \lambda_{(k-1)i'j'} = 0) \end{aligned}$$

As in Section 3 our goal is then to show that we can elect a subset of vectors in this linear combination and still generate vector $\vec{u}$.

The following notion of "star modulo vector dependencies" captures conditions on coefficients of linear combinations that arise from repeatedly applying star to semilinear sets. If $X = \{\vec{x}_1, \ldots, \vec{x}_N\} \subseteq \mathbb{N}^d$ is a finite set of vectors and $W \subseteq X \times X$ a dependency graph on X, define

$$X^{*(W)} = \{\sum_{i=1}^{N} \lambda_i \vec{x}_i \mid \forall i, j \leq N.\ \lambda_i > 0 \wedge (\vec{x}_i, \vec{x}_j) \in W \Rightarrow \lambda_j > 0\}$$

The dependency graph in Theorem 2 would have an edge from each $\vec{b}_{ij}$ to $\vec{a}_i$. The generalization of Theorem 1 to the class of graphs W sufficient for the more general result is the following.

Theorem 5. *Let $X \subseteq \mathbb{Z}^d$ be a finite set of integer vectors with acyclic dependency graph $W \subseteq X \times X$ such that for each node $\vec{x} \in X$ the number of nodes reachable from $\vec{x}$ in W is bounded by a constant C. If $\vec{b} \in X^{*(W)}$ then there exists $\tilde{X} \subseteq X$ such that $\vec{b} \in \tilde{X}^{*(W)}$ and $|\tilde{X}| \leq 2C^2 d \log(4dM)$, where $M = \max_{x \in X} ||x||_\infty$.*

Proof sketch. The idea of the proof of the theorem is to start from the source nodes of W (nodes with no incoming edges) and apply C times Theorem 1.

Let $B = 2C^2 d \log(4dM)$ from Theorem 1. Consider a linear combination $\vec{u} = \sum_i \lambda_i \vec{v}_i$ of vectors from X that satisfies the dependencies in W. Our goal is to find a small number of vectors that generate $\vec{u}$. In the first step we consider the source nodes of W, that is, vectors $Y_0 \subseteq X$ with no incoming edges in the graph. Applying Theorem 1 to $u_0 = \sum_{\vec{v}_i \in Y_0} \lambda_i \vec{v}_i$ we obtain a subset $Z_0 \subseteq Y_0$, with $|Z_0| \leq B$, such that $u_0 = \sum_{\vec{v}_i \in Z_0} \lambda'_i \vec{v}_i$. To enforce the constraints in the graph W, we then take closure of Z_0 under reachability in W and obtain the set Q_0 of size at most CB such that $u_0 \in {Q_0}^{*(W)}$.

Having obtained an efficient representation of $\vec{u}_0$, we continue to represent $\vec{u} - \vec{u}_0$. In the next step we therefore eliminate the sources Y_0 from the graph and consider the vectors that are sources in the subgraph of W induced by the remaining vectors $Y_1 = X \setminus Y_0$. We repeat this procedure as long as there are nodes in the graph. The number of times we need to repeat it is bounded by the longest path in W, which, by assumption, is bounded by C. At each step we select CB vectors, so the total number of nodes that we need in the linear combination is bounded by $C^2 B$. ∎

Using Theorem 5 we obtain a polynomial subset of vectors that satisfy given QFPA formulas and whose linear combination is the given vector $\vec{u}$. We then use results from previous sections to show that a linear combination of solutions of a QFPA formula can be represented as a sum of a polynomial number of solutions of this QFPA formula. This allows us to generalize results of Section 3 to formulas that contain not just one star operator but any regular expression over solution sets of QFPA formulas, which in turn proves that the reachability problem for transition systems described in this section is also in NP.

5 Conclusions

We showed that the satisfiability problem for an extensions of QFPA with star operators is decidable by polynomial-time reduction to QFPA. The result uses bounds on solutions of large QFPA formulas, as well as bounds on number of vectors needed to generate a solution. Our results yield optimal worst-case complexity for deductive verification of invariants that contain sets, multisets, and cardinality constraints, as well as reachability checking for certain counter systems.

Acknowledgements. We thank Nikolaj Bjørner for useful comments on a draft of this paper.

References

1. SPARQL query language for RDF. http://www.w3.org/TR/rdf-sparql-query/, 15 January 2008. W3C Recommendation.
2. Konstantine Arkoudas, Karen Zee, Viktor Kuncak, and Martin Rinard. Verifying a file system implementation. In *Sixth International Conference on Formal Engineering Methods (ICFEM'04)*, volume 3308 of *LNCS*, Seattle, Nov 8-12, 2004 2004.

3. Thomas Ball, Byron Cook, Shuvendu K. Lahiri, and Lintao Zhang. Zapato: Automatic theorem proving for predicate abstraction refinement. In *Tool Paper, CAV*, 2004.
4. Thomas Ball, Rupak Majumdar, Todd Millstein, and Sriram K. Rajamani. Automatic predicate abstraction of C programs. In *Proc. ACM PLDI*, 2001.
5. M. Balser, W. Reif, G. Schellhorn, K. Stenzel, and A. Thums. Formal system development with KIV. In T. Maibaum, editor, *Fundamental Approaches to Software Engineering*, number 1783 in LNCS. Springer, 2000.
6. Jean-Pierre Banâtre and Daniel Le Métayer. Programming by multiset transformation. *Commun. ACM*, 36(1):98–111, 1993.
7. Clark Barrett and Sergey Berezin. CVC Lite: A new implementation of the cooperating validity checker. In *CAV*, volume 3114 of *Lecture Notes in Computer Science*, pages 515–518, 2004.
8. David Basin and Stefan Friedrich. Combining WS1S and HOL. In D.M. Gabbay and M. de Rijke, editors, *Frontiers of Combining Systems 2*, volume 7 of *Studies in Logic and Computation*, pages 39–56. Research Studies Press/Wiley, Baldock, Herts, UK, February 2000.
9. Leonardo de Moura and Nikolaj Bjørner. Efficient E-matching for SMT solvers. In *CADE*, 2007.
10. L. A. Dennis, G. Collins, M. Norrish, R. Boulton, K. Slind, G. Robinson, M. Gordon, and T. Melham. The PROSPER toolkit. In S. Graf and M. Schwartbach, editors, *Tools and Algorithms for Constructing Systems (TACAS 2000)*, number 1785 in Lecture Notes in Computer Science, pages 78–92. Springer-Verlag, 2000.
11. Friedrich Eisenbrand and Gennady Shmonin. Carathéodory bounds for integer cones. *Operations Research Letters*, 34(5):564–568, September 2006. http://dx.doi.org/10.1016/j.orl.2005.09.008.
12. Yeting Ge, Clark Barrett, and Cesare Tinelli. Solving quantified verification conditions using satisfiability modulo theories. In *CADE*, 2007.
13. S. Ginsburg and E. Spanier. Semigroups, Pressburger formulas and languages. *Pacific Journal of Mathematics*, 16(2):285–296, 1966.
14. Thomas A. Henzinger, Ranjit Jhala, Rupak Majumdar, and Grégoire Sutre. Lazy abstraction. In *POPL*, 2002.
15. Viktor Kuncak. *Modular Data Structure Verification*. PhD thesis, EECS Department, Massachusetts Institute of Technology, February 2007.
16. Viktor Kuncak, Patrick Lam, Karen Zee, and Martin Rinard. Modular pluggable analyses for data structure consistency. *IEEE Transactions on Software Engineering*, 32(12), December 2006.
17. Viktor Kuncak and Martin Rinard. Towards efficient satisfiability checking for Boolean Algebra with Presburger Arithmetic. In *CADE-21*, 2007.
18. Shuvendu K. Lahiri and Randal E. Bryant. Indexed predicate discovery for unbounded system verification. In *CAV'04*, 2004.
19. Shuvendu K. Lahiri and Sanjit A. Seshia. The UCLID decision procedure. In *CAV'04*, 2004.
20. Sean McLaughlin, Clark Barrett, and Yeting Ge. Cooperating theorem provers: A case study combining HOL-Light and CVC Lite. In *Proc. 3^{rd} Workshop on Pragmatics of Decision Procedures in Automated Reasoning (PDPAR '05)*, volume 144(2) of *Electronic Notes in Theoretical Computer Science*, pages 43–51. Elsevier, January 2006.

21. Huu Hai Nguyen, Cristina David, Shengchao Qin, and Wei-Ngan Chin. Automated verification of shape, size and bag properties via separation logic. In *VMCAI*, 2007.
22. Tobias Nipkow, Markus Wenzel, Lawrence C Paulson, and Norbert Voelker. Multiset theory version 1.30 (Isabelle distribution). http://isabelle.in.tum.de/dist/library/HOL/Library/Multiset.html, 2005.
23. Christos H. Papadimitriou. On the complexity of integer programming. *J. ACM*, 28(4):765–768, 1981.
24. Ruzica Piskac and Viktor Kuncak. Decision procedures for multisets with cardinality constraints. In *9th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI)*, LNCS, 2008.
25. Loïc Pottier. Minimal solutions of linear diophantine systems: Bounds and algorithms. In *RTA*, volume 488 of *LNCS*, 1991.
26. J. T. Schwartz. On programming: An interim report on the SETL project. Technical report, Courant Institute, New York, 1973.
27. Natarajan Shankar. Using decision procedures with a higher-order logic. In *Proc. 2001 International Conference on Theorem Proving in Higher Order Logics*, 2001.