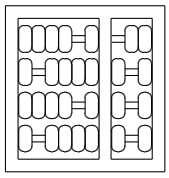


INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

A Complete Approach for Termination Analysis of Linear Programs

Rachid Rebiha *Nadir Matringe*
Arnaldo V. Moura

Technical Report - IC-13-08 - Relatório Técnico

February - 2013 - Fevereiro

A Complete Approach for Termination Analysis of Linear Programs

Rachid Rebiha* Nadir Matringe † Arnaldo Vieira Moura‡

Abstract

We describe powerful computational methods, relying on new decidability results that respond completely to major conjectures on termination analysis of programs with conditional linear loops, on all initial values. Our approach is based on linear algebraic methods: we reduce the verification of the termination problem to checking the orthogonality of a well determined vector-space and a certain vector, both related to the loop. We obtain necessary and sufficient conditions from which we provide the first complete method, which determines the termination of such a class of linear programs. Our examples (dealing with a large number of randomly generated linear loops) show the strength of our results, we actually prove that some of them are beyond the limits of other recent approaches.

1 Introduction

Formal methods for program verification research [1, 2, 3, 4] aim 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 [5, 2, 6], is used to check that a software is free of defects, such as buffers over flow 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 engineered system, safety critcal systems and embedded systems must guarantee. We could list here many verification approaches that are only practical, depending on the facility with which termination can be automatically determinated (e.g., verification of temporal properties of infinite state systems [7] is an other example.). More recent work on automated termination analysis of imperative loop programs has focused on a partial decision procedure based on the discovery and the synthesis of ranking functions. Such function maps the loop variable to a well-defined domain where their value decreases further at each iteration of the loop [8, 9]. Several interesting approaches, based

*Instituto de Computação, Universidade Estadual de Campinas, 13081970 Campinas, SP. Pesquisa desenvolvida com suporte financeiro da FAPESP, processo 2011089471

†Université de Poitiers, Laboratoire Mathématiques et Applications and Institue de Mathematiques de Jussieu Université Paris 7-Denis Diderot, France.

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

on the generation of *linear* ranking functions, have been proposed [10, 11] for loop programs where the guards and the instructions can be expressed in a logic with linear arithmetic. For the generation of such functions, there are effective heuristics [12, 9], and in some cases, there are also complete methods for the synthesis of linear ranking functions [13]. 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 the mentioned complete synthesis methods [13] fail to provide a conclusion on the termination or the non termination of such program.

In this work we address the termination problem for while linear loop programs. In other words we consider the class of loop programs where the loop condition is a conjunction of linear inequalities and the assignements to each of the variables (related to the loop instruction block), are of affine/linear form. In matrix notations, the *linear loop programs* will be represented in our most general form as:

$$\mathsf{while}\ (Bx > b),\ \{x := Ax + c\}.$$

Considering effective program transformations and simplification techniques, the 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 [14, 15, 16, 17, 18, 19, 20, 21], the problem of finding a practical, sound and complete method for determining termination or non termination remains very challenging for this class of programs on all initial values. We started our investigation from the line of research proposed by A.Tiwari [22].

We summarize our contributions as follows:

- First we prove a sufficient condition for the termination of homogeneous linear programs. This statement is contained in the important work proposed in [22], but the proof of the result contains a non trivially fixable mistake. The proof of this sufficient condition requires expertise in several independent mathematical fields. We show how this sufficient condition can be in used to determine termination of linear programs. We also draw its limitations.

- We then generalize the previous results. To the best of our knowledge, we present the *first necessary and sufficient condition* for the termination of linear programs. Infact, this NSC exhibits a complete decidability result for the class of linear programs on all initial values.

- Moreover, departing from this NSC, we show the scalability of our approach by demonstrating that one can directly extract a sound and complete computational method to determine termination or nontermination for linear programs. We reduce the termination analsysis for such program class to the problem consisting in checking if a specific vector (related to the loop condition encoding) belong to a specific vectorial space related to the eigenvalues of the matrix encoding the assignements of the loop variables.

- The analysis of our associated alorithms shows that our method operates in few and fast computational steps. The proposed computational method is of lower complexity than the mathematical foundations of previous methods.
- We provide theoretical results guaranteeing the soundness and completeness of the termination analysis while restrincting the variables interpretation over a specific countable subring of $\mathbb{R}^n$. In other words, we show that it is enough to interprete the variables over a specific countable field (or even its ring of integers) when one wants to check the termination over the reals.
- We provide experiments associated to our prototype. The cpu timing results while determining termination or non termination over a large number of random affine programs clearly demonstrate the practability of our approach.

The reader can find all the complete proofs, written rigourously, in this article or in our other associated technical report [23].

The rest of this article is ordered as follows. Section 2, can be seen as a preliminary section where we introduce our computational model of programs, the notations for the rest of the paper, and the key notions of linear used in order to build our computational methods. Section 3, provides the main theoretical contributions of this work. Infact, we present our decidability results and a very useful necessary and sufficient condition allowing us to propose a complete computational method described in Section 4. In Section 5, we show that our approach and algortihms, scale in the handling of the class of affine programs. In Section 6, we show how we interprete the variables over a countable field determining termination over the reals. Finally, Section 7 exhibits our experiments and Section 8 states our conclusion.

2 Linear Algebra and Linear Loop Programs

Here, we define key notions of linear algebra that are central in the theoretical and algorithmic development of our methods. If V is a *vector space* over a field $\mathbb{K}$, we write $Vect(v_1, ..., v_n)$ for the vector subspace generated by the family $v_1, ..., v_n$ of vectors of V. We denote by $\mathcal{M}(m, n, \mathbb{K})$ the set of $m \times n$ matrices with entries in $\mathbb{K}$ (and simply $\mathcal{M}(n, \mathbb{K})$ if $m = n$). If A belongs to $\mathcal{M}(m, n, \mathbb{K})$, with entry $a_{i,j}$ in position (i, j), we will sometimes denote it $(a_{i,j})$. The transpose of the matrix $A = (a_{i,j})$ is by definition the matrix $M^\top = (b_{i,j})$, such that $b_{i,j} = a_{j,i}$. The Kernel of A, also called its *nullspace*, and denoted by $Ker(A)$, is defined by: $Ker(A) = \{v \in \mathbb{K}^n \mid A \cdot v = 0_{\mathbb{K}^m}\}$. In fact, when we deal with square matrices, these Kernels are *Eigenspaces*. Let A be a $n \times n$ square matrix with entries in $\mathbb{K}$. A nonzero vector $x \in \mathbb{K}$ is an eigenvector for A associated with the eigenvalue $\lambda \in \mathbb{K}$ if: $A \cdot x = \lambda x$, i.e., $(A - \lambda I_n) \cdot x = 0$ 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. A non-zero vector x is said to be a *generalized eigenvector* for A corresponding to λ if $(A - \lambda I_n)^k \cdot x = 0$ for some positive integer k. The spaces $Ker((A - \lambda I_n)^k)$ form an increasing sequence of subspaces of k, which is stationary for $k \geq d$, for some $d \leq n$. We call the subspace $Ker((A - \lambda I_n)^d) = Ker((A - \lambda I_n)^n)$ the *generalized eigenspace* of A associated with λ.

We denote by $< , >$ the canonical scalar product on $\mathbb{R}^n$.

Notationally, as it is standard in static program analysis, a primed symbol x' refers to next state value of x after a transition is taken. First, we present *transition systems* as representations of imperative programs and *automata* as their computational models.

Definition 2.1. *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*, 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*, it is intuitively the pre-condition for the transition to be fired.*
- Θ *is the initial condition, given as a first-order assertion over* x*.*

The transition system is said to be affine *when* ρ_τ *is an affine form. And it is said to be* algebraic *when* ρ_τ *is an algebraic form.* □

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

Definition 2.2. *Let* P *be a loop program represented by the transition system*

$$\langle x = (x_1, ..., x_n), l_0, \mathcal{T} = \langle l_0, l_0, q_\tau, \rho_\tau \rangle, l_0, \Theta \rangle.$$

We say that P *is a* linear loop program *if the following conditions hold:*

- *the loop condition (i.e. the transition guard* g_τ*) is a conjunction of linear inequalities. We represent the loop condition in the matrix form* $Bx > b$ *where* $B \in \mathcal{M}(m, n, \mathbb{R})$ *and* $b \in \mathbb{R}^m$ *(by* $Bx > b$*, we mean that each coordinate of the column* Bx *is strictly greater than the corresponding coordinate of* b*).*
- *the transition relation* ρ_τ*, representing the assignements to each of the variables, is an affine/linear form. We represent the linear assignements (related to the loop instructions block) in the matrix form* $x := Ax + c$ *where* $A \in \mathcal{M}(n, \mathbb{R})$ *and* $c \in \mathbb{R}^n$*.*

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

In this work, we use the following linear loop program classifications.

Definition 2.3. *Let* P *be a linear loop program. We identify the following three type of linear loop programs, from the more specific to the more general form:*

- Homogeneous*: We denote by $P^{\mathbb{H}}$ the set of programs where all linear assignements consist of* homogeneous *expressions, and where the linear condition loop consists of at most one inequality. If P is in $P^{\mathbb{H}}$, then P will be interpreted in matrix terms as* **while** $(< w^{\top}, x >> 0)$, $\{x := Ax\}$*, where w 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 assignements of the loop. We say that P has a* homogeneous *form and it will be identified as $P(A, w)$.*

- Generalized Condition*: We denote by $P^{\mathbb{G}}$ the type of linear loop programs where the condition of the loop is* generalized *to a conjonction of multiple linear inequalities. Also the considered inequalities and assignements remain as homogeneous expressions. If P is in $P^{\mathbb{G}}$ then P will be interpreted as* **while** $(Bx > 0)$, $\{x := Ax\}$ *where B 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 $P(A, B)$.*

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

□

Example 2.1. *Consider the program depicted at the left below, for multiplying two numbers. Its computational model is described by the automaton at the right:*

(i) Pseudo code:

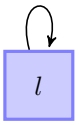
```
      ...
While (j>0){

  s := s+i;
  j := j-1;

 }
      ...
```

(ii) Transition systems:

$$\tau = \langle g_\rho = (j > 0), \rho_\tau = \begin{bmatrix} s' = s + i \\ j' = j - 1 \end{bmatrix} \rangle$$



with $V = \{s, i\}$, $\Theta = (s = 0 \wedge j = j_0)$, $l_0 = l$, $L = \{l\}$ and $\mathcal{T} = \{\tau\}$.

(iii) Matrix notations: $P(A, B, b, c)$ with $A = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $B = (0, 1, 0)$, $b = (0, 0, 0)^{\top}$ *and $c = (0, -1, 0)^{\top}$.*

□

3 New Decibability Results for Termination of Linear Programs

In this section we introduce the theoretical foundations of our approach. Here, we provide decidability results for the termination of the complete class of linear programs.

For this section, it is enough to consider only the class of homogeneous linear programs $P^{\mathbb{H}}$ (see Definition 2.3). In fact, as we will show in section 5, the problem of termination of linear programs in $P^{\mathbb{A}}$ (i.e. the class of affine programs, see Definition 2.3) reduces to the problem of termination of homogeneous linear programs $P^{\mathbb{H}}$.

First we establish a sufficient condition for the termination of homogeneous linear programs. Then, we present the main result, which provides the first necessary and sufficient condition for the termination problem considering the complete class of linear programs. Those decidability results lead us to a complete method, associated to fast algorithms to determines termination of linear programs.

3.1 Sufficient Condition for the Termination of Homogeneous Linear Programs

Here, we prove a sufficient condition for the termination of homogeneous linear programs $P(A, w) \in P^{\mathbb{H}} : \mathsf{while}\ (< w^{\top}, x >> 0),\ \{x := Ax\}$.

Theorem 3.1. *[23]-Theorem 1.*
Let n be a positive integer, and let $P(A, w)$ be program in $P^{\mathbb{H}}$, defined by the linear assignements encoded by a matrix A in $\mathcal{M}(n, \mathbb{R})$, and the inequality loop condition described by the vector $w \in \mathbb{R}^n - \{0\}$. If $P(A, w)$ is nonterminant, i.e. if there exists a vector $x \in \mathbb{R}^n$ such that $\langle A^k x, w\rangle > 0$ for all $k \geq 0$, then A has a positive eigenvalue.

The proof of Theorem 3.1 requires notions from in several independent mathematical fields. In fact, the core of the proof requires three lemmas and two propositions. The reader can find the complete proof in the annex of this article or in our associated technical report [23]. This statement can actually be found as Theorem 1 of the important work proposed in [22], however, the proof of the result contains a non trivially fixable mistake, which we explain. The author of [22] applies the Brouwer's fixed point theorem to a subspace of the projective space $P(\mathbb{R}^n)$ (not $\mathbb{R}^{n-1}$ as said in [22]). However, this is not an euclidian space, and convexity is not well defined in it, hence one can't apply Brouwer's fixed point theorem to such a set. Moreover, using notations of the proof of Theorem 1 of [22], the closure NT' of the set NT can contain zero, so that its image in $P(\mathbb{R}^n)$ is not well defined. Actually this extremal case needs to be treated carefully.

Theorem 3.1 provides a sufficient condition for the termination of linear program. In other words, Theorem 3.1 says that the linear program terminates when there is no positive eigenvalues, but one can not conclude on the termination problem using theorem 3.1 if there exists at least one positive eigenvalue. Intuitively, we could say that theorem 3.1 provides us with a decidability result for the termination problem considering the subclass of linear program where the associated assignement matrix A has no positive eigenvalues (i.e., all

```
/*...*/
 while(3x - y > 0){
   x := 3x - 2y;
   y := 4x - y;
   }
/*...*/
```

(a)

```
/*...*/
 while(z > 0){
   x:= x + y;
   z:= -z;
   }
/*...*/
```

(b)

Figure 1: Examples of homogeneous linear programs

eigenvalues are complex or negative). In the following example, we illustrate when Theorem 3.1 applies and when it does not.

Example 3.1. *Consider the homogeneous linear program 1a depicted in the figure 1 that we denote by* $P(A, v)$*. The associated matrix* A *is given by* $A = \begin{pmatrix} 3 & -2 \\ 4 & -1 \end{pmatrix}$*, and the vector* v *enconding the loop condition, is such that* $v = (3, -1)^\top$*. The eigenvalues of the matrix* A *are the complex numbers:* $1 + 2i$ *and* $1 - 2i$*. As* S *does not have any positive eigenvalues, we can consider the contrapose of Theorem 3.1's statement, and conclude that the program* $P(A, v)$ *terminates on all possible inputs.*

Example 3.2. *Now, consider the homogeneous linear program 1b depicted in Figure 1, that we denote by* $P(A_1, v_1)$*. The associated matrix* A_1 *given by* $A_1 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$*, has eigenvalues* 1 *and* -1*. As* A *has a positive eigenvalues, one can not determine the termination (or the nontermination) of* $P(A_1, v_1)$ *using the theorem 3.1.*

However, We will see how to handle this case in a very automated efficient way in the following sections. □

In the next section, we generalize Theorem 3.1, and obtain stronger decidability results.

3.2 Necessary and Sufficient Condition for the Termination of Linear Program

In this section, we strengthen the theorem 3.1, in order to obtain a complete decidability result leading us to a sound and complete methods with very few computational steps executed by fast algorithms.

Infact, in the following main theorem 3.2 we provide a necessary and sufficient condition for the termination of programs $P(A, v) \in P^{\mathbb{H}} : \mathsf{while}\ (< v^\top, x >> 0),\ \{x := Ax\}$.

Theorem 3.2. *[23]-Theorem 2.*
Let $A \in \mathcal{M}_n(\mathbb{R})$ *and* $w \neq 0 \in \mathbb{R}^n$*. The program* $P(A, v) : \{x := Ax,\ < v, x >> 0\}$ *terminates if and only if for every positive eigenvalue* λ *of* A*, the generalised eigenspace* $E_\lambda(A)$ *is orthogonal to* v *(i.e.* $< E_\lambda(A), v >= 0$*).*

Theorem 3.2 gives a necessary and sufficient condition that we use as the foundation to build a complete procedure. In order to determine termination, we have to check, for each positive eigenvalues, if the vector v, encoding the loop condition, is orthogonal to the associated generalized eigenspace. In other words we want to verify if v is orthogonal to the nullspace $Ker((A - \lambda I_n)^n)$.

Example 3.3. *Consider the program 1b depicted in Figure 1 that we denoted as $P(A_1, v_1)$. The matrix A_1 is given in Example 3.1. The vector enconding the loop condition is $v_1 = e_3 = (0, 0, 1)^\top$. We recall that A_1 has eigenvalues 1 and -1. The generalised eigenspace $E_1(A_1)$ is equal to $Vect(e_1, e_2)$, where e_1 and e_2 are the first two vectors of the canonical basis of $\mathbb{R}^3$. Hence $E_1(A_1)$ is orthogonal to v_1. According to Theorem 3.2, the program $P(A, w)$ terminates.* □

Example 3.4. *Now, if we change the loop condition of the program 1b depicted in Figure 1 to become $(y > 0)$. Then, we obtain the program $P(A_1, v_2)$ with the new considered loop condition encoded by $v_2 = e_2 = (0, 1, 0)^\top$. The eigenvalues of A_1 are (still) 1 and -1 and the generalised eigenspace $E_1(A_1) = Vect(e_1, e_2)$. Hence $E_1(A)$ is not orthogonal to v_2, because it contains v_2. Theorem 3.2 tells us that the program $P(A_1, v_2)$ does not terminate in this case.* □

In both of these examples, we are able to determine the termination/nontermination using Theorem 3.2. On the other hand, the first Theorem 3.1 does not allow us to say anything about the termination of these programs (because the assignement matrix A' exhibit at least one positive eigenvalue).

In order to avoid the computation of the basis of generalised eigenspaces, we first introduce the notion of *Row_Space*(M), and use the following lemma. Let $M \in \mathcal{M}(m, n, \mathbb{R})$, by definition, *Row_Space*(M) denotes the vector subspace of $\mathbb{R}^n$ spanned by the row vectors of M.

Lemma 3.1. *Let M be a matrix in $\mathcal{M}(m, n, \mathbb{K})$. Then every vector in the nullspace of M is orthogonal to every vector in the Row_Space(M).* □

Proof. Let w be in $Ker(M)$ and v in the column space of $M^\top$. We denote by $\{c_1, ..., c_m\}$ the set of column vectors of $M^\top$. Then, exists a vector $k \in \mathbb{R}^n$ such that $v =< M, k >$ (because v is a linear combination of the column vectors of $M^\top$). Now, we have $< w, v >= w^\top \cdot v = w^\top \cdot < M, k >= w^\top \cdot M^\top \cdot k = (M \cdot w)^\top \cdot k = 0$ because $w \in Ker(M)$ and $M \cdot w = 0$. □

From Lemma 3.1, a basis of *Row_Space*(M) is a basis of the orthogonals of $Ker(M)$. Thus, for square matrix A, a vector v is orthogonal to $Ker((A - \lambda I_n)^n)$ (i.e. $< E_\lambda(A), v >= 0$) if an only if v belongs to *Row_Space*$((A - \lambda I_n)^n)$. We directly deduce the following corollary.

Corollary 3.1. *Let $A \in \mathcal{M}_n(\mathbb{R})$ and $v \neq 0 \in \mathbb{R}^n$. The program $P(A, v)$ terminates if and only if for every positive eigenvalue λ of A, v belongs to the vector space Row_Space$((A - \lambda I_d)^n)$.* □

Proof. Using Lemma 3.1, we know that the basis of $Row\_Space((A-\lambda I_d)^n))$ is the basis of the orthogonals of $Ker((A-\lambda I_d)^n))$. We can then us directly Theorem 3.2 to complete the proof. □

In practice, we will use the corollary 3.1 through three computational steps associated to fast algorithms as it is illustrated in the following example.

Example 3.5. (Running example) *Consider the program $P(A,v)$ depicted as follow:*

(i) Pseudo code:

```
while(z+t-x-y>0){
  x := 2x - y;
  y := -x + 2y -z;
  z := -y + 2z +t;
  t := 2t;}
```

(ii) Associated matrices:

$$A=\begin{pmatrix}2&-1&0&0\\-1&2&-1&0\\0&-1&2&1\\0&0&0&2\end{pmatrix},\ \textit{and}\ v=\begin{pmatrix}0\\0\\-1\\1\end{pmatrix}.$$

Step 1: We compute the list e_λ of positive eigenvalues for A:

```
[[2 - sqrt(2), sqrt(2) + 2,2], [1, 1, 2]]
```

Hence, we have three positive eigenvalue $\lambda_1=2,\lambda_2=2-\sqrt{2},\lambda_3=2+\sqrt{2}$ (with, respectively, the multiplicity 2, 1 and 1).
Step 2: We compute the matrix $E_\lambda=(A-\lambda I_n)^n$ for $\lambda\in e_\lambda$:

```
(A - (e[i])*Id_m)^d
[         18  16*sqrt(2)          14  -4*sqrt(2)]
[ 16*sqrt(2)          32  16*sqrt(2)         -14]
[         14  16*sqrt(2)          18 -12*sqrt(2)]
[          0           0           0           4]
```

Step 3: We check if $v\in Row\_Space(E)$:
Here we use a standard procedure, from linear algebra, used to check if a given vector belongs to a vector space spanned by a list of vectors. We compute the unique reduced row echelon form *of the matrix $E_\lambda^\top$. It means that we operate a* Gaussian elimination *on the rows using the* Gauss-Jordan elimination *algorithm. The generated matrix provides us a basis of $Row\_Space(E_\lambda)$:*

```
(E[i].T).echelon_form()
[       1        0        0 -sqrt(2)]
[       0        1        0        2]
[       0        0        1 -sqrt(2)]
[       0        0        0        0]
```

We augement the computed basis with the vector v to form the following matrix.

```
block_matrix([ [Er[i], V.T]])
[       1        0        0 -sqrt(2)|       1]
[       0        1        0        2|      -1]
[       0        0        1 -sqrt(2)|       0]
[       0        0        0        0|       1]
```

Finally, we generate its reduced row echelon form to obtain the matrix R_S_λ:

```
(block_matrix([ [Er[i], V.T]])).echelon_form()
[       1        0        0 -sqrt(2)|       0]
[       0        1        0        2|       0]
[       0        0        1 -sqrt(2)|       0]
[       0        0        0        0|       1]
```

From the Gauss-Jordan elimination properties, it is well-known that v belongs to the space $Row\_Space(E_\lambda)$ if and only if $R\_S_\lambda(n, n+1) = 0$. Here we have $R\_S_\lambda(n, n+1) = 1$, which means that v is not in $Row\_Space(E_\lambda)$. Thus, by Corollary 3.1 one concludes that the program $P(A, v)$ is nonterminant. □

The necessary and sufficent conditions (see Theorem 3.2 and its Corollary 3.1) obtained, allows us to determine the termination of any homogeneous linear program, considering all initial values. In section 5, we will see that the termination analysis of affine linear programs in $P^{\mathbb{A}}$, reduces to the class of homogeneous linear programs. Thus the presented necessary and sufficient condition provides a decidability result and a complete computational method for determining the termination of the full class of linear/affine programs. As we show in Example 3.5, we avoid the computation of generalized eigenspaces in practice, and instead, use the exact algorithm associated to Corollary 3.1. The following section presents, in more details, our complete approach and its associated algorithm.

4 Complete Procedure to Determine Termination and Non-termination

We use the necessary and sufficient conditions provided by Theorem 3.2 and its related practical corollary 3.1 to build a sound and complete procedure to establish the termination of linear programs. Moreover, the method obtained is based on few computational steps associated to fast numerical algorithms.

The pseudo code depicted in Algorithm 1 illustrates the strategy.

Our algorithm takes as input the number of variables, the choosen field where the variables are interpreted, the assignement matrix A and the vector w encoding the loop condition. We first compute the list of posivite eigenvalues (see 1, line 1 and 2), if this list is empty we can then respond that the loop is terminant (see 1, line 3 and 4). Otherwise we will have to countinue our analysis on the nonempty list of positive eigenvalues. For each positive eigenvalues $e'[i]$ we will first need to compute the matrix $E_i = (A - e'[i]I_n)^n$ (see 1, line 6). Using Corollary 3.1, we know that the loop is terminant if and only if w is in the *Row_Space* of $(A - e'[i]I_n)^n$ for every positive eigenvalue $e'[i]$. In other words, for each positive eigenvalue, we have to check if w is in the vector space spanned by the basis of the *Row_Space* of the associated matrix E_i. In order to do so, one first need to consider the linearly independent vectors $\{r_1, ..., r_n\}$ that form a basis of the *Row_Space* (this basis is obtained from the list of row vectors of E_i). The efficient way to check if w is in the vector space spanned by the basis $\{r_1, ..., r_n\}$ is composed by the following computational steps:

1. We build the augmented matrix E_A formed by the vectors $r_1, ..., r_n$ and w (see 1, line 7).

2. We compute the *reduced row echelon form* of matrix E_A (see 1, line 8). It means that we applied *Gaussian elimination* on the rows. This reduced, canonical form is unique and exactly computed by *Gauss-Jordan elimination*.

3. We know that the added vector w is in the vector space spanned by $r_1, ..., r_n$ if and only if the bottom right entry of the reduced row echelon matrix E_R is null.

Thus if $E_R(n, n+1) \neq 0$, we conclude that there exists a positive eigenvalue $e'[i]$ such that w is not in $Row\_Space(A - e'[i]I_n)^n$, which is equivalent to say that the loop is nonterminant (see 1, line 9 and line 10). Otherwise if he have exhausted the list of positive eigenvalues and always found that w is in the *Row_Space* of the associated matrix, we conclude that the loop is terminant see (1, line 11).

Algorithm 1: Termination_linear_Loop $(n, \mathbb{K}, A, w)$

```
   /*Determining the termination for linear homogeneous programs.*/;
   Data: n the number of program variables, K the field, P(A,w) ∈ P^H where
         A ∈ M(n,K) and w ∈ M(n,1,K)
   Result: Determine the Termination/Nontermination
   begin
1      {e[1],...,e[r]} ⟵ eigenvalues(A);
2      {e'[1],...,e'[s]} ⟵ striclty_positives({e[1],...,e[r]});
3      if {e'[1],...,e'[s]} = ∅ then there is no positive eigenvalues.
4          return TERMINANT;
5      for i = 1 to s do
6          E ⟵ (A − e'[i]I_n)^n;
7          E_A ⟵ augmented(E_A^⊤, w);
8          E_R ⟵ echelon_form(E_A);
9          if E_R(n, n+1) ≠ 0 then
10             return NONTERMINANT;
11     return TERMINANT;
```

The function **echelon_form** computes the reduced row echelon by Guass-Jordan elimination and its time complexity is of order $O(n^3)$. We interprete the variables in a specified field (i.e. an extension of $\mathbb{Q}$) choosen according to the discussion made in section 6. By using efficient mathematical packages (e.g. Maple, Mathematica, Sage, Lapack, Eispack, ...) one can expect the eigenvalues to be in closed-form algebraic expression (i.e. the solution of an algebraic equation in terms of the coefficients, relying only on addition, subtraction,

multiplication, division, and the extraction of roots) for our experiments. Also, with $n < 5$, the eigenvalues computed by the function **eigenvalues** are already exhibited as such algebraic numbers. Moreover, the algorithm for eigenvalue computation has a time complexity that is of order $O(n^3)$, and this beeing said, the overall time complexity of the algorithm **Termination_linear_Loop** remains of the same order.

5 Termination for Linear Programs Reduce to Homogeneous Forms

In this section we show how the termination problem for the classes $P^{\mathbb{G}}$ and $P^{\mathcal{A}}$ (see Definition 2.3) can be reduced to the problem of termination of programs in the class $P^{\mathbb{H}}$. In other words, we show how Theorem 3.2, Corollary 3.1 and the induced computational method detailed in the previous Section 4 extends to the complete class of linear programs.

5.1 From Generalized Condition to Homogeneous Programs

In this section, we treat the case where the loop condition is *generalized* to a conjunction of a finite number of linear inequalities. These inequalities are encoded by a matrix B of $\mathcal{M}(m, n, \mathbb{R})$. Let $P(A, B) : \mathsf{while}(Bx > 0)\{x := Ax\}$ be a program in the class $P^{\mathbb{G}}$, where B is an element of $\mathcal{M}(m, n, \mathbb{R})$, x is a vector in $\mathbb{R}^n$ and A is an element of $\mathcal{M}(n, \mathbb{R})$. We will say that Bx is positive, and write $Bx > 0$, if each coordinate of the vector $Bx \in \mathbb{R}^m$ is > 0. If B has top row row_1, then second row row_2, ..., last bottom row row_m, and $Bx = y = (y_1, \ldots, y_m)^\top$, then $y_i = row_i.x =< row_i^\top, x >$. Hence to say that $Bx > 0$, is to say that for i between 1 and m, the scalar product $< row_i^\top, x >$ is strictly positive.

Hence, if we consider the general program associated $P(A, B)$ which does $x := Ax$ as long as $Bx > 0$, it will be terminating if and only if one of the programs $P(A, row_i^\top)$ is terminating for i between 1 and m. Following this statement, we establish the following Theorem 5.1.

Theorem 5.1. *Let A be a matrix in $\mathcal{M}(n, \mathbb{R})$ and B be a matrix in $\mathcal{M}(m, n, \mathbb{R})$. And we denote by $(row_1, \cdots row_m)$ the m row vectors of B. The program $P(A, B)$ is terminating if and only if there is $i \in 1, \ldots, m$ such that for all positive eigenvalues λ of A, the generalised eigenspace $E_\lambda(A)$ is orthogonal to $row_i^\top$.* □

Proof. if we consider the general program associated $P(A, B)$ which does $x := Ax$ as long as $Bx > 0$, it will be terminating if and only if one of the programs $P(A, row_i^\top)$ is terminating for i between 1 and m. By Theorem 3.2, we know that the statement "$P(A, row_i^\top)$ is terminating" is equivalent to say that for every positive eigenvalue λ of A, the generalised eigenspace $E_\lambda(A)$ is orthogonal to $row_i^\top$ □

The following example illustrates the application of Theorem 5.1 on two programs.

Example 5.1. *Consider the program $P(A_1, B_1)$ depicted as follow:*

(i) Pseudo code:

```
while((x+z>0)&&(x+y>0)){
  x := 2x + y +3z;
  y := -y;
  z := 2z;}
```

(ii) Associated matrix:

$$A_1 = \begin{pmatrix} 2 & 1 & 3 \\ 0 & -1 & 0 \\ 0 & 0 & 2 \end{pmatrix}, \text{ and } B_1 = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}.$$

Then the program $P(A_1, B_1)$ is nonterminating, because 2 is the only positive eigenvalue of A, and the generalised eigenspace $E_2(A_1) = Vect(e_1, e_3)$ has $Vect(e_2)$ as an orthognal, which contains neither $(1,0,1)^\top$, nor $(1,1,0)^\top$. □

Example 5.2. *Consider the same program depicted in 5.1, but we change the loop condition to the following one: $(x++y+z>0) \wedge (y>0)$. This modified loop condition is encoded by the matrix matrix $B_2 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$. The assignement matrix remains unchanged and we still consider A_1.*

The program $P(A_1, B_2)$ terminates because the second row of B_2 is $e_2^\top$, and e_2 is orthogonal to $E_2(A_1)$. □

In practice we implement and use the following corollary.

Corollary 5.1. *Let A be a matrix in $\mathcal{M}(n, \mathbb{R})$ and B be a matrix in $\mathcal{M}(m, n, \mathbb{R})$. We denote by $\{row_1, ..., row_m\}$ the row vectors of B. The program $P(A, B) \in P^{\mathbb{G}}$ is terminating if and only if there is $i \in 1, \ldots, m$ such that for all positive eigenvalues λ of A, the vector $row_i^\top$ belong to $Row\_Space((A - \lambda I_d)^n)$.* □

Proof. The proof is very similar to the one detailed in the proof of the previous Theorem 5.1. Here we just need to apply Corollary 3.1 instead. □

Example 5.3. *Consider the program $P(A, B) \in P^{\mathbb{G}}$ depicted as follow:*

(i) Pseudo code:

```
while((x+z>0)&&
      (y+z>0)&&
      (x+y>0)){
  x := 4x + 2y +6z;
  y := 2y;
  z := 6x+ 4y + 4z;}
```

(ii) Associated matrices:

$$A = \begin{pmatrix} 4 & 2 & 6 \\ 0 & 2 & 0 \\ 6 & 4 & 4 \end{pmatrix}, \text{ and } B = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}.$$

First, We compute the list e_λ of positive eigenvalues for A: [10, 2, [1,1]]. Hence, we have two positive eigenvalues $\lambda_1 = 10, \lambda_2 = 2$ (both with multiplicity 1).
We check if there exists a row r_i in matrix B such that $r_i^\top$ is in $Row\_Space((A - \lambda_1 I_d)^3))$ and in $Row\_Space((A - \lambda_2 I_d)^3))$. By Corollary 5.1, we know that if so, the program $P(A, B)$ terminates, and if there is no such r_i for all $i \in \{1 \ldots 3\}$, the program $P(A, B)$ is nonterminating.

In this case, matrix B has three rows: $r_1 = (1,0,1)$, $r_2 = (0,1,1)$ and $r_3 = (1,1,0)$.

- *We start with the row r_1 and we check first if it is in $Row\_Space((A - \lambda_1 I_d)^3))$. Our algorithm performs the computational steps illustrated in Example 3.5, by calling the*

procedure described in Section 4. We obtain the following final augemented matrix in row reduced echelon form:

```
(block_matrix([ [Er[i], B[j].T]])).echelon_form()
[     8  20992 -18376      0]
[     0  55296 -48384      0]
[     0      0      0      1]
```

We can see that the last element of the last column of the final matrix R_S (described just above) is not null as $R\_S(3,4) = 1$*, which is equivalent to say that* $r_1^\top$ *is not in* $Row\_Space((A-\lambda_1 I_d)^3)$*. So we have to try with an other row* r_i *of* B *with* $i \in \{2,3\}$*.*

- *We try with* $r_2 = (0,1,1)$*, and we find that* $r_2^\top$ *is not in* $Row\_Space((A-\lambda_1 I_d)^3))$ *calling the same procedure.*

- *Then, we try with the last row* $r_3 = (1,1,0)$ *of* B*.*

  – *We check if* $r_3^\top \in Row\_Space((A-\lambda_1 I_d)^3))$*. We obtain the final augemented matrix in row reduced echelon form:*

  ```
  (block_matrix([ [Er[i], B[j].T]])).echelon_form()
  [     8  20992 -18376      1]
  [     0  55296 -48384      1]
  [     0      0      0      0]
  ```

  Hence, we know that $r_3^\top \in Row\_Space((A-\lambda_1 I_d)^3))$ *(as the bottom row of the matrix above is null) and we have now to consider the other positive eigenvalue* λ_2*.*

  – *We check if* $r_3^\top \in Row\_Space((A-\lambda_2 I_d)^3))$*. We obtain the final augemented matrix in row reduced echelon form:*

  ```
  (block_matrix([ [Er[i], B[j].T]])).echelon_form()
  [  8   0 184   1]
  [  0   0 256   1]
  [  0   0   0   0]
  ```

  Hence we have $r_3^\top \in Row\_Space((A-\lambda_2 I_d)^3))$*.*

Finally, one can conclude that $r_3^\top$ *is in the row space of* $(A-\lambda_1 I_d)^3)$ *associated to the first positive eigenvalue* λ_1 *and it also belongs to the row space of* $(A-\lambda_2 I_d)^3)$ *associted to the other and last positive eigenvalue* λ_2*. By Corollary 5.1, we conclude that the program* $P(A,B)$ *terminates.*

□

5.2 Termination Analysis for Affine Programs

We ow show that the affine case reduces to the homogeneous case. Moreover, in the following procedure, we show how one can apply directly Theorem 3.2 and its corollary 3.1 to establish termination of affine programs.

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

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

We augmented the matrix A with the vector c and the row $(0,\cdots 0,1)$, and the matrix B with the vector $-b$ and the row $(0,\cdots 0,1)$. Here we adapt the Proposition 2 of [22] on the reduction of affine program in order to extend our necessary and sufficient condition to the class $P^{\mathbb{A}}$. The programm $P(A,B,b,c)$ terminates if and only if the homogeneous program $P(A',B')$ (which does $x' := A'x'$ as long as $B'x' > 0$) terminates. Considering the reduction of the termination analysis to the class $P^{\mathbb{H}}$ done in the previous section. We can already note that the termination analysis of programs in $P^{\mathbb{A}}$ reduces to the same analysis for programs in $P^{\mathbb{H}}$.

Theorem 5.2. *For $A \in \mathcal{M}(n,\mathbb{R})$, $B \in \mathcal{M}(m,n,\mathbb{R})$, $b = (b_1,\ldots,b_m)^\top$ a vector in $\mathcal{M}(1,m,\mathbb{R})$ and c a vector in $\mathcal{M}(1,n,\mathbb{R})$. Let $B' \in \mathcal{M}(m+1,n+1,\mathbb{R})$ and $A' \in \mathcal{M}(n+1,\mathbb{R})$ be the matrices built as such: $B' = \begin{pmatrix} B & -b \\ 0 & 1 \end{pmatrix}$ and $A' = \begin{pmatrix} A & c \\ 0 & 1 \end{pmatrix}$. We denote by row_i the i-th row of B and by r_i the i-th row of B'. By the definition of B', we have $r_1 = (row_1,-b_1),\ldots,r_m = (row_1,-b_m)$, and $r_{m+1} = (0,\ldots,0,1)$.*
The program $P(A,B,b,c)$ is terminating if and only if there is $i \in 1,\ldots,m+1$ such that for all positive eigenvalues λ of A', the generalised eigenspace $E_\lambda(A')$ is orthogonal to r_i. □

Proof. The programm $P(A,B,b,c)$ terminates if and only if the homogeneous program $P(A',B')$ by construction of the matrix A' and B'. To prove this statement we can thus apply directly Theorem 5.1 for $P(A',B')$. □

The following Example 5.4, illustrate the application of Theorem 5.2 on two programs.

Example 5.4. *Consider the affine program $P(A_1,B_1,b_1,c_1) \in P^{\mathbb{A}}$ depicted as follow:*

(i) Pseudo code:

```
while((x+z>1)&&(x+y>1)){
  x := x + y +3z+1;
  y := -y;
  z := z+1;}
```

(ii) Associated matrix:

$A_1 = \begin{pmatrix} 1 & 1 & 3 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, $B_1 = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$, $b_1 = (1,1)^\top$ *and* $c_1 = (1,0,1)^\top$.

We define the matrix A'_1, B'_1 *such that:*

$$A'_1 = \begin{pmatrix} 1 & 1 & 3 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \text{ and } B'_1 = \begin{pmatrix} 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Then the program $P(A_1, B_1, b_1, c_1)$ *is nonterminating, because* 1 *is the only positve eigenvalue of* A'_1*, and the generalised eigenspace* $E_1(A'_1) = Vect(e_1, e_3, e_4)$ *has* $Vect(e_2)$ *as an orthognal, which contains none of the transpose of the rows of* B'_1. □

Example 5.5. *Consider again the program depicted in 5.4, where we change the loop condition to the following one:* $(x{+}{+}y{+}z > 1) \wedge (y > 0)$*. This modified loop condition is encoded by the matrix matrix* $B_2 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$ *and the vector* $b_2 = (1,0)^\top$*. The assignment matrix remains unchanged and we still consider* A_1 *and* c_1*. Then the matrix* A'_1 *introduced in the previous program at Example 5.4, remains unchanged and we have* $B'_2 = \begin{pmatrix} 1 & 0 & 1 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$.

The program $P(A_1, B_2, b_2, c_1)$ *terminates because the second row of* B_2 *is* $e_2^\top$*, but* e_2 *is orthogonal to* $E_1(A'_1)$*, and* 1 *is the only positive eigenvalue of* A_1. □

In practice we use the following corollary.

Corollary 5.2. *Let* A*,* A'*,* B*,* B'*,* b *and* c *be the matrices and vectors introduced in the statement of Theorem 5.2. We denote by* $\{r_1, \ldots, r_m\}$ *the row of the matrix* B'*. The program* $P(A, B, b, c)$ *is terminating if and only if there is* $i \in 1, \ldots, m+1$ *such that for all positive eigenvalues* λ *of* A'*, the vector* $r_i^\top$ *belong to* $Row\_Space((A' - \lambda I_d)^n)$. □

Proof. The proof is obtain directly using Theorem 5.2 and Corollary 3.1. □

Example 5.6. *Consider the program* $P(A, B, b, c) \in P^{\mathbb{A}}$ *depicted as follow:*

(i) Pseudo code:

```
while((2x+y+z>-2)&&
      (y-z>-1)){
  x := 7x -3z +2;
  y := -9x -2y -2/3z +1;
  z := 18x -8z +1;}
```

(ii) Associated matrices:

$A = \begin{pmatrix} 7 & 0 & -3 \\ -9 & -2 & -2/3 \\ 18 & 0 & -8 \end{pmatrix}$, $B = \begin{pmatrix} 2 & 1 & 1 \\ 0 & 1 & -1 \end{pmatrix}$,
$b = (-2,-1)^\top$ *and* $c = (2,1,1)^\top$.

First we build the matrix A' *and* B' *according to their definition given in Theorem 5.2.*

```
Aprime                                  Bprime
[   7    0   -3|   2]                  [ 2  1  1| 2]
[  -9   -2 -2/3|   1]                  [ 0  1 -1| 1]
[  18    0   -8|   1]                  [--------+--]
[--------------+----]                  [ 0  0  0| 1]
[   0    0    0|   1]
```

Following the Corollary 5.2, we know that program $P(A, B, b, c) \in P^{\mathbb{A}}$ *is terminant if and only if the program* $P(A', B') \in P^{\mathbb{G}}$ *is terminant. Hence we have to proceed with the procedure given by Corollary 5.1, and illustrated in Example 5.3.*

We compute the positive eigenvalues of A'*.*

```
[1, [2]]
```

We have one positive eigenvalue $\lambda_1 = 1$ *(with multiplicity* 2*). Thus, we only need to check if there exists a row* r_i *in matrix* B' *such that* $r_i^\top$ *is in* $Row\_Space((A-\lambda_1 I_d)^4))$*. By Corollary 5.1, we know that if so, the program* $P(A', B')$ *terminates, whereas if there is no such* r_i *for all* $i \in \{1 \dots 3\}$*, the program* $P(A', B')$ *is nonterminating.*

In this case, the matrix B' *has three rows:* $r_1 = (2, 1, 1, 2)$*,* $r_2 = (0, 1, -1, 1)$ *and* $r_3 = (0, 0, 1)$*. We start by considering* r_1*. We augment the basis of the row space of* $(A' - I_d)^4$ *with the vector* r_1*. Then put the obtained matrix in its redruced row echelon form* R_S *given below.*

```
[      1       0    -1/2    -1/2|      0]
[      0       1   31/18 -247/54|      0]
[      0       0       0       0|      1]
[      0       0       0       0|      0]
```

We conclude that r_1 *is in* $Row\_Space((A - \lambda_1 I_d)^4))$ *because* $R\_S(4, 5) = 0$*. As* λ_1 *is the unique positive eigenvalue, we can already conclude, by using Corollary 5.2, that the program* $P(A, B, b, c)$ *is terminant.* □

6 Interpreting the Variables Over Countable Sets is Sufficient

In this section, we show that we can restrict the interpretation of the variables to a specific countable subset of $\mathbb{R}^n$ while we prove the termination over the reals.

First, we study an example, which is already interesting in itself, and which will prove that we can not restrict the interpretation of the variable to the rational field $\mathbb{Q}$ if we want to prove the termination for all real inputs.

We start with two elements of $\mathbb{Q}(\sqrt{2}) - \mathbb{Q}$, which are conjugate under the Galois group $Gal_{\mathbb{Q}}(\mathbb{Q}(\sqrt{2}))$, of opposite signs, and the negative one of absolute value strictly greater than the one with positive absolute value. For instance, take

$$\lambda^- = -1 - \sqrt{2}, \text{ and } \lambda^+ = -1 + \sqrt{2}.$$

They are the roots of the polynomial $P(X) = (X - \lambda^-)(X - \lambda^+) = X^2 + 2X - 1$. Now let $A = \begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix}$ be the associated compagnon matrix, so that its characteristic polynomial is P, and its eigenvalues are λ^- and λ^+. Its generalised eigenspaces are easy to compute, and we find:
$E_{\lambda^-}(A) = \mathbb{R}.\begin{pmatrix} 1 \\ \lambda^- \end{pmatrix} = \mathbb{R}.e^-$ and $E_{\lambda^+}(A) = \mathbb{R}.\begin{pmatrix} 1 \\ \lambda^+ \end{pmatrix} = \mathbb{R}.e^+$. Now let $v = (1,0)^\top$, we have $< v, e^+ >= 1$, hence, according to Theorem 3.2, the program $P(A, v)$ associated to A and v does not terminate. We can actually locate the points of $\mathbb{R}^2$ for which the program is not terminating.

Proposition 6.1. *Let A, v and P_1 be as above, the program P_1 does not terminate for initial condition $x \in \mathbb{R}^2$, if and only if $x \in E_{\lambda^+}(A)$ and $< x, v >> 0$, i.e. $x \in \mathbb{R}_{>0}.e^+$.* □

Proof. If $x = t.e^+$, with $t > 0$, then $A^k(x) = t\lambda^{+k}.x$, and $< v, A^k(x) >= t\lambda^{+k} > 0$ for all $k \geq 0$, hence the program does not terminate for such x as initial condition.
Conversely, suppose that x satsifies $< v, A^k(x) >> 0$ for all $k \geq 0$. Decompose x on the basis (e^-, e^+). Then $x = s.e^- + t.e^+$, and $A^k(x) = s\lambda^{-k}.e^- + t\lambda^{+k}.e^+$, so that $< v, A^k(x) >= s\lambda^{-k} + t\lambda^{+k}$.
Suppose that s is not zero. As $|\lambda^-| > |\lambda^+|$, for k large enough, the scalar $< v, A^k(x) >$ will be of the same sign as $s\lambda^{-k}$, which is alternatively positive and negative. This is absurd, hence $s = 0$.
Now as $< v, A^k(x) >= t\lambda^{+k}$, this imples that $t > 0$, and we the Proposition is proved. □

Proposition 6.1 leads us to the following corollary.

Corollary 6.1. *With A and v as above, the program P_1 is terminating on $\mathbb{Q}^2$, but not on $\mathbb{R}^2$* □

Proof. We already saw that P_1 does not terminate on $\mathbb{R}^2$. Now let x be an element of $\mathbb{Q}^2$. If P_1 was not terminating with x as an initial value, this would imply that x belongs to $\mathbb{R}_{>0}.e^+$ according to Lemma 6.1. However, no element of $\mathbb{Q}^2$ belongs to $\mathbb{R}_{>0}.e^+$, because the quotient of the coordinates of e^+ is not rational. This implie sthat P_1 terminates on $\mathbb{Q}^2$. □

This proves that even if A and v are rational, one can not guarantee the termination over the reals if the interpretation of the variables are restricted to rationals. It is clear that one cannot hope to produce any valid conjecture of this type if A and v have wild coefficients (transcendental for example).

However, when A and v have algebraic coefficients, thanks to Corollary 3.1, one can find a simple remedy. It is indeed enough to replace $\mathbb{Q}$ by a finite extension of the field $\mathbb{Q}$. Such an extension K is called a **number field**, and is known to be countable, indeed, it is a $\mathbb{Q}$-vector space of finite dimension (i.e. $K = \mathbb{Q}.k_1 \oplus \cdots \oplus \mathbb{Q}.k_l$ for some $l \geq 1$, and elements k_i of K).
It is moreover known that K is the fraction field of its **ring of integers** O_K, which is a free $\mathbb{Z}$-module of finite type, (in fact $O_K = \mathbb{Z}.o_1 \oplus \cdots \oplus \mathbb{Z}.o_l$ for the same $l \geq 1$, and the

elements o_i can be choosed equal to the k_i, for well chosen k_i's).
We say that a number field is **real** if it is a subfield of $\mathbb{R}$.

Theorem 6.1. *Let $A \in \mathcal{M}_n(\mathbb{R})$ and $v \neq 0 \in \mathbb{R}^n$, and suppose that their coefficients are actually in $\mathbb{Q}$ (or more generally in a **real** number field K). Then there is a well-determined **real** finite extension L of $\mathbb{Q}$ (or of K in the general case) contained in $\mathbb{R}$, such that the program $P(A, v)$ associated to A and v terminates, if and only if it terminates on the countable set L^n. We can choose L to be the extension $\mathbb{Q}(\lambda_1, \ldots, \lambda_t)$ of $\mathbb{Q}$ ($K(\lambda_1, \ldots, \lambda_t)$ in general) spanned by the positive eigenvalues $(\lambda_1, \ldots, \lambda_t)$ of A. It is actually enough to check the termination of the program on O_L^n.* □

Proof. We deal with the general case, the reader not familiar with field extensions can just replace K by $\mathbb{Q}$.
It is obvious that if the program terminates, it terminates on L^n for any subset L of $\mathbb{R}$. Now $\lambda_1, \ldots, \lambda_r$ be the positive eigenvalues of A. They ar all roots of the minimal (or characteristic) polynomial Q of A, which belongs to $K[X]$, they are thus all algebraic on K (hence on $\mathbb{Q}$, as K/Q is finite). Let $L = K(\lambda_1, \ldots, \lambda_r) \subset \mathbb{R}$. Suppose that the program P_1 does not terminate. Then there is $i \in \{1, \ldots, r\}$, such that $< E_{\lambda_i}, v > \neq 0$ according to Corollary 3.2. Let r be the integer ≥ 1 such that $Ker((A - \lambda_i I_n)^r) \not\subset v^\perp$, but $Ker((A - \lambda_i I_n)^{r-1}) \subset v^\perp$. We saw in the proof of Theorem **??**, that for any x in $Ker((A - \lambda_i I_n)^r) - Ker((A - \lambda_i I_n)^{r-1})$, such that $< v, x > 0$, the programm does not terminate. We fix such an x. Both spaces $Ker((A - \lambda_i I_n)^r)$ and $Ker((A - \lambda_i I_n)^{r-1})$ are defined by linear equations with coefficients in L, hence there is a basis of $Ker((A - \lambda_i I_n)^r)$ with coefficients in L^n, containing a basis of $Ker((A - \lambda_i I_n)^{r-1})$ with coefficients in L^n. It is easy to see, that this fact implies that $L^n \cap [Ker((A - \lambda_i I_n)^r) - Ker((A - \lambda_i I_n)^{r-1})]$ is dense in $Ker((A - \lambda_i I_n)^r) - Ker((A - \lambda_i I_n)^{r-1})$ (because L contains Q which is dense in R). Hence there is a sequence x_k in $L^n \cap [Ker((A - \lambda_i I_n)^r) - Ker((A - \lambda_i I_n)^{r-1})]$ wich tends to x, in particular $< v, x_k > 0$ for k large enough. The programm does thus not terminate for x_k for k such that $< v, x_k >> 0$. This shows that P_1 does not terminate on L^n. Now, the fact that P_1 doesn't terminate on O_L is a trivial consequence of the fact that any element of L is the quotient of two elements of O_l, in particular, if P_1 doesn't terminate on $x \in L^n$, take $a > 0$ in O_L, such that $ax \in O_L^n$, then the program does not terminate on ax. □

Let's see how Theorem 6.1 work on our previous example.

Example 6.1. *For the program associated to the matrix $A = \begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix}$, and the vector $v = (0, 1)^\top$, the field L is equal to $L = \mathbb{Q}(\lambda^+) = \mathbb{Q}(\sqrt{2}) = \{a + b\sqrt{2}, a \in \mathbb{Q}, b \in \mathbb{Q}\}$. It's ring of integers is equal $O_L = \mathbb{Z}(\lambda^+) = \mathbb{Z}(\sqrt{2}) = \{a + b\sqrt{2}, a \in \mathbb{Z}, b \in \mathbb{Z}\}$. Theorem 6.1 asserts that, as the programm $P(A, v)$ is non terminating, it is already non terminating on O_L^2. Indeed, Take x^+ as an intial value, then $x^+ = \begin{pmatrix} 1 \\ -1 + \sqrt{2} \end{pmatrix}$ belongs to O_L^2, and the we saw that $P(A, v)$ does not terminate on x^+.* □

7 Experimental Results

In Table 1 we list some experimental results. The column **Set-i** refers to a set of loops gerenated randomly. The column **#Loops** gives the number of loops treated (each set provides the analysis of 500 loops). The column **P** gives the class of the linear loop programs of the considered set. There are the three definied linear class defined in this article: $P^{\mathbb{H}}$, $P^{\mathbb{G}}$ and $P^{\mathbb{A}}$. We give the fields associated to the entries of the matrix considered in the column **Fi**. The ring $\mathbb{L}$ refers to the considered countable subset described in Section 6. The column **Dim** refers to the dimention of the initial systems (the number of variables). The column **#T** returns the number of programs found as terminant and the column **#NT** gives the number of loops programs found nontermiant. Finaly, column **CPU/s[T]** refers to the cpu time results while proving all the terminant loop programs and **CPU/s[N]** gives the cpu time taken to conclude for all the nonterminant programs. The column **CPU/s[total]** gives the cpu time results for concluding on the termination for the given set of 500 loops. We have implemeted our prototype using **Sage** [24] through interfaces written in python. By doing so, we were able to have access to several useful mathematical packages. As expected we can see that it produces more nonterminant programs (easier to write) and it take much more time to prove termination than to prove nontermination.

8 Conclusion

We present the *first necessary and sufficient condition* for the termination of linear programs. Infact, this NSC exhibits a complete decidability result for the class of linear programs on all initial values and provides us with a sound, complete and fast computational method for the termination analysis of such linear programs. The analysis of our associated alorithms are the evidences showing that our method operates in few and fast computational steps. The proposed computational method is of lower complexity than the mathematical foundations of previous methods. Section 6, and especially the example of this section, shows that an important notion is the locus of initial values for which a linear program terminates.

References

[1] Cousot, P., Cousot, R.: Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: Conf. Record of the 4th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Los Angeles, California, ACM Press, NY (1977) 238–252

[2] Manna, Z.: Mathematical Theory of Computation. McGrw-Hill (1974)

[3] Clarke, E.M., Grumberg, O., Peled, D. MIT Press, Cambridge, MA (2000)

[4] Queille, J.P., Sifakis, J.: Specification and verification of concurrent systems in cesar. In: Proceedings of the 5th Colloquium on International Symposium on Programming, London, UK, Springer-Verlag (1982) 337–351

Table 1: Experimental results on randomly generated linear loop programs

RandSet	#Loops	P	Fi	Dim	#T	#NT	CPU/s[T]	CPU/s[N]	CPU/s[total]
Set-1	500	$P^{\mathbb{H}}$	$\mathbb{Z}$	3	152	348	10.02	8.79	18,24
Set-2	500	$P^{\mathbb{H}}$	$\mathbb{Q}$	3	195	305	8.97	9.11	18.08
Set-5	500	$P^{\mathbb{G}}$	$\mathbb{Z}$	3	233	267	15.07	12, 78	27,85
Set-6	500	$P^{\mathbb{G}}$	$\mathbb{Q}$	3	223	277	12.49	10.42	22.91
Set-9	500	$P^{\mathbb{A}}$	$\mathbb{Z}$	3	246	254	12.52	11.59	24,11
Set-10	500	$P^{\mathbb{A}}$	$\mathbb{Q}$	3	222	278	13.30	10.35	23.66
Set-13	500	$P^{\mathbb{H}}$	$\mathbb{R}$	4	122	378	27.8	16.51	44.31
Set-14	500	$P^{\mathbb{H}}$	$\mathbb{Q}$	4	184	316	42,67	21.90	53.80
Set-17	500	$P^{\mathbb{G}}$	$\mathbb{R}$	4	145	355	31.91	18.05	49.97
Set-18	500	$P^{\mathbb{G}}$	$\mathbb{Q}$	4	171	329	41.16	22.37	63.54
Set-21	500	$P^{\mathbb{A}}$	$\mathbb{Z}$	4	185	315	43.03	24.22	67.25
Set-22	500	$P^{\mathbb{A}}$	$\mathbb{Q}$	4	176	324	40.36	19.95	60.32
Set-1	500	$P^{\mathbb{H}}$	$\mathbb{R}$	5	183	317	126.24	66.95	193.20
Set-1	500	$P^{\mathbb{H}}$	$\mathbb{Q}$	5	227	273	155.80	81.29	237.10
Set-21	500	$P^{\mathbb{G}}$	$\mathbb{Z}$	5	178	322	103.90	43.47	146.57
Set-22	500	$P^{\mathbb{G}}$	$\mathbb{Q}$	5	161	339	169.92	54.00	223.92
Set-23	500	$P^{\mathbb{A}}$	$\mathbb{R}$	5	199	299	171.92	66.75	238.68
Set-24	500	$P^{\mathbb{A}}$	$\mathbb{Q}$	5	209	201	174.91	70.32	254.24
Set-1	500	$P^{\mathbb{H}}$	$\mathbb{Z}$	6	141	359	236.0	70.19	306.20
Set-25	500	$P^{\mathbb{H}}$	$\mathbb{Q}$	6	173	327	387.80	105.69	493.50
Set-26	500	$P^{\mathbb{G}}$	$\mathbb{Z}$	6	192	308	342.70	101.89	444,59
Set-27	500	$P^{\mathbb{G}}$	$\mathbb{Q}$	6	188	312	352.40	165.41	517.81
Set-28	500	$P^{\mathbb{A}}$	$\mathbb{Z}$	6	231	270	402.71	174.56	577.28
Set-29	500	$P^{\mathbb{A}}$	$\mathbb{Q}$	6	184	316	385.00	190.94	575.94
Set-30	500	$P^{\mathbb{H}}$	$\mathbb{Q}$	7	171	329	851.18	194.21	1044.39
Set-31	500	$P^{\mathbb{G}}$	$\mathbb{Q}$	7	139	361	699.03	174.65	873.68
Set-32	500	$P^{\mathbb{A}}$	$\mathbb{Q}$	7	174	336	876.62	238.94	1115.56

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

[6] Cousot, P., Halbwachs, N.: Automatic discovery of linear restraints among variables of a program. In: Conference Record of the Fifth Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Tucson, Arizona, ACM Press, New York, NY (1978) 84–97

[7] Sipma, H.B., Uribe, T.E., Manna, Z.: Deductive model checking. Form. Methods Syst. Des. **15**(1) (July 1999) 49–74

[8] Colón, M., Sipma, H.: 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, London, UK, UK, Springer-Verlag (2001) 67–81

[9] Col'on, M.A., Sipma, H.B.: Practical methods for proving program termination. In: In CAV2002: Computer Aided Verification, volume 2404 of LNCS, Springer (2002) 442–454

[10] Bradley, A.R., Manna, Z., Sipma, H.B.: Linear ranking with reachability. In: In CAV, Springer (2005) 491–504

[11] Bradley, A.R., Manna, Z., Sipma, H.B.: Termination analysis of integer linear loops. In: In CONCUR, Springer-Verlag (2005) 488–502

[12] Dams, D., Gerth, R., Grumberg, O.: A heuristic for the automatic generation of ranking functions. In: Workshop on Advances in Verification. (2000) 1–8

[13] Podelski, A., Rybalchenko, A.: A complete method for the synthesis of linear ranking functions. In: VMCAI. (2004) 239–251

[14] Braverman, M.: Termination of integer linear programs. In: In Proc. CAV06, LNCS 4144, Springer (2006) 372–385

[15] Bradley, A.R., Manna, Z., Sipma, H.B.: Termination of polynomial programs. In: In VMCAI'2005: Verification, Model Checking, and Abstract Interpretation, volume 3385 of LNCS, Springer (2005) 113–129

[16] Chen, H.Y., Flur, S., Mukhopadhyay, S.: Termination proofs for linear simple loops. In: Proceedings of the 19th international conference on Static Analysis. SAS'12, Berlin, Heidelberg, Springer-Verlag (2012) 422–438

[17] Cousot, P.: Proving program invariance and termination by parametric abstraction, lagrangian relaxation and semidefinite programming. In: Sixth International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI'05), Paris, France, LNCS 3385, Springer, Berlin (January 17–19 2005) 1–24

[18] Cousot, P., Cousot, R.: An abstract interpretation framework for termination. SIGPLAN Not. **47**(1) (January 2012) 245–258

[19] Cook, B., Podelski, A., Rybalchenko, A.: Termination proofs for systems code. SIGPLAN Not. **41**(6) (June 2006) 415–426

[20] Ben-Amram, A.M., Genaim, S., Masud, A.N.: On the termination of integer loops. In: VMCAI. (2012) 72–87

[21] Ben-Amram, A.M., Genaim, S.: 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, New York, NY, USA, ACM (2013) 51–62

[22] Tiwari, A.: Termination of linear programs. In Alur, R., Peled, D., eds.: Computer Aided Verification, 16th International Conference, CAV 2004, Boston, MA, USA. Volume 3114 of Lecture Notes in Computer Science., Springer (2004) 70–82

[23] Rebiha, R., Matringe, N., Moura, A.V.: Necessary and sufficient condition for termination of linear programs. Technical Report TR-IC-13-07, Institute of Computing, University of Campinas (January 2013)

[24] Stein, W., Joyner, D.: SAGE: System for Algebra and Geometry Experimentation. ACM SIGSAM Bulletin, volume 39, number 2, pages 61–64 (2005)