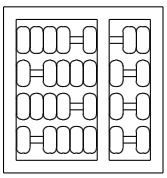

INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Automated Generation of Asymptotically Non-Terminant Initial Variable Values for Linear Programs

Rachid Rebiha *Arnaldo V. Moura*
Nadir Matringe

Technical Report - IC-14-03 - Relatório Técnico

January - 2014 - Janeiro

Automated Generation of Asymptotically Non-Terminant Initial Variable Values for Linear Programs

Rachid Rebiha* Arnaldo Vieira Moura† Nadir Matringe ‡

Abstract

We present the new notion of *asymptotically non-terminant initial variable values* for linear loop programs. Those specific values are directly associated to initial variable values for which the corresponding program does not terminate. Our theoretical contributions provide us with powerful computational methods for automatically generating sets of asymptotically non-terminant initial variable values. Moreover, we reduce the termination problem of linear programs to the emptyness check of a specific set of asymptotically non-terminant initial variable values. Those sets are represented symbolically and exactly by a semi-linear space, *e.g.*, conjunctions and disjunctions of linear equalities and inequalities. Also, by taking the complement of these sets, we obtain a precise under-approximations of input sets for which linear programs do terminate.

1 Introduction

Research on formal methods for program verification [12, 15, 8, 17] aims at discovering mathematical techniques and developing their associated algorithms to establish the correctness of software, hardware, concurrent systems, embedded systems or hybrid systems. Static program analysis [12, 15], is used to check that a software is free of defects, such as buffer overflows or segmentation faults, which are safety properties, or termination and non-termination, which are liveness properties. Proving termination of while loop programs is necessary for the verification of liveness properties that any well behaved and engineered system, or any safety critical embedded system, must

*Instituto de Computação, Universidade Estadual de Campinas, 13081970 Campinas, SP. Pesquisa desenvolvida com suporte financeiro da FAPESP, processos 2011089471 e FAPESP BEPE 2013047349

†Instituto de Computação, Universidade Estadual de Campinas, 13081970 Campinas, SP.

‡Université de Poitiers, Laboratoire Mathématiques et Applications, France.

guarantee. We could list here many verification approaches that are only practical depending on the facility with which termination can be automatically determined. Verification of temporal properties of infinite state systems [22] is another example.

Recent work on automated termination analysis of imperative loop programs has focused on a partial decision procedure based on the discovery and synthesis of ranking functions. Such functions map the loop variable to a well-defined domain where their value decreases further at each iteration of the loop [9, 10]. Several interesting approaches, based on the generation of *linear* ranking functions, have been proposed [3, 4] for loop programs where the guards and the instructions can be expressed in a logic supporting linear arithmetic. For the generation of such functions, there are effective heuristics [14, 10] and, in some cases, there are also complete methods for the synthesis of such functions [16]. On the other hand, it is easy to generate a simple linear terminant loop program that does not have a linear ranking function. And in this case such complete synthesis methods [16] fail to provide a conclusion about the termination or the non termination of such a program.

In this work we address the non-termination problem for linear while loop programs. In other words, we consider the class of loop programs where the loop condition is a conjunction of linear inequalities and the assignments to each of the variables in the loop instruction block are affine or linear forms. In matrix notation, *linear loop programs* will be represented in a general form as: while $(Vx > b)$, $\{x := Ax + c\}$, where A and V are matrices, b and c are vectors of real numbers, and that x is a vector of variables. Without loss of generality, we show that the termination/non-termination analysis for such a class of linear programs could be reduced to the problem of termination/non-termination for homogeneous linear programs. The latter being programs where linear assignments consist of homogeneous expressions and where the linear loop condition consists of at most one inequality. Concerning effective program transformations and simplification techniques, non-termination analysis for programs presented in a more complex form can often be reduced to an analysis of a program expressed in this basic affine form. Despite tremendous progress over the years [6, 5, 7, 13, 11, 2, 1], the problem of finding practical, sound and complete methods for determining or analyzing non-termination remains very challenging for this class of programs, considering all initial variable values. We started our investigation from our preliminary technical reports [20, 18] where we introduced a termination analysis in which algorithms ran in polynomial time complexity. In [21] and [19], we approached the problem of generating the asymptotically non-terminant initial variable values set for a more restricted class of linear programs with only one loop condition, and where the associated linear forms of the loop had to lead to diagonalizable systems with no complex eigenvalues.

Here, we introduce new static analysis methods that compute automatically, and in polynomial time complexity, the set of initial input variable values for which the program does not terminate, and also a set of initial inputs values for which the

program does terminate. This justifies the innovation of our contributions, *i.e.*, none of the other mentioned related work is capable of generating such critical information over non-terminating loops. We summarize our contributions as follows:

- To the best of our knowledge, we introduce a new key notion for non-termination and termination analysis for loop programs: we identify the important concept of *asymptotically non-terminant initial variable values*, *ANT* for short. Any asymptotically non-terminant initial variable value is directly related to an initial variable value for which the considered program does not terminate.
- Considering linear affine loop programs with a loop condition composed by m conjunctions of linear inequalites, we reduce the termination/non-termination problem for such class of linear programs to the emptyness check of the *ANT* set of specific homogeneous linear programs.
- Our theoretical contributions are *necessary and sufficient condition* for the termination of linear programs. They provide us with efficient computational methods allowing the exact computation of the ANT set for a given linear loop program. We generate automatically a set of linear equalities and inequalities describing a semi-linear space that represents symbolically and exactly the set of all asymptotically non-terminant initial variable values. The set of *ANT* values contains the set of non-terminant initial variable values. On the other hand the complementary set of *ANT* values is a precise under-approximation of the set of terminant inputs for the same program. Looking at the affine forms associated to the loop, we are able to handle the case where they induce non-diagonalizable systems with complex eigenvalues.
- These contributions are rigorously stated with proofs that are quite technical. In a practical static termination analysis, however, one can focus directly on the provided ready-to-use generic formulas representing the ANT set for the class of affine programs.

Example 1.1. *(Motivating Example)* Consider the following program depicted below on the left. We show part of the output of our algorithm below on the right.

(i) Pseudo code:

```
while(x−1/2y−2z>0){
  x:= −20x−9y+75z;
  y:= 7x+8y−21z;
  z:= −7x−3y+26z;}
```

(ii) The initial variable values of x, y and z are represented respectively by the parameters u_1, u_2 and u_3. Output of our prototype:

```
Locus of ANT
[[u1 < -u2 + 3*u3]]
OR
[[u1 == -u2 + 3*u3, -u3 < u2]]
OR
[[u1 == 4*u3, u2 == -u3, 0 < u3]]
```

As mentioned before, the problem of termination is reduced to the emptyness check of the ANT set. The semi-linear $ANT = \{u = (u_1, u_2, u_3)^\top \in E \mid u_1 < -u_2 + 3 * u_3\} \cup \{u = (u_1, u_2, u_3)^\top \in E \mid u_1 = -u_2 + 3 * u_3 \wedge -u_3 < u_2\} \cup \{u = (u_1, u_2, u_3)^\top \in E \mid u_1 = 4 * u_3 \wedge u_2 = -u_3 \wedge 0 < u_3\}$ represents symbolically all asymptotically initial variable values that are directly associated to initial variable values for which the program does not terminate. On the other hand, the complement set, $ANT^{\mathcal{C}}$, is a precise under-approximation of the set for all initial variable values on which the program terminates. □

The rest of this article is organized as follows. Section 2 is a preliminary section where we introduce some key notions and results from linear algebra, which will be used to build our computational methods. In this section, we also present our computational model for programs and some further notations. Section 3 introduces the new notion of *asymptotically non-terminant initial variable values* and their important associated results for termination analysis. In Section 4 we treat the more general case where the induced systems have complex eigenvalues. In Section 5 we study the ANT sets for generalized homogeneous and affine linear programs. In Section 6 we present our computational method that generate a symbolic representation of the asymptotically non-terminant variable values for linear programs over the reals, and in Section 7 we further generalize our method. We provide a complete discussion in Section 8. Finally, Section 9 states our conclusions.

2 Linear Algebra and Linear Loop Programs

Here, we present key linear algebraic notions and results which are central to the theoretical and algorithmic development of our methods.

Let E be a real vector space, and let $\mathbf{a}$ belong to $End_{\mathbb{R}}(E)$, the latter being the space of linear maps from E to itself. The notation $Mat_B(\mathbf{a})$ describes the matrix representation of the form $\mathbf{a} \in End_{\mathbb{R}}(E)$ expressed in a basis B of E. We write $B_C = (e_1, \ldots, e_n)$, with n a positive integer, for the canonical basis of E. We denote by E^* the space of linear maps from E to $\mathbb{R}$. The Kernel of $\mathbf{a}$, also called its *nullspace*, denoted by $Ker(\mathbf{a})$, is $Ker(\mathbf{a}) = \{x \in E \mid \mathbf{a}(x) = 0_E\}$, where 0_E denotes the null element of E. We denote by $\mathcal{M}(m, n, \mathbb{K})$ the set of $m \times n$ matrices with entries in $\mathbb{K}$, and write simply $\mathcal{M}(n, \mathbb{K})$ when $m = n$. In matrix notation, the Kernel of $A \in \mathcal{M}(m, n, \mathbb{K})$ is $Ker(A) = \{v \in \mathbb{K}^n \mid A \cdot v = 0_{\mathbb{K}^m}\}$. Let A be a $n \times n$ square matrix with entries in $\mathbb{K}$. A nonzero vector $x \in \mathbb{K}$ is an eigenvector of A associated with the eigenvalue $\lambda \in \mathbb{K}$ if $A \cdot x = \lambda x$, that is, $(A - \lambda I_n) \cdot x = 0$, where I_n is the $n \times n$ identity matrix. The nullspace of $(A - \lambda I_n)$ is called the *eigenspace* of A associated with eigenvalue λ.

Definition 2.1. *The eigenvalues, eigenvectors and eigenspaces of a form* $\mathbf{a} \in End_{\mathbb{R}}(E)$ *are those obtained considering its matrix representation* $Mat_{B_C}(\mathbf{a}) \in$

$\mathcal{M}(n, \mathbb{K})$. *The characteristic polynomial of* $\mathbf{a}$*, denoted by* $\chi_{\mathbf{a}}$*, is the polynomial over the scalar* $\lambda \in \mathbb{K}$ *defined by* $\mathsf{det}(Mat_{B_C}(\mathbf{a}) - \lambda I_n)$ *where* det *is the standard matrix determinant function. We denote by* $Spec(\mathbf{a})$ *the set of its real roots, which are the real eigenvalues of* $\mathbf{a}$*. We denote by* $E_\lambda(\mathbf{a})$ *the generalized eigenspace of* $\mathbf{a}$ *attached to* λ*, which is* $Ker((\mathbf{a} - \lambda I_n)^{d_\lambda})$*, of dimension* d_λ*.* □

We recall the Jordan basis theorem for $\mathbf{a}$'s restriction to $E_\lambda(\mathbf{a})$. If $A_1, \ldots, A_r$ are square matrices in $\mathcal{M}(n_1, \mathbb{R}), \ldots, \mathcal{M}(n_r, \mathbb{R})$, respectively, we write $diag(A_1, \ldots, A_r)$ for the corresponding diagonal block matrix in $\mathcal{M}(n_1 + \cdots + n_r, \mathbb{R})$.

Theorem 2.1. *Let* λ *belong to* $Spec(\mathbf{a})$*. There is a basis* J_λ *of* $E_\lambda(\mathbf{a})$ *such that* $Mat_{J_\lambda}(\mathbf{a}) = diag(U_{\lambda,1}, \ldots, U_{\lambda,r_\lambda})$*, where each* $U_{\lambda,i}$ *is of the form* $\begin{pmatrix} \lambda & 1 & & & & \\ & \lambda & 1 & & & \\ & & \ddots & \ddots & & \\ & & & \lambda & 1 & \\ & & & & \lambda & 1 \\ & & & & & \lambda \end{pmatrix}$. □

Throughout the paper we write $< , >$ for the canonical scalar product on E, which is given by $< x, y >= \sum_{i=1}^{n} x_i y_i$, and recall that, as it is standard in static program analysis, a primed symbol x' refers to the next state value of x after a transition is taken.

In the following we show how we represent the linear and affine loop programs in terms of linear forms and their matrix representations. First, we present *transition systems* as representations of imperative programs and *automata* as their computational models.

Definition 2.2. *A* transition system *is given by* $\langle x, L, \mathcal{T}, l_0, \Theta \rangle$*, where* $x = (x_1, ..., x_n)$ *is a set of variables,* L *is a set of locations and* $l_0 \in L$ *is the initial location. A* state *is given by an interpretation of the variables in* x*. A* transition $\tau \in \mathcal{T}$ *is given by a tuple* $\langle l_{pre}, l_{post}, q_\tau, \rho_\tau \rangle$*, where* l_{pre} *and* l_{post} *designate the pre- and post- locations of* τ*, respectively, and the transition relation* ρ_τ *is a first-order assertion over* $x \cup x'$*. The transition guard* q_τ *is a conjunction of inequalities over* x*.* Θ *is the initial condition, given as a first-order assertion over* x*. The transition system is said to be* linear *when* ρ_τ *is an affine form.* □

We will use the following matrix notations to represent loop programs and their associated transitions systems.

Definition 2.3. *Let* $P = \langle x = (x_1, ..., x_n), l_0, \mathcal{T} = \langle l, l, q_\tau, \rho_\tau \rangle, l_0, \Theta \rangle$ *be a loop program. We say that* P *is a* linear loop program *if the following conditions hold:*

- *Transition guards are conjunctions of linear inequalities. We represent the loop condition in matrix form as* $Vx > b$ *where* $V \in \mathcal{M}(m, n, \mathbb{R})$ *and* $b \in \mathbb{R}^m$. *By* $Vx > b$ *we mean that each coordinate of vector* Vx *is greater than the corresponding coordinate of vector* b.

- *Transition relations are affine or linear forms. We represent the linear assignments in matrix form as* $x := Ax + c$, *where* $A \in \mathcal{M}(n, \mathbb{R})$ *and* $c \in \mathbb{R}^n$.

The linear loop program $P = P(A, V, b, c)$ *will be represented in its most general form as* **while** $(Vx > b),\ \{x := Ax + c\}$. □

In the rest of this article, we may also use matrix notations in place of their equivalent notations in terms of linear forms. For instance, the program $P(A, v)$ will also be identified and denoted as $P(\mathbf{a}, \mathbf{v})$ where $Mat_B(\mathbf{a}) = A$ and $Mat_B(\mathbf{v}) = v$ in a choosen basis B.

We recognize the following classification for linear loop programs:

Definition 2.4. *We identify the following three types of linear loop programs, from the more specific to the more general forms:*

- Homogeneous*: We denote by* $P^{\mathbb{H}}$ *the set of programs where all linear assignments consist of* homogeneous *expressions, and where the linear loop condition consists of at most one inequality. If* P *is in* $P^{\mathbb{H}}$, *then* P *will be written in matrix form as* **while** $(< v, x >> 0),\ \{x := Ax\}$, *where* v *is a* $(n \times 1)$*-vector corresponding to the loop condition, and where* $A \in \mathcal{M}(n, \mathbb{R})$ *is related to the list of assignments in the loop. We say that* P *has a* homogeneous *form and it will also be denoted as* $P(A, v)$.

- Generalized Condition*: We denote by* $P^{\mathbb{G}}$ *the set of linear loop programs where the loop condition is* generalized *to a conjunction of multiple linear homogeneous inequalities. Also the loop inequalities and assignments remain as homogeneous expressions. If* P *is in* $P^{\mathbb{G}}$ *then* P *will be interpreted as* **while** $(Vx > 0),\ \{x := Ax\}$ *where* V *is a* $(m \times n)$*-matrix corresponding to the loop condition. We say that* P *is in a* generalized loop condition *form and it will be identified as* $P(A, V)$.

- Affine Form*: We denote by* $P^{\mathbb{A}}$ *the set of loop programs where the inequalities and the assignments are generalized to affine expressions. If* P *is in* $P^{\mathbb{A}}$, *it will be written as* **while** $(Vx > b),\ \{x := Ax + c\}$, *for* A *and* V *in* $\mathcal{M}(n, \mathbb{R})$, $b \in \mathbb{R}^m$, *and* $c \in \mathbb{R}^n$. *We say that* P *is in an* affine *form and it will be identified by the signature* $P(A, V, b, c)$.

□

In the following section, we show that the static analysis for the general class of linear affine programs $P^{\mathbb{A}}$ can be reduced to the problem of termination and non-termination for homogeneous linear programs in $P^{\mathbb{H}}$.

3 Asymptotically Non-terminant Variable Values

In this section we present the new notion of *asymptotically non-terminant* variable values. It will prove to be central in the analysis of non termination, in general. We provide the formal definitions of an ANT set and give the first important results dealing with homogeneous linear programs.

Let A be a matrix in $\mathcal{M}(n, \mathbb{R})$, and v be a vector in E. Consider the program $P(A, v)$: while $(< v, x >> 0)$, $\{x := Ax\}$, which takes values in E. We first give a result about the termination of such a program.

Lemma 3.1. *The program $P(A, v)$ is terminating on the input $x \in E$ if and only if $< v, A^k(x) >$ is not positive, for some $k \geq 0$.* □

Proof. If the program $P(A, v)$ performs $k \geq 0$ loop iterations from the initial variables $x \in E$, we obtain $x := A^k(x)$. Thus, if $< v, A^k(x) > \leq 0$, the loop condition is violated, and so $P(a, v)$ terminates. Conversely, if $P(a, v)$ terminates, then the loop condition is violated after a number $k \geq 0$ of iterations. And so $< v, A^k(x) > \leq 0$. □

Next, we introduce the important notion of asymptotically non terminating value.

Definition 3.1. *We say that $x \in E$ is an asymptotically non terminating value for $P(A, v)$ if there exists $k_x \geq 0$ such that $P(A, v)$ is non terminating on $A^{k_x}(v)$. We will write that x is ANT for $P(A, v)$, or simply that x is ANT. We will also write that $P(A, v)$ is ANT on x.* □

Note that if $P(A, v)$ is non terminating on $A^{k_x}(x)$ then $< v, A^k(x) >$ is positive for $k \geq k_x$.

The following result follows directly from the previous definition.

Corollary 3.1. *An element $x \in E$ is ANT if and only if $< v, A^k(x) >$ is positive for a k large enough.* □

The following example illustrates the definition and properties of an ANT inputs.

Example 3.1. Consider the motivating example depicted in Ex. 1.1 Section 1. It is easy to check that the initial variable values $(u_1, u_2, u_3) = (-9, 3, -2)$ belong to the computed ANT set. Thus the input $u_0 = (-9, 3, -2)^\top$ is an ANT point for that program. But the program terminates on u_0 because with those initial variable values, the loop condition is violated the first time the loop location is reached, so

that no loop iteration will be performed, because $< v, A^0(u_0) >= -13/2$. From definition 3.1, we know that there exists $k_{u_o} \geq 0$ such that $P(A, v)$ is non terminating on $A^{k_{u_0}}(v)$. Here we have $< v, A^1(u_0) >= -5/2$, *i.e.*, the program is terminating on $A^1(u_0) = (3, 3, 2)^\top$. But $< v, A^2(u_0) >> 0$ and the program is not terminating on $A^2(u_0) = (63, 3, 22)^\top$. In other words, the inputs $u_0 = (-9, 3, -2)^\top$ is an ANT point for $P(A, v)$ with $k = 2$. □

We denote by $ANT(P(A, v))$ the set of ANT points for the program $P(A, v)$. If the set $ANT(P(A, v))$ of ANT points is not empty, we say that the program $P(A, v)$ is ANT. We will also write NT for non terminant. The following theorem already shows the importance of such a set: the problem of termination for linear programs is reduced to an emptiness check on the ANT set.

Theorem 3.1. *A program $P(A, v)$ in $P^{\mathbb{H}}$ is NT if and only if it is ANT, that is, $ANT(P(A, v)) \neq \emptyset$.* □

Proof. It is clear that NT implies ANT, that is, $NT \subseteq ANT$, as a NT point of $P(A, v)$ is of course ANT, with $k_x = 0$. Conversely, if $P(A, v)$ is ANT, let x be an ANT point. Then $A^{k_x}(x)$ is a NT point for $P(A, v)$, and so $P(A, v)$ is NT. □

As one can easily see, the set of NT points is included in the set of ANT points. But the most important property of the ANT set is the fact that each of its points provides us with an associated element in NT for the corresponding program. In other words, each element x in the ANT set, even if it does not necessarily belong to the NT set, refers directly to initial variable values $y_x = A^{k_x}(x)$ for which the program does not terminate, *i.e.*, y_x is an NT point. We can say that there exists a number k_x of loop iterations, departing from the initial variable values x, such that $A^{k_x}(x)$ correspond to initial variable values for which $P(A, v)$ does not terminate. But, this does not necessarily imply that x is also an NT point for $P(A, v)$. In fact, program $P(A, v)$ could terminate on the initial variable values x by performing a number of loop iterations strictly smaller than k_x. The ANT set is more than an over-approximation of the set of NT points. It also provides us with a deterministic and efficient way to decide termination for the program.

We denote by $ANT^{\mathcal{C}}$ the complement of the ANT set. This complement set provides us with a quite precise under approximation of the set of all initial variable values for which the program terminates.

Corollary 3.2. *Let $P(A, V)$ be a program in $P^{\mathbb{H}}$. Consider the set of ANT initial variable values $ANT(P(A, v))$ and its complement set $ANT^{\mathcal{C}}$. Then $P(A, v)$ is terminant on all x in $ANT^{\mathcal{C}}$.* □

Proof. We know that $NT \subseteq ANT$. So, we have $NT \cap ANT^{\mathcal{C}} = \emptyset$. □

In this section, the set $Spec(\mathbf{a})$ of eigenvalues involved was supposed to be over the reals. In the following section we remove this restriction.

4 The ANT set and complex eigenvalues

In this section, we deal with the general case where there is no restriction on $Spec(\mathbf{a})$. In other words $Spec(\mathbf{a})$ can contain complex eigenvalues that are not reals. As the main contribution of this section, we show that the non-real eigenvalues of $Spec(\mathbf{a})$ do not affect the static termination analysis for linear homogeneous programs. In other words, we prove that the problem of checking automatically and statically the termination of linear programs is completely reduced to an analysis considering only the real eigenvalues in $Spec(\mathbf{a})$. We, thus, provide a complete and deterministic procedure to verify statically and automatically the termination of linear loop programs. In the following we write ANT^r as the ANT subset restricted to the real eigenvalues in $Spec(\mathbf{a})$ and ANT^{nr} as the subset restricted to the complex eigenvalues that are not real. We reduce the problem of termination for linear programs to the emptiness check of the ANT^r subset. We can always compute the ANT subset ANT^r. See Section 3. Also, by taking the complement set of ANT^r we obtain a set of initial variable values for which programs do terminate.

Recall Definition 2.1. In $\mathbb{R}[X]$, the characteristic polynomial $\chi_{\mathbf{a}}$ factors uniquely as the product $\prod_{\lambda \in Spec(\mathbf{a})}(X-\lambda)^{d_\lambda}\chi_{\mathbf{a}}^{nr}$, where $\chi_{\mathbf{a}}^{nr}$ has no real roots. We denote by $\chi_{\mathbf{a}}^{+}$ the product $\prod_{\lambda>0 \in Spec(\mathbf{a})}(X-\lambda)^{d_\lambda}$, and by $\chi_{\mathbf{a}}^{-}$ the product $\prod_{\lambda<0 \in Spec(\mathbf{a})}(X-\lambda)^{d_\lambda}$.

Definition 4.1. *We denote by E^+ the space $Ker(\chi_{\mathbf{a}}^{+}(\mathbf{a}))$ and by E^- the space $Ker(\chi_{\mathbf{a}}^{-}(\mathbf{a}))$. We further write E^r for the space $E^+ \oplus E_0(\mathbf{a}) \oplus E^-$ and E^{nr} for the space $Ker(\chi_{\mathbf{a}}^{nr}(\mathbf{a}))$.* □

The space E^r is based on the reals and E^{nr} has no real elements. Considering the products forms of $\chi_{\mathbf{a}}^{+}$ and $\chi_{\mathbf{a}}^{+}$, we obtain the following complete decomposition for E. First, we recall without proof the following classical decompositions from basic linear algebra.

Proposition 4.1. *$E^+ = \oplus_{\lambda>0 \in Spec(\mathbf{a})}E_\lambda(\mathbf{a})$, $E^- = \oplus_{\lambda<0 \in Spec(\mathbf{a})}E_\lambda(\mathbf{a})$, and $E = E^r \oplus E^{nr}$.* □

All the spaces $E_\lambda(\mathbf{a})$, E^+, E^-, E^r and E^{nr} are $\mathbf{a}$-stable.

Next, we give necessary and sufficient conditions for $P(\mathbf{a}, \mathbf{v})$ to be terminant.

Theorem 4.1. *Let E be an $\mathbb{R}$-vector space of finite dimension, let $\mathbf{a}$ be an endomorphism of E, and $\mathbf{v}$ be a nonzero linear form on E. There exists a vector $x \in E$, such that $\mathbf{v}(\mathbf{a}^k(x)) > 0$ for all $k \geq 0$, if and only if there is $\lambda > 0 \in Spec(\mathbf{a})$, such that $E_\lambda(\mathbf{a}) \not\subset Ker(f)$.* □

Combined with Theorem 4.1, Proposition 7.2 has the following important consequence.

Proposition 4.2. *Let* $\mathbf{v}$ *belong to* $\mathcal{L}(E,\mathbb{R})$*. If the program* $P(\mathbf{a},\mathbf{v})$ *is asymptotically non terminating on* $x = x^+ + x'$*, where* $x^+ \in E^+$ *and* $x' \in E' = E^- \oplus E_0(\mathbf{a}) \oplus E^{nr}$*, it is asymptotically non terminating on* x^+*.* □

Proof. We write $x^+ = x_1 + \cdots + x_t$, with $x_i \in E_{\lambda_i}(\mathbf{a})$. We conclude that there are polynomials $P_1, \ldots, P_t$ in $\mathbb{R}[X]$, such that

$$\mathbf{v}(\mathbf{a}^k(x)) = \lambda_1^k P_1(k) + \cdots + \lambda_t^k P_t(k).$$

Let k_0 be the smallest integer i such that P_i is nonzero, and set $\lambda = \lambda_i$. Then, for k large, $\mathbf{v}(\mathbf{a}^k(x))$ becomes equivalent to $a\lambda^k k^m$, for a the leading coefficient of P_{k_0}, and m the degree of P_{k_0}. On the other hand, according to Theorem 4.1, program $P(\mathbf{a},\mathbf{v})$ is terminating on E', and hence on x'. So, more generally, it is terminating on any $\mathbf{a}^l(x')$, for $l \geq 0$, because E' is $\mathbf{a}$-stable. In particular, there are infinitely many $l \geq 0$ such that $\mathbf{a}^l(x^-)$ is ≤ 0. This implies that if $P(\mathbf{a},\mathbf{v})$ is asymptotically non terminating on $x = x^+ + x'$, then we have $a > 0$, and thus $P(\mathbf{a},\mathbf{v})$ is asymptotically non terminating on x^+. □

Using the following result we can refine Proposition 4.2.

Theorem 4.2. *If the program* $P(\mathbf{a},\mathbf{v})$ *is asymptotically non terminating on* $x = x^r + x^{nr}$*, with* $x^r \in E^r$ *and* $x^{nr} \in E^{nr}$*, it is asymptotically non terminating on* x^r*.* □

Proof. Indeed, if one considers $\mathbf{a}^2$ instead of $\mathbf{a}$, then x^r is equal to x^+ for $\mathbf{a}^2$, and x^{nr} corresponds to x' for $\mathbf{a}^2$. As $P(\mathbf{a},\mathbf{v})$ is ANT on x, so is $P(\mathbf{a}^2,\mathbf{v})$. But according to the previous proposition, this means that $P(\mathbf{a}^2,\mathbf{v})$ is ANT on x^+. Similarly, $\mathbf{a}(x^r) = \mathbf{a}(x)^r$ is equal to $\mathbf{a}(x)^+$ for $\mathbf{a}^2$, and $\mathbf{a}(x^{nr}) = \mathbf{a}(x)^{nr}$ is equal to $\mathbf{a}(x)'$ for $\mathbf{a}^2$. Again, $P(\mathbf{a}^2, f)$ is ANT on $\mathbf{a}(x)$. Hence, by the same argument, it is ANT on $\mathbf{a}(x^r)$. We thus conclude that $P(\mathbf{a},\mathbf{v})$ is ANT on x^r. □

This has as a corollary the following theorem.

Theorem 4.3. *If one denotes by* $\mathbf{a}^r$ *the restriction of* u *to* E^r*, and by* $\mathbf{v}^r$ *the restriction of* $\mathbf{v}$ *to* E^r*, and if one writes*

$$ANT^r(P(\mathbf{a},\mathbf{v})) = ANT(P(\mathbf{a},\mathbf{v})) \cap E^r,$$

then $ANT(P(\mathbf{a},\mathbf{v}))$ *is non empty iff* $ANT^r(P(\mathbf{a},\mathbf{v}))$ *is non empty. In particular, the program* $P(\mathbf{a},\mathbf{v})$ *terminates iff the program* $P(\mathbf{a}^r,\mathbf{v}^r)$ *terminates. In any case, one has*

$$ANT^r(P(\mathbf{a},\mathbf{v})) = ANT(P(\mathbf{a}^r,\mathbf{v}^r)).$$

□

Proof. It is clear that if $ANT^r(P(\mathbf{a}, \mathbf{v}))$ is non empty, then $ANT(P(\mathbf{a}, \mathbf{v}))$ is not. The converse is a consequence of Theorem 4.2. It remains to prove the last assertion, but it follows almost by definition. □

Example 4.1. Consider the following program depicted below:

```
while (3 t+7s+x−1/2y−2z >0){
  t:= t−s ;
  s:= t+s ;
  x:= −20x−9y+75z ;
  y:= 7x+8y−21z ;
  z:= −7x−3y+26z ;}
```

The associated matrices are: $A' = \begin{pmatrix} 1 & -1 & \\ 1 & 1 & \\ & & A \end{pmatrix}$ and $v' = \begin{pmatrix} 3 \\ 7 \\ v \end{pmatrix}$, where the submatrix A and subvector v are the same than the one obtained in Example 3.1.

As the submatrix $\begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$ has only complex eigenvalues not over the reals, Theorem 4.3 says that $ANT^r(P(A', v')) = ANT(P(A, v)$. The set $ANT(P(A, v))$ has already been computed in Example 3.1, that is why we obtain similar conditions. If the initial variable values of t, s, x, y and z are represented respectively by the parameters u_1, u_2, u_3, u_4 and u_5, then we obtain conditions similar to the one obtained in Example 3.1. But this time the ANT locus is over the parameters u_3, u_4 and u_5: [[u3 < -u4 + 3*u5]] OR [[u3 == -u4 + 3*u5, -u5 < u4]] OR [[u3 == 4*u5, u4 == -u5, 0 < u5]]. □

5 The ANT set for general linear affine programs

In Section 5.1, without any restriction on $Spec(\mathbf{a})$, we give a deterministic procedure to generate the ANT set and to check the termination of programs $P(\mathbf{a}, \mathbf{v}_1, \ldots, \mathbf{v}_m) \in P^{\mathbb{H}}$ involving several linear forms $\mathbf{v}_i$. In Section 5.2, we further generalize our results to the complete class of affine programs $P(A, V, b, c) \in P^{\mathbb{A}}$.

5.1 Generalized homogeneous linear programs

We give a deterministic procedure for the termination problem and the non-termination analysis for linear homogeneous programs with generalized loop conditions $P(A, V) \in P^{\mathbb{G}}$. Recall Definition 2.4. Here, the loop condition is generalized to a conjunction of multiple linear homogeneous inequalities. Let $P(A, V)$ be in $P^{\mathbb{G}}$. Recall that $P(A, V)$ is interpreted as while $(Vx > 0)$, $\{x := Ax\}$ where V is

a $(m \times n)$-matrix corresponding to the generalized loop condition. In the following, if the $(n \times 1)$-vectors $v_1, \ldots v_m$ describe the rows of V, we also denote the program $P(A, V)$ as $P(A, v_1, \ldots, v_m)$. First, we formalize the notion of termination for programs in $P^{\mathbb{G}}$.

Lemma 5.1. *Let $P(A, V)$ be a program in $P^{\mathbb{G}}$ and let $v_1, \ldots, v_m$ be the vectors describing the m rows of V. The program $P(A, V)$ is terminating on the input $x \in E$ if and only if the programs $P(A, v_i)$s are terminating on x, for all $i \in \{1, \ldots, m\}$.* □

Proof. If $P(A, V)$ is terminating on $x \in E$ then exists an integer k_x such that

$$V \cdot A^{k_x}(x) \leq 0.$$

So, $< v_i, A^{k_x}(x) >\leq 0$ for all $i \in \{1, \ldots, m\}$ which is equivalent to $P(A, v_i)$ being terminating on x, for all $i \in \{1, \ldots, m\}$. See Lemma 3.1. Conversely, if the $P(A, v_i)$ are terminating on x for all $i \in \{1, \ldots, m\}$, then there exist $k_1, \ldots, k_m$ such that

$$< v_i, A^{k_i}(x) >\leq 0,$$

for all $i \in \{1, \ldots, m\}$. See Lemma 3.1 again. Taking

$$k_x = max\{k_1, \ldots k_m\}$$

we have $V \cdot A^{k_x}(x) \leq 0$ and $P(A, V)$ is terminating on x. □

Now, we define the notion of ANT inputs for linear programs with generalized loop conditions.

Definition 5.1. *Let $P(A, v_1, \ldots, v_m)$ be a program in $P^{\mathbb{G}}$. We say that $x \in E$ is ANT for $P(A, v_1, \ldots, v_m)$ if thee exists k_x such that, for each i, we have $< v_i, A^k(x)) >> 0$ for all $k > k_x$. Equivalently, x is ANT for all $P(A, v_i)$, with $i \in \{1, \ldots, m\}$.* □

Here again we are able to reduce the termination problem for generalized homogeneous linear programs to the emptiness checks for ANT sets of homogeneous programs. More formally, writing

$$ANT^r(P(A, v_1, \ldots, v_m)) = ANT(P(A, v_1, \ldots, v_m)) \cap E^r,$$

we have

$$ANT^r(P(A, v_1, \ldots, v_m)) = \bigcap_i ANT^r(P(A, v_i).$$

We first state the following easy lemma.

Lemma 5.2. *A program $P(A, v_1, \ldots, v_m)$ is NT if and only if it is ANT, that is, if $\emptyset \neq ANT(P(A, v_1, \ldots, v_m))$.* □

Proof. If $P(A, v_1, \ldots, v_m)$ is NT, it is obviously ANT as NT points are ANT. Conversely, if it is ANT, take $x \in ANT(P(A, v_1, \ldots, v_m))$, so that $A^{k_x}(x)$ is NT. □

Now, the following proposition will be the crucial step of the main result of this section.

Proposition 5.1. *If program $P(A, v_1, \ldots, v_m)$ is ANT, then $ANT^r(P(A, v_1, \ldots, v_m)) \neq \emptyset$.* □

Proof. Indeed, suppose that $P(A, v_1, \ldots, v_m)$ is ANT. Then there is x in $\bigcap_i ANT(P(A, v_i))$. Now we write $x = x^r + x^{nr}$ in a unique way. Then, according to Theorem 4.2, we know that x^r is is ANT for every $P(A, v_i)$. Hence x^r belongs to

$$\bigcap_i ANT^r(P(A, v_i)) = ANT^r(P(A, v_1, \ldots, v_m)),$$

which is thus non empty. □

Now we can state the following main result.

Theorem 5.1. *The program $P(A, v_1, \ldots, v_l)$ is NT if and only if $\bigcap_i ANT^r(P(A, v_i)) \neq \emptyset$. In particular, this gives a deterministic procedure to check if $P(A, v_1, \ldots, v_m)$ is NT, as we can always compute $\bigcap_i ANT^r(P(A, v_i))$. Moreover, when $P(A, v_1, \ldots, v_m)$ is NT, we can compute $ANT^r(P(A, v_1, \ldots, v_m))$, and thus $E^r - ANT^r(P(A, v_1, \ldots, v_m))$ gives an under approximation of the set of terminant points of $P(A, v_1, \ldots, v_m)$.* □

Proof. We recall that $P(A, v_1, \ldots, v_m)$ is NT if and only if it is ANT due to Lemma 5.2. It remains to check that if $\bigcap_i ANT^r(P(A, v_i)) \neq \emptyset$, then $P(A, v_1, \ldots, v_m)$ is ANT. This is actually fairly obvious since we have

$$\bigcap_i ANT^r(P(A, v_i)) = ANT^r(P(A, m_1, \ldots, v_m)) \subset ANT(P(A, v_1, \ldots, v_m)),$$

and so if the set on the left is non empty, so is the set on the right. □

5.2 Generalization to affine programs

Finally, we now show that the affine case reduces to the homogeneous case as well. First we define the notion of ANT initial values for this class of programs.

Definition 5.2. *Let $P(A, V, b, c)$ be an affine program in $P^{\mathbb{A}}$. Let $x = x_0$ be a vector. Denote by x_1 the vector $Ax + c$ and, recursively, let $x_k = Ax_{k-1} + c$. We say that a vector x is ANT for $P(A, V, b, c)$ if there is a k_x, such that $k \geq k_x$ implies that $Bx_k > b$.* □

Let $A \in \mathcal{M}(n, \mathbb{R})$, $V \in \mathcal{M}(m, n, \mathbb{R})$, $b = (b_1, \ldots, b_m)^\top$ a vector in $\mathcal{M}(1, m, \mathbb{R})$ and c a vector in $\mathcal{M}(1, n, \mathbb{R})$. We denote by $P(A, V, b, c) \in P^{\mathbb{A}}$ the program which does $x := Ax + c$ as long as $Vx > b$. Now we build the matrices $A' \in \mathcal{M}(n+1, \mathbb{R})$ and $B' \in \mathcal{M}(m+1, n+1, \mathbb{R})$ as follows:

$$A' = \left(\begin{array}{ccc|c} & & & c_1 \\ & \mathrm{A} & & \vdots \\ & & & c_n \\ \hline 0 & \cdots & 0 & 1 \end{array}\right) \qquad V' = \left(\begin{array}{ccc|c} & & & -b_1 \\ & \mathrm{V} & & \vdots \\ & & & -b_m \\ \hline 0 & \cdots & 0 & 1 \end{array}\right)$$

The following theorem shows that the generation of the ANT set for a program in $P^{\mathbb{A}}$ reduces to the generation of ANT sets for specific homogeneous programs in $P^{\mathbb{H}}$.

Theorem 5.2. *Let $P(A, V, b, c)$ be an affine program in $P^{\mathbb{A}}$. Consider the matrices A' and V' as constructed just before. A vector x is ANT for $P(A, V, b, c)$ if and only if the vector $\begin{pmatrix} x \\ 1 \end{pmatrix}$ is ANT for $P(A', V')$.* □

Proof. To say that $(x, 1)^\top$ is ANT for $P(A', V')$ means that there exists k_x, such that if $k \geq k_x$, then $V'A'^k \cdot (x, 1)^\top$ is positive. But as $A' \cdot (x, 1)^\top = \begin{pmatrix} Ax + c \\ 1 \end{pmatrix} = (x_1, 1)^\top$, by induction we obtain $A'^k \cdot (x, 1)^\top = (x_k, 1)^\top$. We now have $V'A'^k \cdot (x, 1)^\top = V' \cdot (x_k, 1)^\top = \begin{pmatrix} Bx_k - b \\ 1 \end{pmatrix}$. Hence $V'A'^k \cdot (x, 1)^\top$ positive is equivalent to $Vx_k > b$, and the result follows. □

The theorems presented in this section hold for the complete class of linear programs. In the following section, we show how those ANT sets can be computed efficiently and can be symbolically represented.

6 Automatic generation of ANT loci

In this section we show how to compute exactly the ANT loci. As we saw in Section 5, the termination analysis and the generation problem for ANT sets can be reduced to the case where $Spec(\mathbf{a})$ is over the reals without loss of generality. Hence, we will assume that $Spec(\mathbf{a})$ is over the reals. In this section we address the main case, *i.e.*, the one where $\cap_{k=0}^{n-1} Ker(\mathbf{v} \circ \mathbf{a}^k)$ is the empty set. This restriction can be insured by taking the quotient of E by $K(\mathbf{a}, \mathbf{v})$. Another subspace which doesn't count for the localization of ANT points is the subspace $E_0(\mathbf{a})$. Indeed, for any x in $E_0(u)$, we

have $u^k(x) = 0$ as soon as k is greater than $E_0(u)$'s dimension. Moreover, we will also assume that $E_0(\mathbf{a}) = 0$, which can also be done by taking the quotient of E by E_0.

We first recall the following consequence of the existence of a Jordan form for $\mathbf{a}$.

Lemma 6.1. *Let λ be a nonzero eigenvalue of u. We can produce a basis B_λ of $E_\lambda(\mathbf{a})$, such that $Mat_{B_\lambda}(\mathbf{a})$ is of the form $\lambda.diag(T_{\lambda,1}, \dots, T_{\lambda,r_\lambda})$, where each $T_{\lambda,i}$ is a matrix of size $n_{\lambda,i}$, of the form*

$$\begin{pmatrix} 1 & 1 & & & & \\ & 1 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & 1 & 1 & \\ & & & & 1 & 1 \\ & & & & & 1 \end{pmatrix},$$

and $n_{\lambda,i} \leq n_{\lambda,i+1}$ for i between 1 and $r_\lambda - 1$. □

Proof. If $(e_1, \dots, e_{d_\lambda})$ is the Jordan basis of $E_\lambda(\mathbf{a})$, we take $B_\lambda = (e_1, \lambda^{-1}e_2, \dots, \lambda^{1-d_\lambda}e_{d_\lambda})$. □

In fact,since the space $\cap_{k=0}^{n-1} Ker(\mathbf{v} \circ \mathbf{a}^k)$ is assumed to be null, there is only one block.

Proposition 6.1. *For every λ in $Spec(\mathbf{a})$, the matrix of $\mathbf{a}$ restricted to $E_\lambda(\mathbf{a})$, in the basis B_λ, satisfies $Mat_{B_\lambda}(\mathbf{a}) = \lambda.T_{\lambda,1}$. So we simply write $T_\lambda = T_{\lambda,1}$.* □

Proof. If this was not the case, u would have two linearly independant eigenvectors v and w associated to λ. But as $K(\mathbf{a}, \mathbf{v})$ is zero, $\mathbf{v}(\mathbf{a}^k(v))$ is non constantly zero, and is equal to $\lambda^k \mathbf{v}(v)$, we deduce that $\mathbf{v}(v) \neq 0$. Hence we can actually normalise v so that $\mathbf{v}(v) = 1$. Similarly, we can suppose that $\mathbf{v}(w) = 1$. But then, $\mathbf{v}(\mathbf{a}^k((v - w))$ is constantly 0, *i.e.*, $v - w \in K(\mathbf{a}, \mathbf{v}) = \cap_{k=0}^{n-1} Ker(\mathbf{v} \circ \mathbf{a}^k)$, which contradicts $\cap_{k=0}^{n-1} Ker(\mathbf{v} \circ \mathbf{a}^k) = \{0\}$. □

Now we compute the power of T_λ.

Lemma 6.2. *We have*

$$(T_\lambda)^k = \begin{pmatrix} 1 & \binom{k}{1} & \binom{k}{2} & \dots & \dots & \binom{k}{d_\lambda - 1} \\ & 1 & \binom{k}{1} & \binom{k}{2} & \ddots & \vdots \\ & & \ddots & \ddots & \ddots & \vdots \\ & & & 1 & \binom{k}{1} & \binom{k}{2} \\ & & & & 1 & \binom{k}{1} \\ & & & & & 1 \end{pmatrix}.$$

□

Proof. It is a consequence of a simple induction using Pascal's triangle equality. □

We write

$$B = B_{\lambda_1} \cup \cdots \cup B_{\lambda_t}$$

as a basis for E, and we also let

$$Mat_B(f) = (a_{\lambda_1,1}, a_{\lambda_1,2}, \ldots, a_{\lambda_1,d_{\lambda_1}}, \ldots, a_{\lambda_t,1}, \ldots, a_{\lambda_t,d_{\lambda_t}}).$$

The previous lemma implies the following.

Proposition 6.2. *Let λ belong to Spec($\mathbf{a}$). There are well determined polynomials $P_{\lambda,j} \in \mathbb{R}[X]$, for j between 1 and d_λ, such that $Mat_B(\mathbf{v} \circ \mathbf{a}^k) = (P_{\lambda,1}(k), \ldots, P_{\lambda,d_\lambda}(k))$. That is,*

$$P_{\lambda,j}(k) = a_{\lambda,1}\binom{k}{j-1} + a_{\lambda,2}\binom{k}{j-2} + \cdots + a_{\lambda,j}. \quad (1)$$

In particular, $P_{\lambda,j}$, as a polynomial in k, is of degree at most $j-1$. We can thus write it as

$$P_{\lambda,j}(k) = b^j_{\lambda,j-1}k^{j-1} + b^j_{\lambda,j-2}k^{j-2} + \cdots + b^j_{\lambda,1}k + b^j_{\lambda,0}, \quad (2)$$

where we can compute each $b^j_{\lambda,l}$ explicitly as a linear combination of the $a_{\lambda,i}$'s. □

Proof. It is a consequence of Lemma 6.2, as $Mat_B(\mathbf{v} \circ \mathbf{a}^k) = Mat_B(\mathbf{v})Mat_B(\mathbf{a}^k)$. □

Remark 6.1. If $P_{\lambda,j}$ is nonzero, its dominant term is the first nonzero $a_{\lambda,m}$, for m between 1 and j. □

We are now able to give a procedure to determine the set of ANT points. Then, by taking the complement of this set, we can determine an under approximation of terminant points of such a program. We do this in a few steps. To lighten the notations, for $x = \sum_{\lambda \in Spec(\mathbf{a})} x_\lambda$, we write, $P_\lambda(x_\lambda, k) = \sum_{j=1}^{d_\lambda} x_{\lambda,j}P_{\lambda,j}(k)$. By convention, if λ is not an eigenvalue, the polynomial $P_{\lambda,j}$ is zero. We set $b^j_{\lambda,i} = 0$ as soon as $i > d_\lambda$.

We obtain the expression

$$P_\lambda(x_\lambda, k) = \sum_{j=0}^{d_\lambda - 1} \phi_{\lambda,j}(x_\lambda)k^j, \quad (3)$$

where

$$\phi_{\lambda,j}(x_\lambda) = b^{j+1}_{\lambda,j}x_{\lambda,j+1} + b^{j+2}_{\lambda,j}x_{\lambda,j+2}x_{\lambda,j+2} + \cdots + b^{d_\lambda}_{\lambda,j}x_{\lambda,d_\lambda}.$$

We set $\phi_{\lambda,j}(x_\lambda) = 0$ as soon as $j \geq d_\lambda$.

We also write

$$Q^+_{\pm\lambda}(x_{\pm\lambda}) = P_{|\lambda|}(x_{|\lambda|}) + P_{-|\lambda|}(x_{-|\lambda|}),$$

and

$$Q^{-}_{\pm\lambda}(x_{\pm\lambda}) = P_{|\lambda|}(x_{|\lambda|}) - P_{-|\lambda|}(x_{-|\lambda|}).$$

In particular, we have

$$Q^{+}_{\pm\lambda}(x_{\pm\lambda}, k) = \sum_{j=0}^{e_{|\lambda|}-1} \phi^{+}_{\pm\lambda,j}(x_{|\lambda|}, x_{-|\lambda|})k^j, \tag{4}$$

where $e_{|\lambda|} = max(d_{|\lambda|}, d_{-|\lambda|})$, and

$$\phi^{+}_{\pm\lambda,j}(x_{|\lambda|}, x_{-|\lambda|}) = \phi_{|\lambda|,j}(x_{|\lambda|}) + \phi_{-|\lambda|,j}(x_{-|\lambda|}). \tag{5}$$

Also,

$$Q^{-}_{\pm\lambda}(x_{\pm\lambda}, k) = \sum_{j=0}^{e_{|\lambda|}-1} \phi^{-}_{\pm\lambda,j}(x_{|\lambda|}, x_{-|\lambda|})k^j, \quad \text{where} \tag{6}$$

$$\phi^{-}_{\pm\lambda,j}(x_{|\lambda|}, x_{-|\lambda|}) = \phi_{|\lambda|,j}(x_{|\lambda|}) - \phi_{-|\lambda|,j}(x_{-|\lambda|}). \tag{7}$$

Example 6.1. *(Running example)*
We recall the matrix $Mat_B(\mathbf{a}) = T$, and $Mat_B(\mathbf{v}) = v$ corresponding respectively to the linear forms $\mathbf{a}$ and $\mathbf{v}$ expressed in the constructed basis B:

$$T = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 \end{pmatrix},$$

$$Mat_B(\mathbf{v}) = w = (1, 1, 1, 1, 1, 1) = (a_{1,1}, a_{1,2}, a_{-1,1}, a_{-1,2}, a_{2,1}, a_{-2,1}).$$

Using the notations introduced just above we show how to calculate all the needed terms appearing in the computation of the ANT set. Here we have four eigenvalues: 1 of multiplicity 2, -1 of multiplicity 2, 2 of multiplicity 1, and -2 of multiplicity 1. First we calculate the polynomials $P_{\lambda,j}(k)$, described by Eq. (1):

$$\begin{aligned} &P_{1,1}(k) = a_{1,1} = 1, \\ &P_{1,2}(k) = a_{1,1}\tbinom{k}{1} + a_{1,2} = k + 1, \\ &P_{-1,2}(k) = a_{-1,1} = 1, \\ &P_{-1,2}(k) = a_{-1,1}\tbinom{k}{1} + a_{-1,2} = k + 1, \\ &P_{2,1}(k) = a_{2,1} = 1, \\ &P_{-2,1}(k) = a_{-2,1} = 1. \end{aligned}$$

One can now identify the $b^j_{\lambda,l}$ using equation 2. we get:

$$b^1_{1,0} = b^2_{1,0} = b^1_{-1,0} = b^2_{-1,0} = b^1_{2,0} = b^1_{-2,0} = 1.$$

Knowing the $b^j_{\lambda,l}$ values one can nest compute the $\phi_{\lambda,j}(x_\lambda)$, in order to obtain the $P_\lambda(x_\lambda, k)$ values. We obtain the $\phi_{\lambda,j}(x_\lambda)$ terms directly :

$$\begin{aligned}
&\phi_{1,0}(x_1) = b^1_{1,0}x_{1,1} + b^2_{1,0}x_{1,2} = x_{1,1} + x_{1,2},\\
&\phi_{1,1}(x_1) = b^2_{1,1}x_{1,2} = x_{1,2},\\
&\phi_{-1,0}(x_{-1}) = x_{-1,1} + x_{-1,2},\\
&\phi_{-1,1}(x_{-1}) = x_{-1,2},\\
&\phi_{2,0}(x_2) = x_{2,1},\\
&P_1(x_1, k) = \phi_{1,0}(x_1) + \phi_{1,1}(x_1)k = (x_{1,1} + x_{1,2}) + (x_{1,2})k,\\
&\phi_{-2,0}(x_{-2}) = x_{-2,1}.
\end{aligned}$$

Using Eq. (3) we can now evaluate exactly the $P_\lambda(x_\lambda, k)$ as they are only determined by the $\phi_{\lambda,j}(x_\lambda)$ values:

$$\begin{aligned}
&P_1(x_1, k) = \phi_{1,0}(x_1) + \phi_{1,1}(x_1)k = (x_{1,1} + x_{1,2}) + (x_{1,2})k,\\
&P_{-1}(x_{-1}, k) = \phi_{-1,0}(x_{-1}) + \phi_{-1,1}(x_{-1})k = (x_{-1,1} + x_{-1,2}) + (x_{-1,2})k,\\
&P_2(x_2, k) = x_{2,1},\\
&P_{-2}(x_{-2}, k) = x_{-2,1}.
\end{aligned}$$

Knowing the $\phi_{\lambda,j}(x_\lambda)$ values one also obtains immediately the $Q^+_{|\lambda|}(x_{|\lambda|}, k)$ values using Eq. (4), the $\phi^+_{|\lambda|,j}(x_{|\lambda|}, x_{-|\lambda|})$ values using Eq. (5), the $Q^-_{|\lambda|}(x_{|\lambda|}, k)$ values using Eq. (6), and the $\phi^-_{|\lambda|,j}(x_{|\lambda|}, x_{-|\lambda|})$ values using Eq. (7). We get:

$$\begin{aligned}
&\phi^+_{1,0}(x_1, x_{-1}) = \phi_{1,0}(x_1) + \phi_{-1,0}(x_{-1}) = x_{1,1} + x_{1,2} + x_{-1,1} + x_{-1,2},\\
&\phi^+_{1,1}(x_1, x_{-1}) = \phi_{1,1}(x_1) + \phi_{-1,1}(x_{-1}) = x_{1,2} + x_{-1,2},\\
&\phi^+_{2,0}(x_2, x_{-2}) = \phi_{2,0}(x_2) + \phi_{-2,0}(x_{-2}) = x_{2,1} + x_{-2,1},\\
&\phi^-_{1,0}(x_1, x_{-1}) = x_{1,1} + x_{1,2} - x_{-1,1} - x_{-1,2},\\
&\phi^-_{1,1}(x_1, x_{-1}) = x_{1,2} - x_{-1,2},\\
&\phi^-_{2,0}(x_2, x_{-2}) = x_{2,1} - x_{-2,1}.
\end{aligned}$$

We now have all the required terms to generated the linear constraint representing symbolically and exactly the ANT set. □

Proposition 6.3. *Let λ be a positive eigenvalue, such that $Q^+_{\pm\lambda}(x_{|\lambda|})$ and $Q^-_{\pm\lambda}(x_{|\lambda|})$ both have a positive dominant term, and such that $P_\mu(x_\mu)$ is zero whenever $|\mu| > \lambda$. Then x is an ANT point of $P(u, f)$.* □

Proof. As P_μ is zero as soon as $|\mu| > \lambda$, asymptotically, $f(u^k(x))$ is equivalent to $\lambda^k(P_{|\lambda|}(x_{|\lambda|}, k) + (-1)^k P_{-|\lambda|}(x_{-|\lambda|}, k))$. So $f(u^{2k}(x))$ is equivalent to $\lambda^{2k} Q^+_{\pm\lambda}(x_{\pm\lambda}, 2k)$, and $f(u^{2k+1}(x))$ is equivalent to $\lambda^{2k+1} Q^-_{\pm\lambda}(x_{\pm\lambda}, 2k+1)$, and the result follows. □

Writing this more explicitly, in terms of the $\phi_{|\lambda|,j}$, and $\phi^\pm_{|\lambda|,j}$ linear forms, we obtain the following.

Proposition 6.4. *For $\lambda > 0$ in Spec(u), and two integers k and k' between 0 and $d_\lambda - 1$, denote by $S^\lambda_{k,k'}$ the set of x in E which satisfy:*

1. *For all μ with $|\mu| > \lambda$, and all j between 0 and $d_\mu - 1$: $\phi_{\mu,j}(x_\mu) = 0$.*
2. *For all $e_\lambda - 1 \geq j > k$: $\phi^+_{\pm\lambda,j}(x_\lambda, x_{-\lambda}) = 0$.*
3. *For all $e_\lambda - 1 \geq j > k'$: $\phi^-_{\pm\lambda,j}(x_\lambda, x_{-\lambda}) = 0$.*
4. *$\phi^+_{\pm\lambda,k}(x_\lambda, x_{-\lambda}) > 0$.*
5. *$\phi^-_{\pm\lambda,k'}(x_\lambda, x_{-\lambda}) > 0$.*

If x belongs to

$$S = \coprod_{\substack{\lambda>0\in Spec(u)\\ k\in\{1,\ldots,d_\lambda-1\}\\ k'\in\{1,\ldots,d_\lambda-1\}}} S^\lambda_{k,k'}, \tag{8}$$

then x is ANT. □

Applying directly Proposition 6.4, we need only consider only the positive eigenvalues. From the associated multiplicities of these positive eigenvalues, we can determine the possible values for k and k'. Then, for each item, we consider the corresponding equality or inequality, expressed in terms of $\phi_{|\lambda|,j}$ and of $\phi^\pm_{|\lambda|,j}$ as constraint describing a subset of the ANT set for the considered program. For instance, if we consider item (1) in Proposition 6.4, we first check if there are eigenvalues μ with $|\mu| > \lambda$, and j between 0 and $d_\mu - 1$. Then we consider the equality $\phi_{\mu,j}(x_\mu) = 0$ replacing the terms $\phi_{\mu,j}(x_\mu)$ by their precomputed values, depending on the initial input parameters of the program. This constraint describes a set of ANT initial inputs for the program.

Returning to the running example we now illustrate how the constraints associated to Proposition 6.4 are automatically and exactly generated.

Example 6.2. *(Running example)* We apply Proposition 6.4 directly. Considering only the positive eigenvalues 2 and 1, from the multiplicity of these eigenvalues, we know that we have to consider $k \in \{0, 1\}$ and $k' \in \{0, 1\}$. There are several cases.

- Case 1: $\lambda = 2; k = 0; k' = 0$:

- Item (1) in Proposition 6.4 gives no constraint because there is no μ such that $|\mu| > 2$.
- Also items (2) and (3) give no constraint because there is no j such that $e_2 - 1 \geq j > 0$, because $e_2 - 1 = 0$.
- With item (4) we obtain the constraint $\phi^+_{2,0}(x_2, x_{-2}) > 0$. It can be rewritten as $x_{2,1} + x_{-2,1} > 0$. See Example 6.1 where we found that $\phi^+_{2,0}(x_2, x_{-2}) = x_{2,1} + x_{-2,1}$.
- From item (5) we obtain the constraint $(\phi^+_{2,0}(x_2, x_{-2}) > 0)$. It can be rewritten as $(x_{2,1} - x_{-2,1} > 0)$. Consult the value computed for $(phi^-_{2,0}(x_2, x_{-2})$ in Example 6.1.

Using the same notation, we have

$$S^2_{0,0} \equiv (x_{2,1} + x_{-2,1} > 0) \wedge (x_{2,1} - x_{-2,1} > 0).$$

Since there is no $k > 0$, $k' > 0$ between 0 and $d_2 - 1$ because $d_2 = 1$, we find no other cases to consider when $\lambda = 2$.

- Case 2: $\lambda = 1; k = 0; k' = 0$:
  - Looking at condition (1) in Proposition. 6.4, the only possible values for μ is 2 and -2. Also we have $j \in \{0\}$. Thus, for the case $\mu = 2$ we obtain the constraint $\phi_{2,0} = 0$, which can be evaluated as $x_{2,1} = 0$. See the first part of Example 6.1 where the value of ϕ is given. For $\mu = -2$ we generate the constraint $\phi_{-2,0} = 0$ which is equivalent to $x_{-2,1} = 0$.
  - From item (2) in Proposition 6.4, we only have $j = 1$ and the constraint obtained is $\phi^+_{1,1}(x_1, x_{-1}) = 0$, which can also be written as $x_{1,2} + x_{-1,2} = 0$.
  - Again, for item (3) in Proposition 6.4, we only have $j = 1$ and the constraint obtained is $\phi^-_{1,1}(x_1, x_{-1}) = 0$, which can also be written as $x_{1,2} - x_{-1,2} = 0$.
  - Item (4) in Proposition 6.4 gives the constraint $\phi^+_{1,0}(x_1, x_{-1}) > 0$ which is evaluated as $x_{1,1} + x_{1,2} + x_{-1,1} + x_{-1,2} > 0$.
  - Item (5) Proposition 6.4, gives the constraint $x_{1,1} + x_{1,2} - x_{-1,1} - x_{-1,2} > 0$.

  In this case we have

$$\begin{aligned} S^1_{0,0} \equiv\ & (x_{2,1} = 0) \wedge (x_{-2,1} = 0) \wedge \\ & (x_{1,2} + x_{-1,2} = 0) \wedge (x_{1,2} - x_{-1,2} = 0) \wedge \\ & (x_{1,1} + x_{1,2} + x_{-1,1} + x_{-1,2} > 0) \wedge \\ & (x_{1,1} + x_{1,2} - x_{-1,1} - x_{-1,2} > 0). \end{aligned}$$

- Case 3: $\lambda = 1; k = 0; k' = 1$:
  - Here again, with items (1) and (2) in Proposition 6.4 we obtain the constraints $x_{2,1} = 0$); ($x_{-2,1} = 0$ and $x_{1,2} + x_{-1,2} = 0$.
  - Item (3) in Proposition 6.4, gives no constraint because there is no j such that $1 \geq j > 1$.
  - Item (4) gives $\phi^{+}_{1,0}(x_1, x_{-1}) > 0$ which is equivalent to $x_{1,1} + x_{1,2} + x_{-1,1} + x_{-1,2} > 0$.
  - Item (5) gives $\phi^{-}_{1,1}(x_1, x_{-1}) > 0$ which is equivalent to $x_{1,2} - x_{-1,2} > 0$.

  In this case we obtain

$$S^1_{0,1} \equiv (x_{2,1} = 0) \wedge (x_{-2,1} = 0) \wedge (x_{1,2} + x_{-1,2} = 0) \wedge (x_{1,1} + x_{1,2} + x_{-1,1} + x_{-1,2} > 0) \\ \wedge (x_{1,2} - x_{-1,2} > 0).$$

- Case 4: $\lambda = 1; k = 1; k' = 0$:
  - Item (1) provides the constraints $x_{2,1} = 0$ and $x_{-2,1} = 0$ and item (2) gives no constraint.
  - Item (3) fixes $j = 1$ and we obtain the constraint $\phi^{-}_{1,1}(x_1, x_{-1}) = 0$ which is equivalent to $x_{1,2} - x_{-1,2} = 0$.
  - Item (4) provides the equation $\phi^{+}_{1,1}(x_1, x_{-1}) > 0$ which is equivalent to $x_{1,2} + x_{-1,2} > 0$.
  - Item (5) says that $\phi^{-}_{1,0}(x_1, x_{-1}) > 0$ which is equivalent to $x_{1,1} + x_{1,2} - x_{-1,1} - x_{-1,2} > 0$.

  For this case, we obtain:

$$S^1_{1,0} \equiv (x_{2,1} = 0) \wedge (x_{-2,1} = 0) \wedge (x_{1,2} + x_{-1,2} > 0) \wedge (x_{1,1} + x_{1,2} - x_{-1,1} - x_{-1,2} > 0).$$

- Case 5: $\lambda = 1; k = 1; k' = 1$:
  - Item (1) provides again the constraints $x_{2,1} = 0$ and $x_{-2,1} = 0$. Item (2) and (3) provide no constraints.
  - Item (4) generates the constraint $\phi^{+}_{1,1}(x_1, x_{-1}) > 0$ which is the same as $x_{1,2} + x_{-1,2} > 0$.
  - Item (5) gives the constraint $x_{1,2} - x_{-1,2} > 0$.

With this case, we have

$$\begin{aligned} S^1_{1,1} \equiv\ & (x_{2,1} = 0) \wedge (x_{-2,1} = 0) \wedge \\ & (x_{1,2} + x_{-1,2} > 0) \wedge \\ & (x_{1,2} - x_{-1,2} > 0). \end{aligned}$$

Finally, we can generate automatically all the ANT conditions provided by Proposition 6.4 using Eq. (8). Thus, we obtain:

$$S = S^2_{0,0} \cup S^1_{0,0} \cup S^1_{0,1} \cup S^1_{1,0} \cup S^1_{1,1}. \tag{9}$$

All the involved set $S^{\lambda}_{k,k'}$ have already been computed.

□

The following propositions provide the other sets of constraints describing disjoint sets of ANT points.

Proposition 6.5. *Let λ be a positive eigenvalue and $|\lambda'| < \lambda$ in $|Spec(u)|$, and such that:*

1. *$P_\mu(x_\mu)$ is zero whenever $|\mu| > \lambda$,*
2. *$Q^-_{\pm\lambda}(x_{\pm\lambda}) = 0$ and $Q^+_{\pm\lambda}(x_{\pm\lambda}) = 2P_\lambda(x_\lambda)$ has a positive dominant term,*
3. *$Q^-_{\pm\mu}(x_{\pm\mu})$ is zero whenever $|\lambda'| < |\mu| < \lambda$,*
4. *$Q^-_{\pm\lambda'}(x_{\pm\lambda'})$ has a positive dominant term*

Then $P(u, f)$ is ANT on x. □

Proof. Because of our first condition, the quantity $f(u^{2k}(x))$ is asymptotically equivalent to

$$\lambda^{2k} Q^+_{\pm\lambda}(x_{\pm\lambda}, 2k)$$

for k large. From our second condition, the quantity $f(u^{2k+1}(x)$ is asymptotically equivalent to

$$|\lambda'|^{2k+1} Q^-_{\pm\lambda'}(x_{\pm\lambda'}, 2k+1).$$

In both cases, as $Q^+_{\pm\lambda}(x_{\pm\lambda})$ and $Q^-_{\pm\lambda'}(x_{\pm\lambda'})$ both have positive dominant terms, we conclude that $f(u^{2k}(x))$ and $f(u^{2k+1}(x))$ are both positive when k is large. □

We now write this in terms of the linear forms $\phi_{\lambda,j}$ and $\phi^{\pm}_{\pm\lambda,j}$.

Proposition 6.6. *For $\lambda > 0$ and λ' in Spec(u), with $|\lambda'| < \lambda$, an integer k between 1 and $d_\lambda - 1$, and an integer k' between 1 and $d_{\lambda'} - 1$, we denote by $U_{k,k'}^{\lambda,|\lambda'|}$ the set of $x \in E$ which satisfy:*

1. *For all μ with $|\mu| > \lambda$, and all j between 0 and $d_\mu - 1$: $\phi_{\mu,j}(x_\mu) = 0$.*
2. *For all $0 \leq j \leq e_\lambda - 1$: $\phi^-_{\pm\lambda,j}(x_{|\lambda|}, x_{-|\lambda|}) = 0$.*
3. *For all $|\mu|$ with $\lambda' < |\mu| < \lambda$, and all $0 \leq j \leq e_{|\mu|} - 1$: $\phi^-_{\pm\mu,j}(x_{|\mu|}, x_{-|\mu|}) = 0$.*
4. *For all $d_\lambda - 1 \geq j > k$: $\phi_{\lambda,j}(x_\lambda) = 0$.*
5. *For all $e_{|\lambda'|} - 1 \geq j > k'$: $\phi^-_{\pm\lambda',j}(x_{|\lambda'|}, x_{-|\lambda'|}) = 0$.*
6. *$\phi_{\lambda,k}(x_\lambda) > 0$.*
7. *$\phi^-_{\pm\lambda',k'}(x_{|\lambda'|}, x_{-|\lambda'|}) > 0$.*

If x belongs to

$$U = \coprod_{\substack{\lambda > 0 \in Spec(u) \\ |\lambda'| < \lambda \in |Spec(u)| \\ k \in \{1,\ldots,d_\lambda - 1\} \\ k' \in \{1,\ldots,e_{|\lambda'|} - 1\}}} U_{k,k'}^{\lambda,|\lambda'|}, \tag{10}$$

then x is ANT. □

Example 6.3. *(Running example)*:
Here again, we illustrate how the constraint are automatically generated by considering the running example exposed in Example 6.1 and Example 6.2. By definition, there are only two possible cases.

- Case 1: $\lambda = 2$, $|\lambda'| = 1$, $k = 0$, $k' = 0$:
  - items (1), (3) and (4) return an empty set of constraints.
  - With item (2) we have $j = 0$ and it returns the constraint $\phi^-_{1,0}(x_2, x_{-2}) = 0$ which is equivalent to $x_{2,1} - x_{-2,1} = 0$.
  - Item (5) gives $1 \geq j > 0$ and so $j = 1$. We get $\phi^-_{1,1}(x_2, x_{-2}) = 0$ which is the constraint $x_{1,2} - x_{-1,2} = 0$.
  - Item (6) gives $\phi_{2,0}(x_2) > 0$ which is equivalent to $x_{2,1} > 0$.
  - Item (7) gives $\phi^-_{1,0}(x_2, x_{-2}) > 0$ which is equivalent to $x_{1,1} + x_{1,2} - x_{-1,1} - x_{-1,2} > 0$.

  In this case we obtain

  $U_{0,0}^{2,1} \equiv (x_{2,1} - x_{-2,1} = 0) \wedge (x_{1,2} - x_{-1,2} = 0) \wedge (x_{2,1} > 0) \wedge (x_{1,1} + x_{1,2} - x_{-1,1} - x_{-1,2} > 0)$.

- Case 2: $\lambda = 2$, $|\lambda'| = 1$, $k = 0$, $k' = 1$:
  - items (1), (3), (4) and (5) return an empty set of constraints.
  - item (2) returns the constraint $\phi^-_{2,0}(x_2, x_{-2}) = 0$ which is $x_{2,1} - x_{-2,1} = 0$.
  - Item (6) returns the constraint $\phi_{2,0}(x_2) > 0$ which is equivalent to $x_{2,1} > 0$.
  - Item (7) returns the constraint $\phi^-_{1,1}(x_2, x_{-2}) > 0$ which is equivalent to $x_{1,2} - x_{-1,2} > 0$.

  In this case we obtain

$$U^{2,1}_{0,1} \equiv (x_{2,1} - x_{-2,1} = 0) \wedge (x_{2,1} > 0) \wedge (x_{1,2} - x_{-1,2} > 0).$$

Using Eq. (10) we can automatically generate all the ANT conditions. We obtain:

$$U = U^{2,1}_{0,0} \cup U^{2,1}_{0,1}. \tag{11}$$

All the involved sets $U^{\lambda,|\lambda'|}_{k,k'}l$ have already been computed.

□

The following two propositions allow the automatic generation of a third set of constraints describing the final remaining disjoint set of ANT points.

Proposition 6.7. *Let λ be a positive eigenvalue, let $\lambda' \in Spec(u)$ with $|\lambda'| < \lambda$, satisfy:*

1. *$P_\mu(x_\mu)$ is zero whenever $|\mu| > \lambda$,*
2. *$Q^+_{\pm\lambda}(x_{\pm\lambda}) = 0$ and $Q^-_{\pm\lambda}(x_{\pm\lambda}) = 2P_\lambda(x_\lambda)$ has a positive dominant term,*
3. *$Q^+_{\pm\mu}(x_{\pm\mu})$ is zero whenever $|\lambda'| < |\mu| < \lambda$,*
4. *$Q^+_{\pm\lambda'}(x_{\pm\lambda'})$ has a positive dominant term.*

Then $P(u, f)$ is ANT on x. □

Proof. Because of our first condition, the quantity $f(u^{2k+1}(x))$ is asymptotically equivalent to $\lambda^{2k+1}Q^-_{\pm\lambda}(x_{\pm\lambda}, 2k+1)$ for k large. From our second condition, $f(u^{2k}(x))$ is asymptotically equivalent to $|\lambda'|^{2k}Q^+_{\pm\lambda'}(x_{\pm\lambda'}, 2k)$. In both case, as $Q^-_{\pm\lambda}(x_{\pm\lambda})$ and $Q^+_{\pm\lambda'}(x_{\pm\lambda'})$ both have positive dominant terms, we conclude that $f(u^{2k}(x))$ and $f(u^{2k+1}(x))$ are both positive when k is large. □

We now write the preceding proposition in terms of the linear forms $\phi_{\lambda,j}$ and $\phi^\pm_{\pm\lambda,j}$.

Proposition 6.8. *For $\lambda > 0$ and λ' in $Spec(u)$, with $|\lambda'| < \lambda$, an integer k between 1 and $d_\lambda - 1$ and an integer k' between 1 and $d_{\lambda'} - 1$, we denote by $V^{\lambda,|\lambda'|}_{k,k'}$ the set of $x \in E$ which satisfy:*

1. *for all* μ *with* $|\mu| > \lambda$, *and all* j *between* 0 *and* $d_\mu - 1$: $\phi_{\mu,j}(x_\mu) = 0$.
2. *for all* $0 \leq j \leq e_{|\lambda|} - 1$: $\phi^+_{\pm\lambda,j}(x_\lambda, x_{-\lambda}) = 0$.
3. *for all* $|\mu|$ *with* $\lambda' < |\mu| < \lambda$, *and all* $0 \leq j \leq e_{|\mu|} - 1$: $\phi^+_{\pm\mu,j}(x_{|\mu|}, x_{-|\mu|}) = 0$.
4. *for all* $d_\lambda - 1 \geq j > k$: $\phi_{\lambda,j}(x_\lambda) = 0$.
5. *for all* $e_{|\lambda'|} - 1 \geq j > k'$: $\phi^+_{\pm\lambda',j}(x_{|\lambda'|}, x_{-|\lambda'|}) = 0$.
6. $\phi_{\lambda,k}(x_\lambda) > 0$.
7. $\phi^+_{|\lambda'|,k'}(x_{|\lambda'|}, x_{-|\lambda'|}) > 0$.

If x *belongs to*

$$V = \coprod_{\substack{\lambda > 0 \in Spec(u) \\ |\lambda'| < \lambda \in |Spec(u)| \\ k \in \{1,\ldots,d_\lambda - 1\} \\ k' \in \{1,\ldots,e_{|\lambda|} - 1\}}} V^{\lambda,|\lambda'|}_{k,k'}, \tag{12}$$

then x *is ANT.* □

Example 6.4. *(Running example)*: We consider again our running example in order to illustrate how the last set of constraints is generated. Here, there are only two possible cases.

- Case 1: $\lambda = 2$, $|\lambda'| = 1$, $k = 0$, $k' = 0$:
  - Items (1), (3) and (4) return an empty set of constraints.
  - Item (2) fixes $j = 0$ and returns the constraint $\phi^+_{2,0}(x_2, x_{-2}) = 0$ which is $x_{2,1} + x_{-2,1} = 0$.
  - Item (5) fixes $j = 1$ and returns the constraint $\phi^+_{1,1}(x_2, x_{-2}) = 0$ which is $x_{1,2} + x_{-1,2} = 0$.
  - With item (6) we have $\phi_{2,0}(x_2) > 0$ which is $x_{2,1} > 0$.
  - Item (7) returns $\phi^+_{1,0}(x_1, x_{-1} > 0$ which is $x_{1,1} + x_{-1,2} + x_{-1,1} + x_{-1,2} > 0$.

  In this case we obtain

  $V^{2,1}_{0,0} \equiv (x_{2,1}+x_{-2,1} = 0) \wedge (x_{1,2}+x_{-1,2} = 0) \wedge (x_{2,1} > 0) \wedge (x_{1,1}+x_{-1,2}+x_{-1,1}+x_{-1,2} > 0)$.

- Case 2: $\lambda = 2$, $|\lambda'| = 1$, $k = 0$, $k' = 1$:
  - Items (1), (3), (4) and (5) return an empty set of constraints.

– Item (2) gives $\phi^+_{2,0}(x_2, x_{-2}) = 0$ which is $x_{2,1} + x_{-2,1} = 0$.
– Again with item (6) we have $\phi_{2,0}(x_2) > 0$ which is $x_{2,1} > 0$.
– Item (7) returns $\phi^+_{1,1}(x_1, x_{-1} > 0$ which is $x_{1,2} + x_{-1,2} > 0$.

In this case we obtain

$$V^{2,1}_{0,1} \equiv (x_{2,1} + x_{-2,1} = 0) \wedge (x_{2,1} > 0) \wedge (x_{1,2} + x_{-1,2} > 0).$$

We can now automatically generate all the ANT conditions as given by Eq. (12). We obtain:

$$V = V^{2,1}_{0,0} \cup V^{2,1}_{0,1}. \tag{13}$$

All the involved set $V^{\lambda,|\lambda'|}_{k,k'}$ have already been computed. □

We are now able to state our main theorem.

Theorem 6.1. *An element $x \in E$ is ANT for $P(\mathbf{a}, \mathbf{v})$ if and only if we are in the situation of Proposition 6.3, 6.5, or 6.7.* □

Proof. Let x be an ANT point. If the eigenvalue λ of largest absolute value such that $P_\lambda(x_\lambda)$ is nonzero is negative, and $P_{-\lambda}(x_{-\lambda})$ equals zero, then $\mathbf{v}(\mathbf{a}^k(x)$ would be asymptotically equivalent to $\lambda^k P_\lambda(x_\lambda, k)$. As $P_\lambda(x_\lambda, k)$ is asymptotically of the sign of its dominant term, and λ^k is alternatively positive and negative, the program $P(\mathbf{a}, \mathbf{v})$ would terminate on x. Hence, if λ is of largest absolute value such that $P_\lambda(x_\lambda)$ is nonzero, then $P_{|\lambda|}(x_{|\lambda|})$ is nonzero, and we can actually suppose that λ is positive.
If $Q^+_{\pm\lambda}(x_{\pm\lambda})$ and $Q^-_{\pm\lambda}(x_{\pm\lambda})$ are nonzero, as

$$\mathbf{v}(\mathbf{a}^{2k}(x)) \sim \lambda^{2k} Q^+_{\pm\lambda}(x_{\pm\lambda}, 2k)$$

and

$$\mathbf{v}(\mathbf{a}^{2k+1}(x)) \sim \lambda^{2k+1} Q^-_{\pm\lambda}(x_{\pm\lambda}, 2k+1),$$

they will both be positive for k large if and only if $Q^+_{\pm\lambda}(x_{\pm\lambda})$ and $Q^-_{\pm\lambda}(x_{\pm\lambda})$ have a positive dominant term. In this case, we are in the sitation of Proposition 6.3.
If $Q^-_{\pm\lambda}(x_{\pm\lambda})$ is equal to zero, then $Q^+_{\pm\lambda}(x_{\pm\lambda})$ is not, otherwise $P_\lambda(x_\lambda)$ and $P_{-\lambda}(x_{-\lambda})$ would both be zero. So

$$\mathbf{v}(\mathbf{a}^{2k}(x)) \sim \lambda^{2k} Q^+_{\pm\lambda}(x_{\pm\lambda}, 2k),$$

and $Q^+_{\pm\lambda}(x_{\pm\lambda})$ must have a positive dominant term. If $Q^-_{\pm\mu}(x_{\pm\mu})$ was zero for all eigenvalues μ, we would have $\mathbf{v}(\mathbf{a}^{2k+1}(x)) = 0$ for all k, which is absurd because x is ANT. Hence there is λ' of absolute value as large as possible, with $|\lambda'| < \lambda$, such that $Q^-_{\pm\lambda'}(x_{\pm\lambda'})$ is nonzero. In this case, $\mathbf{v}(\mathbf{a}^{2k+1}(x))$ is equivalent to $|\lambda'|^{2k+1} Q^-_{\pm\lambda'}(x_{\pm\lambda'})$,

and as x is ANT. This forces $Q^-_{\pm\lambda'}(x_{\pm\lambda'})$ to have a positive dominant term, and we are in the situation of Proposition 6.5.
Finally, in the last possible case, $Q^+_{\pm\lambda}(x_{\pm\lambda})$ is equal to zero, and $Q^-_{\pm\lambda}(x_{\pm\lambda})$ is necessarily nonzero. As $\mathbf{v}(\mathbf{a}^{2k+1}(x)) \sim \lambda^{2k+1} Q^-_{\pm\lambda}(x_{\pm\lambda}, 2k+1)$, this implies that $Q^-_{\pm\lambda}(x_{\pm\lambda}, 2k+1)$ has a positive dominant term. Again, if $Q^+_{\pm\mu}(x_{\pm\mu})$ was equal to zero for all $k \geq 0$, we would have $\mathbf{v}(\mathbf{a}^{2k}(x)) = 0$, which is absurd as x is ANT. Hence there is a λ' of absolute value as large as possible, with $|\lambda'| < \lambda$, such that $Q^+_{\pm\lambda'}(x_{\pm\lambda'})$ is nonzero. In this case, $\mathbf{v}(\mathbf{a}^{2k}(x))$ is equivalent to $|\lambda'|^{2k} Q^+_{\pm\lambda'}(x_{\pm\lambda'})$, and as x is ANT. This forces $Q^+_{\pm\lambda'}(x_{\pm\lambda'})$ to have a positive dominant term, and we are in the situation of Proposition 6.7. □

We can now restate the previous theorem as the following result.

Theorem 6.2. *The set $ANT(P(\mathbf{a}, \mathbf{v}))$ of ANT points of $P(\mathbf{a}, \mathbf{v})$, is equal to the disjoint union $S \cup U \cup V$, where S, U and V are defined in Propositions 6.4, 6.6, and 6.8.* □

Let us continue our development by illustrating the conclusion of Theorem 6.1 considering our running example.

Example 6.5. *(Running example)*:
According to Theorem 6.2 the ANT set is described by $S \cup U \cup V$ where the sets S, U and V have already been explicitly computed. See Eqs. (9), (11) and (13). Our prototype generates the equivalent semi-linear system:

```
Locus of ANT
[
[[max(-X[-2,1], X[-2,1]) < X[2,1]]]
OR
[[X[1,2] == 0, X[-1,2] == 0, X[2,1] == 0,
X[-2,1] == 0, max(-X[-1,1], X[-1,1]) < X[1,1]]]
OR
[[X[1,2] == -X[-1,2], X[2,1] == 0, X[-2,1] == 0,
-X[-1,1] < X[1,1], X[-1,2] < 0]]
OR
[[X[2,1] == 0, X[-2,1] == 0,
X[-1,1] - X[1,2] + X[-1,2] < X[1,1],
-X[-1,2] < X[1,2]]]
OR
[[X[2,1] == 0, X[-2,1] == 0,
max(-X[-1,2], X[-1,2]) < X[1,2]]]
]
OR
[
```

```
[[X[1,2] == X[-1,2], X[2,1] == X[-2,1],
X[-1,1] < X[1,1], 0 < X[-2,1]]]
OR
[[X[2,1] == X[-2,1], X[-1,2] < X[1,2],
 0 < X[-2,1]]]
]
OR
[
[[X[1,2] == X[-1,2], X[2,1] == -X[-2,1],
-X[-1,1] - 2*X[-1,2] < X[1,1], X[-2,1] < 0]]
OR
[[X[2,1] == -X[-2,1], -X[-1,2] < X[1,2],
X[-2,1] < 0]]
]
```

□

7 The general case

In this section we do not assume that the spaces $\cap_{k=0}^{n-1} Ker(\mathbf{v} \circ \mathbf{a}^k)$ and $E_0(\mathbf{a})$ are reduced to zero, but only that $Spec(\mathbf{a})$ is real. We start with a few definitions.

Definition 7.1. *For $x \in E$, we denote by $E(\mathbf{a}, x)$ the subspace of E generated by the family $(\mathbf{a}^k(x))_{k\geq 0}$. It is a $\mathbf{a}$-stable subspace.* □

Definition 7.2. *We denote by $K(\mathbf{a}, \mathbf{v})$ the subspace $\cap_{k\geq 0} Ker(\mathbf{v} \circ \mathbf{a}^k) = \cap_{k=0}^{n-1} Ker(\mathbf{v} \circ \mathbf{a}^k)$ of E. It is a $\mathbf{a}$-stable subspace.* □

First, we recall the asymptotic behavior of $\mathbf{v}(\mathbf{a}^k(x))$ for k large, $x \in E_\lambda(\mathbf{a})$, and $\lambda \in Spec_{\mathbb{R}}(\mathbf{a}) - \{0\}$.

Proposition 7.1. *For $\lambda \in Spec(\mathbf{a}) - \{0\}$, and $x \in E_\lambda(\mathbf{a})$, there exits $P \in \mathbb{R}[X]$ of degree $m \leq d_\lambda - 1$, such that $\mathbf{v}(\mathbf{a}^k(x)) = \lambda^k P(k)$. In particular, $\mathbf{v}(\mathbf{a}^k(x))$ is equivalent to $a\lambda^k k^m$ when k goes to infinity, and where a is the leading coefficient of P, and m is the degree of P.* □

Proof. The restriction of $\mathbf{a}$ to $E_\lambda(\mathbf{a})$ admits a matrix of the form $T = \begin{pmatrix} \lambda & \dots & . \\ & \ddots & \vdots \\ & & \lambda \end{pmatrix}$ in a Jordan basis of $E_\lambda(\mathbf{a})$. It is easy to check, by induction on d_λ and using the theory of Bernoulli polynomials, that T^k is upper triangular, with diagonal entries equal to λ^k, and non diagonal nonzero entries of the form $\lambda^k Q(k)$, for $Q \in \mathbb{R}[X]$ of degree $\leq d_\lambda$. The result then follows. □

In the situation above, we will write $P_\lambda(v,x,k)$ for $P_\lambda(v,x)(k)$. So P_λ is a map from $\mathbb{R}^n \times E_\lambda(\mathbf{a}) \times \mathbb{N}$ to $\mathbb{R}$, which is linear in the first two variables, and polynomial in the last. In terms of vector spaces and morphisms, this translates thus:

Proposition 7.2. *Let E be a real vector space, with $\mathbf{a}$ in $End_{\mathbb{R}}(E)$. For λ a real nonzero eigenvalue of $\mathbf{a}$, there exits P_λ from $E^* \times E_\lambda(\mathbf{a}) \times \mathbb{N}$ to $\mathbb{R}$, which is linear in the first two variables and polynomial in the last, of degree $m \leq d_\lambda - 1 = dim(E_\lambda(\mathbf{a})) - 1$, and such that $\mathbf{v}(\mathbf{a}^k(x)) = \lambda^k P_\lambda(\mathbf{v},x,k)$ for any f, x and k.* □

We have the following obvious property:

Proposition 7.3. *If $\lambda \neq 0$ is a real eigenvalue of $\mathbf{a}$, $\mathbf{v}$ belongs to E^*, and x belongs to $E_\lambda(\mathbf{a})$, then $P_\lambda(\mathbf{v},x,.)$ is nonzero if and only if $x \notin K(\mathbf{v},\mathbf{a})$.* □

When studying the loci of ANT points in E, or more generally for any question related to the termination of the program $P(\mathbf{a},\mathbf{v})$, this subspace does not interfere, given the following proposition.

Proposition 7.4. *For any $k \geq 0$, the linear form $\mathbf{v} \circ \mathbf{a}^k$ factors through the quotient $E/K(\mathbf{a},\mathbf{v})$,* i.e.*, for any $x \in E$, the value of $\mathbf{v}(\mathbf{a}^k(x))$ only depends on the set $x + K(\mathbf{a},\mathbf{v})$.* □

We denote by $\overline{\mathbf{a}}$ the endomorphism $\overline{E} = E/K(\mathbf{a},\mathbf{v})$ induced by $\mathbf{a}$, and by $\overline{\mathbf{v}}$ the linear form on $E/K(\mathbf{a},\mathbf{v})$ induced by $\mathbf{v}$. Then, we write $\overline{E} = \overline{E}_0(\overline{\mathbf{a}}) \oplus \overline{E}^a$, where $\overline{E}^a = \oplus_{\lambda \in Spec(\overline{\mathbf{a}}) - \{0\}} \overline{E}_\lambda(\overline{\mathbf{a}})$, and then consider the restriction $\overline{\mathbf{a}}^a$ of $\overline{\mathbf{a}}$ to $\overline{E}^a$, as well as the restriction $\overline{\mathbf{v}}^a$ of $\overline{\mathbf{v}}$ to $\overline{E}^a$. The program $P(\overline{\mathbf{a}}^a, \overline{\mathbf{v}}^a)$ is of the form studied in the previous section. In other words, we have $\overline{E}^a_0(\overline{\mathbf{a}}^a) = K(\overline{\mathbf{a}}^a, \overline{\mathbf{v}}^a) = \{0\}$. Hence we know how to compute $ANT(P(\overline{\mathbf{a}}^a, \overline{\mathbf{v}}^a))$.

By the discussion at the end of section 5, we have the following main theorem.

Theorem 7.1. *The program $P(\mathbf{a},\mathbf{v})$ terminates if and only if the program $P(\overline{\mathbf{a}}^a, \overline{\mathbf{v}}^a)$ terminates. Moreover, if we write $p : E \to \overline{E}$ for the canonical projection, we have the relation*

$$ANT(P(\mathbf{a},\mathbf{v})) = p^{-1}(ANT(P(\overline{\mathbf{a}}^a, \overline{\mathbf{v}}^a)) + \overline{E}_0(\overline{\mathbf{a}})).$$

□

Under this form, it might not be obvious to the reader not familiar with linear algebra how to apply this in a concrete situation, where we are given a program $P(A,v)$, corresponding to a matrix $A \in M(n,\mathbb{R})$, and $v = (v_1, \ldots, v_n)$ a row vector in $\mathbb{R}^n$. We now discuss this situation.

First, one computes a basis $B_{A,v}$ of the space $K(A,v) = \cap_{k \geq 0} Ker(vA^k) = \cap_{k=0}^{n-1} Ker(vA^k)$, which is nothing else than the kernel of the matrix given by the

rows

$$\begin{pmatrix} (v_1,\dots,v_n) \\ (v_1,\dots,v_n)A \\ \vdots \\ (v_1,\dots,v_n)A^{n-2} \\ (v_1,\dots,v_n)A^{n-1} \end{pmatrix}$$

of $M(n,\mathbb{R})$.

Then we take any family B_1, such that $B' = B_{A,v} \cup B_1$ is a basis of $\mathbb{R}^n$, and let P be the transformation matrix, the columns of which are the vectors of B'. We have $P^{-1}AP = \begin{pmatrix} X & Y \\ 0 & A_1 \end{pmatrix}$, for A_1 of size of B_1. Now consider the matrix $A_1 \in M(n_1,\mathbb{R})$, and take B_J as the modified Jordan basis, where the first vectors of B_J are a Jordan basis B_0 of $E_0(A_1)$, and the next vectors in B_J are ordered as a union of basis B_λ for each $E_\lambda(A_1)$, with $\lambda \neq 0 \in Spec(A_1)$, where B_λ is the modified Jordan basis defined in Lemma 6.1. Then, if P_1 is the transformation matrix in $M(n_1,\mathbb{R})$ the columns of which are the vectors of B_J, then $T = P_1^{-1}A_1P_1$ is of the form

$$T = \begin{pmatrix} T_0 & & & & \\ & \lambda_1 T_{\lambda_2} & & & \\ & & \ddots & & \\ & & & \lambda_{t-1}T_{\lambda_{t-1}} & \\ & & & & \lambda_t T_{\lambda_t} \end{pmatrix},$$

where each $\lambda_1,\dots,\lambda_t$ are the nonzero eigenvalues of A_1, T_{λ_i} is of the form described in Lemma 6.1, and T_0 is of the form $\begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ & & & & 0 \end{pmatrix}$. Write T^a for the matrix

$$T^a = \begin{pmatrix} \lambda_1 T_{\lambda_2} & & & \\ & \ddots & & \\ & & \lambda_{t-1}T_{\lambda_{t-1}} & \\ & & & \lambda_t T_{\lambda_t} \end{pmatrix},$$

so that $T = \begin{pmatrix} T_0 & \\ & T^a \end{pmatrix}$ in $M(n_a,\mathbb{R})$, with $n_a = \sum_{\lambda\neq 0\in Spec(A_1)} dim(E_\lambda(A_1))$. Hence, if we write $Q = \begin{pmatrix} I_{n-n_1} & \\ & P_1 \end{pmatrix}$, and $R = PQ$, we have

$$B = R^{-1}AR = \begin{pmatrix} X & Y \\ 0 & T \end{pmatrix} = \begin{pmatrix} X & & Y \\ 0 & \begin{pmatrix} T_0 & \\ & T^a \end{pmatrix} \end{pmatrix}. \tag{14}$$

For $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^n$, we write $x^a = \begin{pmatrix} x_{n-n_a+1} \\ \vdots \\ x_n \end{pmatrix}$. Then, we may set $w = vR$ in $M(1, n, \mathbb{R})$, and write $w^a = (w_{n-n_a+1}, \ldots, w_n)$. We know how to compute the set $ANT(P(T^a, w^a))$ by the results of the previous section. Finally we obtain the following theorem.

Theorem 7.2. *The vector x belongs to $ANT(P(B, w))$ if and only if x^a belongs to $ANT(P(T^a, w^a))$. The vector y belongs to $ANT(A, v)$ if and only if $R^{-1}y$ belongs to $ANT(P(B, w))$,* i.e.*,*

$$ANT(P(A, v)) = R(ANT(P(B, w))).$$

In particular, the program $P(A, v)$ terminates if and only if the program $P(T^a, w^a)$ terminates. □

Example 7.1. Consider the program $P(\mathbf{a}, \mathbf{v})$ and the matrices $Mat_C(\mathbf{a}) = A$, and $Mat_C(\mathbf{v}) = w$ corresponding respectively to the linear forms $\mathbf{a}$ and $\mathbf{v}$ expressed in the canonical basis C:

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & -2 & -1 & 1 & 0 & 0 & 0 & 0 \\ 10 & -3 & -4 & 3 & 0 & 0 & 0 & 0 \\ 30 & -15 & -6 & 7 & 0 & -1 & 0 & 0 \\ 44 & -28 & -6 & 9 & 1 & -2 & 0 & 0 \\ 90 & -55 & -9 & 12 & 4 & -7 & 2 & 0 \\ 57 & -19 & -9 & 1 & 5 & -8 & 4 & -2 \end{pmatrix} \text{ and } v = (-1, -2, 1, 0, 0, 0, 0, 1).$$

The main step remains in the construction of a basis $B_{E_0,K}$ where $\mathbf{a}$ and $\mathbf{v}$ are, respectively, expressed by matrices B and w. See Eq. (14)). In order to proceed, we just need to follow the construction steps described just above. We obtain the following transform matrix R, and the matrices $Mat_{B_{E_0,K}}(\mathbf{a}) = B$ and $Mat_{B_{E_0,K}}(\mathbf{v}) =$

w:

$$R = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 3 & 1 & 0 & 0 & 0 & 0 & 0 \\ 4 & 5 & 2 & 1 & 0 & 0 & 0 & 0 \\ 6 & 5 & 3 & 2 & 1 & 0 & 0 & 0 \\ 4 & 2 & 5 & 2 & 1 & 1 & 0 & 0 \\ 3 & 8 & 8 & 2 & 1 & 2 & 1 & 0 \\ 2 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 \end{pmatrix},$$

and $w = (0, 1, 1, 1, 1, 1, 1, 1,)$.

Matrix R is constructed in such a way that $B = R^{-1}AR = \begin{pmatrix} X & Y \\ 0 & \begin{pmatrix} T_0 & \\ & T^a \end{pmatrix} \end{pmatrix}$, according to Eq.(14). Also, matrix B has the expected form with $X = \begin{pmatrix} 1 \end{pmatrix}$, $Y = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$, $T_0 = (0)$, and $T^a = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 \end{pmatrix}$.

Also, if we denote by e_1 and e_2 the two first elements of the canonical basis C, we obtain

$$K(A, v) = R(K(B, w)) = Vect(R(e_1)),$$

that is, $Vect(R(e_1))$ is the vector space spanned by the first column of the matrix R, and

$$E_0(B) = Vect(e_2).$$

We have $w = vR = (0, 1, 1, 1, 1, 1, 1, 1)$, and so $w^a = (1, 1, 1, 1, 1, 1)$, by construction. Finally, one can now directly apply Theorem 7.2. We obtain the following equivalence: (y is ANT for $P(A, v)$) if and only if ($x = (x_1, \ldots x_8)^\top = R^{-1}y$ is ANT for $P(B, w)$) if and only if ($x^a = (x_3, \ldots, x_8)^\top$ is ANT for $P(T^a, w^a)$). Then, the analysis of the ANT set for $P(A, v)$ is reduced to the generation of $ANT(P(T^a, w^a))$. As T^a, w^a describe the same systems than the one in Example 6.5, we already have the symbolic representation for $ANT(P(T^a, w^a))$, as the semi-linear system obtained in Example 6.5. To conclude and generate the ANT set for $P(A, v)$, one just need to rewrite that system considering the variables $(x_3, \ldots, x_8)$. □

8 Discussion

This work is complementary to our previous work [20], which dealt with termination analysis. As we showed in [20], the actual proof of the sufficiency condition required expertise in several independent mathematical fields. We then went on to generalize the previous result and, to the best of our knowledge, we presented the *first necessary and sufficient condition* (NSC, for short) for the termination of linear programs. Moreover, departing from this NSC, we showed the scalability of these approaches by demonstrating that one can directly extract a sound and complete computational method, running in polynomial time complexity, to determine termination or non-termination for linear programs. On the other hand, all other related and previous works mentioned in this paper do not provide any techniques capable of generating automatically the set of initial input variable values for which a loop does not terminate.

The main contributions of this paper resides on a sound and complete computational method to compute the set of input variable values for which programs do not terminate. The overall time complexity of our algorithm is also of order $\mathcal{O}(n^3)$. In our preliminary work [21, 19] we restricted our analysis to situations where the systems associated to the linear loops contained only one loop condition and the linear forms associated to the loop instructions were restricted to diagonalizable systems. In this preliminary work, we also assumed that $Spec(\mathbf{a})$ was over the reals. In this work we do not have all those restrictions: we handle complex eigenvalues, linear affine programs with conjunctions of n loop conditions, and the system does not have to be diagonalizable, as we also handle triangularizable systems.

As can be seen, Theorems 3.1, 5.1 and 5.2 show the importance of the ANT sets. They make it possible to reduce the problem of termination for linear programs to the emptiness check of the generated ANT set. Moreover, Theorem 6.1, Propositions 6.4, 6.6 and 6.8, and Theorem 6.2 provide us with a direct symbolic representation of the ANT set. Even if those theorems are mathematical in nature and proofs are quite technical, they are really easy to apply: we only need to compute the explicit terms S, U and V in order to directly obtain a formula representing exactly and symbolically the ANT set.

9 Conclusions

We presented the new notion of *asymptotically non-terminant initial variable values* for linear programs. Our theoretical results provided us with powerful computational methods that allow the automated generation of the sets of all asymptotically non-terminant initial variable values, represented symbolically and exactly by a semi-linear space, *e.g.*, conjunctions and disjunctions of linear equalities and inequalities. We

reduced the termination and non-termination problems for linear affine programs to the emptiness check of the *ANT* set of specific homogeneous linear programs. Also, by taking the complement set of the semi-linear set of ANT initial variable values, we obtain a precise under-approximation of the set of terminant initial variable values for the same program.

These theoretical contributions are rigorously stated with proofs that are quite technical. On the other hand, we show that those results can be directly applied in practical ways: one could only focus on the ready-to-use formulas representing the ANT set provided in this article. This type of method can be vastly generalized, to tackle the termination and non-termination problem of linear programs on rational or integer initial values, for instance. We leave this investigation for an ensuing report.

References

[1] Amir M. Ben-Amram and Samir Genaim. On the linear ranking problem for integer linear-constraint loops. In *Proceedings of the 40th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, POPL '13, pages 51–62, New York, NY, USA, 2013. ACM.

[2] Amir M. Ben-Amram, Samir Genaim, and Abu Naser Masud. On the termination of integer loops. In *VMCAI*, pages 72–87, 2012.

[3] Aaron R. Bradley, Zohar Manna, and Henny B. Sipma. Linear ranking with reachability. In *In CAV*, pages 491–504. Springer, 2005.

[4] Aaron R. Bradley, Zohar Manna, and Henny B. Sipma. Termination analysis of integer linear loops. In *In CONCUR*, pages 488–502. Springer-Verlag, 2005.

[5] Aaron R. Bradley, Zohar Manna, and Henny B. Sipma. Termination of polynomial programs. In *In VMCAI'2005: Verification, Model Checking, and Abstract Interpretation, volume 3385 of LNCS*, pages 113–129. Springer, 2005.

[6] Mark Braverman. Termination of integer linear programs. In *In Proc. CAV06, LNCS 4144*, pages 372–385. Springer, 2006.

[7] Hong Yi Chen, Shaked Flur, and Supratik Mukhopadhyay. Termination proofs for linear simple loops. In *Proceedings of the 19th international conference on Static Analysis*, SAS'12, pages 422–438, Berlin, Heidelberg, 2012. Springer-Verlag.

[8] Edmund M. Clarke, Orna Grumberg, and Doron Peled. *Model Checking*. MIT Press, Cambridge, MA, 2000.

[9] Michael Colón and Henny Sipma. Synthesis of linear ranking functions. In *Proceedings of the 7th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, TACAS 2001, pages 67–81, London, UK, 2001. Springer-Verlag.

[10] Michael A. Colón and Henny B. Sipma. Practical methods for proving program termination. In *In CAV2002: Computer Aided Verification, volume 2404 of LNCS*, pages 442–454. Springer, 2002.

[11] Byron Cook, Andreas Podelski, and Andrey Rybalchenko. Termination proofs for systems code. *SIGPLAN Not.*, 41(6):415–426, June 2006.

[12] P. Cousot and R. Cousot. Abstract interpretation and application to logic programs. *Journal of Logic Programming*, 13(2–3):103–179, 1992.

[13] Patrick Cousot and Radhia Cousot. An abstract interpretation framework for termination. *SIGPLAN Not.*, 47(1):245–258, January 2012.

[14] Dennis Dams, Rob Gerth, and Orna Grumberg. A heuristic for the automatic generation of ranking functions. In *Workshop on Advances in Verification*, pages 1–8, 2000.

[15] Zohar Manna. *Mathematical Theory of Computation.* McGraw-Hill, 1974.

[16] Andreas Podelski and Andrey Rybalchenko. A complete method for the synthesis of linear ranking functions. In *VMCAI*, pages 239–251, 2004.

[17] Jean-Pierre Queille and Joseph Sifakis. Specification and verification of concurrent systems in cesar. In *Proceedings of the 5th International Symposium in Programming*, pages 337–351, London, UK, 1982. Springer-Verlag.

[18] Rachid Rebiha, Nadir Matringe, and Arnaldo V. Moura. A complete approach for termination analysis of linear programs. Technical Report IC-13-08, Institute of Computing, University of Campinas, February 2013.

[19] Rachid Rebiha, Nadir Matringe, and Arnaldo V. Moura. Generating asymptotically non-terminant initial variable values for linear diagonalizable programs. Technical Report 13-06, Research Institute for Symbolic Computation, Johannes Kepler University Linz, Austria, July 2013.

[20] Rachid Rebiha, Nadir Matringe, and Arnaldo V. Moura. Necessary and sufficient condition for termination of linear programs. Technical Report IC-13-07, Institute of Computing, University of Campinas, February 2013.

[21] Rachid Rebiha, Nadir Matringe, and Arnaldo Vieira Moura. Generating asymptotically non-terminant initial variable values for linear diagonalizable programs. In Laura Kovacs and Temur Kutsia, editors, *SCSS 2013*, volume 15 of *EPiC Series*, pages 81–92. EasyChair, 2013.

[22] Henny B. Sipma, Tomás E. Uribe, and Zohar Manna. Deductive model checking. *Form. Methods Syst. Des.*, 15(1):49–74, July 1999.