Formal Methods in System Design
https://doi.org/10.1007/s10703-023-00440-z

Termination of triangular polynomial loops

Marcel Hark[1] · **Florian Frohn**[1,2] · **Jürgen Giesl**[1]

Received: 16 October 2021 / Accepted: 20 September 2023

Abstract

We consider the problem of proving termination for triangular weakly non-linear loops (*twn*-loops) over some ring $\mathcal{S}$ like $\mathbb{Z}$, $\mathbb{Q}$, or $\mathbb{R}$. The guard of such a loop is an arbitrary quantifier-free Boolean formula over (possibly non-linear) polynomial inequations, and the body is a single assignment of the form $\begin{bmatrix} x_1 \\ \dots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \dots \\ c_d \cdot x_d + p_d \end{bmatrix}$ where each x_i is a variable, $c_i \in \mathcal{S}$, and each p_i is a (possibly non-linear) polynomial over $\mathcal{S}$ and the variables $x_{i+1}, \dots, x_d$.

We show that the question of termination can be reduced to the existential fragment of the first-order theory of $\mathcal{S}$. For loops over $\mathbb{R}$, our reduction implies decidability of termination. For loops over $\mathbb{Z}$ and $\mathbb{Q}$, it proves semi-decidability of non-termination.

Furthermore, we present a transformation to convert certain non-*twn*-loops into *twn*-form. Then the original loop terminates iff the transformed loop terminates over a specific subset of $\mathbb{R}$, which can also be checked via our reduction. Moreover, we formalize a technique to *linearize* (the updates of) *twn*-loops in our setting and analyze its complexity. Based on these results, we prove complexity bounds for the termination problem of *twn*-loops as well as *tight* bounds for two important classes of loops which can *always* be transformed into *twn*-loops.

Finally, we show that there is an important class of linear loops. where our decision procedure results in an *efficient* procedure for termination analysis, i.e., where the parameterized complexity of deciding termination is *polynomial*.

Keywords Termination · Polynomial loops · Decision procedure · Complexity · Closed form

✉ Jürgen Giesl
giesl@informatik.rwth-aachen.de

Marcel Hark
marcel.hark@cs.rwth-aachen.de

Florian Frohn
florian.frohn@cs.rwth-aachen.de

[1] LuFG Informatik 2, RWTH Aachen University, Aachen, Germany

[2] Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany

Published online: 04 December 2023

1 Introduction

Termination is one of the most important properties of a program. In this work, we study *complete* approaches for analyzing termination of certain classes of polynomial loops. Compared to incomplete techniques, such approaches have the advantage that they always yield a definite result. In particular, we investigate *decidability* of termination for our classes of loops.

In the following, we give a short overview on the contributions of our article and highlight how it extends our earlier conference paper [20]. There are already several decidability results for termination of linear loops [8, 10, 19, 31, 42, 45, 49, 62, 66], but only few results on the decidability of termination for certain forms of non-linear loops [43, 44, 47, 64, 65]. Moreover, these previous works only deal with loops whose guards only contain conjunctions, besides [47] which is restricted to guards defining compact sets. In this work, we regard possibly non-linear loops with arbitrary guards, i.e., they may also contain disjunctions and define non-compact sets. More precisely, we consider so-called *twn*-loops, where the update is mildly restricted to be "triangular" and "weakly non-linear" (see Sect. 2 for a formal definition). We study such loops over $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}_{\mathbb{A}}$ (the real algebraic numbers), and $\mathbb{R}$, whereas existing decidability results for non-linear loops are restricted to loops over the reals.

Most techniques for proving termination of loops rely on *polynomial ranking functions*, see, e.g., [1, 5–7, 9, 51]. However, such ranking functions are only *sound* for proving termination, i.e., in general, they cannot refute termination. In contrast to ranking functions, we use the computability of *closed forms* for the iterated update of the loop (Sect. 3). In this way, we can reduce termination of a loop to (in)validity of a certain formula. This reduction, which is a generalization of our earlier results for linear loops over $\mathbb{Z}$ with conjunctive guards [19], is sound and complete, i.e., validity of the resulting formula proves non-termination, whereas invalidity implies termination. Moreover, our reduction is computable. Analogously to our earlier work for linear loops [19], our decidability results on termination then follow from existing results on the decidability of certain theories. In this way, we show that termination of *twn*-loops is decidable over $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$, and non-termination is semi-decidable over $\mathbb{Z}$ and $\mathbb{Q}$ (Sect. 4).

In Sect. 5 we use concepts from algebra to enlarge the classes of loops to which our reduction is applicable. This is done by transforming (certain) non-*twn* loops into *twn*-form without affecting their termination behavior (Sect. 5.1). In Sect. 5.2, we discuss for which loops our transformation is applicable. In this way, we generalize our results to a broader class of polynomial loops (Sect. 5.3).

In contrast to our earlier conference paper [20], in Sect. 6 we formalize the technique of [48] to linearize (the updates of) *twn*-loops in our setting. Using this formalization, we develop novel results on the complexity of linearization.

Afterwards, based on our decision procedure for termination in Sect. 4, on the transformation of Sect. 5, and on our complexity results for linearization from Sect. 6, we study the complexity of deciding termination in Sect. 7.

In Sect. 7.1 we show that deciding termination of linear loops with rational spectrum over $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}_{\mathbb{A}}$, and $\mathbb{R}$ is Co-NP-complete. Moreover, we show that deciding termination of linear-update loops (where the update is linear but the guard may be non-linear) with real spectrum over $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$ is $\forall\mathbb{R}$-complete. Here, a loop has *rational* or *real spectrum*, respectively, if its update matrix has rational or real eigenvalues only, and $\forall\mathbb{R}$ is the complexity class of problems which can be reduced to validity of a universally quantified formula of polynomial inequations over the reals. We also analyze the complexity of deciding termination for arbitrary *twn*-loops (with possibly non-linear updates) in Sect. 7.2. In our

$$\textbf{while } (\varphi) \textbf{ do } \vec{x} \leftarrow \vec{u}$$

Fig. 1 Polynomial loop

conference paper [20, Thm. 45], we had only analyzed this case for a *bounded* number of variables. In contrast, we now extend our analysis to the general case where the number of variables is *not* restricted (Theorem 7.6). To this end, we need our new results from Sect. 6 on the complexity of linearizing *twn*-loops.

Finally, in contrast to [20], we identify a class of linear loops (*uniform loops*) where termination can be interpreted as a parameterized problem which is decidable in polynomial time when fixing such a parameter (Sect. 7.3). Based on the transformation of Sect. 5, we show that the closed forms arising from uniform loops have a special structure. Therefore, here (in)validity of the formula from Sect. 4 which encodes termination can be checked in polynomial time.

Related work is discussed in Sect. 8 and all missing proofs can be found in App. A. So the current paper extends [20] by the following new material:

- Section 6 on the linearization of *twn*-loops and its complexity.
- Theorem 7.6 on the complexity of deciding termination for arbitrary *twn*-loops where the number of variables is *not* restricted.
- Section 7.3 on *uniform loops* where the parameterized complexity of deciding termination is *polynomial*.
- Several additional explanations and remarks.

2 Preliminaries

A *(polynomial) loop over a ring* $\mathcal{S}$ has the form in Fig. 1, where $\mathbb{Z} \leq \mathcal{S} \leq \mathbb{R}_{\mathbb{A}}$ and $\leq$ denotes the subring relation. Here, $\vec{x}$ is a vector of $d \geq 1$ pairwise different variables that range over $\mathcal{S}$ and $\vec{u} \in \left(\mathcal{S}[\vec{x}]\right)^d$ where $\mathcal{S}[\vec{x}]$ is the set of polynomials over $\vec{x}$ with coefficients in $\mathcal{S}$. To improve readability, we use row- and column-vectors interchangeably. The guard φ is an arbitrary propositional (i.e., quantifier-free) formula over the atoms $\{p \rhd 0 \mid p \in \mathcal{S}[\vec{x}], \rhd \in \{\geq, >\}\}$. We denote the set of all such formulas by $Th_{qf}(\mathcal{S})$. In our setting, negation is syntactic sugar as, e.g., $\neg(p > 0)$ is equivalent to $-p \geq 0$. So w.l.o.g. the guard (or *condition*) φ of a loop is built from atoms, $\wedge$, and $\vee$.

We require $\mathcal{S} \leq \mathbb{R}_{\mathbb{A}}$ instead of $\mathcal{S} \leq \mathbb{R}$, as it is unclear how to represent transcendental numbers on computers. However, in Sect. 4 we will see that the loops considered in this work terminate over $\mathbb{R}$ iff they terminate over $\mathbb{R}_{\mathbb{A}}$. Thus, our results immediately carry over to loops where the variables range over $\mathbb{R}$. Hence, we sometimes also consider loops over $\mathcal{S} = \mathbb{R}$. However, even then we restrict ourselves to loops where all constants in φ and $\vec{u}$ are algebraic.

We often represent a loop as in Fig. 1 by the tuple $(\varphi, \vec{u})$ of the *condition* φ and the *update* $\vec{u} = (u_1, \dots, u_d)$. Unless stated otherwise, $(\varphi, \vec{u})$ is always a loop over $\mathcal{S}$ using the variables $\vec{x} = (x_1, \dots, x_d)$ where $\mathbb{Z} \leq \mathcal{S} \leq \mathbb{R}_{\mathbb{A}}$. We use the following notions for certain classes of loops: A *linear-update loop* has the form $(\varphi, A \cdot \vec{x} + \vec{b})$, and it has *rational* or *real spectrum*, respectively, if A has rational or real eigenvalues only. Then all eigenvalues of A are real algebraic numbers, since $\vec{u} \in \left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)^d$ implies that all constants in $\vec{u}$ (and thus in A) are algebraic. A *linear loop* is a linear-update loop where φ is linear (i.e., its atoms are only linear inequations). Here, "linear" refers to "linear polynomials", i.e., of degree at most 1, so it includes *affine* functions. A *conjunctive loop* is a loop $(\varphi, \vec{u})$ where φ does not contain disjunctions.

For any entity s, $s[x/t]$ results from s by replacing all free occurrences of x by t. Similarly, if $\vec{x} = (x_1, \ldots, x_d)$ and $\vec{t} = (t_1, \ldots, t_d)$, then $s[\vec{x}/\vec{t}]$ results from s by replacing all free occurrences of x_i by t_i, for each $1 \leq i \leq d$.

Any vector of polynomials $\vec{u} \in \left(\mathcal{S}[\vec{x}]\right)^d$ can also be regarded as a function $\vec{u} : \left(\mathcal{S}[\vec{x}]\right)^d \to \left(\mathcal{S}[\vec{x}]\right)^d$ where for any $\vec{p} \in \left(\mathcal{S}[\vec{x}]\right)^d$, $\vec{u}(\vec{p}) = \vec{u}[\vec{x}/\vec{p}]$ results from *applying* the polynomials $\vec{u}$ to the polynomials $\vec{p}$. Similarly, we can also apply a formula to polynomials $\vec{p} \in \left(\mathcal{S}[\vec{x}]\right)^d$. To this end, we define $\psi(\vec{p}) = \psi[\vec{x}/\vec{p}]$ for first-order formulas ψ with the free variables $\vec{x}$. As usual, function application associates to the left, i.e., $\vec{u}(\vec{b})(\vec{p})$ stands for $(\vec{u}(\vec{b}))(\vec{p})$. However, obviously $(\vec{u}(\vec{b}))(\vec{p}) = \vec{u}(\vec{b}(\vec{p}))$ since applying polynomials only means that one instantiates variables.

Definition 2.1 formalizes the intuitive notion of termination for a loop $(\varphi, \vec{u})$ and the related notion of *eventual* termination [10, 62]. Here, $\vec{u}^n$ denotes the n-fold application of $\vec{u}$, i.e., $\vec{u}^0(\vec{e}) = \vec{e}$ and $\vec{u}^{n+1}(\vec{e}) = \vec{u}(\vec{u}^n(\vec{e}))$.

Definition 2.1 (Termination) Let $(\varphi, \vec{u})$ be a loop over $\mathcal{S}$ and $\vec{e} \in \mathcal{S}^d$.

If $\forall n \in \mathbb{N}.\ \varphi(\vec{u}^n(\vec{e}))$ holds, then $\vec{e}$ is a *witness for non-termination*.

If $(\varphi, \vec{u})$ does not have any witnesses for non-termination, it *terminates* (over $\mathcal{S}$).

If $\vec{u}^{n_0}(\vec{e})$ is a witness for non-termination for some $n_0 \in \mathbb{N}$, then $\vec{e}$ is called a *witness for eventual non-termination*.

$(E)NT_{(\varphi,\vec{u})}$ denotes the set of witnesses for (eventual) non-termination of $(\varphi, \vec{u})$ and we define $T_{(\varphi,\vec{u})} = \mathcal{S}^d \setminus NT_{(\varphi,\vec{u})}$.

For any entity s, $\mathcal{V}(s)$ is the set of all its free variables. Given an assignment $\vec{x} \leftarrow \vec{u}$, the relation $\succ_{\vec{u}} \in \mathcal{V}(\vec{u}) \times \mathcal{V}(\vec{u})$ is the transitive closure of $\{(x_i, x_j) \mid i,j \in \{1, \ldots, d\}, i \neq j, x_j \in \mathcal{V}(u_i)\}$, i.e., $x_i \succ_{\vec{u}} x_j$ means that x_i depends on x_j. For example, if $\vec{u} = (x_1 + x_2^2,\ x_2 + 1)$ then we have $\succ_{\vec{u}} = \{(x_1, x_2)\}$.

Now we can introduce the class of *twn*-loops. A loop $(\varphi, \vec{u})$ is *triangular* if $\succ_{\vec{u}}$ is well founded. It is *weakly non-linear* if there is no $1 \leq i \leq d$ such that x_i occurs in a non-linear monomial of u_i, i.e., $u_i = c_i \cdot x_i + p_i$ where $c_i \in \mathcal{S}$ and $p_i \in \mathcal{S}[\vec{x}]$ does *not* contain x_i. A *twn-loop* is triangular and weakly non-linear. We call a loop *non-negative* if it is weakly non-linear and the coefficient c_i of the monomial x_i in u_i is non-negative for all $1 \leq i \leq d$. A *tnn-loop* is triangular and non-negative, i.e., a *tnn*-loop is a special form of a *twn*-loop.

The restriction to triangular loops prohibits “cyclic dependencies” of variables (e.g., where the new values of x_1 and x_2 both depend on the old values of x_1 and x_2). For example, the loop whose body consists of the assignment $(x_1, x_2) \leftarrow (x_1 + x_2^2,\ x_2 + 1)$ is triangular since $\succ = \{(x_1, x_2)\}$ is well founded, whereas a loop with the body $(x_1, x_2) \leftarrow (x_1 + x_2^2,\ x_1 + 1)$ is not triangular.

By the restriction to *twn*-loops we can compute a closed form for the n-fold application of the update $\vec{u}$ by handling one variable after the other. A vector $\vec{q}$ of d arithmetic expressions over $\vec{x}$ and a distinguished variable n is a *closed form* for $\vec{u} \in (\mathcal{S}[\vec{x}])^d$ if $\vec{q}[\vec{x}/\vec{e}, n/n'] = \vec{u}^{n'}(\vec{e})$ for all $\vec{e} \in \mathcal{S}^d$ and $n' \in \mathbb{N}$, i.e., both vectors of expressions evaluate to the same element of $\mathcal{S}^d$. Thus, $\vec{q} = \vec{u}^n$.

Triangular loops are very common in practice. For example, in [21], 1511 polynomial loops were extracted from the *Termination Problems Data Base (TPDB)* [63], the

$$\textbf{while } (x_1 + x_2^2 > 0) \textbf{ do } \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} x_1 + x_2^2 \cdot x_3 \\ x_2 - 2 \cdot x_3^2 \\ x_3 \end{bmatrix}$$

Fig. 2 Loop $\mathbf{L}_{ex}$ over $\mathbb{Z}$

benchmark collection used at the annual *Termination and Complexity Competition* [25], and only 26 of them were non-triangular.

A loop with the body $(x_1, x_2) \leftarrow (x_1 + x_2^2,\ x_2 + 1)$ is weakly non-linear, while a loop with $(x_1, x_2) \leftarrow (x_1 \cdot x_2,\ x_2 + 1)$ is not. In particular, weak non-linearity excludes assignments like $x_1 \leftarrow x_1^2$ that need exponential space, as x_1 grows doubly exponentially. By permuting variables, the update of every *twn*-loop can be transformed to the following form where $c_i \in \mathcal{S}$ and $p_i \in \mathcal{S}[x_{i+1}, \dots, x_d]$:

$$\begin{bmatrix} x_1 \\ \dots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \dots \\ c_d \cdot x_d + p_d \end{bmatrix}$$

Example 2.2 Consider the loop $\mathbf{L}_{ex}$ over the ring $\mathbb{Z}$ in Fig. 2. This loop is triangular since $\succ_{\vec{u}} = \{(x_1, x_2), (x_1, x_3), (x_2, x_3)\}$ is well founded. Moreover, it is weakly non-linear. Since the coefficient of x_i is 1 in the update of x_i for all $1 \le i \le 3$, this loop is also non-negative, i.e., $\mathbf{L}_{ex}$ is *tnn*.

Our *twn*-loops are a special case of *solvable loops*.

Definition 2.3 (Solvable Loops [56]) A loop $(\varphi, \vec{u})$ is *solvable* if there is a partitioning $\mathcal{J} = \{J_1, \dots, J_k\}$ of $\{1, \dots, d\}$ such that for each $1 \le i \le k$ we have $\vec{u}_{J_i} = A_i \cdot \vec{x}_{J_i} + \vec{p}_i$, where $\vec{u}_{J_i}$ is the vector of all u_j with $j \in J_i$ (and $\vec{x}_{J_i}$ is defined analogously), $d_i = |J_i|$, $A_i \in \mathcal{S}^{d_i \times d_i}$, and $\vec{p}_i \in (\mathcal{S}[\vec{x}_{J_{i+1}}, \dots, \vec{x}_{J_k}])^{d_i}$.

The eigenvalues of a solvable loop are the union of the eigenvalues of all A_i.

So solvable loops allow for blocks of variables with linear dependencies, and *twn*-loops correspond to the case that all blocks have size 1. While our approach could be generalized to solvable loops with real eigenvalues, Theorem 5.15 (Sect. 5) shows that this generalization does not increase its applicability.

For our decidability results in Sect. 4, we reduce termination to the *existential fragment* $\mathrm{Th}_\exists(\mathcal{S})$ of the first-order theory of $\mathcal{S}$ (see, e.g., [52, 57]). $\mathrm{Th}_\exists(\mathcal{S})$ consists of all formulas $\exists \vec{y} \in \mathcal{S}^k .\, \psi$ where $k \in \mathbb{N}$ and the propositional formula ψ is built from $\wedge$ and $\vee$ over the atoms $\{p \rhd 0 \mid p \in \mathbb{Q}[\vec{y}, \vec{z}],\ \rhd \in \{\ge, >\}\}$. Here, $\vec{y}$ and $\vec{z}$ are pairwise disjoint vectors of variables. The *free* variables $\vec{z}$ range over $\mathbb{R}_\mathbb{A}$ and they are needed in Sect. 5 to characterize subsets of real (algebraic) numbers by formulas.

The *existential fragment of the first-order theory of $\mathcal{S}$ and $\mathbb{R}_\mathbb{A}$* is the set $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$ of all formulas $\exists \vec{y}\,' \in \mathbb{R}_\mathbb{A}^{k'}, \vec{y} \in \mathcal{S}^k .\, \psi$, with a propositional formula ψ over $\{p \rhd 0 \mid p \in \mathbb{Q}[\vec{y}\,', \vec{y}, \vec{z}],\ \rhd \in \{\ge, >\}\}$ where $k', k \in \mathbb{N}$ and the variables $\vec{y}\,'$, $\vec{y}$, and $\vec{z}$ are pairwise disjoint. A formula without free variables is *closed*.

In the following, we also consider formulas over inequations $p \rhd 0$ where p's coefficients are from $\mathbb{R}_{\mathbb{A}}$ to be elements of $\mathrm{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$ (resp. $\mathrm{Th}_{\exists}(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$). The reason is that real algebraic numbers are $\mathrm{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$-definable.

Validity of closed formulas from $\mathrm{Th}_{\exists}(\mathcal{S})$ or $\mathrm{Th}_{\exists}(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$ is decidable if $\mathcal{S} \in \{\mathbb{R}_{\mathbb{A}}, \mathbb{R}\}$ and semi-decidable if $\mathcal{S} \in \{\mathbb{Z}, \mathbb{Q}\}$ [13, 61]. By undecidability of Hilbert's Tenth Problem [46], it is undecidable for $\mathcal{S} = \mathbb{Z}$. While validity of full first-order formulas (i.e., also containing universal quantifiers) over $\mathcal{S} = \mathbb{Q}$ is undecidable [53], it is still open whether validity of closed formulas from $\mathrm{Th}_{\exists}(\mathbb{Q})$ or $\mathrm{Th}_{\exists}(\mathbb{Q}, \mathbb{R}_{\mathbb{A}})$ is decidable. However, validity of *linear* closed formulas from $\mathrm{Th}_{\exists}(\mathcal{S})$ or $\mathrm{Th}_{\exists}(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$ is decidable for all $\mathcal{S} \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}_{\mathbb{A}}, \mathbb{R}\}$ [15, 26, 33, 37]. Here, a formula is *linear* if it only contains atoms $p \rhd 0$ where p is linear.

3 Reducing termination of *twn*-loops to termination of *tnn*-loops

For analyzing termination of *twn*-loops, we can restrict ourselves to *tnn*-loops, as *twn*-loops can be (automatically) transformed into *tnn*-loops via *chaining*.

Definition 3.1 (Chaining) *Chaining* a loop $(\varphi, \vec{u})$ yields $(\varphi \wedge \varphi(\vec{u}), \vec{u}(\vec{u}))$.

So for example, chaining the loop $(x_1 > 0, -x_1)$ (i.e., "**while** $(x_1 > 0)$ **do** $x_1 \leftarrow -x_1$") yields $(x_1 > 0 \wedge -x_1 > 0,\ x_1)$. Analogous to [19] where chaining was used for triangular linear loops, we obtain the following theorem.

Theorem 3.2 (Soundness of Chaining) *Let* $(\varphi, \vec{u})$ *be a twn-loop on* $\mathcal{S}^d$.

(a) $(\varphi \wedge \varphi(\vec{u}), \vec{u}(\vec{u}))$ *is tnn, i.e., it is triangular and the coefficient of each* x_i *in* $u_i(\vec{u})$ *is non-negative.*
(b) $(\varphi, \vec{u})$ *terminates on* $\vec{e} \in \mathcal{S}^d$ *iff* $(\varphi \wedge \varphi(\vec{u}), \vec{u}(\vec{u}))$ *terminates on* $\vec{e}$.

As chaining is clearly computable, we get the following corollary.

Corollary 3.3 *Termination of twn-loops is reducible to termination of tnn-loops.*

It is well known that closed forms for *tnn*-loops are computable (see, e.g., [35, 67]) since their bodies correspond to *C-finite recurrences*, which are known to be solvable [34]. The resulting closed forms may contain polynomial arithmetic and exponentiation w.r.t. n (as, e.g., $x_1 \leftarrow 2 \cdot x_1$ has the closed form $x_1 \cdot 2^n$) as well as certain piecewise defined functions. For example, the closed form $x_1^{(n)}$ of $x_1 \leftarrow 1$ is $x_1^{(0)} = x_1$ and $x_1^{(n)} = 1$ for all $n \in \mathbb{N}$ with $n > 0$.

We represent closed forms using poly-exponential expressions [19], where instead of handling piecewise defined functions via disjunctions (as in [35]), we simulate them via Iverson brackets. For a formula ψ over n, its *Iverson bracket* $[\psi] : \mathbb{N} \to \{0, 1\}$ evaluates to 1 iff ψ is satisfied (i.e., $[\psi](e) = 1$ if $\psi[n/e]$ holds and $[\psi](e) = 0$, otherwise). Later, Iverson brackets can be replaced by the constants 0 or 1, as we only use them for formulas ψ that are constantly false or true for large enough values of n, see Sect. 4. *Poly-exponential expressions* are sums of arithmetic terms over the variables $\vec{x}$ and the additional designated variable n, where it is always clear which addend determines the asymptotic growth of the expression when increasing n. This is crucial for our reducibility proof in Sect. 4. Definition 3.4 slightly generalizes the poly-exponential expressions from [19, Def. 9] by allowing arbitrary polynomials over $\vec{x}$ (instead of just linear expressions) as coefficients. In the following, for any set $X \subseteq \mathbb{R}$, any $k \in X$, and $\rhd \in \{\geq, >\}$, let $X_{\rhd k} = \{x \in X \mid x \rhd k\}$.

Definition 3.4 (Poly-Exponential Expressions) Let $\mathcal{C}$ be the set of all finite conjunctions over $\{n = c, n \neq c \mid c \in \mathbb{N}\}$ where n is a designated variable and let $\mathcal{Q}_{\mathcal{S}} = \left\{ \frac{r}{s} \mid r \in \mathcal{S}, s \in \mathcal{S}_{>0} \right\}$ be the quotient field of $\mathcal{S}$. Then the set of all *poly-exponential expressions* with the variables $\vec{x}$ over $\mathcal{S}$ is

$$\mathbb{PE}_{\mathcal{S}}[\vec{x}] = \{ \sum_{j=1}^{\ell} [\psi_j] \cdot \alpha_j \cdot n^{a_j} \cdot b_j^n \mid \ell, a_j \in \mathbb{N},\ \psi_j \in \mathcal{C},\ \alpha_j \in \mathcal{Q}_{\mathcal{S}}[\vec{x}],\ b_j \in \mathcal{S}_{>0} \}.$$

An example for a poly-exponential expression over $\mathbb{Z}$ (with $\mathcal{Q}_{\mathbb{Z}} = \mathbb{Q}$) is

$$[n \neq 0 \wedge n \neq 1] \cdot \left(\tfrac{1}{2} \cdot x_1^2 + \tfrac{3}{4} \cdot x_2 - 1 \right) \cdot n^3 \cdot 3^n + [n = 1] \cdot (x_1 - x_2).$$

The restriction to *tnn*-loops ensures that for the closed form $\vec{q}$ of the update we indeed have $\vec{q} \in \left(\mathbb{PE}_{\mathcal{S}}[\vec{x}]\right)^d$. For example, for arbitrary matrices $A \in \mathbb{R}_{\mathbb{A}}^{d \times d}$, the update $\vec{x} \leftarrow A \cdot \vec{x}$ is known to admit a closed form as in Definition 3.4 with complex b_j's, whereas real numbers suffice for triangular matrices. Moreover, non-negativity is required to ensure $b_j > 0$ (e.g., a non-*tnn* loop with the update $x_1 \leftarrow -x_1$ has the closed form $x_1 \cdot (-1)^n$).

Example 3.5 For $\mathbf{L}_{ex}$ in Fig. 2, the closed form is $\vec{q} \in \left(\mathbb{PE}_{\mathbb{Z}}[\vec{x}]\right)^3$ with

$$\vec{q} = \begin{bmatrix} \frac{4}{3} \cdot x_3^5 \cdot n^3 + \left(-2 \cdot x_3^5 - 2 \cdot x_2 \cdot x_3^3\right) \cdot n^2 + \left(x_2^2 \cdot x_3 + \frac{2}{3} \cdot x_3^5 + 2 \cdot x_2 \cdot x_3^3\right) \cdot n + x_1 \\ -2 \cdot x_3^2 \cdot n + x_2 \\ x_3 \end{bmatrix}.$$

4 Reducing termination of *tnn*-loops to $\mathrm{Th}_\exists(\mathcal{S})$

It is known that the bodies of *tnn*-loops can be linearized [48], i.e., one can reduce termination of a *tnn*-loop $(\varphi, \vec{u})$ to termination of a linear-update *tnn*-loop $(\varphi', \vec{u}')$, where φ' may be *non*-linear. See Sect. 6 for a discussion of linearization and novel results on the linearization procedure. Moreover, [64, 65] showed decidability of termination for certain classes of conjunctive linear-update loops over $\mathbb{R}$, which cover conjunctive linear-update *tnn*-loops. So, by combining the results of [48] and [64, 65], one can conclude that termination of *conjunctive tnn*-loops over $\mathbb{R}$ is decidable.

In contrast, we now present a reduction of termination of *tnn*-loops to $\mathrm{Th}_\exists(\mathcal{S})$ which applies to *tnn*-loops over *any* ring $\mathbb{Z} \leq \mathcal{S} \leq \mathbb{R}$ and which can also handle *disjunctions* in the loop condition. Moreover, our reduction yields tight complexity results on termination of linear loops over $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}_{\mathbb{A}}$, and $\mathbb{R}$, and on termination of linear-update loops over $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$ (see Sect. 7.1).

Similar to [19], our reduction exploits that for *tnn*-loops $(\varphi, \vec{u})$ there is a closed form $\vec{q}$ for $\vec{u}$ with $\vec{q} \in \left(\mathbb{PE}_{\mathcal{S}}[\vec{x}]\right)^d$. However, in [19] we only considered conjunctive linear loops over $\mathbb{Z}$. In contrast, we now analyze loops over $\mathcal{S}$ for any $\mathbb{Z} \le \mathcal{S} \le \mathbb{R}_{\mathbb{A}}$ and allow *non*-linearity and arbitrary propositional formulas in the condition. Thus, the correctness proofs differ substantially from [19].

More precisely, we show that there is a function with the following specification that is computable in polynomial time:

$$\begin{array}{rl} \text{Input}: & \text{a } tnn\text{-loop}(\varphi, \vec{u}) \text{ over } \mathcal{S} \text{ with closed form } \vec{q} \in \left(\mathbb{PE}_{\mathcal{S}}[\vec{x}]\right)^d \\ \text{Result}: & \text{a closed formula } \chi \in \text{Th}_{\exists}(\mathcal{S}) \text{ such that} \\ & \chi \text{ is valid iff } (\varphi, \vec{u}) \text{ does not terminate on } \mathcal{S}^d \end{array} \tag{1}$$

We use the concept of *eventual non-termination*, i.e., the condition of the loop may be violated finitely often, see Definition 2.1. Clearly, $(\varphi, \vec{u})$ is non-terminating iff it is eventually non-terminating [49]. The formula χ in (1) will encode the existence of a witness for eventual non-termination. By the definition of $\vec{q}$, eventual non-termination of $(\varphi, \vec{u})$ on $\mathcal{S}^d$ is equivalent to validity of

$$\exists \vec{x} \in \mathcal{S}^d,\ n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \varphi(\vec{q}). \tag{2}$$

Example 4.1 Continuing Example 3, $\mathbf{L}_{ex}$ is eventually non-terminating over $\mathbb{Z}$ iff there is a corresponding witness $\vec{e} \in \mathbb{Z}^3$, i.e., iff

$$\begin{aligned} & \exists x_1, x_2, x_3 \in \mathbb{Z},\ n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ p > 0 \\ \text{is valid where}\quad & p = (x_1 + x_2^2) + (x_2^2 \cdot x_3 + \tfrac{2}{3} \cdot x_3^5 + 2 \cdot x_2 \cdot x_3^3 - 4 \cdot x_2 \cdot x_3^2) \cdot n \\ & \quad + (-2 \cdot x_3^5 - 2 \cdot x_2 \cdot x_3^3 + 4 \cdot x_3^4) \cdot n^2 + (\tfrac{4}{3} \cdot x_3^5) \cdot n^3. \end{aligned} \tag{3}$$

Let $\vec{q}_{norm}$ be like $\vec{q}$, but each factor $[\psi]$ is replaced by 0 if it contains an equation and by 1, otherwise. The reason is that for large enough n, equations in ψ become false and negated equations become true. So (2) is equivalent to

$$\exists \vec{x} \in \mathcal{S}^d,\ n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \varphi(\vec{q}_{norm}). \tag{4}$$

In this way, we obtain *normalized* poly-exponential expressions.

Definition 4.2 (Normalized $\mathbb{PE}$s) We call $p \in \mathbb{PE}_{\mathcal{S}}[\vec{x}]$ *normalized* if it is in

$$\mathbb{NPE}_{\mathcal{S}}[\vec{x}] = \left\{ \sum\nolimits_{j=1}^{\ell} \alpha_j \cdot n^{a_j} \cdot b_j^n \mid \ell, a_j \in \mathbb{N},\ \alpha_j \in \mathcal{Q}_{\mathcal{S}}[\vec{x}],\ b_j \in \mathcal{S}_{>0} \right\},$$

where w.l.o.g. $(b_i, a_i) \neq (b_j, a_j)$ if $i \neq j$. We define $\mathbb{NPE}_{\mathcal{S}} = \mathbb{NPE}_{\mathcal{S}}[\emptyset]$.

As φ is a propositional formula over $\mathcal{S}[\vec{x}]$-inequations, $\varphi(\vec{q}_{norm})$ is a propositional formula over $\mathbb{NPE}_{\mathcal{S}}[\vec{x}]$-inequations. By (4), we need to check whether $\exists \vec{x} \in \mathcal{S}^d.\ \varphi(\vec{q}_{norm})$ is valid for large enough n. To this end, we will examine the dominant terms in the inequations of $\varphi(\vec{q}_{norm})$.

Definition 4.3 (Asymptotic Domination [38]) A function $g : \mathbb{N} \to \mathbb{R}$ *dominates* a function $f : \mathbb{N} \to \mathbb{R}$ asymptotically ($f \in o(g)$) if for all $m > 0$ there is an $n_0 \in \mathbb{N}$ such that[1] $|f(n)| < m \cdot |g(n)|$ for all $n \in \mathbb{N}_{>n_0}$.

Now we can state the following lemma which generalizes [19, Lemma 24].

Lemma 4.4 *Let $b_1, b_2 \in \mathcal{S}_{>0}$ and $a_1, a_2 \in \mathbb{N}$. If $(b_2, a_2) >_{lex} (b_1, a_1)$, then $n^{a_1} \cdot b_1^n \in o(n^{a_2} \cdot b_2^n)$, where $(b_2, a_2) >_{lex} (b_1, a_1)$ iff $b_2 > b_1$ or $b_2 = b_1 \wedge a_2 > a_1$.*

In the following, let $p \geq 0$ or $p > 0$ occur in $\varphi(\vec{q}_{norm})$. We can order the coefficients of p according to the asymptotic growth of their addends w.r.t. n.

Definition 4.5 (Ordering Coefficients) *Marked coefficients* are of the form $\alpha^{(b,a)}$ where $\alpha \in \mathcal{Q}_{\mathcal{S}}[\vec{x}], b \in \mathcal{S}_{>0}$, and $a \in \mathbb{N}$. We define $\operatorname{unmark}(\alpha^{(b,a)}) = \alpha$ and $\alpha_2^{(b_2,a_2)} \succ_{coef} \alpha_1^{(b_1,a_1)}$ if $(b_2, a_2) >_{lex} (b_1, a_1)$. Let $p = \sum_{j=1}^{\ell} \alpha_j \cdot n^{a_j} \cdot b_j^n \in \mathbb{NPE}_{\mathcal{S}}[\vec{x}]$, where $\alpha_j \neq 0$ for all $1 \leq j \leq \ell$. Then the marked coefficients of p are

$$\operatorname{coefs}(p) = \begin{cases} \{0^{(1,0)}\}, & \text{if } \ell = 0 \\ \{\alpha_j^{(b_j,a_j)} \mid 1 \leq j \leq \ell\}, & \text{otherwise.} \end{cases}$$

Example 4.6 Continuing Example 4.1, $\operatorname{coefs}(p)$ is $\{\alpha_1^{(1,0)}, \alpha_2^{(1,1)}, \alpha_3^{(1,2)}, \alpha_4^{(1,3)}\}$ where

$$\alpha_1 = x_1 + x_2^2 \qquad \alpha_2 = x_2^2 \cdot x_3 + \tfrac{2}{3} \cdot x_3^5 + 2 \cdot x_2 \cdot x_3^3 - 4 \cdot x_2 \cdot x_3^2$$
$$\alpha_3 = -2 \cdot x_3^5 - 2 \cdot x_2 \cdot x_3^3 + 4 \cdot x_3^4 \qquad \alpha_4 = \tfrac{4}{3} \cdot x_3^5$$

Note that $p(\vec{e}) \in \mathbb{NPE}_{\mathcal{S}}$ for any $\vec{e} \in \mathcal{S}^d$, i.e., the only variable in $p(\vec{e})$ is n. Now the $\succ_{coef}$-maximal addend determines the asymptotic growth of $p(\vec{e})$:

$$o(p(\vec{e})) = o(k \cdot n^a \cdot b^n) \quad \text{where } k^{(b,a)} = \max_{\succ_{coef}} \left(\operatorname{coefs}\left(p(\vec{e})\right)\right). \tag{5}$$

Note that (5) would be incorrect for the case $k = 0$ if we replaced $o(p(\vec{e})) = o(k \cdot n^a \cdot b^n)$ with $o(p(\vec{e})) = o(n^a \cdot b^n)$ as $o(0) = \emptyset \neq o(1)$. Obviously, (5) implies

$$\exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \operatorname{sign}\left(p(\vec{e})\right) = \operatorname{sign}(k), \tag{6}$$

where $\operatorname{sign}(0) = 0$, $\operatorname{sign}(k) = 1$ if $k > 0$, and $\operatorname{sign}(k) = -1$ if $k < 0$. This allows us to reduce eventual non-termination to $\operatorname{Th}_{\exists}(\mathcal{S})$ if φ is an atom. In the following, let $\operatorname{coefs}(p) = \{\alpha_1^{(b_1,a_1)}, \ldots, \alpha_\ell^{(b_\ell,a_\ell)}\}$, where $\alpha_i^{(b_i,a_i)} \prec_{coef} \alpha_j^{(b_j,a_j)}$ for all $1 \leq i < j \leq \ell$. Then we define

[1] Our definition is slightly more general than the original definition of [38] (which requires $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$), but both definitions are equivalent if $g(n)$ is positive for large enough n.

$$\begin{aligned} \text{red}\,(p > 0) &= \textstyle\bigvee_{i=1}^{\ell} (\alpha_i > 0 \wedge \bigwedge_{j=i+1}^{\ell} \alpha_j = 0) \\ \text{and} \quad \text{red}\,(p \geq 0) &= \text{red}\,(p > 0) \vee \textstyle\bigwedge_{i=1}^{\ell} \alpha_i = 0. \end{aligned} \tag{7}$$

Lemma 4.7 *Given* $p \in \mathbb{NPE}_{\mathcal{S}}[\vec{x}]$ *and* $\triangleright \in \{\geq, >\}$, *validity of*

$$\exists \vec{x} \in \mathcal{S}^d,\ n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ p \triangleright 0 \tag{8}$$

can be reduced to validity of the closed formula $\exists \vec{x} \in \mathcal{S}^d.\ \text{red}\,(p \triangleright 0)$ *from* $\text{Th}_\exists\,(\mathcal{S})$. *This reduction takes polynomially many steps in the size of* p.

To gain intuition for the formula $\text{red}\,(p \triangleright 0)$, note that by (6), we have $p(\vec{e}) > 0$ for large enough values of n iff the coefficient of the asymptotically fastest growing addend $\alpha(\vec{e}) \cdot n^a \cdot b^n$ that does not vanish (i.e., where $\alpha(\vec{e}) \neq 0$) is *positive*. Similarly, we have $p(\vec{e}) < 0$ for large enough n iff $\alpha(\vec{e}) < 0$. If *all* addends of p vanish when instantiating $\vec{x}$ with $\vec{e}$, then $p(\vec{e}) = 0$. In other words, (8) holds iff there is an $\vec{e} \in \mathcal{S}^d$ such that $\text{unmark}\left(\max_{\succ_{coef}}\left(\text{coefs}\left(p(\vec{e})\right)\right)\right) \triangleright 0$. The formula $\text{red}\,(p \triangleright 0)$ expresses the latter in $\text{Th}_\exists\,(\mathcal{S})$.

Example 4.8 We continue Example 4.6. By the construction in Lemma 4.7, (3) is valid iff $\exists x_1, x_2, x_3 \in \mathbb{Z}.\ \text{red}\,(p > 0)$ is valid, where $\text{red}\,(p > 0)$ is

$$\begin{aligned} &\left(\alpha_1 > 0 \wedge \alpha_2 = 0 \wedge \alpha_3 = 0 \wedge \alpha_4 = 0\right) \vee \left(\alpha_2 > 0 \wedge \alpha_3 = 0 \wedge \alpha_4 = 0\right) \\ \vee\ &\left(\alpha_3 > 0 \wedge \alpha_4 = 0\right) \qquad\qquad\qquad\qquad \vee\ \alpha_4 > 0. \end{aligned}$$

For example, $[x_1/-4, x_2/2, x_3/1]$ satisfies $\alpha_4 > 0$ as $\left(\frac{4}{3} \cdot x_3^5\right)[x_1/-4, x_2/2, x_3/1] > 0$. Thus, $(-4, 2, 1)$ witnesses eventual non-termination of $\mathbf{L}_{ex}$ over $\mathbb{Z}$.

Now we lift our reduction to propositional formulas. To handle disjunctions, the proof of Theorem 4.9 exploits the crucial insight that a *tnn*-loop $(\varphi \vee \varphi', \vec{u})$ terminates iff $(\varphi, \vec{u})$ and $(\varphi', \vec{u})$ terminate, which is not true in general (as, e.g., witnessed by the loop $(x_1 > 0 \vee -x_1 > 0, -x_1)$). In the following, the formula $\text{red}\,(\xi)$ results from ξ by replacing each atom $p \triangleright 0$ in ξ by $\text{red}\,(p \triangleright 0)$.

Theorem 4.9 (Reducing Eventual Non-Termination) *For a propositional formula* ξ *over the atoms* $\{p \triangleright 0 \mid p \in \mathbb{NPE}_{\mathcal{S}}[\vec{x}], \triangleright \in \{\geq, >\}\}$, *validity of*

$$\exists \vec{x} \in \mathcal{S}^d, n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \xi \tag{9}$$

can be reduced to validity of the closed formula $\exists \vec{x} \in \mathcal{S}^d.\ \text{red}\,(\xi) \in \text{Th}_\exists\,(\mathcal{S})$. *This reduction takes polynomially many steps in the size of* ξ.

The time needed to compute the formula (9) is polynomial in the sum of the sizes of all poly-exponential expressions in ξ. So the function (1) is computable in polynomial time w.r.t. the size of its input: $\vec{q}_{norm}$ can clearly be computed in polynomial time from $\vec{q}$ and we can then apply Theorem 4.9 to $\varphi(\vec{q}_{norm})$. Combining Corollary 3.3, (4), and Theorem 4.9 leads to the main result of this section.

Input: a *twn*-loop $(\varphi, \vec{u})$
Result: $\top$ resp. $\bot$ if (non-)termination of $(\varphi, \vec{u})$ on $\mathcal{S}^d$ is proven, ? otherwise
if $(\varphi, \vec{u})$ *is not tnn* **then** $(\varphi, \vec{u}) \leftarrow (\varphi \wedge \varphi(\vec{u}), \vec{u}(\vec{u}))$
$\vec{q} \leftarrow$ closed form for $\vec{u}$
if *(in)validity of* $\exists \vec{x} \in \mathcal{S}^d.\ \mathrm{red}(\varphi(\vec{q}_{norm}))$ *cannot be proven* **then return** ?
if $\exists \vec{x} \in \mathcal{S}^d.\ \mathrm{red}(\varphi(\vec{q}_{norm}))$ *is valid* **then return** $\bot$ **else return** $\top$

Algorithm 1 Checking Termination

Theorem 4.10 (Reducing Termination) *Termination of tnn-loops (resp. twn-loops) on $\mathcal{S}^d$ is reducible to* $\mathrm{Th}_\exists(\mathcal{S})$.

However, if the update contains *non-linear* terms, then its closed form and hence this reduction are not always computable in polynomial space (and thus, also not in polynomial time). Consider the following *tnn*-loop $\mathbf{L}_{non-pspace}$:

$$\textbf{while}\ (\mathrm{true})\ \textbf{do}\big(x_1, x_2, \ldots, x_{d-1}, x_d\big) \leftarrow \big(x_2^d, x_3^d, \ldots, x_d^d, d \cdot x_d\big) \tag{10}$$

The closed form for x_i (i.e., the value of x_i after n loop iterations) is $q_i = d^{(d^{d-i}\cdot(n-d+i))} \cdot x_d^{(d^{d-i})}$ for all $n \geq d$. Thus, the closed form q_1 for x_1 contains constants like $d^{(d^{d-1})}$ whose logarithm grows faster than any polynomial in d. Hence, q_1 cannot be computed in polynomial space.

Instead of computing closed forms directly, one could first linearize the loop (see [48] and Sect. 6) and then compute the closed form for the resulting linear-update loop. However, this approach cannot be computed in polynomial space either, because the linearization of $\mathbf{L}_{non-pspace}$ contains the constant $d^{(d^{d-1})}$ as well (see Example 6.14 in Sect. 6). We refer to Sect. 7 for an analysis of the complexity of deciding termination for *twn*-loops.

Our reduction also works if $\mathcal{S} = \mathbb{R}$, i.e., termination over $\mathbb{R}$ is reducible to $\mathrm{Th}_\exists(\mathbb{R})$, since $\mathbb{R}$ and $\mathbb{R}_\mathbb{A}$ are *elementary equivalent* (i.e., a first-order formula is valid over $\mathbb{R}$ iff it is valid over $\mathbb{R}_\mathbb{A}$, see, e.g., [2]). Thus, we get the following corollary by using that validity of closed formulas from $\mathrm{Th}_\exists(\mathcal{S})$ is decidable for $\mathcal{S} \in \{\mathbb{R}_\mathbb{A}, \mathbb{R}\}$ and semi-decidable for $\mathcal{S} \in \{\mathbb{Z}, \mathbb{Q}\}$ [13, 61].

Corollary 4.11 ((Semi-)*Deciding (Non-)Termination*) *Let $(\varphi, \vec{u})$ be a twn-loop.*

(a) *The loop $(\varphi, \vec{u})$ terminates over $\mathbb{R}_\mathbb{A}$ iff it terminates over $\mathbb{R}$.*
(b) *Termination of $(\varphi, \vec{u})$ on $\mathcal{S}^d$ is decidable if $\mathcal{S} = \mathbb{R}_\mathbb{A}$ or $\mathcal{S} = \mathbb{R}$.*
(c) *Non-termination of $(\varphi, \vec{u})$ on $\mathcal{S}^d$ is semi-decidable if $\mathcal{S} = \mathbb{Z}$ or $\mathcal{S} = \mathbb{Q}$.*

Our technique does not yield witnesses for non-termination, but the formula constructed by Theorem 4.9 describes the set of *all* witnesses for *eventual* non-termination. So this set can be characterized by a formula from $\mathrm{Th}_\exists(\mathcal{S})$ (i.e., it is $\mathrm{Th}_\exists(\mathcal{S})$-*definable*, see Sect. 5.1), while in general the set of witnesses for non-termination cannot be characterized in this way (see [14]).

Lemma 4.12 *Let $\xi = \varphi(\vec{q}_{norm})$. Then $\vec{e} \in \mathcal{S}^d$ witnesses eventual non-termination of $(\varphi, \vec{u})$ on $\mathcal{S}^d$ iff* $\mathrm{red}(\xi)(\vec{e})$ *holds.*

In [28], we show how to compute witnesses for non-termination from witnesses for eventual non-termination of *twn*-loops. Thus, combining Lemma 4.12 with [28] shows that $NT_{(\varphi,\vec{u})}$

is recursively enumerable for *twn*-loops over $\mathbb{Z} \leq \mathcal{S} \leq \mathbb{R}_{\mathbb{A}}$. Algorithm 1 summarizes our technique to check termination of *twn*-loops.

5 Transformation to triangular weakly non-linear form

In this section, we show how to handle loops that are not yet *twn*. To this end, we introduce a transformation of loops via *polynomial automorphisms* in Sect. 5.1 and show that our transformation preserves (non-)termination (Theorem 5.10). In Sect. 5.2, we use results from algebraic geometry to show that the question whether a loop can be transformed into *twn*-form is reducible to validity of $Th_{\exists}(\mathbb{R}_{\mathbb{A}})$-formulas (Theorem 5.19). Moreover, we show that it is decidable whether a *linear* automorphism can transform a loop into a special case of the *twn*-form (Theorem 5.22). Finally, based on the transformation of Sects. 5.1 and 5.2 we generalize our results from Sect. 4 to certain non-*twn* loops in Sect. 5.3.

5.1 Transforming loops

Clearly, the *polynomials* $x_1, \ldots, x_d$ are *generators* of the $\mathcal{S}$-algebra $\mathcal{S}[\vec{x}]$, i.e., every polynomial from $\mathcal{S}[\vec{x}]$ can be obtained from $x_1, \ldots, x_d$ and the operations of the algebra (i.e., addition and multiplication). So far, we have implicitly chosen a special "representation" of the loop based on the generators $x_1, \ldots, x_d$.

We now change this representation, i.e., we use d different polynomials which are also generators of $\mathcal{S}[\vec{x}]$. Then the loop has to be modified accordingly to adapt it to this new representation. This modification does not affect the loop's termination behavior, but it may transform a non-*twn*-loop into *twn*-form. This change of representation is described by $\mathcal{S}$-*automorphisms* of $\mathcal{S}[\vec{x}]$.

Definition 5.1 ($\mathcal{S}$-Endomorphisms) A mapping $\eta : \mathcal{S}[\vec{x}] \to \mathcal{S}[\vec{x}]$ is an $\mathcal{S}$-*endomorphism* of $\mathcal{S}[\vec{x}]$ if it is $\mathcal{S}$-linear and multiplicative, i.e., for all $c, c' \in \mathcal{S}$ and all $p, p' \in \mathcal{S}[\vec{x}]$ we have $\eta(c \cdot p + c' \cdot p') = c \cdot \eta(p) + c' \cdot \eta(p')$, $\eta(1) = 1$, and $\eta(p \cdot p') = \eta(p) \cdot \eta(p')$. We denote the set of all $\mathcal{S}$-endomorphisms of $\mathcal{S}[\vec{x}]$ by $\mathrm{End}_{\mathcal{S}}\big(\mathcal{S}[\vec{x}]\big)$. The set $\mathrm{Aut}_{\mathcal{S}}\big(\mathcal{S}[\vec{x}]\big)$ of $\mathcal{S}$-*automorphisms* of $\mathcal{S}[\vec{x}]$ consists of those $\eta \in \mathrm{End}_{\mathcal{S}}\big(\mathcal{S}[\vec{x}]\big)$ which are *invertible*, i.e., there exists an $\eta^{-1} \in \mathrm{End}_{\mathcal{S}}\big(\mathcal{S}[\vec{x}]\big)$ with $\eta \circ \eta^{-1} = \eta^{-1} \circ \eta = id_{\mathcal{S}[\vec{x}]}$, where $id_{\mathcal{S}[\vec{x}]}$ is the identity function on $\mathcal{S}[\vec{x}]$. $\mathrm{Aut}_{\mathcal{S}}\big(\mathcal{S}[\vec{x}]\big)$ is a group under function composition $\circ$ with identity $id_{\mathcal{S}[\vec{x}]}$.

Example 5.2 Let $\eta \in \mathrm{End}_{\mathcal{S}}\big(\mathcal{S}[x_1, x_2]\big)$ with $\eta(x_1) = x_2$ and $\eta(x_2) = x_1 - x_2^2$. Then $\eta \in \mathrm{Aut}_{\mathcal{S}}\big(\mathcal{S}[x_1, x_2]\big)$, where $\eta^{-1}(x_1) = x_1^2 + x_2$ and $\eta^{-1}(x_2) = x_1$.

As $\mathcal{S}[\vec{x}]$ is free on the generators $\vec{x}$, an endomorphism $\eta \in \mathrm{End}_{\mathcal{S}}\big(\mathcal{S}[\vec{x}]\big)$ is uniquely determined by the images of the variables, i.e., by $\eta(x_1), \ldots, \eta(x_d)$. Hence, we have a one-to-one correspondence between elements of $\big(\mathcal{S}[\vec{x}]\big)^d$ and $\mathrm{End}_{\mathcal{S}}\big(\mathcal{S}[\vec{x}]\big)$. In particular, every tuple $\vec{u} = (u_1, \ldots, u_d) \in \big(\mathcal{S}[\vec{x}]\big)^d$ corresponds to the unique endomorphism $\widetilde{u} \in \mathrm{End}_{\mathcal{S}}\big(\mathcal{S}[\vec{x}]\big)$ which is defined as follows:

$$\widetilde{u}(x_i) = u_i \qquad \text{for all } 1 \le i \le d$$

So for any $p \in \mathcal{S}[\vec{x}]$ we have $\widetilde{u}(p) = p(\vec{u})$. Thus, the update of a loop induces an endomorphism which operates on polynomials.

Example 5.3 Consider the loop $\mathbf{L}_{aut} = (\varphi, \vec{u})$ where $\vec{u} = (u_1, u_2)$:

$$\textbf{while}\ (x_2^3 + x_1 - x_2^2 > 0)\ \textbf{do} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} ((-x_2^2 + x_1)^2 + x_2)^2 - 2 \cdot x_2^2 + 2 \cdot x_1 \\ (-x_2^2 + x_1)^2 + x_2 \end{bmatrix} \tag{11}$$

Then $\vec{u}$ induces the endomorphism $\widetilde{u}$ with $\widetilde{u}(x_1) = u_1$ and $\widetilde{u}(x_2) = u_2$. So we have $\widetilde{u}(2 \cdot x_1 + x_2^3) = (2 \cdot x_1 + x_2^3)(\vec{u}) = 2 \cdot u_1 + u_2^3$.

Therefore, for a tuple of numbers like $\vec{e} = (5, 2)$, the induced endomorphism $\widetilde{e}$ is $\widetilde{e}(x_1) = 5$ and $\widetilde{e}(x_2) = 2$. Thus, we have $\widetilde{e}(x_2^3 + x_1 - x_2^2) = (x_2^3 + x_1 - x_2^2)(5, 2) = 2^3 + 5 - 2^2 = 9$.

We extend the application of endomorphisms $\eta : \mathcal{S}[\vec{x}] \to \mathcal{S}[\vec{x}]$ to vectors of polynomials $\vec{u} = (u_1, \dots, u_d)$ by defining $\eta(\vec{u}) = (\eta(u_1), \dots, \eta(u_d))$ and to formulas $\varphi \in \mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$ by defining $\eta(\varphi) = \varphi(\eta(\vec{x}))$, i.e., $\eta(\varphi)$ results from φ by applying η to all polynomials that occur in φ. This allows us to transform $(\varphi, \vec{u})$ into a new loop $\mathrm{Tr}_\eta(\varphi, \vec{u})$ using any automorphism $\eta \in \mathrm{Aut}_\mathcal{S}\big(\mathcal{S}[\vec{x}]\big)$.

Definition 5.4 (Tr) For $\eta \in \mathrm{Aut}_\mathcal{S}\big(\mathcal{S}[\vec{x}]\big)$, we define $\mathrm{Tr}_\eta(\varphi, \vec{u}) = (\varphi', \vec{u}\,')$ where

$$\varphi' = \eta^{-1}(\varphi) \qquad \text{and} \qquad \vec{u}\,' = (\eta^{-1} \circ \widetilde{u} \circ \eta)(\vec{x}).$$

In other words, we have $\vec{u}\,' = (\eta(\vec{x}))\,(\vec{u})\,(\eta^{-1}(\vec{x}))$ since $(\eta^{-1} \circ \widetilde{u} \circ \eta)(\vec{x}) = \eta^{-1}(\eta(\vec{x})[\vec{x}/\vec{u}]) = \eta(\vec{x})[\vec{x}/\vec{u}][\vec{x}/\eta^{-1}(\vec{x})] = (\eta(\vec{x}))(\vec{u})(\eta^{-1}(\vec{x}))$.

Example 5.5 We transform the loop $\mathbf{L}_{aut}$ in (11) with the automorphism η from Example 5.2. We obtain $\mathrm{Tr}_\eta(\varphi, \vec{u}) = (\varphi', \vec{u}\,')$ where

$$\begin{aligned} \varphi' &= \eta^{-1}(\varphi) = ((\eta^{-1}(x_2))^3 + \eta^{-1}(x_1) - (\eta^{-1}(x_2))^2 > 0) \\ &= (x_1^3 + x_1^2 + x_2 - x_1^2 > 0) = (x_1^3 + x_2 > 0) \text{ and} \\ \vec{u}\,' &= ((\eta^{-1} \circ \widetilde{u} \circ \eta)(x_1), (\eta^{-1} \circ \widetilde{u} \circ \eta)(x_2)) = (\eta^{-1}(\widetilde{u}(x_2)), \eta^{-1}(\widetilde{u}(x_1 - x_2^2))) \\ &= (\eta^{-1}(u_2), \eta^{-1}(u_1 - u_2^2)) = (x_1 + x_2^2, 2 \cdot x_2). \end{aligned}$$

So the resulting transformed loop is $(x_1^3 + x_2 > 0,\ (x_1 + x_2^2, 2 \cdot x_2))$. Note that while the original loop $(\varphi, \vec{u})$ is neither triangular nor weakly non-linear, the resulting transformed loop is *twn*. Also note that we used a *non-linear* automorphism with $\eta(x_2) = x_1 - x_2^2$ for the transformation.

While the above example shows that our transformation can indeed transform non-*twn*-loops into *twn*-loops, it remains to prove that this transformation preserves (non-) termination. Then we can use our techniques for termination analysis of *twn*-loops for *twn-transformable*-loops as well, i.e., for all loops $(\varphi, \vec{u})$ where $\mathrm{Tr}_\eta(\varphi, \vec{u})$ is *twn* for some automorphism η. (The question how to find such automorphisms will be addressed in Sect. 5.2.)

As a first step, by Lemma 5.6, our transformation is "compatible" with the operation $\circ$ of the group $\mathrm{Aut}_{\mathcal{S}}(\mathcal{S}[\vec{x}])$, i.e., it is an *action*.

Lemma 5.6 $\mathrm{Aut}_{\mathcal{S}}(\mathcal{S}[\vec{x}])$ *acts via* Tr *on loops, i.e., for* $\eta_1, \eta_2 \in \mathrm{Aut}_{\mathcal{S}}(\mathcal{S}[\vec{x}])$, *we have* $\mathrm{Tr}_{id_{\mathcal{S}[\vec{x}]}}(\varphi, \vec{u}) = (\varphi, \vec{u})$ *and* $\mathrm{Tr}_{\eta_1 \circ \eta_2}(\varphi, \vec{u}) = Tr_{\eta_2}(\mathrm{Tr}_{\eta_1}(\varphi, \vec{u}))$.

The following lemma will enable us to generalize our results on witnesses for (eventual) non-termination to loops which can be transformed into *twn*-form.

Lemma 5.7 *Let* $\mathrm{Tr}_{\eta}(\varphi, \vec{u}) = (\varphi', \vec{u}\,')$ *and let* $\hat{\eta} : \mathcal{S}^d \to \mathcal{S}^d$ *map* $\vec{e}$ *to* $\hat{\eta}(\vec{e}) = \tilde{e}(\eta(\vec{x})) = (\eta(\vec{x}))(\vec{e})$. *Then* $\varphi(\vec{u}^n(\vec{e})) = \varphi'((\vec{u}\,')^n(\hat{\eta}(\vec{e})))$ *for all* $\vec{e} \in \mathcal{S}^d$ *and* $n \in \mathbb{N}$.

Lemma 5.7 yields the following corollary which shows that $\eta(\vec{x})$ transforms witnesses for (eventual) non-termination of $(\varphi, \vec{u})$ into witnesses for $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$.

Corollary 5.8 *If* $\vec{e}$ *witnesses* (*eventual*) *non-termination of* $(\varphi, \vec{u})$, *then* $\hat{\eta}(\vec{e})$ *witnesses* (*eventual*) *non-termination of* $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$.

Example 5.9 For the tuple $\vec{e} = (5, 2)$ from Example 5.3 and the automorphism η from Example 5.2 with $\eta(x_1) = x_2$ and $\eta(x_2) = x_1 - x_2^2$, we obtain

$$\hat{\eta}(\vec{e}) = (\eta(x_1), \eta(x_2))\,(\vec{e}) = (2, 5 - 2^2) = (2, 1).$$

As $\vec{e} = (5, 2)$ witnesses non-termination of the loop $\mathbf{L}_{aut} = (\varphi, \vec{u})$ in (11), $\hat{\eta}(\vec{e}) = (2, 1)$ witnesses non-termination of $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$ due to Corollary 5.8.

Finally, Theorem 5.10 states that transforming loops preserves (non-)termination.

Theorem 5.10 (Tr Preserves Termination) *If* $\eta \in \mathrm{Aut}_{\mathcal{S}}(\mathcal{S}[\vec{x}])$, *then* $\hat{\eta}$ *is a bijection between the respective sets of witnesses for* (*eventual*) *non-termination, i.e., for* $\mathrm{Tr}_{\eta}(\varphi, \vec{u}) = (\varphi', \vec{u}\,')$ *we have* $\vec{e} \in (E)NT_{(\varphi,\vec{u})}$ *iff* $\hat{\eta}(\vec{e}) \in (E)NT_{(\varphi',\vec{u}\,')}$. *Therefore,* $(\varphi, \vec{u})$ *terminates iff* $\mathrm{Tr}_{\eta}(\varphi, \vec{u}) = (\varphi', \vec{u}\,')$ *terminates.*

Up to now, we only transformed a loop $(\varphi, \vec{u})$ on $\mathcal{S}^d$ using elements of $\mathrm{Aut}_{\mathcal{S}}(\mathcal{S}[\vec{x}])$. To see that this is a limitation, consider a linear-update loop where $\vec{u} = A \cdot \vec{x}$ and A only has real eigenvalues. In Sect. 7.1 we will show that these loops can always be transformed into *twn*-form and a suitable automorphism η can be obtained by computing the Jordan normal form of A. This automorphism η is only an element of $\mathrm{Aut}_{\mathcal{S}}(\mathcal{S}[\vec{x}])$ if the eigenvalues of A are from $\mathcal{S}$. So if $\mathcal{S} = \mathbb{Z}$, then this transformation is only applicable if all eigenvalues of A are integers.

However, we can also transform $(\varphi, \vec{u})$ into the loop $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$ on $\mathbb{R}_{\mathbb{A}}^d$ using an automorphism $\eta \in \mathrm{Aut}_{\mathbb{R}_{\mathbb{A}}}(\mathbb{R}_{\mathbb{A}}[\vec{x}])$. Nevertheless, our goal remains to prove termination on $\mathcal{S}^d$ instead of $\mathbb{R}_{\mathbb{A}}^d$, which is not equivalent in general. Thus, in Sect. 5.3 we will show how to analyze termination of loops on certain subsets F of $\mathbb{R}_{\mathbb{A}}^d$. This allows us to analyze termination of $(\varphi, \vec{u})$ on $\mathcal{S}^d$ by checking termination of $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$ on the subset $\hat{\eta}(\mathcal{S}^d) \subseteq \mathbb{R}_{\mathbb{A}}^d$ instead.

By our definition of loops over a ring $\mathcal{S}$, we have $\vec{u}(\vec{e}) \in \mathcal{S}^d$ for all $\vec{e} \in \mathcal{S}^d$, i.e., $\mathcal{S}^d$ is $\vec{u}$*-invariant*. This property is preserved by our transformation.

Definition 5.11 ($\vec{u}$-Invariance) Let $(\varphi, \vec{u})$ be a loop on $\mathcal{S}^d$ and let $F \subseteq \mathcal{S}^d$. We call F $\vec{u}$*-invariant* or *update-invariant* if for all $\vec{e} \in F$ we have $\vec{u}(\vec{e}) \in F$.

Lemma 5.12 *Let* $(\varphi, \vec{u})$ *be a loop on* $\mathcal{S}^d$, $F \subseteq \mathcal{S}^d$ *be* $\vec{u}$*-invariant,* $\eta \in \mathrm{Aut}_{\mathbb{R}_{\mathbb{A}}}\left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)$*, and let* $\mathrm{Tr}_\eta(\varphi, \vec{u}) = (\varphi', \vec{u}\,')$*. Then* $\hat{\eta}(F)$ *is* $\vec{u}\,'$*-invariant.*

Our goal is to reduce termination to a $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$-formula. Clearly, *termination on F* cannot be encoded with such a formula if F cannot be defined via $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$. Thus, we require that F is $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$*-definable*, i.e., that there is a $\psi \in \mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$ with free variables $\vec{x}$ such that we have $\vec{e} \in F$ iff $\psi(\vec{e})$ is valid. Then we also say that ψ *defines F*. An example for a $\mathrm{Th}_\exists(\mathbb{Z}, \mathbb{R}_{\mathbb{A}})$-definable set is $\{(a, 0, a) \mid a \in \mathbb{Z}\}$, which is characterized by the formula $\exists a \in \mathbb{Z}.\ x_1 = a \wedge x_2 = 0 \wedge x_3 = a$.

To analyze termination of $(\varphi, \vec{u})$ on $\mathcal{S}^d$, we can analyze termination of $\mathrm{Tr}_\eta(\varphi, \vec{u})$ on $\hat{\eta}(\mathcal{S}^d) \subseteq \mathbb{R}_{\mathbb{A}}^d$ instead. The reason is that $\vec{e} \in \mathcal{S}^d$ is a witness for (eventual) non-termination of $(\varphi, \vec{u})$ iff $\hat{\eta}(\vec{e})$ is a witness for $\mathrm{Tr}_\eta(\varphi, \vec{u})$ due to Corollary 5.8, i.e., $\mathcal{S}^d$ contains a witness for (eventual) non-termination of $(\varphi, \vec{u})$ iff $\hat{\eta}(\mathcal{S}^d)$ contains a witness for $\mathrm{Tr}_\eta(\varphi, \vec{u})$. While $\mathcal{S}^d$ is clearly $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$-definable, Lemma 5.13 shows that $\hat{\eta}(\mathcal{S}^d)$ is $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$-definable, too. More precisely, $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$-definability is preserved by polynomial endomorphisms.

Lemma 5.13 *Let* $\mathbb{Z} \leq \mathcal{S} \leq \mathbb{R}_{\mathbb{A}}$ *and let* $\eta \in \mathrm{End}_{\mathbb{R}_{\mathbb{A}}}\left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)$*. If* $F \subseteq \mathbb{R}_{\mathbb{A}}^d$ *is* $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$*-definable then so is* $\hat{\eta}(F)$*.*

Example 5.14 The set $\mathbb{Z}^2$ is $\mathrm{Th}_\exists(\mathbb{Z}, \mathbb{R}_{\mathbb{A}})$-definable as we have $(x_1, x_2) \in \mathbb{Z}^2$ iff

$$\exists a, b \in \mathbb{Z}.\ x_1 = a \wedge x_2 = b.$$

Let $\eta \in \mathrm{End}_{\mathbb{R}_{\mathbb{A}}}\left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)$ with $\eta(x_1) = \frac{1}{2} \cdot x_1^2 + x_2^2$ and $\eta(x_2) = x_2^2$.

Then $\hat{\eta}(\mathbb{Z}^2)$ is also $\mathrm{Th}_\exists(\mathbb{Z}, \mathbb{R}_{\mathbb{A}})$-definable because for $x_1, x_2 \in \mathbb{R}_{\mathbb{A}}$ we have $(x_1, x_2) \in \eta(\mathbb{Z}^2)$ iff

$$\exists y_1, y_2 \in \mathbb{R}_{\mathbb{A}},\ a, b \in \mathbb{Z}.\ \ y_1 = a \wedge y_2 = b \ \wedge\ x_1 = \frac{1}{2} \cdot y_1^2 + y_2^2 \wedge x_2 = y_2^2.$$

Theorem 5.15 shows that instead of regarding *solvable loops* [56], w.l.o.g. we can restrict ourselves to *twn*-loops. The reason is that every solvable loop with real eigenvalues can be transformed into a *twn*-loop by a *linear* automorphism η, i.e., the degree $\deg(\eta)$ of η is 1, where $\deg(\eta) = \max\{\deg(\eta(x_i)) \mid 1 \leq i \leq d\}$.

Theorem 5.15 (*twn*-Transformability of Solvable Loops) *Let* $(\varphi, \vec{u})$ *be a solvable loop with real eigenvalues. Then one can compute a linear automorphism* $\eta \in \mathrm{Aut}_{\mathbb{R}_{\mathbb{A}}}\left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)$ *such that* $\mathrm{Tr}_\eta(\varphi, \vec{u})$ *is twn.*

We recapitulate our most important results on Tr in the following corollary. Here, we generalize the result of Theorem 5.10 to the setting where we consider termination on some update-invariant and $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$-definable set.

Corollary 5.16 (Properties of Tr) *Let* $(\varphi, \vec{u})$ *be a loop,* $\eta \in \mathrm{Aut}_{\mathbb{R}_\mathbb{A}}(\mathbb{R}_\mathbb{A}[\vec{x}])$, $\mathrm{Tr}_\eta(\varphi, \vec{u}) = (\varphi', \vec{u}\,')$, *and* $F \subseteq \mathcal{S}^d$ *be* $\vec{u}$*-invariant and* $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$*-definable.*

(a) $\widehat{\eta}(F) \subseteq \mathbb{R}_\mathbb{A}^d$ *is* $\vec{u}\,'$*-invariant and* $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$*-definable.*
(b) $(\varphi, \vec{u})$ *terminates on* F *iff* $(\varphi', \vec{u}\,')$ *terminates on* $\widehat{\eta}(F)$.
(c) $\vec{e} \in F$ *witnesses (eventual) non-termination of* $(\varphi, \vec{u})$ *iff* $\widehat{\eta}(\vec{e}) \in \widehat{\eta}(F)$ *witnesses (eventual) non-termination of* $(\varphi', \vec{u}\,')$.

5.2 Finding automorphisms to transform loops into *twn*-form

The goal of Tr_η from Sect. 5.1 is to transform $(\varphi, \vec{u})$ into *twn*-form such that our techniques from Sect. 4 can be used to decide termination. So the two remaining challenges are to find a suitable automorphism $\eta \in \mathrm{Aut}_{\mathbb{R}_\mathbb{A}}(\mathbb{R}_\mathbb{A}[\vec{x}])$ such that $\mathrm{Tr}_\eta(\varphi, \vec{u})$ is *twn*, and to adapt our techniques from Sect. 4 such that they can be applied to *twn*-loops where one only wants to show termination on an update-invariant $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$-definable subset. We discuss the latter challenge in Sect. 5.3. In this section, we present two techniques to check the existence of automorphisms for the transformation into *twn*-form *constructively*, i.e., these techniques can also be used to compute such automorphisms.

The search for suitable automorphisms is closely related to the question if a polynomial automorphism can be conjugated into a "de Jonquiéres"-automorphism, a difficult question from algebraic geometry [18]. So future advances in this field may help to improve the results of the current section.

The first technique (Theorem 5.19) reduces the search for a suitable automorphism *of bounded degree* to $\mathrm{Th}_\exists(\mathbb{R}_\mathbb{A})$. For any automorphism, the degree of its inverse is bounded in terms of the length d of $\vec{x}$.

Theorem 5.17 (Degree of Inverse [18, Corollary 2.3.4]) *Let* $\eta \in \mathrm{Aut}_{\mathbb{R}_\mathbb{A}}(\mathbb{R}_\mathbb{A}[\vec{x}])$. *Then we have* $\deg(\eta^{-1}) \leq (\deg(\eta))^{d-1}$.

By Theorem 5.17, checking if an endomorphism is an automorphism can be reduced to $\mathrm{Th}_\exists(\mathbb{R}_\mathbb{A})$. To do so, one encodes the existence of coefficients for the polynomials $\eta^{-1}(x_1), \ldots, \eta^{-1}(x_d)$, which all have at most degree $(\deg(\eta))^{d-1}$.

Lemma 5.18 *Let* $\eta \in \mathrm{End}_{\mathbb{R}_\mathbb{A}}(\mathbb{R}_\mathbb{A}[\vec{x}])$. *Then the question if* $\eta \in \mathrm{Aut}_{\mathbb{R}_\mathbb{A}}(\mathbb{R}_\mathbb{A}[\vec{x}])$ *holds is reducible to* $\mathrm{Th}_\exists(\mathbb{R}_\mathbb{A})$.

Based on Lemma 5.18, we now present our first technique to find an automorphism η that transforms a loop into *twn*-form.

Theorem 5.19 (Tr With Automorphisms of Bounded Degree) *For any* $\delta \geq 0$, *the question whether there exists an* $\eta \in \mathrm{Aut}_{\mathbb{R}_\mathbb{A}}(\mathbb{R}_\mathbb{A}[\vec{x}])$ *with* $\deg(\eta) \leq \delta$ *such that* $\mathrm{Tr}_\eta(\varphi, \vec{u})$ *is twn is reducible to* $\mathrm{Th}_\exists(\mathbb{R}_\mathbb{A})$.

$$\textbf{while}\ (4 \cdot x_2{}^2 + x_1 + x_2 + x_3 > 0)\ \textbf{do}\ (x_1, x_2, x_3) \leftarrow (u_1, u_2, u_3)$$

$$\begin{aligned} \text{with } u_1 &= x_1 + 8 \cdot x_1 \cdot x_2{}^2 + 16 \cdot x_2{}^3 + 16 \cdot x_2{}^2 \cdot x_3 \\ u_2 &= x_2 - x_1{}^2 - 4 \cdot x_1 \cdot x_2 - 4 \cdot x_1 \cdot x_3 - 4 \cdot x_2{}^2 - 8 \cdot x_2 \cdot x_3 - 4 \cdot x_3{}^2 \\ u_3 &= x_3 - 4 \cdot x_1 \cdot x_2{}^2 - 8 \cdot x_2{}^3 - 8 \cdot x_2{}^2 \cdot x_3 + x_1{}^2 + 4 \cdot x_1 \cdot x_2 + \\ &\quad 4 \cdot x_1 \cdot x_3 + 4 \cdot x_2{}^2 + 8 \cdot x_2 \cdot x_3 + 4 \cdot x_3{}^2 \end{aligned}$$

Fig. 3 Loop $\mathbf{L}_{non-twn}$

So if the degree of η is bounded a priori, it is decidable whether there exists an $\eta \in \mathrm{Aut}_{\mathbb{R}_{\mathbb{A}}}\left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)$ such that $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$ is *twn* since $\mathrm{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$ is decidable.

We call a loop *twn -transformable* if there is an $\eta \in \mathrm{Aut}_{\mathbb{R}_{\mathbb{A}}}\left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)$ such that $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$ is *twn*. By Theorem 5.19, *twn*-transformability is *semi-decidable* since one can increment δ until a suitable automorphism is found. So in other words, any loop which is transformable to a *twn*-loop can be transformed via Theorem 5.19.

We call our transformation Tr *complete* for a class of loops if *every* loop from this class is *twn*-transformable. For such classes, a suitable automorphism is *computable* by Theorem 5.19. Together with Theorem 5.15, we get Corollary 5.20.

Corollary 5.20 Tr *is complete for solvable loops with real eigenvalues.*

Note that for solvable loops $(\varphi, \vec{u})$, instead of computing η using Theorem 5.19, the proof of Theorem 5.15 yields a more efficient way to compute a linear automorphism η such that $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$ is *twn*. For this, one computes the Jordan normal form of each A_i (see Definition 2.3), which is possible in polynomial time (see, e.g., [23, 54]).

Our second technique to find suitable automorphisms for our transformation is restricted to *linear* automorphisms. In this case, it is decidable whether a loop can be transformed into a *twn*-loop $(\varphi', \vec{u}')$ where the monomial for x_i has the coefficient 1 in each u_i'. The decision procedure checks if a certain Jacobian matrix is *strongly nilpotent*, i.e., it is not based on a reduction to $\mathrm{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$.

Definition 5.21 (Strong Nilpotence) Let $J \in \left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)^{d \times d}$ be a matrix of polynomials. For all $1 \le i \le d$, let $\vec{y}^{(i)}$ be a vector of fresh variables. J is *strongly nilpotent* if $\prod_{i=1}^{d} J[\vec{x}/\vec{y}^{(i)}] = 0^{d \times d}$, where $0^{d \times d}$ is the zero matrix.

Our second technique is formulated in the following theorem which follows from an existing result in linear algebra [17, Thm. 1.6.].

Theorem 5.22 (Tr With Linear Automorphisms [17]) *Let* $(\varphi, \vec{u})$ *be a loop. The Jacobian matrix* $\left(\frac{\partial(u_i - x_i)}{\partial x_j}\right)_{1 \le i,j \le d} \in \left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)^{d \times d}$ *is strongly nilpotent iff there exists a linear automorphism* $\eta \in \mathrm{Aut}_{\mathbb{R}_{\mathbb{A}}}\left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)$ *with*

$$\mathrm{Tr}_{\eta}(\varphi, \vec{u}) = (\varphi', (x_1 + p_1, \ldots, x_d + p_d)) \tag{12}$$

and $p_i \in \mathbb{R}_{\mathbb{A}}[x_{i+1}, \ldots, x_d]$ *for all* $1 \le i \le d$. *Thus,* $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$ *is twn.*

As strong nilpotence of the Jacobian matrix is clearly decidable, Theorem 5.22 gives rise to a decision procedure for the existence of a linear automorphism that transforms $(\varphi, \vec{u})$ to the form (12).

Example 5.23 The loop $\mathbf{L}_{non-twn}$ on $\mathcal{S}^3$ in Fig. 3 shows how our results enlarge the class of loops where termination is reducible to $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$. This loop is clearly *not* in *twn*-form. To transform it, we use Theorem 5.22. The Jacobian matrix J of $(u_1 - x_1, u_2 - x_2, u_3 - x_3)$ is:

$$\begin{bmatrix} 8 \cdot x_2^2 & 16 \cdot x_1 \cdot x_2 + 48 \cdot x_2^2 + 32 \cdot x_2 \cdot x_3 & 16 \cdot x_2^2 \\ -2 \cdot x_1 - 4 \cdot x_2 - 4 \cdot x_3 & -4 \cdot x_1 - 8 \cdot x_2 - 8 \cdot x_3 & -4 \cdot x_1 - 8 \cdot x_2 - 8 \cdot x_3 \\ -4 \cdot x_2^2 + 2 \cdot x_1 + 4 \cdot x_2 + 4 \cdot x_3 & -8 \cdot x_1 \cdot x_2 - 24 \cdot x_2^2 - 16 \cdot x_2 \cdot x_3 + 4 \cdot x_1 + 8 \cdot x_2 + 8 \cdot x_3 & -8 \cdot x_2^2 + 4 \cdot x_1 + 8 \cdot x_2 + 8 \cdot x_3 \end{bmatrix}$$

One easily checks that J is strongly nilpotent. Thus, by Theorem 5.22 the loop can be transformed into *twn*-form by a linear automorphism. Indeed, consider the linear automorphism $\eta \in \mathrm{Aut}_{\mathbb{R}_\mathbb{A}}\left(\mathbb{R}_\mathbb{A}[\vec{x}]\right)$ induced by the matrix $M = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 2 & 0 \\ 1 & 2 & 2 \end{bmatrix}$, i.e.,

$$\begin{array}{llll} \eta: & x_1 \mapsto x_1 + x_2 + x_3, & x_2 \mapsto 2 \cdot x_2, & x_3 \mapsto x_1 + 2 \cdot x_2 + 2 \cdot x_3 \text{ and} \\ \eta^{-1}: & x_1 \mapsto 2 \cdot x_1 - x_3, & x_2 \mapsto \frac{1}{2} \cdot x_2, & x_3 \mapsto -x_1 - \frac{1}{2} \cdot x_2 + x_3. \end{array}$$

If we transform $\mathbf{L}_{non-twn}$ with η, we obtain the *twn*-loop $\mathbf{L}_{ex}$ in Fig. 2. If $\mathcal{S} = \mathbb{R}_\mathbb{A}$, then $\mathbf{L}_{ex}$ terminates on $\mathbb{R}_\mathbb{A}^3$ iff $\mathbf{L}_{non-twn}$ terminates on $\mathbb{R}_\mathbb{A}^3$ by Theorem 5.10. Thus, as seen in Example 4.8, $\mathbf{L}_{non-twn}$ does not terminate on $\mathbb{R}_\mathbb{A}^3$. Now assume $\mathcal{S} = \mathbb{Z}$, i.e., we analyze termination of $\mathbf{L}_{non-twn}$ on $\mathbb{Z}^3$ instead of $\mathbb{R}_\mathbb{A}^3$. Note that $\hat{\eta}$ maps $\mathbb{Z}^3$ to the set of all $\mathbb{Z}$-linear combinations of columns of M, i.e.,

$$\hat{\eta}(\mathbb{Z}^3) = \{a \cdot (1,0,1) + b \cdot (1,2,2) + c \cdot (1,0,2) \mid a,b,c \in \mathbb{Z}\}.$$

By Corollary 5.16, $\mathbf{L}_{ex}$ terminates on $\hat{\eta}(\mathbb{Z}^3)$ iff $\mathbf{L}_{non-twn}$ terminates on $\mathbb{Z}^3$. Moreover, $\hat{\eta}(\mathbb{Z}^3)$ is $\mathrm{Th}_\exists(\mathbb{Z}, \mathbb{R}_\mathbb{A})$-definable: We have $(x_1, x_2, x_3) \in \hat{\eta}(\mathbb{Z}^3)$ iff

$$\exists a,b,c \in \mathbb{Z}.\ x_1 = a \cdot 1 + b \cdot 1 + c \cdot 1 \wedge x_2 = b \cdot 2 \wedge x_3 = a \cdot 1 + b \cdot 2 + c \cdot 2. \tag{13}$$

We will discuss how to analyze termination of *tnn*-loops like $\mathbf{L}_{ex}$ on update-invariant and $\mathrm{Th}_\exists(\mathbb{Z}, \mathbb{R}_\mathbb{A})$-definable sets like $\hat{\eta}(\mathbb{Z}^3)$ in Sect. 5.3.

To summarize, if a loop is *twn*-transformable, then we can *always* find a suitable automorphism via Theorem 5.19. So whenever Theorem 5.22 is applicable, a suitable linear automorphism can also be obtained by Theorem 5.19. Hence, our first technique from Theorem 5.19 subsumes our second one from Theorem 5.22. However, while Theorem 5.19 is *always* applicable, Theorem 5.22 is *easier* to apply. The reason is that for Theorem 5.19 one has to check validity of a possibly *non-linear* formula over the reals, where the degree of the occurring polynomials depends on δ and the update $\vec{u}$ of the loop, and the number of variables can be exponential in d, see Theorem 5.17 and Lemma 5.18. So even when searching for a linear automorphism, one may obtain a non-linear formula if the loop is non-linear. In contrast, Theorem 5.22 only requires linear algebra. So it is preferable to first check whether the loop can be transformed into a *twn*-loop $(\varphi', (x_1 + p_1, \ldots, x_d + p_d))$ with $x_i \notin \mathcal{V}(p_i)$ via a linear automorphism. This check is *decidable* by Theorem 5.22.

Input: a *twn*-transformable-loop $(\varphi, \vec{u})$ and $\psi_F \in \mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$
Result: $\top$ resp. $\bot$ if (non-)termination of $(\varphi, \vec{u})$ on F is proven, ? otherwise
$(\varphi, \vec{u}) \leftarrow \mathrm{Tr}_\eta(\varphi, \vec{u})$, $\psi_F \leftarrow \psi_{\hat{\eta}(F)}$, such that $(\varphi, \vec{u})$ becomes *twn*
if $(\varphi, \vec{u})$ *is not tnn* **then** $(\varphi, \vec{u}) \leftarrow (\varphi \wedge \varphi(\vec{u}), \vec{u}(\vec{u}))$
$\vec{q} \leftarrow$ closed form for $\vec{u}$
if *(in)validity of* $\exists \vec{x} \in \mathbb{R}_\mathbb{A}^d.\ \psi_F \wedge \mathrm{red}(\varphi(\vec{q}_{norm}))$ *cannot be proven* **then return** ?
if $\exists \vec{x} \in \mathbb{R}_\mathbb{A}^d.\ \psi_F \wedge \mathrm{red}(\varphi(\vec{q}_{norm}))$ *is valid* **then return** $\bot$ **else return** $\top$

Algorithm 2 Checking Termination on Sets

Note that the proof of Theorem 5.19 is constructive. Moreover, the proof of [17, Thm. 1.6.] which implies Theorem 5.22 is also constructive: the idea is to use basic results from linear algebra to compute an invertible matrix $T \in \mathbb{R}_\mathbb{A}^{d\times d}$ such that $T \cdot J \cdot T^{-1}$ is triangular where J is the Jacobian matrix $\left(\frac{\partial(u_i - x_i)}{\partial x_j}\right)_{1\le i,j\le d}$. Then η with $\eta(\vec{x}) = T \cdot \vec{x}$ transforms the loop into the form (12). Hence, Theorem 5.22 is also constructive. Thus, we can not only check the existence of a suitable automorphism, but we can also compute it whenever it exists.

5.3 Analyzing *twn*-transformable loops

In this section, we generalize our results from Sect. 4 to *twn*-transformable loops. Our transformation from Sect. 5.1 and 5.2 transforms *twn*-transformable loops over update-invariant and $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$-definable sets into *twn*-loops over update-invariant and $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$-definable sets. Thus, in this section we fix a *twn*-loop $(\varphi, \vec{u})$ and such a set $F \subseteq \mathbb{R}_\mathbb{A}^d$. Let $\psi_F \in \mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$ define F.

While in Sect. 4 we were concerned with the termination of loops on a set $\mathcal{S}^d$ for a ring $\mathcal{S}$, we now show that termination of $(\varphi, \vec{u})$ *on* F can also be reduced to an existential formula (from $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$). Here, we indeed rely on the update-invariance of F as otherwise eventual non-termination and non-termination of $(\varphi, \vec{u})$ on F would not be equivalent. In Sect. 4, this equivalence is crucial since we reduce non-termination via eventual non-termination to $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$.

Again, let $\vec{q}_{norm}$ be the normalized closed form of $\vec{u}$. Similar to (4), $(\varphi, \vec{u})$ is eventually non-terminating on F iff

$$\exists \vec{x} \in F,\ n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \varphi(\vec{q}_{norm}). \tag{14}$$

In Theorem 4.9, we have seen that given a propositional formula ξ over the atoms $\{p \rhd 0 \mid p \in \mathbb{NPE}_\mathcal{S}[\vec{x}], \rhd \in \{\ge, >\}\}$, one can reduce validity of $\exists \vec{x} \in \mathcal{S}^d, n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \xi$ to validity of $\exists \vec{x} \in \mathcal{S}^d.\ \mathrm{red}(\xi) \in \mathrm{Th}_\exists(\mathcal{S})$ and the resulting formula can be computed in polynomial time from ξ. Thus, by using the formula $\exists \vec{x} \in \mathbb{R}_\mathbb{A}^d.\ \psi_F \wedge \mathrm{red}(\xi) \in \mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$ instead, we obtain Corollary 5.24.

Corollary 5.24 (Reducing Eventual Non-Termination on a Set) *For a propositional formula* ξ *over* $\{p \rhd 0 \mid p \in \mathbb{NPE}_{\mathbb{R}_\mathbb{A}}[\vec{x}], \rhd \in \{\ge, >\}\}$, *validity of*

$$\exists \vec{x} \in F, n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \xi$$

can be reduced to validity of a closed formula in $\mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$ *in polynomial time.*

By combining (14) and Corollary 5.24, one obtains the following refined version of Theorem 4.10.

Corollary 5.25 (Reducing Termination on Sets) *Termination of tnn- or twn-loops, respectively, on update-invariant* $\text{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$*-definable sets is reducible to* $\text{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$.

Example 5.26 Reconsider Example 5.23, where we have seen that $\mathbf{L}_{non-twn}$ (see Fig. 3) terminates on $\mathbb{Z}^3$ iff $\mathbf{L}_{ex}$ (see Fig. 2) terminates on the update-invariant and $\text{Th}_\exists(\mathbb{Z}, \mathbb{R})$-definable set $\hat{\eta}(\mathbb{Z}^3) = F$ defined by the formula (13). In Example 4.8, we showed that $(-4, 2, 1)$ witnesses eventual non-termination of $\mathbf{L}_{ex}$. As $\hat{\eta}(-9, 1, 4) = (-9 + 1 + 4, 1 \cdot 2, -9 + 1 \cdot 2 + 4 \cdot 2) = (-4, 2, 1)$, we have $(-4, 2, 1) \in F$. Furthermore, $(-9, 1, 4)$ witnesses eventual non-termination of $\mathbf{L}_{non-twn}$ on $\mathbb{Z}^3$ by Corollary 5.16 (c). Hence, $\mathbf{L}_{non-twn}$ does *not* terminate on $\mathbb{Z}^3$.

In addition, we get the following refined version of Corollary 4.11.

Corollary 5.27 ((Semi-)*Deciding* (*Non-*)*Termination on a Set*) *Let* $(\varphi, \vec{u})$ *be a twn-loop and let* $F \subseteq \mathbb{R}_\mathbb{A}^d$ *be update-invariant and* $\text{Th}_\exists(\mathcal{S}, \mathbb{R}_\mathbb{A})$*-definable.*

(a) *The loop* $(\varphi, \vec{u})$ *terminates over* $\mathbb{R}_\mathbb{A}$ *iff it terminates over* $\mathbb{R}$.
(b) *Termination of* $(\varphi, \vec{u})$ *on* F *is decidable if* $\mathcal{S} = \mathbb{R}_\mathbb{A}$ *or* $\mathcal{S} = \mathbb{R}$.
(c) *Non-termination of* $(\varphi, \vec{u})$ *on* F *is semi-decidable if* $\mathcal{S} = \mathbb{Z}$ *or* $\mathcal{S} = \mathbb{Q}$.

Of course, Lemma 4.12 also holds in this setting.

Corollary 5.28 *Let* $\xi = \varphi(\vec{q}_{norm})$*. Then* $\vec{e} \in \mathbb{R}_\mathbb{A}^d$ *witnesses eventual non-termination of* $(\varphi, \vec{u})$ *on* F *iff* $\psi_F(\vec{e}) \wedge \text{red}(\xi)(\vec{e})$ *holds.*

Finally, Algorithm 2 generalizes Algorithm 1 to *twn*-transformable loops.

6 Linearization of *tnn*-loops

In [48], a technique was proposed to linearize polynomial loops. As mentioned, by combining the linearization with existing decidability results [64, 65], one can conclude decidability of termination for *conjunctive twn*-loops over $\mathbb{R}$ (whereas Corollary 4.11 (b) extends this result also to non-conjunctive loops).

In this section, we adapt the linearization technique of [48] to our setting and formalize it. This allows us to obtain novel results on the complexity of linearization which we use to analyze the complexity of deciding termination for arbitrary *twn*-loops in Sect. 7.2. We start with the definition of *linearization*.

Definition 6.1 (Linearization) Let $\vec{u} \in (\mathcal{S}[\vec{x}])^d$ and $\vec{y}$ be a vector of d' fresh variables with $d' \geq d$. Let $\vec{u}\,' \in (\mathcal{S}[\vec{y}])^{d'}$ be *linear* and $\vec{w} \in (\mathcal{S}[\vec{x}])^{d'}$ with $w_i = x_i$ for all $1 \leq i \leq d$. Then $\vec{u}\,'$ is a *linearization of* $\vec{u}$ via $\vec{w}$ if $\vec{w}(\vec{u}(\vec{e})) = \vec{u}\,'(\vec{w}(\vec{e}))$ holds for all $\vec{e} \in \mathcal{S}^d$, where $\vec{u}\,'(\vec{w}(\vec{e}))$ stands for $\vec{u}\,'[\vec{y}/\vec{w}(\vec{e})]$. Instead of y_i we often write y_{w_i} for all $1 \leq i \leq d'$.

Input: *tnn*-loop $(\varphi, \vec{u})$ using the variables $\vec{x}$
Output: linear-update *tnn*-loop $(\varphi', \vec{u}')$ and $\vec{w}$
such that $(\varphi', \vec{u}')$ is a linearization of $(\varphi, \vec{u})$ via $\vec{w}$

$\vec{v} \leftarrow (x_1, \dots, x_d)$ and $\vec{w} \leftarrow (x_1, \dots, x_d)$
while $\vec{v} \neq ()$ **do**
  remove the first monomial $m = x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}$ from $\vec{v}$ and insert m at the end of $\vec{w}$
  $u'_m \leftarrow u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$, where the monomials p in non-constant addends of $u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$ are replaced by y_p and we insert p at the end of $\vec{v}$ if p is not yet contained in $\vec{w}$
return $\vec{w}$ *and* $\left(\varphi[x_1/y_{x_1}, \dots, x_d/y_{x_d}] \wedge \bigwedge_{\substack{m = \vec{x}^{\vec{z}} \text{ is contained in } \vec{w} \\ \text{where } m \notin \{x_1, \dots, x_d\}}} \left(y_m - \prod_{i=1}^{d} y_{x_i}^{z_i} = 0\right), \quad \vec{u}'\right)$

Algorithm 3 Linearizing *tnn*-Loops

So $y_1, \dots, y_d$ (i.e., $y_{w_1}, \dots, y_{w_d}$) correspond to the variables $x_1, \dots, x_d$, whereas $y_{d+1}, \dots, y_{d'}$ are used to mimic the non-linear part of $\vec{u}$ in a linear way in $\vec{u}'$. This non-linear behavior is captured by the polynomials $w_{d+1}, \dots, w_{d'}$.

Example 6.2 Let $\vec{u} = (x_2^2, x_3^2, x_3) \in (\mathbb{Z}[x_1, x_2, x_3])^3$. Then $\vec{u}' = (y_{x_2^2}, y_{x_3^2}, y_{x_3}, y_{x_3^4}, y_{x_3^2}, y_{x_3^4})$ over the variables $(y_{x_1}, y_{x_2}, y_{x_3}, y_{x_2^2}, y_{x_3^2}, y_{x_3^4})$ is a linearization of $\vec{u}$ via $\vec{w} = (x_1, x_2, x_3, x_2^2, x_3^2, x_3^4)$, since for all $\vec{e} = (e_1, e_2, e_3) \in \mathbb{Z}^3$ we have:

$$\vec{w}(\vec{u}(\vec{e})) = (e_2^2, e_3^2, e_3, e_3^4, e_3^2, e_3^4) = \vec{u}'(e_1, e_2, e_3, e_2^2, e_3^2, e_3^4) = \vec{u}'(\vec{w}(\vec{e})).$$

Here, the non-linear part of $\vec{u}$ is mimicked by the variables $y_{x_2^2}$, $y_{x_3^2}$, and $y_{x_3^4}$.

The linearization of Definition 6.1 also works when applying the update repeatedly.

Corollary 6.3 (Iterated Update of Linearization) *Let $\vec{u} \in (\mathcal{S}[\vec{x}])^d$ and $\vec{u}' \in (\mathcal{S}[\vec{y}])^{d'}$ be its linearization via $\vec{w} \in (\mathcal{S}[\vec{x}])^{d'}$.*

Then for all $\vec{e} \in \mathcal{S}^d$ and all $n \in \mathbb{N}$ we have $\vec{w}(\vec{u}^n(\vec{e})) = (\vec{u}')^n(\vec{w}(\vec{e}))$.

We now define the linearization of a loop to be a linearization of its update where the loop guard is extended to ensure that the fresh variables $y_{w_{d+1}}, \dots, y_{w_{d'}}$ indeed correspond to $w_{d+1}, \dots, w_{d'}$.

Definition 6.4 (Linearization of a Loop) Let $(\varphi, \vec{u})$ be a loop on $\mathcal{S}^d$ using the variables $\vec{x}$. A loop $(\varphi', \vec{u}')$ on $\mathcal{S}^{d'}$ using the variables $\vec{y}$ is a *linearization of* $(\varphi, \vec{u})$ via $\vec{w} \in (\mathcal{S}[\vec{x}])^{d'}$ if both

(a) $\vec{u}'$ is a linearization of $\vec{u}$ via $\vec{w}$
(b) $\varphi' = \varphi[x_1/y_{x_1}, \dots, x_d/y_{x_d}] \wedge \bigwedge_{i=d+1}^{d'} \left(y_{w_i} - w_i[x_1/y_{x_1}, \dots, x_d/y_{x_d}] = 0\right)$.

Example 6.5 Consider the loop $(\varphi, \vec{u})$ on $\mathbb{Z}^3$ where φ is $x_2 > x_3$ and $\vec{u} = (x_2^2, x_3^2, x_3)$. Then the linearization of $(\varphi, \vec{u})$ via $\vec{w}$ is $(\varphi', \vec{u}')$ where $\vec{u}'$ is as in Example 6.2 and φ' is

$$
\begin{aligned}
& \varphi[x_1/y_{x_1}, x_2/y_{x_2}, x_3/y_{x_3}] && \wedge\, y_{x_2^2} - y_{x_2}^2 = 0 \wedge y_{x_3^2} - y_{x_3}^2 = 0 \wedge y_{x_3^4} - y_{x_3}^4 = 0 \\
= \quad & y_{x_2} > y_{x_3} && \wedge\, y_{x_2^2} - y_{x_2}^2 = 0 \wedge y_{x_3^2} - y_{x_3}^2 = 0 \wedge y_{x_3^4} - y_{x_3}^4 = 0
\end{aligned}
$$

To illustrate the correspondence between $(\varphi, \vec{u})$ and $(\varphi', \vec{u}\,')$, consider the initial value $\vec{e} = (1, 3, 2)$. Here, the original loop yields the trace $(\vec{e}, \vec{u}(\vec{e}), \vec{u}^2(\vec{e}), [2]\ldots) = ((1, 3, 2), (9, 4, 2), (16, 4, 2), \ldots)$. The linearized loop operates over the variables $(y_{x_1}, y_{x_2}, y_{x_3}, y_{x_2^2}, y_{x_3^2}, y_{x_3^4})$. Thus, the first three variables correspond to x_1, x_2, x_3 and the latter ones correspond to x_2^2, x_3^2, x_3^4. So the corresponding initial value is $\vec{e}\,' = (1, 3, 2, 9, 4, 16)$ and the resulting trace is $(\vec{e}\,', \vec{u}\,'(\vec{e}\,'), [2](\vec{u}\,')^2(\vec{e}), \ldots) = ((1, 3, 2, 9, 4, 16), (9, 4, 2, 16, 4, 16), (16, 4, 2, 16, 4, 16), \ldots)$.

Lemma 6.6 shows that linearization preserves the termination behavior.

Lemma 6.6 *Let $(\varphi', \vec{u}\,')$ on $\mathcal{S}^{d'}$ be a linearization of $(\varphi, \vec{u})$ on $\mathcal{S}^d$ via $\vec{w}$.*

(a) *$(\varphi', \vec{u}\,')$ terminates on $\vec{e}\,' \in \mathcal{S}^{d'}$ if there is no $\vec{e} \in \mathcal{S}^d$ such that $\vec{e}\,' = \vec{w}(\vec{e})$.*
(b) *The loop $(\varphi, \vec{u})$ terminates on $\vec{e} \in \mathcal{S}^d$ iff $(\varphi', \vec{u}\,')$ terminates on $\vec{w}(\vec{e})$.*

While Lemma 6.6 proves the soundness of linearization, we now show how to find $\vec{u}\,'$ and $\vec{w}$ automatically, where it suffices to only use monomials (instead of arbitrary polynomials) in $\vec{w}$. A *monomial* over $\vec{x}$ has the form $x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}$ with $z_i \in \mathbb{N}$ for all $1 \le i \le d$. Let $\vec{x}^{\vec{z}}$ with $\vec{z} = (z_1, \ldots, z_d)$ abbreviate $x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}$.

The original update $\vec{u}$ consists of polynomials u_i to update the variable x_i, for all $1 \le i \le d$. The linearized update $\vec{u}\,'$ consists of polynomials u'_m to update the variables y_m for all monomials m in $\vec{w}$. Here, for any monomial $m = x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}$, the polynomial u'_m results from *replacing* each monomial p in $u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$ (i.e., the monomial p in each addend $c \cdot p$ of the polynomial $u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$) by the variable y_p. More precisely, if $u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$ has the form $c_1 \cdot p_1 + \ldots + c_k \cdot p_k$ for monomials $p_1, \ldots, p_k$ and numbers $c_1, \ldots, c_k \in \mathcal{S}$, then $u'_m = c_1 \cdot y_{p_1} + \ldots + c_k \cdot y_{p_k}$.

Algorithm 3 summarizes the linearization procedure. The vector $\vec{v}$ always contains those monomials p for which we still have to define the linearized update u'_p. So initially, $\vec{v}$ consists of the original variables, i.e., $\vec{v} = \vec{x}$. Whenever a new variable y_p is introduced in the linearized update, p is inserted into $\vec{v}$ at the end.

Example 6.7 We apply Algorithm 3 to linearize the loop $(\varphi, \vec{u})$ from Example 6.5 where φ is $x_2 > x_3$ and $\vec{u} = (x_2^2, x_3^2, x_3)$. In the beginning, we have $\vec{v} = (x_1, x_2, x_3)$. We start with x_1 and remove it from $\vec{v}$. In $u_1 = x_2^2$ we have to replace the monomial x_2^2 by the fresh variable $y_{x_2^2}$ when constructing u'_{x_1}. Hence, $u'_{x_1} = y_{x_2^2}$ and we obtain $\vec{v} = (x_2, x_3, x_2^2)$.

Next, we consider x_2, where $u_2 = x_3^2$. Thus, we obtain $u'_{x_2} = y_{x_3^2}$ and $\vec{v} = (x_3, x_2^2, x_3^2)$. Then we take x_3, where $u_3 = x_3$. Hence, $u'_{x_3} = y_{x_3}$, but we do not insert x_3 into $\vec{v}$ again, since we just computed u'_{x_3}. So we have $\vec{v} = (x_2^2, x_3^2)$.

We now handle x_2^2. For the linearized update, we take $u_2^2 = x_3^4$ but replace the monomial x_3^4 by a fresh variable $y_{x_3^4}$. Hence, $u'_{x_2^2} = y_{x_3^4}$ and $\vec{v} = (x_3^2, x_3^4)$.

Next, we take the monomial x_3^2 and in $u_3^2 = x_3^2$ we have to replace the monomial by $y_{x_3^2}$. This leads to the linearized update $u'_{x_3^2} = y_{x_3^2}$, but we do not insert x_3^2 into $\vec{v}$ again, since we just computed $u'_{x_3^2}$. Hence, we have $\vec{v} = (x_3^4)$.

Finally, we treat x_3^4 and in $u_3^4 = x_3^4$ we replace the monomial by $y_{x_3^4}$, i.e., $u'_{x_3^4} = y_{x_3^4}$. Now $\vec{v} = ()$. Hence, Algorithm 3 terminates and returns the loop with the guard φ' from Example 6.5 and the (linear) update $(y_{x_2^2}, y_{x_3^2}, y_{x_3}, y_{x_3^4}, y_{x_3^2}, y_{x_3^4})$ over the variables $(y_{x_1}, y_{x_2}, y_{x_3}, y_{x_2^2}, y_{x_3^2}, y_{x_3^4})$, as in Example 6.2.

Now we infer an upper bound on Algorithm 3's complexity. To this end, we will show that the degrees of the monomials in $\vec{w}$ which are used for the linearization can be bounded by the maximal *dependency degree* of the loop's update $\vec{u}$. For $1 \le i \le d$, the dependency degree $\operatorname{depdeg}_{\vec{u}}(x_i)$ is the degree of u_i, but this degree is expressed in terms of those variables that are minimal w.r.t. $\succ_{\vec{u}}$. Recall that u_i has the form $c_i \cdot x_i + p_i$ where p_i is a polynomial which only contains variables that are smaller than x_i w.r.t. $\succ_{\vec{u}}$. W.l.o.g. we may assume that $x_i \succ_{\vec{u}} x_j$ implies $i > j$ for all $1 \le i,j \le d$. Then for every monomial $x_{i+1}^{z_{i+1}} \cdot \ldots \cdot x_d^{z_d}$ in p_i, the corresponding dependency degree is $z_{i+1} \cdot \operatorname{depdeg}_{\vec{u}}(x_{i+1}) + \ldots + z_d \cdot \operatorname{depdeg}_{\vec{u}}(x_d)$. The dependency degree of p_i is the maximal dependency degree of its monomials.

Definition 6.8 (Dependency Degree) Let $(\varphi, \vec{u})$ be a *twn*-loop with $u_i = c_i \cdot x_i + p_i$ for all $1 \le i \le d$, where p_i only contains variables that are smaller than x_i w.r.t. $\succ_{\vec{u}}$. We define the *dependency degree* w.r.t. $\vec{u}$ as follows:

- $\operatorname{depdeg}_{\vec{u}}(x_i) = \max\{1, \operatorname{depdeg}_{\vec{u}}(p_i)\}$ for all $1 \le i \le d$.
- $\operatorname{depdeg}_{\vec{u}}(p) = \max\{ \operatorname{depdeg}_{\vec{u}}(m) \mid m$ is a monomial in $p\}$ for every non-zero $p \in \mathcal{S}[\vec{x}]$ and $\operatorname{depdeg}_{\vec{u}}(0) = -\infty$.
- $\operatorname{depdeg}_{\vec{u}}(x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}) = \sum_{i=1}^{d} z_i \cdot \operatorname{depdeg}_{\vec{u}}(x_i)$ for all $z_1, \ldots, z_d \in \mathbb{N}$.

Since $\succ_{\vec{u}}$ is well founded by the triangularity of the loop, $\operatorname{depdeg}_{\vec{u}}$ is well defined: for the variables x_i which are minimal w.r.t. $\succ_{\vec{u}}$, p_i is a constant and thus, $\operatorname{depdeg}_{\vec{u}}(x_i) = 1$. For other variables x_i with $p_i \neq 0$, we can compute $\operatorname{depdeg}_{\vec{u}}(p_i)$ because $\operatorname{depdeg}_{\vec{u}}(x_j)$ is already known for all variables x_j occurring in p_i. Lemma 6.9 states three easy observations on depdeg. Here, deg denotes the degree of monomials or polynomials, i.e., $\deg(x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}) = z_1 + \ldots + z_d$.

Lemma 6.9 *Let $(\varphi, \vec{u})$ be a twn-loop.*

(a) *For every monomial m over $\vec{x}$, we have* $\deg(m) \le \operatorname{depdeg}_{\vec{u}}(m)$.
(b) *If deg is the maximum of* 1 *and the highest degree of any polynomial in $\vec{u}$, then for any* $1 \le i \le d$ *we have* $\operatorname{depdeg}_{\vec{u}}(x_i) \le deg^{d-i}$.
(c) *For mdepdeg* $= \max\{ \operatorname{depdeg}_{\vec{u}}(x_i) \mid 1 \le i \le d\}$, *we have* $mdepdeg \le deg^{d-1}$.

Example 6.10 Again, we consider a loop as in Example 6.7 with update $\vec{u} = (x_2^2, x_3^2, x_3)$. Then, $\operatorname{depdeg}_{\vec{u}}(x_3) = \max\{1, \operatorname{depdeg}_{\vec{u}}(0)\} = \max\{1, -\infty\} = 1$, $\operatorname{depdeg}_{\vec{u}}(x_2) = \operatorname{depdeg}_{\vec{u}}(x_3^2) = 2 \cdot \operatorname{depdeg}_{\vec{u}}(x_3) = 2 \cdot 1 = 2$, and $\operatorname{depdeg}_{\vec{u}}(x_1) = \operatorname{depdeg}_{\vec{u}}(x_2^2) = 2 \cdot \operatorname{depdeg}_{\vec{u}}(x_2) = 2 \cdot 2 = 4$. So intuitively, the update of x_1 has degree 4 in terms of the $\succ_{\vec{u}}$-minimal variable x_3 since the update of x_2 is quadratic in x_3 and x_2^2 then has degree 4 w.r.t. x_3. Here, *mdepdeg* $= 4$

and for the maximal degree $deg = 2$ occurring in the update, we indeed have $mdepdeg = deg^{d-1} = 2^{3-1}$. So the bound on $mdepdeg$ in Lemma 6.9 (c) is tight.

As another example, consider the update $\vec{u} = (3 \cdot x_1 + 5 \cdot x_2^4 \cdot x_3^6 + 7 \cdot x_3^8, x_3^2, 9)$. Now we have $\operatorname{depdeg}_{\vec{u}}(x_3) = \max\{1, \operatorname{depdeg}_{\vec{u}}(9)\} = \max\{1, 0\} = 1$, $\operatorname{depdeg}_{\vec{u}}(x_2) = \operatorname{depdeg}_{\vec{u}}(x_3^2) = \operatorname{depdeg}_{\vec{u}}(x_3^2) = 2 \cdot \operatorname{depdeg}_{\vec{u}}(x_3) = 2 \cdot 1 = 2$, and $\operatorname{depdeg}_{\vec{u}}(x_1) = \operatorname{depdeg}_{\vec{u}}(5 \cdot x_2^4 \cdot x_3^6 + 7 \cdot x_3^8)$ $= \max\{\operatorname{depdeg}_{\vec{u}}(x_2^4 \cdot x_3^6), \operatorname{depdeg}_{\vec{u}}(x_3^8)\} = \max\{4 \cdot \operatorname{depdeg}_{\vec{u}}(x_2) + 6 \cdot \operatorname{depdeg}_{\vec{u}}(x_3), 8 \cdot \operatorname{depdeg}_{\vec{u}}(x_3)\}$ $= \max\{4 \cdot 2 + 6 \cdot 1, 8 \cdot 1\} = 14$.

Now we prove that Algorithm 3 only constructs updates u'_m for monomials m with $\operatorname{depdeg}(m) \leq mdepdeg$. Hence, this also proves termination of the algorithm since there are only finitely many such monomials, and it allows us to give a bound on the number of iterations of the algorithm's **while**-loop.

Theorem 6.11 (Dependency Degree Suffices for Linearization)

(a) *Algorithm* 3 *only computes* u'_m *for monomials* m *with* $\operatorname{depdeg}(m) \leq mdepdeg$.

(b) *The* **while**-*loop of Algorithm* 3 *is executed at most* $\binom{d + mdepdeg}{mdepdeg} - 1$ *times.*

(c) *Algorithm* 3 *terminates.*

The following theorem summarizes the main properties of Algorithm 3.

Theorem 6.12 (Soundness of Algorithm 3)

(a) *For any tnn-loop* $(\varphi, \vec{u})$, *Algorithm* 3 *computes a linearization* $(\varphi', \vec{u}')$ *via* $\vec{w} = (m_1, \ldots, m_{d'})$, *where* $\deg(m_i) \leq \operatorname{depdeg}(m_i) \leq mdepdeg$ *for all* $1 \leq i \leq d'$.

(b) *The loop* $(\varphi, \vec{u})$ *terminates iff* $(\varphi', \vec{u}')$ *does.*

(c) *The loop* $(\varphi', \vec{u}')$ *is a linear-update tnn-loop.*

As mentioned, the technique in this section is based on the linearization method of [48], where instead of *tnn*-loops as in Theorem 6.12, [48] works in the setting of solvable loops (Definition 2.3). But [48] has no notion like the dependency degree of Definition 6.8. Instead they only consider the degree of the polynomials in the update $\vec{u}$. However, Example 6.7 shows that the polynomials in $\vec{w}$ that are used for the linearization may have a higher degree than the ones in $\vec{u}$. Here, the polynomials in $\vec{u} = (x_2^2, x_3^2, x_3)$ only have degree 2. However, x_1 is (eventually) updated to x_3^4. Thus, to linearize this loop, polynomials up to degree 2 do *not* suffice, but $\vec{w}$ must contain a polynomial of degree 4 like x_3^4.

As we showed in Theorem 6.11 (a), the dependency degree (and hence, also the degree) of the polynomials in $\vec{w}$ is bounded by $mdepdeg = \max\{\operatorname{depdeg}_{\vec{u}}(x_i) \mid 1 \leq i \leq d\}$. Indeed, in Example 6.7 we have $\operatorname{depdeg}_{\vec{u}}(x_1) = 4$. Hence, our new concept of the dependency degree was needed for the upper bound on the number of iterations of the linearization algorithm in Theorem 6.11 (b). Based on this, we can now infer the asymptotic complexity of Algorithm 3. As mentioned, we will need this in Sect. 7.2 to analyze the complexity of deciding termination of *twn*-loops.

By Theorem 6.11 (b), the **while**-loop of Algorithm 3 is executed at most $\binom{d + mdepdeg}{mdepdeg} - 1$ times. Since $\binom{n}{k} \in \mathcal{O}(n^k)$ for any natural numbers $n \geq k$, we have

$$\binom{d + mdepdeg}{mdepdeg} = \binom{d + mdepdeg}{d} \in \mathcal{O}((d + mdepdeg)^{mdepdeg}) \cap \mathcal{O}((d + mdepdeg)^d).$$

By Lemma 6.9 (c), we have $mdepdeg \leq deg^{d-1}$ where deg is the maximum of 1 and the highest degree of any polynomial in the update $\vec{u}$. Hence,

$$\binom{d + mdepdeg}{mdepdeg} \in \mathcal{O}((d + deg^{d-1})^d) \cap \mathcal{O}((d + deg^{d-1})^{deg^{d-1}}) \subseteq \mathcal{O}((d + deg^{d-1})^d).$$

For the expression $(d + deg^{d-1})^d$ we have (see also App. A.23):

$$(d + deg^{d-1})^d \leq 2 \cdot 2^{d + \mathrm{ld}\,(deg) \cdot (d-1) \cdot d} \tag{15}$$

Here, as usual, ld denotes the logarithm to the base 2.

Thus, $(d + deg^{d-1})^d$ is at most *exponential* in d, i.e., the number of iterations of Algorithm 3 is at most exponential in d. In each such iteration, one has to compute a new polynomial $u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$. By Theorem 6.11 (a), this polynomial only contains monomials m with $\mathrm{depdeg}\,(m) \leq mdepdeg$ and there are $\binom{d + mdepdeg}{mdepdeg} \in \mathcal{O}((d + deg^{d-1})^d)$ many such monomials (see Theorem 6.11 (b)). To compute their coefficients, one has to multiply up to $z_1 + \ldots + z_d$ factors, where $z_1 + \ldots + z_d \leq mdepdeg \leq deg^{d-1}$. This corresponds to a nested multiplication of two factors, where the result of one multiplication step is the input to the next multiplication, and the depth of the nesting is exponential in d. So the results and the factors of the multiplications grow at most doubly exponentially in d. Therefore, this proves Lemma 6.13 (a), i.e., the runtime of Algorithm 3 is at most double exponential.

However, if the number of variables d is bounded by a constant D, then the number of iterations of Algorithm 3 and the number of monomials in the linearized updates is bounded by $\binom{d + mdepdeg}{mdepdeg} \in \mathcal{O}((deg^D + D)^D)$, which is polynomial in deg. For their coefficients, one has to multiply up to $mdepdeg \leq deg^{D-1}$ (i.e., polynomially) many factors, i.e., this corresponds to a nested multiplication where the depth of the nesting is polynomial in deg. So the results and the factors of the multiplications grow at most exponentially in deg. Therefore, then linearization can be computed in exponential time. This proves Lemma 6.13 (b).

Lemma 6.13 *Let $D \in \mathbb{N}$ be fixed. The lineariza tion of a tnn-loop*

(a) *can be computed in double exponential time.*
(b) *can be computed in exponential time if the number of variables d is at most D.*

Example 6.14 While Lemma 6.13 only gives upper bounds on the complexity of linearization, the loop $\mathbf{L}_{non-pspace}$ from (10) can be used to infer lower bounds. Here, the linearized loop operates on the variables

$$y_{x_1}, \ldots, y_{x_d}, \quad y_{x_2^d}, \ldots, y_{x_d^d}, \quad y_{x_3^{(d^2)}}, \ldots, y_{x_d^{(d^2)}}, \quad \ldots \quad y_{x_{d-1}^{(d^{d-2})}}, y_{x_d^{(d^{d-2})}}, \quad y_{x_d^{(d^{d-1})}}$$

$$\textbf{while } (\varphi) \textbf{ do } \vec{x} \leftarrow A \cdot \vec{x} + \vec{b}$$

Fig. 4 Linear-update loop

$$\textbf{while } (\varphi \wedge x_{\vec{b}} = 1) \textbf{ do } \begin{bmatrix} \vec{x} \\ x_{\vec{b}} \end{bmatrix} \leftarrow \begin{bmatrix} A & \vec{b} \\ \vec{0}^T & 1 \end{bmatrix} \cdot \begin{bmatrix} \vec{x} \\ x_{\vec{b}} \end{bmatrix}$$

Fig. 5 Homogeneous linear-update loop

and the corresponding linearized update $\vec{u}'$ instantiates

- y_{x_1} by $y_{x_2^d}$,
- $y_{x_2^{(d^i)}}$ with $y_{x_3^{(d^{(i+1)})}}$ for all $0 \leq i \leq 1$,
- ...
- $y_{x_{d-1}^{(d^i)}}$ with $y_{x_d^{(d^{(i+1)})}}$ for all $0 \leq i \leq d-2$, and
- $y_{x_d^{(d^i)}}$ with $d^{(d^i)} \cdot y_{x_d^{(d^i)}}$ for all $0 \leq i \leq d-1$.

So in particular, the update contains the constant $d^{(d^{d-1})}$ which shows that this linearization requires exponential space if d is not bounded.

7 Complexity of deciding termination

In this section, we study the complexity of deciding termination for different classes of loops by using our results from Sects. 4 to 6. We first regard *linear-update* loops in Sect. 7.1, where the update is of the form $\vec{x} \leftarrow A \cdot \vec{x} + \vec{b}$ with $A \in \mathcal{S}^{d \times d}$ and $\vec{b} \in \mathcal{S}^d$. The reason for this restriction is that such loops can always be transformed into *twn*-form by our transformation $\mathbb{Tr}$ from Sect. 5. More precisely, we show that termination of linear loops with rational spectrum is Co-NP-complete if $\mathcal{S} \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}_{\mathbb{A}}\}$ and that termination of linear-update loops with real spectrum is $\forall\mathbb{R}$-complete if $\mathcal{S} = \mathbb{R}_{\mathbb{A}}$. Since our proof is based on a reduction to $\mathrm{Th}_{\exists}(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$, and $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$ are elementary equivalent, our results also hold if the program variables range over $\mathbb{R}$. By combining these results with our observations on the complexity of linearization from Sect. 6, we then analyze the complexity of deciding termination for arbitrary *twn*-loops in Sect. 7.2. In Sect. 7.3, we show that there is an important subclass of linear loops where our decision procedure for termination works efficiently, i.e., when the number of eigenvalues of the update matrix is bounded, then termination can be decided in polynomial time. Here, we again use our transformation $\mathbb{Tr}$ from Sect. 5.

For our complexity results, we assume the usual dense encoding of univariate polynomials, i.e., a polynomial of degree k is represented as a list of $k + 1$ coefficients. As discussed in [55], many problems which are considered to be efficiently solvable become intractable if polynomials are encoded sparsely (e.g., as lists of monomials where each monomial is a pair of its non-zero coefficient and its degree). With densely encoded polynomials, all common representations of algebraic numbers can be converted into each other in polynomial time [3].

7.1 Complexity of deciding termination for linear-update loops

When analyzing linear-update loops, w.l.o.g. we can assume $\vec{b} = \vec{0}$ since a loop of the form in Fig. 4 terminates iff the loop in Fig. 5 terminates, where $x_{\vec{b}}$ is a fresh variable (see [31, 49]). Moreover, $\vec{e}$ witnesses (eventual) non-termination for the loop in Fig. 4 iff $\begin{bmatrix}\vec{e}\\1\end{bmatrix}$ witnesses (eventual) non-termination for the loop in Fig. 5. Note that the only eigenvalue of $\begin{bmatrix}A & \vec{b}\\ \vec{0}^T & 1\end{bmatrix}$ whose multiplicity increases in comparison to A is 1. Thus, to decide termination of linear-update loops with rational or real spectrum, respectively, it suffices to decide termination of *homogeneous* loops of the form $(\varphi, A \cdot \vec{x})$ where A has only rational or real eigenvalues.

Such loops can *always* be transformed into *twn*-form using our transformation Tr from Sect. 5. To compute the required automorphism η, we compute the Jordan normal form Q of A together with the corresponding transformation matrix T, i.e., T is an invertible real matrix such that $A = T^{-1} \cdot Q \cdot T$. Then Q is a triangular real matrix whose diagonal consists of the eigenvalues $\lambda \in \mathbb{R}_{\mathbb{A}}$ of A. We define $\eta \in \mathrm{End}_{\mathbb{R}_{\mathbb{A}}}\left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)$ by $\eta(\vec{x}) = T \cdot \vec{x}$. Then $\eta \in \mathrm{Aut}_{\mathbb{R}_{\mathbb{A}}}\left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)$ has the inverse $\eta^{-1}(\vec{x}) = T^{-1} \cdot \vec{x}$. Thus, $\mathrm{Tr}_{\eta}(\varphi, A \cdot \vec{x})$ is a *twn*-loop with the update

$$(\eta(\vec{x}))\,(A \cdot \vec{x})\,(\eta^{-1}(\vec{x})) = T \cdot A \cdot T^{-1} \cdot \vec{x} = Q \cdot \vec{x}.$$

To analyze termination of the loop on $\mathcal{S}^d$, we have to consider termination of the transformed loop on $F = \widehat{\eta}(\mathcal{S}^d) = T \cdot \mathcal{S}^d$ (see Corollary 5.16).

The Jordan normal form Q as well as the matrices T and T^{-1} can be computed in polynomial time [23, 54]. Hence, we can decide whether all eigenvalues are rational or real numbers in polynomial time by checking the diagonal entries of Q. Thus, we obtain the following lemma.

Lemma 7.1 *Let* $(\varphi, A \cdot \vec{x})$ *be a linear-update loop.*

(a) *It is decidable in polynomial time whether A has only rational or real eigenvalues.*
(b) *If A has only real eigenvalues, we can compute a linear* $\eta \in \mathrm{Aut}_{\mathbb{R}_{\mathbb{A}}}\left(\mathbb{R}_{\mathbb{A}}[\vec{x}]\right)$ *such that* $\mathrm{Tr}_{\eta}(\varphi, A \cdot \vec{x})$ *is a linear-update twn-loop in polynomial time.*
(c) *If* $(\varphi, A \cdot \vec{x})$ *is a linear loop, then so is* $\mathrm{Tr}_{\eta}(\varphi, A \cdot \vec{x})$.

So every linear(-update) loop with real spectrum can be transformed into a linear(-update) *twn*-loop, i.e., the transformation Tr from Sect. 5 is *complete* for such linear(-update) loops. Note that Lemma 7.1 (a) yields an efficient check whether a given linear\(-update)\ loop has rational or real spectrum.

As chaining (Definition 3.1) can clearly be done in polynomial time, w.l.o.g. we may assume that $\mathrm{Tr}_{\eta}(\varphi, A \cdot \vec{x}) = (\varphi', Q \cdot \vec{x})$ is *tnn*. Next, to analyze termination of a *tnn*-loop, our technique of Sect. 4 (resp. Sect. 5.3) uses a closed form for the update. For *tnn*-loops $(\varphi', Q \cdot \vec{x})$ where Q is a triangular matrix with non-negative diagonal entries, a suitable (i.e., poly-exponential) closed form can be computed in polynomial time analogously to [35, Prop. 5.2]. This closed form is linear in $\vec{x}$ (we will discuss this closed form in Sect. 7.3, see Lemma 7.12).

According to our approach in Sect. 5.3, we now proceed as in Algorithm 2 and construct the formula $\exists \vec{x} \in \mathbb{R}_{\mathbb{A}}^d.\ \psi_F \wedge \mathrm{red}\,(\varphi(\vec{q}_{norm})) \in \mathrm{Th}_{\exists}\,(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$ in polynomial time due to Corollary 5.24. Hence, we get the following lemma.

Lemma 7.2 *Let $(\varphi, A \cdot \vec{x})$ be a linear-update loop with real spectrum. Then we can compute a closed formula $\psi \in \mathrm{Th}_{\exists}(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$ in polynomial time such that ψ is valid iff the loop is non-terminating. If φ is linear, then so is ψ.*

Note that ψ is existentially quantified. Hence, if the loop has rational spectrum and coefficients, φ (and thus also ψ) is linear, and $\mathcal{S} \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}_{\mathbb{A}}, \mathbb{R}\}$, then invalidity of ψ is in Co-NP as validity of such formulas is in NP, see [50]. Thus, we get the first main result of this section. Here, we fix an inaccuracy in [20, Thm. 42], where we also allowed irrational eigenvalues and coefficients, and thus ψ may contain irrational coefficients. However, to the best of our knowledge, it is not known whether validity of linear formulas from $\mathrm{Th}_{\exists}(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$ with irrational algebraic coefficients is in NP.

Theorem 7.3 (Co-NP-Completeness) *For linear loops $(\varphi, A \cdot \vec{x} + \vec{b})$ with rational spectrum where $\varphi \in \mathrm{Th}_{qf}(\mathbb{Q}), A \in \mathbb{Q}^{d\times d}$, and $\vec{b} \in \mathbb{Q}^d$, termination over $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}_{\mathbb{A}}$, and $\mathbb{R}$ is Co-NP-complete.*

Co-NP-hardness follows from Co-NP-hardness of unsatisfiability of propositional formulas: let ξ be a propositional formula over the variables $\vec{x}$. Then the loop $(\xi[x_i/(x_i > 0) \mid 1 \leq i \leq d], \vec{x})$ terminates iff ξ is unsatisfiable.

We now consider linear-update loops with real spectrum (and possibly non-linear conditions) on $\mathbb{R}_{\mathbb{A}}^d$ and $\mathbb{R}^d$. Here, non-termination is $\exists\mathbb{R}$-complete.

Definition 7.4 ($\exists\mathbb{R}$ [57, 58]) Let

$$\mathrm{Th}_{\exists}(\mathbb{R})_{\top} = \{\psi \in \mathrm{Th}_{\exists}(\mathbb{R}) \mid \psi \text{ closed and valid}\}.$$

The complexity class $\exists\mathbb{R}$ is the closure of $\mathrm{Th}_{\exists}(\mathbb{R})_{\top}$ under poly-time-reductions.

We have NP $\subseteq \exists\mathbb{R} \subseteq$ PSPACE (see [12]). By Lemma 7.2, non-termination of linear-update loops over $\mathbb{R}_{\mathbb{A}}$ with real spectrum is in $\exists\mathbb{R}$. It is also $\exists\mathbb{R}$-hard since $(\varphi, \vec{x})$ is non-terminating iff $\exists \vec{x} \in \mathbb{R}_{\mathbb{A}}^d.\ \varphi$ is valid. So non-termination is $\exists\mathbb{R}$-complete, i.e., termination is Co-$\exists\mathbb{R}$-complete (where Co-$\exists\mathbb{R} = \forall\mathbb{R}$ [58]).

Theorem 7.5 ($\forall\mathbb{R}$-Completeness) *Termination of linear-update loops with real spectrum over $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$ is $\forall\mathbb{R}$-complete.*

7.2 Complexity of deciding termination for *twn*-loops over $\mathcal{S} \in \{\mathbb{R}_{\mathbb{A}}, \mathbb{R}\}$

By Corollary 4.11 (b) termination is decidable for arbitrary *twn*-loops over $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$. So in this section, we discuss the complexity of this decision problem. First of all, deciding termination of arbitrary *twn*-loops over $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$ is $\forall\mathbb{R}$-hard since termination of linear-update loops with real spectrum over $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$ is $\forall\mathbb{R}$-hard by Theorem 7.5 and any such linear-update loop can be transformed into a *twn*-loop in polynomial time by Lemma 7.1 (b). Thus, we have a lower bound on the complexity of deciding termination for arbitrary *twn*-loops.

$$\textbf{while } (x_1 \geq -x_3) \textbf{ do} \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} x_1 + x_2 \\ x_2 + x_3 \\ x_3 - 1 \end{bmatrix}$$

Fig. 6 Uniform loop [7] via Polynomials

$$\textbf{while } (x_1 \geq -x_3 \land x_4 = 1) \textbf{ do} \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \leftarrow \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Fig. 7 Uniform loop [7] via Matrix

To find an upper asymptotic bound for deciding termination of *twn*-loops, first note that we can restrict ourselves to *tnn*-loops again, as any *twn*-loop can be transformed into a *tnn*-loop by chaining (Definition 3.1) in polynomial time.

The next step of our decision procedure in Algorithm 1 is to compute closed forms for the update. However, the loop $\mathbf{L}_{non-pspace}$ in (10) showed that in general, the computation of closed forms cannot be done in polynomial space. On the other hand, as mentioned in Sect. 7.1, for *linear-update* loops, closed forms can be computed in polynomial time. To benefit from this upper bound, we therefore do not proceed directly as in Algorithm 1, but instead we first linearize the *tnn*-loop. While linearization cannot be computed in polynomial space either (see Example 6.14), in Sect. 6 we formalized and analyzed the complexity of the linearization technique from [48]. Given a *tnn*-loop $(\varphi, \vec{u})$ we can compute a linear-update *tnn*-loop $(\varphi', \vec{u}\,')$ such that $(\varphi, \vec{u})$ terminates if and only if $(\varphi', \vec{u}\,')$ does (Theorem 6.12). Clearly, linear-update *tnn*-loops over $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$ always have real spectrum as the eigenvalues of a triangular matrix are its diagonal entries. So by using this linearization, we can give an upper complexity bound for deciding termination of arbitrary *twn*-loops.

As linearization can be computed in double exponential time and thus, also in double exponential space, and termination of linear-update loops is in $\forall\mathbb{R} \subseteq$ PSPACE $\subseteq$ EXPTIME by Lemma 7.2 (where the size of the linear-update loop may be at most double exponential), we obtain that deciding termination of *twn*-loops is in 3-EXPTIME (i.e., it is between $\forall\mathbb{R}$ and 3-EXPTIME). Moreover, if the number of variables is bounded, checking validity of a formula in $\text{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$ is in P (see [2]). In this case, combining linearization which can be computed in exponential time and thus, also in exponential space when d is bounded (Lemma 6.13 (b)), and deciding termination of linear-update loops which is polynomial in this case (where the size of the linear-update loop may be at most exponential), we obtain Theorem 7.6 (b).

Theorem 7.6 (Membership in 3-EXPTIME) *Let $D \in \mathbb{N}$ be fixed. Termination of twn-loops over $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$*

(a) *is in* 3-EXPTIME.
(b) *is in* EXPTIME *if the number of variables d is at most D.*

7.3 Complexity of deciding termination for uniform loops

In Sect. 7.1, we showed that termination of linear loops with rational spectrum is Co-NP-complete. For proving Co-NP-hardness, we used the trivial update $\vec{x} \leftarrow \vec{x}$ induced by the identity matrix. Therefore, the question arises whether imposing suitable restrictions to the update matrix (which exclude the identity matrix) leads to a "more efficient" decision procedure for termination (assuming $\mathsf{P} \neq \mathsf{NP}$). We now analyze a special case of linear loops (so-called *uniform* loops) and show that for these loops deciding termination is *polynomial*, if one fixes the number of eigenvalues of the update matrix.

In Sect. 7.3.1, we introduce uniform loops and parameterized decision problems, and state the main result of Sect. 7.3 (Theorem 7.10). To prove it, we show that for uniform loops, instantiating the variables in the loop guard by $\vec{q}_{norm}$ (as required by our decision procedure from Sect. 4) results in formulas of a special structure (so-called *interval conditions*, see Sects. 7.3.2 and 7.3.3). Validity of these formulas can be checked in polynomial time (Sect. 7.3.4) which proves Theorem 7.10 for uniform loops over $\mathbb{Q}$, $\mathbb{R}_{\mathbb{A}}$, and $\mathbb{R}$. In Sect. 7.3.5 we show that our result holds for uniform loops over $\mathbb{Z}$ as well.

7.3.1 Uniform loops and the parameterized complexity class XP

Definition 7.7 (Uniform Loop) A linear loop $(\varphi, A \cdot \vec{x})$ over $\mathcal{S} \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}_{\mathbb{A}}\}$ is *uniform* if each eigenvalue λ of A is a *non-negative* number from $\mathcal{S}$ whose eigenspace w.r.t. A is one-dimensional, i.e., λ has *geometric multiplicity* 1.

The latter property is equivalent to requiring that there is exactly one Jordan block for each eigenvalue in A's Jordan normal form. To grasp uniform loops intuitively, consider triangular linear loops with updates $x_i \leftarrow \lambda \cdot x_i + p_i$ for all $1 \leq i \leq d$, where the factor $\lambda \geq 0$ is the same for all i. These loops are uniform iff the relation $\succ_{\vec{u}}$ is total (or equivalently, iff the variables can be ordered such that the super-diagonal of A does not contain zeros).

Lemma 7.8 *A triangular linear loop $(\varphi, A \cdot \vec{x})$ where all diagonal entries are identical and non-negative is uniform iff $\succ_{\vec{u}}$ is a total ordering.*

Thus, loops like the leading example from [7] in Fig. 6 which is equivalent to Fig. 7 are uniform. In contrast, a loop is not uniform if each x_i is updated to $\lambda \cdot x_i + c_i$ for constants $c_i \in \mathcal{S}$. The reason is that the x_i do not occur in each other's updates. Hence, we have $x_i \not\succ_{\vec{u}} x_j$ and $x_j \not\succ_{\vec{u}} x_i$ for all $1 \leq i, j \leq d$.

So in particular, a uniform loop cannot have more than one update of the form $x_i \leftarrow x_i$. However, the loop condition can still be an arbitrary Boolean formula over linear inequations. Thus, our complexity result is quite surprising since it shows that for this class of loops, termination is easier to decide than satisfiability of the condition (e.g., unsatisfiability of linear formulas over $\mathbb{R}_{\mathbb{A}}$ is $\mathsf{Co\text{-}NP}$-complete). Intuitively, the reason is that our class *prohibits* multiple updates like $x_i \leftarrow x_i$ where variables "stabilize" and where termination is essentially equivalent to unsatisfiability of the condition.

To give an intuition how hard the restriction to uniform loops is, we analyzed the *TPDB* [63] used at the *Termination and Complexity Competition* [25]. In the category for "Termination of Integer Transition Systems (ITSs)" we identified 467 polynomial loops with non-constant guard (i.e., termination is not trivial) and 290 (62 %) of them are uniform loops over $\mathbb{Z}$. Similarly, in the category for "Complexity Analysis of ITSs" we found 1,258 such polynomial loops and 452 (36 %) are uniform. In fact, in practice one is often interested in termination of triangular loops where after chaining, all variables belong to the eigenvalues 0 or 1. The reason is that termination is usually easy to show if there are eigenvalues greater than 1, because they lead to exponential growth. Thus, if the loop terminates,

$$Q_\lambda = \begin{bmatrix} \lambda & 1 & 0 & \cdots & 0 \\ 0 & \lambda & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda & 1 \\ 0 & 0 & \cdots & 0 & \lambda \end{bmatrix}$$

Fig. 8 Jordan block

$$Q_\lambda^n = \begin{bmatrix} \binom{n}{0}\cdot\lambda^n & \binom{n}{1}\cdot\lambda^{n-1} & \binom{n}{2}\cdot\lambda^{n-2} & \cdots & \binom{n}{\nu-1}\cdot\lambda^{n-\nu+1} \\ 0 & \binom{n}{0}\cdot\lambda^n & \binom{n}{1}\cdot\lambda^{n-1} & \cdots & \binom{n}{\nu-2}\cdot\lambda^{n-\nu+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \binom{n}{1}\cdot\lambda^{n-1} \\ 0 & 0 & 0 & \cdots & \binom{n}{0}\cdot\lambda^n \end{bmatrix}$$

Fig. 9 Multiplication of Jordan block

termination is usually reached after few steps (e.g., consider a loop $(x \leq c, 2 \cdot x)$ for any constant c). Hence, the number of eigenvalues k is usually smaller than the number d of program variables. In this section we show that the complexity for deciding termination of uniform loops is exponential in k but not in d. More precisely, termination of uniform loops is in the *parameterized complexity class* XP, where the *parameter* is the number k of eigenvalues.

Definition 7.9 (Parameterized Decision Problem, XP [16]) A *parameterized decision problem* is a language $\mathcal{L} \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet. The second component (from $\mathbb{N}$) is called the *parameter* of the problem.

A parameterized problem $\mathcal{L}$ is *slicewise polynomial* if the time needed for deciding the question "$(x, k) \in \mathcal{L}$?" is in $\mathcal{O}(|x|^{f(k)})$ where f is a computable function depending only on k.

The corresponding complexity class is called XP.

In the remainder of this section, we prove that for any fixed $k \in \mathbb{N}$, termination of uniform loops with k eigenvalues is decidable in polynomial time.

Theorem 7.10 (Parameterized Complexity of k-Termination) *We define the parameterized decision problem* k-termination *as follows*: $((\varphi, A \cdot \vec{x}), k) \in \mathcal{L}_{k\text{-termination}}$ *iff the loop* $(\varphi, A \cdot \vec{x})$ *terminates over* $\mathcal{S}$ *and* A *has* k *eigenvalues.*

For uniform loops, k*-termination is in* XP. *Moreover, for such loops*, k*-termination over* $\mathbb{R}$ *is in* XP *as well.*

7.3.2 Hierarchical expressions and partitions

We now elaborate on the closed forms arising from uniform loops. To this end, we fix a uniform loop $(\varphi, A \cdot \vec{x})$. Let $\operatorname{spec}(A) = \{\lambda_1, \ldots, \lambda_k\}$ be A's eigenvalues where $0 \leq \lambda_1 < \ldots < \lambda_k$, let Q be A's Jordan normal form where the Jordan blocks are ordered such that the numbers on the diagonal are weakly monotonically increasing, and let T be the corresponding transformation matrix, i.e., $A = T^{-1} \cdot Q \cdot T$. Moreover, let η be the automorphism defined by $\eta(\vec{x}) = T \cdot \vec{x}$ and let $\operatorname{Tr}_\eta(\varphi, A \cdot \vec{x}) = (\eta^{-1}(\varphi), Q \cdot \vec{x}) = (\varphi', Q \cdot \vec{x})$ as in Sect. 7.1.

$$\textbf{while } (\varphi) \textbf{ do} \quad \vec{x} \leftarrow A \cdot \vec{x} \qquad A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix} \qquad \vec{q}_{norm} = \begin{bmatrix} x_1 + n \cdot x_2 \\ x_2 \\ x_3 \cdot 2^n + (\frac{x_4}{2} - \frac{x_5}{8}) \cdot n \cdot 2^n + \frac{x_5}{8} \cdot n^2 \cdot 2^n \\ x_4 \cdot 2^n + \frac{x_5}{2} \cdot n \cdot 2^n \\ x_5 \cdot 2^n \end{bmatrix}$$

Fig. 10 Uniform loop and normalized closed form

Instead of termination of the original loop on $\mathcal{S}^d$, we now have to prove termination of the transformed loop on $\widehat{\eta}(\mathcal{S}^d) = T \cdot \mathcal{S}^d$. For $\mathcal{S} \in \{\mathbb{Q}, \mathbb{R}_{\mathbb{A}}\}$, if the eigenvalues of A are from $\mathcal{S}$, then the transformation matrix T is an invertible matrix over $\mathcal{S}$. Therefore, we obtain $T \cdot \mathcal{S}^d = \mathcal{S}^d$. Hence, we now have to analyze termination of $(\varphi', Q \cdot \vec{x})$ over $\mathcal{S}$. In contrast, if $\mathcal{S} = \mathbb{Z}$, then even if the eigenvalues of A are integers, the transformation matrix T or its inverse may contain non-integer rational numbers. Thus, we first regard uniform loops over $\mathcal{S} \in \{\mathbb{Q}, \mathbb{R}_{\mathbb{A}}\}$ and discuss the case $\mathcal{S} = \mathbb{Z}$ in Sect. 7.3.5.

By uniformity, $Q = \operatorname{diag}(Q_{\lambda_1}, \ldots, Q_{\lambda_k})$ has k Jordan blocks Q_{λ_1}, ..., Q_{λ_k} where Q_λ is as in Fig. 8. For each eigenvalue λ, let $\nu(\lambda)$ be the dimension of Q_λ. Since each eigenvalue has geometric multiplicity 1, $\nu(\lambda)$ is the *algebraic multiplicity* of λ, i.e., the multiplicity as a root of the characteristic polynomial of A. For $\nu = \nu(\lambda)$, Fig. 9 shows the form of Q_λ^n as in, e.g., [35, 49], where $\binom{n}{s} = 0$ if $n < s$. This directly yields a closed form $\vec{q}$ for the n-fold application of the update $Q \cdot \vec{x}$ of the transformed loop. Since our approach from Sect. 4 works by analyzing *eventual* non-termination, we are only interested in validity of formulas for large enough n. Thus, we may assume that n is larger than the algebraic multiplicities $\nu(\lambda)$ of all eigenvalues $\lambda \in \operatorname{spec}(A)$. Then one obtains a resulting normalized closed form $\vec{q}_{norm}$ which consists of normalized poly-exponential expressions of a special form, so-called *hierarchical expressions*. Here, for any $\alpha \in \mathcal{Q}_{\mathcal{S}}[\vec{x}]$, $\deg_{x_i}(\alpha)$ is the highest power of x_i occurring in a monomial of α, i.e., it is the degree of α when interpreting all variables besides x_i as constants.

Definition 7.11 (Hierarchical Expression) Let $\mathcal{Q}_{\mathcal{S}}[\vec{x}]_{\text{lin}}$ denote the set of *linear polynomials* from $\mathcal{Q}_{\mathcal{S}}[\vec{x}]$, i.e., of degree at most 1. An expression $h \in \mathbb{NPE}_{\mathcal{S}}[\vec{x}]$ for some ring $\mathbb{Z} \leq \mathcal{S} \leq \mathbb{R}_{\mathbb{A}}$ is a *hierarchical expression* over the indices $1 \leq i_1 < \ldots < i_\nu \leq d$ if there exist $1 \leq r \leq \nu$ and $\lambda \in \mathcal{S}_{>0}$ such that

$$h = \sum\nolimits_{s=r}^{\nu} \alpha_s \cdot n^{s-r} \cdot \lambda^n,$$

where $\alpha_s \in \mathcal{Q}_{\mathcal{S}}[x_{i_s}, \ldots, x_{i_\nu}]_{\text{lin}}$, $\alpha_s(0, \ldots, 0) = 0$, and $\deg_{x_{i_s}}(\alpha_s) = 1$ for $r \leq s \leq \nu$. Here, $\alpha_s(v_s, \ldots, v_\nu)$ abbreviates $\alpha_s[x_{i_s}/v_s, \ldots, x_{i_\nu}/v_\nu]$ for $v_s, \ldots, v_\nu \in \mathbb{R}_{\mathbb{A}}$. We call $\operatorname{off}(h) = r$ the *offset*, ν the *order*, and $\operatorname{base}(h) = \lambda$ the *base* of h.

Lemma 7.12 states this observation on $\vec{q}_{norm}$ formally, where the *index* $\operatorname{idx}(\lambda)$ is the sum of the algebraic multiplicities of all smaller eigenvalues than λ, i.e.,

$$\operatorname{idx}(\lambda) = \sum\nolimits_{\lambda' \in \operatorname{spec}(A), \lambda' < \lambda} \nu(\lambda').$$

If A's smallest eigenvalue λ_1 is 0, then all entries $q_1, \ldots, q_{\operatorname{idx}(\lambda_2)}$ of $\vec{q}_{norm}$ are 0, i.e., when inserting $\vec{q}_{norm}$ into the loop condition, the variables $x_1, \ldots, x_{\operatorname{idx}(\lambda_2)}$ vanish. So from now on we assume that 0 is not an eigenvalue of A. (Note that since we just ignore the variables which belong to the eigenvalue zero, we can also permit uniform loops where the eigenvalue 0 may have a higher geometric multiplicity.)

Lemma 7.12 *For all* $\lambda \in \operatorname{spec}(A)$ *and* $1 \le r \le \nu(\lambda)$, *the* $(\operatorname{idx}(\lambda) + r)$*-th element of* $\vec{q}_{norm}$ *is a hierarchical expression over the indices* $\operatorname{idx}(\lambda) + 1, \operatorname{idx}(\lambda) + 2, \ldots, \operatorname{idx}(\lambda) + \nu(\lambda)$ *with offset* r, *order* $\nu(\lambda)$, *and base* λ.

Example 7.13 Consider the uniform loop and its normalized closed form $\vec{q}_{norm}$ in Fig. 10, where the update matrix A has $k = 2$ eigenvalues $\lambda_1 = 1$ of algebraic multiplicity $\nu_1 = 2$ and $\lambda_2 = 2$ of algebraic multiplicity $\nu_2 = 3$, both of which have geometric multiplicity one. Moreover, $\operatorname{idx}(\lambda_1) = 0$ and $\operatorname{idx}(\lambda_2) = 0 + \nu_1 = 2$. Here, we have $d = 5$.

For $\vec{q}_{norm}$ in Fig. 10, $h = q_4 = x_4 \cdot 2^n + \frac{x_5}{2} \cdot n \cdot 2^n$ is a hierarchical expression over the indices $i_1 = 3, i_2 = 4, i_3 = 5$ with offset 2, order 3, and base 2, as $h = \sum_{s=2}^{3} \alpha_s \cdot n^{s-2} \cdot 2^n$ for $\alpha_2 = x_4 \in \mathbb{R}_{\mathbb{A}}[x_4, x_5]$ and $\alpha_3 = \frac{x_5}{2} \in \mathbb{R}_{\mathbb{A}}[x_5]$.

To describe the form of the whole vector $\vec{q}_{norm}$, we now introduce *hierarchical partitions*. To this end, similar to the concept of solvable loops in Definition 2.3, we consider a partitioning of $\{1, \ldots, d\}$.

Definition 7.14 (Hierarchical Partition) For $k \ge 1$, let $\nu_1, \ldots, \nu_k \in \mathbb{N}$ form a *k-partition* of d, i.e., $\nu_1 + \ldots + \nu_k = d$ and $\nu_i > 0$ for all $1 \le i \le k$. The *blocks* associated to the partition $\nu_1, \ldots, \nu_k$ are $B_1 = \{1, \ldots, \nu_1\}$, $B_2 = \{\nu_1 + 1, \ldots, \nu_1 + \nu_2\}$, ..., and $B_k = \{\nu_1 + \ldots + \nu_{k-1} + 1, \ldots, d\}$.

The hierarchical expressions $h_1, \ldots, h_d$ are a *hierarchical k-partition* via $\nu_1, \ldots, \nu_k$ with bases $0 < \lambda_1 < \ldots < \lambda_k$ from $\mathcal{S}$ if for all $1 \le i \le k$:

(a) h_j is a hierarchical expression over the indices B_i for all $j \in B_i$,
(b) $\operatorname{base}(h_j) = \lambda_i$ for all $j \in B_i$,
(c) h_j has order ν_i for all $j \in B_i$,
(d) $\operatorname{off}(h_{\min(B_i)}) = 1$, and
(e) If $j, j + 1 \in B_i$, then $\operatorname{off}(h_j) + 1 = \operatorname{off}(h_{j+1})$.

Indeed, when transforming the update of a uniform loop to Jordan normal form, then the normalized closed form is always a hierarchical partition.

Corollary 7.15 ($\vec{q}_{norm}$ is Hierarchical Partition) *Let* A *be the update matrix of a uniform loop with eigenvalues* $0 < \lambda_1 < \ldots < \lambda_k$ *and algebraic multiplicities* $\nu_1, \ldots, \nu_k$, *and let* Q *be its Jordan normal form where the numbers on the diagonal are weakly monotonically increasing. Then the normalized closed form* $\vec{q}_{norm}$ *of the update* Q *is a hierarchical k-partition via* $\nu_1, \ldots, \nu_k$ *and bases* $\lambda_1 < \ldots < \lambda_k$. *Here the i-th block is* $B_i = \{\operatorname{idx}(\lambda_i) + 1, \ldots, \operatorname{idx}(\lambda_i) + \nu_i\}$.

Example 7.16 In Fig. 10, $\vec{h} = \vec{q}_{norm}$ is a hierarchical 2-partition via $\nu_1 = 2$ and $\nu_2 = 3$, blocks $B_1 = \{1, 2\}$, $B_2 = \{3, 4, 5\}$, and $\lambda_1 = 1 < 2 = \lambda_2$. Moreover, $\operatorname{off}(h_1) = 1$ and $\operatorname{off}(h_2) = 2$, while $\operatorname{off}(h_3) = 1$, $\operatorname{off}(h_4) = 2$, and $\operatorname{off}(h_5) = 3$.

7.3.3 Interval conditions

Our decision procedure in Sect. 4 instantiates the variables in the polynomials f of the loop guard by $\vec{q}_{norm} = (h_1, \dots, h_d)$, resulting in poly-exponential expressions $p = f(h_1, \dots, h_d)$. We now prove that in the setting of a hierarchical k-partition $\vec{q}_{norm}$, the atoms in red $(p \rhd 0)$ (see Lemma 4.7 and (7)) are equivalent to so-called *interval conditions*, whose satisfiability is particularly easy to check (Sect. 7.3.4). In Lemma 7.19, Lemma 7.21, and Corollary 7.23, we introduce the different subformulas occurring in red $(p \rhd 0)$. Here, it is convenient to define the *active variables* of polynomials.

Definition 7.17 (Active Variables) Let $f = c_0 + \sum_{i=1}^{d} c_i \cdot x_i \in \mathcal{S}[\vec{x}]_{\text{lin}}$. We define $\text{actVar}(f) = \left\{x_i \mid c_i \neq 0\right\}$. If $x_i \in \text{actVar}(f)$, then $\text{coeff}(f, x_i) = c_i$.

Example 7.18 Consider $f = -x_1 + 3 \cdot x_3 + 4 \in \mathcal{S}[x_1, \dots, x_5]_{\text{lin}}$. Then $\text{actVar}(f) = \{x_1, x_3\}$, $\text{coeff}(f, x_1) = -1$, and $\text{coeff}(f, x_3) = 3$.

From the addends $\alpha_j \cdot n^{a_j} \cdot b_j^n$ of $p = f(h_1, \dots, h_d)$ with $\alpha_j \in \mathcal{Q}_{\mathcal{S}}[\vec{x}]_{\text{lin}}$, $a_j \in \mathbb{N}$, and $b_j \in \mathcal{S}_{>0}$, we again compute the set $\text{coefs}(p)$ of marked coefficients as in Definition 4.5, which have the form $\alpha_j^{(b_j, a_j)}$. For any $b \in \mathcal{S}_{>0}$ and $a \in \mathbb{N}$, we now define a formula $\text{zero}(b, a)$ which is equivalent to requiring that all addends $\alpha_j \cdot n^{a_j} \cdot b_j^n$ vanish where $b_j = b$ and where $a_j \geq a$.

Lemma 7.19 (Formulas for Vanishing of Addends) *Let $h_1, \dots, h_d$ be a hierarchical k-partition via $v_1, \dots, v_k$ with bases $0 < \lambda_1 < \dots < \lambda_k$, and let $f \in \mathcal{S}[\vec{x}]_{\text{lin}}$. For any $1 \leq i \leq k$, let $F(i) = \{j \in B_i \mid x_j \in \text{actVar}(f)\}$. Let $M = \frac{-f(0,\dots,0)}{\text{coeff}(f, x_{\min F(i)}) \cdot c_{\min F(i)}}$ if $\lambda_i = 1$ and $F(i) \neq \emptyset$, where $c_{\min F(i)}$ is the coefficient of $x_{\min F(i)}$ in $h_{\min F(i)}$.*[2]

For any $b \in \mathcal{S}_{>0}$ and $a \in \mathbb{N}$, we define $\text{zero}(b, a)$:

(a) *If $\lambda_i \neq 1$ for all $1 \leq i \leq k$ or $\lambda_i = 1$ for some $1 \leq i \leq k$ and $F(i) = \emptyset$, then* $\text{zero}(1, 0)$ *is the formula $f(0, \dots, 0) = 0$.*
(b) *If $\lambda_i = 1$ for some $1 \leq i \leq k$ and $F(i) \neq \emptyset$, then* $\text{zero}(1, 0)$ *is the formula* $x_{\min F(i)} = M \wedge \bigwedge_{j \in B_i, \min F(i) < j} (x_j = 0)$.
(c) *If $\lambda_i = b$ for some $1 \leq i \leq k$, $F(i) \neq \emptyset$, and $(b, a) \neq (1, 0)$, then* $\text{zero}(b, a) = \bigwedge_{j \in B_i, a + \min F(i) \leq j} (x_j = 0)$.
(d) *Otherwise, we define* $\text{zero}(b, a) = \text{true}$.

Let $p = f(h_1, \dots, h_d) \in \mathbb{NPE}_{\mathcal{S}}[\vec{x}]$. As in Definition 4.5, let $\text{coefs}(p) = \{\alpha_1^{(b_1, a_1)}, \dots, \alpha_\ell^{(b_\ell, a_\ell)}\}$ where $\alpha_i^{(b_i, a_i)} \prec_{coef} \alpha_j^{(b_j, a_j)}$ for all $1 \leq i < j \leq \ell$.

Then $\text{zero}(b, a)$ *is equivalent to the requirement that $\alpha_s = 0$ holds for all $\alpha_s^{(b_s, a_s)} \in \text{coefs}(p)$ with $b_s = b$ and $a_s \geq a$.*

[2] By Definitions 7.14 and 7.11, $x_{\min F(i)}$ occurs only in a unique (linear) monomial of $h_{\min F(i)}$, whose coefficient is not 0.

Example 7.20 Consider $f = -x_1 + 3 \cdot x_3 + 4$ from Example 7.18 and $\vec{h} = \vec{q}_{norm}$ from Fig. 10.

We have $p = f(h_1, \dots, h_5) = -h_1 + 3 \cdot h_3 + 4 =$

$$(-x_1 + 4) - x_2 \cdot n + 3 \cdot x_3 \cdot 2^n + \left(\tfrac{3 \cdot x_4}{2} - \tfrac{3 \cdot x_5}{8}\right) \cdot n \cdot 2^n + \tfrac{3 \cdot x_5}{8} \cdot n^2 \cdot 2^n.$$

In the notation of Lemma 7.19, since $B_1 = \{1, 2\}$, $B_2 = \{3, 4, 5\}$, and $\mathrm{actVar}(f) = \{x_1, x_3\}$, we have $F(1) = \{j \in \{1, 2\} \mid x_j \in \{x_1, x_3\}\} = \{1\}$ and $F(2) = \{j \in \{3, 4, 5\} \mid x_j \in \{x_1, x_3\}\} = \{3\}$. Thus, $\min F(1) = 1$, $c_{\min F(1)} = 1$ is the coefficient of x_1 in h_1, $\mathrm{coeff}(f, x_{\min F(1)}) = -1$ and thus $M = \frac{-4}{-1} = 4$. Hence,

$$\mathrm{zero}(1,0) = (x_1 = 4) \wedge (x_2 = 0) \text{ and } \mathrm{zero}(2,0) = (x_3 = 0) \wedge (x_4 = 0) \wedge (x_5 = 0).$$

So $\vec{v} \in \mathbb{R}^5_{\mathbb{A}}$ satisfies $\mathrm{zero}(1,0)$ or $\mathrm{zero}(2,0)$, respectively, iff all terms with base 1 or 2 vanish in $f(h_1, \dots, h_5)[\vec{x}/\vec{v}]$.

After introducing $\mathrm{zero}(b, a)$, we are now ready to show that the formulas in Lemma 4.7 and Theorem 4.9 have a special form when considering uniform loops. In the following lemmas, corollaries, and definitions, let $h_1, \dots, h_d$, $v_1, \dots, v_k$, f, $F(i)$, p, $\mathrm{coefs}(p)$, and ℓ always be as in Lemma 7.19.

Lemma 7.21 *For any $1 \le s_0 \le \ell$, the formula $\alpha_{s_0} > 0 \wedge \bigwedge_{s=s_0+1}^{\ell} (\alpha_s = 0)$ is equivalent to the following formula ρ_{f,s_0}, which can be computed in polynomial time from $h_1, \dots, h_d$ and f:*

(a) *If $(b_{s_0}, a_{s_0}) = (1, 0)$ and either $\lambda_i \ne 1$ for all $1 \le i \le k$ or $\lambda_{i_0} = 1$ for some $1 \le i_0 \le k$ and $F(i_0) = \emptyset$, then ρ_{f,s_0} is*

$$f(0, \dots, 0) > 0 \wedge \bigwedge\nolimits_{i \in \{1,\dots,k\}, \lambda_i > 1} \mathrm{zero}(\lambda_i, 0).$$

(b) *If $(b_{s_0}, a_{s_0}) = (1, 0)$, $\lambda_{i_0} = 1$ for some $1 \le i_0 \le k$, and $F(i_0) \ne \emptyset$, then there is a*[3] *$C \in \mathcal{Q}_{\mathcal{S}}$ with $C \ne 0$ such that ρ_{f,s_0} is*

$$\mathrm{sign}(C) \cdot x_{\min F(i_0)} + \tfrac{f(0,\dots,0)}{|C|} > 0 \ \wedge \ \mathrm{zero}(1,1) \wedge \bigwedge\nolimits_{i=i_0+1}^{k} \mathrm{zero}(\lambda_i, 0).$$

(c) *If $b_{s_0} < 1$, $f(0, \dots, 0) \ne 0$, and either $\lambda_i \ne 1$ for all $1 \le i \le k$ or $\lambda_{i_0} = 1$ for some $1 \le i_0 \le k$ and $F(i_0) = \emptyset$, then ρ_{f,s_0} is* false.

(d) *Otherwise, we have $\lambda_{i_0} = b_{s_0}$ for some $1 \le i_0 \le k$, $F(i_0) \ne \emptyset$, and there is a number*[4] *$sg \in \{1, -1\}$ such that ρ_{f,s_0} is*

$$sg \cdot x_{\min F(i_0) + a_{s_0}} > 0 \wedge \mathrm{zero}(\lambda_{i_0}, a_{s_0} + 1) \wedge \bigwedge\nolimits_{i=i_0+1}^{k} \mathrm{zero}(\lambda_i, 0).$$

Example 7.22 For $f, \vec{h}, p$ of Example 7.20, $\mathrm{coefs}(p)$ is

[3] $C = \mathrm{coeff}(f, x_{\min F(i_0)}) \cdot c_{\min F(i_0)}$ for the coefficient $c_{\min F(i_0)}$ of $x_{\min F(i_0)}$ in $h_{\min F(i_0)}$.

[4] More precisely, $sg = \mathrm{sign}(\mathrm{coeff}(f, x_{\min F(i_0)}) \cdot c_{\min F(i_0)+a_{s_0}})$, where $c_{\min F(i_0)+a_{s_0}}$ is the unique coefficient of $x_{\min F(i_0)+a_{s_0}}$ in $h_{\min F(i_0)}$'s addend of the form $\beta \cdot n^{a_{s_0}} \cdot b_{s_0}^n$ with $\beta \in \mathcal{Q}_{\mathcal{S}}[x_{\min F(i_0)+a_{s_0}}, \dots, x_{v_1+\dots+v_{i_0}}]_{\mathrm{lin}}$.

$$\{\alpha_1^{(1,0)}, \alpha_2^{(1,1)}, \alpha_3^{(2,0)}, \alpha_4^{(2,1)}, \alpha_5^{(2,2)}\} \qquad \text{where}$$
$$\alpha_1 = -x_1 + 4, \quad \alpha_2 = -x_2, \quad \alpha_3 = 3 \cdot x_3, \quad \alpha_4 = \tfrac{3 \cdot x_4}{2} - \tfrac{3 \cdot x_5}{8}, \quad \alpha_5 = \tfrac{3 \cdot x_5}{8}.$$

Let us compute $\rho_{f,4}$. For $\alpha_4^{(2,1)}$, we have $i_0 = 2$, $\min F(i_0) = 3$, and $a_4 = 1$. Moreover, we have $sg = \operatorname{sign}\left(\operatorname{coeff}(f, x_3) \cdot c_4\right) = \operatorname{sign}\left(3 \cdot \frac{1}{2}\right) = 1$ and $\operatorname{zero}(2,2) = \bigwedge_{j \in \{3,4,5\}, 2+3 \le j} (x_j = 0) = (x_5 = 0)$ and hence by Lemma 7.21:

$$\begin{aligned}(\alpha_4 > 0) \wedge (\alpha_5 = 0) \quad &\iff \quad (sg \cdot x_{a_4 + \min F(i_0)} > 0) \wedge \operatorname{zero}(2, a_4 + 1) \\ &\iff \quad (x_4 > 0) \wedge (x_5 = 0)\end{aligned}$$

For $\alpha_1^{(1,0)}$, as $\lambda_1 = 1$ and $F(1) = \{1\} \neq \emptyset$, by Lemma 7.21 we use $C = \operatorname{coeff}(f, x_1) \cdot 1 = -1$, $f(0, \ldots, 0) = 4$, and $\operatorname{zero}(1,1) = \bigwedge_{j \in \{1,2\}, 1+1 \le j} (x_j = 0) = (x_2 = 0)$ to obtain $\rho_{f,1}$, where $\operatorname{zero}(2,0) = \bigwedge_{j=3}^{5} (x_j = 0)$ by Example 7.20:

$$(\alpha_1 > 0) \wedge \bigwedge_{s=2}^{5} (\alpha_s = 0) \iff (-x_1 + 4 > 0) \wedge (x_2 = 0) \wedge \operatorname{zero}(2,0)$$

In addition to the formulas $\rho_{f,s}$ for $1 \le s \le \ell$, we also introduce a formula $\rho_{f,0}$ which expresses that all coefficients of p vanish.

Corollary 7.23 $\bigwedge_{s=1}^{\ell} \left(\alpha_s = 0\right)$ *is equivalent to* $\rho_{f,0}$: $\operatorname{zero}(1,0) \wedge \bigwedge_{\lambda \in \{1,\ldots,k\}, \lambda \neq 1} \operatorname{zero}(\lambda, 0)$

Example 7.24 Reconsider Example 7.22. By Corollary 7.23, $\bigwedge_{s=1}^{5} (\alpha_s = 0)$ is equivalent to $\rho_{f,0} = \operatorname{zero}(1,0) \wedge \operatorname{zero}(2,0)$, where $\operatorname{zero}(1,0) = (x_1 = 4) \wedge (x_2 = 0)$ and $\operatorname{zero}(2,0) = (x_3 = 0) \wedge (x_4 = 0) \wedge (x_5 = 0)$ by Example 7.20.

We can now combine Lemma 7.21 and Corollary 7.23 with Lemma 4.7 to obtain the following result. Here, "ic" stands for interval conditions.

Corollary 7.25 *For* $\rhd \in \{\ge, >\}$, $\operatorname{red}(p \rhd 0)$ *is equivalent to the formula* $\operatorname{ic}(p \rhd 0)$, *where* $\operatorname{ic}(p > 0) = \bigvee_{s=1}^{\ell} \rho_{f,s}$ *and* $\operatorname{ic}(p \ge 0) = \operatorname{ic}(p > 0) \vee \rho_{f,0}$.

Example 7.26 Reconsider Examples 7.22 and 7.24. By Corollary 7.25, $\operatorname{red}(p > 0)$ is equivalent to $\operatorname{ic}(p > 0) = \rho_{f,1} \vee \rho_{f,2} \vee \rho_{f,3} \vee \rho_{f,4} \vee \rho_{f,5} =$

$$\begin{aligned}&\left(-x_1 + 4 > 0 \wedge x_2 = 0 \wedge \operatorname{zero}(2,0)\right) \vee \left(-x_2 > 0 \wedge \operatorname{zero}(2,0)\right) \\ \vee &\left(x_3 > 0 \wedge x_4 = 0 \wedge x_5 = 0\right) \qquad \vee (x_4 > 0 \wedge x_5 = 0) \qquad \vee (x_5 > 0), \text{and}\end{aligned}$$

$\operatorname{red}(p \ge 0)$ is equivalent to $\operatorname{ic}(p \ge 0) = \operatorname{ic}(p > 0) \vee (\operatorname{zero}(1,0) \wedge \operatorname{zero}(2,0))$.

The formulas $\rho_{f,s}$ in Lemma 7.21 and Corollary 7.23 are so-called *interval conditions*.

Definition 7.27 (Interval Condition) For $1 \le i, i' \le d$, $i \neq i'$, $I \subseteq \{1, \ldots, d\}$, $sg \in \{-1, 1\}$, and $0 \neq c \in \mathcal{Q}_{\mathcal{S}}$, an *interval condition* has the following forms:

$$\begin{array}{llll}
\text{(a)} & & & \bigwedge_{j\in I} \quad (x_j = 0)\\
\text{(b)} & sg\cdot x_i & > 0 \ \wedge & \bigwedge_{j\in I\setminus\{i\}} \quad (x_j = 0)\\
\text{(c)} & & x_{i'} = c \ \wedge & \bigwedge_{j\in I\setminus\{i'\}} \quad (x_j = 0)\\
\text{(d)} & sg\cdot x_i & > 0 \ \wedge \ x_{i'} = c \ \wedge & \bigwedge_{j\in I\setminus\{i,i'\}} \quad (x_j = 0)\\
\text{(e)} & sg\cdot x_i + c & > 0 \ \wedge & \bigwedge_{j\in I\setminus\{i\}} \quad (x_j = 0)
\end{array}$$

Example 7.28 The formulas $\rho_{f,4} = (x_4 > 0) \wedge (x_5 = 0)$ and $\rho_{f,1} = (-x_1 + 4 > 0) \wedge \bigwedge_{j=2}^{5} (x_j = 0)$ from Example 7.22 are interval conditions as in Definition 7.27 (b) and (e).

7.3.4 Checking satisfiability of interval conditions

We now show that to decide satisfiability of the formulas ic $(p \rhd 0)$, we only have to regard instantiations of the variables with values from $\{0, 1, -1, \star\}$, where $\star$ stands for one additional non-zero value. There are only polynomially many such instantiations and the particular value for $\star$ is later determined by SMT solving. This SMT solving only takes polynomial time, because the resulting SMT problem only contains a single variable. Definition 7.29 instantiates variables accordingly and performs Boolean simplifications as much as possible.

Definition 7.29 (Evaluation) Let ρ be a propositional formula built from the connectives $\wedge$ and $\vee$ over atoms of the form $sg \cdot x + c > 0$ and $x = c$ for $sg \in \{1, -1\}$, $c \in \mathcal{Q}_{\mathcal{S}}$, and $x \in \{x_1, \ldots, x_d\}$. Moreover, let $\vec{v} \in \{0, 1, -1, \star\}^d$. The *evaluation of* ρ *w.r.t.* $\vec{v}$ (written $\rho(\vec{v})\!\downarrow$) results from $\rho(\vec{v}) = \rho[\vec{x}/\vec{v}]$ by simplifying (in)equations without $\star$ to true or false, and by simplifying conjunctions and disjunctions with true resp. false. We write $\vec{v} \models^? \rho$ if $\rho(\vec{v})\!\downarrow \neq$ false.

For example, if ρ is the formula $(x_1 - \frac{5}{2} > 0) \wedge (x_2 = 0)$ and $\vec{v} = (\star, 0)$, then $\rho(\vec{v})\!\downarrow$ is $\star - \frac{5}{2} > 0$. Hence, $\vec{v} \models^? \rho$. So in general, $\vec{v} \models^? \rho$ means that $\rho(\vec{v})\!\downarrow =$ true or that there *could* be a value w for $\star$ such that $\rho[\vec{x}/\vec{v}, \star/w]\!\downarrow =$ true.

Now we define candidate assignments cndAssg $(\rho_{f,s})$ for the formulas $\rho_{f,s}$ in Lemma 7.21 and Corollary 7.23 which contain all $\vec{v} \in \{0, 1, -1, \star\}^d$ that *may* satisfy $\rho_{f,s}$ (if a suitable value for $\star$ is found). However, for each Block B_i, *at most one* variable x_j with $j \in B_i$ may be assigned a non-zero value (i.e., $1, -1$, or $\star$). Moreover, the value $\star$ may only be used in the block for the eigenvalue $\lambda_i = 1$.

Definition 7.30 (Sets of Candidate Assignments) For all $0 \le s \le \ell$, we define:

$$\begin{aligned}
\text{cndAssg}\,(\rho_{f,s}) \;=\; \{\vec{v} \in \{0, 1, -1, \star\}^d \mid\; & \vec{v} \models^? \rho_{f,s},\\
& \forall\, 1 \le i \le k.\ \text{there is at most one } j \in B_i \text{ with } v_j \neq 0,\\
& v_j = \star \implies j \in B_{i_0} \text{ where } \lambda_{i_0} = 1\}
\end{aligned}$$

Example 7.31 In Examples 7.22 and 7.24, for $\rho_{f,4} = (x_4 > 0) \wedge (x_5 = 0)$, $\vec{v} \models^? \rho_{f,4}$ implies $v_4 = 1$ and $v_5 = 0$. Here, $v_4 = \star$ is not possible, because 4 does not belong to the block $B_1 = \{1, 2\}$ for the eigenvalue 1. Since at most one value for each block may be non-zero,

Input: a formula φ over the atoms $\{f \triangleright 0 \mid f \in \mathcal{S}[\vec{x}]_{\text{lin}}, \triangleright \in \{>, \geq\}\}$,
a hierarchical k-partition $\vec{h} = (h_1, \ldots, h_d)$, and a ring $\mathcal{S} \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}_{\mathbb{A}}\}$
Result: $\top$ if $\exists \vec{x} \in \mathcal{S}^d.\ \text{ic}(\varphi(\vec{h}))$ is valid, $\bot$ otherwise
$\psi \leftarrow \text{ic}(\varphi(\vec{h}))$
foreach $\vec{v} \in \text{cndAssg}(\psi)$ **do**
  $\psi' \leftarrow \psi(\vec{v})\downarrow$
  if $\text{SMT}((\psi' \wedge \star \neq 0), \{\star\}, \mathcal{S})$ **then return** $\top$
return $\bot$

Algorithm 4 Checking Interval Conditions

we have $v_3 = 0$. In contrast, v_1 and v_2 can be arbitrary, but at most one of them may be non-zero. Hence, we obtain the following for $\rho_{f,4}$ and for $\rho_{f,0} = (x_1 = 4) \wedge \bigwedge_{j=2}^{5}(x_j = 0)$:

$$\text{cndAssg}\,(\rho_{f,4}) = \left\{ \begin{bmatrix}1\\0\\0\\1\\0\end{bmatrix}, \begin{bmatrix}\star\\0\\0\\1\\0\end{bmatrix}, \begin{bmatrix}-1\\0\\0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\1\\0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\\star\\0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\-1\\0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\0\\0\\1\\0\end{bmatrix} \right\}, \quad \text{cndAssg}\,(\rho_{f,0}) = \left\{ \begin{bmatrix}\star\\0\\0\\0\\0\end{bmatrix} \right\}$$

Lemma 7.32 $|\,\text{cndAssg}\,(\rho_{f,s})| \leq \left(3\cdot\max\{v_i \mid 1 \leq i \leq k\} + 1\right)^k$ *for all* $0 \leq s \leq \ell$.

Lemma 7.32 is crucial for our algorithm to decide k-termination for uniform loops: $|\,\text{cndAssg}\,(\rho_{f,s})|$ is bounded by a polynomial in d if k is assumed to be a parameter. This is because the v_i form a k-partition of d, i.e., $v_i \leq d$ for $1 \leq i \leq k$. Hence, computing $\text{cndAssg}\,(\rho_{f,s})$ can be done in polynomial time for fixed k.

Example 7.33 In Example 7.31, we have $v_1 = 2$ and $v_2 = 3$, and thus, $k = 2$ and $\max\{v_i \mid 1 \leq i \leq k\} = 3$. Here, $|\,\text{cndAssg}\,(\rho_{f,4})| = 7 \leq 100 = 10^2 = (3 \cdot 3 + 1)^2$.

This example shows that the bound in Lemma 7.32 is coarse, but it suffices for our analysis. We now combine Corollary 7.25 and Definition 7.30 to obtain the sets of candidate assignments for the disjunctions $\text{ic}\,(p > 0)$ and $\text{ic}\,(p \geq 0)$.

Corollary 7.34 *We lift* cndAssg *to inequations by defining*

$$\begin{aligned} \text{cndAssg}\,(\text{ic}\,(p > 0)) &= \textstyle\bigcup_{s=1}^{\ell} \text{cndAssg}\,(\rho_{f,s}) \\ \text{and}\quad \text{cndAssg}\,(\text{ic}\,(p \geq 0)) &= \text{cndAssg}\,(\text{ic}\,(p > 0)) \cup \text{cndAssg}\,(\rho_{f,0}). \end{aligned}$$

Then we have

$$|\,\text{cndAssg}\,(\text{ic}\,(p \triangleright 0))| \leq (d+2) \cdot \left(3 \cdot \max\{v_i \mid 1 \leq i \leq k\} + 1\right)^k.$$

For a uniform loop with condition φ and normalized closed form $\vec{q}_{norm} = \vec{h}$, let $\varphi(\vec{h})$ contain the atoms $f(\vec{h}) \triangleright 0$, where $f \in \mathcal{S}[\vec{x}]_{\text{lin}}$. To decide termination, our algorithm computes $\text{cndAssg}\,(\text{ic}\,(f(\vec{h}) \triangleright 0))$ for all these atoms $f(\vec{h}) \triangleright 0$, and then checks for each of the candidate assignments whether it is a witness for eventual non-termination. We first lift ic and cndAssg to linear formulas.

Definition 7.35 (ic and cndAssg for Linear Formulas) Let φ be a linear formula over the atoms $\{f \rhd 0 \mid f \in \mathcal{S}[\vec{x}]_{\text{lin}}, \rhd \in \{>, \geq\}\}$ and let $\vec{h} = (h_1, \ldots, h_d)$ be a hierarchical k-partition. Then the formula $\text{ic}\,(\varphi(\vec{h}))$ results from replacing each atom $f(\vec{h}) \rhd 0$ in $\varphi(\vec{h}) = \varphi[\vec{x}/\vec{h}]$ by $\text{ic}\,(f(\vec{h}) \rhd 0)$. By $\text{cndAssg}\,(\,\text{ic}\,(\varphi(\vec{h})))$ we denote the set $\bigcup_{f(\vec{h}) \rhd 0 \text{ atom in } \varphi(\vec{h})} \text{cndAssg}\,(\,\text{ic}\,(f(\vec{h}) \rhd 0))$.

To analyze termination of uniform loops, we now present an algorithm to decide whether for a hierarchical k-partition $\vec{h}$ and a linear formula φ,

$$\exists \vec{x} \in \mathcal{S}^d, n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \varphi(\vec{h}) \qquad \text{(see (9))}$$

is valid. Our algorithm calls a method SMT(ψ,$\mathcal{V}$,$\mathcal{S}$) which checks whether the linear formula ψ in the variables $\mathcal{V}$ is satisfiable. Here, the variables $\mathcal{V}$ range over $\mathcal{S} \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}_{\mathbb{A}}\}$ and the coefficients of the polynomials are from $\mathcal{Q}_{\mathcal{S}}$. (So for $\mathcal{S} = \mathbb{Z}$, one can first multiply all inequations in ψ by the main denominator to result in coefficients from $\mathbb{Z}$.) In our case, $\mathcal{V} = \{\,\star\,\}$ and thus, $|\mathcal{V}| = 1$. With these restrictions, the method SMT has *polynomial* runtime (see [2, 41]). More precisely, SMT is called in Algorithm 4 to determine whether $\star$ can be assigned a non-zero value such that $\psi(\vec{v})\!\downarrow$ is satisfiable. Here, we have to assign *all* occurrences of $\star$ in the formula $\psi(\vec{v})\!\downarrow$ the same value.

Let us discuss the complexity of Algorithm 4. The formula $\text{ic}\,(\varphi(\vec{h}))$ and each element of $\text{cndAssg}\,(\,\text{ic}\,(\varphi(\vec{h})))$ can be computed in polynomial time. By Corollary 7.34, $\text{cndAssg}\,(\,\text{ic}\,(\varphi(\vec{h})))$ has at most $|\varphi| \cdot (d+2) \cdot \left(3 \cdot \max\{v_i \mid 1 \leq i \leq k\} + 1\right)^k$ elements, where $|\varphi|$ is the number of atoms in φ and $v_i \leq d$ for all $1 \leq i \leq k$. Thus, when considering k to be a parameter, $\text{cndAssg}\,(\,\text{ic}\,(\varphi(\vec{h})))$ can be computed in polynomial time. Moreover, evaluating a formula w.r.t. $\vec{v}$ according to Definition 7.29 is possible in polynomial time, too. Finally, SMT has polynomial runtime as discussed before. So the runtime of the algorithm is polynomial when regarding k as a parameter. We now prove that Algorithm 4 is sound and complete.

Theorem 7.36 *Algorithm* 4 *returns* $\top$ *iff* $\exists \vec{x} \in \mathcal{S}^d.\ \text{ic}\,(\varphi(\vec{h}))$ *is valid.*

Example 7.37 Consider the uniform loop in Fig. 10 where $\varphi = f > 0 \wedge f' > 0$ for $f = -x_1 + 3 \cdot x_3 + 4$ and $f' = 2 \cdot x_1 - 5$. Let $\vec{h} = \vec{q}_{norm}$ as in Fig. 10 and let $p = f(\vec{h})$ and $p' = f'(\vec{h}) = (2 \cdot x_1 - 5) + 2 \cdot x_2 \cdot n$. Here, $\psi = \text{ic}\,(\varphi(\vec{h})) = \text{ic}\,(p > 0) \wedge \text{ic}\,(p' > 0)$, where $\text{ic}\,(p > 0) = \bigvee_{s=1}^{5} \rho_{f,s}$ is stated in Example 7.26. Note that $\text{coefs}\,(p') = \{\alpha_1'^{(1,0)}, \alpha_2'^{(1,1)}\}$ with $\alpha_1' = 2 \cdot x_1 - 5$ and $\alpha_2' = 2 \cdot x_2$. Hence, $\text{ic}\,(p' > 0) = \rho_{f',1} \vee \rho_{f',2}$. To compute $\rho_{f',1}$, for $C = \text{coeff}\,(f', x_1) \cdot c_1 = 2 \cdot 1 = 2$ and $f'(0, \ldots, 0) = -5$, Lemma 7.21 (b) results in $\rho_{f',1} = (x_1 - \frac{5}{2} > 0) \wedge (x_2 = 0)$. For $\rho_{f',2}$, with $sg = \text{sign}\left(\text{coeff}\,(f', x_1) \cdot c_2\right) = \text{sign}\,(2 \cdot 1) = 1$, Lemma 7.21 (d) results in $\rho_{f',2} = (x_2 > 0)$. Now from $\text{ic}\,(p > 0)$ let us choose the disjunct $\rho_{f,1} = (-x_1 + 4 > 0) \wedge \bigwedge_{j=2}^{5} (x_j = 0)$ and from $\text{ic}\,(p' > 0)$ let us choose the disjunct $\rho_{f',1} = (x_1 - \frac{5}{2} > 0) \wedge (x_2 = 0)$. We consider $\vec{v} = (\,\star\,, 0, 0, 0, 0)$. Then

$$(\rho_{f,1} \wedge \rho_{f',1})(\vec{v})\downarrow \quad = \quad (-\star\ + 4 > 0)\ \wedge\ (\star - \tfrac{5}{2} > 0)$$

is satisfiable with the model $\star\ = 3$. Hence, this model also satisfies $\psi(\vec{v})\downarrow \wedge\ \star\ \neq 0$. Thus, for both $\mathcal{S} \in \{\mathbb{Q}, \mathbb{R}_{\mathbb{A}}\}$, Algorithm 4 proves validity of $\exists \vec{x} \in \mathcal{S}^d.\ \mathrm{ic}\,(\varphi(\vec{h}))$ and therefore, non-termination of the uniform loop over $\mathcal{S}$.

So for a uniform loop over $\mathcal{S} \in \{\mathbb{Q}, \mathbb{R}_{\mathbb{A}}\}$, non-termination is equivalent to validity of $\exists \vec{x} \in \mathcal{S}^d.\ \forall n \in \mathbb{N}_{>n_0}.\ \varphi(\vec{q}_{norm})$, which in turn is equivalent to a formula only containing interval conditions. This insight reduces the search space for proving validity drastically. Thus, we can now prove Theorem 7.10 for $\mathcal{S} \in \{\mathbb{Q}, \mathbb{R}_{\mathbb{A}}, \mathbb{R}\}$.

Proof of Theorem 7.10 For $\mathcal{S} \in \{\mathbb{Q}, \mathbb{R}_{\mathbb{A}}\}$, we first transform the uniform loop such that the update matrix is in Jordan normal form and then compute the normalized closed form as in Lemma 7.12 in polynomial time. This closed form is a hierarchical partition by Corollary 7.15. By combining Corollary 7.25 and Theorem 7.36, Algorithm 4 can decide validity of the formula from Theorem 4.9, i.e., termination of the transformed loop (which is equivalent to termination of the original loop by Corollary 5.16).

As the computation of the equivalent interval conditions in Corollary 7.25 clearly works in polynomial time and we have discussed that Algorithm 4 runs in polynomial time when k is assumed to be a parameter, this proves the statement.

Finally, these loops terminate over $\mathbb{R}_{\mathbb{A}}$ iff they terminate over $\mathbb{R}$ by Corollary 4.11.

□

For Theorem 7.10, it was crucial to *transform* the loop such that the update matrix is in Jordan normal form. Here we relied on a special closed form for the Jordan normal form, while in Sect. 7.1 we only used the transformation to argue why the closed form is computable in polynomial time. Thus, the transformation from Sect. 5 does not only generalize our results from Sect. 4 to a wider class of loops but it also gives rise to novel results like Theorem 7.10.

The approach in the proof of Theorem 7.10 also works for uniform loops over $\mathbb{Z}$ if the update matrix is already in Jordan normal form. But otherwise, in addition to $\varphi(\vec{q}_{norm})$ we also have an update-invariant and $\mathrm{Th}_{\exists}(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$-definable subset F which stems from the transformation into Jordan normal form (see Sect. 5). Thus, to decide termination we have to decide validity of $\exists \vec{x} \in \mathbb{R}_{\mathbb{A}}^d.\ \forall n \in \mathbb{N}_{>n_0}.\ \varphi(\vec{q}_{norm}) \wedge \psi_F$. We will discuss this in the next section.

7.3.5 Termination of uniform loops over the integers

Now we show that deciding termination of uniform loops $(\varphi, A \cdot \vec{x})$ over the *integers* is also in XP. Let $A \in \mathbb{Z}^{d\times d}$ with k integer eigenvalues each of geometric multiplicity one. Then there is a matrix $T \in \mathbb{Q}^{d\times d}$ such that $A = T^{-1} \cdot Q \cdot T$ for a matrix Q in Jordan normal form. However, in general we do not have $T, T^{-1} \in \mathbb{Z}^{d\times d}$ (see, e.g., [59]). As before, let $\eta(\vec{x}) = T \cdot \vec{x}$ and $\varphi' = \eta^{-1}(\varphi)$. Then termination of $(\varphi, A \cdot \vec{x})$ on $\mathbb{Z}^d$ is equivalent to termination of $(\varphi', Q \cdot \vec{x})$ on $\hat{\eta}(\mathbb{Z}^d) = T \cdot \mathbb{Z}^d$ by Corollary 5.16. Here, $\mathcal{L}_T = T \cdot \mathbb{Z}^d$ is the set of all *integer* linear combinations of T's columns, i.e., their *lattice*. In general, we have $\mathcal{L}_T \neq \mathbb{Z}^d$.

Hence, we now call Algorithm 4 with the input $(\varphi', \vec{h}, \mathbb{Q})$, where $\vec{h} = \vec{q}_{norm}$ is the normalized closed form for the update $Q \cdot \vec{x}$. For $\psi = \text{ic}\,(\varphi'(\vec{h}))$, we want to find out if $\psi(\vec{v})$ holds for some $\vec{v} \in \mathcal{L}_T$. Such a $\vec{v} \in \mathcal{L}_T$ would witness eventual non-termination of $(\varphi', Q \cdot \vec{x})$, and since $\mathcal{L}_T$ is update-invariant under $Q \cdot \vec{x}$ by Lemma 5.12, this is equivalent to non-termination of $(\varphi', Q \cdot \vec{x})$ on $\mathcal{L}_T$.

We modify Algorithm 4 such that it computes *all* $\vec{v} \in \text{cndAssg}\,(\psi)$ where $\psi(\vec{v})\!\downarrow \wedge \star \neq 0$ is satisfied by some $\vec{v}'$ that results from $\vec{v}$ by instantiating $\star$ with a suitable number from $\mathbb{Q}$. As shown in Corollary 7.34, for a fixed number of eigenvalues k, there are only polynomially many such candidate assignments $\vec{v}$. Note that the formula φ' is only built from the connectives $\wedge$ and $\vee$, and ψ results from φ' by replacing each atom $f \rhd 0$ by $\text{ic}\,(f(\vec{h}) \rhd 0)$. Hence, for every such $\vec{v}$ there is a subset $\{\, \text{ic}\,(f_1(\vec{h}) \rhd_1 0), \dots, \text{ic}\,(f_e(\vec{h}) \rhd_e 0)\}$ of these formulas such that $\vec{v}'$ satisfies them all and such that satisfying these formulas is sufficient for satisfying ψ. For each $1 \leq r \leq e$, let $\ell_r = |\,\text{coefs}\,(f_r(\vec{h}))|$. By Corollary 7.25, $\text{ic}\,(f_r(\vec{h}) \rhd_r 0)$ has the form $\bigvee_{s=1}^{\ell_r} \rho_{f_r,s}$ or $\bigvee_{s=0}^{\ell_r} \rho_{f_r,s}$. So for every r there is *at least* one s where $\rho_{f_r,s}(\vec{v}')$ is true. But due to the construction of $\rho_{f_r,s}$ in Lemma 7.21 and Corollary 7.23, there is *at most* one $0 \leq s \leq \ell_r$ where $\rho_{f_r,s}(\vec{v}')$ is true. Thus, for every $1 \leq r \leq e$, there is a *unique* $0 \leq s_r \leq \ell_r$ where $\rho_{f_r,s_r}(\vec{v}')$ is true.

By Lemma 7.21 and Corollary 7.23, all ρ_{f_r,s_r} are interval conditions. Thus, for each entry v_j of $\vec{v}'$ we can find out whether $x_j = v_j$ is required by some ρ_{f_r,s_r}, or whether $v_j = 0$ is just due to setting variables to zero by default, i.e., the formula would still hold when assigning an arbitrary value from $\mathbb{Q}$ to v_j. So every $\vec{v} \in \text{cndAssg}\,(\psi)$ gives rise to a certain set of formulas $\{\rho_{f_1,s_1}, \dots, \rho_{f_e,s_e}\}$, which in turn results in a certain *abstract assignment* that indicates for each entry of $\vec{v}'$ whether its actual value is necessary to be a model.

Definition 7.38 (Abstract Assignment) Let $\mathbb{I}$ be the set of all intervals of the forms $[c, c]$, $(-\infty, c)$, (c, ∞), $(-\infty, \infty)$, or (c, d) for $c, d \in \mathbb{Q}$ with $c \leq d$. Then an *abstract assignment* is an element of $\mathbb{I}^d$.

For each of the obtained abstract assignments, we now have to check whether it is satisfied by some value from $\mathcal{L}_T$. Let $N \subseteq \{1, \dots, d\}$ be those indices where $j \in N$ iff the j-th component of the abstract assignment is $[0, 0]$, i.e., iff the j-th component must be 0 in order to satisfy all ρ_{f_s,m_s}. Then we compute a basis of the sublattice $\mathcal{L}_N = \{\vec{w} \in \mathcal{L}_T \mid w_j = 0 \text{ for } j \in N\}$. To this end, we solve the system of linear equations $\bigwedge_{j \in N} (T \cdot \vec{x})_j = 0$ where $\vec{x} \in \mathbb{Z}^d$. Here as usual, $(T \cdot \vec{x})_j$ denotes the j-th component of the vector $T \cdot \vec{x}$. This problem can be solved in polynomial time (see, e.g., [22, 60]). Since in general this system contains more variables than equations, the solutions yield a certain linear dependence between the variables. This dependence can then be used to reduce the number of variables in the system, i.e., it gives rise to a basis of $\mathcal{L}_N$, where each basis vector is represented by a $\mathbb{Z}$-linear combination of the columns of T. Let $d' \leq d$ be the rank of the sublattice $\mathcal{L}_N$ (i.e., the number of its basis vectors) and let $P \in \mathbb{Q}^{d \times d'}$ be the matrix whose columns form the basis of $\mathcal{L}_N$.

Let $N' \subseteq \{1, \dots, d\}$ be those indices where $j \in N'$ iff the j-th component of the abstract assignment is neither $[0, 0]$ nor $(-\infty, \infty)$. So the j-th component must be from a certain interval in order to satisfy all ρ_{f_s,m_s}. To ease notation, we define $I_j = (-\infty, \infty)$ if $j \notin N'$ and let $K = \prod_{j=1}^{d} I_j \subseteq \mathbb{Q}^d$. Now we have to decide whether there exists an $X \in \mathbb{Z}^{d'}$ such that $P \cdot X \in K$.

Let $B_1, \ldots, B_k$ again be the blocks from the k-partition $h_1, \ldots, h_d$. Note that if there is a block B_i where some ρ_{f_r,s_r} requires x_j with $j \in B_i$ to be non-zero (i.e., $j \in N'$), then ρ_{f_r,s_r} requires all $x_{j'}$ with $j' > j$ and $j' \in B_i$ to be zero (i.e., $j' \in N$). Thus, since $\vec{v}'$ satisfies *all* formulas ρ_{f_r,s_r} for $1 \leq r \leq e$, for each block B_i there can be *at most* one $r \in B_i$ where some ρ_{f_r,s_r} requires x_j to be non-zero. Hence, for each block B_i there is at most one $j \in B_i$ where $j \in N'$. Note that containment in an interval can be described by at most 2 inequations, where the strict inequations can be turned into weak ones since the variables only range over the integers. Thus, to describe the required containment in the intervals for all x_j with $j \in N'$, we need at most $2 \cdot k$ inequations. In other words, the requirement $P \cdot X \in K$ can be described by $2 \cdot k$ linear inequations where the coefficients are from $\mathbb{Q}$. Since linear integer programming with rational coefficients and a fixed number of constraints is possible in polynomial time (see [41]), this shows that checking whether a candidate assignment $\vec{v}$ gives rise to a solution in $\mathcal{L}_T$ can be done in XP. As $|\operatorname{cndAssg}(\psi)|$ is also polynomial for fixed k, k-termination of uniform loops over $\mathbb{Z}$ is in XP as well.

8 Conclusion and related work

In this work, we studied termination of *twn*-loops, i.e., loops where the update $\vec{x} \leftarrow \vec{u}$ is a triangular system of polynomial equations and the use of non-linearity in $\vec{u}$ is mildly restricted. We first presented a reduction from termination of *twn*-loops to $\mathrm{Th}_\exists(\mathcal{S})$ in Sects. 3 and 4. This implies decidability of termination over $\mathcal{S} \in \{\mathbb{R}_\mathbb{A}, \mathbb{R}\}$ and semi-decidability of non-termination over $\mathcal{S} \in \{\mathbb{Z}, \mathbb{Q}\}$.

In addition, we showed how to transform certain non-*twn*-loops into *twn*-form in Sect. 5, and discussed how this generalizes our results to a wider class of loops. We also showed that *twn*-transformability is semi-decidable.

Afterwards, we analyzed the complexity of deciding termination for different subclasses of *twn*-loops. In Sect. 6, we first showed that linearizing *twn*-loops can be done in double exponential time. In Sect. 7, we used our transformation and decision procedure to prove Co-NP-completeness ($\forall\mathbb{R}$-completeness) of termination of linear (linear-update) loops with rational (real) spectrum, and based on linearization, that deciding termination of arbitrary *twn*-loops over $\mathbb{R}_\mathbb{A}$ or $\mathbb{R}$ is in 3-EXPTIME.

Finally, we showed that for the subclass of uniform loops over $\mathcal{S} \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}_\mathbb{A}, \mathbb{R}\}$, termination can be decided in polynomial time, if the number of eigenvalues of the update matrix is fixed. So here our decision procedure can be used as an efficient technique for termination analysis.

8.1 Related work

In contrast to automated termination analysis (see e.g., [1, 6, 7, 9, 11, 24, 25, 39, 40, 51]), we investigated *decidability* of termination for certain classes of loops in Sect. 4. As termination is undecidable in general, decidability results can only be obtained for very restricted classes of programs.

Nevertheless, many techniques used in automated tools for termination analysis (e.g., ranking functions [1, 5–7, 9, 51]) focus on similar classes of loops, since such loops occur as sub-programs in (abstractions of) real programs. Tools based on these techniques have turned out to be very successful, also for larger classes of programs. Thus, these tools could benefit from integrating our (semi-)decision procedures and applying them instead of incomplete techniques for any sub-program that can be transformed into a *twn*-loop.

Related work on decidability of termination also considers similar (and often more restricted) classes of loops. For linear conjunctive loops, termination over $\mathbb{R}$ [42, 45, 62, 66] and $\mathbb{Q}$ [10] is decidable. Decidability of termination of linear conjunctive loops over $\mathbb{Z}$ was conjectured to be decidable in [62]. After several partial results [8, 19, 49] this conjecture was confirmed recently in [31]. However, [4] shows that for slight generalizations of linear conjunctive loops over $\mathbb{Z}$, where a non-deterministic update or a single piecewise update of a variable are allowed, termination is undecidable. Tiwari [62] uses the special case of our *twn*-transformation from Sect. 5 where the loop and the automorphism are linear. In contrast to these results, our approach applies to *non-linear* loops with *arbitrary* conditions over *various rings*.

Linearization is another attempt to handle non-linearity, see Sect. 6. While the *update* of solvable loops can be linearized [48], the *condition* cannot. Otherwise, one could linearize any loop $(p = 0, \vec{x})$, which terminates over $\mathbb{Z}$ iff p has no integer root. By [31], this would imply decidability of Hilbert's Tenth Problem.

In the non-linear case, [43] proves decidability of termination for conjunctive loops on $\mathbb{R}^d$ for the case that the condition defines a compact and connected subset of $\mathbb{R}^d$. In [65], decidability of termination of conjunctive linear-update loops on $\mathbb{R}^d$ with the *non-zero minimum property* is shown, which covers conjunctive linear-update loops with real spectrum. For general conjunctive linear-update loops on $\mathbb{R}^d$ undecidability is conjectured. Moreover, [64] shows that termination of conjunctive linear-update loops where the update matrix has only periodic real eigenvalues is decidable, which also covers conjunctive linear-update loops with real spectrum. Here, a special case of our transformation from Sect. 5 with linear automorphisms is used. In combination with [48], the papers [64, 65] both yield a decision procedure for termination of conjunctive *twn*-loops over $\mathbb{R}$. Furthermore, [47] proves that termination of (not necessarily conjunctive) linear-update loops is decidable if the condition describes a compact set. Finally, [67] gives sufficient criteria for (non-)termination of solvable loops and [44] presents sufficient conditions under which termination of non-deterministic non-linear loops on $\mathbb{R}^d$ can be reduced to satisfiability of a semi-algebraic system.

For linear-update loops with real spectrum over $\mathbb{R}$, we prove $\forall\mathbb{R}$-completeness of termination, whereas [64, 65] do not give tight complexity results. The approach from [67] is incomplete, whereas we present a complete reduction from termination to the respective existential fragment of the first-order theory. The work in [44] is orthogonal to ours as it only applies to loops that satisfy certain non-trivial conditions. Moreover, we consider loops with arbitrary conditions over various rings, while [43, 44, 64, 65] only consider conjunctive loops over $\mathbb{R}$ and [47] only considers loops over $\mathbb{R}$ where the condition defines a compact set.

Regarding complexity, [49] proves that termination of conjunctive linear loops over $\mathbb{Z}$ with update $\vec{x} \leftarrow A \cdot \vec{x} + \vec{b}$ is in PSPACE if $|\vec{x}| \leq 4$ resp. in EXPSPACE if A is diagonalizable. Moreover, in [5] it is shown that existence of a linear (lexicographic) ranking function for linear conjunctive loops over $\mathbb{Q}$ or $\mathbb{Z}$ is Co-NP-complete.

Our Co-NP-completeness result is orthogonal to those results as we allow disjunctions in the condition. Moreover, Co-NP-completeness also holds for termination over $\mathbb{Z}$, while

[10, 62] only consider termination over $\mathbb{Q}$ resp. $\mathbb{R}$. Additionally, we showed that k-termination of uniform loops over $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}_{\mathbb{A}}$, and $\mathbb{R}$ is in XP, where the parameter k is the number of eigenvalues. This result is also orthogonal to [10, 62] since we again allow disjunctions in the condition. Furthermore, existence of a linear (lexicographic) ranking function is not necessary for termination of linear loops. We refer to [29] for further discussion on possible extensions of our results to uniform loops over $\mathcal{S}$, where however the eigenvalues are *not* from $\mathcal{S}$.

Several works exploit the existence of closed forms for solvable (or similar classes of) loops, e.g., to analyze termination on a *given* input, to infer runtime bounds, or to reason about invariants [28, 32, 35, 36, 48, 56]. While our approach covers solvable loops with real eigenvalues (by Corollary 5.20), it also applies to loops which are not solvable, see Example 5.23. Our transformation of Sect. 5 may also be of interest for other techniques for solvable or other sub-classes of polynomial loops, as it may be used to extend the applicability of such approaches.

Appendix

A Proofs

A.1 Proof of Theorem 3.2

Proof

(a) By weak non-linearity, $u_i = c_i \cdot x_i + p_i$ with $x_i \notin \mathcal{V}(p_i)$ for all $1 \le i \le d$. Then

$$u_i(\vec{u}) = c_i \cdot (c_i \cdot x_i + p_i) + p_i(\vec{u}) = c_i^2 \cdot x_i + c_i \cdot p_i + p_i(\vec{u}).$$

Assume $x_i \in \mathcal{V}(p_i(\vec{u}))$. As $x_i \notin \mathcal{V}(p_i)$ by weak non-linearity, there is an $x_j \in \mathcal{V}(p_i)$ with $x_j \neq x_i$ and $x_i \in \mathcal{V}(u_j)$, which implies $x_j \succ_{\vec{u}} x_i$. But $x_j \in \mathcal{V}(p_i)$ also implies $x_i \succ_{\vec{u}} x_j$, which violates well-foundedness of $\succ_{\vec{u}}$, i.e., it contradicts the triangularity of $(\varphi, \vec{u})$. Hence, c_i^2 is the coefficient of x_i in $u_i(\vec{u})$. Since $c_i^2 \ge 0$, $(\varphi \wedge \varphi(\vec{u}), \vec{u}(\vec{u}))$ is non-negative. Note that $x_i \succ_{\vec{u}(\vec{u})} x_j$ implies $x_j \in \mathcal{V}(p_i)$ (then we also have $x_i \succ_{\vec{u}} x_j$) or it implies that there is an $x_k \in \mathcal{V}(p_i)$ with $x_j \in \mathcal{V}(u_k)$ (then we have $x_i \succ_{\vec{u}} x_k$ and $x_k \succeq_{\vec{u}} x_j$). So in both cases, $x_i \succ_{\vec{u}(\vec{u})} x_j$ implies $x_i \succ_{\vec{u}} x_j$. Thus, $\succ_{\vec{u}(\vec{u})} \subseteq \succ_{\vec{u}}$. As $\succ_{\vec{u}}$ is well founded, this means that $\succ_{\vec{u}(\vec{u})}$ is well founded, too. Hence, $(\varphi \wedge \varphi(\vec{u}), \vec{u}(\vec{u}))$ is triangular.

(b) Now we prove that $(\varphi, \vec{u})$ does not terminate on $\vec{e} \in \mathcal{S}^d$ iff $(\varphi \wedge \varphi(\vec{u}), \vec{u}(\vec{u}))$ does not terminate on $\vec{e}$.

$$
\begin{array}{lll}
 & (\varphi, \vec{u}) \text{ does not terminate on } \vec{e} & \\
\iff & \forall n \in \mathbb{N}.\ \varphi(\vec{u}^n(\vec{e})) & \text{(by Definition 2.1)} \\
\iff & \forall n \in \mathbb{N}.\ \varphi(\vec{u}^{2\cdot n}(\vec{e})) \wedge \varphi(\vec{u}^{2\cdot n+1}(\vec{e})) & \\
\iff & \forall n \in \mathbb{N}.\ \varphi(\vec{u}^{2\cdot n}(\vec{e})) \wedge \varphi(\vec{u})(\vec{u}^{2\cdot n}(\vec{e})) & \\
\iff & \forall n \in \mathbb{N}.\ (\varphi \wedge \varphi(\vec{u}))\,(\vec{u}(\vec{u}))^n(\vec{e}) & \\
\iff & (\varphi \wedge \varphi(\vec{u}), \vec{u}(\vec{u})) \text{ does not terminate on } \vec{e} &
\end{array}
$$

A.2 Proof of Lemma 4.4

Proof Recall that for $f, g : \mathbb{N} \to \mathbb{R}$, $f(n) \in o(g(n))$ means

$$\forall m > 0.\ \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ |f(n)| < m \cdot |g(n)|.$$

First consider the case $b_2 > b_1$. We have $b_2^n = b_1^n \cdot (\frac{b_2}{b_1})^n$, where $\frac{b_2}{b_1} > 1$. As $n^{a_1} \in o((\frac{b_2}{b_1})^n)$, we obtain $n^{a_1} \cdot b_1^n \in o((\frac{b_2}{b_1})^n \cdot b_1^n) = o(b_2^n) \subseteq o(n^{a_2} \cdot b_2^n)$, i.e., $n^{a_1} \cdot b_1^n \in o(n^{a_2} \cdot b_2^n)$.

Now consider the case $b_2 = b_1$ and $a_2 > a_1$. Then $n^{a_1} \cdot b_1^n \in o(n^{a_2} \cdot b_2^n)$ trivially holds.

A.3 Proof of Equation (5)

Proof If $p(\vec{e}) = 0$, then $k = 0$ by Definition 4.5 and hence $o(p(\vec{e})) = o(k \cdot n^a \cdot b^n) = o(0)$. Otherwise, $p(\vec{e})$ has the form

$$k \cdot n^a \cdot b^n + \sum_{i=1}^{m} k_i \cdot n^{a_i} \cdot b_i^n$$

for $k \neq 0$ and $m \geq 0$. We have $k_i^{(b_i,a_i)} \in \text{coefs}\left(p(\vec{e})\right)$ and hence $(b, a) >_{lex} (b_i, a_i)$ for all $1 \leq i \leq m$. Thus, Lemma 4.4 implies $n^{a_i} \cdot b_i^n \in o(n^a \cdot b^n)$ and we get

$$o(p(\vec{e})) = o\left(k \cdot n^a \cdot b^n + \textstyle\sum_{i=1}^{m} k_i \cdot n^{a_i} \cdot b_i^n\right) = o(n^a \cdot b^n) = o(k \cdot n^a \cdot b^n).$$

A.4 Proof of Equation (6)

Proof If $k = 0$, the claim is trivial, so assume $k \neq 0$, i.e., $p(\vec{e}) = k \cdot b^n \cdot n^a + p'$ for some $p' \in \mathbb{NPE}_{\mathcal{S}}$. By Lemma 4.4 we have

$$\begin{aligned}
& p' \in o(k \cdot b^n \cdot n^a) \\
\iff & \forall m > 0.\ \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ |p'| < m \cdot |k \cdot b^n \cdot n^a| \\
\implies & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ |p'| < |k \cdot b^n \cdot n^a|.
\end{aligned}$$

Assume $k > 0$. Then

$$\begin{aligned}
& \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ |p'| < |k \cdot b^n \cdot n^a| \\
\implies & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ -p' < |k \cdot b^n \cdot n^a| \\
\iff & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ -p' < k \cdot b^n \cdot n^a \\
\iff & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ 0 < k \cdot b^n \cdot n^a + p' \\
\iff & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ 0 < p(\vec{e}) \\
\iff & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \text{sign}\left(p(\vec{e})\right) = \text{sign}(k).
\end{aligned}$$

If $k < 0$, then

$$
\begin{aligned}
& \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ |p'| < |k \cdot b^n \cdot n^a| \\
\implies & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ p' < |k \cdot b^n \cdot n^a| \\
\iff & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ p' < -k \cdot b^n \cdot n^a \\
\iff & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ k \cdot b^n \cdot n^a + p' < 0 \\
\iff & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ p(\vec{e}) < 0 \\
\iff & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \operatorname{sign}\left(p(\vec{e})\right) = \operatorname{sign}(k).
\end{aligned}
$$

A.5 Proof of Lemma 4.7

Proof We have $p \in \mathbb{NPE}_{\mathcal{S}}[\vec{x}]$, so $p(\vec{e}) \in \mathbb{NPE}_{\mathcal{S}}$ for any $\vec{e} \in \mathcal{S}^d$. Hence,

$$\exists n_0 \in \mathbb{N}. \forall n \in \mathbb{N}_{>n_0}. p(\vec{e}) \rhd 0 \quad \text{iff} \quad \operatorname{unmark}(\max_{\succ_{coef}}\left(\operatorname{coefs}\left(p(\vec{e})\right)\right)) \rhd 0 \qquad \text{(by (6))}.$$

Let $\operatorname{coefs}(p) = \{\alpha_1^{(b_1,a_1)}, \ldots, \alpha_\ell^{(b_\ell,a_\ell)}\}$, where $\alpha_i^{(b_i,a_i)} \prec_{coef} \alpha_j^{(b_j,a_j)}$ for all $1 \le i < j \le \ell$. If $p(\vec{e}) = 0$, then $\alpha_1(\vec{e}) = \ldots = \alpha_\ell(\vec{e}) = 0$ and thus $\operatorname{coefs}\left(p(\vec{e})\right) = \{0^{(1,0)}\}$ and $\operatorname{unmark}(\max_{\succ_{coef}}\left(\operatorname{coefs}\left(p(\vec{e})\right)\right)) = 0$. Otherwise, there is an $1 \le i \le \ell$ with

$$\operatorname{unmark}(\max_{\succ_{coef}}\left(\operatorname{coefs}\left(p(\vec{e})\right)\right)) = \alpha_i(\vec{e}) \neq 0 \text{ and } \alpha_j(\vec{e}) = 0 \text{ for all } i+1 \le j \le \ell.$$

So when defining $\operatorname{red}(p > 0)$ and $\operatorname{red}(p \ge 0)$ as in (7), we obviously have

$$\operatorname{unmark}(\max_{\succ_{coef}}\left(\operatorname{coefs}\left(p(\vec{e})\right)\right)) \rhd 0 \quad \text{iff} \quad (\operatorname{red}(p \rhd 0))(\vec{e}) \text{ holds.}$$

Hence, (8) is equivalent to

$$\exists \vec{x} \in \mathcal{S}^d.\ \operatorname{red}(p \rhd 0). \tag{16}$$

The time needed to compute and sort $\operatorname{coefs}(p)$ is polynomial. Furthermore, $\operatorname{red}(p \rhd 0)$ is a disjunction of at most $\ell + 1$ subformulas, where each subformula is a conjunction of at most ℓ (in-)equations over $\operatorname{coefs}(p)$. Thus, the time needed to compute $\operatorname{red}(p \rhd 0)$ resp. (16) is polynomial in the size of p.

A.6 Proof of Theorem 4.9

Proof We have to prove

$$(9) \iff \exists \vec{x} \in \mathcal{S}^d.\ \operatorname{red}(\xi) \tag{17}$$

where $\operatorname{red}(\xi)$ results from replacing each atom $p \rhd 0$ in ξ by $\operatorname{red}(p \rhd 0)$. Since each $\operatorname{red}(p \rhd 0)$ can be computed in polynomial time due to Lemma 4.7, the computation of the formula "$\exists \vec{x} \in \mathcal{S}^d.\ \operatorname{red}(\xi)$" clearly works in polynomial time, too.

To prove (17), we introduce the notion of a *fundamental set*. Let $p_1 \rhd_1 0, \ldots, p_k \rhd_k 0$ denote the atoms in ξ. We call a subset $I \subseteq \{1, \ldots, k\}$ *fundamental* if $\bigwedge_{i \in I} p_i \rhd_i 0 \implies \xi$. Recall that w.l.o.g., we can assume that ξ does not contain any connectives except $\wedge$ and $\vee$. Thus, whenever $\xi \neq \text{false}$, the formula ξ must have fundamental sets. Clearly, we have

$$\exists \vec{x} \in \mathcal{S}^d.\ \text{red}(\xi) \iff \exists \text{ fundamental set } I.\ \exists \vec{x} \in \mathcal{S}^d.\ \bigwedge\nolimits_{i \in I} \text{red}(p_i \rhd_i 0).$$

Thus, to prove (17), it suffices to show the following:

$$(9) \iff \exists \text{ fundamental set } I.\ \exists \vec{x} \in \mathcal{S}^d.\ \bigwedge\nolimits_{i \in I} \text{red}(p_i \rhd_i 0). \tag{18}$$

For the "$\Longleftarrow$"-direction, assume there is such a fundamental set and $\vec{e} \in \mathcal{S}^d$, i.e.,

$$\bigwedge\nolimits_{i \in I} \text{red}(p_i \rhd_i 0)(\vec{e})$$

is valid. Then as in the proof of Lemma 4.7, for each $i \in I$, there is an $n_i \in \mathbb{N}$ such that

$$\exists \vec{x} \in \mathcal{S}^d.\ \forall n \in \mathbb{N}_{>n_i}.\ p_i \rhd_i 0.$$

As I is finite, $n_{\max} = \max \{n_i | i \in I\}$ exists. Hence, we get

$$\exists \vec{x} \in \mathcal{S}^d.\ \forall n \in \mathbb{N}_{>n_{max}}.\ \bigwedge\nolimits_{i \in I} p_i \rhd_i 0.$$

Since I is fundamental, this implies (9).

For the "$\Longrightarrow$"-direction, assume (9). Then there is an $\vec{e} \in \mathcal{S}^d$ and an $n_0 \in \mathbb{N}$ such that for each $n \in \mathbb{N}_{>n_0}$, there is a fundamental set I_n such that $\bigwedge_{i \in I_n} p_i(\vec{e}) \rhd_i 0$ holds. As there are only finitely many fundamental sets, there is some fundamental set I that occurs infinitely often in $(I_n)_{n \in \mathbb{N}_{>n_0}}$. Hence we get

$$\exists n_0 \in \mathbb{N}.\ \exists^\infty n \in \mathbb{N}_{>n_0}.\ \bigwedge\nolimits_{i \in I} p_i(\vec{e}) \rhd_i 0. \tag{19}$$

By definition of poly-exponential expressions, each $p_i(\vec{e})$ is weakly monotonic in n for large enough n. Thus, (19) implies[5]

$$\exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \bigwedge\nolimits_{i \in I} p_i(\vec{e}) \rhd_i 0.$$

As $\vec{e} \in \mathcal{S}^d$, there is a fundamental set I such that $\exists \vec{x} \in \mathcal{S}^d.\ \bigwedge_{i \in I} \text{red}(p_i \rhd_i 0)$ holds.

A.7 Proof of Theorem 4.10

Proof By Corollary 3.3, termination of *twn*-loops is reducible to termination of *tnn*-loops. Given a *tnn*-loop $(\varphi, \vec{u})$, we obtain $\vec{q}_{norm} \in \left(\mathbb{NPE}_{\mathcal{S}}[\vec{x}]\right)^d$ such that $(\varphi, \vec{u})$ is (eventually) non-terminating iff (4) holds, where φ is a propositional formula over the atoms $\{\alpha \geq 0, \alpha > 0 \mid \alpha \in \mathcal{S}[\vec{x}]\}$. Hence, $\varphi(\vec{q}_{norm})$ is a propositional formula over the atoms

5 This corresponds to the observation that if the loop condition is a disjunction (and hence also ξ is a disjunction of the form $\xi_1 \vee \xi_2$), then non-termination of the original loop implies non-termination of one of the loops where instead of the disjunction one only takes one of the disjuncts as the loop guard. The reason is that for every fundamental set I and every $\vec{e} \in \mathcal{S}^d$, $\bigwedge_{i \in I} p_i(\vec{e}) \rhd_i 0 \implies (\xi_1(\vec{e}) \vee \xi_2(\vec{e}))$ implies $\bigwedge_{i \in I} p_i(\vec{e}) \rhd_i 0 \implies \xi_1(\vec{e})$ or $\bigwedge_{i \in I} p_i(\vec{e}) \rhd_i 0 \implies \xi_2(\vec{e})$. The restriction to *tnn*-loops implies that the closed forms are poly-exponential expressions and hence, that the $p_i(\vec{e})$ are weakly monotonic in n for large enough n. Therefore, the above argumentation in the proof shows that there is a fundamental set I such that $\bigwedge_{i \in I} p_i(\vec{e}) \rhd_i 0$ holds for all large enough n and thus, for some $j \in \{1, 2\}$, $\xi_j(\vec{e})$ holds for all large enough n as well. Hence, the loop would also be non-terminating if one only takes the corresponding disjunct as the loop guard.

$\{p \rhd 0 \mid p \in \mathbb{NPE}_{\mathcal{S}}[\vec{x}], \rhd \in \{\geq, >\}\}$. Thus, by Theorem 4.9, validity of (4) resp. (9) is reducible to $\mathrm{Th}_{\exists}(\mathcal{S})$.

A.8 Proof of Corollary 4.11

Proof Again, by Corollary 3.3, termination of *twn*-loops is reducible to termination of *tnn*-loops. By Theorem 4.10, termination of *tnn*-loops is reducible to invalidity of a closed formula $\chi \in \mathrm{Th}_{\exists}(\mathcal{S})$. If $\mathcal{S} = \mathbb{R}_{\mathbb{A}}$, then validity of χ is decidable, and if $\mathcal{S} = \mathbb{Z}$ or $\mathcal{S} = \mathbb{Q}$, then validity of χ is semi-decidable [13, 61]. But χ is valid iff the loop is non-terminating. Hence, non-termination is decidable for $\mathcal{S} = \mathbb{R}_{\mathbb{A}}$ and semi-decidable if $\mathcal{S} = \mathbb{Z}$ or $\mathcal{S} = \mathbb{Q}$. The claim (b) for $\mathcal{S} = \mathbb{R}_{\mathbb{A}}$ follows since *deciding* non-termination is equivalent to deciding termination. Finally, (a) and the claim (b) for $\mathcal{S} = \mathbb{R}$ follow due to elementary equivalence of $\mathbb{R}_{\mathbb{A}}$ and $\mathbb{R}$.

A.9 Proof of Lemma 4.12

Proof We have:

$$
\begin{aligned}
& \vec{e} \text{ witnesses eventual non-termination of } (\varphi, \vec{u}) \\
\iff & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \big(\varphi(\vec{q}_{norm})\big)(\vec{e}) && \text{(by (4))} \\
\iff & \exists n_0 \in \mathbb{N}.\ \forall n \in \mathbb{N}_{>n_0}.\ \xi(\vec{e}) \\
\iff & \mathrm{red}\,(\xi)(\vec{e}) && \text{(as in the proof of Theorem 4.9)}
\end{aligned}
$$

A.10 Proof of Lemma 5.6

Proof Let $(\varphi, \vec{u})$ be a loop. Since $id^{-1}_{\mathcal{S}[\vec{x}]} = id_{\mathcal{S}[\vec{x}]}$, we obtain $\mathrm{Tr}_{\,id_{\mathcal{S}[\vec{x}]}}(\varphi, \vec{u}) = (\varphi', \vec{u}\,')$ with

$$
\begin{aligned}
\varphi' &= id^{-1}_{\mathcal{S}[\vec{x}]}(\varphi) = \varphi \\
\vec{u}\,' &= (id^{-1}_{\mathcal{S}[\vec{x}]} \circ \widetilde{u} \circ id_{\mathcal{S}[\vec{x}]})(\vec{x}) = \widetilde{u}(\vec{x}) = \vec{u}
\end{aligned}
$$

Now we take $\eta_1, \eta_2 \in \mathrm{Aut}_{\mathcal{S}}\big(\mathcal{S}[\vec{x}]\big)$. Note that $(\eta_1 \circ \eta_2)^{-1} = \eta_2^{-1} \circ \eta_1^{-1}$. Let $\mathrm{Tr}_{\,\eta_1 \circ \eta_2}(\varphi, \vec{u}) = (\varphi', \vec{u}\,')$, $\mathrm{Tr}_{\,\eta_1}(\varphi, \vec{u}) = (\varphi'', \vec{u}\,'')$, and $Tr_{\eta_2}(\varphi'', \vec{u}\,'') = (\varphi''', \vec{u}\,''')$. We have

$$
\begin{aligned}
\varphi' &= (\eta_2^{-1} \circ \eta_1^{-1})(\varphi) \\
\varphi'' &= \eta_1^{-1}(\varphi) \\
\varphi''' &= \eta_2^{-1}(\varphi'') \\
&= \eta_2^{-1}(\eta_1^{-1}(\varphi)) \\
&= (\eta_2^{-1} \circ \eta_1^{-1})(\varphi) \\
&= \varphi'
\end{aligned}
$$

Moreover, we have

$$\begin{aligned}
\vec{u}\,' &= (\eta_2^{-1} \circ \eta_1^{-1} \circ \widetilde{\vec{u}} \circ \eta_1 \circ \eta_2)(\vec{x}) \\
&= (\eta_2(\vec{x}))\ (\eta_1(\vec{x}))\ (\vec{u})\ (\eta_1^{-1}(\vec{x}))\ (\eta_2^{-1}(\vec{x})) \\
\vec{u}\,'' &= (\eta_1^{-1} \circ \widetilde{\vec{u}} \circ \eta_1)(\vec{x}) \\
&= (\eta_1(\vec{x}))\ (\vec{u})\ (\eta_1^{-1}(\vec{x})) \\
\vec{u}\,''' &= (\eta_2^{-1} \circ \widetilde{\vec{u}\,''} \circ \eta_2)(\vec{x}) \\
&= (\eta_2(\vec{x}))\ (\eta_1(\vec{x}))\ (\vec{u})\ (\eta_1^{-1}(\vec{x}))\ (\eta_2^{-1}(\vec{x})) \\
&= \vec{u}\,'
\end{aligned}$$

A.11 Proof of Lemma 5.7

Proof Let $\vec{e} \in \mathcal{S}^d$ and $n \in \mathbb{N}$. Then

$$\begin{aligned}
& \varphi'((\vec{u}\,')^n((\eta(\vec{x}))\,(\vec{e}))) \\
=\ & \eta^{-1}(\varphi)\ ((\vec{u}\,')^n((\eta(\vec{x}))\,(\vec{e}))) \\
=\ & \varphi\,[\vec{x}/\eta^{-1}(\vec{x})]\ \underbrace{[\vec{x}/\vec{u}\,']}_{n\text{ times}}\,[\vec{x}/\eta(\vec{x})]\,[\vec{x}/\vec{e}] \\
=\ & \varphi\,[\vec{x}/\eta^{-1}(\vec{x})]\,[\vec{x}/\eta(\vec{x})]\ \underbrace{[\vec{x}/\vec{u}]}_{n\text{ times}}\ [\vec{x}/\eta^{-1}(\vec{x})]\,[\vec{x}/\eta(\vec{x})]\,[\vec{x}/\vec{e}] \\
=\ & \varphi\ \underbrace{[\vec{x}/\vec{u}]}_{n\text{ times}}\ [\vec{x}/\vec{e}] \\
=\ & \varphi(\vec{u}^n(\vec{e}))
\end{aligned}$$

A.12 Proof of Theorem 5.10

Proof In Corollary 5.8, we have seen that if $\vec{e}$ is a witness for (eventual) non-termination of $(\varphi, \vec{u})$, then $\widehat{\eta}(\vec{e})$ witnesses (eventual) non-termination of $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$. Now let $\vec{e}\,'$ be a witness for (eventual) non-termination of $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$. Then by Corollary 5.8, $\widehat{\eta^{-1}}(\vec{e}\,')$ witnesses (eventual) non-termination of $Tr_{\eta^{-1}}(\mathrm{Tr}_{\eta}(\varphi, \vec{u})) \overset{\text{Lemma 5.6}}{=} \mathrm{Tr}_{\eta \circ \eta^{-1}}(\varphi, \vec{u}) = (\varphi, \vec{u})$. Hence, $\widehat{\eta}$ maps witnesses for (eventual) non-termination of $(\varphi, \vec{u})$ to witnesses for (eventual) non-termination of $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$ and $\widehat{\eta^{-1}}$ maps witnesses for (eventual) non-termination of $\mathrm{Tr}_{\eta}(\varphi, \vec{u})$ to witnesses for $(\varphi, \vec{u})$. These two mappings are inverse to each other: For $\vec{e}\,' \in \mathcal{S}^d$ we have

$$\begin{aligned}
& \widehat{\eta}(\widehat{\eta^{-1}}(\vec{e}\,')) \\
= \;& \widehat{\eta}((\eta^{-1}(\vec{x}))(\vec{e}\,')) && \text{(by definition of } \widehat{\eta^{-1}}) \\
= \;& (\eta(\vec{x}))((\eta^{-1}(\vec{x}))(\vec{e}\,')) && \text{(by definition of } \widehat{\eta}) \\
= \;& \eta(\vec{x})[\vec{x}/\eta^{-1}(\vec{x})][\vec{x}/\vec{e}\,'] \\
= \;& \vec{e}\,'
\end{aligned}$$

$$\begin{aligned}
& \widehat{\eta^{-1}}(\widehat{\eta}(\vec{e})) \\
= \;& \widehat{\eta^{-1}}((\eta(\vec{x}))(\vec{e})) && \text{(by definition of } \widehat{\eta}) \\
= \;& (\eta^{-1}(\vec{x}))((\eta(\vec{x}))(\vec{e})) && \text{(by definition of } \widehat{\eta^{-1}}) \\
= \;& \eta^{-1}(\vec{x})[\vec{x}/\eta(\vec{x})][\vec{x}/\vec{e}] \\
= \;& \vec{e}.
\end{aligned}$$

Hence, $\widehat{\eta}$ is indeed a bijection with inverse mapping $\widehat{\eta^{-1}}$.

A.13 Proof of Lemma 5.12

Proof Let $\vec{e}\,' \in \widehat{\eta}(F)$. Then $\vec{e}\,' = \widehat{\eta}(\vec{e})$ for some $\vec{e} \in F$. As F is $\vec{u}$-invariant, we have $\vec{u}(\vec{e}) \in F$. We obtain

$$\begin{aligned}
\vec{u}\,'(\vec{e}\,') &= (\eta(\vec{x}))\,(\vec{u})\,(\eta^{-1}(x))\,(\vec{e}\,') \\
&= (\eta(\vec{x}))\,(\vec{u})\,(\eta^{-1}(x))\,(\eta(\vec{x}))\,(\vec{e}) \\
&= (\eta(\vec{x}))\,(\vec{u})\,(\vec{e}) \\
&= \widehat{\eta}(\vec{u}(\vec{e})) \in \widehat{\eta}(F).
\end{aligned}$$

A.14 Proof of Lemma 5.13

Proof Let F be defined by $\psi_F \in \mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$. Consider the following formula $\psi \in \mathrm{Th}_\exists(\mathcal{S}, \mathbb{R}_{\mathbb{A}})$:

$$\exists \vec{y} \in \mathbb{R}_{\mathbb{A}}^d.\ \psi_F(\vec{y}) \wedge \vec{x} = \big(\eta(\vec{x})\big)(\vec{y}).$$

Then $\psi(\vec{e})$ holds for $\vec{e} \in \mathbb{R}_{\mathbb{A}}^d$ iff $\vec{e} = \widehat{\eta}(\vec{u})$ for some $\vec{u} \in \mathbb{R}_{\mathbb{A}}^d$ with $\psi_F(\vec{u})$, i.e., with $\vec{u} \in F$.

A.15 Proof of Theorem 5.15

Proof As $(\varphi, \vec{u})$ is solvable, there is a partitioning $\mathcal{J} = \{J_1, \dots, J_k\}$ as in Definition 2.3, i.e., $\{1, \dots, d\} = \biguplus_{i=1}^k J_i$ and $\vec{u}_{J_i} = A_i \cdot \vec{x}_{J_i} + \vec{p}_i$ for all $1 \le i \le k$, where $\vec{p}_i \in (\mathcal{S}[\vec{x}_{J_{i+1}}, \dots, \vec{x}_{J_k}])^{d_i}$. W.l.o.g., $\vec{x}$ is ordered according to $\mathcal{J}$, i.e., if $x_{i_1} \in J_{j_1}$ and $x_{i_2} \in J_{j_2}$ for $j_1 < j_2$, then $i_1 < i_2$.

For each A_i, let $Q_i = T_i \cdot A_i \cdot T_i^{-1}$ be its Jordan normal form, where T_i is the corresponding transformation matrix. Since A_i has only real eigenvectors, this means that the entries of Q_i, T_i, and T_i^{-1} are real algebraic numbers. Let η be the endomorphism defined by $\eta(\vec{x}_{J_i}) = T_i \cdot \vec{x}_{J_i}$. This means that η is induced by the block diagonal matrix

$Diag(T_1, T_2, \dots, T_k)$. Then η is an automorphism and its inverse satisfies $\eta^{-1}(\vec{x}_{J_i}) = T_i^{-1} \cdot \vec{x}_{J_i}$. Furthermore, the degree of η is obviously 1. Moreover, η and η^{-1} are compatible with the partition, i.e., the images of the variables in $\vec{x}_{J_i}$ under η and η^{-1} are polynomials only using the variables $\vec{x}_{J_i}$. For each $1 \le i \le k$ we have:

$$\begin{aligned}
& (\eta^{-1} \circ \widetilde{u} \circ \eta)(\vec{x}_{J_i}) \\
= \; & \eta(\vec{x}_{J_i})[\vec{x}/\vec{u}][\vec{x}/\eta^{-1}(\vec{x})] \\
= \; & \left(T_i \cdot \vec{x}_{J_i}\right)[\vec{x}/\vec{u}][\vec{x}/\eta^{-1}(\vec{x})] \\
= \; & (T_i \cdot \vec{u}_{J_i})[\vec{x}/\eta^{-1}(\vec{x})] \\
= \; & \left(T_i \cdot \left(A_i \cdot \vec{x}_{J_i} + \vec{p}_i\right)\right)[\vec{x}/\eta^{-1}(\vec{x})] \\
= \; & \left(T_i \cdot A_i \cdot \vec{x}_{J_i} + T_i \cdot \vec{p}_i\right)[\vec{x}/\eta^{-1}(\vec{x})] \\
= \; & \left(T_i \cdot A_i \cdot \vec{x}_{J_i}\right)[\vec{x}/\eta^{-1}(\vec{x})] + \left(T_i \cdot \vec{p}_i\right)[\vec{x}/\eta^{-1}(\vec{x})] \\
= \; & \left(T_i \cdot A_i \cdot T_i^{-1} \cdot \vec{x}_{J_i}\right) + \left(T_i \cdot \vec{p}_i\right)[\vec{x}/\eta^{-1}(\vec{x})] \\
= \; & \left(Q_i \cdot \vec{x}_{J_i}\right) + \left(T_i \cdot \vec{p}_i\right)[\vec{x}/\eta^{-1}(\vec{x})]
\end{aligned}$$

We have $T_i \cdot \vec{p}_i \in \mathcal{S}[\vec{x}_{J_{i+1}}, \dots, \vec{x}_{J_k}]^{d_i}$. Therefore, $\left(T_i \cdot \vec{p}_i\right)[\vec{x}/\eta^{-1}(\vec{x})] \in \mathcal{S}[\vec{x}_{J_{i+1}}, \dots, \vec{x}_{J_k}]^{d_i}$ as well, since η^{-1} is compatible with the partitioning. This implies that $\mathrm{Tr}_\eta(\varphi, \vec{u})$ is weakly non-linear. As we assumed that $\vec{x}$ is ordered w.r.t. the partitioning and each Q_i is triangular, $\mathrm{Tr}_\eta(\varphi, \vec{u})$ is triangular, too. Thus, $\mathrm{Tr}_\eta(\varphi, \vec{u})$ is in *twn*-form.

A.16 Proof of Lemma 5.18

Proof Let $\delta = \deg(\eta)$. For any $k \in \mathbb{N}$, there is only a finite number of monomials over $\vec{x}$ of degree k. (The number of monomials of exactly degree k is $\binom{d+k-1}{k}$, see the proof of Theorem 6.11 (b).) Hence, for any $1 \le i \le d$ we can construct the following term that stands for $\eta^{-1}(x_i)$:

$$\sum\nolimits_{m \text{ is a monomial of (at most) degree } \delta^{d-1}} a_{i,m} \cdot m$$

Here, the monomials m contain the variables $\vec{x}$ and the $a_{i,m}$ are variables that stand for the unknown coefficients of the polynomial $\eta^{-1}(x_i)$.

Hence, for any $1 \le i \le d$ we now build a formula $\rho_{r,i}$ which stands for the requirement "$\left(\eta \circ \eta^{-1}\right)(x_i) = \left(\eta^{-1}(x_i)\right)(\eta(\vec{x})) = x_i$" (i.e., that η^{-1} is a right inverse of η):

$$\rho_{r,i}: \qquad \sum\nolimits_{m \text{ is a monomial of (at most) degree } \delta^{d-1}} a_{i,m} \cdot \eta(m) = x_i$$

Similarly, for any $1 \le i \le d$ we construct a formula $\rho_{l,i}$ which stands for the requirement "$\left(\eta^{-1} \circ \eta\right)(x_i) = (\eta(x_i))\left(\eta^{-1}(\vec{x})\right) = x_i$" (i.e., that η^{-1} is a left inverse of η):

$$\rho_{l,i}: \qquad \eta(x_i)\left(\sum\nolimits_{m \text{ is a monomial of (at most) degree } \delta^{d-1}} a_{i,m} \cdot m\right) = x_i$$

Thus, the formula

$$\forall \vec{x} \in \mathbb{R}^d_{\mathbb{A}}. \quad \bigwedge\nolimits_{i=1}^{d} \rho_{r,i} \wedge \bigwedge\nolimits_{i=1}^{d} \rho_{l,i} \tag{20}$$

is valid iff η has an inverse of degree at most δ^{d-1}. By Theorem 5.17, this is equivalent to the question whether η has an inverse, i.e., whether η is an automorphism. Unfortunately, (20) $\notin \text{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$. However, $\bigwedge_{i=1}^{d} \rho_{r,i} \wedge \bigwedge_{i=1}^{d} \rho_{l,i}$ has to hold for all $\vec{x} \in \mathbb{R}^d_{\mathbb{A}}$. So, we can reduce this formula to a system of equations: one simply has to check whether there is an instantiation of the unknown coefficients $a_{i,m}$ such that all monomials in $\rho_{r,i}$ and $\rho_{l,i}$ except x_i get the coefficient 0 and the monomial x_i gets the coefficient 1. When building the conjunction of these equations and existentially quantifying the unknown coefficients $a_{i,m}$, one indeed obtains a formula from $\text{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$.

A.17 Proof of Theorem 5.19

Proof For every $1 \leq i \leq d$, let

$$\eta(x_i) = \sum\nolimits_{m \text{ is a monomial of (at most) degree } \delta} b_{i,m} \cdot m,$$

where the $b_{i,m}$ are variables that stand for unknown coefficients. By Lemma 5.18 there is a $\text{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$-formula that contains both $b_{i,m}$ and the variables $a_{i,m}$ (for the coefficients of η^{-1}) which expresses that η is an automorphism.

Furthermore, using these coefficients we can construct a formula from $\text{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$ which expresses that the update $\vec{u}' = (u'_1, \dots, u'_d) = (\eta^{-1} \circ \widetilde{u} \circ \eta)(\vec{x})$ is *twn*: We have $\deg(\vec{u}') = \deg((\eta^{-1} \circ \widetilde{u} \circ \eta)(\vec{x})) \leq \deg(\eta^{-1}) \cdot \deg(\widetilde{u}) \cdot \deg(\eta) \leq \delta^{d-1} \cdot \deg(\widetilde{u}) \cdot \delta$. So there is a bound on the degree of the polynomials in the transformed loop $\text{Tr}_{\eta}(\varphi, \vec{u})$. For every $1 \leq i \leq d$, let

$$u'_i = \sum\nolimits_{m \text{ is a monomial of (at most) degree } \delta^{d-1} \cdot \deg(\widetilde{u}) \cdot \delta} c_{i,m} \cdot m,$$

where the variables $c_{i,m}$ stand again for unknown coefficients. Now we can build a $\text{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$-formula which is valid iff $\vec{u}'$ is in *twn*-form by requiring that certain coefficients $c_{i,m}$ are zero. Moreover, we can construct a $\text{Th}_{\exists}(\mathbb{R}_{\mathbb{A}})$-formula which is valid iff $\vec{u}' = (\eta^{-1} \circ \widetilde{u} \circ \eta)(\vec{x})$.

A.18 Proof of Corollary 6.3

Proof The proof is by induction on n. The induction base $n = 0$ is trivial. In the induction step $n > 0$ we obtain

$$\begin{aligned}
& \vec{w}(\vec{u}^{n+1}(\vec{e})) \\
= \; & \vec{w}(\vec{u}^{n}(\vec{u}(\vec{e}))) \\
= \; & (\vec{u}\,')^{n}(\vec{w}(\vec{u}(\vec{e}))) && \text{(by the induction hypothesis for } \vec{u}(\vec{e}) \in \mathcal{S}^d) \\
= \; & (\vec{u}\,')^{n}(\vec{u}\,'(\vec{w}(\vec{e}))) && \text{(by Definition 6.1)} \\
= \; & (\vec{u}\,')^{n+1}(\vec{w}(\vec{e}))
\end{aligned}$$

A.19 Proof of Lemma 6.6

Proof

(a) If there is no $\vec{e} \in \mathcal{S}^d$ such that $\vec{e}\,' = \vec{w}(\vec{e})$, then $\varphi'[\vec{y}/\vec{e}\,']$ is false by Definition 6.4.
(b) For any $n \in \mathbb{N}$, we have

$$\begin{aligned}
& \varphi'[\vec{y}/(\vec{u}\,')^n(\vec{w}(\vec{e}))] \\
\iff & \varphi'[\vec{y}/\vec{w}(\vec{u}^n(\vec{e}))] && \text{(by Corollary 6.3)} \\
\iff & \varphi[\vec{x}/\vec{u}^n(\vec{e})] \land \textstyle\bigwedge_{i=d+1}^{d'} \left(y_{w_i}[\vec{y}/\vec{w}(\vec{u}^n(\vec{e}))] - w_i[\vec{x}/\vec{u}^n(\vec{e})] = 0\right) && \text{(by Definition 6.4)} \\
\iff & \varphi[\vec{x}/\vec{u}^n(\vec{e})] \land \textstyle\bigwedge_{i=d+1}^{d'} \left(w_i(\vec{u}^n(\vec{e})) - w_i(\vec{u}^n(\vec{e})) = 0\right) \\
\iff & \varphi[\vec{x}/\vec{u}^n(\vec{e})] \land \textstyle\bigwedge_{i=d+1}^{d'} (0 = 0) \\
\iff & \varphi[\vec{x}/\vec{u}^n(\vec{e})].
\end{aligned}$$

Hence, $(\varphi, \vec{u})$ terminates on $\vec{e} \in \mathcal{S}^d$ iff $(\varphi', \vec{u}\,')$ terminates on $\vec{w}(\vec{e})$.

A.20 Proof of Lemma 6.9

Proof The claim (a) is obvious. The claim (b) is proved by induction on i. In the induction base, let $i = d$. Since x_d is minimal w.r.t. $\succ_{\vec{u}}$, we have $\operatorname{depdeg}_{\vec{u}}(x_d) = 1 \leq deg^0 = deg^{d-d}$.

In the induction step $i < d$, the claim is obviously true if $p_i = 0$. Otherwise, we obtain:

$$\begin{aligned}
& \operatorname{depdeg}_{\vec{u}}(x_i) \\
= \;& \max\{1, \operatorname{depdeg}_{\vec{u}}(p_i)\} \\
= \;& \max(\{1\} \cup \{\operatorname{depdeg}_{\vec{u}}(m) \mid m \text{ is a monomial in } p_i\}) \\
= \;& \max(\{1\} \cup \{\textstyle\sum_{j=i+1}^{d} z_j \cdot \operatorname{depdeg}_{\vec{u}}(x_j) \mid x_{i+1}^{z_{i+1}} \cdot \ldots \cdot x_d^{z_d} \text{ occurs in } p_i\}) \\
\leq \;& \max(\{1\} \cup \{\textstyle\sum_{j=i+1}^{d} z_j \cdot deg^{d-j} \mid x_{i+1}^{z_{i+1}} \cdot \ldots \cdot x_d^{z_d} \text{ occurs in } p_i\}) \quad \text{(by induction hypothesis)} \\
\leq \;& \max(\{1\} \cup \{deg^{d-i-1} \cdot \textstyle\sum_{j=i+1}^{d} z_j \mid x_{i+1}^{z_{i+1}} \cdot \ldots \cdot x_d^{z_d} \text{ occurs in } p_i\}) \\
\leq \;& \max(\{1\} \cup \{deg^{d-i-1} \cdot deg\}) && (\text{as } \textstyle\sum_{j=i+1}^{d} z_j \leq deg) \\
= \;& deg^{d-i}
\end{aligned}$$

The claim in (c) immediately follows from (b).

A.21 Proof of Theorem 6.11

Proof Let $z_1, \ldots, z_d \in \mathbb{N}$. We first show that for all monomials m in $u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$ we have

$$\operatorname{depdeg}(m) \leq \operatorname{depdeg}(x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}). \tag{21}$$

To prove (21), note that m must have the form $m_{1,1} \cdot \ldots \cdot m_{1,z_1} \cdot \ldots \cdot m_{d,1} \cdot \ldots \cdot m_{d,z_d}$ where the monomial $m_{i,j}$ occurs in u_i for all $1 \leq i \leq d$ and $1 \leq j \leq z_i$. Therefore, we have $\operatorname{depdeg}(m_{i,j}) \leq \operatorname{depdeg}(x_i)$. This is clear for $m_{i,j} = x_i$ and for $m_{i,j} \neq x_i$ it follows directly from the definition of the dependency degree in Definition 6.8. Hence, we can now prove (21):

$$\begin{aligned}\mathrm{depdeg}\,(m) &= \mathrm{depdeg}\,(m_{1,1}) + \ldots + \mathrm{depdeg}\,(m_{d,z_d}) \\ &\leq z_1 \cdot \mathrm{depdeg}\,(x_1) + \ldots + z_d \cdot \mathrm{depdeg}\,(x_d) \\ &= \mathrm{depdeg}\,(x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}).\end{aligned}$$

Now assume that (a) were not true. Then consider the first execution of Line 4 where we compute an update u'_m for a monomial m with $\mathrm{depdeg}_{\vec{u}}(m) > mdepdeg$. The monomial m resulted from taking a monomial $x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}$ from $\vec{v}$ and constructing $u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$, where m occurs in $u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$. Since m is the first monomial whose dependency degree is greater than $mdepdeg$, we have $\mathrm{depdeg}_{\vec{u}}(x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}) \leq mdepdeg$. But by (21), this implies $\mathrm{depdeg}_{\vec{u}}(m) \leq \mathrm{depdeg}_{\vec{u}}(x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}) \leq mdepdeg$, which contradicts our assumption.

For (b), since we do not build any update u'_m for the constant monomial m, (a) implies that the number of non-constant monomials m over the variables $\vec{x}$ with $\mathrm{depdeg}\,(m) \leq mdepdeg$ is an upper bound on the number of executions of the **while**-loop. As we have $\deg(m) \leq \mathrm{depdeg}\,(m)$ for any monomial by Lemma 6.9 (a), this number is bounded by the number of monomials over the variables $\vec{x}$ with a degree between 1 and $mdepdeg$.

The number of monomials over d variables with the exact degree $mdepdeg$ is $\binom{d + mdepdeg - 1}{mdepdeg}$ (this is the number of so-called *weak compositions* of $mdepdeg$ into d parts) and the number of monomials over d variables with a degree between 1 and $mdepdeg$ is $\binom{d}{1} + \binom{d+1}{2} + \ldots + \binom{d + mdepdeg - 1}{mdepdeg} = \binom{d + mdepdeg}{mdepdeg} - 1$.

Termination of Algorithm 3 follows from (b) since the **while**-loop is only executed finitely often.

A.22 Proof of Theorem 6.12

Proof The statement in (a) is obvious from Algorithm 3 and Theorem 6.11, the claim in (b) follows from Lemma 6.6, and the linearity of the update in (c) is again obvious from Algorithm 3.

It remains to show that $(\varphi', \vec{u}')$ is triangular and non-negative (i.e., it is weakly non-linear and the coefficient of the monomial y_m in u'_m is always non-negative).

For triangularity, we again assume that $x_i \succ_{\vec{u}} x_j$ implies $i > j$ for all $1 \leq i, j \leq d$. Then we show that $y_{m'} \succ_{\vec{u}'} y_m$ implies $m' \succ m$, where $\succ$ is the lexicographic ordering on monomials. Thus, if $m' = \vec{x}^{\vec{z}'}$ and $m = \vec{x}^{\vec{z}}$, then $m' \succ m$ holds iff $\vec{z}' >_{lex} \vec{z}$. Since $\succ$ is well founded, this implies the well-foundedness of $\succ_{\vec{u}'}$, i.e., the loop $(\varphi', \vec{u}')$ is triangular.

So let $m' \neq m$ and y_m occur in $u'_{m'}$. If $m' = x_1^{z_1} \cdot \ldots \cdot x_d^{z_d}$, then this means that m occurs in $u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$. Thus, $m = m_{1,1} \cdot \ldots \cdot m_{1,z_1} \cdot \ldots \cdot m_{d,1} \cdot \ldots \cdot m_{d,z_d}$ where $m_{i,j}$ occurs in u_i for all $1 \leq i \leq d$ and $1 \leq j \leq z_i$. Hence, we have $m_{i,j} = x_i$ or $m_{i,j}$ only contains variables x_j with $x_i \succ_{\vec{u}} x_j$. Thus, $x_i \succeq m_{i,j}$, where $\succeq$ is the reflexive closure of $\succ$. Hence, this implies $m' = x_1^{z_1} \cdot \ldots \cdot x_d^{z_d} \succeq m_{1,1} \cdot \ldots \cdot m_{1,z_1} \cdot \ldots \cdot m_{d,1} \cdot \ldots \cdot m_{d,z_d} = m$. Since $m' \neq m$, we have $m' \succ m$. As $\succ_{\vec{u}'}$ is the transitive closure of $\{(y_{m'}, y_m) \mid m \text{ occurs in } u'_{m'}\}$, the claim follows.

For non-negativity, note that $u'_{x_1^{z_1}\cdot\ldots\cdot x_d^{z_d}}$ results from $u_1^{z_1} \cdot \ldots \cdot u_d^{z_d}$, where $u_i = c_i \cdot x_i + p_i$ for all $1 \leq i \leq d$, and where $c_i \geq 0$ since $(\varphi, \vec{u})$ is a *tnn*-loop. Hence, $y_{x_1^{z_1}\cdot\ldots\cdot x_d^{z_d}}$ only occurs in $u'_{x_1^{z_1}\cdot\ldots\cdot x_d^{z_d}}$ in the addend $c_1 \cdot \ldots \cdot c_d \cdot y_{x_1^{z_1}\cdot\ldots\cdot x_d^{z_d}}$. Since this is a linear monomial and since $c_1 \cdot \ldots \cdot c_d \geq 0$, this implies non-negativity of $(\varphi', \vec{u}')$.

A.23 Proof of Equation (15)

Proof We have

$$\begin{array}{ll} & (d + deg^{d-1})^d \\ = & \sum_{i=0}^{d} \binom{d}{i} \cdot \left(deg^{d-1}\right)^i \cdot d^{d-i} \\ \leq & \sum_{i=0}^{d} \binom{d}{i} \cdot \max\left\{deg^{d-1}, d\right\}^d \\ = & \sum_{i=0}^{d} \binom{d}{i} \cdot \max\left\{\left(deg^{d-1}\right)^d, d^d\right\} \\ \leq & \sum_{i=0}^{d} \binom{d}{i} \cdot \left(\left(deg^{d-1}\right)^d + d^d\right) \\ = & 2^d \cdot \left(\left(deg^{d-1}\right)^d + d^d\right) \\ = & 2^{d+\operatorname{ld}(deg)\cdot(d-1)\cdot d} + 2^{d+\operatorname{ld}(d)\cdot d} \\ \leq & 2 \cdot 2^{d+\operatorname{ld}(deg)\cdot(d-1)\cdot d} \end{array}$$

A.24 Proof of Lemma 7.8

Proof In the following, let $\vec{u} = A \cdot \vec{x}$.

$\Leftarrow$: Let $\succ_{\vec{u}}$ be a total ordering and λ the unique eigenvalue of A, i.e., the diagonal of A only contains λ. We now prove that the matrix $A - \lambda \cdot I^{d\times d}$ has rank at least $d-1$, i.e., its kernel, which is the eigenspace of A w.r.t. λ, has dimension at most 1. As λ is an eigenvalue of A, its eigenspace then must be exactly one-dimensional.

Since A is triangular, so is $A - \lambda \cdot I^{d\times d}$. As $\succ_{\vec{u}}$ is a total ordering, the super-diagonal of $A - \lambda \cdot I^{d\times d}$ contains only non-zero values, whereas its diagonal contains only zeros. Thus, by deleting the first row and the last column of this matrix, we obtain a triangular $(d-1) \times (d-1)$ submatrix B whose diagonal is the super-diagonal of $A - \lambda \cdot I^{d\times d}$, i.e., it contains only non-zero values. Thus, the product of the diagonal entries of B is non-zero, i.e., $\det(B) \neq 0$. But B is a submatrix of $A - \lambda \cdot I^{d\times d}$, i.e., $A - \lambda \cdot I^{d\times d}$ has a non-zero $(d-1) \times (d-1)$ minor. Hence, $\operatorname{rank}(A - \lambda \cdot I^{d\times d}) \geq d-1$.

$\Longrightarrow$: Let us assume that $\succ_{\vec{u}}$ is not a total ordering. In this case, the super-diagonal of $A - \lambda \cdot I^{d\times d}$ contains a zero and its diagonal contains only zeros. Due to triangularity, its rank can be at most $d-2$. Thus, A has at least two linear independent eigenvectors.

A.25 Proof of Lemma 7.12

Proof Let Q be the Jordan normal form of A. From the form of Q^n_λ in Fig. 9, we directly obtain the following observation:

$$\text{For all } \lambda \in \operatorname{spec}(A) \text{ and } 1 \le r \le \nu(\lambda), \text{ the } (\operatorname{idx}(\lambda) + r)\text{-th element of } \vec{q} \\ = Q^n \cdot \vec{x} \text{ is } \sum\nolimits_{s=r}^{\nu(\lambda)} \binom{n}{s-r} \cdot \lambda^{n-s+r} \cdot x_{\operatorname{idx}(\lambda)+s}. \tag{22}$$

For $s - r \in \mathbb{N}$, in general $\binom{n}{s-r}$ is not a polynomial in the variable n, as one has to distinguish the cases $n < s - r$ and $n \ge s - r$. But since our approach from Sect. 4 analyzes *eventual* non-termination, we are only interested in validity of formulas for large enough n. Thus, we may assume $n \ge s - r$. Then, $\binom{n}{s-r}$ is indeed a polynomial from $\mathbb{Q}[n]$ of degree $s - r$, i.e., there are coefficients $c_{s-r,j} \in \mathbb{Q}$ such that for all $n \ge s - r$ we have $\binom{n}{s-r} = \sum_{j=0}^{s-r} c_{s-r,j} \cdot n^j$.

In fact, $c_{s-r,j} = \frac{\operatorname{stir}(s-r,j)}{(s-r)!}$ where stir is the *signed Stirling number of the first kind* (see, e.g., [27, Ch. 6]). While stir's formal definition is not of interest for us, we use $\operatorname{stir}(s-r, s-r) = 1 \neq 0$. We obtain the following from (22).

$$\text{For all } \lambda \in \operatorname{spec}(A) \text{ and } 1 \le r \le \nu(\lambda), \text{ the } (\operatorname{idx}(\lambda) + r)\text{-th element of } \vec{q}_{norm} \text{ is} \\ \sum\nolimits_{s=r}^{\nu(\lambda)} (\sum\nolimits_{j=0}^{s-r} c_{s-r,j} \cdot x_{\operatorname{idx}(\lambda)+s} \cdot \lambda^{r-s} \cdot n^j \cdot \lambda^n) \in \mathbb{NPE}_{\mathcal{S}}[\vec{x}]. \tag{23}$$

Re-arranging the order of summation of the closed form in (23) yields

$$\begin{aligned} &\sum\nolimits_{s=r}^{\nu(\lambda)} \left(\sum\nolimits_{j=0}^{s-r} c_{s-r,j} \cdot x_{\operatorname{idx}(\lambda)+s} \cdot \lambda^{r-s} \cdot n^j \cdot \lambda^n\right) \\ = \; &\sum\nolimits_{s=r}^{\nu(\lambda)} \underbrace{(\sum\nolimits_{j=0}^{\nu(\lambda)-s} c_{s-r+j,s-r} \cdot x_{\operatorname{idx}(\lambda)+s+j} \cdot \lambda^{r-s-j})}_{=\alpha_s} \cdot n^{s-r} \cdot \lambda^n. \end{aligned} \tag{24}$$

Here, α_s is a linear polynomial in the variables $x_{\operatorname{idx}(\lambda)+s}, \dots, x_{\operatorname{idx}(\lambda)+\nu(\lambda)}$. Note that the coefficient of $x_{\operatorname{idx}(\lambda)+s}$ in α_s is $c_{s-r,s-r} \cdot \lambda^{r-s} = \frac{\operatorname{stir}(s-r,s-r)}{(s-r)!} \cdot \lambda^{r-s} = \frac{1}{(s-r)!} \cdot \lambda^{r-s} \neq 0$.

By (23) and (24), the elements of $\vec{q}_{norm}$ are hierarchical expressions. Note that here we indeed need $\operatorname{stir}(s-r, s-r) = 1 \neq 0$, i.e., $\deg_{x_{\operatorname{idx}(\lambda)}+s}(\alpha_s) = 1$.

A.26 Proof of Lemma 7.19

Proof We have

$$p \; = \; f(0, \dots, 0) + \sum\nolimits_{1 \le i \le k,\, r \in F(i)} \operatorname{coeff}(f, x_r) \cdot h_r. \tag{25}$$

Moreover, for every $1 \le i \le k$ and every $r \in B_i$, the hierarchical expression h_r has the form

$$h_r = \sum_{s=r}^{v_1+\ldots+v_i} \beta_{r,s} \cdot n^{s-r} \cdot \lambda_i^n, \text{ where } \beta_{r,s} \in \mathcal{Q}_{\mathcal{S}}[x_s, \ldots, x_{v_1+\ldots+v_i}]_{\text{lin}}. \tag{26}$$

(d) Here, $\lambda_i \neq b$ for all $1 \leq i \leq k$ or $\lambda_i = b$ for some $1 \leq i \leq k$ but $F(i) = \emptyset$. Moreover, $b \neq 1$ or $a \geq 1$. Hence, (25) and (26) imply that there is no α_s with $b_s = b$ and $a_s \geq a$.

(a) If $\lambda_i \neq 1$ for all $1 \leq i \leq k$ or $\lambda_i = 1$ for some $1 \leq i \leq k$ and $F(i) = \emptyset$, then in (d) we already showed that there is no α_s with $b_s = 1$ and $a_s \geq 1$. However, there is an α_s with $b_s = 1$ and $a_s = 0$, viz. $\alpha_s = f(0, \ldots, 0)$. Thus, we have $\alpha_s = 0$ iff $f(0, \ldots, 0) = 0$.

(c) The largest number in B_i is $v_1 + \ldots + v_i$. So if $a > v_1 + \ldots + v_i - \min F(i)$ then we have zero $(b, a) =$ true. This is sound because then there is no α_s with $b_s = b$ and $a_s = a$. Hence, we now consider the case $a \leq v_1 + \ldots + v_i - \min F(i)$. Since $\lambda_i = b$ and $F(i) \neq 0$, there is an α_m with $b_m = b$. Let $1 \leq s_0 \leq \ell$ be the largest number with $b_{s_0} = b = \lambda_i$. By (25) and (26), the corresponding addend $\alpha_{s_0} \cdot n^{a_{s_0}} \cdot b_{s_0}^n$ of p has the form $\text{coeff}\,(f, x_r) \cdot \beta_{r,v_1+\ldots+v_i} \cdot n^{v_1+\ldots+v_i-r} \cdot \lambda_i^n$ for the smallest possible $r \in F(i)$. So we get

$$\alpha_{s_0} \cdot n^{a_{s_0}} \cdot b_{s_0}^n \;=\; \text{coeff}\,(f, x_{\min F(i)}) \cdot \beta_{\min F(i), v_1+\ldots+v_i} \cdot n^{v_1+\ldots+v_i-\min F(i)} \cdot \lambda_i^n.$$

Here, $\beta_{\min F(i), v_1+\ldots+v_i} \in \mathcal{Q}_{\mathcal{S}}[x_{v_1+\ldots+v_i}]_{\text{lin}}$ and in fact, $\beta_{\min F(i), v_1+\ldots+v_i} = c \cdot x_{v_1+\ldots+v_i}$ for some $c \neq 0$. Hence, $\alpha_{s_0} = 0$ is equivalent to $x_{v_1+\ldots+v_i} = 0$.

Now consider the second largest number $s_1 = s_0 - 1 \leq \ell$ such that $b_{s_1} = b = \lambda_i$. By (25) and (26), for $a_{s_1} = v_1 + \ldots + v_i - \min F(i) - 1$ we obtain

$$\alpha_{s_1} \cdot n^{a_{s_1}} \cdot b_{s_1}^n \;=\; \sum_{r \in F(i),\, \min F(i) \leq r \leq \min F(i)+1} \text{coeff}\,(f, x_r) \cdot \beta_{r, r+a_{s_1}} \cdot n^{a_{s_1}} \cdot \lambda_i^n,$$

where $\beta_{r,r+a_{s_1}} \in \mathcal{Q}_{\mathcal{S}}[x_{r+a_{s_1}}, \ldots, x_{v_1+\ldots+v_i}]_{\text{lin}}$. By taking into account that $x_{v_1+\ldots+v_i} = 0$, this simplifies to

$$\alpha_{s_1} \cdot n^{a_{s_1}} \cdot b_{s_1}^n \;=\; \text{coeff}\,(f, x_{\min F(i)}) \cdot \beta_{\min F(i), v_1+\ldots+v_i-1} \cdot n^{v_1+\ldots+v_i-\min F(i)-1}.$$

Here, $\beta_{\min F(i), v_1+\ldots+v_i-1} \in \mathcal{Q}_{\mathcal{S}}[x_{v_1+\ldots+v_i-1}, x_{v_1+\ldots+v_i}]_{\text{lin}}$. But when again taking into account that $x_{v_1+\ldots+v_i} = 0$, in fact we have $\beta_{\min F(i), v_1+\ldots+v_i-1} = c' \cdot x_{v_1+\ldots+v_i-1}$ for some $c' \neq 0$. Hence, $\alpha_{s_1} = 0$ is equivalent to $x_{v_1+\ldots+v_i-1} = 0$, or in other words (as $v_1 + \ldots + v_i - 1 = \min F(i) + a_{s_1}$) to $x_{\min F(i)+a_{s_1}} = 0$.

We repeat this reasoning until we reach an $s' \leq \ell$ with $a_{s'} = a$. Thus, $\alpha_s = 0$ for *all* $s \leq \ell$ with $b_s = b = \lambda_i$ and $a_s \geq a$ is equivalent to $x_j = 0$ for all $\min F(i) + a \leq j \leq v_1 + \ldots + v_i$, i.e., to $\bigwedge_{j \in B_i,\, a+\min F(i) \leq j} (x_j = 0)$.

(b) To ensure that $\alpha_s = 0$ for all $1 \leq s \leq \ell$ where $b_s = 1$ and $a_s \geq 0$, we have to show that this holds if $a_s \geq 1$ and if $a_s = 0$. The former case is equivalent to zero $(1, 1) = \bigwedge_{j \in B_i,\, \min F(i) < j} (x_j = 0)$ according to (c).

If $b_s = 1$ and $a_s = 0$, then (25) and (26) imply

$$\alpha_s = f(0,\dots,0) + \sum_{r\in F(i)} \operatorname{coeff}(f, x_r)\cdot \beta_{r,r},$$

where $\beta_{r,r} \in \mathcal{Q}_{\mathcal{S}}[x_r,\dots,x_{v_1+\dots+v_i}]_{\text{lin}}$. Clearly $r \in F(i)$ implies $r \geq \min F(i)$. Taking into account that $x_j = 0$ for all $\min F(i) < j \leq v_1 + \dots + v_i$, we therefore obtain

$$\alpha_s = f(0,\dots,0) + \operatorname{coeff}(f, x_{\min F(i)})\cdot \beta_{\min F(i),\min F(i)},$$

where $\beta_{\min F(i),\min F(i)} = c_{\min F(i)} \cdot x_{\min F(i)}$. Therefore,

$$\begin{aligned} & \alpha_s = 0 \\ \iff\ & f(0,\dots,0) + \operatorname{coeff}(f, x_{\min F(i)})\cdot c_{\min F(i)}\cdot x_{\min F(i)} = 0 \\ \iff\ & x_{\min F(i)} = M. \end{aligned}$$

A.27 Proof of Lemma 7.21

Proof

(a) As in the proof of Case (a) of Lemma 7.19, we have $\alpha_{s_0} = f(0,\dots,0)$ and thus $\alpha_{s_0} > 0 \iff f(0,\dots,0) > 0$. In this case, for all $s_0 < s \leq \ell$ we must have $b_s > 1$. Hence, $\bigwedge_{s=s_0+1}^{\ell} \left(\alpha_s = 0\right) \iff \bigwedge_{i\in\{1,\dots,k\},\lambda_i>1} \operatorname{zero}(\lambda_i, 0)$ by Lemma 7.19.

(b) Now $(b_{s_0}, a_{s_0}) = (1,0)$, $\lambda_{i_0} = 1$ for some $1 \leq i_0 \leq k$, and $F(i_0) \neq \emptyset$. By Lemma 7.19 we obtain that $\alpha_s = 0$ for all $s_0 < s \leq \ell$ with $b_s = 1$ is equivalent to $\operatorname{zero}(1,1)$, and $\alpha_s = 0$ for all $s_0 < s \leq \ell$ with $b_s > 1$ is equivalent to $\bigwedge_{i=i_0+1}^{k} \operatorname{zero}(\lambda_i, 0)$.

Finally, as in the proof of Case (b) of Lemma 7.19, we get

$$\alpha_{s_0} = f(0,\dots,0) + \operatorname{coeff}(f, x_{\min F(i_0)})\cdot \beta_{\min F(i_0),\min F(i_0)},$$

with $\beta_{\min F(i_0),\min F(i_0)} = c_{\min F(i_0)} \cdot x_{\min F(i_0)}$. For $C = \operatorname{coeff}(f, x_{\min F(i_0)}) \cdot c_{\min F(i_0)}$,

$$\begin{aligned} & \alpha_{s_0} > 0 \\ \iff\ & f(0,\dots,0) + C\cdot x_{\min F(i_0)} > 0 \\ \iff\ & \tfrac{C}{|C|}\cdot x_{\min F(i_0)} + \tfrac{f(0,\dots,0)}{|C|} > 0 \\ \iff\ & \operatorname{sign}(C)\cdot x_{\min F(i_0)} + \tfrac{f(0,\dots,0)}{|C|} > 0. \end{aligned}$$

(c) If $b_{s_0} < 1$, $f(0,\dots,0) \neq 0$, and either $\lambda_i \neq 1$ for all $1 \leq i \leq k$ or $\lambda_{i_0} = 1$ for some $1 \leq i_0 \leq k$ and $F(i_0) = \emptyset$, then $\bigwedge_{s=s_0+1}^{\ell} \left(\alpha_s = 0\right)$ is false. The reason is that $\operatorname{coefs}(p)$ contains $\alpha_s^{(1,0)}$ with $s_0 < s$ and since there is either no $\lambda_i = 1$, or $\lambda_{i_0} = 1$ for some $1 \leq i_0 \leq k$ but $F(i_0) = \emptyset$, we have $\alpha_s = f(0,\dots,0) \neq 0$.

(d) In the remaining case, since $b_{s_0} \neq 1$, there must be an $1 \leq i_0 \leq k$ with $\lambda_{i_0} = b_{s_0}$ and $F(i_0) \neq \emptyset$. By Lemma 7.19, $\alpha_s = 0$ for all $s_0 < s \leq \ell$ with $b_s = b_{s_0} = \lambda_{i_0}$ is equivalent to $\operatorname{zero}(\lambda_{i_0}, a_{s_0} + 1)$, and $\alpha_s = 0$ for all $s_0 < s \leq \ell$ with $b_s \neq b_{s_0}$ is equivalent to $\bigwedge_{i=i_0+1}^{k} \operatorname{zero}(\lambda_i, 0)$. Finally, (25) and (26) from the proof of Lemma 7.19 imply

$$\alpha_{s_0} = \sum_{r \in F(i_0), r \le v_1 + \ldots + v_{i_0} - a_{s_0}} \operatorname{coeff}(f, x_r) \cdot \beta_{r, r+a_{s_0}} \cdot n^{a_{s_0}} \cdot \lambda_{i_0}^n,$$

for $\beta_{r,r+a_{s_0}} \in \mathcal{Q}_{\mathcal{S}}[x_{r+a_{s_0}}, \ldots, x_{v_1+\ldots+v_{i_0}}]_{\text{lin}}$. Clearly $r \in F(i_0)$ implies $r \ge \min F(i_0)$. Considering that $x_j = 0$ for all $\min F(i_0) + a_{s_0} < j \le v_1 + \ldots + v_{i_0}$, we therefore obtain

$$\alpha_{s_0} = \operatorname{coeff}(f, x_{\min F(i_0)}) \cdot \beta_{\min F(i_0), \min F(i_0)+a_{s_0}} \cdot n^{a_{s_0}} \cdot \lambda_{i_0}^n,$$

where $\beta_{\min F(i_0), \min F(i_0)+a_{s_0}}$ is $c_{\min F(i_0)+a_{s_0}} \cdot x_{\min F(i_0)+a_{s_0}}$. Let $sg = \operatorname{sign}(\operatorname{coeff}(f, x_{\min F(i_0)}) \cdot c_{\min F(i_0)+a_{s_0}})$. Thus, $\alpha_{s_0} > 0$ is equivalent to $sg \cdot x_{\min F(i_0)+a_{s_0}} > 0$.

A.28 Proof of Lemma 7.32

Proof For any $0 \le s \le \ell$ we have $\operatorname{cndAssg}(\rho_{f,s}) \subseteq \mathrm{cA}$, where

$$\mathrm{cA} = \{v \in \{0, 1, -1, \star\}^d \mid \forall\, 1 \le i \le k.\ \text{there is at most one } j \in B_i \text{ with } v_j \ne 0\}.$$

Now we over-approximate the cardinality of cA. If $\vec{v} \in \mathrm{cA}$, then for every $1 \le i \le k$, we have $v_j \ne 0$ for at most one $j \in B_i$. So for the values v_j with $j \in B_i$, there are $3 \cdot |B_i| + 1$ possibilities: either exactly one of them is $1, -1$, or $\star$, or we have $v_j = 0$ for all $j \in B_i$. When combining this result for all $1 \le i \le k$ by multiplication, we obtain $|\operatorname{cndAssg}(\rho_{f,m})| \le$

$$|\mathrm{cA}| = \prod_{i=1}^{k} \left(3 \cdot |B_i| + 1\right) = \prod_{i=1}^{k} \left(3 \cdot v_i + 1\right) \le \left(3 \cdot \max\{v_i \mid 1 \le i \le k\} + 1\right)^k.$$

A.29 Proof of Corollary 7.34

Proof We show that $\ell \le d + 1$ for $|\operatorname{coefs}(p)| = \ell$. Then the result follows from Lemma 7.32 and the definition of cndAssg, since $\operatorname{cndAssg}(\operatorname{ic}(p \rhd 0)) \subseteq \bigcup_{s=0}^{\ell} \operatorname{cndAssg}(\rho_{f,s})$. The number of coefficients of p is determined by the number of terms of the form $\alpha \cdot n^a \cdot b^n$ occurring in p. But these terms are connected to the bases $0 < \lambda_1 < \ldots < \lambda_k$ of the hierarchical k-partition: either $b = \lambda_i$ for some $1 \le i \le k$ and $0 \le a \le v_i - 1$ or $a = 0$ and $b = 1$. So we have at most $1 + \sum_{i=1}^{k} v_i = 1 + d$ such terms in p, i.e., $\ell \le d + 1$.

A.30 Proof of Theorem 7.36

Proof *Soundness:* If Algorithm 4 returns $\top$, then there is a $\vec{v} \in \operatorname{cndAssg}(\operatorname{ic}(\varphi(\vec{h})))$ where $\operatorname{ic}(\varphi(\vec{h}))(\vec{v})$ is satisfiable, i.e., a $w \in \mathcal{S}$ where $\vec{v}[\star/w]$ is a model of $\operatorname{ic}(\varphi(\vec{h}))$. So $\exists \vec{x} \in \mathcal{S}^d.\ \operatorname{ic}(\varphi(\vec{h}))$ is valid.

Completeness: Let $\psi = \operatorname{ic}(\varphi(\vec{h}))$ and let $\exists \vec{x} \in \mathcal{S}^d.\ \psi$ be valid, i.e., there is a $\vec{v} \in \mathcal{S}^d$ such that $\psi(\vec{v})$ holds.

We have to prove that there is a $\vec{v}' \in \operatorname{cndAssg}(\operatorname{ic}(\varphi(\vec{h})))$ where $\psi(\vec{v}')\!\downarrow$ is satisfiable. Then the claim follows as the algorithm calls SMT on $\psi(\vec{v}')\!\downarrow$ and thus returns $\top$.

Note that φ is a propositional formula only built from the connectives $\wedge$ and $\vee$, i.e., it does not contain $\neg$. By construction, $\operatorname{ic}(\varphi(\vec{h}))$ results from φ by replacing each atom $f \rhd 0$

by ic $(f(\vec{h}) \triangleright 0)$. So similar to the concept of *fundamental sets* in the proof of Theorem 4.9, there is a subset $\{\text{ic}(f_1(\vec{h}) \triangleright_1 0), \dots, \text{ic}(f_e(\vec{h}) \triangleright_e 0)\}$ of these formulas such that $\vec{v}$ satisfies them all and such that satisfying these formulas is sufficient for satisfying ψ.

Let ℓ_1, ..., ℓ_e be the numbers of coefficients in the poly-exponential expressions $f_1(\vec{h})$, ..., $f_e(\vec{h})$. By Corollary 7.25, ic $(f_r(\vec{h}) \triangleright_r 0)$ has the form $\bigvee_{s=1}^{\ell_r} \rho_{f_r,s}$ or $\bigvee_{s=0}^{\ell_r} \rho_{f_r,s}$ for each $1 \le r \le e$. So for every r there is *at least* one s where $\rho_{f_r,s}(\vec{v})$ is true. But due to the construction of $\rho_{f_r,s}$ in Lemma 7.21 and Corollary 7.23, for every $\vec{v}$ there is *at most* one $0 \le s \le \ell_r$ where $\rho_{f_r,s}(\vec{v})$ is true. Thus, for every $1 \le r \le e$, there is a *unique* $0 \le s_r \le \ell_r$ where $\rho_{f_r,s_r}(\vec{v})$ is true.

Let $B_1, \dots, B_k$ again be the blocks from the k-partition $h_1, \dots, h_d$. Note that if there is a block B_i and some ρ_{f_r,s_r} requires x_j with $j \in B_i$ to be non-zero, then ρ_{f_r,s_r} requires all $x_{j'}$ with $j' > j$ and $j' \in B_i$ to be zero. Thus, since $\vec{v}$ satisfies *all* formulas ρ_{f_r,s_r} for $1 \le r \le e$, for each B_i there is *at most* one $r \in B_i$ where some ρ_{f_r,s_r} requires x_j to be non-zero. Hence, we can assume that for each block B_i there is at most one $j \in B_i$ where $v_j \neq 0$.

(i) Case $\lambda_{i_0} = 1$ for some $1 \le i_0 \le k$: In this case, for the non-zero entries in $\vec{v}$ belonging to indices in B_i with $i \neq i_0$, only their sign is important since the formulas requiring them to be non-zero are interval conditions according to Definition 7.27 (b) resulting from Lemma 7.21 (d). So these entries can be chosen to be 1 or -1. If there is a (unique) non-zero value v_j with $j \in B_{i_0}$, its value is indeed important: the formulas requiring this value to be non-zero have the form of Definition 7.27 (e) (in Lemma 7.21 (b)) or Definition 7.27 (d) (in Lemma 7.21 (d)) or Definition 7.27 (c) (in Corollary 7.23). Thus, let

$$v'_j = \begin{cases} 0, & \text{if } v_j = 0 \\ \star, & \text{if } v_j \neq 0 \wedge j \in B_{i_0} \\ 1, & \text{if } v_j > 0 \wedge j \notin B_{i_0} \\ -1, & \text{if } v_j < 0 \wedge j \notin B_{i_0} \end{cases}$$

As discussed before, $\psi(\vec{v}')\!\downarrow$ is satisfiable as $(\rho_{f_1,s_1} \wedge \dots \wedge \rho_{f_e,s_e})(\vec{v}')\!\downarrow$ is satisfiable by construction: If the formula contains $\star$, then assigning $\star$ the unique non-zero value $v_j \neq 0$ for $j \in B_{i_0}$ is a satisfying assignment.

(ii) Case $\lambda_{i_0} \neq 1$ for all $1 \le i_0 \le k$: In this case, ρ_{f_1,s_1}, ..., ρ_{f_e,s_e} are interval conditions according to Definition 7.27 (a) or (b). Thus, only the sign of the non-zero values in $\vec{v}$ is important to satisfy these formulas. Hence we define

$$v'_j = \begin{cases} 0, & \text{if } v_j = 0 \\ 1, & \text{if } v_j > 0 \\ -1, & \text{if } v_j < 0 \end{cases}$$

Thus, $\psi(\vec{v}')\!\downarrow$ is satisfiable since $(\rho_{f_1,s_1} \wedge \dots \wedge \rho_{f_e,s_e})(\vec{v}')\!\downarrow$ is true by construction.

So in both cases $\vec{v}' \in \bigcup_{1 \le r \le e} \text{cndAssg}\left(\rho_{f_r,s_r}\right) \subseteq \text{cndAssg}(\psi)$, i.e., Algorithm 4 returns $\top$.

Acknowledgements We thank Alberto Fiori for help with the example $\mathbf{L}_{non-pspace}$ from (10) and Arno van den Essen for useful discussions.

Funding Open Access funding enabled and organized by Projekt DEAL. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 235950644 (Project GI 274/6-2), by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 389792660 as part of TRR 248, and by the DFG Research Training Group 2236 UnRAVeL.

Data availability Data sharing is not applicable to this article as no data sets were generated or analyzed during the current study. All missing proofs are included in Appendix A and they are also available in [30].

References

1. Alias C, Darte A, Feautrier P, Gonnord L (2010) Multi-dimensional rankings, program termination, and complexity bounds of flowchart programs. In: Proceedings of the SAS, LNCS 6337, pp 117–133, https://doi.org/10.1007/978-3-642-15769-1_8
2. Basu S, Pollack R, Roy MF (2006) Algorithms in real algebraic geometry. Algorithms and Comp. in Math. 10, Springer https://doi.org/10.1007/3-540-33099-2
3. Basu S, Mishra B (2017) Computational and quantitative real algebraic geometry. In: Goodman JE, O'Rourke J, Tóth CD (eds) Handbook of discrete and computational geometry, 3rd edn. CRC, Boca Raton, pp 969–1002
4. Ben-Amram AM, Genaim S, Masud AN (2012) On the termination of integer loops. ACM Trans Prog Lang Syst 34(4). https://doi.org/10.1145/2400676.2400679
5. Ben-Amram AM, Genaim S (2014) Ranking functions for linear-constraint loops. J ACM 61(4). https://doi.org/10.1145/2629488
6. Ben-Amram AM, Genaim S (2017) On multiphase-linear ranking functions. In: Proceedings of the CAV, LNCS 10427, pp 601–620, https://doi.org/10.1007/978-3-319-63390-9_32
7. Ben-Amram AM, Doménech JJ, Genaim S (2019) Multiphase-linear ranking functions and their relation to recurrent sets. In: Proceedings of the SAS, LNCS 11822, pp 459–480, https://doi.org/10.1007/978-3-030-32304-2_22
8. Bozga M, Iosif R, Konecný F (2014) Deciding conditional termination. Log Methods Comput Sci 10(3). https://doi.org/10.2168/LMCS-10(3:8)2014
9. Bradley AR, Manna Z, Sipma HB (2005) Linear ranking with reachability. In: Proceedings of the CAV, LNCS 3576, pp 491–504, https://doi.org/10.1007/11513988_48
10. Braverman M (2006) Termination of integer linear programs. In: Proceedings of the CAV, LNCS 4144, pp 372–385, https://doi.org/10.1007/11817963_34
11. Brockschmidt M, Cook B, Fuhs C (2013) Better termination proving through cooperation. In: Proceedings of the CAV, LNCS 8044, pp 413–429, https://doi.org/10.1007/978-3-642-39799-8_28
12. Canny JF (1988) Some algebraic and geometric computations in PSPACE. In: Proceedings of the STOC, pp 460–467, https://doi.org/10.1145/62212.62257
13. Cohen PJ (1969) Decision procedures for real and *p*-adic fields. Commun Pure Appl Math 22(2):131–151. https://doi.org/10.1002/cpa.3160220202
14. Dai L, Xia B (2012) Non-termination sets of simple linear loops. In: Proceedings of the ICTAC, LNCS 7521, pp 61–73, https://doi.org/10.1007/978-3-642-32943-2_5
15. Dantzig GB (1949) Programming in a linear structure. Econometrica 17:73–74 (Report of the September 9, 1948 meeting in Madison)
16. Downey RG, Fellows MR (1999) Parameterized complexity. Monographs in computer science. Springer, Berlin. https://doi.org/10.1007/978-1-4612-0515-9
17. van den Essen A, Hubbers E (1996) Polynomial maps with strongly nilpotent Jacobian matrix and the Jacobian conjecture. Linear Algebra Appl 247:121–132. https://doi.org/10.1016/0024-3795(95)00095-X

18. van den Essen A (2000) Polynomial automorphisms and the Jacobian conjecture. Springer, Berlin. https://doi.org/10.1007/978-3-0348-8440-2
19. Frohn F, Giesl J (2019) Termination of triangular integer loops is decidable. In: Proceedings of the CAV, LNCS 11562, pp 269–286, https://doi.org/10.1007/978-3-030-25543-5_24
20. Frohn F, Hark M, Giesl J (2020) Termination of polynomial loops. In: Proceedings of the SAS, LNCS 12389, pp 89–112, https://doi.org/10.1007/978-3-030-65474-0_5
21. Frohn F (2020) A calculus for modular loop acceleration. In: Proceedings of the TACAS, LNCS 12078, pp 58–76, https://doi.org/10.1007/978-3-030-45190-5_4
22. Frumkin MA (1977) Polynomial time algorithms in the theory of linear diophantine equations. In: Proceedings of the FCT, LNCS 56, pp 386–392, https://doi.org/10.1007/3-540-08442-8_106
23. Giesbrecht M (1995) Nearly optimal algorithms for canonical matrix forms. SIAM J Comput 24(5):948–969. https://doi.org/10.1137/S0097539793252687
24. Giesl J, Aschermann C, Brockschmidt M, Emmes F, Frohn F, Fuhs C, Hensel J, Otto C, Plücker M, Schneider-Kamp P, Ströder T, Swiderski S, Thiemann R (2017) Analyzing program termination and complexity automatically with AProVE. J Autom Reason 58(1):3–31. https://doi.org/10.1007/s10817-016-9388-y
25. Giesl J, Rubio A, Sternagel C, Waldmann J, Yamada A (2019) The termination and complexity competition. In: Proceedings of the TACAS, LNCS 11429, pp 156–166, https://doi.org/10.1007/978-3-030-17502-3_10
26. Gomory R (1960) An algorithm for the mixed integer problem. Tech. Rep. RM-2597, The RAND Corporation, https://www.rand.org/pubs/research_memoranda/RM2597.html
27. Graham RL, Knuth DE, Patashnik O (1994) Concrete mathematics: a foundation for computer science, 2nd edn. Addison-Wesley, Boston
28. Hark M, Frohn F, Giesl J (2020) Polynomial loops: Beyond termination. In: Proceedings of the LPAR, EPiC 73, pp 279–297, https://doi.org/10.29007/nxv1
29. Hark M (2021) Towards complete methods for automated complexity and termination analysis of (probabilistic) programs. PhD thesis, RWTH Aachen University, Germany, https://doi.org/10.18154/RWTH-2021-06073
30. Hark M, Frohn F, Giesl J (2022) Termination of triangular polynomial loops. CoRR abs/1910.11588, https://doi.org/10.48550/arXiv.1910.11588
31. Hosseini M, Ouaknine J, Worrell J (2019) Termination of linear loops over the integers. In: Proceedings of the ICALP, LIPIcs 132, https://doi.org/10.4230/LIPIcs.ICALP.2019.118
32. Humenberger A, Kovács L (2021) Algebra-based synthesis of loops and their invariants (invited paper). In: Proceedings of the VMCAI, LNCS 12597, pp 17–28, https://doi.org/10.1007/978-3-030-67067-2_2
33. Kantorovich LV (1960) Mathematical methods of organizing and planning production. Manage Sci 6(4):366–422. https://doi.org/10.1287/mnsc.6.4.366
34. Kauers M, Paule P (2011) The concrete tetrahedron—symbolic sums. Recurrence equations, generating functions, asymptotic estimates. Springer https://doi.org/10.1007/978-3-7091-0445-3
35. Kincaid Z, Breck J, Cyphert J, Reps TW (2019) Closed forms for numerical loops. Proc ACM Progr Lang 3(POPL). https://doi.org/10.1145/3290368
36. Kovács L (2008) Reasoning algebraically about *p*-solvable loops. In: Proceedings of the TACAS, LNCS 4963, pp 249–264, https://doi.org/10.1007/978-3-540-78800-3_18
37. Land AH, Doig AG (1960) An automatic method of solving discrete programming problems. Econometrica 28(3):497–520. https://doi.org/10.2307/1910129
38. Landau E (1909) Handbuch der Lehre von der Verteilung der Primzahlen. Teubner
39. Larraz D, Oliveras A, Rodríguez-Carbonell E, Rubio A (2013) Proving termination of imperative programs using Max-SMT. In: Proceedings of the FMCAD, pp 218–225, https://doi.org/10.1109/FMCAD.2013.6679413
40. Leike J, Heizmann M (2015) Ranking templates for linear loops. Log Methods Comput Sci 11(1). https://doi.org/10.2168/LMCS-11(1:16)2015
41. Lenstra HW Jr (1983) Integer programming with a fixed number of variables. Math Oper Res 8(4):538–548. https://doi.org/10.1287/moor.8.4.538
42. Li Y (2014) A recursive decision method for termination of linear programs. In: Proceedings of the SNC, pp 97–106, https://doi.org/10.1145/2631948.2631966
43. Li Y (2016) Termination of single-path polynomial loop programs. In: Proceedings of the ICTAC, LNCS 9965, pp 33–50, https://doi.org/10.1007/978-3-319-46750-4_3
44. Li Y (2017a) Termination of semi-algebraic loop programs. In: Proceedings of the SETTA, LNCS 10606, pp 131–146, https://doi.org/10.1007/978-3-319-69483-2_8
45. Li Y (2017b) Witness to non-termination of linear programs. Theor Comput Sci 681:75–100. https://doi.org/10.1016/j.tcs.2017.03.036

46. Matijasevič JV (1970) The Diophantineness of enumerable sets. Dokl Akad Nauk SSSR 191:279–282
47. Neumann E, Ouaknine J, Worrell J (2020) On ranking function synthesis and termination for polynomial programs. In: Proceedings of the CONCUR, LIPIcs 171, https://doi.org/10.4230/LIPIcs.CONCUR.2020.15
48. de Oliveira S, Bensalem S, Prevosto V (2016) Polynomial invariants by linear algebra. In: Proceedings of the ATVA, LNCS 9938, pp 479–494, https://doi.org/10.1007/978-3-319-46520-3_30
49. Ouaknine J, Pinto JS, Worrell J (2015) On termination of integer linear loops. In: Proceedings of the SODA, pp 957–969, https://doi.org/10.1137/1.9781611973730.65
50. Pia AD, Dey SS, Molinaro M (2017) Mixed-integer quadratic programming is in NP. Math Program 162(1–2):225–240. https://doi.org/10.1007/s10107-016-1036-0
51. Podelski A, Rybalchenko A (2004) A complete method for the synthesis of linear ranking functions. In: Proceedings of the VMCAI, LNCS 2937, pp 239–251, https://doi.org/10.1007/978-3-540-24622-0_20
52. Renegar J (1992) On the computational complexity and geometry of the first-order theory of the reals, Part I: Introduction. Preliminaries. The geometry of semi-algebraic sets. The decision problem for the existential theory of the reals. J Symb Comput 13(3):255–300. https://doi.org/10.1016/S0747-7171(10)80003-3
53. Robinson J (1949) Definability and decision problems in arithmetic. J Symb Log 14(2):98–114. https://doi.org/10.2307/2266510
54. Roch JL, Villard G (1996) Fast parallel computation of the Jordan normal form of matrices. Parallel Process Lett 06(02):203–212. https://doi.org/10.1142/S0129626496000200
55. Roche DS (2018) What can (and can't) we do with sparse polynomials? In: Proceedings of the ISSAC, pp 25–30, https://doi.org/10.1145/3208976.3209027
56. Rodríguez-Carbonell E, Kapur D (2004) Automatic generation of polynomial loop invariants: Algebraic foundation. In: Proceedings of the ISSAC, pp 266–273, https://doi.org/10.1145/1005285.1005324
57. Schaefer M (2009) Complexity of some geometric and topological problems. In: Proceedings of the GD, LNCS 5849, pp 334–344, https://doi.org/10.1007/978-3-642-11805-0_32
58. Schaefer M, Stefankovic D (2017) Fixed points, Nash equilibria, and the existential theory of the reals. Theory Comput Syst 60(2):172–193. https://doi.org/10.1007/s00224-015-9662-0
59. Sidorov SV (2019) On the similarity of certain integer matrices with single eigenvalue over the ring of integers. Math Notes 105(5):756–762. https://doi.org/10.1134/S0001434619050122
60. Storjohann A, Labahn G (1996) Asymptotically fast computation of Hermite normal forms of integer matrices. In: Proceedings of the ISSAC, pp 259–266, https://doi.org/10.1145/236869.237083
61. Tarski A (1998) A decision method for elementary algebra and geometry. In: Caviness BF, Johnson JR (eds) Quantifier elimination and cylindrical algebraic decomposition, Springer, pp 24–84, https://doi.org/10.1007/978-3-7091-9459-1 originally appeared in 1951, U California Press, Berkeley and Los Angeles
62. Tiwari A (2004) Termination of linear programs. In: Proceedings of the CAV, LNCS 3114, pp 70–82 https://doi.org/10.1007/978-3-540-27813-9_6
63. TPDB (2003–2021) Termination Problems Data Base. http://www.termination-portal.org/wiki/TPDB
64. Wu B, Shen L, Bi Z, Zeng Z (2010) Termination of loop programs with polynomial guards. In: Proceedings of the ICCSA, LNCS 6019, pp 482–496, https://doi.org/10.1007/978-3-642-12189-0_42
65. Xia B, Zhang Z (2010) Termination of linear programs with nonlinear constraints. J Symb Comput 45(11):1234–1249. https://doi.org/10.1016/j.jsc.2010.06.006
66. Xia B, Yang L, Zhan N, Zhang Z (2011) Symbolic decision procedure for termination of linear programs. Formal Aspects Comput 23(2):171–190. https://doi.org/10.1007/s00165-009-0144-5
67. Xu M, Li ZB (2013) Symbolic termination analysis of solvable loops. J Symb Comput 50:28–49. https://doi.org/10.1016/j.jsc.2012.05.005