**School voor Informatietechnologie**
Kennistechnologie, Informatica, Wiskunde, ICT

# Upper and Lower Complexity Bounds for Some Problems in Elementary Geometry

Proefschrift voorgelegd tot het behalen van de graad van
Doctor in de Wetenschappen, richting Informatica
te verdedigen door

Rafael Grimson

Promotors: Prof. dr. Bart Kuijpers and Prof. dr. Joos Heintz

2nd February, 2010

# Preface

This thesis would not be if not for the support of a number of people. First of all, my gratitude goes to my supervisors Bart Kuijpers and Joos Heintz, for their guidance and support.

I would like to thank Lucas Galfasó, Ariel Molinuevo, Alberto Carrassi and Walied Othman for many stimulating discussions on different subjects related to this thesis.

I am also grateful to the Theoretical Computer Science Group of Hasselt University for the quiet and inspiring atmosphere they created. Part of this work was realized during some visits to Joos Heintz at the University of Buenos Aires. I'm grateful to both universities for their hospitality.

This research has been partially funded by the tUL-impulse programme and by the Research Foundation Flanders (FWO-Vlaanderen, Research Project G.0344.05).

Finally, my gratitude goes to my parents, brothers and sister for their faith and love.

Rafael Grimson,
Brussels, November 2009

# Contents

**Introduction** **1**

**1 Lower Bounds for Linear Programming** **5**

1.1 Introduction . . . 6

1.1.1 Definitions and Summary of Results . . . 6

1.1.2 Related Work . . . 7

1.2 The Parametric Feasibility Problem . . . 8

1.2.1 A Quantifier-Elimination Problem . . . 8

1.3 Limiting Hypersurfaces . . . 9

1.4 Polyhedra and Elimination . . . 10

1.4.1 Preliminaries on Polyhedra and Polytopes . . . 10

1.4.2 Polyhedra in $\mathbf{R}^n$ defined by $n+1$ Inequalities . . . 11

1.4.3 The Geometry of the Parameters defining Polyhedra . . . 13

1.5 Counting the Limiting Hypersurfaces . . . 14

1.6 Complexity Lower Bounds . . . 17

1.6.1 Dense Representation . . . 18

1.6.2 Sparse Representation . . . 19

1.6.3 Algebraic Computation Trees . . . 20

**2 The Sign Condition Problem: Upper and Lower Bounds** **23**

2.1 Introduction . . . 23

2.1.1 The Sign Condition and Point Location Problems for a Family of Polynomials . . . 24

2.1.2 Basic Observations . . . 24

2.1.3 Related Work . . . 25

2.2 Computational Models and Representations of Polynomials . . . 27

2.2.1 Representation of the Polynomials . . . 28

2.3 The Dialytic Method to solve the Sign Condition Problem . . . 29

2.4 Lower Bounds for the Sign Condition Problem . . . 32

2.4.1 The Algebraic Model . . . 33

2.4.2 Restricted Models . . . . . . . . . . . . . . . . . . . . . . 38

**3 The Point Location Problem for a Family of Polynomials 43**

3.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . 43
3.2 The dialytic method for Point Location . . . . . . . . . . . . . 45
3.3 Point Location using Cylindrical Algebraic Decomposition . . . 46
3.3.1 An Elimination Step and its Data Structures . . . . . . 46
3.3.2 CAD for Point Location . . . . . . . . . . . . . . . . . . 51
3.3.3 Construction of the Database . . . . . . . . . . . . . . . 53
3.4 Point Location for Polynomials with Integer Coefficients . . . . 54
3.4.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . 54
3.4.2 The Arithmetic Relation between Distance and Height . 55
3.4.3 The Mathematical Insight . . . . . . . . . . . . . . . . . 57
3.4.4 The Sign Condition Algorithm . . . . . . . . . . . . . . 59
3.4.5 The Point Location Algorithm . . . . . . . . . . . . . . 62
3.5 Local Ray Shooting . . . . . . . . . . . . . . . . . . . . . . . . 64
3.5.1 The algorithm . . . . . . . . . . . . . . . . . . . . . . . 64
3.6 Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 67

**4 Quantifier Elimination using Intrinsic Data Structures 69**

4.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . 70
4.2 Theoretical Foundations . . . . . . . . . . . . . . . . . . . . . 70
4.2.1 Constructible Sets . . . . . . . . . . . . . . . . . . . . . 70
4.2.2 The Language $\mathcal{L}$ . . . . . . . . . . . . . . . . . . . . . 71
4.2.3 A Computational Model for Intrinsic Data . . . . . . . . 72
4.2.4 Intrinsic Descriptions . . . . . . . . . . . . . . . . . . . 74
4.2.5 Definition of Filtrations . . . . . . . . . . . . . . . . . . 75
4.2.6 Filtrations and Locally Closed Sets . . . . . . . . . . . . 76
4.2.7 Canonic Descriptions . . . . . . . . . . . . . . . . . . . 77
4.2.8 Particles . . . . . . . . . . . . . . . . . . . . . . . . . . 80
4.3 The Algorithms . . . . . . . . . . . . . . . . . . . . . . . . . . 81
4.3.1 Conversion Algorithms . . . . . . . . . . . . . . . . . . 82
4.3.2 Quantifier-Elimination Algorithm . . . . . . . . . . . . . 84
4.3.3 Canonic Form Conversion Algorithm . . . . . . . . . . . 88
4.3.4 Our algorithm in other models . . . . . . . . . . . . . . 92
4.4 Lower Bounds . . . . . . . . . . . . . . . . . . . . . . . . . . . 93
4.4.1 A Lower Bounds for the Conversion to Canonic Form . . 93
4.4.2 A Doubly-Exponential Lower Bound for QE . . . . . . . 94

**5 Quantifier Elimination for Elementary Geometry 97**
5.1 Introduction . . . 97
5.1.1 Origins of the problem . . . 97
5.1.2 Outline and Summary . . . 99
5.2 Preliminaries and definitions . . . 100
5.2.1 Semi-algebraic and geometric relations . . . 100
5.2.2 Affine and similarity transformations of the plane . . . 101
5.2.3 Affine-invariant and similarity-invariant relations . . . 102
5.2.4 The theories $\mathcal{R}$, $\mathcal{E}$, $\mathcal{A}$ and their expressive power . . . 103
5.2.5 Translations . . . 105
5.2.6 Quantifier elimination for $\mathcal{R}$, $\mathcal{E}$ and $\mathcal{A}$ . . . 106
5.3 The new languages . . . 108
5.3.1 The two basic segment-arithmetic functions . . . 108
5.3.2 The affine projection function . . . 110
5.3.3 The two basic metric functions . . . 110
5.3.4 The language $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ and the theory $\mathcal{A}'$ . . . 111
5.3.5 The language $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$ and the theory $\mathcal{E}'$ 113
5.4 The translation $\mathcal{S}$ . . . 114
5.4.1 A translation given an affine coordinate system for **E** . . 114
5.4.2 Finding a basis . . . 116
5.5 The translation $\mathcal{T}$ . . . 120
5.6 Quantifier elimination for the theories $\mathcal{A}'$ and $\mathcal{E}'$ . . . 123
5.7 Final Remarks . . . 124
5.7.1 Discussion on the primitive notions . . . 124
5.7.2 Axiom systems for the new languages . . . 125

**A Algebraic computational models 127**
A.1 Algebraic Decision Trees over the Reals . . . 128
A.2 Algebraic Computation Trees over the Reals . . . 128
A.2.1 Syntax of computation trees . . . 129
A.2.2 Semantics of computation trees . . . 130
A.2.3 Pragmatics of computation trees . . . 132
A.3 Boolean-Arithmetic Circuits over the Reals . . . 133
A.4 The Bit Models . . . 134
A.4.1 Bit Representation of Algebraic Numbers . . . 134
A.5 Representation of Polynomials . . . 135
A.5.1 Non-Scalar vs. Total Complexity . . . 136

**Summary 147**

**Samenvatting 149**

# Introduction

This thesis is mainly dedicated to the study of upper and lower algebraic-complexity bounds of some problems in the context of semi-algebraic geometry. The main computational models considered are described in Appendix A.

In this Introduction, we present an overview of the contents of the thesis and we finish with some historical notes concerning the development of the algorithmic aspects of the elimination of quantifiers in the elementary theory of the reals.

## Overview of the Work

We first study the linear programming feasibility problem. This problem can be stated as follows: given positive integers $m > n$, a matrix $H \in \mathbf{R}^{m \times n}$ and a vector $h \in \mathbf{R}^m$ decide whether there exists a column vector $x \in \mathbf{R}^n$ such that $H \cdot x \leq h$, where the $\leq$ is interpreted componentwise.

The simplex method for linear programming solves this problem. It is well known that this method is exponentially slow in the worst case. On the other hand, the ellipsoid method (see [Kha79]) solves the feasibility problem over the rational numbers in polynomial time in the bit model, but is not strongly polynomial (*i.e.*, it is not based on algebraic operations).

In fact, the existence of a polynomial-time algorithm, in the algebraic computational model, solving the linear programming feasibility problem is an open problem. It has been proposed by Smale as one of the great problems for the present century (see Problem 9 in [Sma00]).

In Chapter 1, we analyze the algebraic complexity of the linear programming feasibility problem over the reals and prove non-trivial lower bounds for this problem. In particular, a lower bound for algebraic computation trees based on the notion of *limiting hypersurface* is presented. However, our lower bounds method does not provide an exponential lower bound in the considered model.

We may ask then whether a non-uniform polynomial time algorithm can be constructed to solve this problem. The method of *limiting hypersurfaces* introduced in Chapter 1 suggests that the feasibility of a given matrix may be determined by the sign conditions satisfied by all its minors.

This is why, in Chapter 2, we study the sign condition problem for any given a family of polynomials; this problem consists in determining the sign condition satisfied by a fixed family of polynomials at a query point, performing as little arithmetic operations as possible.

After defining precisely the sign condition and the point location problems, we introduce a method called *the dialytic method* to solve the first problem efficiently. This method involves a *linearization* of the original polynomials and provides the best known algorithm to solve the sign condition problem. Finally, using a technique that resembles that of Chapter 1, we prove a lower bounds showing that the dialytic method is almost optimal.

However, if we would apply the dialytic method to the linear programming feasibility problem, we would obtain an exponential-time algorithm. So, the question of the algebraic complexity of this problem, remains open.

At this point, we continue our investigations on a refined version of the sign condition problem: the point location problem. In Chapter 3, we discuss different data structures that can be used to solve the point location problem for a given family of polynomials. This problem asks to determine, not only the sign condition satisfied by a family of polynomials at a query point, but also the connected component of the realization of this sign condition containing the query point. After showing how to adapt the dialytic method to this problem, we introduce, in Section 3.3, a method based on an Adapted Cylindrical Algebraic Decomposition of the space that solves the point location problem for any given family. In Section 3.4, we discuss the case of polynomials with integer coefficients given in dense bit representation introducing a method that, based on diophantine geometry, solves the point location problem for generic families of polynomials. At the end of the chapter, we include a brief discussion of the local ray shooting problem.

In Chapter 4, we continue the line of the investigations carried on in Chapter 1, where we have shown that the *limiting hypersurfaces* of a set are *intrinsic* to it. First, we introduce the notion of *intrinsic description* of a linearly-constructible set and study the complexity of quantifier-elimination methods in a computational model where the output is required to be an intrinsic description of the underlying set. We present a quantifier-elimination algorithm in this model. It turns out that our elimination algorithm has a doubly-exponential-time complexity in the worst case, when the complexity is measured in terms of syntactic parameters (number of polynomials and of quantifier alternations, dimension of the ambient space). We show that in our computational model, our algorithm is optimal, *i.e.*, we prove a doubly-exponential lower bound in the number of quantifier alternations of the input formula.

Remarkably, we obtain simply-exponential complexity bounds on intrinsic geometric parameters of the input problem. Thus, our algorithm distinguishes between well-posed and ill-posed problems and can be inscribed in the new generation of algorithms which take also into account intrinsic, semantic invariants of the input in order to measure the complexity of the procedure.

Chapter 5 is the only chapter not related to complexity theory. Following the tradition of mathematical logic, we introduce new first-order languages for the elementary $n$-dimensional geometry and elementary $n$-dimensional affine geometry ($n \geq 2$), based on extending the traditional languages $\mathsf{FO}(\beta, \equiv)$ and $\mathsf{FO}(\beta)$, respectively, with new function symbols. Here, $\beta$ stands for the betweenness relation and $\equiv$ for the congruence relation. We show that the associated theories admit effective quantifier elimination.

## Historical Notes on Quantifier Elimination

In the tradition of logic, quantifier-elimination has almost a century of history, and has been used mainly as a tool to prove the decidability of a theory, *i.e.*, as a decision procedure. Recently, in the context of Constraint Databases [KLP00], it has been proposed as a query evaluation method (see Section 3.1 for a discussion of this use).

The first quantifier-elimination method for the theory of real closed fields was presented by Tarski in the 1930s, based on previous work by Sturm and Sylvester. The result was published in [Tar51], where Tarski presents an effective decision procedure for the elementary theory of the reals based on the elimination of quantifiers.

The complexity study of such quantifier-elimination procedures started in the 1970s with the design of doubly-exponential elimination algorithms by Collins (see [Col75]) and, independently, by Monk and Solovay (see [Wüt76], inspired by [Mon74]).

Modern quantifier-elimination procedures work in doubly-exponential time in the number of quantifier alternations of the input formula (see the seminal paper of Grigoriev and Vorobjov [GV88] and [Can88, Ren88, HSR89] for the existential theory and [HRS90b, Ren92a, Ren92b, Ren92c, HRS93, BPR96] for the general case; a complete account can be found in [BPR06]).

On the other hand, Davenport and Heintz [DH88] gave a doubly-exponential lower bound for the general quantifier-elimination problem over the reals, if polynomials are encoded in dense form (this result is also implicitly contained in [Wei88]; both papers are motivated by the paradigm of [FR74]). Davenport and Brown presented, in [BD07], a simplified proof of this doubly-exponential lower bound that works for both, dense and sparse codification of polynomials. Thus, in order of magnitude, upper and lower complexity bounds meet for *classic* data-structures (*i.e.*, when polynomials are represented in dense or sparse form).

In [BOKR84], Ben-Or, Kozen and Reif attempted to design a singly- exponential parallel-complexity decision procedure for the elementary theory of the reals. Nevertheless, the authors failed to observe that the sequential complexity of their algorithm becomes uncontrolled. This drawback was corrected in [FGM90] and the outcome was a quantifier-elimination procedure in single-exponential parallel time using a doubly-exponential number of processors. Moreover, the optimality of this procedure was shown. In [MP93], this last lower-bound result was extended to a slightly more general computational model.

The reason of the inefficiency of general purpose algorithms for quantifier elimination is unknown. There is still the hope that using alternative data structures this high intractability will be overcome. The algorithms can only deal with syntactic descriptions of the geometric objects involved and the examples show that the impact of these description on the complexity is enormous.

It is a major open question in complexity theory whether, using boolean-arithmetic circuits to codify first-order formulas, a polynomial-time algorithm can be designed for the elimination of a single quantifier block. In fact, this question is equivalent to the $\mathsf{P}_{\mathbf{R}} = \mathsf{NP}_{\mathbf{R}}$ problem (see [Koi00]). Up to now, no general procedure has been designed able to improve substantially the worst-case complexity of well-known algorithms based on classic encodings of polynomials.

Complexity improvements based on alternative data-structures, such as boolean-arithmetic circuits (also called arithmetic networks, see [vzG86a]) for constructible sets, were only achieved for particular instances of elimination problems and only few is known about lower complexity bounds for this kind of encodings. Remarkably, it is proven in [HMPW98, GH01] and [CGH$^+$03] that any geometric elimination algorithm, using circuit encoding of polynomials and being *geometrically robust*—a property owned by all known symbolic methods—requires exponential time on infinitely many inputs. We shall continue the discussion on the relation of these articles and the present work in Section 4.4.

# 1

# Some Lower Bounds for the Complexity of the Linear Programming Feasibility Problem over the Reals

**Abstract.** In this chapter, we analyze the algebraic complexity of the linear programming feasibility problem over the reals and prove non-trivial lower bounds for this problem. The linear programming feasibility problem can be stated as follows: given positive integers $m > n$, a matrix $H \in \mathbf{R}^{m\times n}$ and a vector $h \in \mathbf{R}^m$ decide whether there exists a column vector $x \in \mathbf{R}^n$ such that $H \cdot x \leq h$. For the case of polyhedra defined by $2n$ halfspaces in $\mathbf{R}^n$, we prove that the set $\mathcal{I}^{(2n,n)}$ of parameters describing non-empty polyhedra, has an exponential number of *limiting hypersurfaces*. From this geometric result we obtain, as a corollary, the existence of a constant $c > 1$ such that, if dense or sparse representation is used to encode polynomials, the length of any quantifier-free formula expressing the set $\mathcal{I}^{(2n,n)}$ is bounded from below by $\Omega(c^n)$. Other related complexity results are stated; in particular, a lower bound for algebraic computation trees based on the notion of *limiting hypersurface* is presented.

## 1.1 Introduction

### 1.1.1 Definitions and Summary of Results

The *linear programming feasibility problem* over the reals can be stated as follows: given two positive integers $m$ and $n$, a matrix $H \in \mathbf{R}^{m\times n}$ and a vector $h \in \mathbf{R}^m$ decide whether there exists a column vector $x \in \mathbf{R}^n$ such that $H \cdot x \leq h$, where the $\leq$ is interpreted componentwise.

In this chapter, we analyze the complexity of this problem for different data structures.

We study the geometry of the set

$$\mathcal{I}^{(m,n)} = \{(H,h) \in \mathbf{R}^{m\times n} \times \mathbf{R}^m \mid \exists x \in \mathbf{R}^n \, (H \cdot x \leq h)\}$$

and show that it has at least $\binom{m}{n+1}$ different *limiting hypersurfaces* (Corollary 1.5.5). Geometrically, to an existential quantifier block corresponds a projection. Hence, this set is the projection over $\mathbf{R}^{m\times n} \times \mathbf{R}^m$ of $\{(H,h,x) \in \mathbf{R}^{m\times n} \times \mathbf{R}^m \times \mathbf{R}^n \mid H \cdot x \leq h\}$, that has only $m$ *limiting hypersurfaces* (given by the $m$ equations in the system $H \cdot x = h$).

These hypersurfaces turn out to be intrinsic, in the sense that any description of a set must involve the descriptions of its limiting hypersurfaces.

From these geometric results, we derive exponential lower bounds for the size of any quantifier-free formula in the first-order language of the reals, expressing the set $\mathcal{I}^{(2n,n)}$, if polynomials are codified using dense or sparse representation (see Corollaries 1.6.2 and 1.6.6). Also, we obtain a linear lower bound for the depth of any computation tree (see Appendix A.2 for a definition) solving the linear programming feasibility problem (see Corollary 1.6.10).

We further obtain (see Corollaries 1.6.3 and 1.6.7) a sub-exponential lower bound for the complexity of any algorithm for the elimination of a single quantifier block of quantifiers in the elementary theory of the reals, if polynomials are codified using dense or sparse representation . Although it is not hard to find examples (see, *e.g.*, [CGH+03]) showing that the complexity swell occurring in the elimination of a single block of quantifiers may be exponential for these data structures, all such known examples are highly artificial. We prove the first sub-exponential lower bound for a completely *natural* problem, namely the linear programming feasibility problem.

This chapter is organized as follows. In Section 1.2, we state the feasibility problem as a quantifier-elimination problem and define the set $\mathcal{I}^{(m,n)} \subseteq \mathbf{R}^{m\times n} \times \mathbf{R}^m$. In Section 1.3, we define the notions of *limiting hypersurface* of a semi-algebraic set and of a polynomial *intervening* in a formula. Afterwards, we prove Proposition 1.3.2, stating that if $Z$ is a limiting hypersurface for a set $W$ and $Q$ is an irreducible polynomial defining $Z$, then $Q$ intervenes in any quantifier-free description of $W$. The fourth section is devoted to the study of the geometry of the set $\mathcal{I}^{(m,n)}$. Since, for fixed $H$ and $h$, the set $\{x \in \mathbf{R}^n \mid H \cdot x \leq h\}$ is a polyhedron, the section begins with some preliminaries on polyhedra. Finally, in Section 1.5, we prove that the set $\mathcal{I}^{(2n,n)}$ has at least $\binom{2n}{n+1}$ limiting hypersurfaces. In Section 1.6, we use this geometrical fact to prove complexity lower bounds for the different data structures considered.

### 1.1.2 Related Work

**Quantifier Elimination.** See the Introduction to this thesis for historical notes concerning the algorithmic aspects of the elimination of quantifiers in the elementary theory of the reals.

**Limiting Hypersurfaces.** As explained before, our proofs are based on the fact that the limiting hypersurfaces of a set are intrinsic. Lazard used a similar technique in [Laz88] to prove the optimality of solutions to two classical quantifier-elimination problems.

Combining methods from abstract real algebraic geometry and complexity theory, Lickteig [Lic90, Lic96] developed a technique to prove lower complexity bounds in the algebraic-computation-tree model. This technique is also related to ours, since it allows the use of limiting hypersurfaces as a complexity source.

**Linear Programming.** The Dantzig simplex method for linear programming is known to be exponentially slow in the worst case. On the other hand, the ellipsoid algorithm solves the feasibility problem over the rational numbers in polynomial time in the bit model (see [Kha79]), but is not strongly polynomial. In fact, the existence of a polynomial-time algorithm, in the BSS computational model, solving the linear programming feasibility problem is an open problem. It has been proposed by Smale as one of the great problems for the present century (see Problem 9 in [Sma00]). It follows from our results that, for any boolean-arithmetic circuit accepting the set $\mathcal{I}^{(m,n)}$, a multiple of the polynomial describing each limiting hypersurface of $\mathcal{I}^{(m,n)}$ will be evaluated in the execution of the circuit, for some input. This result implies a lower bound for the linear programming feasibility problem that is far from being strong enough to give a negative answer to Smale's ninth problem. It is our belief that if a proof of a negative answer is to be found, the notion of uniformity will play a central role in it.

## 1.2 The Parametric Feasibility Problem

The feasibility problem for linear optimization over the reals can be stated as follows:

*Given a matrix $H \in \mathbf{R}^{m\times n}$ and a column vector $h \in \mathbf{R}^m$, decide whether there exists $x \in \mathbf{R}^n$ such that $H \cdot x \leq h$*, where the $\leq$ is interpreted componentwise.

### 1.2.1 A Quantifier-Elimination Problem

The above decision problem can be reformulated as a quantifier-elimination problem. Let us first fix the notation. For each $n, m \in \mathbf{N}$, $m \geq n+1$, we consider $x_1, \dots, x_n$, $t_1^{(1)}, \dots, t_n^{(1)}, \dots, t_1^{(m)}, \dots, t_n^{(m)}$, and $b^{(1)}, \dots, b^{(m)}$ to be indeterminates over $\mathbf{R}$. We call $x_1, \dots, x_n$ the *variables* and the remaining $m \cdot n + m$ indeterminates, the *parameters* of the problem. Furthermore, we shall use the shorthand notations $x := (x_1, \dots, x_n)$ and

$$T := \begin{pmatrix} t_1^{(1)} & \dots & t_n^{(1)} & b^{(1)} \\ \vdots & \ddots & \vdots & \vdots \\ t_1^{(m)} & \dots & t_n^{(m)} & b^{(m)} \end{pmatrix}.$$

For the sake of readability, we shall not use different symbols for the indeterminates and their realizations as elements of $\mathbf{R}$. The distinction will be clear from the context.

We define the formulas

$$\sigma_i^n(x,T) := t_1^{(i)} \cdot x_1 + \dots + t_n^{(i)} \cdot x_n - b^{(i)} \leq 0, \; (i = 1, \dots, m),$$

$$\phi^{(m,n)}(T) := \exists x \, \sigma_1^n(x,T) \wedge \dots \wedge \sigma_m^n(x,T), \tag{1.2.1}$$

and call $\mathcal{I}^{(m,n)}$ the realization of $\phi^{(m,n)}$ in the parameter space. We observe that $\mathcal{I}^{(m,n)} \subseteq \mathbf{R}^{m\times n} \times \mathbf{R}^m$ is the set of parameters defining $m$ half-spaces in $\mathbf{R}^n$ with non-empty intersection. In other words, the linear programming feasibility problem in $\mathbf{R}^n$ with $m$ constraints, is the membership problem for the set $\mathcal{I}^{(m,n)}$.

Finding quantifier-free formulas, $\psi^{(m,n)}$, expressing the sets $\mathcal{I}^{(m,n)}$ is a way to solve the parametric feasibility problem. We prove that there do not exist quantifier-free formulas $\psi^{(m,n)}$, expressing the sets $\mathcal{I}^{(m,n)}$, of length bounded by a polynomial function in $m$ and $n$ if classic data structures (*i.e.*, dense or sparse encoding) are used to represent polynomials.

For the sake of clarity, we shall write

$$T_H := \begin{pmatrix} t_1^{(1)} & \dots & t_n^{(1)} \\ \vdots & \ddots & \vdots \\ t_1^{(m)} & \dots & t_n^{(m)} \end{pmatrix}, \quad T_h := \begin{pmatrix} b^{(1)} \\ \vdots \\ b^{(m)} \end{pmatrix},$$

and use the augmented matrix notation $(T_H|T_h) = T$.

Let $i \in \mathbf{N}$, $1 \leq i \leq n$. We shall call

$$t_1^{(i)} \cdot x_1 + \dots + t_n^{(i)} \cdot x_n - b^{(i)} = 0,$$

$$t_1^{(i)} \cdot x_1 + \dots + t_n^{(i)} \cdot x_n - b^{(i)} \leq 0, \text{ and}$$

$$t_1^{(i)} \cdot x_1 + \dots + t_n^{(i)} \cdot x_n - b^{(i)} < 0$$

the *equality*, the *inequality*, and the *strict inequality associated with the $i^{th}$ row of $T$*, respectively.

## 1.3 Limiting Hypersurfaces

Let $W \subseteq \mathbf{R}^k$ be a semi-algebraic set. We give the definition of *limiting hypersurface* of $W$ and prove that a description of each of these hypersurfaces must intervene in any quantifier-free description of $W$. In this sense, we say that limiting hypersurfaces of a set are *intrinsic*.

We refer the reader to [BCR98] for notions and notations from real algebraic geometry, *e.g.*, the notions of zeros of an ideal, of semi-algebraic set, of dimension of a set and of non-singular point.

We denote by $\partial W$ the set of points in the border of $W$ (neither interior to $W$ nor to the complement). We call $Z \subseteq \mathbf{R}^k$ an *irreducible hypersurface* if $dim(Z) = k - 1$ and there exists an irreducible polynomial $P \in \mathbf{R}[x_1, \dots, x_k]$ such that $Z = \{(x_1, \dots, x_k) \in \mathbf{R}^k \mid P(x_1, \dots, x_k) = 0\}$.

**Definition 1.3.1.** Let $Z$ be an irreducible hypersurface in $\mathbf{R}^k$. We call $Z$ a *limiting hypersurface* of $W$ if its intersection with $\partial W$ has dimension $k - 1$.

We consider first-order formulas built from atomic formulas of the form $P = 0$, $P \leq 0$, where $P \in \mathbf{R}[x_1, \dots, x_k]$ is a polynomial with real coefficients. Let $\psi$ be a first-order formula and $P \in \mathbf{R}[x_1, \dots, x_k]$. If $\psi$ contains an atomic subformula of the form $P = 0$ or $P \leq 0$, we say that $P$ *appears* in $\psi$. If a non-zero polynomial $P$ appears in $\psi$ and $Q \in \mathbf{R}[x_1, \dots, x_k]$ is non-constant and divides $P$, then we say that $Q$ *intervenes* in $\psi$.

**Proposition 1.3.2.** Suppose that $W \subseteq \mathbf{R}^k$ is a semi-algebraic set described by a quantifier-free formula $\psi$. If $Z_Q$ is a limiting hypersurface for $W$ and $Q$ is an irreducible polynomial describing $Z_Q$, then $Q$ intervenes in $\psi$.

*Proof.* Let us denote by $P_1, \ldots, P_s$ the polynomials appearing in $\psi$ and suppose, without loss of generality, that none of them is the zero polynomial. We define $U := Z_Q \cap \partial W$ and we recall that, by hypothesis, it is a semi-algebraic subset of $Z_Q$ of dimension $k-1$.

First, we remark that since $dim(Z_Q) = k-1$ and $Q$ is irreducible, a particular form of the real Nullstellensatz for principal ideals (see Theorem 4.5.1 in [BCR98]) implies that a polynomial $P \in \mathbf{R}[x_1, \ldots, x_k]$ vanishes on $Z_Q = \mathcal{Z}(Q)$ if and only if $Q$ divides $P$. Thus, to complete the proof, it remains to show that at least one $P_j$ $(1 \le j \le s)$ vanishes on $Z_Q$.

To prove this, we consider, for any $u \in U$, the sign conditions $C(u) \in \{-1, 0, 1\}^s$ satisfied by the polynomials $P_1, \ldots, P_s$ in this point. It is clear that the truth value of the formula $\psi$ in a point $u$ depends only on $C(u)$, since the truth value of atomic formulas depend only on these sign conditions.

These sign conditions partition the set $U$ is a finite number of disjoint semi-algebraic components, $U_1, \ldots, U_t$, namely the non-empty supports in $U$ of each possible sign condition. By Proposition 2.8.5 in [BCR98], one of these sets, say $U_i$, must have the same dimension as $U$, *i.e.*, $dim(U_i) = k-1$.

Now, since the polynomials $P_1, \ldots, P_s$ have constant signs over $U_i$, it follows that $U_i \subseteq W$ or $U_i \subseteq W^c$. Let us suppose, with out loss of generality, $U_i \subseteq W$.

We claim that one of the polynomials $P_1, \ldots, P_s$ vanishes in $U_i$. Let $u \in U_i$; if none of the polynomials is zero in $u$ then there exists and open neighborhood in $\mathbf{R}^k$ of this point with the same sign conditions implying that $u$ is an interior point of $W$, contradicting $u \in \partial W$. Hence, there exists a positive integer $j \le s$ such that $P_j$ is vanishes on $U_i$. Now, since $U_i \subseteq Z_Q$, $Z_Q$ is irreducible and both set have the same dimension, we conclude that the Zariski closure of $U_i$ equals $Z_Q$. Hence, $P_j$ vanishes on the whole $Z_Q$. Thus, $Q$ intervenes in $\psi$. $\square$

## 1.4 Polyhedra and Elimination

### 1.4.1 Preliminaries on Polyhedra and Polytopes

In this subsection, we recall the notions of *polyhedron* and *polytope* and prove some basic properties. We use the notation from [Pad99] and refer there for the proofs of some known results.

**Definition 1.4.1.** A set $P \subseteq \mathbf{R}^n$ is a *polyhedron* if and only if there exists $m \in \mathbf{N}$, an $m \times n$ matrix $H$ and a vector $h$ of $m$ real numbers such that $P = \{x \in \mathbf{R}^n \mid H \cdot x \le h\}$. The system of inequalities $H \cdot x \le h$ is a linear description of $P$. A polyhedron $P$ is *fully dimensional* if $dim(P) = n$. If the polyhedron $P$ is bounded we call it a *polytope* in order to distinguish it from an unbounded polyhedron.

In other words, a polyhedron is the intersection of finitely many halfspaces in $\mathbf{R}^n$. We shall write $P(H, h)$ to denote the polyhedron defined by $H$ and $h$, or simply $P(M)$ when $M = (H|h)$. We remark that polyhedra are convex sets.

**Definition 1.4.2.** Let $P \subseteq \mathbf{R}^n$ be any set. A point $x \in P$ is an *extreme point* of $P$ if and only if for any points $x^1, x^2 \in P$ and any $\mu \in \mathbf{R}$ with $0 < \mu < 1$ such that $x = \mu x^1 + (1 - \mu)x^2$, it follows that $x = x^1 = x^2$.

So, $x$ is an extreme point of a subset $P$ of $\mathbf{R}^n$ if its representation as a convex combination by elements of $P$ is unique, *i.e.*, the trivial one involving only $x$. Extreme points of polyhedra are also called *vertices*.

The following lemma (Proposition 7.2(b) in [Pad99]) shows how to calculate the vertices of a polyhedron.

**Lemma 1.4.3.** The point $x^0$ is a vertex of the polyhedron $P(H, h)$ if and only if $H \cdot x^0 \leq h$ and $H_1 \cdot x^0 = h_1$ for some $n \times (n+1)$ submatrix $(H_1|h_1)$ of $(H|h)$ with $rank(H_1) = n$.

Let $P$ be a polyhedron and let $S = \{x \mid x \text{ is a vertex of } P\}$ be its set of vertices. From the convexity of $P$ it follows that the convex hull of $S$ is contained in $P$, *i.e.*, $conv(S) \subseteq P$. The next lemma shows that the equality holds if and only if $P$ is a polytope. We refer the reader to Proposition 7.3(b) in [Pad99] for a proof.

**Lemma 1.4.4.** Let $S$ be the set of vertices of a polyhedron $P$. Then, $conv(S) = P$ if and only if $P$ is a polytope.

The Platonic solids are examples of polytopes in $\mathbf{R}^3$. We remark that a polytope might be not fully dimensional, as even the empty set is considered to be a polytope.

### 1.4.2 Polyhedra in $\mathbf{R}^n$ defined by $n + 1$ Inequalities

In this paragraph, we analyze the geometry of polyhedra in $\mathbf{R}^n$ defined by $n+1$ inequalities. We remark that these systems define, among other polyhedra, the *$n$-simplices*.

Consider $m \geq n+1$ and $T$ as in Section 1.2. We call, for the remaining of this chapter,

$$A(T) := \begin{pmatrix} t_1^{(1)} & \dots & t_n^{(1)} \\ \vdots & \ddots & \vdots \\ t_1^{(n)} & \dots & t_n^{(n)} \\ t_1^{(n+1)} & \dots & t_n^{(n+1)} \end{pmatrix}, \ b(T) := \begin{pmatrix} b^{(1)} \\ \vdots \\ b^{(n)} \\ b^{(n+1)} \end{pmatrix}$$

and $D(T)$ the determinant of the submatrix $(A(T)|b(T))$ of $T$. When the dependence on $T$ is clear from the context, we write simply $A, b$ and $D$ for $A(T), b(T)$ and $D(T)$. We remark that $D$ depends only on the first $n+1$ rows of $T$.

For a fixed value of the parameters, consider the polyhedron $P := P(A, b)$ defined by $\{x \in \mathbf{R}^n \mid A \cdot x \leq b\}$. We first show that if $D = 0$ then $P$ is empty, unbounded or a singleton.

By Lemma 1.4.3, if $x$ is a vertex of $P$, then there exists $1 \leq j \leq n+1$ such that $det(A^{(j)}) \neq 0$ and $A^{(j)} \cdot x = b^{(j)}$ where $(A^{(j)}|b^{(j)})$ results from the elimination of the $j^{th}$ row in $(A|b)$.

Now, if $D^{(j)} := det(A^{(j)}) \neq 0$, Cramer's method can be applied to find $x^{(j)} \in \mathbf{R}^n$, the unique solution to the system $A^{(j)} \cdot x = b^{(j)}$. For $1 \leq i \leq n$, call $A_i^{(j)}$ the matrix formed by replacing the $i^{th}$ column in $A^{(j)}$ by $b^{(j)}$. As a result of Cramer's method we obtain

$$x^{(j)} = \left( \frac{det(A_1^{(j)})}{D^{(j)}}, \dots, \frac{det(A_n^{(j)})}{D^{(j)}} \right)^{\mathsf{T}}. \tag{1.4.2}$$

**Lemma 1.4.5.** For $1 \leq j \leq n+1$, if $D(T) = 0$ and $D^{(j)}(T) \neq 0$ then $A(T) \cdot x^{(j)} = b(T)$.

*Proof.* From Cramer's method we know that $A^{(j)} \cdot x^{(j)} = b^{(j)}$ and so it remains to verify the equality

$$(t_1^{(j)}, \dots, t_n^{(j)}) \cdot x^{(j)} = b^{(j)}.$$

This last equality is equivalent to $D(T) = 0$, for multiplying

$$(t_1^{(j)}, \dots, t_n^{(j)}) \cdot x^{(j)} - b^{(j)}$$

by $D^{(j)}$ we obtain

$$t_1^{(j)} \cdot det(A_1^{(j)}) + \dots + t_n^{(j)} \cdot det(A_n^{(j)}) - b^{(j)} \cdot D^{(j)}$$

that is exactly the Laplace expansion of the determinant of $(A|b)$ by its $j^{th}$ row. □

**Proposition 1.4.6.** If $D(T) = 0$ and there exists $1 \leq j \leq n+1$ such that $D^{(j)}(T) \neq 0$ then $P(A, b)$ is unbounded or contains exactly one point.

*Proof.* Suppose that $P := P(A, b)$ is bounded, *i.e.*, suppose that it is a polytope. Let $j \leq n+1$ be a positive integer such that $D^{(j)}(T)$ is different from zero. The previous lemma shows that $x^{(j)} \in P$. We shall prove that $P$ equals $\{x^{(j)}\}$.

Since a polytope is the convex hull of its vertices (Lemma 1.4.4), if $P$ contains more than one point then it contains at least two vertices. But any vertex of the polytope $P$ is a solution of a non-singular subsystem $A^{(k)} \cdot x = b^{(k)}$ (for some $1 \leq k \leq n+1$, see Lemma 1.4.3). Since $D(T) = 0$, the previous lemma shows that the solution $x^{(k)}$ of any of these non-singular subsystems satisfies also the remaining equality (*i.e.*, $A \cdot x^{(k)} = b$). Thus, all these non-singular systems have the same unique solution. Hence, $P$ is a singleton. □

### 1.4.3 The Geometry of the Set of Parameters defining non-empty Polyhedra

We turn now to the study of the geometry of the set $\mathcal{I}^{(m,n)} = \{T \in \mathbf{R}^{m\times n} \times \mathbf{R}^m \mid P(T) \neq \emptyset\}$, *i.e.*, the geometry in the parameter space of the set of coefficients defining non-empty polyhedra. For any $k \in \mathbf{N}$ and any $x \in \mathbf{R}^k$, we use the usual notation for the maximum norm, $\|x\| = sup\{|x_i| \mid 1 \leq i \leq k\}$, and denote, for any $\varepsilon > 0$, by $B_\varepsilon(x)$ the ball $\{y \in \mathbf{R}^k \mid \|x - y\| < \varepsilon\}$.

We define $\mathcal{B} := \{T \in \mathbf{R}^{m\times n} \times \mathbf{R}^m \mid P(T) \text{ is a polytope}\}$ and remark that $T \notin \mathcal{B}$ if and only if $P(T)$ is unbounded. By Paragraph 7.2.3 in [Pad99], we know that:

- $P = P(T_H, T_h)$ is unbounded if and only if it contains a halfline, *i.e.*, a subset of the form $L_{xy} = \{x + \lambda y \mid \lambda \geq 0\}$, with $\|y\| = 1$;
- the halfline $L_{xy}$ is contained in $P$ if and only if $x \in P$ and $T_H \cdot y \leq 0$ holds.

Hence, the polyhedron $P(T_H, T_h)$ is unbounded if and only if it is non-empty and $L(T_H) := \{y \in \mathbf{R}^n \mid T_H \cdot y \leq 0 \text{ and } \|y\| = 1\}$ is non-empty.

Although it is not true that $\mathcal{B}$ is an open set, we have the following two results.

**Lemma 1.4.7.** The set $\{T \in \mathbf{R}^{m\times n} \times \mathbf{R}^m \mid L(T_H) \neq \emptyset\}$ is closed.

*Proof.* We use the closed map lemma: For any Hausdorff space $X$ and any compact space $Y$, the canonical projection map $\pi_1 : X \times Y \to X$ is closed.

Since the set $S^{n-1} := \{y \in \mathbf{R}^n \mid \|y\| = 1\}$ is compact and $\mathcal{C} := \{(T, y) \in \mathbf{R}^{m\times n} \times \mathbf{R}^m \times S^{n-1} \mid T_H \cdot y \leq 0 \text{ and } \|y\| = 1\ \}$ is closed, the canonic projection, $\pi_1(\mathcal{C})$, of $\mathcal{C}$ the onto $\mathbf{R}^{m\times n} \times \mathbf{R}^m$ is closed. Since $\pi_1(\mathcal{C}) = \{T \in \mathbf{R}^{m\times n} \times \mathbf{R}^m \mid L(T_H) \neq \emptyset\}$, the lemma is proved. □

**Proposition 1.4.8.** Let $T$ be such that $P(T)$ is a non-empty polytope. Then, $T$ is an interior point of $\mathcal{B}$.

*Proof.* If $P(T)$ is a non-empty polytope, then $T \in \mathcal{B}$ and $L(T_H) = \emptyset$. By the previous lemma there exists an $\varepsilon > 0$ such that from $\widehat{T} \in B_\varepsilon(T)$ it follows $L(\widehat{T}_H) = \emptyset$. Then, for any such $\widehat{T} \in B_\varepsilon(T)$, $P(\widehat{T})$ is a polytope. Thus, $T$ is an interior point of $\mathcal{B}$. □

**Proposition 1.4.9.** If $T \in \mathcal{I}^{(m,n)}$ is a point in the parameter space such that the polyhedron $P(T) \subseteq \mathbf{R}^n$ is not fully dimensional, then $T \in \partial\mathcal{I}^{(m,n)}$.

*Proof.* Let $T \in \mathcal{I}^{(m,n)}$, then $P := P(T)$ is non-empty. Suppose it is not fully dimensional. We claim that there exists at least one row in $T$ —say, the row $(a|b)$— such that its associated equality is satisfied by all the points in $P$, *i.e.*, $x \in P$ implies $a \cdot x = b$.

For if not, there exists for each $i$, $1 \leq i \leq n$, a point $x^{<i>} \in P$ not satisfying the equality associated with the $i^{th}$ row of $T$. We define $x := \sum_{i=1}^{n} \frac{x^{<i>}}{n}$. Being a convex combination of points in $P$, $x \in P$, *i.e.*, $x$ satisfies the $m$ inequalities associated with $T$. moreover, because of the linearity of the equations, the point $x$ satisfies the $m$ *strict* inequalities associated with $T$. Then, $x$ is an interior point of $P$, contradicting the fact that $P$ is not fully dimensional. This proves the claim.

Suppose now that all the points in $P$ satisfy the equality associated with the $i^{th}$ row of $T$. For any given $\varepsilon > 0$ we construct new parameters $T_\varepsilon$ such that $\|T - T_\varepsilon\| = \varepsilon$ and $T_\varepsilon \notin \mathcal{I}^{(m,n)}$.

To do so, we replace in $T$ the parameter $b^{(i)}$ by $b^{(i)} - \varepsilon$ to get the new parameters $T_\varepsilon$. Since $P(T_\varepsilon) \subseteq P(T)$ and no point in $P(T)$ satisfies the $i^{th}$ inequality associated with $P(T_\varepsilon)$, we conclude that $P(T_\varepsilon)$ is empty. Hence, $T_\varepsilon \notin \mathcal{I}^{(m,n)}$. Thus, $T \in \partial\mathcal{I}^{(m,n)}$. □

## 1.5 Counting the Limiting Hypersurfaces

In this section, we consider $T \in \mathbf{R}^{m\times n} \times \mathbf{R}^m$ with $m \geq n+1$. We will prove that there exists a limiting hypersurface for $\mathcal{I}^{(m,n)}$, associated with the first $n+1$ rows of $T$ (among the original $m$), involving all the $(n+1) \times (n+1)$ parameters in these rows. Afterwards, by a simple symmetry argument, it will follow that there are at least $\binom{m}{n+1}$ different limiting hypersurfaces for $\mathcal{I}$.

**Lemma 1.5.1.** The set $Z_D := \{T \in \mathbf{R}^{m\times n} \times \mathbf{R}^m \mid D(T) = 0\}$ is an irreducible hypersurface.

*Proof.* Since the polynomial $D(T)$ (as the determinant of a generic matrix) takes positive and negative values in $\mathbf{R}^{m\times n} \times \mathbf{R}^m$, Proposition 4.5.1 in [BCR98] implies that $dim(Z_D) = m(n+1) - 1$. The fact that $Z_D$ is an irreducible hypersurface follows now from the irreducibility of the determinant. □

**Proposition 1.5.2.** The irreducible hypersurface in the parameters space $Z_D$ defined by the equation $D(T) = 0$ is a limiting hypersurface for the set $\mathcal{I}^{(m,n)}$.

We prove the proposition directly from the definition of limiting hypersurface, *i.e.*, we prove that $dim(Z_D \cap \partial\mathcal{I}^{(m,n)}) = m(n+1) - 1$. To do so, we construct a non-singular point $\widetilde{T} \in Z_D$. We then prove that there exists $\varepsilon > 0$ such that any $T \in B_\varepsilon(\widetilde{T}) \cap Z_D$ satisfies $T \in \partial\mathcal{I}^{(m,n)}$. Before we prove Proposition 1.5.2, we need two lemmas.

We define $\widetilde{T} \in \mathbf{R}^{m\times n} \times \mathbf{R}^m$ as follows:

$$\widetilde{T} := \begin{pmatrix} & & & 0 \\ & I_{n\times n} & & \vdots \\ & & & 0 \\ -1 & \cdots & -1 & 0 \\ 0 & \cdots & 0 & 1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 1 \end{pmatrix}. \tag{1.5.3}$$

Since the origin of the standard coordinate system in $\mathbf{R}^n$ is the unique solution to the inequalities associated with the first $n+1$ rows of $\widetilde{T}$ and satisfies the remaining $m - n - 1$ inequalities associated with $\widetilde{T}$, we conclude that $P(\widetilde{T}) = \{0\}$.

We now prove that any $T$ in a small neighborhood of $\widetilde{T}$ satisfies

- $P(T)$ is a polytope contained in $B_1(0) \subset \mathbf{R}^n$ and
- $P(T)$ equals $P(A(T), b(T))$.

We remark that $P(\widetilde{T})$ satisfies both properties. We define $\widetilde{A} := A(\widetilde{T})$ and $\widetilde{b} := b(\widetilde{T})$.

**Lemma 1.5.3.** There exists $\varepsilon > 0$ such that any $(H|h) \in B_\varepsilon(\widetilde{A}|\widetilde{b})$ satisfies $P(H, h) \subseteq B_1(0)$.

*Proof.* By Proposition 1.4.8, there exists an $\varepsilon_1 > 0$ such that for any $(H|h) \in B_{\varepsilon_1}(\widetilde{A}|\widetilde{b})$, $P(H, h)$ is a polytope.

If $P(H, h)$ is a non-empty polytope, by Lemma 1.4.4, it is contained in $B_1(0)$ if and only if all its vertices are contained in $B_1(0)$. Hence, we shall bound the vertices. To do this we use the fact that the vertices move continuously with respect to the parameters, near $(\widetilde{A}|\widetilde{b})$.

We first remark that, since all the minors $D^{(j)}(\widetilde{A})$ are non-zero (for $1 \leq j \leq n+1$), there exists $\varepsilon_2 > 0$, $\varepsilon_2 \leq \varepsilon_1$, such that $(H|h) \in B_{\varepsilon_2}(\widetilde{A}|\widetilde{b})$ implies $D^{(j)}(H) \neq 0$ $(1 \leq j \leq n+1)$.

Hence, for any $(H|h) \in B_{\varepsilon_2}(\widetilde{A}|\widetilde{b})$ the polytope $P(H|h)$ has at most $n+1$ different vertices (see Lemma 1.4.3) defined by the $n+1$ non-singular subsystems resulting from the elimination of one row from $(H|h)$. In fact, for each $j \leq n+1$ a continuous functions $V^{(j)} : B_{\varepsilon_2}(\widetilde{A}|\widetilde{b}) \to \mathbf{R}$ can be defined, associating to each matrix the norm of the result of the application of Cramer's method to the subsystem resulting from the elimination of the $j^{th}$ row from $(H|h)$ (*i.e.*, $V^{(j)}(H|h)$ is the norm of the $j^{th}$ hypothetical vertex of $P(H, h)$). Now, since for all $j < n+1$ the equality $V^{(j)}(\widetilde{A}|\widetilde{b}) = 0$ holds, the continuity of $V^{(j)}$ at $(\widetilde{A}|\widetilde{b})$ implies that there exists a neighborhood of $(\widetilde{A}|\widetilde{b})$ where the $V^{(j)}$ are all bounded by 1.

Whence, there exists $\varepsilon > 0$, $\varepsilon \leq \varepsilon_2$ such that for all $(H|h) \in B_\varepsilon(\widetilde{A}|\widetilde{b})$, the polytope $P(H, h)$ is contained in $B_1(0)$. □

For any fixed $T \in \mathbf{R}^{m \times n} \times \mathbf{R}^m$, let us write, as before, $A$ for $A(T)$ and $b$ for $b(T)$. Recall that $(A|b) \in \mathbf{R}^{(n+1)\times(n+1)}$.

**Lemma 1.5.4.** Let $\widetilde{T}$ be as in Equation 1.5.3. There exists an $\varepsilon > 0$ such that any $T \in B_\varepsilon(\widetilde{T})$ satisfies $P(T) = P(A, b) \subseteq B_1(0)$ and $D^{(j)}(T) \neq 0$, for $1 \leq j \leq n+1$.

*Proof.* Consider $0 < \varepsilon < \frac{1}{n+1}$ satisfying the previous lemma and $T \in B_\varepsilon(\widetilde{T})$. Then, $P(A, b) \subseteq B_1(0)$ and $D^{(j)}(T) \neq 0$ by construction, for $1 \leq j \leq n+1$. It remains to prove that $P(T) = P(A, b)$.

Clearly, $P(T) \subseteq P(A, b)$. To prove the other inclusion, let $x \in P(A, b)$ and consider for any $i \in \mathbf{N}$, with $n+1 < i \leq m$, the inequality associated with the $i^{th}$ row of $T$. We prove that $x$ satisfies this inequality, *i.e.*, we prove $t_1^{(i)} \cdot x_1 + \cdots + t_n^{(i)} \cdot x_n \leq b^{(i)}$.

Since $\widetilde{b}^{(i)} = 1$, $\widetilde{t}_1^{(i)} = \cdots = \widetilde{t}_n^{(i)} = 0$ and $\|T - \widetilde{T}\| < \frac{1}{n+1}$, we have that $b^{(i)} > \frac{n}{n+1}$ and that $\|(t_1^{(i)}, \ldots, t_n^{(i)})\| < \frac{1}{n+1}$ holds.

Since $\|x\| < 1$, we have $t_1^{(i)} \cdot x_1 + \cdots + t_n^{(i)} \cdot x_n \leq \frac{n}{n+1} < b^{(i)}$. Hence, $x$ satisfies the inequality associated with the $i^{th}$ row of $T$, for any $n+1 < i \leq m$.

Thus, $P(T) = P(A, b)$, which completes the proof. □

We are ready to prove Proposition 1.5.2.

*Proof of Proposition 1.5.2.* Let $\widetilde{T}$ be as in the Equation (1.5.3). Since the column vector $b(\widetilde{T})$ is composed of zeros, $D(\widetilde{T}) = 0$, *i.e.*, $\widetilde{T} \in Z_D$.

Since $\widetilde{T}$ is a non-singular point of $Z_D$ (for instance, $\frac{\partial D}{\partial b^{(1)}}(\widetilde{T}) = D^{(1)}(\widetilde{T}) = 1 \neq 0$), the implicit function theorem implies that, for any $\varepsilon > 0$, $dim(Z_D \cap B_\varepsilon(\widetilde{T})) = dim(Z_D) = m(n+1) - 1$, *i.e.*, the local dimension of $Z_D$ at $\widetilde{T}$ is $dim(Z_D)$.

Let $\varepsilon$ be a positive number satisfying Lemma 1.5.4. We show that $Z_D \cap B_\varepsilon(\widetilde{T}) \subseteq \partial\mathcal{I}^{(m,n)}$.

Consider $T \in B_\varepsilon(\widetilde{T})$ and suppose that $D(T) = 0$. By Lemma 1.5.4, we have that, for $1 \leq j \leq n+1$, $P(T) = P(A, b) \subseteq B_1(0)$ and that $D^{(j)}(T) \neq 0$, for $1 \leq j \leq n+1$. Thus, Proposition 1.4.6 implies that $P(T)$ is a singleton. Proposition 1.4.9 shows then that $T \in \partial\mathcal{I}^{(m,n)}$. Hence, $dim(\partial\mathcal{I}^{(m,n)} \cap Z_D) = m(n+1) - 1$.

Now, the fact that $Z_D$ is a limiting hypersurface follows from Lemma 1.5.1. □

**Corollary 1.5.5.** The set $\mathcal{I}^{(m,n)}$ has $\Omega(\binom{m}{n+1})$ different limiting hypersurfaces given by the $(n+1) \times (n+1)$ minors of the parameter matrix.

*Proof.* By the previous proposition, the first minor defines a limiting hypersurface. Considering any other $(n+1) \times (n+1)$ minor of the parameters matrix $T$ we can argue analogously getting an irreducible hypersurface. Since there are $\binom{m}{n+1}$ such minors and the variables involved in each minor are different, the set $\mathcal{I}^{(m,n)}$ has at least $\binom{m}{n+1}$ different limiting hypersurfaces. □

## 1.6 Complexity Lower Bounds

In this section, we use Proposition 1.3.2 and Corollary 1.5.5 to prove exponential lower bounds for the length of any quantifier-free formula expressing the set $\mathcal{I}^{(2n,n)}$, if polynomials are given in dense or sparse representation. Afterwards, we analyze the consequences of these results for algebraic computation trees.

The notion of *length of a formula* strongly depends on the way terms (*i.e.*, polynomials) are represented in the formula. Once this representation is fixed, the notion of length of an atomic formula follows naturally as the sum of the lengths of the terms involved, plus one. The recursive definition of the length of a formula is completed stating $|\exists x\ \varphi| = |\varphi| + 1$, $|\neg\varphi| = |\varphi| + 1$ and $|\varphi \star \psi| = |\varphi| + |\psi| + 1$ for $\star \in \{\vee, \wedge\}$.

We shall consider the dense and the sparse representation of polynomials. In the case of dense representation two parameters are commonly used in order to measure the size of a polynomial: the degree and the number of variables. Let $f \in \mathbf{R}[x_1, \dots, x_k]$ be a polynomial of degree $d$, we define its *dense length* as $\binom{d+k}{k}$.

The sparse representation is restricted to polynomials with integer coefficients, since the heights of these coefficients will play an important role in the bounds. The sparse representation of a polynomial $f \in \mathbf{Z}[x_1, \dots, x_k]$ consists of the list of pairs $(\mu, a_\mu)$, where $\mu = (\mu_1, \dots, \mu_k) \in \mathbf{N}^k$ and $a_\mu \in \mathbf{Z}$, corresponding to all non-zero coefficients of $f$. The *sparse length* of a polynomial is defined as the bit length of the concatenation of the absolute values of all the numbers $\mu_1, \dots, \mu_k, a_\mu, \dots$ in this list, written in binary.

For the sake of succinctness, the length of a formula $\psi$ with polynomials codified in dense form, will be called the *dense length* of $\psi$, and denoted $|\psi|_d$. If polynomials are codified in sparse form, we call it the *sparse length* of $\psi$ and denote $|\psi|_s$.

## 1.6.1 Dense Representation

**Proposition 1.6.1.** If $\psi$ be a first-order formula with polynomials codified in dense form, then $|\psi|_d$ is bounded from bellow by the sum of the degrees of the different irreducible polynomials intervening in $\psi$.

*Proof.* Let $Q_1, \dots, Q_s$ be the non-constant polynomials appearing in $\psi$, with factorizations $Q_i = P_{i,1} \cdots P_{i,k_i}$ where $P_{i,j}$ are the irreducible polynomials of positive degree intervening in $\psi$. Let $d_i = \deg(Q_i)$. Since the dense length of $Q_i$ is at least $d_i + 1$, the dense length of $\psi$ is bounded from bellow by $\sum_{i=1}^{s}(d_i + 1)$. Since $d_i = \sum_{j=1}^{k_i} \deg(P_{i,j})$, the sum of the degrees of the different irreducible polynomials intervening in $\psi$ is a lower bound for the dense length of $\psi$. □

**Corollary 1.6.2.** Any quantifier-free formula, written using dense representation of polynomials and expressing the set $\mathcal{I}^{(2n,n)}$, has size $\Omega(4^n \cdot \sqrt{n})$.

*Proof.* Let $\psi$ be a quantifier-free formula describing the set $\mathcal{I}^{(2n,n)}$. Corollary 1.5.5 shows that the $\binom{2n}{n+1}$ minors of the parameter matrix $T$ define different limiting hypersurfaces for $\mathcal{I}^{(2n,n)}$. Proposition 1.3.2 implies that these minors intervene in $\psi$. Since these polynomials have degree $n + 1$, Proposition 1.6.1 implies that the dense length of any quantifier-free formula describing this set is $\Omega(\binom{2n}{n+1}(n+1))$. The conclusion follows immediately from an application of Stirling's formula. □

This gives a sub-exponential lower bound for the worst case complexity of the elimination of a single quantifier block, for any algorithm using dense representation of polynomials.

**Corollary 1.6.3.** There exist a real constant $c > 1$ such that, if polynomials are codified using the dense representation, any algorithm for the elimination of one existential block of quantifiers performs, on inputs of length $L$, $\Omega(c^{\sqrt{L}})$ operations in the worst case.

*Proof.* A straightforward computation shows that $|\phi^{(2n,n)}|_d = \mathcal{O}(n^2)$, where $\phi^{(2n,n)}$ is the formula introduced in the Equation (1.2.1). Corollary 1.6.2 shows that any quantifier-free formula $\psi$ expressing the same set has $|\psi|_d = \Omega(4^n)$. Since any algorithm for the elimination of a single quantifier block has to write down the output, we conclude that the worst case complexity for inputs of dense length $L$ is bounded from below by $\Omega(c^{\sqrt{L}})$ for a real constant $c > 1$. □

## 1.6.2 Sparse Representation

To prove an analog lower bound for sparse codification of polynomials we use the following result from [AKS07].

**Proposition 1.6.4.** Let $f \in \mathbf{Z}[x_1, \ldots, x_k]$ and consider the factorization

$$f = q \cdot \prod_p p^{e_p}$$

where $q$ is a cyclotomic polynomial, $p \in \mathbf{Q}[x_1, \ldots, x_k]$ runs over all non-cyclotomic irreducible factors of $f$, and $e_p$ is the corresponding multiplicity. Then,

$$\sum_p e_p \leq 5^6 \cdot k^3 \cdot \log \|f\|_1 \cdot \log^3(8k \deg(f)).$$ □

We immediately obtain that the number of different non-cyclotomic irreducible polynomials intervening in a formula $\psi$ is polynomially bounded in terms of the sparse length of $\psi$.

**Corollary 1.6.5.** There exists a positive constant $c_1 \in \mathbf{R}$ such that, any quantifier-free first-order formula $\psi$, has sparse length

$$|\psi|_s = \Omega(\mathrm{nf}(\psi)^{c_1}),$$

where $\mathrm{nf}(\psi)$ is the number of different non-cyclotomic irreducible factors intervening in $\psi$. □

**Corollary 1.6.6.** There exists a constant $c_2 \in \mathbf{R}$, $c_2 > 1$ such that any quantifier-free formula expressing the set $\mathcal{I}^{(2n,n)}$ has sparse length $|\psi|_s = \Omega({c_2}^n)$.

*Proof.* We argue as in the proof of Corollary 1.6.2. Let $\psi$ be a quantifier-free formula describing the set $\mathcal{I}^{(2n,n)}$. Corollary 1.5.5 shows that the $\binom{2n}{n+1}$ minors of the parameter matrix $T$ define different limiting hypersurfaces for $\mathcal{I}^{(2n,n)}$. Proposition 1.3.2 shows that these minors intervene in $\psi$. Thus, Proposition 1.6.5 implies that any quantifier-free formula describing this set has sparse length $\Omega(\binom{2n}{n+1}^{c_1})$.

Applying Stirling's formula yields $\binom{2n}{n+1}^{c_1} \sim (\frac{4^n}{\sqrt{\pi n}})^{c_1}$. Taking $c_2$ such that $1 < c_2 < 4^{c_1}$ we obtain $(\frac{4^n}{\sqrt{\pi n}})^{c_1} = \Omega({c_2}^n)$, which completes the proof. □

We conclude with a sub-exponential lower bound for the worst case complexity of any algorithm for the elimination of a quantifier block, if sparse representation of polynomials is used.

**Corollary 1.6.7.** There exists a real constant $c > 1$ such that, if polynomials are codified using the sparse representation, any algorithm for the elimination of one existential block of quantifiers performs $\Omega(c^{\sqrt[3]{L}})$ operations in the worst case on inputs of length $L$.

*Proof.* A straightforward computation shows that $|\phi^{(2n,n)}|_s = \mathcal{O}(n^3)$. Corollary 1.6.6 shows that any quantifier-free formula $\psi$ expressing the same set has sparse length $|\psi|_s = \Omega({c_2}^n)$. Since any algorithm for the elimination of a quantifier block has to write down the output, we conclude that the worst case complexity for inputs of sparse length $L$ is bounded from below by $\Omega(c^{\sqrt[3]{L}})$, for a real constant $c > 1$. □

### 1.6.3 Algebraic Computation Trees

A natural model to prove complexity lower bounds is that of algebraic computation trees (see Appendix A.2 for a definition, see also [BO83, Str83] and [BCS97, Bür01] for more references). Given an algebraic computation tree $S$, we write, following [BCS97], $C^{*,\leq}(S)$ for the multiplicative branching complexity of $S$.

We prove the following general lower bound for the multiplicative branching complexity of algebraic computation trees, based on the notion of limiting hypersurface.

**Proposition 1.6.8.** Consider an algebraic computation tree, $S$, deciding membership of a set $W \subset \mathbf{R}^n$. Let $H_1, \ldots, H_s$ be the different limiting hypersurfaces of $W$. Suppose that these hypersurfaces are described by irreducible polynomials of degrees $d_1, \ldots, d_s$ respectively, and call $D = \sum_{i=1}^{s} d_i$. Then, the multiplicative branching complexity of $S$ is bounded from bellow by $\log(D)$:

$$C^{*,\leq}(S) > \log(D).$$

*Proof.* Since algebraic computation trees can be naturally translated to first-order formulas, and, in such a translation, branching nodes translate to atomic formulas, Proposition 1.3.2 implies that a multiple of the polynomial describing each limiting hypersurface of $W$ must be evaluated at some branching node of $S$. This implies that the sum of the degrees of the polynomials involved in the different branching nodes of $S$ is at least $D$.

On the other hand, a routine computation shows that the sum of the degrees of the polynomials involved in the different branching nodes of $S$ is bounded from above by $2^{C^{*,\leq}(S)} - 1$.

Thus, $2^{C^{*,\leq}(S)} > D$. Taking logarithms we conclude that $C^{*,\leq}(S) > \log(D)$, which completes the proof. □

**Proposition 1.6.9.** For $n$ sufficiently large, the multiplicative branching complexity of any algebraic computation tree, $S$, accepting the set $\mathcal{I}^{(2n,n)}$ satisfies

$$C^{*,\leq}(S) > 2n.$$

*Proof.* Let $S$ be an algebraic computation tree accepting the set $\mathcal{I}^{(2n,n)}$. Proposition 1.5.5 shows that $\mathcal{I}^{(2n,n)}$ has $\binom{2n}{n+1}$ limiting hypersurfaces of degree $n+1$. Thus, by Proposition 1.6.8, we have $C^{*,\leq}(S) > \log((n+1)\binom{2n}{n+1})$. Application of Stirling's formula immediately yields $(n+1)\binom{2n}{n+1} > c \cdot 4^n \cdot \sqrt{n}$, for a positive constant $c$. Thus, $C^{*,\leq}(S) > 2n + \frac{\log(n)}{2} + \log(c)$, which completes the proof. □

In other words, we have proved the following.

**Corollary 1.6.10.** For $n$ sufficiently large, the multiplicative branching complexity of any computation tree that solves the linear programming feasibility problem in $\mathbf{R}^n$ for $2n$ constraints is bounded from bellow by $2n$.

# 2

# The Sign Condition Problem: Upper and Lower Bounds

**Abstract.** In this chapter, we study the algebraic complexity of the sign condition problem for any given a family of polynomials. Essentially, the problem consists in determining the sign condition satisfied by a fixed family of polynomials at a query point, performing as little arithmetic operations as possible. After defining precisely the sign condition and the point location problems, we introduce a method called *the dialytic method* to solve the first problem efficiently. This method involves a *linearization* of the original polynomials and provides the best known algorithm to solve the sign condition problem. Moreover, using a technique that resembles that of Chapter 1, we prove a lower bounds showing that the dialytic method is almost optimal.

## 2.1 Introduction

Given a partition $\mathcal{S}$ of $\mathbf{R}^n$ into disjoint regions, the *point-location problem* for the partition $\mathcal{S}$ asks to determine the region containing a query point. Point location is a basic problem in computational geometry and has inspired several data structures (see [Sno04]). It has applications in different domains, including geographic information systems (GIS) and robot motion planning. We give now a precise definition of this problem.

**Definition 2.1.1.** An algorithm taking as input a point in $\mathbf{R}^n$ and with a finite set of possible outputs $\{O_1, ..., O_k\}$ solves the point location problem for a given partition $\mathcal{S}$ of $\mathbf{R}^n$ if it satisfies the following condition:

for any pair of points $x, y \in \mathbf{R}^n$, the algorithm returns the same output on both inputs $x$ and $y$, if and only if $x$ and $y$ belong to the same element of the partition $\mathcal{S}$ of $\mathbf{R}^n$.

The output of a point location algorithm can be seen as a *label* identifying the region containing the query point. The regions in the given partition may have very complex descriptions. Using labels instead of these descriptions, we obtain algorithms whose query time is independent of their size. Using the terminology from database theory, we are measuring the point location search time and not its report time.

### 2.1.1 The Sign Condition and Point Location Problems for a Family of Polynomials

**Definition 2.1.2.** Let $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$ be a finite family of polynomials. The realizations of the $\mathcal{P}$-sign condition form a partition of $\mathbf{R}^n$ denoted by $\mathcal{S}(\mathcal{P})$. The elements of $\mathcal{S}(\mathcal{P})$ are not necessarily connected subsets of $\mathbf{R}^n$. We define the *arrangement induced by* $\mathcal{P}$ as the partition of $\mathbf{R}^n$ consisting of the connected components of the realization of the sign conditions of the family $\mathcal{P}$ and denote it by $\mathcal{A}(\mathcal{P})$. The elements of $\mathcal{A}(\mathcal{P})$, are called the *faces* of the arrangement induced by $\mathcal{P}$.

The point location problem for the partition $\mathcal{A}(\mathcal{P})$ is called the *point location problem for the family* $\mathcal{P}$ and will be studied in the next chapter. In the present chapter, we study the point location problem for the partition $\mathcal{S}(\mathcal{P})$, called *the sign condition problem for the family* $\mathcal{P}$.

In Section 2.2.1, we introduce the different representations of polynomials that we consider: circuit, dense arithmetic and dense bit representations. In Section 2.3 we introduce the *dialytic method*; it solves the sign condition problem for a given family of polynomials in any of the mentioned representations (see Theorem 2.3.2 and Corollaries 2.3.3 and 2.3.4). Finally, in Section 2.4 we present sharp lower bounds for this problem.

### 2.1.2 Basic Observations

The simplest instance of point location is list searching. Given different points $x_1, x_2, ..., x_s \in \mathbf{R}$, consider indices $1 \leq i_1, ..., i_s \leq s$ such that $x_{i_1} < ... < x_{i_s}$. Then, a partition of $\mathbf{R}$ into disjoint regions is determined by these points and the intervals $(-\infty, x_{i_1}), (x_{i_1}, x_{i_2}), ..., (x_{i_{s-1}}, x_{i_s}), (x_{i_s}, +\infty)$. The list searching problem already illustrates several aspects of the general point location problem. On the one hand, without any preprocessing, the point location query for this partition of $\mathbf{R}$ can be answered in time $\mathcal{O}(s)$ performing a linear search. On the other hand, if we order the points in a preprocessing stage (using $\mathcal{O}(s\log(s))$ operations), the query can be answered performing a binary search involving only $\mathcal{O}(\log(s))$ operations. In what follows, we generalize this second method to higher dimensions and higher degrees.

The space $\mathbf{R}^n$ can be divided in $2^n$ regions by $n$ hyperplanes. If we consider $s > n$ hyperplanes in $\mathbf{R}^n$, we will no longer obtain $2^s$ regions determined. Some *implications* appear; its associated system of equations is overdetermined. It is easy to see this in the plane: two lines divide the plane in four different regions, but no three lines divide the plane in eight regions. Not all syntactically possible sign conditions are simultaneously geometrically realizable by any family of hyperplanes.

An analogous phenomenon can be observed for algebraic hypersurfaces of higher degree. In 1968, Warren [War68] proved that the number of connected components of the realizations of strict sign conditions of a family of $s$ polynomials in $n$ variables of degree at most $d$, is bounded by $(4esd/n)^n$ (see also [Mil64, Gri88, HRS90b, JS00, LB01, BPR10]).

For a fixed $n$, the number of syntactically definable sign conditions, $3^s$, grows exponentially with $s$, while the number of simultaneously geometrically realizable sign conditions grows only polynomially in $s$ and $d$. Moreover, the number of faces of the induced arrangement is also polynomial in $s$ and $d$, for any fixed $n$. The best bound known today [BPR10] for the number of faces of $\mathcal{A}(\mathcal{P})$ is

$$\frac{(2d)^n}{n!}s^n + \mathcal{O}(s^{n-1}).$$

Observing this bound, it is natural to try to design an algorithm that solves the point location problem performing a number of arithmetic operations that grows logarithmically in $s$, the number of polynomials in the family $\mathcal{P}$.

### 2.1.3 Related Work

**Linear Case.** Let $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$ be a family of $s$ linear polynomials. We remark that, since the non-empty realizations of $\mathcal{P}$-sign conditions are convex sets, the sign condition and point location problems for the family $\mathcal{P}$ coincide.

Dobkin and Lipton [DL76] were the first to present an algorithm solving the point location problem, in this context, whose query time is logarithmic in the numbers $s$ of polynomials; the size of the associated data structure is $\mathcal{O}(s^{2^n-2})$. Clarkson [Cla87] improved the space complexity to $\mathcal{O}(s^{n+\varepsilon})$; in both cases the query time is exponential in $n$. Meyer auf der Heide [MadH84] solved a particular instance of this problem (he considered hyperplanes with integer coefficients only), that allowed him to derive the existence of a non-uniform polynomial-time solution to the Knapsack Problem (see also [MadH88]). Finally, in 1993, Meiser [Mei93] gave a solution with running time $\mathcal{O}(n^5 \log(s))$ and space bound $\mathcal{O}(s^{n+\varepsilon})$, for arbitrary $\varepsilon > 0$. The preprocessing is done in expected time $\mathcal{O}(s^{n+1+\varepsilon})$, for arbitrary $\varepsilon > 0$. This last algorithm allowed Meiser to derive a strongly polynomial non-uniform algorithm for the NP-complete Knapsack problem (see also Chapter 3 in [BCS97]). After the next paragraph, we give a brief description of Meiser's algorithm.

**Polynomial case.** Chazelle and Sharir [CS90] (see also [CEGS91]) proposed an algorithm, based on Collins' Cylindrical Algebraic Decomposition [Col75], for the general algebraic point location problem in the traditional unit-cost RAM model. The complexity of their method is logarithmic in the number of polynomials, but in the complexity analysis they ignore the dependency of their method on the degree of the polynomials and on the dimension of the ambient space. This is a usual practice in computational geometry, where the degree of the polynomials and the dimension of the ambient space are assumed to be bounded by a constant.

Grigoriev [Gri00] bounded the branching (or topological) complexity of the sign condition problem from above by the logarithm of the number of faces of the arrangement $\mathcal{A}(\mathcal{P})$. Nevertheless, the algebraic complexity of Grigoriev's method depends linearly on $s$. See also [Koi00] for further details and for its relation with the $\mathsf{P} = \mathsf{NP}$ question over the reals.

Before presenting our work, we briefly summarize Meiser's algorithm for the linear case.

**Meiser's algorithm.** For future reference, we state Meiser's result [Mei93] precisely in the model of algebraic computation trees.

**Theorem 2.1.3.** Given a family, $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$, containing $s$ linear polynomials, there exists an algebraic computation tree $\Gamma_{\mathcal{P}}$ that solves the sign condition problem for the family $\mathcal{P}$ in time $\mathcal{O}(n^5 \log(s))$.

The size of the tree $\Gamma_{\mathcal{P}}$ is bounded by $\mathcal{O}(s^{n+\varepsilon})$ and it can be constructed in expected time $\mathcal{O}(s^{n+1+\varepsilon})$, for arbitrary $\varepsilon > 0$. □

Meiser's original algorithm uses the *trie* data structure. The conversion of this algorithm to the context of algebraic computation trees is straightforward. We observe that the upper bound stated by Meiser for his algorithm is not optimal. He claims an $\mathcal{O}(n^5 \log(s))$ bound, but more precisely it is $n^4 \log(n)^{\mathcal{O}(1)} \log(s)$. Meiser's method behaves well also in the bit model; see his article and the next section for further details.

Roughly, in a first *step* the algorithm evaluates at the query point all the polynomials in a subset $\mathcal{R}$ of $\mathcal{P}$ and determines the degenerated simplex (see [Mei93]) in a triangulation of $\triangle\mathcal{A}(\mathcal{R})$ of $\mathcal{A}(\mathcal{R})$ containing the query point. The set $\mathcal{R}$ and the triangulation $\triangle\mathcal{A}(\mathcal{R})$ are precomputed and have the following key properties:

- The cardinality of $\mathcal{R}$ is bounded by a polynomial in $n$,
- the degenerated simplex in the triangulation $\triangle\mathcal{A}(\mathcal{R})$ containing the query point can be determined in polynomial time in $n$, and
- only a constant fraction $\varepsilon$, $0 < \varepsilon < 1$ of the polynomials in $\mathcal{P}$ change their sign in each degenerated simplex in $\triangle\mathcal{A}(\mathcal{R})$.

In this way, after a logarithmic number of *steps* $(\log(s))$, the problem is reduced to a number of equations whose number depends on $n$ but not on $s$. Then, the sign condition is determined by direct evaluation.

We remark that Meiser's algorithm is completely linear, *i.e.*, it does not perform any non-scalar multiplication.

## 2.2 Computational Models and Representations of Polynomials

Our algorithms in this chapter are represented by algebraic computation trees over the real numbers (see A.2). We measure the number of arithmetic operations performed by an algorithm and call it its *algebraic complexity.*

In some cases, we are also interested in the *bit* or *binary complexity* of our algorithms. To measure this within our computational model, we restrict the arithmetic operations performed by our algorithms to integer numbers. Rational and algebraic numbers are represented by tuples of integers and we measure, besides the number of arithmetic operations, the bitsize of the integers involved in these operations (see Section A.4.1).

In this sense, algorithms (like that of Khachiyan [Kha79] for linear programming) that belong to the bit model but are not based on arithmetic operations, are out of the scope of our model.

Roughly, given a finite family of polynomials we construct, in a preprocessing stage, a data structure. Then, using this data structure, we answer some queries about the original family efficiently. Within the model of algebraic computation trees, the data structure is the algebraic computation tree itself.

The performance of a data structure is measured by the time spent in answering a query (called the *query time*), the time needed to construct the data structure (called the *preprocessing time*) and the *size* of the data structure.

Since the data structure is constructed only once, its query time and size are more important than its preprocessing time. If a data structure supports insertion and deletion operations, the *update time* is also relevant, but we shall not consider this situation.

The complexity of some queries depend on the output size—consider, for instance, the sign condition query described in the next section. We divide the query time in two parts: the *search time* and the *reporting time.* For some applications, it is important to distinguish different answers but not to write them down explicitly. In these cases, the search time plays a fundamental role.

**Lower Bounds.** For the lower bounds, we consider two models: algebraic computation trees and algebraic decision trees (see Appendix A for precise definitions and references). Since we are working over the field of the real numbers, branching nodes have three different immediate successors in the tree, corresponding to the three possible results of the sign test.

### 2.2.1 Representation of the Polynomials

Now, we describe the data types used, in this and the following chapter, to represent polynomials. Let us first introduce some notation.

**Definition 2.2.1.** A *sign condition* is an element of $\{0, 1, -1\}$. For $x \in \mathbf{R}$ we define

$$\operatorname{sgn}(x) := \begin{cases} -1 & \text{if } x < 0; \\ 0 & \text{if } x = 0; \\ 1 & \text{if } x > 0. \end{cases}$$

Let $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$. A $\mathcal{P}$*-sign condition*, $\sigma$, is an element of $\{-1, 0, 1\}^{\mathcal{P}}$. We say that $\mathcal{P}$ realizes the sign condition $\sigma$ at $x \in \mathbf{R}^n$, or that $x$ satisfies the sign condition $\sigma$ if, for every $P \in \mathcal{P}$, $\operatorname{sgn}(P(x)) = \sigma(P)$. We denote the sign condition realized by $\mathcal{P}$ at $x$ by $\operatorname{sgn}(\mathcal{P}, x)$. If $\sigma$ is a $\mathcal{P}$-sign condition, its *level* is defined as the cardinal of the set $\{P \in \mathcal{P} \mid \sigma(P) = 0\}$.

Let us denote by $H(m)$ the *height* (or absolute value) of an integer $m \in \mathbf{Z}$ and by $h(m)$ its *logarithmic height* (or bitsize) defined as $h(m) := \ulcorner \log(H(m)+1) \urcorner$.

For a polynomial $P \in \mathbf{Z}[X_1, ..., X_n]$, we denote by $H(P)$ its *height* defined as the maximal height of all its coefficients and, analogously, by $h(P)$ its *logarithmic height* defined as the maximal logarithmic height of all its coefficients.

Let be given a polynomial $F \in \mathbf{R}[X_1, ..., X_n]$. In this and the following chapter, we shall consider the following different representations of it. Besides the number $n$ of variables, each of these representations has associated some natural parameters measuring the complexity of the representation.

1. **Arithmetic-circuit representation**. The polynomial $F$ is represented by an arithmetic circuit $\Gamma$ over $\mathbf{R}$ (see [vzG86b, BCS97]) that computes it. We limit ourselves to division-free circuits. Let us denote by $L$ the non-scalar size of $\Gamma$ and observe that the degree of $F$ is bounded by $2^L$. The parameters associated with this representation are $n$ and $L$.

2. **Dense arithmetic representation**. Suppose that the polynomial $F$ has degree $d$. The dense arithmetic representation of $F$ consists on the tuple in $\mathbf{R}^{\binom{d+n}{n}}$ of its coefficients in the monomial basis. The parameters associated with this representation are $n$ and $d$.

3. **Dense bit representation**. We assume that the polynomial $F$ has integer coefficients. If $F$ has logarithmic height $\tau$ and degree $d$, its dense bit representation is the tuple in $\mathbf{Z}^{\binom{d+n}{n}}$ of its coefficients in the monomial basis, where each integer is represented by its bit encoding (of size at most $\tau$). The parameters associated with this representation are $n, d$ and $\tau$.

Given a family $\mathcal{F} := \{F_1, ..., F_s\}$ of polynomials in $\mathbf{R}[X_1, ..., X_n]$, the dense (arithmetic or bit) representation of $\mathcal{F}$ is simply the collection of the dense (arithmetic or bit) representations of each polynomial in the family $\mathcal{F}$.

On the other hand, the arithmetic-circuit representation the family $\mathcal{F}$ is division-free arithmetic circuit $\Gamma$ over $\mathbf{R}$ that computes all the polynomials in $\mathcal{F}$. Let us denote by $L$ the non-scalar size of $\Gamma$ and observe that the degrees of the polynomials in $\mathcal{F}$ are bounded by $2^L$. The parameters associated with this representation are $s, n$ and $L$.

We observe that the dense representation can be seen as a special case of the arithmetic-circuit representation with $L$ equal to $\binom{d+n}{n} - n - 1$, the number of monomials of degree between two and $d$ in $n$ variables.

## 2.3 The Dialytic Method to solve the Sign Condition Problem

In this section we introduce the dialytic method [1] and show how it enables us to reduce the sign condition problem to the linear case. We assume the arithmetic-circuit representation of polynomials. We recall that the dense arithmetic representation can be seen as a particular case of this representation.

1. The word *dialytic* comes from the Greek word $\delta\iota\acute{\alpha}\lambda\upsilon\sigma\iota\varsigma$, meaning separation [LSJM40]. This term was used by Sylvester [Syl42] when he introduced the resultant of a monic polynomial treating each monomial as a different variable.

Let us consider a family $\mathcal{F} := \{F_1, ..., F_s\}$ of polynomials in $\mathbf{R}[X_1, ..., X_n]$ and suppose that $\Gamma_{\mathcal{F}}$ is a division-free arithmetic circuit of non-scalar complexity $L$ that computes the family $\mathcal{F}$.

Suppose given a family of polynomials $\mathcal{G} = \{G_1, ..., G_k\} \subset \mathbf{R}[X_1, ..., X_n]$ that generate an $\mathbf{R}$-subspace $\mathbb{V}_{\mathcal{G}}$ of $\mathbf{R}[X_1, ..., X_n]$ that contains $\mathcal{F}$. Let us also suppose that the family $\mathcal{G}$ can be evaluated by a division-free arithmetic circuit of non-scalar complexity $L_{\mathcal{G}}$. We consider the following three different examples of the family $\mathcal{G}$, called *the family of generators*:

- As the family of generators we can take the family $\mathcal{G}_P$ composed of the polynomials computed by the non-scalar multiplication nodes in $\Gamma_{\mathcal{F}}$ together with a basis for the linear polynomials in $\mathbf{R}[X_1, ..., X_n]$. In this case, we have $k = L+n+1$ and $L_{\mathcal{G}_P} = L$. It is clear that any polynomial in $\mathcal{F}$ can be written as a linear combination of the polynomials in $\mathcal{G}_P$.

- Alternatively, as the family of generators we can take a maximal, $\mathbf{R}$-linearly independent subset $\mathcal{G}_{\mathcal{F}}$ of $\{F_1, ..., F_s\}$. In this case, we have $k$ equal to the dimension of the $\mathbf{R}$-subspace generated $\mathcal{F}$ (bounded by $L + n + 1$, as the previous example shows) and $L_{\mathcal{G}_{\mathcal{F}}} \le L$.

- Finally, if the polynomials $F_1, ..., F_s$ have degree bounded by $d$ we can also take, as the family of generators, the monomial basis $\mathcal{G}_B$ of $\mathbf{R}[X_1, ..., X_n]$ obtaining $k = \binom{n+d}{n}$ and $L_{\mathcal{G}_B} = \binom{n+d}{n} - (n+1)$.

We assume fixed any such family of generators $\mathcal{G} = \{G_1, ..., G_k\}$. Then, for $1 \le i \le s$ and $1 \le j \le k$, there exist constants $\alpha_j^{(i)} \in \mathbf{R}$ such that

$$F_i = \Sigma_{j=1}^k \alpha_j^{(i)} G_j.$$

Let $Z_1, ..., Z_k$ be new indeterminates and consider, for $1 \le i \le s$, the polynomials $\overline{F_i} := \Sigma_{j=1}^k \alpha_j^{(i)} Z_j \in \mathbf{R}[Z_1, ..., Z_k]$. Let us denote by $G : \mathbf{R}^n \to \mathbf{R}^k$ the function defined by $G(x) := (G_1(x), ..., G_k(x))$.

**Remark 2.3.1.** For any $x \in \mathbf{R}^n$ and for $1 \le i \le s$, the value of $F_i(x)$ is the same as the value of $\overline{F_i}(G(x))$.

In particular, this implies that a solution for the sign condition problem for the family of linear polynomials $\{\overline{F_1}, ..., \overline{F_1}\}$ induces a solution for the sign condition problem for the original family $\{F_1, ..., F_s\}$.

**Theorem 2.3.2.** Let $\mathcal{F} := \{F_1, ..., F_s\} \subset \mathbf{R}[X_1, ..., X_n]$ be a family of polynomials and suppose given an arithmetic circuit $\Gamma_{\mathcal{F}}$ of non-scalar complexity $L$ that computes the family $\mathcal{F}$.

Then, there exists an algebraic computation tree $\Gamma$ that solves the sign condition problem for the family $\mathcal{F}$ performing $\mathcal{O}((L+n)^5 \log(s))$ arithmetic operations.

The size of $\Gamma$ is bounded by $s^{\mathcal{O}(n+L)}$ and it can be constructed in expected time $s^{\mathcal{O}(n+L)}$.

*Proof.* Let $\mathcal{G} = \{G_1, ..., G_k\}$ be one of the families of generators $\mathcal{G}_P$ or $\mathcal{G}_{\mathcal{F}}$ (see page 30), and denote by $G : \mathbf{R}^n \to \mathbf{R}^k$ the associated function.

Using Meiser's result (see Theorem 2.1.3) we construct an algebraic computation tree $\Gamma_{\overline{\mathcal{F}}}$ that solves the point location problem for the family $\overline{\mathcal{F}} := \{\overline{F_1}, ..., \overline{F_s}\}$ of linear polynomials in $\mathbf{R}[Z_1, ..., Z_k]$, with query time $\mathcal{O}(k^5 \log(s))$.

Then, given a query point $x \in \mathbf{R}^n$ the sign condition satisfied by the family $\{F_1, ..., F_s\}$ at $x$ can be determined using Meiser's algorithm for the family $\overline{\mathcal{F}}$ at the point $G(x)$.

The correctness of this method follows from Remark 2.3.1. The number of arithmetic operations needed to compute $G(x)$ is bounded by $\mathcal{O}((L+n)^2)$ by Lemma A.5.1 and $k$ is bounded by $L+n+1$ by construction. Thus, the total complexity is bounded by $\mathcal{O}((L+n)^5 \log(s) + (L+n)^2) = \mathcal{O}((L+n)^5 \log(s))$. □

If we use the monomials basis $\mathcal{G}_B$ as the family of generators, we obtain the following result.

**Corollary 2.3.3.** Let $\mathcal{F} := \{F_1, ..., F_s\} \subset \mathbf{R}[X_1, ..., X_n]$ be a family of polynomials of degree bounded by $d$.

Then, there exists an algebraic computation tree $\Gamma$ that solves the sign condition problem for the family $\mathcal{F}$ performing $\mathcal{O}(\binom{d+n}{n}^5 \log(s)) = \mathcal{O}(d^{5n} \log(s))$ arithmetic operations

The size of $\Gamma$ is bounded by $s^{\mathcal{O}(d^n)}$ and it can be constructed in expected time $s^{\mathcal{O}(d^n)}$.

*Proof.* Ordering the monomials in $\mathbf{R}[X_1, ..., X_n]$ of degree at most $d$ by ascending degree, each monomial in $\mathcal{G}_B$ can be computed as a product of two preceding monomials. Hence, the family $\mathcal{G}_B$ can be computed by a division-free arithmetic circuit of non-scalar complexity $\binom{n+d}{n} - n - 1$. Thus, the result follows from last theorem. □

If we restrict our algorithms to perform arithmetic operations on integers, using Proposition A.4.3, we obtain the following result.

**Corollary 2.3.4.** Let $\mathcal{F} := \{F_1, ..., F_s\} \subset \mathbf{Q}[X_1, ..., X_n]$ be a family polynomials of total degree bounded by $d$ and logarithmic height bounded by $\tau$.

Then, there exists an algebraic computation tree $\Gamma$ that allows to determine, for any algebraic point $x \in \mathbf{R}^n$ given by a triangular Thom encoding of size $(d', \tau')$, the sign conditions satisfied by the polynomials in $\mathcal{F}$ at $x$ performing $\log(s)\overline{d}^{\mathcal{O}(n)}$ arithmetic operations between integers of logarithmic height bounded by $\overline{\tau}\overline{d}^{\mathcal{O}(n)}$, where $\overline{\tau} = \max\{\tau, \tau'\}$ and $\overline{d} = \max\{d, d'\}$. The size of the algebraic computation tree $\Gamma$ is $\mathcal{O}(\tau s^{(d+1)^n})$ and it can be constructed in expected time $\mathcal{O}(\tau s^{(d+1)^n+1})$. □

**Evaluation of First-Order Quantifier-Free Formulas.** Let us consider $\mathcal{L}$, the first-order language defined as the usual first-order language of the reals but allowing only unary predicates $t > 0$ and $t = 0$ for any term $t$ in the language, instead of the usual binary predicates $s = t$ and $s > t$ for arbitrary terms $s$ and $t$. Of course, this does not change the expressive power of the language.

Let $\varphi$ be a quantifier-free formula in this language with $n$ free variables. The truth value of $\varphi$ evaluated at $x \in \mathbf{R}^n$ depends only on the signs taken at $x$ by the polynomials involved in $\varphi$. Hence, as a consequence of the Corollary 2.3.3 we obtain the following result.

**Proposition 2.3.5.** Let $\varphi$ be a quantifier-free formula with $n$ free variables in the language $\mathcal{L}$ containing $s$ polynomials of degree bounded by $d$.

Then, there exists an algebraic computation tree $\Gamma$ that solves the membership problem for the set $\{x \in \mathbf{R}^n \mid \mathbf{R} \models \varphi(x)\}$ performing $\mathcal{O}(d^{5n}\log(s))$ arithmetic operations.

The size of $\Gamma$ is bounded by $s^{\mathcal{O}(d^n)}$ and it can be constructed in expected time $s^{\mathcal{O}(d^n)}$. □

We remark that the dialytic method performs non-scalar multiplications only to evaluate the function $G$ (defined before Remark 2.3.1) at the input point. The rest of the algorithm is free from non-scalar multiplications, *i.e.*, it performs only linear operations on these results and branches according to their signs.

We can ask now: are these complexity bounds reasonable for the simple sign condition problem? In the next section we shall prove some lower bounds related to this problem.

## 2.4 Lower Bounds for the Sign Condition Problem

We shall analyze the cost of solving the sign condition problem for different examples of families of polynomials. Each example leads to a different lower complexity bounds for the depth of any algorithm in our model solving this problem. In this way, we obtain lower bounds for the worst case complexity of the sign condition problem in terms of natural parameters of the given family.

First, we concentrate on the algebraic computation tree model and measure the multiplicative non-scalar complexity of the algorithms.

In Example 1 we construct, for any positive integers $s$ and $n$, a family of $s$ linear forms in $\mathbf{R}[X_1, ..., X_n]$ that leads to the lower bound $\Omega(n \cdot \log(s))$ for the branching complexity of any algebraic computation tree solving the sign condition problem for this family.

In Example 2 we construct, for any positive integers $s, L$ and $n$ with $s \geq n^2$, a family of non-scalar complexity $L$, containing $s+1$ polynomials in $\mathbf{R}[X_1, ..., X_n]$, that leads to the lower bound $l(L, n, s) := \max\{L, n^{\frac{\log_3(s)}{2}}\}$ for the multiplicative branching complexity of any algebraic computation tree solving the sign condition problem for this family. If we denote by $u(L, n, s) := \mathcal{O}((L+n)^5 \log(s))$ the upper bound given by the dialytic method, we obtain that

$$u(L, n, s) \leq \mathcal{O}(l(L, n, s)^6) = l(L, n, s)^{\mathcal{O}(1)}.$$

Hence, this example shows that the dialytic method is almost optimal.

The main drawback of Example 2 is that the degrees of the polynomials involved in it are exponential on $L$. Example 3 is a modification of it. We obtain, under a suitable hypothesis, the same results as in the referred example with the additional property that the polynomials in the constructed family have degree bounded by $\mathcal{O}(L^2)$.

In Examples 4 and 5, we reduce classical problems in complexity theory to the sign condition problem. In this way, using well known results from Ben-Or, Baur and Strassen we obtain lower bounds for the sign condition problem.

In Example 6, we consider a very restricted model: algebraic decision trees that can only test the sign of the polynomials in the family $\mathcal{F}$ at the input point. Our algorithms do not fit in this model since they evaluate also polynomials that do not belong to the original family. For this model, we show an $\Omega(s)$ lower bound.

Finally, in Example 7, we describe a natural restricted model where the dialytic method fits and deduce another linear lower bounds for the complexity of any algebraic computation tree solving the sign condition problem in this restricted model.

### 2.4.1 The Algebraic Model

In this section, we concentrate on the multiplicative branching complexity of any algebraic computation tree solving the sign condition problem for a given family of polynomials, *i.e.*, we take into account non-scalar multiplications and comparisons.

First we give a lower bound for the linear case, showing that Meiser's original algorithm is almost optimal.

**Example 1.** This example is a simplified linear version of the example used in [JS00] to prove a lower bound on the number of sign conditions satisfied by a family of polynomials.

For any positive integers $s$ and $n$ with $s > n$, consider $s$ linear forms $l_1, ..., l_s \in \mathbf{R}[X_1, ..., X_n]$ satisfying:

- for every subset $\{l_{i_1}, ..., l_{i_n}\}$ of $\{l_1, ..., l_s\}$ consisting of $n$ different linear forms, the linear equation system $l_{i_1}(X) = 0, ..., l_{i_n}(X) = 0$ has exactly one solution in $\mathbf{R}^n$, and

- for every subset $\{l_{i_1}, ..., l_{i_{n+1}}\}$ of $\{l_1, ..., l_s\}$ consisting of $n + 1$ different linear forms, the linear equation system $l_{i_1}(X) = 0, ..., l_{i_{n+1}}(X) = 0$ has no solutions in $\mathbf{R}^n$.

We remark that these conditions define a non-empty open set (complementary to determinantal varieties, see [JS00]) in the space $\mathbf{R}^{s\times(n+1)}$ of coefficient of the linear forms. Hence, it is legitimate to assume the existence of a family $\{l_1, ..., l_s\}$ with the stated properties. The linear forms $l_1, ..., l_s$ are said to be in *general position*. Let us denote by $C_s$ the number of sign conditions realized by this family in $\mathbf{R}^n$.

The two preceding conditions guarantee that for any two different subsets, $\{l_{i_1}, ..., l_{i_n}\}$ and $\{l_{j_1}, ..., l_{j_n}\}$, of $\{l_1, ..., l_s\}$ consisting each of $n$ linear forms, the unique solution of the linear equation system $l_{i_1}(X) = 0, ..., l_{i_n}(X) = 0$ is different from the unique solution of the linear equation system $l_{j_1}(X) = 0, ..., l_{j_n}(X) = 0$. In particular, we obtain that the family $\{l_1, ..., l_s\}$ satisfies at least $\binom{s}{n}$ different sign conditions in $\mathbf{R}^n$, *i.e.*, $C_s \geq \binom{s}{n}$.

The following proposition follows immediately and plays an important role in our lower-bound results.

**Proposition 2.4.1.** If an algebraic computation tree computes a partition $\pi$ of $\mathbf{R}^n$, then its branching complexity is at least $\log_3(\#\pi)$.

*Proof.* We recall that, in any algebraic computation tree, the only nodes with more than one immediate successor are the branching nodes, that have three immediate successors. Taking into account that a computation tree has at least one output node (*i.e.*, one leaf of the subjacent tree) for each element of $\pi$, the proof follows easily by induction. □

Suppose that $\Gamma$ is an algebraic computation tree that solves the sign condition problem for the family $\{l_1, ..., l_s\}$. Hence, it computes a partition of cardinality $C_s$. We conclude, from Proposition 2.4.1, that its branching complexity is at least $\log_3(C_s) > n(\log_3(s) - \log_3(n))$. In particular, if we take $s > n^2$, we obtain that the branching complexity of $\Gamma$ is $n\frac{\log_3(s)}{2} = \Omega(n \cdot \log(s))$.

We remark that the upper bound given by Meiser's algorithm is $\mathcal{O}(n^5 \log(s))$. Thus, the upper bound is bounded by a polynomial function of the lower bound, which is satisfactory.

Meiser's algorithm does not use non-scalar multiplications. From our lower bound, we conclude that using non-scalar multiplications would not help to improve essentially the point location algorithm in the linear case.

For the discussion of the next example we need the following technical lemma that is the key to bound the non-scalar complexity of any algebraic computation tree that solves the sign condition problem (compare with Section 1.3).

**Lemma 2.4.2.** Assume that $\{F_0, ..., F_s\}$ is a family of different irreducible polynomials defining real algebraic hypersurfaces in $\mathbf{R}^n$ and that $\Gamma$ is an algebraic computation tree solving the sign condition problem for this family.

Then, for every $i \in \mathbf{N}$, $0 \leq i \leq s$, there exists a branching node of $\Gamma$ testing the sign of a multiple of $F_i$ evaluated at the input point.

*Proof.* Let $G_1, ..., G_k$ bee the non-zero irreducible factors of the polynomials intervening in the branching nodes of $\Gamma$ and suppose, for the sake of definiteness, that $F_0$ is not associated with any of them. We remark that $G_1, ..., G_k$ and $F_0$ are irreducible.

Then, a particular form of the real Nullstellensatz for principal ideals (see Theorem 4.5.1 in [BCR98]) implies that there exists an $x \in \mathbf{R}^n$ such that $F_0(x) = 0$ and $G_i(x) \neq 0$ for $1 \leq i \leq k$. Choose $\varepsilon \in \mathbf{R}, \varepsilon > 0$ such that the polynomials $G_1, ..., G_k$ do not vanish anywhere in the ball $B = B_\varepsilon(x)$.

Therefore, the signs of $G_1, ..., G_k$ are constants in $B$. In particular, the computation path followed by $\Gamma$ for any two input points of $B$ is exactly the same.

Since $\{x \in \mathbf{R}^n \mid F_0(x) = 0\}$ is an hypersurface that cuts $B$, we conclude that there are two points $y, z \in B$ satisfying the conditions $F_0(y) = 0$ and $F_0(z) \neq 0$. Hence, $\Gamma$ does not solve the sign condition problem for the family $\{F_0, ..., F_s\}$. This contradicts the assumption that $\Gamma$ solves the sign condition problem for the family $\{F_0, ..., F_s\}$. $\square$

**Example 2.** In this example we show that, for any positive integers $n, s$ and $L$ with $s > n^2$ it is possible to construct a family $\mathcal{F}$ of $s + 1$ polynomials in $\mathbf{R}[X_1, ..., X_n]$ such that $L(\mathcal{F}) = L$ and any algebraic computation tree solving the sign condition problem for this family has multiplicative branching complexity at least $\max\{L, n \cdot \log(s)\}$.

Given positive integers $s, n$ and $L$ with $s > n^2$, consider, as in Example 1, $s$ linear forms $F_1, ..., F_s \in \mathbf{R}[X_1, ..., X_n]$ in general position and let us define the polynomial $F_0 := X_1^{2^L} - X_2$. We denote by $\mathcal{F}$ the family $\mathcal{F} := \{F_0, F_1, ..., F_s\}$ composed of these polynomials.

Suppose that $\Gamma$ is an algebraic computation tree that solves the sign condition problem for this family. Since the family $\mathcal{F}$ has at least the same number of realizable sign conditions as the family $\{F_1, ..., F_s\}$, we conclude, as in Example 1, that the branching complexity of $\Gamma$ is at least $n^{\frac{\log_3(s)}{2}}$.

Clearly, the polynomial $F_0 = X_1^{2^L} - X_2$ is irreducible and takes positive and negative values in $\mathbf{R}^n$. Hence, it defines a real algebraic hypersurface. Whence, the family $\mathcal{F}$ satisfies the assumptions of Lemma 2.4.2. Thus, $\Gamma$ evaluates a multiple of $F_0$. Since the degree of any multiple of $F_0$ is at least $2^L$, we conclude that the non-scalar complexity of $\Gamma$ is at least $L$.

Summarizing, we have that the branching complexity of $\Gamma$ is at least $n^{\frac{\log_3(s)}{2}}$ and that its non-scalar complexity is at least $L$. Hence, its multiplicative branching complexity is at least $l(L, n, s) := \max\{L, n^{\frac{\log_3(s)}{2}}\}$.

The upper bound obtained from the dialytic method is $u(L, n, s) := \mathcal{O}((L+n)^5 \log(s))$. In order to compare both bounds, we remark that $l(L, n, s) = \max\{L, n^{\frac{\log_3(s)}{2}}\} \geq \mathcal{O}(L + n \cdot \log(s))$. Hence, we obtain that

$$u(L, n, s) \leq l(L, n, s)^6 = l(L, n, s)^{\mathcal{O}(1)}.$$

This proves that the dialytic method behaves very well for the chosen parameters.

**Discussion.** Let us consider a new parameter $M \in \mathbf{N}$ defined as the maximum of the non-scalar complexity of each polynomial in the given family. We remark that while the upper bound given by the dialytic method depends intrinsically on $L$, our lower bound would depend on $M$ instead of $L$.

The family of polynomials constructed in this example has two characteristics that allowed us to derive the lower complexity bound:

1. the non-scalar complexity of some polynomials in the family is *close to* the non-scalar complexity of the whole family ($L = M^{\mathcal{O}(1)}$), and
2. the family defines enough different sign conditions ($s^{\mathcal{O}(n)}$).

What is not satisfactory about this lower bound is that the degrees of the polynomials involved are exponential on $L$. In the following example, we show that, under a suitable assumption, it is possible to modify our construction to obtain a polynomial $F_0$ whose degree is quadratic in its non-scalar complexity.

**Example 3.** This example is a modification of Example 2 and we shall use the notation introduced there. For any $d > 0$ sufficiently large, there exists, following Corollary 3.1 in [BH99], a univariate polynomial $P_d \in \mathbf{R}[X]$ of degree $d$ such that the non-scalar complexity of any multiple of $P_d$ is at least $\frac{1}{3}d^{\frac{1}{2}}$. In particular, $L(P_d) \geq \frac{1}{3}d^{\frac{1}{2}}$. On the other hand, Horner's rule give the upper bound, $L(P_d) \leq d - 1$.

We make the following (unproven) assumption: For any $d > 0$ sufficiently large and for any constant $c \in \mathbf{R}$ the non-scalar complexity of any multiple of $P_d - c$ is at least $\frac{1}{3}d^{\frac{1}{2}}$.

We assume that this conjecture is true and continue with the following construction.

Consider the polynomial $\widetilde{F_0} := P_d(X_1) - X_2 \in \mathbf{R}[X_1, ..., X_n]$. We claim that the non-scalar complexity of any non-zero multiple of $\widetilde{F_0}$ in $\mathbf{R}[X_1, ..., X_n]$ is at least $\frac{1}{3}d^{\frac{1}{2}}$. To prove the claim, consider a straight line program $(\mathsf{SLP})$ $\gamma$ of non-scalar complexity $L$ that computes a non-zero multiple $P$ of $\widetilde{F_0}$. Take $x = (x_1, ..., x_n) \in \mathbf{R}^n$ such that $P(x) \neq 0$, evaluate the $\mathsf{SLP}$ $\gamma$ in $(X, x_2, ..., x_n)$ and denote by $\gamma'$ the resulting $\mathsf{SLP}$. Then, $\gamma'$ computes a non-zero multiple of $P_d - x_2 \in \mathbf{R}[X]$ and we conclude from our conjecture that its non-scalar complexity is at least $\frac{1}{3}d^{\frac{1}{2}}$. Since the non-scalar complexity of $\gamma$ is at least that of $\gamma'$, our claim follows.

We define now the family $\widetilde{\mathcal{F}}$ as in the last example but using the polynomial $\widetilde{F_0}$ instead of $F_0$. We immediately obtain the following result.

For any three positive integers $n, s$ and $d$ with $s > n^2$ and $d > \mathcal{O}(1)$ there exists a family $\mathcal{F}$ of non-scalar complexity $L(\mathcal{F}) = d^{\mathcal{O}(1)}$ containing $s + 1$ polynomials in $\mathbf{R}[X_1, ..., X_n]$ of degree bounded by $d$ such that any algebraic computation tree solving the sign condition problem for this family has multiplicative branching complexity $\Omega(d + n \cdot \log(s))$.

**Example 4.** Let $k \in \mathbf{N}$ and let $T_{3^k} \in \mathbf{R}[X]$ be the $3^k$-th Chebyshev polynomial. We observe that $T_{3^k}$ has $3^k$ distinct real roots and that $T_{3^k}$ can be evaluated using $3k$ non-scalar multiplications (see [BE95]).

Let $s := n$, $L := 3kn = \mathcal{O}(kn)$ and, for $1 \leq i \leq s$, define $F_i := T_{3^k}(X_i)$. It is easy to see that $F_1, ..., F_s$ may be evaluated by a division-free arithmetic circuit over $\mathbf{R}$ of non-scalar size $L$. Strassen's degree method (see Corollary 8.36 in [BCS97]) implies now that the complexity bound $\mathcal{O}(kn)$ is asymptotically optimal for the evaluation of $F_1, ..., F_s$.

Let us suppose that $\Gamma$ is an algebraic computation tree that solves the sign condition problem for $F_1, ..., F_s$ performing at most $N$ non-scalar multiplications and decisions. This algorithm may be applied to solve the membership problem of $V := \{x \in \mathbf{R}^n \mid F_i(x) = 0 \quad \text{for} \quad 1 \leq i \leq s\}$ using $N$ non-scalar multiplications and decisions. We observe that $V$ is a zero dimensional set consisting of $3^{kn}$ points. Thus, $V$ has $3^{kn}$ connected components. Applying Ben-Or's method to this situation (see Theorem 11.9 in [BCS97]), we deduce now that the solution of the membership problem of $V$ requires at least $\Omega(\log(\#V)) = \Omega(kn)$ non-scalar operations and decisions. Thus, we have $N = \Omega(kn) = \Omega(L)$.

Using the Dialytic method, we obtain the following upper bound $N = \mathcal{O}((L+n)^5 \log(s)) = \mathcal{O}(L^5 \log(n)) = L^{\mathcal{O}(1)}$.

**Example 5.** For any positive integer $m$, let $n := 2m, s = m$ and consider the variables $X_1, ..., X_m, Y_1, ..., Y_m$. For $1 \leq j \leq m$ let $F_j := (X_j - Y_1) \cdots (X_j - Y_m)$. Suppose there is given a sign condition determination algorithm for $F_1, ..., F_s$ which, for any input point $(x, y) \in \mathbf{R}^m \times \mathbf{R}^m = \mathbf{R}^{2m}$, has at most $N$ non-scalar multiplications and branchings. Then, we are able to decide within the same complexity whether for a given point $(x, y) \in \mathbf{R}^{2m}$ $F_1(x, y) \neq 0, ..., F_s(x, y) \neq 0$, *i.e.*, whether $\Pi_{j=1}^{s} F_j(x, y) = \Pi_{1 \leq i,j \leq s}(x_i - y_j) \neq 0$ holds.

This is equivalent to deciding whether $\{x_1, ..., x_m\} \cap \{y_1, ..., y_m\} \neq \emptyset$. Following Ben-Or (see [BO83] or Theorem 11.9 and Corollary 11.11 in [BCS97]), this implies $N = \Omega(n \log(n))$. On the other hand, from Corollary 8.13 in [BCS97] (originally by Baur and Strassen [BS83]) $F_1, ..., F_s$ may be evaluated by a division-free arithmetic circuit of size $L := \mathcal{O}(n \log(n))$.

Thus, we have a lower bound of $\Omega(L)$ and an upper bound of $L^{\mathcal{O}(1)} \log(n) = L^{\mathcal{O}(1)} \log(L) = L^{\mathcal{O}(1)}$.

### 2.4.2 Restricted Models

In this section, we consider two different restricted models.

**Algebraic decision tree model.** The first model we consider is the most restricted model where the sign condition problem for a family $\{F_1, ..., F_s\}$ of polynomials con be solved. In this model, an algorithm is an *algebraic decision tree* (that should not be confused with the *algebraic computation trees* of the previous examples, see Appendix A) whose tests can only be based on the sign satisfied by some polynomial in $\{F_1, ..., F_s\}$ evaluated at the input point.

**Example 6.** In this example, we construct a family of linear polynomials in $\mathbf{R}[X_1, X_2]$ and show that any algebraic decision tree in our restricted model that solves the sign condition problem for this family must have depth $s$.

Consider the unit circle in the plane $S^1 \subset \mathbf{R}^2$ and $s$ different points on it, $p_1, ..., p_s \in S^1$. For $1 \leq i \leq s$, let us denote by $F_i$ the linear equation

$p_i \cdot (x_1, x_2) - 1$ representing the tangent line to $S^1$ passing through $p_i$. We remark that inside the unit circle all these equations take negative values. Also, for $1 \leq i \leq s$, all these linear polynomials take negative values at $p_i$ except $F_i$ which is zero.

Suppose now that $\Gamma$ is an algebraic decision tree satisfying that any of its decisions (branchings) is based on the sign of some polynomial in the family $F_1, .., F_s$ evaluated at the input point. We claim that for any input whose computation path follows the *negative sign* branch of each test in this path, all the polynomials in the family $F_1, ..., F_s$ must be evaluated before the sign taken by all of them at the input is completely determined.

To prove the claim, suppose that $F_1, ..., F_{s-1}$ are evaluated but not $F_s$. Consider the point $p_s$; as remarked before, $F_1, ..., F_{s-1}$ take negative values at $p_s$ and $F_s(p_s) = 0$. Then, there exists a small open ball, $B_\varepsilon(p_s) \subset \mathbf{R}^2$ such that $F_1, ..., F_{s-1}$ take negative values inside this ball. Since $F_s = 0$ describes a line, we conclude that there exist two different points, namely $x$ and $y$, in the ball $B_\varepsilon(p_s)$ such that $F_s(x) > 0$ and $F_s(y) < 0$. Hence, the sign of an input point cannot be determined evaluating a proper subset of $\{F_1, ..., F_s\}$ at this point. Thus, the depth of $\Gamma$ is at least $s$.

**Discussion.** This example can be easily generalized to higher dimensions and, with some more work, to polynomials of any given non-scalar complexity.

This example shows that other polynomials than $F_1, ..., F_s$ must be evaluated in order to obtain an upper bound that depends logarithmically on $s$. Inspecting Meiser's algorithm and the dialytic method, we see that it is enough to admit to test the sign of linear combinations of the polynomials evaluated in previous tests. This motivates the following model.

**Oracle model.** Let us assume given a family $\mathcal{F} := \{F_1, ..., F_s\}$ of polynomials in $\mathbf{R}[X_1, ..., X_n]$ and an oracle that can evaluate any polynomial in this family.

An algorithm solving the sign condition problem for the family $\mathcal{F}$ in the oracle model is an algebraic computation tree that computes the partition $\mathcal{S}(\mathcal{P})$ of $\mathbf{R}^n$ performing, beside the branchings, only the following kinds of computations:

- constants from $\mathbf{R}$;
- additions;
- multiplication (scala and non-scalar);
- evaluation of a polynomial $F_i \in \mathcal{F}$ at the input (oracle call);

Additions and multiplications can be only performed using the results of previous computations; the only computation nodes that have access to the input $x \in \mathbf{R}^n$ are the nodes corresponding to oracle calls. Our complexity model is expressed in terms of the number of oracle calls. □

The dialytic method fits in this model. To see this, consider the set $\mathcal{G}_{\mathcal{F}}$ of generators for the dialytic method (see Section 2.3) and assume the evaluation of these generators as oracle calls. Its complexity is $L + n + 1$ oracle calls, where $L$ is the non-scalar complexity of the family $\mathcal{F}$. Moreover, besides the evaluations of polynomials in the family $\mathcal{F}$, all the operations performed by this method are $\mathbf{R}$-linear, *i.e.*, no non-scalar multiplications are used. We remark that the input $x \in \mathbf{R}^n$ can be reconstructed (using linear algebra) from the result of the evaluation of the polynomials in $\mathcal{G}_{\mathcal{F}}$.

This model fits well in the structure of the known sign condition which evaluate $F_1, ..., F_s$ as if were given by an oracle. For such algorithms, the circuit representation of $F_1, ..., F_s$ is natural.

In the following example we construct, for any $s$ and any $n$ that divides $s$, a family $\mathcal{F} \subset \mathbf{R}[X_1, ..., X_n]$ containing $s$ $\mathbf{R}$-linearly independent polynomials and show that any algorithm in the oracle model that solves the sign condition problem must evaluate all the $s$ polynomials for some input.

The non-scalar complexity of the family $\mathcal{F}$ constructed in the next example is $L = s$. This explains the apparent incompatibility between the upper bound given by the dialytic method (it uses $L + n + 1$ oracle calls) and the lower bound $\Omega(s)$ for the number of oracle calls deduced in the following paragraph.

**Example 7.** Let $n, s$ and $d$ be a positive integers such that $s = dn$. Choose $d + 1$ different real numbers $\tau_0, ..., \tau_d$ and let us consider, for $1 \leq j \leq d$, the polynomials

$$F_j := \prod_{k=0...d, k \neq j} (X - \tau_k)$$

and, for $1 \leq i \leq n$, $F_j^{(i)} := F_j(X_i) \in \mathbf{R}[X_1, ..., X_n]$. We shall denote by $\mathcal{F}$ the family

$$\mathcal{F} := \{F_j^{(i)} \mid 1 \leq i \leq d \text{ and } 1 \leq j \leq n\}.$$

Let us suppose that $\Gamma$ is an algorithm in the oracle model that solves the sign condition problem for the family $\mathcal{F}$. We claim that on input $x := (\tau_0, ..., \tau_0) \in \mathbf{R}^n$, $\Gamma$ evaluates all the polynomials in $\mathcal{F}$.

Suppose that the claim does not hold. We further suppose, without loss of generality, that on input $x$ the algebraic computation tree $\Gamma$ does not evaluate $F_1^{(1)}$.

Consider the point $x' := (\tau_1, \tau_0, ..., \tau_0) \in \mathbf{R}^n$ and remark that $F_1^{(1)}$ is the only polynomial in $\mathcal{F}$ that does not vanish at $x'$, and that all the other polynomials in $\mathcal{F}$ vanish at $x$.

We recall that in this model, the tests performed at a branching node in a computation path in $\Gamma$ are sign tests of polynomials evaluated at the values taken by polynomials previously evaluated in that computation path. Since the results of the evaluation of all the polynomials in $\mathcal{F} \setminus \{F_1^{(1)}\}$ at $x$ and $x'$ are equal, we conclude that both $x$ and $x'$ determine the same computation path in $\Gamma$. In particular the output is the same for both inputs. Since they do not satisfy the same $\mathcal{F}$-sign condition, we obtain a contradiction. This contradiction proves our claim.

In this way, we obtain that any algorithm in the oracle model, that solves the sign condition problem for the family $\mathcal{F}$, must evaluate all the polynomials in $\mathcal{F}$ for some input. Since there are $s = dn$ different polynomials, we obtain that $s$ is a lower bound for the number of oracle calls in this model, in the worst case.

We recall that the dialytic method fits in the oracle model and has un upper bound that grows only logarithmically with $s$. This is not a contradiction. Since all the polynomials in the family $\mathcal{F}$ are linearly independent, its non-scalar complexity is at least $\Omega(s) = \Omega(nd)$. On the other hand, since this family can be computed from the monomials $X_1, X_1^2, ..., X_1^{d+1}, ..., X_n, X_n^2, ..., X_n^{d+1}$ performing linear operations, we conclude that its non-scalar complexity is upper bounded by $L := nd = s$. Thus, the non-scalar complexity of this family equals its cardinality, what explains the origin of our lower bound.

# 3

# The Point Location Problem for a Family of Polynomials

**Abstract.** In this chapter, we discuss different data structures that can be used to solve the point location problem for a given family of polynomials. This problem asks to determine, not only the sign condition satisfied by a family of polynomials at a query point, but also the connected component of the realization of this sign condition containing the query point. After showing how to adapt the dialytic method to this problem, we introduce, in Section 3.3, a method based on an adapted Cylindrical Algebraic Decomposition of the space that solves the point location problem for any given family. In Section 3.4, we discuss the case of polynomials with integer coefficients given in dense bit representation introducing a method that, based on diophantine geometry, solves the point location problem for generic families of polynomials. At the end, we include a brief discussion of the local ray shooting problem.

## 3.1 Introduction

In this chapter, we address the point location problem that was introduced in the previous chapter. We use the notions and notations introduced there. We assume that the polynomials are represented in dense (bit or arithmetic) form and we measure the number of arithmetic operations performed by the algorithms. We first discuss how the dialytic method can be used to solve the point location problem. Then, we introduce two other data structures to solve this problem.

**Relation with Constraint Databases.** Using the framework of Constraint Databases [KLP00, Rev02] with polynomial constraints, we can to deal with geometric figures in the affine space $\mathbf{R}^n$ containing infinitely many points (semi-algebraic sets) which are finitely represented as boolean combinations of polynomial equalities and inequalities. The data model proposed by the literature on Constraint Databases to describe geometric figures in $\mathbf{R}^n$ is based on quantifier-free first-order formulas over the reals. This might be very inefficient. As remarked in [HK04], explicitly giving disjunctive normal and using dense or sparse encoding of polynomials as data structures turns out to be inconvenient for many applications. Another example that shows the complexity problems that faces this traditional vision is given by the membership problem: direct and brutally evaluating a first-order formula at a query point may result too expensive, and it could be evaluated otherwise (see, for instance, Proposition 2.3.5).

Inspired by the indexing techniques used in Relational Databases, we study some queries that arise naturally in the context of spatial databases and propose new data structures specifically designed to answer them.

The philosophy of our work is that each query requires, to be answered efficiently, data structure and/or an evaluation algorithm specifically designed for it. This contrasts with the original viewpoint of Constraint Databases, that is more coarse. There, only simple data structures are used and it is assumed that first-order queries are evaluated using a general purpose quantifier-elimination method.

The algebraic computation trees constructed by the dialytic method presented in the previous chapter can be seen as data structures stored in a database used to answer the sign condition query efficiently for a fixed family of polynomials. In the following pages, we discuss three (this and two other) data structures that can be used to solve the point location problem for a family of polynomials with a complexity depending only logarithmically on $s$. A precise algebraic model of computation allowing the use of precomputed data structes is missing in the literature and further research is needed to give a satisfactory definition of such a computational model.

In Section 3.3, we introduce a method based on an Adapted Cylindrical Algebraic Decomposition of the space that solves the point location problem for any fixed family. This method, being based on projections and a recursion on the dimension of the ambient space, requires doubly-exponential query time.

In Section 3.4, we discuss the case of polynomials with integer coefficients given in dense bit representation, introducing a method that, based on diophantine geometry, solves the point location problem for generic families of polynomials. Assuming that the query space is $[0, 1]^n$ and that the given family of polynomials $\mathcal{F}$ is generic, we construct a uniform grid over $[0,!]^n$ such that each small hypercube determined by this grid is only cut by, at most, $n$ polynomials in $\mathcal{F}$.

In Section 3.5, we discuss how this last data structure can be used to solve the local ray shooting problem. Finally, we shall discuss the advantages and drawbacks of each of them in Section 3.6.

## 3.2 The dialytic method for Point Location

We concentrate on the arithmetic bit model, and show how the dialytic method can be used to solve the point location problem for a given family of polynomials.

We recall that the sign condition problem for a family $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$ of polynomials can be solved in this model using Corollary 2.3.4.

To solve the point location problem for a family of polynomials, we use the following proposition, that is an immediate consequence of Theorem 16.18 in [BPR06] (see [HRS90a, GHR$^+$90, HRS94b, CGV91, GV92, HRS94a, Can93, GR93, BPR99] for the historical development of this result).

**Proposition 3.2.1.** Let $\mathcal{P}$ be a family of $s$ polynomials in $\mathbf{R}[X_1, ..., X_n]$ of degree bounded by $d$. Then, there exists a family $\widetilde{\mathcal{P}}$ containing $s^n d^{\mathcal{O}(n^4)}$ polynomials in $\mathbf{R}[X_1, ..., X_n]$ of degree bounded by $d^{\mathcal{O}(n^3)}$, such that the partition $\mathcal{S}(\widetilde{\mathcal{P}})$ of $\mathbf{R}^n$ induced by the realization of the sign conditions on the family $\widetilde{\mathcal{P}}$ is finer than the partition $\mathcal{A}(\mathcal{P})$ induced by the connected components of the realization of the sign conditions on the family $\mathcal{P}$.

Moreover, there exists an algorithm that, on input $\mathcal{P}$, computes a family $\widetilde{\mathcal{P}}$ with the stated properties in time bounded by $s^{n+1}d^{\mathcal{O}(n^4)}$; if the input polynomials have integer coefficients whose bitsize is bounded by $\tau$, the bitsize of the coefficients of the output is $\tau d^{\mathcal{O}(n^3)}$. □

The last proposition implies that a solution of the sing condition problem for the family $\widetilde{\mathcal{P}}$ leads to a solution of the point location problem for the family $\mathcal{P}$. Combining the Corollary 2.3.4 with the last proposition, and identifying different outputs corresponding to a same face of the arrangement $\mathcal{A}(\mathcal{P})$, we obtain following corollary.

**Corollary 3.2.2.** Let $\mathcal{P} := \{P_1, ..., P_s\}$ be a family polynomials in $\mathbf{R}[X_1, ..., X_n]$ of total degree bounded by $d$ and logarithmic height bounded by $\tau$. Then, there exists a data structure of size $\tau s^{d^{\mathcal{O}(n^4)}}$ that allows to solve the point location problem for the family $\mathcal{P}$. For any $x \in \mathbf{R}^n$ given by a triangular Thom encoding of size $(d', \tau')$, the point location query at $x$ is answered performing $\log(s)\overline{d}^{\mathcal{O}(n^4)}$ arithmetic operations between integers of logarithmic height bounded by $\overline{\tau}\overline{d}^{\mathcal{O}(n)}$, where $\overline{\tau} = \max\{\tau, \tau'\}$ and $\overline{d} = \max\{d^{\mathcal{O}(n^3)}, d'\}$. The data structure can be constructed in expected time $\tau s^{d^{\mathcal{O}(n^4)}}$. □

## 3.3 Point Location using Cylindrical Algebraic Decomposition

Cylindrical Algebraic Decomposition (CAD) was introduced by Collins [Col75] as a method to eliminate quantifiers in formulas of the elementary first-order theory of the reals. CAD is a powerful tool that allows, among other things, to decide the truth a first-order sentence in this theory and to compute stratifications of a semi-algebraic set. Its main drawback is its complexity. The size of a CAD adapted to a family $\{P_1, ..., P_s\}$ of polynomials of degree $d$ in $\mathbf{R}[X_1, ..., X_n]$ is $(sd)^{\mathcal{O}(1)^n}$.

In this section, we show how CAD can be used to solve the point location problem. More than thirty years after its introduction by Collins, CAD has become a standard technique. We base our results on the CAD construction given in the textbook [BPR06], that are, in its turn, based on the cited original work of Collins.

First, we introduce some notation. If $P \in \mathbf{R}[X_1, ..., X_n]$ and $x \in \mathbf{R}^{n-1}$, we denote by $P[x]$ the polynomial $P(x, X_n) \in \mathbf{R}[X_n]$. Given a family $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$, we denote by $\mathcal{P}[x]$ the family $\{P[x] \mid P \in \mathcal{P}\}$. A subset $T$ of $\mathbf{R}^n$ is called $\mathcal{P}$-*invariant* if every polynomial $P \in \mathcal{P}$ has constant sign on $T$.

If $P \in \mathbf{R}[X]$ has $\xi_1 < ... < \xi_l$ as its different real roots, we call $\xi_i$ the $i^{th}$ real root of $P$, for $1 \leq i \leq l$. The $i^{th}$ real root of a family $\{P_1, ..., P_s\} \in \mathbf{R}[X]$ is defined as the $i^{th}$ real root of $\Pi_{i=1}^{s} P_i$.

For any $0 \leq i < j \in \mathbf{N}$ we denote by $\pi_i : \mathbf{R}^j \to \mathbf{R}^i$ the projection onto the first $i$ coordinates, where $\mathbf{R}^0$ is defined as a singleton.

### 3.3.1 An Elimination Step and its Data Structures

**Definition 3.3.1.** Let $P$ be a polynomial in $\mathbf{R}[X_1, ..., X_n]$ and let $S$ be a semi-algebraic subset of $\mathbf{R}^{n-1}$. We say that the real roots of $P$ are *delineable* on $S$ and that $\xi_1, ..., \xi_l$ *delineate* the roots of $P$ on $S$ if there are continuous semi-algebraic functions $\xi_1 < ... < \xi_l : S \to \mathbf{R}$ such that

- for every $x \in S$, the set $\{\xi_1(x), ..., \xi_l(x)\}$ is the set of all different real roots of all non-zero polynomials $P(x, X_n)$, and
- for $1 \leq i \leq l$, the multiplicity of the root $\xi_i(x)$ of $P(x, X_n)$ is constant for $x \in S$

If $\mathcal{P} := \{P_1, ..., P_s\}$ is a family of polynomials in $\mathbf{R}[X_1, ..., X_n]$ we say that $\xi_1, ..., \xi_l$ delineate the roots of $\mathcal{P}$ on $S$ if they delineate the roots of $\Pi_{i=1}^{s} P_i$ on $S$.

We remark that, if they exist, the functions that delineate the roots of the family $\mathcal{P}$ on $S$ are uniquely determined.

Given $P, Q \in \mathbf{R}[X_1, ..., X_n]$ we denote by $\mathrm{sr}_j(P, Q)$ the $j^{th}$ signed subresultant coefficient of $P$ and $Q$ and by $\mathrm{lcof}(P)$ the leading coefficient of $P$ seen as a polynomial in $\mathbf{R}[X_1, ..., X_{n-1}][X_n]$. If $\mathcal{P}$ is a family of polynomials, we denote by $\mathrm{Tru}(\mathcal{P})$ the set of truncations of the polynomials in $\mathcal{P}$ considered as polynomials in the variable $X_n$. See [BPR06] for precise definitions.

Following [BPR06], we denote by $\mathrm{Elim}_{X_n}(\mathcal{P}) \subset \mathbf{R}[X_1, ..., X_{n-1}]$ the set of polynomials defined as follows:

- If $R \in \mathrm{Tru}(\mathcal{P})$, $\deg_{X_n}(R) \geq 2, \mathrm{Elim}_{X_n}(\mathcal{P})$ contains all the nons-constant $\mathrm{sr}_j(R, \partial R/\partial X_n)$, $j = 0, ..., \deg_{X_n}(R) - 2$.
- If $R \in \mathrm{Tru}(\mathcal{P})$, $S \in \mathrm{Tru}(\mathcal{P})$, $\deg_{X_n}(R) \geq 2, \mathrm{Elim}_{X_n}(\mathcal{P})$ contains all $\mathrm{sr}_j(R, S)$ which are not in $\mathbf{R}$, $j = 0, ..., min(\deg_{X_n}(R), \deg_{X_n}(S)) - 1$.
- If $R \in \mathrm{Tru}(\mathcal{P})$ and $\mathrm{lcof}(R)$ is not in $\mathbf{R}$, $\mathrm{Elim}_{X_n}(\mathcal{P})$ contains $\mathrm{lcof}(R)$.

From the Theorems 5.15 and 5.16 in [BPR06], we immediately obtain the following result.

**Theorem 3.3.2.** Let $\mathcal{P}$ be a set of polynomials in $\mathbf{R}[X_1, ..., X_n]$ and let $S$ be a connected semi-algebraic subset of $\mathbf{R}^{n-1}$. If $S$ is $\mathrm{Elim}_{X_n}(\mathcal{P})$–invariant then the real roots of $\mathcal{P}$ are delineable on $S$. □

**Definition 3.3.3.** Let $\mathcal{P} = \{P_1, ..., P_s\} \subset \mathbf{R}[X_1, ..., X_n]$ be a family of polynomials, let $S$ be a semi-algebraic subset of $\mathbf{R}^{n-1}$ such that the family $\mathcal{P}[x]$ has $l$ real roots for any $x \in S$. A `root selection table for` $\mathcal{P}$ `over` $S$ contains $l$ records. The $i^{th}$ record, for $1 \leq i \leq l$, is composed of a pair $(p_i, q_i)$ such that for all $x \in S$, the $i^{th}$ real root of $\mathcal{P}[x]$ coincides with the $q_i$-th real root of $P_{p_i}[x]$.

If $n = 1$, we call any `root selection table for` $\mathcal{P}$ `over` $\mathbf{R}^0$ simply a `root selection table for` $\mathcal{P}$.

We immediately obtain the following result.

**Corollary 3.3.4.** Let $\mathcal{P}$ be a set of polynomials in $\mathbf{R}[X_1, ..., X_n]$ and let $S$ be a connected semi-algebraic subset of $\mathbf{R}^{n-1}$. If $S$ is $\mathrm{Elim}_{X_n}(\mathcal{P})$–invariant then there exist a `root selection table for` $\mathcal{P}$ `over` $S$.

*Proof.* Choose some $x \in S$. Since any real root of $\mathcal{P}[x]$ is a real root of $P[x]$ for some $P \in \mathcal{P}$, there exists a `root selection table for` $\mathcal{P}[x]$. Let us denote it by $\mathcal{T}^{Roots}_{\mathcal{P},x}$.

Suppose that for some $i, j \in \mathbf{N}$ and some $P \in \mathcal{P}$, the $i^{th}$ real root of $\mathcal{P}[x]$ coincides with the $j^{th}$ real root of $P[x]$. By Theorem 3.3.2, the real roots of $\mathcal{P}$ and of every $P \in \mathcal{P}$ are delineable on $S$. Hence, by a simple continuity argument, the $i^{th}$ real root of $\mathcal{P}[x']$ coincides with the $j^{th}$ real root of $P[x']$, for any $x' \in S$. Thus, $\mathcal{T}^{Roots}_{\mathcal{P},x}$ is a `root selection table for` $\mathcal{P}$ `over` $S$. □

**Definition 3.3.5.** Let $\mathcal{P}$ be a finite family of polynomials in $\mathbf{R}[X_1, ..., X_n]$, let $S$ be a connected semi-algebraic subset of $\mathbf{R}^{n-1}$ and suppose that $\xi_1 < ... < \xi_l$ delineate the roots of $\mathcal{P}$ on $S$. A $\mathcal{P}$*-stratum* over $S$ is:

- either the graph of one of the functions $\xi_{S,j}$, for $j = 1, ..., l_S$,
- or a band of the cylinder bounded from bellow and from above by the graphs of the functions $\xi_{S,j}$ and $\xi_{S,j+1}$, for $j = 0, ..., l_S$, where we take $\xi_{S,0} = -\infty$ and $\xi_{S,l_S+1} = +\infty$.

The $\mathcal{P}$-strata over $S$ are numbered in ascending order: $1, 2, ..., 2l + 1$. In this way, the stratum corresponding to the graph of the function $\xi_{S,j}$ has number $2j$, for $j = 1, ..., l_S$. The strata corresponding to the graph of a function are called *graph strata.* The remaining ones are called *band strata.*

If the polynomials in $\mathcal{P}$ are univariate, the $\mathcal{P}$-strata over $\mathbf{R}^0$ are points and open intervals. We refer to them simply as the $\mathcal{P}$-strata.

Remark that the $\mathcal{P}$-strata over $S$ form a partition of $S \times \mathbf{R}$ and that if $\mathcal{P}' \supset \mathcal{P}$ then the $\mathcal{P}'$-strata over $S$ constitute a refinement of the partition given by the $\mathcal{P}$-strata over $S$.

**Remark 3.3.6.** Let $C$ be a $\mathcal{P}$-stratum over $S$. By definition, no polynomial in $\mathcal{P}$ changes its sign on $C$. Hence, to every $\mathcal{P}$-stratum over $S$ there corresponds a single $\mathcal{P}$-sign-condition. In other words, $C$ is $\mathcal{P}$–invariant. In particular, since any stratum is connected, $C$ is contained in a single face of the arrangement $\mathcal{A}(\mathcal{P})$.

On the other hand, different $\mathcal{P}$-strata over $S$ can realize the same $\mathcal{P}$-sign-condition. It is clear that if two strata realize the same sign condition, then they are both band strata or graph strata.

If $P$ is a polynomial in $\mathbf{R}[X_1, ..., X_n]$ we denote by $\mathcal{D}(P)$ the set consisting of $P$ and its successive non-zero derivatives with respect to $X_n$. For $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$ we define $\mathcal{D}(\mathcal{P}) := \cup_{P \in \mathcal{P}} \mathcal{D}(P)$. The following Proposition is a generalized form of Thom's Lemma (see Lemma 5.32 in [BPR06]). It ensures that, adding the corresponding derivatives, each sign condition is realized on, at most, one stratum.

**Proposition 3.3.7.** Let $P \in \mathbf{R}[X_1, ..., X_n]$ and let $S$ be a connected semi-algebraic subset of $\mathbf{R}^{n-1}$. If $S$ is $\mathrm{Elim}_{X_n}(\mathcal{D}(P))$–invariant, then each $\mathcal{D}(P)$-sign-condition realizable in $S \times \mathbf{R}$ is realized on exactly one $\mathcal{D}(P)$-stratum over $S$.

*Proof.* By Theorem 3.3.2, the real roots of $\mathcal{D}(P)$ are delineable on $S$. Hence, the concept of $\mathcal{D}(P)$-stratum over $S$ is well defined. By Remark 3.3.6, each $\mathcal{D}(P)$-stratum over $S$ is $\mathcal{D}(P)$–invariant, *i.e.*, it realizes a single $\mathcal{D}(P)$-sign condition.

Let $\sigma$ be a $\mathcal{D}(P)$-sign-condition realizable in $S \times \mathbf{R}$. Let $x' \in S$ and consider the set $R := \{x \in \mathbf{R} \mid \mathrm{sgn}(\mathcal{D}(P), (x', x)) = \sigma\}$. Remark that $R$ intersects every stratum realizing the sign condition $\sigma$. By Thom's Lemma, $R$ is a point or an open interval. If $R$ is a point, we conclude that there is only one stratum realizing the sign condition $\sigma$.

Let us assume that $R$ is an open interval. Then, the strata that intersect $R$ are band strata. Thus, $R$ is the union of disjoint open intervals corresponding to the different band strata realizing the sign condition $\sigma$. Since $R$ is connected, we conclude that there is a single $\mathcal{D}(P)$-stratum over $S$ realizing the sign condition $\sigma$. This completes the proof. □

We remark that any $P$-stratum over $S$ is the union of some $\mathcal{D}(P)$-strata over $S$.

**Definition 3.3.8.** If $S$ is a connected semi-algebraic subset of $\mathbf{R}^{n-1}$ which is $\mathrm{Elim}_{X_n}(\mathcal{D}(P))$–invariant, we define the `Thom encoding table of` $P$ `over` $S$ as the table that associates to any $\mathcal{D}(P)$-sign-condition realizable in $S \times \mathbf{R}$ the number of the unique $P$-stratum over $S$ realizing it. The existence of this table is guaranteed by Proposition 3.3.7. We require the table to be lexicographically ordered by $\mathcal{D}(P)$-sign-condition and denote it by $\mathcal{T}_{P,S}^{Thom}$.

With this last definition we have completed our basic data structures. We present now the algorithms used to query these tables. In order to specify, in the description of our algorithms, the assumptions concerning preconstructed tables or the parameters of an algorithm, and to differentiate these from the inputs of the algorithm, we use the label **context**.

**Algorithm 3.3.9** (`Thom encoding query over` $S$)**.**

**Context:** Let $P \in \mathbf{R}[X_1, ..., X_n]$ be a polynomial of degree $d$, let $S$ be a connected semi-algebraic subset of $\mathbf{R}^{n-1}$ which is $\mathrm{Elim}_{X_n}(\mathcal{D}(P))$–invariant and assume precomputed the `Thom encoding table of` $P$ `over` $S$, $\mathcal{T}_{P,S}^{Thom}$.
**Input:** A $\mathcal{D}(P)$-sign condition $\sigma$, realizable in $S \times \mathbf{R}$.
**Output:** The number of the $P$-stratum over $S$ realizing $\sigma$.

**Procedure:** Perform a binary search on the (lexicographically ordered) `Thom encoding table` and return the stratum number corresponding to $\sigma$.
**Complexity Analysis:** Since $\mathcal{T}_{P,S}^{Thom}$ is lexicographically ordered by sign condition, the search time is logarithmic in the length of the table. Since the number of real roots of the family $\mathcal{D}(P)$ is bounded by $d(d+1)/2$, the number of $\mathcal{D}(P)$-strata (*i.e.*, the number of rows in the table) is at most $d^2+d+1$. Hence, the total complexity is $\mathcal{O}(log(d))$.
**Proof of Correctness:** Follows from the Proposition 3.3.7 and the definition of the `Thom encoding table of` $P$ `over` $S$. □

**Algorithm 3.3.10** (`Relative position query over` $S$).

**Context:** Let $P \in \mathbf{R}[X_1, ..., X_n]$ be a polynomial of degree $d$ and logarithmic height $\tau$, let $S$ be a connected semi-algebraic subset of $\mathbf{R}^{n-1}$ which is $\mathrm{Elim}_{X_n}(\mathcal{D}(P))$–invariant and assume precomputed the `Thom encoding table of` $P$ `over` $S$, $\mathcal{T}_{P,S}^{Thom}$.
**Input:** A triangular Thom encoding for $x \in \mathbf{R}^n$ of size $(d', \tau')$ and $i \in \mathbf{N}$ such that $P$ has at least $i$ different real roots.
**Output:** *Bellow*, *Inside* or *Above* according to the relative position of $x$ with respect to the $i^{th}$ real root of $P$.
**Procedure:** Use the `sign determination algorithm` (see Proposition A.4.3) to obtain the sign condition satisfied by $\mathcal{D}(P)$ at $x$. A `Thom encoding query over` $S$ then gives the number, *pos*, of the $P$-stratum containing $x$. If $pos > 2i$ return *Above*, if $pos < 2i$ return *Bellow*, otherwise return *Inside*.
**Complexity Analysis:** Denoting by $\overline{d}$ the maximum of $d$ and $d'$ and by $\overline{\tau}$ the maximum of $\tau$ and $\tau'$, the first step requires $\overline{d}^{\mathcal{O}(n)}$ arithmetic operations between integers of logarithmic height bounded by $\overline{\tau}\overline{d}^{\mathcal{O}(n)}$ and dominates the total complexity.
**Proof of Correctness:** Follows from Proposition A.4.3, the correctness of Algorithm 3.3.9 (`Thom encoding query over` $S$) and the numbering of the $\mathcal{P}$-strata over $S$ given in Definition 3.3.5. □

**Algorithm 3.3.11** ($\mathcal{P}$`-stratum-number query over` $S$).

**Context:** Let $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$ be a family of $s$ polynomials of degree bounded by $d$ and logarithmic height $\tau$, let $S$ be a connected semi-algebraic subset of $\mathbf{R}^{n-1}$ which is $\mathrm{Elim}_{X_n}(\mathcal{P})$–invariant and $\mathrm{Elim}_{X_n}(\mathcal{D}(P))$–invariant for every $P \in \mathcal{P}$.
Assume precomputed the `Thom encoding table of` $P$ `over` $S$, $\mathcal{T}_{P,S}^{Thom}$, for each $P \in \mathcal{P}$.
Assume also precomputed $\mathcal{T}_{\mathcal{P},S}^{Roots}$, a `root selection table for` $\mathcal{P}$ `over` $S$ (see Definition 3.3.3) and let us denote by $p_i$ and $q_i$ the two elements of the $i^{th}$

record in this table.
**Input:** A triangular Thom encoding for $x \in S \times \mathbf{R}$ of size $(d', \tau')$.
**Output:** The number of the $\mathcal{P}$-stratum over $S$ containing $x$.
**Procedure:** Let $g$ be the length of the table $\mathcal{T}^{Roots}_{\mathcal{P},S}$, *i.e.*, suppose that the polynomials in $\mathcal{P}$ have $g$ different real roots over $S$.
Perform a binary search on the ordered set of $\mathcal{P}$-strata over $S$ to determine the number of the $\mathcal{P}$-stratum containing $x$ and output the result. More precisely:

```
Let $l := 1, u := 2g + 1$.
While $l \neq u$
        Let $i := \ulcorner \frac{l+u}{4} \urcorner$
        Perform an Relative position query over $S$ for $x$ with respect to the $q_i$-th real root of the polynomial $P_{p_i} \in \mathcal{P}$.
        If the result is $Inside$ return $2i$ ($x$ belongs to the stratum $C_{2i}$).
        If the result is $Bellow$ set $u := 2i - 1$.
        If the result is $Above$ set $l := 2i + 1$.
Loop.
Return $l$ ($x$ belong to the unique remaining stratum, $C_l$).
```

**Complexity Analysis:** The number of comparisons in the binary search is of order $\mathcal{O}(log(g)) = \mathcal{O}(log(ds))$. Each use of the `Relative position query over` $S$ requires $\overline{d}^{\mathcal{O}(n)}$ arithmetic operations, where $\overline{d}$ is defined as the maximum of $d$ and $d'$. Thus, the total complexity is $log(s)\overline{d}^{\mathcal{O}(n)}$. The bitsize of the integers involved is bounded by $\overline{\tau}\overline{d}^{\mathcal{O}(n)}$, where $\overline{\tau}$ is the maximum of $\tau$ and $\tau'$.
**Proof of Correctness:** Follows immediately from the usual properties of binary search (see, for example, [Knu98]), the correctness of the `Relative position query over` $S$ and the properties of the `root selection table`.
□

In particular, the previous query can be applied over connected regions which are $\text{Elim}_{X_n}(\mathcal{D}(\mathcal{P}))$–invariant.

### 3.3.2 CAD for Point Location

We introduce now the notion of *extended cylindrical (algebraic) decomposition of* $\mathbf{R}^n$ *induced by a family of polynomials* $P$.

**Definition 3.3.12.** Let $\mathcal{P}$ be a finite family of non-zero polynomials. We define $\mathcal{P}^\star := \cup_{i=1}^n \mathcal{P}_i$ where $\mathcal{P}_n := \mathcal{P}$ and, for any $i = 1, ..., n-1$, $\mathcal{P}_i := \text{Elim}_{X_{i+1}}(\mathcal{D}(\mathcal{P}_{i+1}))$. We further define $\mathcal{P}_{\leq i} := \cup_{j \leq i} \mathcal{P}_j$.

For $1 \leq i \leq n$, consider the family $\mathcal{S}_i$ consisting of the connected components of the non-empty realizations of sign conditions on $\mathcal{P}_{\leq i}$.

We call the sequence $\mathcal{S}_1, ..., \mathcal{S}_n$ the *extended cylindrical decomposition of* $\mathbf{R}^n$ *induced by the family* $\mathcal{P}$. The elements of $\mathcal{S}_i$ are called *the cells of level* $i$ of the extended cylindrical decomposition of $\mathbf{R}^n$ induced by a family $\mathcal{P}$ or, for short, $\mathcal{P}$-cells of level $i$. We define $\mathbf{R}^0$ as the unique $\mathcal{P}$-cell of level 0 and $\mathcal{S}_0 := \{\mathbf{R}^0\}$.

The following proposition is a direct consequence of the Theorem 5.33 in [BPR06].

**Proposition 3.3.13.** With the notation from previous definition, for each $1 \leq i \leq n$, the $\mathcal{P}$-cells of level $i$ form a finite partition of $\mathbf{R}^i$ into semi-algebraic subsets and satisfy the following properties:

- each cell $S \in \mathcal{S}_1$ is either a point or an open interval;
- for every $1 \leq i \leq n$ and every $S \in \mathcal{S}_i$, $S$ is a $\mathcal{P}_i$-stratum over the $\mathcal{P}$-cell $\pi_{i-1}(S)$ of level $i-1$.

**Definition 3.3.14.** To any cell of level $i$ ($1 \leq i \leq n$) in the extended cylindrical decomposition of $\mathbf{R}^n$ induced by a family $\mathcal{P}$ we associate its *cell code* in $\mathbf{N}^i$ defined as follows:

- the $\mathcal{P}$-cell code associated to a cell $C$ of level 1 is its stratum number;
- if, for some $1 < i \leq n$, $C$ is a $\mathcal{P}$-cell of level $i$ that is the $m^{th}$ stratum over the $\mathcal{P}$-cell $\pi_{i-1}(C)$ of level $i-1$, we define its cell code as $code(C) = (code(\pi_{i-1}(C)), m)$.

If $\alpha$ is a $\mathcal{P}$-cell code, we denote its corresponding $\mathcal{P}$-cell by $C_\alpha$.

**Definition 3.3.15.** Let $\mathcal{P}$ be a finite family of non-zero polynomials. Using the notation from Definition 3.3.12, we define a *CAD database for the family* $\mathcal{P}$ as a database that contains, for $1 \leq i \leq n$, the family of polynomials $\mathcal{P}_i$ and, for each $\mathcal{P}$-cell $S$ of level $i$ ($0 \leq i < n$) in the extended cylindrical decomposition of $\mathbf{R}^n$ induced by $\mathcal{P}$:

- a `root selection table for` $\mathcal{P}_{i+1}$ `over` $S$ and
- for each $P \in \mathcal{P}_{i+1}$ the `Thom encoding table of` $P$ `over` $S$.

Before presenting the $\mathcal{P}$`-stratum-number query` we bound the degree and number of polynomials in $\mathcal{P}_{\leq i}$, for $1 \leq i \leq n$. The next bounds follow directly from the complexity analysis of Algorithm 12.34 in [BPR06] (*cf.* also Theorem 10 in [Col75]).

**Proposition 3.3.16.** Let $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$ be a family of $s$ nonzero polynomials with degree bounded by $d$. Using the notation from Definition 3.3.12, for any $1 \leq i \leq n$, the set $\mathcal{P}_{n-i}$, has at most $\mathcal{O}(sd)^{3^i}$ polynomials of degree bounded by $\mathcal{O}(d)^{2^i}$. □

**Algorithm 3.3.17** ($\mathcal{P}$`-stratum-number query`).

**Context:** Let $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$ be a family of $s$ polynomials of degree at most $d$ and logarithmic height bounded by $\tau$.
Assume precomputed the CAD database for the family $\mathcal{P}$
**Input:** A triangular Thom encoding for $x \in \mathbf{R}^n$ of size $(d', \tau')$.
**Output:** The cell code of the $\mathcal{P}$-cell containing $x$.
**Procedure:**

```
Let $m$ be the answer to a $\mathcal{P}$-stratum-number query on $x_1$.
Let $\alpha$ be an empty list of integers. Append $m$ to $\alpha$.
For $i = 2$ to $n$ do
−Let $m$ be the answer to a $\mathcal{P}_i$-stratum-number query over $C_\alpha$ in $(x_1, ..., x_i)$.
−Add $m$ to the list $\alpha$.
Return $\alpha$.
```

**Complexity Analysis:** For $1 \leq i \leq n$, let $d_i$ and $s_i$ be the degree and the number of polynomials in $\mathcal{P}_i$. By Proposition 3.3.16, $d_i = \mathcal{O}(d)^{2^{n-i}}$ and $s_i = \mathcal{O}(sd)^{3^{n-i}}$. Let us denote $\overline{d_i} := max\{d_i, d'\}$. For $1 \leq i \leq n$, the $\mathcal{P}_i$`-stratum-number query` costs $log(s_i)\overline{d_i}^{\mathcal{O}(n)}$. Adding the complexity of these steps and simplifying, we obtain that the total complexity is $log(s)d^{\mathcal{O}(1)^n}d'^{\mathcal{O}(n)}$. The integers involved in these operations have bitsize bounded by $\overline{\tau}d^{\mathcal{O}(1)^n}d'^{\mathcal{O}(n)}$.
**Proof of Correctness:** Follows from the Definition 3.3.14 (cell-code definition) and the correctness of the Algorithm 3.3.11 ($\mathcal{P}$`-stratum-number query`).
□

### 3.3.3 Construction of the Database

The constructions of the Cylindrical Algebraic Decomposition from [Col75] and [BPR06] contain all the ingredients we need to build our CAD database. We summarize the precise result in the following theorem.

**Theorem 3.3.18.** Let $\mathcal{P} := \{P_1, ..., P_s\}$ be a family non-zero of polynomials in $\mathbf{R}[X_1, ..., X_n]$ of degree bounded by $d \geq 2$.

Then, there exists a data structure of size $(sd)^{\mathcal{O}(1)^n}$, that can be constructed in time $(sd)^{\mathcal{O}(1)^n}$ and that allows to solve the point location problem for the family $\mathcal{P}$. The number of arithmetic operations performed for a query point $x \in \mathbf{R}^n$ represented by a triangular Thom encoding of size $(d', \tau')$ is $\log(s)d^{\mathcal{O}(1)^n}d'^{\mathcal{O}(n)}$.

Moreover, if the bitsize of the coefficients of $\mathcal{P}$ is bounded by $\tau$, the bitsize of the intermediary computation of the preprocessing stage and of the polynomials stored in the database is bounded by $\tau d^{\mathcal{O}(1)^n}$. The bitsize of the integers involved in th query evaluation is bounded by $\overline{\tau} d^{\mathcal{O}(1)^n} d'^{\mathcal{O}(n)}$, where $\overline{\tau} := \max\{\tau, \tau'\}$.

*Proof.* The Algorithm 12.32 (improved cylindrical decomposition) in [BPR06] computes the polynomials in $\mathcal{P}^\star$ (see Definition 3.3.12), a sample point for each cell and the sign conditions valid at each sample point, satisfying the bounds in statement of the theorem. From this information, the tables for the CAD database can be easily computed and ordered. Finally, using sample points, we identify different $\mathcal{P}$-cells corresponding to the same face of $\mathcal{A}(\mathcal{P})$. The complexity bounds of the whole procedure are those of the statement of the theorem, since the $\mathcal{O}$ symbol hides the cost of the last steps.

The query evaluation time follows from the correctness of Algorithm 3.3.17 ($\mathcal{P}$-`stratum-number query`). □

## 3.4 Point Location for Polynomials with Integer Coefficients

In 1984, Meyer auf der Heide [MadH84] solved the point location problem for a family of linear polynomials with integer coefficients. His strategy was to find, given a family of affine linear polynomials in $\mathbf{Z}[X_1, ..., X_n]$ of logarithmic height bounded by $\tau \in \mathbf{N}$, an $\varepsilon > 0$ depending only on $n$ and $\tau$, called the *coarseness* of the given family, such that any hypercube in $\mathbf{R}^n$ of side-length $\varepsilon$ has the following property: all the affine hyperplanes defined by the given family that cut the given hypercube have a common intersection point. This fact allowed him to design a non-uniform polynomial-time algorithm which solves the Knapsack Problem. In the present section, we shall generalize this argumentation to semi-algebraic subsets of $\mathbf{R}^n$ of arbitrary degree.

### 3.4.1 Introduction

Let $R$ be an hypercube in $\mathbf{R}^n$ and let $P \in \mathbf{Z}[X_1, ..., X_n]$ be a polynomial. If the sign of $P$ is not constant on $R$ we say the $P$ *cuts* $R$.

Let $\mathcal{P} := \{P_1, ..., P_s\}$ be a family of polynomials in $\mathbf{Z}[X_1, ..., X_n]$ of total degree and logarithmic height bounded by $d$ and $\tau$ respectively. We remark that this implies a bound on the cardinality $s$ of $\mathcal{P}$: since each polynomial is determined by its $\binom{n+d}{d}$ coefficients and each coefficient by its $\tau$ bits, we obtain $s \leq 2^{\tau\binom{n+d}{d}} \leq 2^{\tau(d+1)^n}$.

We further remark that this bound on the cardinality of $\mathcal{P}$ makes the parameter $s$ disappear from our complexity bounds. Since $s$ is bounded by $2^{\tau(d+1)^n}$, a bound of the kind $\tau d^{\mathcal{O}(n)} \log(s)$ equals $\tau d^{\mathcal{O}(n)}$, where $s$ does not intervene.

In this section, we assume that the family $\{P_1, ..., P_s\}$ is generic in the following sense: for any $1 \leq r \leq n$ and any $1 \leq i_1 < ... < i_r \leq s$, the polynomials $P_{i_1}, ..., P_{i_r}$ form a regular sequence in $\mathbf{Q}[X_1, ..., X_n]$ or generate the trivial ideal. In the sequel, we suppose again $d \geq 2$.

Under this genericity assumption, we decompose the hypercube $[0,1]^n \subset \mathbf{R}^n$ in small closed hypercubes with mutually disjoint interiors. For each of these small hypercubes, there exist at most $n$ polynomials of $\mathcal{P}$ which cut it; the remaining polynomials in $\mathcal{P}$ have constant, positive or negative, sign on the hypercube. The side length of each of these hypercubes is of order $2^{-\tau d^{\mathcal{O}(n^2)}}$.

The information about which polynomials of $\mathcal{P}$ cut a given small hypercube and the signs of the remaining polynomials is stored, at a preprocessing stage, in a data structure.

First, we present an algorithm that solves the sign condition problem. It works as follows. Given a point $x \in [0,1]^n$, the preconstructed data structure allows us to determine in $\tau d^{\mathcal{O}(n^2)}$ steps a small hypercube $R_x$ containing the point $x$ and the polynomials in $\mathcal{P}$ that cut this hypercube. Evaluating at $x$ the (at most $n$) polynomials of $\mathcal{P}$ that cut the hypercube $R_x$, we solve the sign condition problem for the family $\mathcal{P}$ (see the definition in Section 2.1) performing $\tau d^{\mathcal{O}(n^2)}$ operations. We remark again that $s$—the number of polynomials in $\mathcal{P}$—is not involved in the complexity of the query and is bounded by the remaining parameters.

Since different connected components of the same sign condition may intersect the same small hypercube, this algorithm does not solve the point location problem. Under the additional hypothesis that different connected components of a same sign condition have disjoint closures, we present an algorithm that solves the point location problem for the family $\mathcal{P}$ in $\tau d^{\mathcal{O}(n^3)}$ operations.

### 3.4.2 The Arithmetic Relation between Distance and Height

We extend the notion of height to rational numbers. If $r = \frac{p}{q} \neq 0$ is a rational number with $p, q \in \mathbf{Z}$ coprime, we define its *height* as $H(r) := \max\{H(p), H(q)\}$ and its *logarithmic height* as $h(r) := \max\{h(p), h(q)\}$.

From Theorem 1.3.1 in [BPR94] (restated as Theorem 14.21 in [BPR06]) we immediately obtain the following result.

**Proposition 3.4.1.** Let $\varphi(Y_1, ..., Y_l)$ be an existential first-order formula, with $l$ free variables of the form $\exists X_1 ... \exists X_k \psi$, where $\psi$ is a boolean combination of polynomial equalities and inequalities involving polynomials of degree bounded by $d$ and logarithmic height bounded by $\tau$. Then, there exists an equivalent quantifier-free formula involving polynomials of degree bounded by $d^{\mathcal{O}(k)}$ and logarithmic height bounded by $\tau d^{\mathcal{O}(kl)}$. □

**Lemma 3.4.2.** Let $\psi$ be a quantifier-free first-order formula over the reals with only one free variable and involving polynomials with integer coefficients of height bounded by $H$. Suppose that there exists real numbers $\mu \neq 0$ that belongs to the border of the realization of $\psi$, $\mathcal{R}(\psi) \subset \mathbf{R}$. Then, $|\mu| > \frac{1}{H+1}$.

*Proof.* Since the formula $\psi$ is quantifier free, $\mu$ must be a root of some of the polynomials involved in $\psi$, namely $P = \sum_{i=0}^{d} a_i X^i \in \mathbf{Z}[X]$. The height of $P$ is bounded by $H$. We assume, without loss of generality, that $a_0 \neq 0$.

Thus, by Cauchy's bound for the zeroes of a polynomial, the absolute value of any root of $P$ is at least $\frac{1}{H+1}$. In particular, $|\mu| > \frac{1}{H+1}$. □

**Definition 3.4.3.** For $\alpha_1 < \beta_1, \alpha_2 < \beta_2, ..., \alpha_n < \beta_n \in \mathbf{Q}$, the set

$$[\alpha_1, \beta_1] \times ... \times [\alpha_n, \beta_n] \subset \mathbf{R}^n$$

is called a *rational hyper-rectangle* with coordinates $\alpha_1, ..., \alpha_n, \beta_1, ..., \beta_n$. Its *logarithmic height* is defined as the maximal logarithmic height of its coordinates.

**Proposition 3.4.4.** There exists a universal constant $c \in \mathbf{R}, c > 0$ that satisfies, for any three positive integers $\tau, n$ and $d > 2$ the following property.

Let $R \subset \mathbf{R}^n$ be a rational hyper-rectangle of logarithmic height $\tau$ and let $P_1, ..., P_r, Q \in \mathbf{Z}[X_1, ..., X_n]$ be polynomials of degree at most $d$ and of logarithmic height bounded by $\tau$. Consider the subsets of $\mathbf{R}^n$

$$\mathcal{P} := \{P_1 = 0, ..., P_r = 0\} \cap R \text{ and } \mathcal{Q} := \{Q = 0\} \cap R.$$

If $\mathcal{P}$ and $\mathcal{Q}$ are disjoint then their euclidian distance is strictly greater than $2^{-\tau d^{c \cdot n}}$.

*Proof.* Let us assume that $\mathcal{P}$ and $\mathcal{Q}$ are disjoint and non-empty. Since they are closed and bounded, the distance between them is positive, say $\mu \in \mathbf{R}_{>0}$. Denoting by $\chi_R$ the first-order formula that expresses the set $R$, we consider the variable $\varepsilon$ and the following first-order formula:

$$\begin{aligned} \varphi(\varepsilon) := \exists x_1 \, ... \, \exists x_n \exists y_1 \, ... \, \exists y_n \quad & \chi_R(x) \ \wedge \ \chi_R(y) \ \wedge \\ & P_1(x) = 0 \ \wedge \ ... \ \wedge \ P_r(x) = 0 \ \wedge \\ & Q(y) = 0 \ \wedge \ |x - y|^2 \leq \varepsilon^2. \end{aligned}$$

Clearly, a positive real number $\varepsilon$ satisfies this formula if and only if $\varepsilon \geq \mu$. The formula $\varphi$ has $\varepsilon$ as the only free variable, $x_1, ..., x_n, y_1, ..., y_n$ as bounded variables and involves polynomials of degree at most $d$ and logarithmic height at most $\tau$. Hence, by Proposition 3.4.1, there exists an equivalent quantifier-free first-order formula $\psi$ involving polynomials of degree bounded by $d^{\mathcal{O}(n)}$ and height bounded by $2^{\tau d^{\mathcal{O}(n)}}$. Thus, by Lemma 3.4.2, $\mu$ is at least $\frac{1}{1+2^{\tau d^{\mathcal{O}(n)}}} = 2^{-\tau d^{\mathcal{O}(n)}}$. □

### 3.4.3 The Mathematical Insight

We recall that our genericity hypothesis (see Section 3.4.1) implies that, for any point of $\mathbf{R}^n$, there exist at most $n$ polynomials of the family $\mathcal{P}$ intersect simultaneously at any point in $\mathbf{R}^n$. Given an hypercube $R \subset \mathbf{R}^n$, we say that $\mathcal{P}$ is *$k$-determined over $R$* if at most $k$ polynomials of the family $\mathcal{P}$ cut $R$.

**Proposition 3.4.5.** Let $\mathcal{P} := \{P_1, ..., P_s\} \subset \mathbf{Z}[X_1, ..., X_n]$ be a family of polynomials of total degree bounded by $d$ and logarithmic height bounded by $\tau$.

Then there exists a constant $c' \in \mathbf{N}$ such that the rational number $\delta := 2^{-\tau d^{c' \cdot n^2}} > 0$ of logarithmic height $\tau d^{\mathcal{O}(n^2)}$, satisfies the following property: any hypercube $R \subset [0,1]^n$ of side length $\delta$ is $n$-determined for the family $\mathcal{P}$.

Following Meyer auf der Heide, we call the rational number $\delta$ a *coarseness* for the family $\mathcal{P}$.

*Proof.* Let $R_0 = [0,1]^n \subset \mathbf{R}^n$. If $R_0$ is $n$-determined we are done. Otherwise, we proceed with the following construction.

Let $c \in \mathbf{R}, c > 0$ be such that $\delta_1 = 2^{-\tau d^{c \cdot n}}$ satisfies the Proposition 3.4.4 on $R_0$ for every $P_1, ..., P_r, Q \in \mathcal{P}$ with $r \leq n$. Let $0 < \delta_1' < \frac{\delta_1}{\sqrt{n}}$ be a rational number of logarithmic height at most $\tau \ulcorner \log(n) \urcorner d^{c \cdot n}$. Let us consider the subdivision of $R_0$ in rational hypercubes of side length $\delta_1'$ and logarithmic height $\tau n d^{c \cdot n}$ and let $R_1$ be one such hypercube.

We claim that there are two possibilities:

1. $R_1$ is $n$-determined,
2. at most $n-1$ polynomials in $\mathcal{P}$ intersect simultaneously in $R_1$.

To prove our claim, suppose that $R_1$ is not $n$-determined and that there exists $1 \leq i_1 < ... < i_n \leq s$ such that $\mathcal{P} := \{P_{i_1} = 0, ..., P_{i_n} = 0\} \cap R_1 \neq \emptyset$. Since $R_1$ is not $n$-determined, there exists $Q \in \mathcal{P} \setminus \{P_{i_1}, ..., P_{i_n}\}$ that cuts $R_1$. By our genericity hypothesis, $\mathcal{P} \cap \{Q = 0\}$ is not empty. Hence, by Proposition 3.4.4, the distance between $\mathcal{P}$ and $\{Q = 0\} \cap R_1$ is at least $\delta_1$. Let $x \in \mathcal{P}$ and

$y \in \{Q = 0\} \cap R_1$. Then, since both belong to $R_1$, $|x - y| \leq \sqrt{n}\delta_1' < \delta_1$. This contradicts Proposition 3.4.4 and proves our claim.

We iterate this process. Let $1 < i \leq n$ and let us assume as inductive hypothesis that after the $i^{th}$ iteration we obtain a partition of $[0, 1]^n$ into rational hypercubes of side length $\delta_i' > 0$ and logarithmic height bounded by $\tau(\ulcorner\log(n)\urcorner d^{c\cdot n})^i$ such that each hypercube is $(n-i+1)$-determined or at most $n - i$ polynomials in $\mathcal{P}$ intersect simultaneously in it.

Let $R_i$ be one hypercube in this partition. If $R_i$ is $(n - i + 1)$-determined we are done. Thus, let us suppose that at most $n-i$ polynomials in $\mathcal{P}$ intersect simultaneously in $R_i$.

We subdivide $R_i$ in smaller hypercubes. Let $\delta_{i+1} = 2^{-\tau(\ulcorner\log(n)\urcorner d^{c\cdot n})^i d^{c\cdot n}} = 2^{-\tau(\ulcorner\log(n)\urcorner)^i d^{(i+1)c\cdot n}}$ satisfy Proposition 3.4.4 on the hypercube $R_i$ for every $P_1, ..., P_r, Q \in \mathcal{P}$ with $r \leq n$. Let $0 < \delta_{i+1}' < \frac{\delta_{i+1}}{\sqrt{n}}$ be a rational number of logarithmic height $\tau(\ulcorner\log(n)\urcorner d^{c\cdot n})^{(i+1)}$. Let us consider the subdivision of $R_i$ in rational hypercubes of side length $\delta_{i+1}'$ and logarithmic height $\tau(\ulcorner\log(n)\urcorner d^{c\cdot n})^{(i+1)}$ and let $R_{i+1}$ be one such hypercube.

As before, we claim that there are two possibilities:

1. $R_{i+1}$ is $(n - i)$-determined,

2. at most $n - i - 1$ polynomials in $\mathcal{P}$ intersect simultaneously in $R_{i+1}$.

To prove our claim, suppose that $R_{i+1}$ is not $(n - i)$-determined and that exist indices $1 \leq j_1 < ... < j_{n-i} \leq s$ such that $\mathcal{P} := \{P_{j_1} = 0, ..., P_{j_{n-i}} = 0\} \cap R_1$ is not empty. Since $R_{i+1}$ is not $(n - i)$-determined, there exists $Q \in \mathcal{P}\setminus\{P_{j_1}, ..., P_{j_{n-i}}\}$ that cuts $R_{i+1}$. By our inductive hypothesis ($R_i$ is $(n-i+1)$-determined), $\mathcal{P} \cap \{Q = 0\} \cap R_{i+1} = \emptyset$. Hence, by Proposition 3.4.4, the distance between $\mathcal{P}$ and $\{Q = 0\} \cap R_{i+1}$ is at least $\delta_{i+1}$. Let $x \in \mathcal{P}$ and $y \in \{Q = 0\} \cap R_{i+1}$. Then, since both belong to $R_{i+1}$, $|x - y| \leq \sqrt{n}\delta_{i+1}' < \delta_{i+1}$. This contradicts Proposition 3.4.4, proves our claim and completes the inductive step.

It follows that, after $n$ iterations, we arrive to an hypercube $R_n$ which is 1-determined.

In particular, it follows from the construction that any hypercube $R \subset R_0$ of side length $\delta := \delta_n' = 2^{-\tau(\ulcorner\log(n)\urcorner d^{c\cdot n})^n} = 2^{-\tau d^{\mathcal{O}(n^2)}}$ is $n$-determined for the family $\mathcal{P}$. □

We briefly discuss the case of affine hyperplanes ($d = 1$, the affine linear case). Although in this case Proposition 3.4.1 and hence Proposition 3.4.4 can be improved to obtain heights which are simply exponential in the dimension $n$, the height resulting from the iterative method presented in the last proposition remains doubly exponential in the dimension ($2^{\tau \mathcal{O}(n)^n}$). Hence, we keep our assumption $d \geq 2$ in the sequel.

### 3.4.4 The Sign Condition Algorithm

Based on the results of the last paragraph, we construct a big data structure of size $2^{\tau d^{\mathcal{O}(n^2)}}$ that allows, for any $x \in [0,1]^n$, a fast determination of the sign condition satisfied by the polynomials in $\mathcal{P}$ at $x$.

In order to specify, in the description of our algorithms, the assumptions made about the parameters of an algorithm and to differentiate these from the inputs of the algorithm, we use the label **Context:**.

First we state the following algorithm that is a form of multidimensional binary search, with $m$ as a parameter.

**Algorithm 3.4.6** ($\frac{1}{m}$`-grid query`).

**Context:** Let be given a positive integer $m$ of logarithmic height, say $\mu$.
**Input:** A triangular Thom encoding for $x = (x_1, ..., x_n) \in [0,1]^n$ of size $(d', \tau')$.
**Output:** $(i_1, ..., i_n) \in \mathbf{N}^n$ such that $x \in \Pi_{j=1}^n [\frac{i_j}{m}, \frac{i_j+1}{m}]$
**Procedure:**
For $j = 1, ..., n$ perform a binary search to determine $i_j$ such that $x_j \in [\frac{i_j}{m}, \frac{i_j+1}{m}]$.
Each comparison of the binary search is done evaluating the sign of a polynomial $X_j - \frac{q}{m}$ for an appropriate integer $q$.
Return $(i_1, ..., i_n)$.
**Complexity Analysis:** Each binary search requires $\mathcal{O}(\mu)$ comparisons. Each comparison costs $d^{\mathcal{O}(1)}$ arithmetic operations using the `sign determination algorithm` (see Proposition A.4.3). Hence, the algorithm performs $\mu d'^{\mathcal{O}(1)} n$ arithmetic operation between integers of logarithmic height bounded by $\overline{\mu} d'^{\mathcal{O}(1)}$, where $\overline{\mu}$ is the maximum between $\mu$ and $\tau'$.
**Proof of Correctness:** The proof is delicate, but follows the usual argumentation of binary search (see, for instance [Knu98]). □

Let $\mathcal{P} = \{P_1, ..., P_s\} \subset \mathbf{Z}[X_1, ..., X_n]$ be a finite family of polynomials and let $m \in \mathbf{Z}$. We define the $\frac{1}{m}$`-grid cut array` of the family $\mathcal{P}$ as the $n$-dimensional array of size $m^n$ that in the position $(i_1, ..., i_n) \in \{0, ..., m-1\}^n$ contains the list of the indices of the polynomials in $\mathcal{P}$ that cut the hypercube $\Pi_{j=1}^n [\frac{i_j}{m}, \frac{i_j+1}{m}] \subset [0,1]^n$. By Proposition 3.4.5, each of these lists contains at most $n$ indices.

The $\frac{1}{m}$`-grid cut array` can be queried using the following algorithm.

**Algorithm 3.4.7** (`Sign condition query`).

**Context:** Let $\mathcal{P} = \{P_1, ..., P_s\} \subset \mathbf{Z}[X_1, ..., X_n]$ be a generic family of polynomials of degree bounded by $d$ and logarithmic height bounded by $\tau$.
Let $m \in \mathbf{N}$ of logarithmic height $\mu$ such that any hypercube contained in $[0,1]^n$ of side length $\frac{1}{m}$ is $n$-determined for the family $\mathcal{P}$.
Assume precomputed the $\frac{1}{m}$`-grid cut array` for $\mathcal{P}$.
**Input:** A triangular Thom encoding for $x = (x_1, ..., x_n) \in [0,1]^n$ of size $(d', \tau')$.
**Output:** $(i_1, ..., i_n) \in \mathbf{N}^n$ such that $x \in R := \Pi_{j=1}^n [\frac{i_j}{m}, \frac{i_j+1}{m}]$ and the sign condition satisfied at $x$ by the polynomials in $\mathcal{P}$ that cut the hypercube $R$.
**Procedure:**
Perform a $\frac{1}{m}$`-grid query` to determine $(i_1, ..., i_n) \in \mathbf{N}^n$ such that

$$x \in \Pi_{j=1}^n [\frac{i_j}{m}, \frac{i_j+1}{m}].$$

Let $j_1, ... j_k$ be the indices in the position $(i_1, ..., i_n)$ of the $\frac{1}{m}$`-grid cut array`. Using the `sign determination algorithm`, evaluate the sign condition satisfied by $P_{j_1}, ..., P_{j_k}$ at $x$.
*Return*$(i_1, ..., i_n)$ and the sign condition satisfied by $P_{j_1}, ..., P_{j_k}$ at $x$.
**Complexity Analysis:** The first step requires $\mu d'^{\mathcal{O}(n)}$ arithmetic operations between integers of logarithmic height bounded by $\max\{\mu, \tau'\} d'^{\mathcal{O}(n)}$ by the complexity analysis of Algorithm 3.4.6.
By Proposition A.4.3 (`sign determination algorithm`), the evaluation of each polynomials costs $\overline{d}^{\mathcal{O}(n)}$ arithmetic operations between integers of logarithmic height bounded by $\max\{\tau, \tau'\} \overline{d}^{\mathcal{O}(n)}$, where $\overline{d} = \max\{d, d'\}$. Since at most $n$ polynomials in $\mathcal{P}$ cut the hypercube $R$, the complete evaluation phase costs at most $n\overline{d}^{\mathcal{O}(n)} = \overline{d}^{\mathcal{O}(n)}$ arithmetic operations. Thus, the total algebraic complexity is $\mu d'^{\mathcal{O}(n)} + \overline{d}^{\mathcal{O}(n)}$.
**Proof of Correctness:** Follows from the correctness of the Algorithm 3.4.6 ($\frac{1}{m}$`-grid query`) and the properties of the $\frac{1}{m}$`-grid cut array`. □

The following proposition is useful to analyze the cost of the precomputation.

**Proposition 3.4.8.** Let $R \subset \mathbf{R}^n$ be an hypercube, let $P \in \mathbf{Z}[X_1, ..., X_n]$ be a polynomial and suppose that $\tau$ bounds the logarithmic heights of $R$ and $P$. Then, it is possible to decide whether $P$ cuts $R$ in $d^{\mathcal{O}(n)}$ arithmetic operations between numbers of logarithmic height bounded by $\tau d^{\mathcal{O}(n)}$.

*Proof.* Clearly, the formula

$$\varphi_P := \exists x \; (P(x) = 0 \wedge \chi_R(x))$$

is true if and only if $P$ cuts $R$. By Theorem 13.14 in [BPR06], this sentence can be decided within the stated bounds. $\square$

**Theorem 3.4.9.** Let $\mathcal{P} := \{P_1, ..., P_s\}$ be a generic family of polynomials in $\mathbf{Z}[X_1, ..., X_n]$ of total degree and logarithmic height bounded by $d$ and $\tau$ respectively.

Then, there exists a data structure of size $2^{\tau d^{\mathcal{O}(n^2)}}$, that can be constructed in time $2^{\tau d^{\mathcal{O}(n^2)}}$ and that allows to determine, for any $x \in [0,1]^n$ given by a triangular Thom encoding of size $(d', \tau')$, the sign conditions satisfied by the polynomials in $\mathcal{P}$ at $x$ in $\tau d^{\mathcal{O}(n^2)} d'^{\mathcal{O}(n)}$ arithmetic operations between integers of logarithmic height bounded by $\max\{\tau d^{\mathcal{O}(n^2)}, \tau'\} d'^{\mathcal{O}(n)}$.

*Proof.* By Proposition 3.4.5, there exists a constant $c \in \mathbf{N}$ such that, defining $m := 2^{\tau d^{cn^2}}$ and $\delta := \frac{1}{m}$, $\mathcal{P}$ is $n$-determined over any hypercube $R \subset [0,1]^n$ of side length $\delta$, *i.e.*, $\delta$ is a coarseness for the family $\mathcal{P}$. We assume that the constant $c$ has been computed before the construction of the data structure.

We first compute the $\frac{1}{m}$-`grid cut array` for $\mathcal{P}$: for any $0 \leq i_1, ..., i_n < m$ and any $1 \leq k \leq s$, we determine if the polynomial $P_k$ cuts the hypercube $\Pi_{j=1}^{n}[\frac{i_j}{m}, \frac{i_j+1}{m}]$ using Proposition 3.4.8. The complete construction of the $\frac{1}{m}$-`grid cut array` for $\mathcal{P}$ requires $sm^n d^{\mathcal{O}(n)} = 2^{\tau d^{\mathcal{O}(n^2)}}$ arithmetic operation between rational numbers of logarithmic height bounded by $\tau d^{\mathcal{O}(n^2)}$.

Suppose now precomputed the $\frac{1}{m}$-`grid cut array` for $\mathcal{P}$. We claim that the `Sign condition query` computes a partition of $\mathbf{R}^n$ finer that $\mathcal{S}(\mathcal{P})$. To prove our claim let $x, x' \in [0,1]^n$ be such that the `Sign condition query` outputs the same result $(i_1, ..., i_n), \sigma$ for both $x$ and $x'$. Hence, $x$ and $x'$ belong to the same small hypercube $R = \Pi_{j=1}^{n}[\frac{i_j}{m}, \frac{i_j+1}{m}]$ and satisfy the same sign condition $\sigma$ on the polynomials that cut $R$. Since the remaining polynomials in $\mathcal{P}$ do not cut $R$, their sign is invariant on $R$. In particular, $\mathcal{P}$ realizes the same sign condition at $x$ and at $x'$.

Now it is easy to identify (using a precomputed extra table) within the same complexity bounds the different outputs of the `Sign condition query` corresponding to a same sign condition.

The algebraic complexity of the `Sign condition query` is $\mu d'^{\mathcal{O}(n)} + \overline{d}^{\mathcal{O}(n)}$, where $\mu = \tau d^{\mathcal{O}(n^2)}$ is the logarithmic height of $m$ and $\overline{d} = \max\{d, d'\}$. Thus, in our case, the `Sign condition query` requires $\tau d^{\mathcal{O}(n^2)} d'^{\mathcal{O}(n)}$ arithmetic operations between integers of logarithmic height bounded by $\max\{\tau d^{\mathcal{O}(n^2)}, \tau'\} \cdot d'^{\mathcal{O}(n)}$. $\square$

### 3.4.5 The Point Location Algorithm

Different faces of the arrangement $\mathcal{A}(\mathcal{P})$ can satisfy the same sign condition. The following results are used to distinguish these different components.

**Proposition 3.4.10.** Let $S_1, S_2 \subset \mathbf{R}^n$ be two different connected components of a semi-algebraic set described by a quantifier-free formula involving polynomials of degree bounded by $d$ and height bounded by $H$. If the distance $\mu$ between $S_1$ and $S_2$ is positive, then it is at least $H^{-d^{\mathcal{O}(n^3)}}$.

*Proof.* By Proposition 16.22 in [BPR06], $S_1$ and $S_2$ can be described by first-order quantifier-free formulas $\chi_{S_1}$ and $\chi_{S_2}$ involving polynomials of degree bounded by $d^{\mathcal{O}(n^3)}$ and height bounded by $H^{d^{\mathcal{O}(n^3)}}$. Consider the formula

$$\varphi(\varepsilon) := \exists x_1 \dots \exists x_n \exists y_1 \dots \exists y_n \quad (\chi_{S_1}(x) \ \wedge \ \chi_{S_2}(y) \ \wedge \ |x-y|^2 \leq \varepsilon^2).$$

The formula $\varphi$ has one free variable, $2n$ quantified variables and involves polynomials of degree bounded by $d^{\mathcal{O}(n^3)}$ and height bounded by $H^{d^{\mathcal{O}(n^3)}}$. Clearly, $\mu$ belongs to the border of the realization of $\varphi$. Hence, By Proposition 3.4.1, there exists an equivalent quantifier-free first-order formula $\psi$ involving polynomials of height bounded by $H^{d^{\mathcal{O}(n^3)}}$. Thus, by Lemma 3.4.2, $\mu$ is at least $\frac{1}{1+H^{d^{\mathcal{O}(n^3)}}} = H^{-d^{\mathcal{O}(n^3)}}$. □

Adding the hypothesis that the distance between any two different connected components of a same sign condition of the family $\mathcal{P}$ is positive, we can prove the following theorem.

**Theorem 3.4.11.** Let $\mathcal{P} := \{P_1, ..., P_s\}$ be a generic family of polynomials in $\mathbf{Z}[X_1, ..., X_n]$ of total degree and logarithmic height bounded by $d$ and $\tau$ respectively. Suppose that the distance between any two different connected components of a same sign condition of the family $\mathcal{P}$ is positive.

Then, there exists a data structure of size $2^{\tau d^{\mathcal{O}(n^3)}}$, that can be constructed in time $2^{\tau d^{\mathcal{O}(n^3)}}$ and that allows to solve the point location problem, for any $x \in [0,1]^n$ given by a triangular Thom encoding of size $(d', \tau')$, in $\tau d^{\mathcal{O}(n^3)} d'^{\mathcal{O}(n)}$ arithmetic operations between integers of logarithmic height bounded by $d'^{\mathcal{O}(n)} \max\{\tau d^{\mathcal{O}(n^3)}, \tau'\}$.

*Proof.* Since the distance between any two different connected components of a same sign condition of the family $\mathcal{P}$ is positive, Proposition 3.4.10 implies that there exists $m := 2^{\tau d^{\mathcal{O}(n^3)}}$ such that any the distance between any two different connected components of a same sign condition of the family $\mathcal{P}$ is at least $\frac{\sqrt{n}}{m}$. By Proposition 3.4.5, the integer $m$ can be chosen to satisfy also that any hypercube of side length $\delta := \frac{1}{m}$ contained in $[0,1]^n$ is $n$-determined for the family $\mathcal{P}$ ($\delta$ is a coarseness for the family $\mathcal{P}$). We proceed as in the proof of Theorem 3.4.9.

We first compute the $\frac{1}{m}$-`grid cut array` for $\mathcal{P}$: for any $0 \le i_1, ..., i_n < m$, and any $1 \le k \le s$ we determine if the polynomial $P_k$ cuts the hypercube $\Pi_{j=1}^n [\frac{i_j}{m}, \frac{i_j+1}{m}]$, using the Proposition 3.4.8. The complete construction of the $\frac{1}{m}$-`grid cut array` for $\mathcal{P}$ requires $sm^n d^{\mathcal{O}(n)} = 2^{\tau d^{\mathcal{O}(n^3)}}$ arithmetic operation between rational numbers of logarithmic height bounded by $\tau d^{\mathcal{O}(n^3)}$.

Suppose now precomputed the $\frac{1}{m}$-`grid cut array` for $\mathcal{P}$. We claim that the `Sign condition query` computes a partition of $\mathbf{R}^n$ finer that $\mathcal{A}(\mathcal{P})$. To prove our claim let $x, x' \in [0,1]^n$ be such that the `Sign condition query` outputs the same result $(i_1, ..., i_n), \sigma$ for both $x$ and $x'$. Hence, $x$ and $x'$ belong to the same small hypercube $R = \Pi_{j=1}^n [\frac{i_j}{m}, \frac{i_j+1}{m}]$ and satisfy the same sign condition $\sigma$ on the polynomials that cut $R$. Since the remaining polynomials in $\mathcal{P}$ do not cut $R$, their sign is invariant on $R$. In particular, $\mathcal{P}$ realizes the same sign condition at $x$ and at $x'$.

Since $x$ and $x'$ belong to the same small hypercube $R$, their distance is at most $\frac{\sqrt{n}}{m}$. Hence, by construction, they cannot belong to different connected components of a same sign condition of the family $\mathcal{P}$. Since they satisfy the same $\mathcal{P}$-sign condition, we conclude that $x$ and $x'$ belong to the same face of the arrangement $\mathcal{A}(\mathcal{P})$. We remark that it is not difficult to identify, in the preprocessing stage, different outputs corresponding to the same face of $\mathcal{A}(\mathcal{P})$ within the same complexity bounds.

Finally, computing sample points (using Algorithm 13.11 in [BPR06]) corresponding to all the possible outputs of the `Sign condition query`, it is possible to construct (in a preprocessing stage) an ordered table that identifies with a same label different outputs of the `Sign condition query` corresponding to a same face of the arrangement $\mathcal{A}(\mathcal{P})$ (see Theorem 16.18 in [BPR06]). This construction can be done within the same space and time requirements as the $\frac{1}{m}$-`grid cut array` . The query time for this table is less than the sign condition query time. This allows us to solve the point location problem for the family $\mathcal{P}$.

The algebraic complexity of the `Sign condition query` is $\mu d'^{\mathcal{O}(n)} + \overline{d}^{\mathcal{O}(n)}$, where $\mu = \tau d^{\mathcal{O}(n^3)}$ is the logarithmic height of $m$ and $\overline{d} = \max\{d, d'\}$. Thus, in our case, the `Sign condition query` requires $\tau d^{\mathcal{O}(n^3)} d'^{\mathcal{O}(n)}$ arithmetic operations between integers of logarithmic height bounded by $\max\{\tau d^{\mathcal{O}(n^3)}, \tau'\} d'^{\mathcal{O}(n)}$. This completes the proof. □

## 3.5 Local Ray Shooting

In this section, we consider another important query from computational geometry: the *ray shooting query.* Computing the intersection between beams and geometric objects is a central problem in radio-therapy; see [Pel04] for references and the known results in the linear case. We shall study a local version of this query, defined in the sequel, that can be efficiently answered using the database constructed in the last section.

Given a large family of geometric objects in $\mathbf{R}^n$, the ray shooting query asks, given point $p \in \mathbf{R}^n$ and a direction $\overrightarrow{v}$, the first object in the family intersected by the ray $\{p + \lambda\overrightarrow{v} \mid \lambda > 0\}$ defined by the pair $p, \overrightarrow{v}$, if the ray intersects one such object or a message reporting that the ray intersects no object.

The *local ray shooting query* for a family of geometric objects and a fixed $\varepsilon > 0$ asks, given point $p \in \mathbf{R}^n$ and a direction $\overrightarrow{v}$, the first object in the family intersected by the ray defined by the pair $p, \overrightarrow{v}$, if the distance among the first intersection point and the original point $p$ is at most $\varepsilon$, or a message reporting that none exist.

In our case, each object is described as the set of zeroes of a polynomial in $\mathbf{R}[X_1, ..., X_n]$. By the local ray shooting query for a family of polynomials $\mathcal{P} \subset \mathbf{R}[X_1, ..., X_n]$, we understand the local ray shooting query for the family of sets $\{\{x \in [0,1] \mid P(x) = 0\} \mid P \in \mathcal{P}\}$.

Let $\mathcal{P} := \{P_1, ..., P_s\}$ be a family of polynomials in $\mathbf{Z}[X_1, ..., X_n]$ of total degree and logarithmic height bounded by $d$ and $\tau$ respectively. As in the last section, we assume that $\mathcal{P}$ is generic. We recall that in the last section, the unit hypercube $[0,1]^n$ was decomposed into small hypercubes of side length $\delta := 2^{-\tau d^{\mathcal{O}(n^2)}}$ ($\delta$ is the coarseness of the family $\mathcal{P}$ given by Proposition 3.4.5) and a database was constructed associating to each hypercube in this decomposition the (at most $n$) polynomials in the family $\mathcal{P}$ that cut it.

Let $k$ be a positive integer and let us consider $\varepsilon := k\delta$. We shall see that the database constructed in the last section allows to solve the local ray shooting problem for the family $\mathcal{P}$ of polynomials for the previously fixed $\varepsilon$ efficiently.

The local ray shooting query can be used as part of a ray tracing algorithm, for instance, to draw a neighborhood of any singularity of the semi-algebraic set $\{x \in [0,1] \mid P(x) = 0 \text{ for some } P \in \mathcal{P}\}$.

### 3.5.1 The algorithm

Let us discuss now the technique used to solve the local ray shooting problem.

**Theorem 3.5.1.** Let $\mathcal{P} := \{P_1, ..., P_s\}$ be a family of generic polynomials in $\mathbf{Z}[X_1, ..., X_n]$ of total degree and logarithmic height bounded by $d$ and $\tau$

respectively. Let $\delta := 2^{-\tau d^{\mathcal{O}(n^2)}}$ be the coarseness of the family $\mathcal{P}$ given by Proposition 3.4.5. Let $k$ be a positive integer and let $\varepsilon := k\delta$.

Then, the data structure constructed in Theorem 3.4.9 for the family $\mathcal{P}$, allows to solve the local ray shooting problem for this family and for the fixed $\varepsilon$ within the following complexity bounds.

For any $p \in [0,1]^n$ and any non-zero direction $\overrightarrow{v} \in \mathbf{R}^n$ given by triangular Thom encodings of size $(d', \tau')$, the local ray shooting algorithm performs $\overline{\tau} d'^{\mathcal{O}(1)} n + k\overline{d}^{\mathcal{O}(n)}$ arithmetic operations between integers of logarithmic height bounded by $\overline{\tau}\overline{d}^{\mathcal{O}(n)}$, where $\overline{d} := \max\{d, d'\}$ and $\overline{\tau} := \max\{\tau d^{\mathcal{O}(n^2)}, \tau'\}$.

*Proof.* The query algorithm we are going to present can be briefly described as follows. First, the algorithm determines a small hypercube $R$ of side length $\delta$ containing the input point $p$ using the $\frac{1}{m}$-`grid query` (Algorithm 3.4.6) for $m = 1/\delta$, performing $\overline{\tau} n d'^{\mathcal{O}(1)}$ arithmetic operations between integers of logarithmic height bounded by $\overline{\tau} d'^{\mathcal{O}(1)}$. Then, it verifies whether the ray emanating from $p$ with direction $\overrightarrow{v}$ intersects, inside $R$, some of the algebraic sets that cut this hypercube (these are at most $n$ by the genericity hypothesis and their indices are stored in the preconstructed $\frac{1}{m}$-`grid cut array`). If this is the case, it determines the first and reports it.

We remark that, following the direction of the ray emanating from $p$ with direction $\overrightarrow{v}$, the cubes of the partition intersected by this ray admit a natural total order. If no intersection point is found inside $R$ the algorithm proceeds to the next hypercube in the grid, $R'$, intersected by the ray emanating from $p$ with direction $\overrightarrow{v}$ (the determination of $R'$ can be done fast and its complexity is hidden in the total complexity of our method by the $\mathcal{O}$ symbol). Then, the algorithm repeats the previous process for $R'$, *i.e.*, it verifies whether the ray emanating from $x$ with direction $\overrightarrow{v}$ intersects, inside $R'$, some of the algebraic sets that cut this hypercube. If this is the case, it determines the first and report it. If this is not the case, it proceeds to the next hypercube in the grid intersected by the ray. This process is iterated (at most $nk$ times) until an hypercube whose distance to $p$ is bigger than $\varepsilon$ is reached. As soon as an intersection point is found, it is reported. If no intersection point is found, the algorithm reports that none exist.

To analyze the complexity of the method, we have to study the complexity of two different tasks:

- Given a polynomial $P \in \mathcal{P}$, determine whether it intersects the ray defined by $p, \overrightarrow{v}$ inside an hypercube $R$.

- Given the list of polynomials in $\mathcal{P}$ whose set of zeroes intersect, inside an hypercube $R$, the ray defined by $p, \overrightarrow{v}$, determine which of them defines the first intersection point.

Let us start with the first task. This problem can be stated as a decision problem for the existential theory of the reals. Let $\varphi_p(x)$ be the quantifier-free first-order formula based on the triangular Thom encoding of $p$ that is true only when $x = p$ holds and, analogously, for $\varphi_{\overrightarrow{v}}(x)$. Then, $\varphi_p(x)$ involves $d'$ polynomials of degree bounded by $d'$ and logarithmic height bounded by $\tau'$. Let $\varphi_R$ be the formula defining $R$.

Let $P \in \mathcal{P}$ and let us consider the first-order formula

$$\exists\lambda\ \exists p\ \exists v \quad (\varphi_p(p) \wedge \varphi_{\overrightarrow{v}}(v) \wedge \lambda > 0 \wedge P(p + \lambda v) = 0 \wedge \varphi_R(p + \lambda v)).$$

This formula expresses that the set of zeroes of the polynomial $P$ intersects a ray defined by $p, \overrightarrow{v}$ inside the hypercube $R$. Its truth can be determined using Theorem 13.14 in [BPR06]. The formula involves $2(d' + n + 1)$ polynomials of degree bounded by $\overline{d}$ and logarithmic height bounded by $\overline{\tau}$ and $2n + 1$ existentially quantified variables. Hence, its truth can be decided in $\overline{d}^{O(n)}$ arithmetic operations between integers of logarithmic height bounded by $\overline{\tau}\overline{d}^{O(n)}$.

The strategy for the second task is similar. Let us denote by $P_{i_1}, ..., P_{i_c}$ the polynomials in $\mathcal{P}$ whose sets of zeroes intersect the ray defined by $p, \overrightarrow{v}$ inside the hypercube $R$. We assume $c > 1$ since otherwise there is nothing to do. We remark that $c \leq n$, since (by the genericity hypothesis and the construction of $\delta$) at most $n$ polynomials in $\mathcal{P}$ cut the hypercube $R$.

First, considering the first intersection points of the sets of zeroes of $P_{i_1}$ and $P_{i_2}$ with the ray defined by $p, \overrightarrow{v}$, we determine which is nearer to $p$. Then, we compare the distance to $p$ of the nearest of these two points with the first intersection point determined by $P_{i_3}$ and so on. After $c - 1 < n$ comparisons, we will have determined the polynomial that defines the first intersection point.

Let $P_i$ and $P_j$ be two polynomials that intersect the ray defined by $p, \overrightarrow{v}$. Keeping the notation used in the last formula, consider the following one:

$$\begin{aligned}\exists\lambda\ \exists p\ \exists v\quad \forall\lambda' \quad (\varphi_p(p) \wedge \varphi_{\overrightarrow{v}}(v) \wedge \lambda > 0 \wedge P_i(p + \lambda v) = 0 \wedge \\ \wedge [(0 < \lambda' < \lambda) \rightarrow P_j(p + \lambda' v) \neq 0]).\end{aligned}$$

Clearly, this formula is true if and only if the first intersection point of the ray defined by $p, \overrightarrow{v}$ with the set of zeroes of $P_i$ is not further from $p$ than the first intersection point of this ray with the set of zeroes of $P_j$. Its truth can be determined using a quantifier-elimination procedure.

The formula involves $2d' + 5$ polynomials of degree bounded by $\overline{d}$ and logarithmic height bounded by $\max\{\tau, \tau'\}$. It contains $2n + 1$ existentially quantified variables and one universally quantified variable, with only one alternation of quantifiers. Hence, by Theorem 14.21 in [BPR06], its truth can be decided in $\overline{d}^{\mathcal{O}(n)}$ arithmetic operations between integers of logarithmic height bounded by $\max\{\tau, \tau'\}\overline{d}^{\mathcal{O}(n)}$.

Since both tasks are realized at most $n$ times for each visited hypercube, and since the number of hypercubes visited to answer one query is bounded by $kn$, we conclude that the local ray shooting query algorithm performs $\overline{\tau} n d'^{\mathcal{O}(1)} + k\overline{d}^{\mathcal{O}(n)}$ arithmetic operation between integers of logarithmic height bounded by $\overline{\tau}\overline{d}^{\mathcal{O}(n)}$. $\square$

## 3.6 Discussion

We have studied three data structures used to solve the point location problem for a family of polynomials: the one obtained from the dialytic method; the data structure obtained from a Cylindrical Algebraic Decomposition of the space; and a third data structure that is based on a uniform partition of the unit cube in $\mathbf{R}^n$.

All of them require doubly-exponential time to be constructed and doubly-exponential space to be stored.

From a geometric viewpoint the most important difference is that the last two data structures, as well as the original one constructed by Meiser for the linear case, achieve much more than just the determination of the face of the arrangement $\mathcal{A}(\mathcal{P})$ containing a query point $x \in \mathbf{R}^n$. They allow us to obtain information concerning a *simple* neighborhood of $x$.

Using Meiser's original data structure (see Section 2.1.3) for a family $\mathcal{P}$ of linear polynomials, it is easy, not only to locate the query point $x \in \mathbf{R}^n$ among the different faces of the arrangement $\mathcal{A}(\mathcal{P})$ but also to obtain a simplex $\Delta_x$, containing $x$, that is not cut by any of the polynomials in the family $\mathcal{P}$. This simplex $\Delta_x$ can be given in parametric form, what allows to move the point $x$ inside it without altering the sign conditions satisfied by $\mathcal{P}$ at this point.

Using the CAD data structure (see Section 3.3), we can obtain a *Tarski cell* containing the query point, *i.e.* a contractible set of bounded description size (here, the bound depends only on the discrete parameters of the input). A parametrization of such a cell using Nash (*e.g.*, rational) functions is possible.

Finally, the construction of Section 3.4 for generic families yields, for a given query point $x \in \mathbf{R}^n$, an hypercube containing $x$ and the list of polynomials in $\mathcal{P}$ that cut this hypercube. These three data structures give local information about a *topologically trivial* neighborhood of the query point.

On the other hand, the case of the dialytic method given in Section 2.3 is completely different. It has the best query time without requiring any genericity hypothesis but does not allow us to obtain local information. Using the notation from Section 2.3, if we take the polynomial preimage $\Delta'_x$ of the simplex $\Delta_{G(x)}$ obtained in $\mathbf{R}^k$ by Meiser's algorithm to $\mathbf{R}^n$ via $G^{-1}$ we obtain the semi-algebraic set $\Delta'_x$ containing $x$ whose topology can be highly nontrivial and, without considerable computational effort, generally unknown. In particular, the dialytic method does not provide, in contrast with the other methods, a simple geometrical and topological structure with the help of which a set $\Delta'_x$ containing $x$ can be determined allowing to easily find a parametric representation of the query point.

In this sense, the dialytic method provides less than the other studied methods. Hence, it is not surprising that the query time for this method is smaller.

It is precisely this *local* information concerning a neighborhood of the query point that allows to solve efficiently the local ray shooting problem using the last introduced data structure. We remark that also the genericity hypothesis plays an important role in our local ray shooting algorithm.

Under this genericity hypothesis on the family $\mathcal{P}$, our algorithm for the ray shooting problem may be adapted to the CAD data structure. The reasons recently exposed, make us unable to apply a similar method to the data structure obtained from the dialytic method in order to solve this problem efficiently.

We comment on another approach to the ray shooting problem for a family of polynomials that deserves further research (cf. [AE99]). A ray in $\mathbf{R}^n$ can be represented as a point in the space $\mathbf{R}^n \times S^{n-1}$. This space can be partitioned into cells so that all the points in one cell correspond to rays that hit the same object first. In this way, the ray shooting problem in $\mathbf{R}^n$ becomes a point location problem in $\mathbf{R}^n \times S^{n-1}$.

Finally, we remark that this chapter provides a certified example of a well known folklore result: the preconstruction of big databases allows fast queries; there is a trade off between the size of the database and the query time.

# 4

# Quantifier Elimination using Intrinsic Data Structures: the Linear Case

**Abstract.** In this chapter, we introduce the notion of *intrinsic description* of a linearly-constructible set and study the complexity of quantifier-elimination methods in a computational model where the output is required to be an intrinsic description of the underlying set. We introduce a quantifier-elimination algorithm in this model. It turns out that our elimination algorithm has a doubly-exponential-time complexity in the worst case, when the complexity is measured in terms of syntactic parameters (number of polynomials, number of quantifier alternations, number of variables). We show that in our computational model, our algorithm is optimal, *i.e.*, we prove a doubly-exponential lower bound in the number of quantifier alternations of the input formula.

Remarkably, we obtain simply-exponential complexity bounds on intrinsic geometric parameters of the input problem. Thus, our algorithm distinguishes between well-posed and ill-posed problems and can be inscribed in the new generation of algorithms which take also into account intrinsic, semantic invariants of the input in order to measure the complexity of the procedure.

## 4.1 Introduction

Given a quantified first-order formula, *to eliminate the quantifiers* means to give a quantifier-free equivalent formula. The goal of the present chapter is to analyze the behavior of quantifier-elimination algorithms that proceed block by block when the intermediary data obtained after the elimination of each block is required to be *intrinsic* to the geometry of the input. In this way, we fix a *software architecture*.

Since the complexity swell that takes place in the process of quantifier elimination over **R** or **C** can be already observed in the linear case, we restrict ourselves to this simpler framework in our analysis. See the Introduction to this thesis for historical notes concerning the algorithmic aspects of the elimination of quantifiers in the elementary theory of the reals.

In Section 4.2, we introduce the notion of *intrinsic description* of constructible sets. Afterwards, we define their canonic descriptions, and we introduce the computational model and the data structures used to store these descriptions. We further prove that canonic descriptions are intrinsic.

In Section 4.3, we present our quantifier-elimination algorithm. After eliminating one quantifier block, our method cleans the syntactic description thus obtained, reducing it to its canonic description. In this sense, our method follows the philosophy of the Kronecker algorithm for geometric elimination, described in [GLS01, DL07]. It turns out that the elimination algorithm we are going to introduce has a doubly-exponential-time complexity in the worst case, when the complexity is measured classically (using syntactic measures). Nevertheless, proceeding in this way, we obtain simply-exponential complexity bounds on intrinsic geometric parameters of the input problem. Thus, our algorithm distinguishes between well-posed and ill-posed problems and can be inscribed in the new generation of algorithms which take also into account intrinsic, semantic invariants of the input in order to measure the complexity of the procedure.

Finally, in Section 4.4, we prove that our complexity bounds are tight for the previously fixed kind of data structures and discuss the related work on lower bounds.

## 4.2 Theoretical Foundations

### 4.2.1 Constructible Sets

We work over a fixed field, **K**, and we denote $\mathbb{A}^n$ the $n$-dimensional affine space over **K**.

**Definition 4.2.1.** A subset $S \subset \mathbb{A}^n$ is called *affine* if it is the solution space of a system of linear equations with coefficients in $\mathbf{K}$.

A subset of $\mathbb{A}^n$ is called *linearly constructible*, or simply *constructible* in this chapter, if it can be expressed from affine sets using unions, intersections and complements.

For subsets in $\mathfrak{C}$, the notion of *irreducible* subset used in algebraic geometry agrees with our notion of affine subset. In the sequel, we shall only refer to the Zariski topology of the affine spaces which we are going to consider. If $C$ is a constructible subset of $\mathbb{A}^n$, we denote by $\bar(C)$ its closure. Any closed linearly constructible subset $C \subseteq \mathbb{A}^n$ can be uniquely decomposed into irreducible components $C = \cup_{i=1}^{d} C_i$. If $C = \cup_{i=1}^{d} C_i$ is not the union of any proper subset of $\{C_1, ..., C_d\}$, we say that $\{C_1, ..., C_d\}$ is an *irredundant decomposition* of $C$ and that the degree of $C$ is $d$, denoted $\deg(C) = d$.

Finally, given a constructible set $C \subset \mathbb{A}^n$, we define the dimension of $C$ as the maximum of the dimensions of the irreducible components of an irredundant decomposition of its closure, and denote it by $dim(C)$. By definition $dim(\emptyset) := -\infty$. The following statement is an instantiation of Proposition 5.40 in [BPR06] to the linear case and is easy to prove.

**Lemma 4.2.2.** If $C \in \mathfrak{C}$, $C \neq \emptyset$ then $dim(\overline{C} \setminus C) < dim(C)$. □

### 4.2.2 The Language $\mathcal{L}$

Our aim is to associate with any constructible set a canonic description. Let us first remark that the closure of any constructible set owns already a natural description, namely, the list of its irreducible components.

This is the starting point of the following constructions which will lead us to the canonic description of arbitrary constructible sets.

The difficulty we face is the following. Affine sets themselves seem not to own a natural description. Let us analyze the most basic case, an hyperplane in $\mathbb{A}^n$. It can be described naturally with one equation, but all the non-zero multiples of this equation describe the same hyperplane. Which one shall we choose as the *canonic* one? Here it is possible, fixing an order on the variables, to determine a *standard* equation adding a rule like: the main coefficient with respect to the given order of variables has to be one. However, this viewpoint introduces an artifact (the variables order) which has no a priori geometric meaning and is therefore not intrinsic in our sense.

We face a more complex situation when trying to determine the canonic equation system corresponding to a given affine set of higher codimension. For example, a straight line in a three dimensional ambient space has no intrinsic equation system because every description of the line as the solution space of an equation system is, geometrically speaking, a description of the line as the intersection of two hyperplanes and there is no natural criterion known to choose the hyperplanes canonically. As in the previous case, a *standard*, non-intrinsic, equation system can be chosen fixing an order on the variables. The Gröbner basis approach illustrates how this can be done. We shall not go into the details.

In order to overcame this problem of indeterminacy we are going to introduce for each affine set a special predicate symbol. Doing this, we shall obtain a unique and canonical description for each linearly constructible subset $C \in \mathfrak{C}$ such that the atomic formulas involved in these descriptions depend only on the geometry of $C$.

**Language $\mathcal{L}$.** We suppose that the predicates of our first-order language are the characteristic functions of affine sets contained in $\mathfrak{C}$. We shall then be able to associate with any constructible set a canonic description.

For an affine set $S$, we denote by $\Delta_S$ the atomic formula that describes $S$ and we call $\Delta_S$ the *atomic description* of $S$. We build the rest of the descriptions—first-order formulas—from these atomic formulas. We denote by $\mathcal{L}$ the first-order language whose vocabulary consists only of the predicates $\{\Delta_S \mid S \text{ is an affine subset in } \mathbb{A}^n, \text{ for some } n\}$.

The first-order formulas in the language $\mathcal{L}$ are called $\mathcal{L}$-formulas. Given an $\mathcal{L}$-formula $\varphi$, we denote by $\deg(\varphi)$ the number of atomic descriptions in the formula $\varphi$, counted with repetitions. We remark that, defined in this way, the notion of degree of a formula is an algebraic avatar of the syntactic notion of formula length.

### 4.2.3 A Computational Model for Intrinsic Data

We use three data types with their respective constructors and queries to express our intrinsic elimination algorithms. We denote them by $\mathcal{D}_{Af}, \mathcal{D}_{\mathbf{N}}$ and $\mathcal{D}_{list}$. We briefly describe them in the sequel.

In agreement with our first-order language $\mathcal{L}$, the data type $\mathcal{D}_{Af}$ stores atomic descriptions of affine sets. For the sake of simplicity, given an affine set $S$ we also denote by $\Delta_S$ the instance of $\mathcal{D}_{Af}$ that stores the description of $S$. No confusion arises from using the same symbol for atomic first-order predicates and $\mathcal{D}_{Af}$ instances. When needed, we write $\mathcal{D}^n_{Af}$ to emphasize the ambient space dimension.

To deal with the dimensions of constructible sets in the algorithms we need other data type, denoted by $\mathcal{D}_{\mathbf{N}}$, that represents natural numbers. Finally, we have the data type of the list, $\mathcal{D}_{list}$, whose instances represent finite (possibly mixed) sequences of $\mathcal{D}_{Af}, \mathcal{D}_{\mathbf{N}}$ and $\mathcal{D}_{list}$ instances. The reader may think of an instance of $\mathcal{D}_{list}$ as a stack or a linked list.

For $1 \leq i_1 < ... < i_m \leq n$, we denote by $(x_1, \ldots, \widehat{x_{i_1}}, \ldots, \widehat{x_{i_m}}, \ldots, x_n)$ the $n-m$ tuple composed of the coordinates $x_1, ..., x_n$ excluding $x_{i_1}, ..., x_{i_m}$, and by $x$ the tuple $(x_1, ..., x_n)$.

Now we describe some operations on this data types.

- Intersection: $\cap : \mathcal{D}^n_{Af} \times \mathcal{D}^n_{Af} \to \mathcal{D}^n_{Af}$,

$$\cap(\Delta_S, \Delta_T) := \Delta_{S \cap T}.$$

- For $1 \leq i_1 < ... < i_m \leq n$, Projections: $\pi^n_{i_1,...,i_m} : \mathcal{D}^n_{Af} \to \mathcal{D}^{n-m}_{Af}$,

$$\pi^n_{i_1,...,i_m}(\Delta_S) := \Delta_{\{(x_1,...,\widehat{x_{i_1}},...,\widehat{x_{i_m}},...,x_n) \in \mathbb{A}^{n-m} \mid \exists x_{i_1},...,\exists x_{i_m} \in \mathbb{A}^m x \in S\}}.$$

- Dimension: $dim : \mathcal{D}_{Af} \to \mathcal{D}_{\mathbf{N}}$,

$$dim(\Delta_S) := dim(S).$$

- Decrement: $dec : \mathcal{D}_{\mathbf{N}} \to \mathcal{D}_{\mathbf{N}}$,

$$dec(n) := n - 1.$$

These five operations are called *elementary operations.* We further introduce the *elementary tests.*

- Contention ($\subset$): $\mathcal{D}^n_{Af} \times \mathcal{D}^n_{Af} \to \{True, False\}$,

$$(\Delta_S \subset \Delta_T) = True \text{ if and only if } S \subset T.$$

- Order ($<$): $\mathcal{D}_{\mathbf{N}} \times \mathcal{D}_{\mathbf{N}} \to \{True, False\}$,

$$(a < b) = True \text{ if and only if the number represented by } a$$

$$\text{is smaller than the number represented by } b.$$

- Type: $\mathcal{D}_{list} \to \{A, N, L\}$,

$$Type(l) \text{ equals A,N or L if the data type}$$

$$\text{of the data instance } \bar{l} \text{ is } \mathcal{D}_{Af}, \mathcal{D}_{\mathbf{N}} \text{ or } \mathcal{D}_{list}, \text{ respectively.}$$

Our algorithms also have the ability to deal with lists, but we shall not go into the details, as they are completely standard and quite intricate.

An algorithm in our model is specified and programmed in terms of these data types, independently of how they are implemented. Each operation is called an *elementary operation*. Our complexity model is expressed in terms of these elementary operation. The algebraic complexity of our algorithms can be obtained multiplying the number of elementary steps by the number of arithmetic operations they perform. See Section 4.3.4 for a possible implementation of the type $\mathcal{D}_{Af}$ in terms of arithmetic operations. The cost of each elementary operation in this implementation reduces to $M(n) = \mathcal{O}(n^3)$ arithmetic operations, where $n$ is the dimension of the ambient space.

### 4.2.4 Intrinsic Descriptions

**Definition 4.2.3.** Let $H$ be a finite family of affine sets. We define *the descriptive power* of $H$, $\mathcal{D}_H$, as the boolean algebra of sets generated by the elements of $H$. The atoms are the elements of

$$\mathcal{Z}_H := \{Z \in \mathcal{D}_H \mid \exists M \subset H,\, Z = \bigcap_{X \in M} X \cap \bigcap_{X \notin M} X^c \neq \emptyset\}.$$

In particular, the elements of $\mathcal{Z}_H$ form a finite partition of $\mathbb{A}^n$ and every element in $\mathcal{D}_H$ is finite union of elements in $\mathcal{Z}_H$ (compare [Hei83] for the polynomial case).

Clearly, if $Z \in \mathcal{Z}_H$ then $\overline{Z} \in \mathcal{D}_H$.

**Proposition 4.2.4.** If $C \in \mathcal{D}_H$, then $\overline{C} \in \mathcal{D}_H$.

*Proof.* The set $C$ can be written as a union of atoms, $C = \bigcup_{i \le j} Z_i$. Then, since the union is finite, $\overline{C} = \bigcup_{i \le j} \overline{Z_i}$. Thus, $\overline{C} \in \mathcal{D}_H$. □

**Definition 4.2.5.** A *brick* of an $\mathcal{L}$-formula $\varphi$ is the realization of an atomic descriptions appearing in $\varphi$. We denote the set of bricks of $\varphi$ by $B_\varphi$,

$$B_\varphi := \{\mathcal{R}(\Delta) \mid \Delta \text{ is an atomic description appearing in } \varphi\}.$$

We observe that the bricks are affine sets. The descriptive power of $B_\varphi$, namely $\mathcal{D}_{B_\varphi}$, is abbreviated by $\mathcal{D}_\varphi$.

**Definition 4.2.6.** Let $C$ be a constructible set. We define descriptive power of $C$ as the boolean algebra containing those constructible sets that belong to the descriptive power of any description of $C$. In symbols

$$\mathcal{D}_C := \bigcap_{\mathcal{R}(\varphi)=C} \mathcal{D}_\varphi.$$

We observe that $C \in \mathcal{D}_C$.

**Definition 4.2.7.** Let $\varphi$ be an $\mathcal{L}$-formula and let $C = \mathcal{R}(\varphi)$. The formula $\varphi$ is called an *intrinsic description* of $C$ if $\mathcal{D}_\varphi = \mathcal{D}_C$.

In other words, a description $\varphi$ is intrinsic if its descriptive power is minimal.

### 4.2.5 Definition of Filtrations

To define the *canonic description* of a constructible set, we need the geometric results from the present section.

**Definition 4.2.8.** Given $C$ and $D$ two closed sets, we say that $D$ is *strictly contained* in $C$, and denote it by $D \sqsubset C$, if $D \subseteq C$ and $\overline{C \setminus D} = C$.

We show how to decompose a constructible sets into a sequence of closed sets.

**Definition 4.2.9.** A *filtration*, $F = C^{(1)}, \dots, C^{(k)}$, is a finite chain of non-empty closed sets each one strictly contained in the previous one, $C^{(i+1)} \sqsubset C^{(i)}$. The number $k$ is called the *length* of the filtration $F$. For the sake of simplicity, we shall write $C^{(i)} = \emptyset$, for $i > k$.

The *degree* of the filtration $F$ is defined as the sum of the degrees of the closed sets in the chain: $\deg(F) = \sum_{i=1}^{k} \deg(C^{(i)})$.

We denote by $F'$ the filtration $C^{(2)}, \dots, C^{(k)}$ obtained from $F$ by deleting its first element and with $\mathfrak{F}$ the class of all the filtrations. The closed set $C^{(i)}$ is called the $i^{th}$ level of the filtration $F$.

There is a natural way to associate filtrations with constructible sets and viceversa. For this purpose, we consider now the functions $\mathcal{F} : \mathfrak{C} \longrightarrow \mathfrak{F}$ and $\mathcal{E} : \mathfrak{F} \longrightarrow \mathfrak{C}$.

We start defining $\mathcal{F}_C$, the image of $C$ under the map $\mathcal{F}$, by induction on $dim(C)$. If $C = \emptyset$ we define $\mathcal{F}_C$ as the empty chain. Let us suppose given a non-empty $C \in \mathfrak{C}$ and that $\mathcal{F}$ is defined for all the constructible sets of lower dimension than $C$. Using the fact that $dim(\overline{C} \setminus C) < dim(C)$ (Lemma 4.2.2), we define $\mathcal{F}_C := \overline{C}, \mathcal{F}_{\overline{C} \setminus C}$. It is clear that a filtration cannot have length greater than $n + 1$.

For the definition of $\mathcal{E} : \mathfrak{F} \longrightarrow \mathfrak{C}$, we proceed in a similar way. If $F$ is the empty filtration, we define $\mathcal{E}_F = \emptyset$. Let $F = C^{(1)}, \dots, C^{(k)}$ be a filtration of length $k$ and let us suppose $\mathcal{E}$ defined for all the filtrations of smaller length. We define $\mathcal{E}_F := C^{(1)} \setminus \mathcal{E}_{F'}$.

**Proposition 4.2.10.** $\mathcal{F} \circ \mathcal{E} = Id_{\mathfrak{F}}$ and $\mathcal{E} \circ \mathcal{F} = Id_{\mathfrak{C}}$.

*Proof.* Let $C \in \mathfrak{C}$. We prove by induction on $dim(C)$ that $C = \mathcal{E}_{\mathcal{F}_C}$ holds. If $C = \emptyset$, then $\mathcal{F}_C$ is empty and therefore we have $\mathcal{E}_{\mathcal{F}_C} = C$. Suppose now that $dim(C) \geq 0$, then we have $\mathcal{F}_C := \overline{C}, \mathcal{F}_{\overline{C} \setminus C}$ and therefore $\mathcal{E}_{\mathcal{F}_C} = \overline{C} \setminus \mathcal{E}_{\mathcal{F}_{\overline{C} \setminus C}} = \overline{C} \setminus (\overline{C} \setminus C) = C$, where the second equality is valid by inductive hypothesis.

Now, we prove that for all $F \in \mathfrak{F}$, $F = \mathcal{F}_{\mathcal{E}_F}$ holds. If $F$ is the empty filtration, the equality is immediate. Suppose that the equality holds for all the filtrations of length lower than $k > 0$, and let $F = C^{(1)}, \dots, C^{(k)}$ be a filtration. We have, by the definitions of $\mathcal{E}$ and $\mathcal{F}$, that

$$\mathcal{F}_{\mathcal{E}_F} = \mathcal{F}_{C^{(1)} \setminus \mathcal{E}_{F'}} = \overline{C^{(1)} \setminus \mathcal{E}_{F'}}, \mathcal{F}_{(\overline{C^{(1)} \setminus \mathcal{E}_{F'}}) \setminus (C^{(1)} \setminus \mathcal{E}_{F'})}.$$

Since for all $1 \leq i < k$, $C^{(i)} \sqsubset C^{(1)}$ holds, we have that $\overline{C^{(1)} \setminus \mathcal{E}_{F'}} = C^{(1)}$. Hence, in particular, $(\overline{C^{(1)} \setminus \mathcal{E}_{F'}}) \setminus (C^{(1)} \setminus \mathcal{E}_{F'}) = \mathcal{E}_{F'}$. In this way, we obtain $\mathcal{F}_{\mathcal{E}_F} = C^{(1)}, \mathcal{F}_{\mathcal{E}_{F'}}$. Thus, by inductive hypothesis, $\mathcal{F}_{\mathcal{E}_F} = F$. □

**Definition 4.2.11.** Given a constructible set $C$, we define its *degree* as the degree of its filtration, $\deg(C) := \deg(\mathcal{F}_C)$.

Although the similarity of the conceptualization, this definition does not coincide with the definition of grade given in [Hei83].

The fact that we use the same name and notation for the degree of a formula, of a filtration and of a constructible set should not lead to confusions. As we shall see later, the use of an homonym is justified.

## 4.2.6 Filtrations and Locally Closed Sets

We recall that a set $C \subset \mathbb{A}^n$ is called *locally closed* if it is the intersection of an open and a closed set. We remark that, in particular, if $C$ and $D$ are closed sets and $D \sqsubset C$ then $C \setminus D$ is a locally-closed set.

**Proposition 4.2.12.** If $C \in \mathfrak{C}$ and $\mathcal{F}_C = C^{(1)}, \dots, C^{(k)}$ then

$$C = \bigcup_{i=1}^{\lceil \frac{k}{2} \rceil} (C^{(2i-1)} \setminus C^{(2i)}).$$

Moreover, the union is disjoint.

*Proof.* Let $C \in \mathfrak{C}$ and define $F := \mathcal{F}_C = C^{(1)}, \dots, C^{(k)}$. We proceed by induction on $k$. The result being immediate for $k = 1$ and $k = 2$, we suppose $k > 2$ and that the result holds for filtrations of length bounded by $k - 2$.

By definition of $\mathcal{E}$, $\mathcal{E}_F = C^{(1)} \setminus (C^{(2)} \setminus \mathcal{E}_{F''})$. Using the rule of De Morgan, this can be rewritten as, $\mathcal{E}_F = C^{(1)} \cap (C^{(2)} \cap (\mathcal{E}_{F''})^c)^c = C^{(1)} \cap ((C^{(2)})^c \cup \mathcal{E}_{F''})$, where $Z^c$ denotes the complement of the set $Z$. Observing that $\mathcal{E}_{F''} \subset C^{(1)}$, we obtain $C = \mathcal{E}_F = (C^{(1)} \cap (C^{(2)})^c) \cup \mathcal{E}_{F''} = (C^{(1)} \setminus (C^{(2)})) \cup \mathcal{E}_{F''}$. Thus, the result follows by induction. The union is disjoint because $\mathcal{E}_{F''} \subset C^{(2)}$. □

The following proposition shows that the filtration of a constructible set and that of its complement are pretty similar.

**Proposition 4.2.13.** Let $C$ be a constructible subset of $\mathbb{A}^n$ and let us denote by $F = C^{(1)}, \ldots, C^{(k)}$ the filtration associated with $C$. If $C^{(1)} = \mathbb{A}^n$ then $\mathcal{F}_{C^c} = F'$, else $\mathcal{F}_{C^c} = \mathbb{A}^n, C^{(1)}, \ldots, C^{(k)}$.

*Proof.* Since for any constructible set $C \subset \mathbb{A}^n$, $\overline{C} = \mathbb{A}^n$ or $\overline{C^c} = \mathbb{A}^n$ and the complement is involutary (*i.e.*, $C^{cc} = C$) we can assume with out loss of generality that $\overline{C} = \mathbb{A}^n$.

Hence, the result follows immediately, since

$$C^c = (\mathcal{E}_F)^c = (\mathbb{A}^n \setminus \mathcal{E}_{F'})^c = \mathcal{E}_{F'}.$$

□

We observe that, if $C$ is a constructible set, $\deg(C^c) \leq \deg(C) + 1$.

### 4.2.7 Canonic Descriptions

Now we introduce the concept of *canonic description* of constructible sets. If $C$ is a constructible set, its canonic description, denoted by $\Phi_C$, is a first-order formula in the language $\mathcal{L}$ that expresses the set $C$ using its filtration. We explicitly state the way that this description is stored in the algorithms.

Using the filtrations we can determine each constructible set from a chain of closed sets. In its turn, closed sets can be determined as an irredundant union of affine sets. Hence, we first give the definition of the canonic description for closed sets and then we extend it to arbitrary constructible sets.

**Affine Sets:** If $S$ is an affine set, we define $\Phi_C := \Delta_C$, *i.e.*, its canonic description is its atomic description. In the algorithms, this formula is stored as one instance of the $\mathcal{D}_{Af}$ data type.

**Closed Sets:** Let $C \subset \mathbb{A}^n$ be a closed set. Consider $C = \cup_{i=1}^{d} C_i$ an irredundant decomposition of $C$ as union of affine sets. This decomposition is unique up to the order of its members.

We define

$$\Phi_C := \bigvee_{i=1}^{d} \Delta_{C_i}.$$

A description like this is called a *closed-set canonic description.*

In the algorithms, the formula $\Phi_C$ is stored as the list of atomic descriptions $(\Delta_{C_1}, \dots, \Delta_{C_d})$ using the data type $\mathcal{D}_{list}$.

**Proposition 4.2.14.** Given a list of $d$ affine subsets of $\mathbb{A}^n$, the canonic description of its union can be found in $\mathcal{O}(d\log(d))$ elementary operations, in the sense of Section 4.2.3. The algorithm that performs this operation is called CLEAN-CLOSED.

*Proof.* The algorithm simply orders the $d$ affine sets by decreasing dimension (in $\mathcal{O}(d^2)$ elementary operations) and successively classifies them in *selected* and *discarded*, discarding those sets that are subsets of some previously selected one.

It is necessary to classify $d$ sets and each classification takes less than $d$ elementary operations. The whole algorithm performs $\mathcal{O}(d^2)$ elementary operations.

The output of the algorithm is the list of selected sets. This list describes an irredundant decomposition of the input. □

**Constructible Sets:** Let $C$ be a constructible set and let $\mathcal{F}_C = C^{(1)}, \dots, C^{(k)}$ be its filtration. We define

$$\Phi_C := \bigvee_{i=1}^{\lceil \frac{k}{2} \rceil} (\Phi_{C^{(2i-1)}} \wedge \neg\Phi_{C^{(2i)}}).$$

That $\mathcal{R}(\Phi_C) = C$ is a direct consequence of Proposition 4.2.12. The formula $\Phi_C$ is called the *canonic description* of $C$.

We store this description as a list of canonic descriptions of closed sets, $(\Phi_{C^{(1)}}, \dots, \Phi_{C^{(k)}})$. This data structure will also be used to store the filtrations. Thus, from an algorithmic point of view, filtrations and canonic descriptions of constructible sets are indistinguishable.

We remark that for any constructible set $C$, $\deg(\Phi_C) = \deg(\mathcal{F}_C) = \deg(C)$ holds.

**Proposition 4.2.15.** There exists an algorithm, called NEG, such that, on input a canonic description of a constructible set $C \subset \mathbb{A}^n$, it outputs the canonic description of the set $C^c$, performing two elementary operations.

*Proof.* By Proposition 4.2.13, it is sufficient to verify whether the first element of the filtration associated with $C$ represents the whole ambient space, $\mathbb{A}^n$, or not and remove or add the atomic description $\Delta_{\mathbb{A}^n}$ to the input list, accordingly. □

**Canonic Descriptions are Intrinsic** We prove now that canonic descriptions are intrinsic and we give an alternative characterization of $\mathcal{D}_C$.

A boolean algebra of constructible subsets of the affine space $\mathbb{A}^n$ is called *expanded* if for every closed set in the algebra, all its irreducible components also belong to the algebra.

**Proposition 4.2.16.** The descriptive power of $C$, $\mathcal{D}_C$, is the minimal expanded boolean algebra containing $C$ and closed un Zariski closures. Also, $\mathcal{D}_C = \mathcal{D}_{\Phi_C}$.

*Proof.* Certainly, $\mathcal{D}_C$ is a boolean algebra and it contains $C$. By Proposition 4.2.4, it is closed under closures. It is expanded because all the $\mathcal{D}_\varphi$ in the definition of $\mathcal{D}_C$ (see Definition 4.2.5) are expanded.

To prove that it is the minimal with these properties, consider $\Phi_C$, the canonic description of $C$, and observe that the bricks of $\Phi_C$ are the irreducible components of the sets $C^{(1)} = \overline{C}, C^{(2)} = \overline{\overline{C} \setminus C}, C^{(3)} = \overline{\overline{(\overline{C} \setminus C)} \setminus (\overline{C} \setminus C)}, \ldots$. We remark that these bricks belong to any expanded boolean algebra closed under closures and containing $C$. Hence, $\mathcal{D}_C = \mathcal{D}_{\Phi_C}$ and the proof is complete. □

Since, by last proposition, $\mathcal{D}_C = \mathcal{D}_{\Phi_C}$, we immediately obtain the following corollary.

**Corollary 4.2.17.** The canonic description of a constructible set is intrinsic. □

### Striped Affine Sets

In the next section, we shall present a quantifier-elimination algorithm for linear constructible sets. To eliminate one quantifier block is equivalent to find the description of a projection of a constructible set. As a general fact, this is not an easy task. Now, we introduce a particular kind of locally-closed sets, called striped affine sets, for which the elimination process is easy.

**Definition 4.2.18.** If $C$ is an affine set and $D \sqsubset C$ is a closed set then $C \setminus D$ is called a *striped affine set* or *striped set* for short.

Given a striped affine set $R$, we denote by $R^+$ its closure and with $R^-$ the set $R^c \cap R^+$. It is clear that $R = R^+ \setminus R^-$ and that $R^- \sqsubset R^+$. We also use this notation for locally-closed sets.

Following the canonic description introduced in Section 4.2.7, a striped set $R$ is stored as the pair $(\Delta_{R^+}, \Phi_{R^-})$. We remark that $\deg(R) = \deg(R^+) + \deg(R^-) = 1 + \deg(R^-)$, since $R^+$ is always an affine set.

We observe that any consistent conjunction of equalities and negation of equalities expresses a striped affine set. On the other hand, the canonic description of the striped affine set $R$, $\Phi_R$, can be seen as the conjunction of an atomic formula with a conjunction of negations of atomic formulas (remember that the atomic formulas are equivalent to a conjunction of equalities).

We shall see now that the projection of a striped affine set is easy to compute. This will play a central role in our quantifier-elimination algorithm which will reduce the general problem to the projection of striped affine sets.

Consider an orthogonal projection $\pi : \mathbb{A}^n \to \mathbb{A}^{n-m}$. Given an affine set $A \subset \mathbb{A}^n$ and $x \in \pi(A)$, the dimension of the set $A \cap \pi^{-1}(x)$ is independent of the element $x \in \pi(A)$. We we call it the *dimension of the fibers of $A$ for the projection* $\pi$, and denote it by $\mathrm{df}(\pi, A)$.

**Lemma 4.2.19.** Let $R = R^+ \setminus R^- \subset \mathbb{A}^n$ be a striped affine set with $R^- = \bigcup_{i=1}^d R_i^-$, the irredundant decomposition of $R^-$. Let $\pi : \mathbb{A}^n \to \mathbb{A}^{n-m}$ be an orthogonal projection. Then,

$$\pi(R) = \pi(R^+) \setminus ( \bigcup_{\substack{i \\ \mathrm{df}(\pi,R^+)=\mathrm{df}(\pi,R_i^-)}} \pi(R_i^-)).$$

Also, $\deg(\pi(R)) \leq \deg(R)$.

*Proof.* This equality can be simply proven observing that $\pi(R) \subset \pi(R^+)$ and that a point $x \in \pi(R^+)$ does not belong to $\pi(R)$ if and only if for some $i \leq d$, $\pi^{-1}(x) \cap R^+ = \pi^{-1}(x) \cap R_i^-$. Since $R^+$ and $R_i^-$ are affine sets and $R_i^- \subset R^+$, this last condition is equivalent to $\mathrm{df}(\pi, R^+) = \mathrm{df}(\pi, R_i^-)$ and $x \in \pi(R_i^-)$.

The degree bound follows immediately from the description of $\pi(R)$. Hence, the lemma follows. □

The striped affine sets play a central role in this work because, on the one hand they are easy to project, and on the other hand, as Proposition 4.2.12 shows, every constructible set can be decomposed as the union of striped sets.

### 4.2.8 Particles

When trying to bound the degree of the filtration of a set from an arbitrary description of the set, we need to count how many irreducible sets can be defined from the given family of irreducible sets using intersections. For this task, the algebraic notion of *atom* is not well-suited. We introduce the geometric notion of *particle.*

**Definition 4.2.20.** Let $\mathcal{D}$ be an expanded boolean algebra of constructible sets. We call the irreducible sets in $\mathcal{D}$, *particles of* $\mathcal{D}$.

**Remark 4.2.21.** Let $H$ be a family of irreducible sets in $\mathbb{A}^n$ and let $\mathcal{D}_H$ be its descriptive power. Then, the set of particles of $\mathcal{D}_H$ is given by

$$\{P \in \mathcal{D}_H \mid P = \bigcap_{S \in I} S, P \neq \emptyset, \text{ for some } I \subset H\}.$$

We denote by $\mathrm{pcl}(\mathcal{D})$ the number of particles in $\mathcal{D}$. We observe that there exists a bijective relation between atoms and particles of $\mathcal{D}$ (taking closure). Also note that given $C \in \mathfrak{C}$, the irreducible components of the different level of the filtration $\mathcal{F}_C$ are always particles of $\mathcal{D}_C$. Hence, we obtain immediately obtain the following result.

**Proposition 4.2.22.** The number of particles in $\mathcal{D}_C$ is an upper bound for the degree of $C$. □

For $n, d \in \mathbf{N}$, we define

$$\text{particle-bound}(n, d) := \binom{d}{n} + \binom{d}{n-1} + \ldots + \binom{d}{1} + 1.$$

**Proposition 4.2.23.** Let $H$ be family containing $d > 1$ irreducible sets in $\mathbb{A}^n$, then the number of particles in $\mathcal{D}_H$, $\mathrm{pcl}(\mathcal{D}_H)$, is at most particle-bound$(n, d)$. Also, for $n > 1$, particle-bound$(n, d) \leq d^n$.

*Proof.* Since, by Remark 4.2.21, every particle can be written as intersection of at most $n$ elements in $H$, the bound is immediate from the sum over $k = 0, ..., n$ of the number of different possible intersections of $k$ irreducible sets in $H$. The estimation, for $d, n > 1$, particle-bound$(n, d) \leq d^n$ constitutes an elementary fact in combinatorics. □

In Section 4.4.1 we give an example showing that this exponential behavior can actually occur.

## 4.3 The Algorithms

To eliminate one existential quantifier block before a canonic description, we first convert the canonic description to a description of the same set as a union of striped affine sets. Then, we project each striped set and, finally, we convert the result to its canonic description. In Section 4.3.1, we present the two conversion algorithms. In Section 4.3.2, we introduce the projection algorithms and the complete quantifier-elimination method.

### 4.3.1 Conversion Algorithms

#### Decomposition as Union of Striped Affine Sets

**Locally closed sets as union of striped affine sets:** Any locally closed set, $C = C^+ \setminus C^-$ can be described in a natural way as a striped sets union.

If $C^+$ decomposes as $\cup_{i=1}^d C_i^+$ (an affine sets union) then $C$ decomposes as the striped affine sets union $\cup_{i=1}^d (C_i^+ \setminus (C_i^+ \cap C^-))$. We define

$$\Sigma_{C^+ \setminus C^-} := \bigvee_{i=1}^{d} (\Delta_{C_i^+} \wedge \neg \Phi_{C_i^+ \cap C^-}).$$

This gives a description of $C$ as a striped affine sets union. We will store it, naturally, as the list of striped affine set descriptions that compose it.

**Proposition 4.3.1.** If $C$ is a locally closed set, given $\Phi_{C^+}$ and $\Phi_{C^-}$ it is possible to obtain the formula $\Sigma_{C^+ \setminus C^-}$ in $\mathcal{O}(\deg(C^+)\deg(C^-)^2)$ elementary operations. Moreover, the degree of $\Sigma_{C^+ \setminus C^-}$ is bounded by $\deg(C)^2$.

*Proof.* Each closed set $C_i^+ \cap C^-$ has degree bounded by $\deg(C^-)$. Hence, it is possible to find its canonic description, using the algorithm CLEAN-CLOSED, performing $\mathcal{O}(\deg(C^-)^2)$ elementary operations (see Proposition 4.2.14). The algorithm performs $\deg(C^+)$ of these reductions. Clearly, the degree of each striped affine set is bounded by $\deg(C^-) + 1 \leq \deg(C)$ and there are $\deg(C^+)$ of these sets. Hence, the degree of $\Sigma_{C^+ \setminus C^-}$ is bounded by $\deg(C)^2$. □

We call CONV-LC2SA the algorithm underlying the proof of Proposition 4.3.1

**Constructible sets as union of striped affine sets:** Let $\mathcal{F}_C = C^{(1)}, \ldots, C^{(k)}$ be the filtration associated with $C$. Using $\mathcal{F}_C$ and the Proposition 4.2.12, we know how to describe $C$ as a locally closed sets union: $C = \bigcup_{i=1}^{\lceil \frac{k}{2} \rceil} (C^{(2i-1)} \setminus C^{(2i)})$. By the preceding paragraph, we know how to describe each locally closed set $C^{(2i-1)} \setminus C^{(2i)}$ as a striped set union.

We now define

$$\Sigma_C := \bigvee_{i=1}^{\lceil \frac{k}{2} \rceil} \Sigma_{C^{(2i-1)} \setminus C^{(2i)}} \tag{4.3.1}$$

In this way, any constructible set can be described as a union of striped affine sets. The formula $\Sigma_C$ is called *the striped sets union description* of $C$ and it is stored as the list of striped sets descriptions that compose it. Clearly, $\mathcal{R}(\Sigma_C) = C$.

The following lemma will be useful to obtain the bounds of next corollary.

**Lemma 4.3.2.** Let $t, d_1, ..., d_m$ natural numbers and let $d = \sum_{i=0}^{m} d_i$. Then, $d^t \geq \sum_{i=1}^{m} d_i^t$.

*Proof.* We proceed by induction on $m$. The result being tautological for $m = 1$, we assume $m > 1$. Let us denote by $a$ the natural number $\sum_{i=0}^{m-1} d_i$.

By inductive hypothesis $a^t \geq \sum_{i=0}^{m-1} d_i^t$. Now, the result follows immediately from the binomial formula: $d^t = (a + d_m)^t = \sum_{i=0}^{t} \binom{t}{i} a^i d_m^{t-i} \geq a^t + d_m^t$. This completes the proof. $\square$

**Corollary 4.3.3.** Given $\mathcal{F}_C$, the filtration associated with a constructible set $C$, it is possible to find $\Sigma_C$, the description of $C$ as a striped-set union, performing $\mathcal{O}(\deg(C)^3)$ elementary operations.

Also, $\deg(\Sigma_C) \leq \deg(C)^2$.

*Proof.* Let $d$ be the degree of $C$, and for $1 \leq i \leq \lceil \frac{k}{2} \rceil$, let us denote by $d_i$ the sum of the degrees of $C^{(2i-1)}$ and $C^{(2i)}$. Then, $d = \sum_{i=1}^{\lceil \frac{k}{2} \rceil} d_i$. By Proposition 4.3.1, the $i^{th}$ disjunct in Equation 4.3.1 has degree bounded by $d_i^2$ and can be computed performing $\mathcal{O}(d_i^3)$ elementary operations. Hence, last lemma implies that $\Sigma_C$ has degree bounded by $d^2$ and that the whole procedure requires $\mathcal{O}(d^3)$ elementary operations. $\square$

We call CONV-FILT2SA the algorithm that performs this conversion.

### Canonic Form Conversion

We present the algorithm CONV-SA2FILT that finds the canonic description of a set described as a union of striped sets, obtaining the following result:

**Proposition 4.3.4.** Given $\Sigma$, a striped sets union description of a constructible set $C \subset \mathbb{A}^n$, with $\deg(\Sigma) = d$, it is possible to find the canonic description of $C$, $\Delta_C$ in $\mathcal{O}(d^{2n^2})$ elementary operations, using the algorithm CONV-SA2FILT.

Moreover $\deg(C) \leq \mathrm{pcl}(\mathcal{D}_\Sigma) \leq \deg(\Sigma)^n$.

Proposition 4.3.4 is proved in Section 4.3.3. We observe that the canonic form conversion algorithm is the only step that cannot be done in polynomial time. In a certain sense, this is the bottleneck of our whole quantifier-elimination method. Since the output of this algorithm is determined in advance, it makes sense to ask whether this conversion can be done in polynomial time in the degrees of the input and the output. In Section 4.4 we shall exhibit an argument which indicates a negative answer to this question.

## 4.3.2 Quantifier-Elimination Algorithm

In this section we present the quantifier-elimination algorithm. We first analyze the elimination of one existential block before a striped set description, we then generalize it to constructible sets and, finally, to an arbitrary number of quantifiers blocks.

### Projection of Striped Sets

As first step towards a quantifier-elimination algorithm, we are interested in finding the canonic description of a set defined by $Q\Delta$, where $Q = \exists x_{i_1} \dots \exists x_{i_m}$ is an existential-quantifier block and $\Delta$ is the description of a striped set. The main interest in the striped sets is that they are easy to project, and that its projection is again a striped set.

Given $Q = \exists x_{i_1} \dots \exists x_{i_m}$, an existential-quantifier block, we define the projection $\pi_Q : \mathbb{A}^n \to \mathbb{A}^{n-m}$,

$$\pi_Q(x_1, \dots, x_n) = (x_1, \dots, \widehat{x_{i_1}} \dots, \widehat{x_{i_m}} \dots x_n).$$

We remark that for an affine subset $A$ of $\mathbb{A}^n$, the dimension of the $\pi_Q$-fibers of $A$ can be computed by means of the following form of the Dimension Theorem in linear algebra:

$$df(\pi_Q, A) = dim(A) - dim(\pi_Q(A)).$$

Hence, given the atomic description of $A$, it is possible to compute the description of the projection $\pi_Q(A)$ and its fiber dimension, $df(p, A)$, using $\mathcal{O}(1)$ elementary operations.

We obtain the following result.

**Proposition 4.3.5.** Given an existential-quantifier block $Q$ and a canonic description of a striped set $R$, we can compute the canonic description of $\pi_Q(R)$ in $\mathcal{O}(\deg(R)^2)$ elementary operations. Also, $\deg(\pi_Q(R)) \leq \deg(R)$.

*Proof.* Using Lemma 4.2.19, it is possible to obtain a description of the striped-affine set $\pi_Q(R)$ in $\mathcal{O}(\deg(R))$ elementary operations, but the result is not necessarily in canonic form. Using Proposition 4.2.14, we reduce it to obtain the canonic description of $\pi_Q(R)$ performing other $\deg(R^-)^2$ elementary operations. □

The algorithm that perform this computation is called $PROJ - SA$.

### Projection of Canonically Described Constructible Sets

Combining the results established up to here and using the fact that projections commute with unions, *i.e.*, $\pi_Q(C \cup D) = \pi_Q(C) \cup \pi_Q(D)$, we are now able to compute a description of the projection of any constructible set, given its canonic description.

**Proposition 4.3.6.** Let $C \subset \mathbb{A}^n$ be a constructible set of degree $d$. Given its canonic description and an existential-quantifier block $Q$, it is possible to description as union of striped sets, $\Sigma_{\pi_Q(C)}$, of $\pi_Q(C)$, performing $\mathcal{O}(d^3)$ elementary operations.

Moreover, we have $\deg(\Sigma_{\pi_Q(C)}) \leq d^2$ and $\mathrm{pcl}(\mathcal{D}_{\Sigma_{\pi_Q(C)}}) \leq d^n$.

*Proof.* The algorithm performs two steps. First, it converts the given canonic description to a description as a striped sets union using the algorithm CONV-FILT2SA (described in the Corollary 4.3.3). Second, it projects each striped set in this union using the algorithm PROJ-SA (see Proposition 4.3.5). This gives a description of $\pi_Q(C)$ as a striped-sets union. In the first step, the degree grows at most quadratically. In the second step, the degree does not grow. □

We call PROJ-FILT2SA this algorithm. We remark that it perform a polynomial number of steps in the input's degree.

### Canonical Description of the Projection of Canonically Described Constructible Sets

We remark that the description $\Sigma_{\pi_Q(C)}$ of $\pi_Q(C)$ obtained from the last proposition may be redundant. It is possible to apply the algorithm enounced in Proposition 4.3.4 to *clean* this description finding the canonic description of $\pi_Q(C)$. This allows us to iterate the process and eliminate several quantifiers blocks, as we will see.

We present now the following algorithm, called ELIM-EXIST. In the next proposition, we summarize the results.

**Algorithm 4.3.7** (ELIM-EXIST)**.**

**Input:** $\Phi$ a canonic description of a set $C$ and $Q$ an existential-quantifier block.
**Output:** A canonic description of $\pi_Q(C)$
**Procedure:** $\Sigma :=$ PROJ-FILT2SA$(\Phi, Q)$.
Return CONV-SA2FILT$(\Sigma)$. □

**Proposition 4.3.8.** Let $C \subset \mathbb{A}^n$ be a constructible set of degree $d$, let $\Phi_C$ be its canonic description and let $Q$ be an existential-quantifier block. On input $\Phi_C$ and $Q$, the previous algorithm outputs the canonic description of the set $\mathcal{R}(Q\Phi_C)$ performing $\mathcal{O}(d^{4n^2})$ elementary operations.

Also, $\deg(\pi_Q(C)) \leq d^n$.

*Proof.* The algorithm performs two steps. The first step converts the canonic description given as input to a striped-sets-union description of its projection. By last proposition, this step requires $\mathcal{O}(d^3)$ elementary operations and the degree of the resulting description is at most $d^2$. In the second step, we convert this description to a canonic one. The time bound follows immediately from Proposition 4.3.4.

Composing both algorithms with their respective degree bounds, we obtain a $d^{2n}$ degree bound for the output. However, we can do better. Let $B_C$ be the set of bricks of $\Phi_C$, and let us denote by $\pi_Q(B_C)$ the set

$$\{\pi_Q(B) \mid B \text{ is a brick of } \Phi_C\}.$$

It is clear that $\pi_Q(C)$ belongs to the boolean algebra $\mathcal{D}_{\pi_Q(B_C)}$ (our quantifier-elimination method furnishes a proof of this fact). By Proposition 4.2.23 and Proposition 4.2.22 we have $\deg(\pi_Q(C)) \leq d^n$. □

## Quantifier-elimination Algorithm

Combining the algorithms introduced in the last sections, we obtain finally a quantifier-elimination procedure. As usual, we suppose that the input of our algorithm is in prenex normal form (see [Men97]). We also assume that the quantifier-free part of the input is the canonic description of a constructible set.

Let $Q$ be a list of quantifiers that can be divided into alternated blocks $Q = Q^{(r)} \cdots Q^{(1)}$ and let $\Phi_C$ be the canonic description of the constructible set $C$ of degree $d$. We are interested in the canonic description of the set $\mathcal{R}(Q\Phi_C)$.

For a block $Q^{(i)}$ of universal quantifiers, we write $\widetilde{Q^{(i)}}$ for its associated block of existential quantifiers, in symbols $\widetilde{Q^{(i)}} = \neg Q^{(i)}$. If $Q^{(i)}$ is a block of existential quantifiers, we write $\widetilde{Q^{(i)}}$ for the same block.

When $Q$ is a universal-quantifiers block, the classical equivalence

$$Q\Delta \leftrightarrow \neg\widetilde{Q}\neg\Delta \tag{4.3.2}$$

and the fact that finding the canonic description of the complement of a set given in canonic description is simple, allow us to use the previous algorithm also to eliminate also a block of universal quantifiers, leading to the following results.

**Algorithm 4.3.9** (ELIM-QUANT)**.**

**Input:** $\Delta$ a canonic description of a set $C$ and $Q^{(r)} \cdots Q^{(1)}$ alternated quantifiers blocks.
**Output:** Canonic description of the set $\mathcal{R}(Q\Delta)$
**Procedure:** If ($Q^{(1)}$ is universal) $\Delta :=$ NEG($\Delta$).

```
For ($i = 1, \ldots, r$)
    $\Delta :=$ ELIM-EXIST($\Delta, \widetilde{Q^{(i)}}$).
    $\Delta :=$ NEG($\Delta$).
If ($r \equiv 1 \bmod 2$) $\Delta :=$ NEG($\Delta$).
Return $\Delta$.
```
□

The following results, summarizes the complexity of our algorithm when measured in terms of extrinsic (syntactic) parameters.

**Theorem 4.3.10.** Let $C \subset \mathbb{A}^n$ be a constructible set of degree $d$, let $\Phi_C$ be its canonic description and let $Q$ be a list of quantifiers that can be divided into $r$ alternated blocks $Q^{(1)} \cdots Q^{(r)}$. The previous algorithm finds the canonic description of the set $\mathcal{R}(Q\Delta)$, in $\mathcal{O}(d^{2n^{r+1}})$ elementary operations. The degree of the output is bounded by $\mathcal{O}(d^{n^r})$.

*Proof.* The correctness of the algorithm follows from the Equation 4.3.2 and Proposition 4.3.8. Let us denote by $C_i$ the constructible set $\mathcal{R}(Q^{(i)} \cdots Q^{(1)}\Delta)$ and by $d_i$ its degree. Let us also define $d_0 := d$. Then, for $1 \leq i \leq r$, by Proposition 4.3.8, the $i^{th}$ iteration of the for-loop in the above algorithm takes $\mathcal{O}({d_{i-1}}^{4n^2})$ elementary operations and $d_i \leq d_{i-1}^n$. Hence, $\deg(\mathcal{R}(Q\Delta) = d_r \leq d^{n^r}$ and the last iteration of the for-loop takes $\mathcal{O}(d^{n^{r-1}4n^2}) = \mathcal{O}(d^{4n^{r+1}})$ elementary operations. Thus, the theorem follows. □

Our algorithms has a form that is suitable for an intrinsic version. Here, the parameters that governs the complexity are intrinsic to the geometry of the set and the bound is simply exponential in terms of them.

With the hypothesis and notations from the last proposition, let us denote by $C_i$ the constructible set $\mathcal{R}(Q^{(i)} r Q^{(1)}\Delta)$ and $d_i$ its degree. Defining $\delta = \max\{d_i \mid i = 0, \ldots, r\}$, we obtain the following result.

**Theorem 4.3.11.** The previous algorithm finds the canonic description of the set $C_r$ performing $\mathcal{O}(\delta^{2n^2})$ elementary operations.

This means that it is possible to eliminate $r$ quantifiers blocks in simply-exponential time in an intrinsic geometric parameter. In this sense, our algorithm distinguishes between well-posed and ill-posed problems.

### 4.3.3 Canonic Form Conversion Algorithm

The rest of this section is technical and may be skipped in a first reading. We present the algorithm CONV-SA2FILT that finds the filtration of a set given as a union of striped sets, thus proving the Proposition 4.3.4.

Let $\Sigma$ be a formula, with $\deg(\Sigma) = d$, that expresses a set $C$ as union of $s$ striped sets

$$\Sigma = \vee_{i=1,\ldots,s}(\Delta_{R_i^+} \setminus \Phi_{R_i^-}).$$

We start presenting the algorithm that finds the first two levels, $C^{(1)}$ and $C^{(2)}$, of the filtration $\mathcal{F}_C$.

**Algorithm 4.3.12** (CONV-SA2FILT-L1L2)**.**

**Input:** $\Sigma$, description of $C$ as a union of striped sets $R_1, ..., R_s$.
**Output:** $C^{(1)}$ and $C^{(2)}$, the first two levels of the filtration $\mathcal{F}_C$.
**Procedure:** Define $C^{(1)}$ :=CLEAN-CLOSED($\Delta_{R_1^+}, ..., \Delta_{R_s^+}$). (irreducible components of the closure)

```
Define $C^{(2)} := \emptyset$;
Define $\Phi_S$ :=CLEAN-CLOSED($\Phi_{R_1^-}, ..., \Phi_{R_s^-}$). (the list of potential stripes)

Let $m$ be the number of affine sets in $S$;
($S$ is the union of the affine sets, $S_1, ..., S_m$)

For $i = 1, \ldots, m$
    $C^{(2)}$ :=CLEAN-CLOSED($C^{(2)}$, TRUE-HOLES($\Delta_{S_i}, \Sigma$)).
Return $C^{(1)}, C^{(2)}$.
```
□

In each step of the unique *for* of the previous algorithm, the stripes are reduced as much as possible to find the second level of the filtration. The stripes coming from each $R_i$ are not necessarily components of $C^{(2)}$ since they can be covered by some other $R_j$. The algorithm TRUE-HOLES is in charge of performing this verification.

The algorithm CONV-SA2FILT-L1L2 uses the procedure CLEAN-CLOSED (see Proposition 4.2.14) and the procedure TRUE-HOLES that, given $\Delta_S$ (an atomic description of an affine set $S$) and $\Sigma$ (a description of a striped affine sets union) outputs an irredundant decomposition of the set $\overline{S \setminus \mathcal{R}(\Sigma)}$. We present now this last algorithm.

We observe that it is here where the complexity explosion takes place. It is interesting to remark the analogy with the algorithm that finds the DNF (disjunctive normal form) of the negation of a formula given in DNF. Also in that case, a combinatorial explosion takes place (see [JS00]).

If $T$ is an affine set and $R$ is a striped set we say that $R$ *covers* $T$ *bulkily* if $\overline{R \cap T} = T$.

**Algorithm 4.3.13** (TRUE-HOLES).

**Input:** $\Delta_T$ a description of an affine set $T$; $\Sigma$ description of a striped sets union $R_1, \ldots, R_s$.
**Output:** $\Delta_{T_1}, \ldots, \Delta_{T_m}$, a description of an irredundant decomposition of the set $\overline{T \setminus \mathcal{R}(\Sigma)}$.

```
Procedure: For $i = 1, \ldots, s$
        If $((T \subset R_i^+)$ and $(T \cap R_i^- = \emptyset))$ ($R_i$ covers it completely)
            Return $\Delta_\emptyset$.

Let $g = 0$;
For $i = 1, \ldots, s$
        If $((T \subset R_i^+)$ and $(T \cap R_i^- \sqsubseteq T))$ ($R_i$ covers it bulkily)
            $g = i$.

If $(g = 0)$ (i.e., it is not covered bulkily by any $R_i$)
        Return $\Delta_T$.(then, $T = \overline{T \setminus \mathcal{R}(\Sigma)}$, i.e., $T$ it is a true hole)

(at this point we know that $R_g$ covers $T$ bulkily)
(so the new potential true holes are contained in $T \cap R_g^-$)
Let $m$ be the number of affine sets in $R_g^-$.

$\Delta := \emptyset$.
For $i = 1, \ldots, m$
        $\Delta$ :=CLEAN-CLOSED($\Delta$, TRUE-HOLES($\Delta_{T \cap (R_g^-)_i}, \Sigma$)).
Return $\Delta$.                                                              □
```

The plan for the rest of this section is the following: first, we analyze the complexity of the two algorithms recently presented (TRUE-HOLES and CONV-SA2FILT-L1L2) and prove their correctness. Then, we present the algorithm CONV-SA2FILT (that is the one that converts striped affine sets union description to canonic descriptions) with its corresponding complexity analysis.

**Lemma 4.3.14.** Given $\Sigma$—a description of degree $d$ of a constructible set $C \subset \mathbb{A}^n$ as a striped sets union, and given $\Delta_T$—the description of an affine set $T \subset \mathbb{A}^n$ of dimension $k$, the algorithm TRUE-HOLES returns a description of an irredundant decomposition of $\overline{T \setminus \mathcal{R}(\Sigma)}$ performing $\mathcal{O}(d^{2k+1})$ elementary operations.

*Proof.* If $dim(T) = 0$, then $T$ consists of one point and there are two possible outputs: $T$ and $\emptyset$ depending on whether $T \nsubseteq C$ or $T \subseteq C$. The previous algorithm determines this in $\mathcal{O}(d)$ elementary operations. The correctness, in this case, follows immediately.

Let us suppose now that the lemma is proven for $dim(T) < k$ and consider an input with $dim(T) = k$. Looking at the definition of the TRUE-HOLES algorithm, it is clear that before the line defining $\Delta := \emptyset$ the algorithm performs $\mathcal{O}(d)$ elementary operations between affine sets. Hence, we only need to bound the time used in $m$ calls to the instruction

$$\Delta = \text{CLEAN-CLOSED}(\Delta, \text{TRUE-HOLES}(\Delta_{T\cap(R_g^-)_i}, \Sigma)).$$

We observe that $m \leq d$ and that we have that $dim(\Delta_{T\cap(R_g^-)_i}) < k$. By the inductive hypothesis, each call to TRUE-HOLES takes less than $\mathcal{O}(d^{2k-1})$ elementary operations. Hence, in the whole for-loop, a total of $\mathcal{O}(d^{2k})$ elementary operations are performed in the recursive calls to TRUE-HOLES.

We observe that, since the output of TRUE-HOLES is a list of particles of the original input family, Proposition 4.2.23 implies that deg(TRUE-HOLES($\Delta_{T\cap(R_g^-)_i}, \Sigma$)) $\leq d^k$ and deg($\Delta$) $\leq d^k$ hold during the execution of the for-loop. Hence, each call to CLEAN-CLOSED takes less than $\mathcal{O}(d^{2k})$ elementary operations. Thus, the complete algorithm performs a total of $\mathcal{O}(d^{2k+1})$ elementary operations.

To prove the correctness of the algorithm, let us first suppose that $\overline{T \setminus \mathcal{R}(\Sigma)}$ is empty. In this case, the algorithm finishes after the first IF, returning $\Delta_\emptyset$.

Secondly, if $\overline{T \setminus \mathcal{R}(\Sigma)} = T$, the algorithm halts after the third IF returning $\Delta_T$.

Hence, let us finally assume that $\emptyset \neq \overline{T \setminus \mathcal{R}(\Sigma)} \neq T$. Thus, $T$ is bulkily covered by some $R_i$ and the correctness follows by inductive hypothesis on the dimension of $T$. From the properties of the CLEAN-CLOSED algorithm it follows that the output is irredundant. □

Now, analyze the complexity of the algorithm CONV-SA2FILT-L1L2.

**Lemma 4.3.15.** Given $\Sigma$, a degree $d$ description of a constructible set $C$ of dimension $k$, as a striped affine sets union, the algorithm CONV-SA2FILT-L1L2 finds the first two levels of the filtration $\mathcal{F}_C$ performing $\mathcal{O}(d^{2k})$ elementary operations.

*Proof.* The first three lines in the definition of the algorithm CONV-SA2FILT-L1L2 can be done using $\mathcal{O}(d^2)$ elementary operations. The unique for-loop in the algorithm can be bounded in the same way as in the previous proof, obtaining the announced bounds.

Since closure commutes with finite unions, the first level of the filtration is the one computed in the first line of the algorithm.

The second level of the filtration is, by definition, equal to $\overline{\overline{C} \setminus C}$. Since, $C = \mathcal{R}(\Sigma)$ and

$$\overline{C} \setminus C \subset \cup_{i=1,\dots,m} S_i,$$

we conclude that the second level of the filtration equals

$$\overline{\cup_i = 1, ..., m(S_i \setminus \mathcal{R}(\Sigma))}.$$

That $C^{(2)}$ is given in canonic form is ensured by the properties of the CLEAN-CLOSED algorithm. Thus, the correctness of the algorithm follows from the correctness of TRUE-HOLES and the proof is complete. □

We present, finally, the algorithm that finds the filtration of a space given as a striped sets union.

**Algorithm 4.3.16** (CONV-SA2FILT)**.**

```
Input: $\Sigma$, description of a constructible set $C$ as tripped set union.
Output: $\mathcal{F}_C$.
Procedure: Let $i = 1$.
Let $\Phi = \Sigma$.
While ($\Phi \neq \emptyset$)
        ($C^{(i)}, C^{(i+1)}$) =CONV-SA2FILT-L1L2($\Phi$). (compute the next two levels)
        $\Phi = \Sigma \cap C^{(i+1)}$. (compute a description of the remaining points)
        Let $i = i + 2$.
Return ($C^{(1)}, C^{(2)}, \ldots, C^{(i-1)}$).
```
□

**Lemma 4.3.17.** Given $\Sigma$, a degree $d$ description of a constructible set $C$ of dimension $k$, as a striped set union, the algorithm CONV-SA2FILT finds the filtration $\mathcal{F}_C$ performing $\mathcal{O}(d^{2k^2})$ elementary operations.

*Proof.* The time bound is deduced from the following inequalities: for every $i$, $\deg(C^{(i)}) \leq \deg(C) \leq d^k$ and $\deg(\Phi) \leq \deg(C^{(i)}) \cdot \deg(\Sigma) \leq d^{k+1}$, and then—taking into account that the dimension decreases with $i$—the number of elementary operations in each call to the algorithm CONV-SA2FILT-L1L2 can be bounded by $\mathcal{O}(d^{2k}), \mathcal{O}((d^{k+1})^{2(k-2)})$, $\mathcal{O}((d^{k+1})^{2(k-4)}), \ldots, \mathcal{O}(d^{k+1})$ respectively, summing up a total of $\mathcal{O}(d^{2k^2})$ elementary operations. □

Thus, we obtain:

**Proposition 4.3.18.** Given $\Sigma$, a degree $d$ description of a constructible subset $C$ of $\mathbb{A}^n$ as a striped affine sets union, the algorithm $CONV - SA2FILT$ returns the filtration (and then the canonic description) $\mathcal{F}_C$ performing $\mathcal{O}(d^{2n^2})$ elementary operations,

In addition $\deg(C) \leq \mathrm{pcl}(\Sigma) \leq$ particle-bound$(n, \deg(\Sigma)) \leq \deg(\Sigma)^n$. □

### 4.3.4 Our algorithm in other models

We briefly sketch how our algorithm can be implemented in the BSS and Turing machine computational models.

An algorithm in the *BSS* computational model over $\mathbf{K}$ [BCSS98, BSS89] consists of a finite directed connected graph with four types of nodes: *input, computation, branch* and *output*. Computation nodes perform arithmetic operation, branch nodes test equalities. The complexity of an algorithm is a function of the input size and bounds the number of fundamental operation performed from input to output.

#### A matrix library for the data type $\mathcal{D}_{Af}$

In this section we briefly sketch a possible implementation of the data type $\mathcal{D}_{Af}$ with its four elementary operations (see section 4.2.3) in terms of arithmetic operations.

If $T \subset \mathbb{A}^n$ is an affine set, the instance of $\mathcal{D}_{Af}$ describing it has the following information: $n := dim(\mathbb{A}^n)$ the dimension of the ambient space, $r := dim(\mathbb{A}^n) - dim(T)$ the rank of the system and a $k \times (n+1)$-matrix, $M_T$, containing the coefficients of an inhomogeneous linear equation system in row echelon form (*i.e.*, triangulated) whose solution is $T$. We observe that this representation is *natural* when an order in the variables has been fixed.

In this way, the cost of knowing the dimension of the set is one elementary operation ($dim = n - r$). The three other elementary operations in this library can be easily implemented with a complexity bound $M(n) = \mathcal{O}(n^3)$ using Gaussian elimination [1].

The algorithms to intersect and to verify containment are very simple. To compute the intersection of two affine sets, just paste the two matrices, one bellow the other, and triangulate the new system. To verify contention, compute the intersection and verify if the rank has augmented. The projection algorithm is sketched bellow.

**Affine sets projections:** Let $T$ be an affine set in $\mathbb{A}^n$, and consider the matrix in row echelon form $M_T$ of size $r \times (n+1)$ associated with this instance of $\mathcal{D}_{Af}$. Re-triangulating the matrix if necessary, we can assume that we are projecting the first $m$ coordinates.

1. Gaussian elimination, was the first systematic method for solving linear systems of equations. In his famous paper [Str69], Strassen showed that an $n \times n$ matrix can be inverted in $\mathcal{O}(n^{\log 7})$ time. Winograd [Win70] originally proved that matrix multiplication is no harder than matrix inversion, and the converse is due to Aho, Hopcroft, and Ullman [AHU74]. The most asymptotically efficient algorithm for multiplying $n \times n$ matrices to date, due to Coppersmith and Winograd [CW90], has a running time of $\mathcal{O}(n^{2.376})$.

The projection of $T$ (that is an affine subset of $\mathbb{A}^{n-m}$) is determined by the right inferior sub-matrix, $M_{p(T)}$, of size $(n - m - dim(p(T))) \times (n - m + 1))$, corresponding to those equations in which the variables $x_1, \dots, x_m$ do not intervene. The correctness of the method follows from the following remark: being the original system in row echelon form, $x_{m+1}, \dots, x_n$ is a solution for the subsystem $M_{p(T)}$ if and only if it can be extended to a solution, $x_1, \dots, x_n$ for the original system $M_T$. Hence, the subsystem $M_{p(T)}$ corresponds the projection of the set $T$.

### Results in other models

Using the previously described matrix library to perform operations among linear sets, our algorithm can be translated to the BSS context obtaining the following result.

**Theorem 4.3.19.** Let $C \subset \mathbb{A}^n$ be a constructible set of degree $d$, let $\Phi_C$ be its canonic description and let $Q$ be a list of quantifiers that can be divided into $r$ alternated blocks $Q^{(1)} \cdots Q^{(r)}$. Our algorithm in the BSS model finds the canonic description of the set $\mathcal{R}(Q\Delta)$, in $\mathcal{O}(d^{2n^{r+1}} n^3)$ arithmetic operations.

Finally, we remark that our algorithm and its complexity bound results can be transferred mutatis mutandis to the context of Turing complexity, since the coefficient growth of intermediary computations remain under control.

## 4.4 Lower Bounds

### 4.4.1 A Lower Bounds for the Conversion to Canonic Form

We present now two examples showing that the conversion from striped-set union to the canonic description can lead to great complexity swell, independently of the method followed to perform this conversion. In particular, the second example shows that the exponential behavior predicted by the bound on the number of particles can actually occur.

**Example 1.** Consider, for $1 \leq i \leq r$ and $1 \leq j \leq s$, the lines in $\mathbf{K}^2$ given by the equations $R_i : x = i$ and $S_j : y = j$. The realization of the formula

$$\varphi_{rs} := \bigvee_{i=1}^{r} R_i \wedge \neg(\bigvee_{i=1}^{s} S_j)$$

has a canonic description's degree $r(s+1)$, although the formula $\varphi_{rs}$ has degree $r + s$.

In the canonic description of this set it is not possible to refer to the lines $S_j$ because they are not intrinsic. Then, every point of intersection has to be described individually. This example shows that the canonic descriptions might be much longer than simple *natural* descriptions.

**Example 2.** Consider, for $1 \leq i \leq r$, the striped affine sets $T_i := \{(x_1, \dots, x_n) \in \mathbb{A}^n \mid x_i \neq 0 \wedge\ x_i \neq 1\}$ and the set $C := \cup i = 1^r T_i$, given as a union of $n$ striped affine sets of degree three. The set $C$ equals $\mathbf{R}^n \setminus \{0,1\}^n$ and is described by the formula $(x_1 \neq 0 \wedge\ x_i \neq 1) \vee \dots \vee (x_n \neq 0 \wedge\ x_n \neq 1)$ of degree $2n$. The canonic description for $C$ has degree $2^n + 1$.

This kind of examples can arise in the elimination process: a short intrinsic formula can define a set that, when projected, has no short intrinsic description. If we abandoned the idea of using intrinsic data structures, we could avoid the exponential swell in these and other examples. Nevertheless, it is not known whether this is enough to solve the doubly-exponential explosion in Example 3 that follows.

### 4.4.2 A Doubly-Exponential Lower Bound for Quantifier Elimination

The following example follows immediately from Theorem 3 in [BD07]. It is an adaptation to the linear case of the results of Davenport and Heintz [DH88], whose origins can be traced back to Fischer and Rabin [FR74]. It will allow us to prove a doubly-exponential lower bound for quantifier elimination in the linear case over any field of characteristic 0.

We remark that the solution of the next example is related to the Wilkinson - Pochhammer polynomial (see [Par95] and [HM93]).

**Example 3.** We define the predicate $\Phi_0(x,y)$ by the formula $((y = 2x) \vee (y = 2 - 2x))$. We further define

$$\Phi_n(x,y) := \exists z_n \forall x_{n-1} y_{n-1} \begin{pmatrix} (y_{n-1} = y_n \wedge x_{n-1} = z_n) \\ \vee \\ (y_{n-1} = z_n \wedge x_{n-1} = x_n) \end{pmatrix} \to \Phi_{n-1}(x_{n-1}, y_{n-1})$$

If $R$ is a binary relation, we denote by $R^n(x,y)$ the relation $\exists x_1 \cdots x_{n-1}$ $(R(x,x_1) \wedge R(x_1,x_2) \wedge \dots \wedge R(x_{n-1},y))$. If we denote $R_0(x,y)$ the relation defined by $\Phi_0(x,y)$, the formula $\Phi_n(x,y)$ encodes the relation $R_0^{2^n}(x,y)$.

In this way, the formula $\Phi_n(x, \frac{1}{2})$ has length $\mathcal{O}(n)$ (or $\mathcal{O}(n\log(n))$ if we count that the variable $x_n$ needs space $\log(n)$ to be written down) and encodes a set with $2^{2^n}$ points.

This last example shows that any quantifier-elimination algorithm that uses at least one bit of information to describe each irreducible component in a closed set (we call them disjunctive forms, as the canonic form we have previously presented and also as the disjunctive normal form) needs a doubly-exponential amount of time to write down the output. In this way, we obtain the following result.

**Theorem 4.4.1.** Any algorithm in that takes a quantified $\mathcal{L}$-formula as input and outputs a quantifier-free equivalent $\mathcal{L}$-formula, requires, in the worst-case, at least a doubly-exponential amount of time in the number of quantifier alternations to write down the output. $\square$

Also, as shown in [BD07], if sparse or dense representation of polynomials is used and the coefficients are stored classically (as the binary representation of the numbers) the bit length of the output is still doubly exponential.

Related to the notion of quantifier elimination, there is the informal notion of geometric elimination. This notion includes, for instance, polynomial equation solving in algebraically or real-closed fields or, more generally, algebraic varieties in algebraically closed fields. The elimination of an existential-quantifier block preceding the description of a constructible set may be seen as a particular geometric-elimination task. The most efficient quantifier-elimination algorithms require exponential time in the number of variables (in the worst case) to eliminate a single quantifier block. It is not clear whether this phenomenon is due to the algorithms and the data structures or to the intrinsic nature of quantifier and geometric elimination.

One may ask whether this issue changes for elimination procedures based on more flexible data structures, as straight-line programs (see [BCS97]) to store polynomials and boolean-arithmetic circuits (see [vzG86a]) for constructible sets. Complexity improvements based on these data structures were only be achieved for particular instances of elimination problems and only partial results are known about lower complexity bounds for this kind of encodings. Remarkably, it is proven in [HMPW98, GH01] that any geometric elimination algorithm, using the arithmetic-circuit encoding of polynomials and being *geometrically robust* –a property owned by all known symbolic methods–, requires exponential time on infinitely many inputs. These results where generalized to continuous encodings in [CGH$^+$03].

We remark that the definitions of the notions of *geometric elimination procedure* and of *geometrically robust* involve two notions that are related to our notion of *intrinsic description*.

On the one hand, any parametric elimination procedure is *branching parsimonious* (by definition, see [GH01]). This requires that the procedure does not branch on non-intrinsic conditions (in the sense of Section 4.2.4) to solve a given elimination problem .

On the other hand, a *geometrically-robust* parametric elimination procedure produces, by definition, outputs that depend only on the input equation system but not on their circuit representation. This condition can be seen as a mild intrinsicity requisite.

We finally remark that the conclusions of these works suggest that any polynomial-time geometric-elimination algorithm must have a huge topological complexity. Hence, a hypothetical efficient elimination procedure would depend on complicated casuistics.

# 5

# Quantifier elimination for elementary geometry and elementary affine geometry

**Abstract.** Following the tradition of mathematical logic, in this chapter, we introduce new first-order languages for elementary $n$-dimensional geometry and elementary $n$-dimensional affine geometry ($n \geq 2$), based on extending the traditional languages $\mathsf{FO}(\beta, \equiv)$ and $\mathsf{FO}(\beta)$, respectively, with new function symbols. Here, $\beta$ stands for the betweenness relation and $\equiv$ for the congruence relation. We show that the associated theories admit effective quantifier elimination. This is the only chapter not related to complexity theory.

## 5.1 Introduction

### 5.1.1 Origins of the problem

Elementary $n$-dimensional Euclidean geometry, $\mathcal{E}_n$, is the theory dealing with the elementary properties of the $n$-dimensional Euclidean space. In this context, *elementary* means the portion of geometry that can be developed without the help of set-theoretic notions. Tarski's axiom system for this theory, already presented by him in his course given at the Warsaw University in 1926-27 and finally published in [Tar59] and [WST83], is based on two primitive notions: betweenness and equidistance. The theory $\mathcal{E}_n$ is complete but not categorical:

its models are, up to isomorphisms, the $n$-dimensional Cartesian spaces over some real closed fields [Tar59]. The first axiom system based on these primitive notions was proposed by Veblen [Veb04].

As remarked by Szczebra and Tarski [ST79], it is easy to give an axiom system for the elementary theory of $n$-dimensional affine geometry, $\mathcal{A}_n$. It is also a complete theory and the only primitive notion of this theory is the betweenness relation. In her monograph [Szm83], Szmielew showed that this last primitive notion can be replaced by parallelity, leading to a more abstract development of affine geometry, including representation theorems for subsystems of the axiom system of affine geometry.

The interested reader can consult [TG99] and Chapter 7 in [BGKV07] for more references and historical remarks on the development of these theories.

We present two new first-order theories, $\mathcal{E}'_n$ and $\mathcal{A}'_n$, in the languages $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$ and $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$, respectively, that are definitional extensions (see Section 4.6 in [Sho67]) of $\mathcal{E}_n$ and $\mathcal{A}_n$, respectively, and that admit effective quantifier elimination.

Like in Szczebra and Tarski [ST79], the detailed discussion will be restricted to the case $n = 2$, *i.e.*, to the geometry of the plane. We denote by $\mathcal{E}$ and $\mathcal{A}$ the theories $\mathcal{E}_2$ and $\mathcal{A}_2$ respectively. In the last section we shall indicate how our results can be extended to higher dimensions.

There are classical examples of this technique, based on extending the vocabulary with new symbols—expressing properties already definable by quantified formulas in the original language—to obtain a new theory that admits quantifier elimination and has the same expressive power as the original language. For instance, by adding the binary relation symbol "$<$" to the vocabulary $\langle +, \times, 0, 1\rangle$, Tarski[Tar51] obtained a theory, $\mathcal{R}$, for real closed fields that admits quantifier elimination . Another classic example is that of the congruence relations in Presburger arithmetic [End00].

Languages that admit the elimination of quantifiers for elementary algebra and fragments of geometry have been the subjects of several investigations, but as far as we know, no language for elementary geometry which allows quantifier elimination has been proposed. In a way, this omission is surprising, because quantifier elimination is a natural requirement of expressivity for a language.

Quantifier elimination methods have been mainly used to obtain decision procedures. Recently, within the theory of constraint databases [KLP00], they have also been used to evaluate queries. In particular, within the context of spatial databases, the languages $\mathsf{FO}(\beta, \equiv)$ and $\mathsf{FO}(\beta)$ have been proposed [GBG99] as query languages for geometric databases. The results we present here lead to a query evaluation procedure for these query languages. Finally, the problem of finding *minimal* languages for elementary geometry and elementary affine geometry that admit the elimination of quantifier is interesting from a metamathematical viewpoint.

We remark that, sharing some primitive notions, the languages that we obtain are related to the languages used in constructive geometry [MS68, Pam01, Pam08]. One difference is the absence of constant symbols in our language. We remark that in the presence of constant symbols, formulas can express relations that are not invariant under similarity transformations of the plane. Our languages preserve this basic characteristic of Euclidean geometry.

### 5.1.2 Outline and Summary

The chapter is organized as follows. In Section 5.2 we introduce the concepts of affine-invariant and similarity-invariant relation. We also introduce the theories, $\mathcal{R}$, $\mathcal{A}$ and $\mathcal{E}$ with their associated languages $\mathsf{FO}(+, \times, <, 0, 1)$, $\mathsf{FO}(\beta)$ and $\mathsf{FO}(\beta, \equiv)$. Being all three complete theories, we fix a standard model for each and use the fact that a formula holds in this model if and only if it is true in the corresponding theory.

We stress the difference between geometric variables and algebraic variables, and introduce the concept of *translation*. In particular, we recall the existence of a translation from $\mathsf{FO}(\beta, \equiv)$ (and hence, also from $\mathsf{FO}(\beta)$) to $\mathsf{FO}(+, \times, <, 0, 1)$. This translation is based on the fact that the Euclidean plane can be embedded in the Cartesian plane by taking coordinates in a fixed coordinate system.

We recall that $\mathcal{R}$ admits the elimination of quantifiers, and we denote by $\mathfrak{E}_{\mathcal{R}}$ a quantifier-elimination function for this theory. We prove that no finite predicative extension of $\mathsf{FO}(\beta)$ or $\mathsf{FO}(\beta, \equiv)$ admits quantifier elimination.

In Section 5.3, we define the *basic segment-arithmetic functions*, $\oplus$ and $\otimes$, the *affine projection function*, $\pi$, and the two *basic metric functions*, $\pi^{\perp}$ and $\kappa$ (corresponding to the orthogonal projection and the segment construction function), and expand the vocabularies of $\mathsf{FO}(\beta)$ and $\mathsf{FO}(\beta, \equiv)$ adding new function symbols for some of these basic functions, and the 0-ary relation symbol $\top$. The interpretation of the new symbols in the resulting languages, $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ and $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$, are given by $\mathsf{FO}(\beta)$-formulas and $\mathsf{FO}(\beta, \equiv)$-formulas respectively. In this way, the resulting theories, $\mathcal{A}'$ and $\mathcal{E}'$ are a *definitional extension* of $\mathcal{A}$ and $\mathcal{E}$ respectively. This ensures that the new languages have the same expressive power as the original ones and also the existence of translations $\mathcal{B}$ from $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ to $\mathsf{FO}(\beta)$ and $\mathcal{M}$ from $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$ to $\mathsf{FO}(\beta, \equiv)$.

In Section 5.4, we define a translation $\mathcal{S} : \mathsf{FO}(+, \times, <, 0, 1)_{QF,AI} \to \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)_{QF}$, translating any formula in the affine-invariant quantifier-free fragment of $\mathsf{FO}(+, \times, <, 0, 1)$ into the quantifier-free fragment of $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ in such a way that, for any affine-invariant quantifier-free $\mathsf{FO}(+, \times, <, 0,$

1)-formula $\varphi$, $\mathcal{S}(\varphi)$ and $\varphi$ *express the same relation.* The technical difficulty in the construction of this translation is due to the absence of constant symbols in $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ to use as coordinate system and the subsequent need to use some of the variables already involved in the formula as a reference system.

Analogously, in Section 5.5, we define a translation $\mathcal{T} : \mathsf{FO}(+, \times, <, 0, 1)_{QF,SI} \rightarrow \mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)_{QF}$, translating any formula in the similarity-invariant quantifier-free fragment of $\mathsf{FO}(+, \times, <, 0, 1)$ into the quantifier-free fragment of $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$.

In Section 5.6, we define $\mathfrak{E}_{\mathcal{A}'} := \mathcal{S} \circ \mathfrak{E}_{\mathcal{R}} \circ \mathcal{C} \circ \mathcal{B} : \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi) \rightarrow \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)_{QF}$ as the composition of the translations $\mathcal{C}$, $\mathcal{B}$ and $\mathcal{S}$ with the quantifier-elimination function $\mathfrak{E}_{\mathcal{R}}$. The map $\mathfrak{E}_{\mathcal{A}'}$ results to be a quantifier-elimination function for the theory $\mathcal{A}'$. In this sense, we prove that $\mathcal{A}'$ is a *conservative extension* of $\mathcal{A}$ that admits quantifier elimination. Analogously, we prove that the map $\mathfrak{E}_{\mathcal{E}'} := \mathcal{T} \circ \mathfrak{E}_{\mathcal{R}} \circ \mathcal{C} \circ \mathcal{M} : \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi) \rightarrow \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)_{QF}$ is a quantifier-elimination function for the theory $\mathcal{E}'$.

In the last section, we discuss the dispensability of the primitive notions of our new languages. Finally, we briefly indicate how, performing only minor changes in the argumentation, analogous constructions could be carried on for higher-dimensional theories.

For the fluidity of the exposition, we do not prove every geometrical statement in our argumentation. The missing arguments may be filled in using basic tools from analytic geometry.

## 5.2 Preliminaries and definitions

### 5.2.1 Semi-algebraic and geometric relations

Let $\mathbf{R}$ be the set of real numbers and let $\mathbf{E}$ be a model of Tarski's elementary plane geometry isomorphic to $\mathbf{R}^2$. We call $\mathbf{E}$ the *Euclidean plane* and we refer to $\mathbf{R}^2$ as the *Cartesian plane.* We fix an affine coordinate system in $\mathbf{E}$, that is, we fix an origin $O$ and two points $A_1$ and $A_2$ such that these three points are not collinear, and define $C_{O,A_1,A_2}$ as the function from $\mathbf{E}$ to $\mathbf{R}^2$ that maps points to their coordinates with respect to the affine coordinate system $O, A_1, A_2$. We also fix an Euclidean coordinate system in $\mathbf{E}$, that is, we fix two points $E_1$ and $E_2$ such that the segments $\overline{OE_1}$ and $\overline{OE_2}$ are orthogonal and congruent and define $C_{O,E_1,E_2}$ as the function from $\mathbf{E}$ to $\mathbf{R}^2$ that maps points to their coordinates with respect to the Euclidean coordinate system $O, E_1, E_2$.

We shall deal with the following two different kinds of relations.

**Definition 5.2.1.** A $k$-ary *semi-algebraic relation* ($k \geq 1$) is a subset of $\mathbf{R}^k$ that can be described as a boolean combination (intersection, union, complement) of sets of the form

$$\{(x_1, ...., x_k) \in \mathbf{R}^k \mid p(x_1, ..., x_k) > 0\},$$

where $p$ is a polynomial with integer coefficients in the variables $x_1, ..., x_k$.

A $k$-ary *geometric relation* ($k \geq 1$) is a subset of $\mathbf{E}^k$ such that its image under $C^k_{O,E_1,E_2}$ is a semi-algebraic relation of $\mathbf{R}^{2k}$. □

We have allowed only rational coefficients in the definition of semi-algebraic relation for simplicity: as we will see, in this way semi-algebraic relations correspond exactly to the relations expressible in the language $\mathsf{FO}(+, \times, <, 0, 1)$.

We will refer to variables ranging over $\mathbf{E}$ as *geometric variables*, whereas variables ranging over $\mathbf{R}$ will be called *algebraic variables.* Also, for ease of reading, we shall consistently use the letters $o, p, q, r, s, u, v, e_1, e_2, p_1, p_2, ...$, to represent geometric variables, and $a, b, x, y, t, x_1, y_1, x_2, y_2, ...$, for algebraic variables. Variables ranging over the natural numbers $\mathbf{N}$ will be denoted by $i, j, k, l, m\, n$. Finally, with the exception of the already fixed points $O, E_1, E_2, A_1$ and $A_2 \in \mathbf{E}$, we differentiate geometric variables from points in $\mathbf{E}$ writing $p_i$ and $\underline{p_i}$ respectively. In this way, $p_i$ is a geometric variable while $\underline{p_i}$ represents some fixed point in $\mathbf{E}$. Analogously, we write $x_i$ for algebraic variables and $\underline{x_i}$ for fixed elements of $\mathbf{R}$.

### 5.2.2 Affine and similarity transformations of the plane

**Definition 5.2.2.** An *affine transformation* of $\mathbf{R}^2$ is a bijective function $f : \mathbf{R}^2 \to \mathbf{R}^2$, for which there exist $\underline{a_{11}}, \underline{a_{12}}, \underline{a_{21}}, \underline{a_{22}}, \underline{b_1}, \underline{b_2} \in \mathbf{R}$, with

$$f(x, y)^T = \begin{pmatrix} \underline{a_{11}} & \underline{a_{12}} \\ \underline{a_{21}} & \underline{a_{22}} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \underline{b_1} \\ \underline{b_2} \end{pmatrix}.$$

An *affine transformation* of $\mathbf{E}$ is a bijective function $f : \mathbf{E} \to \mathbf{E}$, such that $C^{-1}_{O,A_1,A_2} \circ f \circ C_{O,A_1,A_2}$ is an affine transformation of $\mathbf{R}^2$. □

Since any two affine coordinate systems are equal up to an affine transformation of the plane, the notion of affine transformation of $\mathbf{E}$ is independent of the chosen affine coordinate system $O$, $A_1$, $A_2$.

In particular, translation, rotation, scaling, and reflection over an axis are affine transformations. We remark that our definition of affine transformation coincides with what are usually called non-degenerate affine transformations.

We denote by $\| \cdot \| : \mathbf{R}^2 \to \mathbf{R}$ the norm of points in the Cartesian plane, $\|(x, y)\| = \sqrt{x^2 + y^2}$.

**Definition 5.2.3.** A *similarity transformation* of $\mathbf{R}^2$ is a bijective function $f : \mathbf{R}^2 \to \mathbf{R}^2$, for which there exist $\underline{r} \in \mathbf{R}$, $\underline{r} > 0$ such that for all pairs, $(\underline{x_1}, \underline{y_1})$ and $(\underline{x_2}, \underline{y_2})$, of points in $\mathbf{R}^2$, the following holds:

$$\|f(\underline{x_1}, \underline{y_1}) - f(\underline{x_2}, \underline{y_2})\| = r \cdot \|(\underline{x_1}, \underline{y_1}) - (\underline{x_2}, \underline{y_2})\|.$$

A *similarity transformation* of $\mathbf{E}$ is a bijective function $f : \mathbf{E} \to \mathbf{E}$, such that $C^{-1}_{O,E_1,E_2} \circ f \circ C_{O,E_1,E_2}$ is a similarity transformation of $\mathbf{R}^2$. □

Since any two Euclidean coordinate systems are equal up to a similarity transformation of the plane, the notion of similarity transformation of $\mathbf{E}$ is independent of the chosen Euclidean coordinate system $O$, $E_1$, $E_2$.

In particular, translation, rotation, dilatations, and reflection over an axis are similarity transformations. Clearly, any similarity transformation is an affine transformation but the converse does not hold.

### 5.2.3 Affine-invariant and similarity-invariant relations

Now, we define the concept of an affine-invariant relation.

**Definition 5.2.4.** A $k$-ary geometric relation $P$ is called *affine invariant* if for any tuple $(\underline{p_1}, ..., \underline{p_k})$ in $\mathbf{E}^k$ and any affine transformation $f$ of $\mathbf{E}$, we have that $(\underline{p_1}, ..., \underline{p_k}) \in P$ implies $(f(\underline{p_1}), ..., f(\underline{p_k})) \in P$.

A $2k$-ary semi-algebraic relation $Q$ is called *affine invariant* if for any tuple $(\underline{x_1}, \underline{y_1}, ..., \underline{x_k}, \underline{y_k})$ in $\mathbf{R}^{2k}$ and any affine transformation $f$ of $\mathbf{R}^2$, we have that $(\underline{x_1}, \underline{y_1}, ..., \underline{x_k}, \underline{y_k})$ implies $(f(\underline{x_1}, \underline{y_1}), ..., f(\underline{x_k}, \underline{y_k})) \in Q$. □

We remark that affine-invariant semi-algebraic relations range over pairs of real numbers while affine-invariant geometric relations range over points in the plane $\mathbf{E}$. As an example for previous definition, we consider the geometric relation $\mathsf{L} \subset \mathbf{E}^3$ consisting of triples $(p, q, r) \in \mathbf{E}^3$ that are collinear. Since any affine transformation preserves collinearity, this relation is affine invariant. A finer relation that will play an important role is $\beta$, which consists of all triples $(p, q, r) \in \mathbf{E}^3$ for which $q$ belongs to the closed line segment between $p$ and $r$. Clearly, $\beta$ is also affine invariant. Their semi-algebraic counterparts are subsets of $\mathbf{R}^6$ and can be expressed algebraically, as will be later.

Certainly, not all geometric relations are affine invariant. For example, the unary relation $\{O\}$, containing the origin of the affine coordinate system $O, A_1, A_2$, is not affine invariant.

**Definition 5.2.5.** A $k$-ary geometric relation $P$ is called *similarity invariant* if for any tuple $(\underline{p_1}, ..., \underline{p_k})$ in $\mathbf{E}^k$ and any similarity transformation $f$ of $\mathbf{E}$, we have that $(\underline{p_1}, ..., \underline{p_k}) \in P$ implies $(f(\underline{p_1}), ..., f(\underline{p_k})) \in P$.

A $2k$-ary semi-algebraic relation $Q$ is called *similarity invariant* if for any tuple $(\underline{x_1}, \underline{y_1} ..., \underline{x_k}, \underline{y_k})$ in $\mathbf{R}^{2k}$ and any similarity transformation $f$ of $\mathbf{R}^2$, we have that $(\underline{x_1}, \underline{y_1} ..., \underline{x_k}, \underline{y_k})$ implies $(f(\underline{x_1}, \underline{y_1}), ..., f(\underline{x_k}, \underline{y_k})) \in Q$. $\square$

Since similarity transformations are affine transformations, affine-invariant relations are similarity invariant.

In Euclidean geometry there is no notion of unit length. Hence, no intrinsic metric can be defined in the Euclidean plane. Although, the relation $\equiv$, consisting of all quadruples $(\underline{p}, \underline{r}, \underline{q}, \underline{s}) \in \mathbf{E}^4$ such that the segments $\overline{\underline{pr}}$ and $\overline{\underline{qs}}$ are congruent (*i.e.*, for which the distance between $\underline{p}$ and $\underline{r}$ is equal to the distance between $\underline{q}$ and $\underline{s}$), is a similarity-invariant relation. It gives an example of a similarity-invariant relations that is not affine invariant.

Further examples of affine-invariant (and thus, similarity-invariant) geometric relations concern parallelism and equal ratio. Indeed, if four points define two parallel lines, then the results of any affine transformation applied to them, also define two parallel lines. Also, the ratio of a triple $(\underline{p}, \underline{q}, \underline{r})$ of collinear points, defined (when $\underline{p} \neq \underline{r}$) as $\frac{\|C_{O,A_1,A_2}(\underline{q}) - C_{O,A_1,A_2}(\underline{p})\|}{\|C_{O,A_1,A_2}(\underline{r}) - C_{O,A_1,A_2}(\underline{p})\|}$ and denoted $(\underline{p} : \underline{q} : \underline{r})$, is independent of the affine coordinate system $O, A_1, A_2$ of $\mathbf{E}$. Therefore, the 6-ary geometric relation *equal ratio* $(\underline{p} : \underline{q} : \underline{r}) = (\underline{p'} : \underline{q'} : \underline{r'})$ is affine invariant.

### 5.2.4 The theories $\mathcal{R}$, $\mathcal{E}$, $\mathcal{A}$ and their expressive power

We define the first-order languages $\mathsf{FO}(+, \times, <, 0, 1)$, $\mathsf{FO}(\beta, \equiv)$ and $\mathsf{FO}(\beta)$ and their standard interpretations.

We suppose that first-order formulas are built using the connectives $\neg$ and $\wedge$ and the existential quantifier $\exists$. The symbols $\vee$, $\rightarrow$ and $\forall$ and $\neq$ stand for their usual abbreviations.

**Definition 5.2.6.** Suppose that $\sigma$ is a first-order vocabulary, $S$ is a $\sigma$-structure, and $\psi$ a $\mathsf{FO}(\sigma)$-formula with $m$ free variables. The *relation expressed* by $\psi$ in $S$ is the set of $m$-tuples of elements of $S$ that satisfy $\psi$.

If $k < m$ and $(\underline{s_1}, ..., \underline{s_k})$ is an $k$-tuple of elements of $S$, we define the *relation expressed* by $\psi[\underline{s_1}, ..., \underline{s_k}]$ in $S$ as the set of $(m-k)$-tuples $(\underline{s_{k+1}}, ..., \underline{s_m})$ of elements of $S$ such that $(\underline{s_1}, ..., \underline{s_m})$ satisfy $\psi$. $\square$

Since we consider only one interpretation of each language, we shall use the same symbol for relation/functional symbols and their interpretations, not to overload the notation. We also refer to *the relation expressed by a formula* without reference to the structure considered. As we shall see, the theories that we are going to introduce now express precisely, semi-algebraic, similarity-invariant and affine-invariant geometric relations, respectively.

The language $\mathsf{FO}(\beta)$ is the first-order logic with a vocabulary consisting only of the ternary relation symbol $\beta$. As the standard interpretation for this language, we consider the structure $(\mathbf{E},\beta)$, where variables are assumed to range over the Euclidean plane $\mathbf{E}$ and where $(\underline{p},\underline{q},\underline{r}) \in \beta$ if and only if $\underline{p}$, $\underline{q}$ and $\underline{r}$ are collinear points and $\underline{q}$ belongs to the closed line segment between $\underline{p}$ and $\underline{r}$. In particular, $(\underline{p},\underline{p},\underline{q}) \in \beta$ for any $\underline{p},\underline{q} \in \mathbf{E}$. We denote by $\mathcal{A}$ the first-order theory resulting from this standard interpretation. The next proposition follows immediately from Proposition 5.4 in [GBG99].

**Proposition 5.2.7.** The relations expressible in $\mathcal{A}$, correspond exactly to the affine-invariant geometric relations. $\square$

The language $\mathsf{FO}(\beta,\equiv)$ is the first-order logic with a vocabulary consisting only of the ternary relation symbol $\beta$ and the quaternary relation symbol $\equiv$. As the standard interpretation for this language, we consider the structure $(\mathbf{E},\beta,\equiv)$, where variables are assumed to range over the Euclidean plane $\mathbf{E}$, $\beta$ is defined as before and $(\underline{p},\underline{r},\underline{q},\underline{s}) \in \equiv$ if and only if the segments $\overline{\underline{pr}}$ and $\overline{\underline{qs}}$ are congruent. We denote by $\mathcal{E}$ the first-order theory resulting from this standard interpretation. For the sake of readability and following the tradition, we denote $\equiv (p_i,p_j,p_k,p_l)$ by $p_ip_j \equiv p_kp_l$. The next proposition follows immediately from Proposition 5.5 in [GBG99].

**Proposition 5.2.8.** The relations expressible in $\mathcal{E}$, correspond exactly to the similarity-invariant geometric relations. $\square$

Finally, $\mathsf{FO}(+,\times,<,0,1)$ is a first-order language with a signature consisting of the binary function symbols $+$ and $\times$; the binary relation symbol $<$; and the constant symbols 0 and 1. We call this language *the language of real closed fields*. As its standard interpretation, we consider the structure $(\mathbf{R},+,\times,<,0,1)$, that is, the reals with the well-known functions, relation and constants. We denote by $\mathcal{R}$ the theory resulting from this interpretation, usually called *the first-order theory of the real closed fields*. The next proposition follows from Theorem 2.74 in [BCR98].

**Proposition 5.2.9.** The relations expressible in $\mathcal{R}$, correspond exactly to the semi-algebraic relations. $\square$

Clearly, not any $\mathsf{FO}(+,\times,<,0,1)$-formula expresses a similarity-invariant relation. The formula $x_1 = 0 \wedge y_1 = 0$ exemplifies this. We shall denote by $\mathsf{FO}(+,\times,<,0,1)_{SI}$ the similarity-invariant fragment of $\mathsf{FO}(+,\times,<,0,1)$, *i.e.*, the set of $\mathsf{FO}(+,\times,<,0,1)$-formulas expressing similarity-invariant semi-algebraic relations. Analogously, we denote by $\mathsf{FO}(+,\times,<,0,1)_{AI}$ the affine-invariant fragment of $\mathsf{FO}(+,\times,<,0,1)$.

Also, for any first-order vocabulary $\sigma$, we denote by $\mathsf{FO}(\sigma)_{QF}$ the quantifier-free fragment of $\mathsf{FO}(\sigma)$.

### 5.2.5 Translations

In order to compare relations defined on the Euclidean plane with relations defined on the Cartesian plane, we introduce the following definitions.

Let us call the languages with geometric variables (whose standard interpretation is given over $\mathbf{E}$) *geometric languages*; $\mathsf{FO}(\beta)$ and $\mathsf{FO}(\beta, \equiv)$ are examples of geometric languages.

**Definition 5.2.10.** Let $\varphi$ be a formula in a geometric language expressing the $m$-ary geometric relation $G_\varphi$ ($m \geq 0$) and let $\psi$ be a $\mathsf{FO}(+, \times, <, 0, 1)$-formula expressing the $2m$-ary semi-algebraic relation $A_\psi$. If, for any points $\underline{p_1}, \ldots, \underline{p_m}$ in $\mathbf{E}$, with coordinates $(\underline{x_1}, \underline{y_1}), \ldots, (\underline{x_m}, \underline{y_m})$ with respect to the coordinate system $O, E_1, E_2$,

$$G_\varphi(\underline{p_1}, \ldots, \underline{p_m}) \text{ holds if and only if } A_\psi(\underline{x_1}, \underline{y_1}, \ldots, \underline{x_m}, \underline{y_m}) \text{ holds,}$$

then $\varphi$ and $\psi$ are said to *express the same relation.* □

We remark that, since $\mathsf{FO}(\beta, \equiv)$-formulas express similarity-invariant relations, in the case $\varphi \in \mathsf{FO}(\beta, \equiv)$, the previous definition is independent of the Euclidean coordinate system $O, E_1, E_2$. Analogously, for $\varphi \in \mathsf{FO}(\beta)$ the definition remains invariant if we change $O, E_1, E_2$ to any other affine coordinate system.

The following two fundamental examples are basic results in analytic geometry.

**Example 1.** The $\mathsf{FO}(+, \times, <, 0, 1)$-formula

$$\begin{aligned}&\mathsf{Equidistance_{coord}}(x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4) := \\ &\qquad (x_1 - x_2)^2 + (y_1 - y_2)^2 = (x_3 - x_4)^2 + (y_3 - y_4)^2\end{aligned}$$

and the $\mathsf{FO}(\beta, \equiv)$-formula $p_1p_2 \equiv p_3p_4$ express the same relation.

**Example 2.** Another important example is given by the $\mathsf{FO}(+, \times, <, 0, 1)$-formula

$$\begin{aligned}&\beta_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k) := \\ &\quad [(x_i - x_j)(y_k - y_j) = (x_k - x_j)(y_i - y_j)] \wedge \\ &\qquad [((x_k - x_j)(x_j - x_i) > 0) \vee ((x_k - x_j)(x_j - x_i) = 0)] \wedge \\ &\qquad\qquad [((y_k - y_j)(y_j - y_i) > 0) \vee ((y_k - y_j)(y_j - y_i) = 0)]\end{aligned}$$

and the $\mathsf{FO}(\beta)$-formula $\beta(p_i, p_j, p_k)$. They both express the same relation.

**Definition 5.2.11.** Given two syntactic fragments $\mathcal{L}_1$ and $\mathcal{L}_2$, of two first-order languages with a fixed interpretation, a recursive function $\mathcal{M}$ that maps any $\mathcal{L}_1$-formula, $\varphi$, to a $\mathcal{L}_2$-formula, $\mathcal{M}(\varphi)$, expressing the same relation as $\varphi$ will be called a *translation* between these fragments. $\square$

Using the previous examples, we define a translation $\mathcal{C}$ from $\mathsf{FO}(\beta, \equiv)$ to $\mathsf{FO}(+, \times, <, 0, 1)_{SI}$. For any $i$ in $\mathbf{N}$ and any point $\underline{p_i}$ in $\mathbf{E}$, we denote by $\underline{x_i}$ and $\underline{y_i}$ the first and the second coordinates of $\underline{p_i}$ with respect to our fixed coordinate system $O, E_1, E_2$. Since for all $\underline{p_1}, \underline{p_2}, \underline{p_3}, \underline{p_4} \in \mathbf{E}$, $\mathcal{E} \models \beta[\underline{p_1}, \underline{p_2}, \underline{p_3}]$ if and only if $\mathcal{R} \models \beta_{\mathsf{coord}}[\underline{x_1}, \underline{y_1}, \underline{x_2}, \underline{y_2}\ , \underline{x_3}, \underline{y_3}]$ and $\mathcal{E} \models [\underline{p_1}, \underline{p_2}] \equiv [\underline{p_3}, \underline{p_4}]$ if and only if $\mathcal{R} \models \mathsf{Equidistance}_{\mathsf{coord}}[\underline{x_1}, \underline{y_1}, \underline{x_2},\ \underline{y_2}, \underline{x_3}, \underline{y_3}, \underline{x_4}, \underline{y_4}]$, we immediately obtain a translation, $\mathcal{C}$, defined on the quantifier-free fragment of $\mathsf{FO}(\beta, \equiv)$.

Indeed, $\mathcal{C}$ is obtained by translating $p_i = p_j$ by $x_i = x_j \wedge y_i = y_j$ and by defining $\mathcal{C}(\beta(p_i, p_j, p_k))$ as $\beta_{\mathsf{coord}}(x_i, y_i, x_j, y_j,\ x_k, y_k)$, $\mathcal{C}(p_i p_j \equiv p_k p_l)$ as $\mathsf{Equidistance}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k, x_l, y_l)$, and further $\mathcal{C}(\varphi \wedge \psi)$ as $\mathcal{C}(\varphi) \wedge \mathcal{C}(\psi)$ and $\mathcal{C}(\neg\varphi)$ as $\neg\mathcal{C}(\varphi)$. We extend $\mathcal{C}$, by recursion on the quantifier-depth, to the whole language $\mathsf{FO}(\beta, \equiv)$ defining $\mathcal{C}(\exists p_i \varphi)$ as $\exists x_i \exists y_i \mathcal{C}(\varphi)$.

Using the two previous examples it is easy to prove, by induction on the structure of formulas, that for any $\mathsf{FO}(\beta, \equiv)$-formula $\varphi$ with $m$ free variables and for any points $\underline{p_1}, \ldots, \underline{p_m}$ in $\mathbf{E}$, with coordinates $(\underline{x_1}, \underline{y_1}), \ldots, (\underline{x_m}, \underline{y_m})$ the following holds:

$$\mathcal{E} \models \varphi[\underline{p_1}, \ldots, \underline{p_m}] \quad \text{if and only if} \quad \mathcal{R} \models \mathcal{C}(\varphi)[\underline{x_1}, \underline{y_2}, \ldots, \underline{x_m}, \underline{y_m}].$$

We summarize this result in the following proposition.

**Proposition 5.2.12.** The function $\mathcal{C}$ is a translation from $\mathsf{FO}(\beta, \equiv)$ to $\mathsf{FO}(+, \times, <, 0, 1)_{SI}$. $\square$

In particular, we obtain the following corollary.

**Corollary 5.2.13.** The function $\mathcal{C}$ is a translation from $\mathsf{FO}(\beta)$ to $\mathsf{FO}(+, \times, <, 0, 1)_{AI}$. $\square$

### 5.2.6 Quantifier elimination for $\mathcal{R}$, $\mathcal{E}$ and $\mathcal{A}$

We recall that a first-order theory $\mathcal{S}$ over a vocabulary $\sigma$ *admits quantifier elimination* if, for any formula $\varphi$ in $\mathsf{FO}(\sigma)$, there exist a quantifier-free equivalent $\mathsf{FO}(\sigma)$-formula. A recursive function $\mathfrak{E}_{\mathcal{S}} : \mathsf{FO}(\sigma) \to \mathsf{FO}(\sigma)_{QF}$ is called a *quantifier-elimination function* if for any $\mathsf{FO}(\sigma)$-formula $\varphi$, $\mathfrak{E}_{\mathcal{S}}(\varphi)$ is a quantifier-free $\mathsf{FO}(\sigma)$-formula, equivalent to $\varphi$. If such a function exists, the theory is said to admit *effective* quantifier elimination. We remark that a quantifier-elimination function for $\mathcal{S}$ is a translation from $\mathsf{FO}(\sigma)$ to $\mathsf{FO}(\sigma)_{QF}$.

In the 1930s, Tarski showed that the theory of real closed fields, $\mathcal{R}$, admits effective quantifier elimination (see [Tar51], or [BPR06] for a modern account). In the same article, Tarski used this result and the translation $\mathcal{C}$, from $\mathsf{FO}(\beta, \equiv)$ to $\mathsf{FO}(+, \times, <, 0, 1)$, to give a decision procedure for elementary geometry. We denote by $\mathfrak{E}_{\mathcal{R}}$ a quantifier-elimination function for the theory of real closed fields.

The theories $\mathcal{E}$ and $\mathcal{A}$ do not admit quantifier elimination. We consider, as a first trivial example, the quantified $\mathsf{FO}(\beta)$-sentence $\exists p\,(p = p)$. Since the languages $\mathsf{FO}(\beta)$ and $\mathsf{FO}(\beta, \equiv)$ have no 0-ary predicate symbols and no constant symbols, they admit no quantifier-free sentences. Hence, $\mathcal{A}$ and $\mathcal{E}$ do not admit quantifier elimination.

We prove a much stronger result: it is not possible to obtain a theory that admits quantifier elimination by extending $\mathsf{FO}(\beta)$ (nor $\mathsf{FO}(\beta, \equiv)$) with finitely many relation symbols. For every $k \in \mathbf{N}$, consider the semi-algebraic affine-invariant relation $P^k$ consisting of the triplets of aligned points $(\underline{o}, \underline{p}, \underline{s})$ such that the segment $\overline{\underline{os}}$ is equal to $k$ times the segment $\overline{\underline{op}}$. Clearly, if $k \neq j$ then the relations $P^k$ and $P^j$ are different. This implies that there are countably infinite different ternary affine-invariant relations. By Proposition 5.2.7, all these ternary relations are expressible in $\mathsf{FO}(\beta)$. We denote by $\psi_k$ a $\mathsf{FO}(\beta)$-formula expressing the relation $P^k$.

**Proposition 5.2.14.** Any extension of $\mathsf{FO}(\beta)$ with a finite number of new relation symbols does not admit quantifier elimination.

*Proof.* We suppose than an extension of $\mathsf{FO}(\beta)$ with a finite number of new relation symbols is given. If this extension admitted quantifier elimination, all the different ternary relations $P^i(o, p, q)$, $i \in \mathbf{N}$, would be expressible in this language by a quantifier-free formula. Since there are no constant nor function symbols in the new language, the only terms that can be built in the extended language using the variables $o, p$ and $q$ are the atomic terms $o, p$ and $q$ themselves. Thus, the number of different atomic formulas that can be built using only the given variables is finite. Hence, the number of non-equivalent quantifier-free formulas in this language is finite. Therefore, the extended language cannot express, without quantifiers, all the infinite different relations expressed by the quantified $\mathsf{FO}(\beta)$-formulas $\psi^k$, $k \in \mathbf{N}$. This concludes the proof. $\square$

The previous proof yields immediately the following corollary.

**Corollary 5.2.15.** Any extension of $\mathsf{FO}(\beta, \equiv)$ with a finite number of new relation symbols does not admit quantifier elimination. $\square$

## 5.3 The new languages

In this section, we introduce the two *basic segment arithmetic functions*, $\oplus$ and $\otimes$, the *affine projection function* $\pi$, and the two *basic metric functions*, $\pi^\perp$ and $\kappa$, and define the new languages $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ and $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)$ with their standard interpretations. We prove that they express precisely the affine-invariant and similarity-invariant geometric relations, respectively. We also show the existence of translations $\mathcal{B} : \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi) \to \mathsf{FO}(\beta)$, $\mathcal{M} : \mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa) \to \mathsf{FO}(\beta, \equiv)$.

First, we show how to express some affine-invariant relations in the language $\mathsf{FO}(\beta)$, that we need later on to define these functions.

- The formula:

$$\mathsf{L}(p, q, r) := \beta(p, q, r) \vee \beta(p, r, q) \vee \beta(q, p, r)$$

  expresses that the points $p, q$ and $r$ are collinear. We remark that this is a quantifier-free expression.

- The formula:

$$\mathsf{P}(p, q, r, s) := (\mathsf{L}(p, q, r) \wedge \mathsf{L}(p, q, s)) \vee r = s \vee \\ \forall u(\neg\mathsf{L}(p, q, u) \vee \neg\mathsf{L}(r, s, u))$$

  expresses that the segments $\overline{pq}$ and $\overline{rs}$ are parallel. We remark that the first line expresses that the four points are aligned or that a segment is just a point, in both cases $\overline{pq}$ and $\overline{rs}$ are considered parallel. The second line expresses that no point is collinear with $p$ and $q$ and with $r$ and $s$, at the same time.

We shall also need the following similarity-invariant relation.

- The $\mathsf{FO}(\beta, \equiv)$-formula:

$$\mathsf{R}(p, q, r) := \neg\mathsf{L}(p, q, r) \wedge \exists o(\beta(o, p, q) \wedge or \equiv rq \wedge op \equiv pq)$$

  expresses that the points $p, q$ and $r$ form a non-degenerate triangle with a straight angle at p.

### 5.3.1 The two basic segment-arithmetic functions

Now, we present the formulas that implicitly define the *basic segment-arithmetic functions.*

- **Sum**: the relation "the vector $\overrightarrow{os}$ is the result of the vector sum of $\overrightarrow{op}$ and $\overrightarrow{oq}$" is, certainly, an affine-invariant geometric relation. Thus, by Proposition 5.2.7, there exists a $\mathsf{FO}(\beta)$-formula expressing it. We explicitly state one such formula, based on the *parallelogram rule*:

$$\begin{aligned}\mathsf{Sum}(o,p,q,s) := (o = p \wedge s = q) \vee (o = q \wedge s = p) \vee \\ \exists u \exists v(\neg \mathsf{L}(o,p,u) \wedge \neg \mathsf{L}(o,q,v) \wedge \mathsf{P}(o,p,u,v) \wedge \\ \wedge \mathsf{P}(o,u,p,v) \wedge \mathsf{P}(q,u,s,v)).\end{aligned}$$

  We note that the quantifier-free formula $\mathsf{P}(o,p,q,s) \wedge \mathsf{P}(o,q,p,s)$ expresses the same relation as $\mathsf{Sum}(o,p,q,s)$ when $o,p$ and $q$ are not aligned. The existential quantifiers in the previous formula are needed to express the desired relation when $o,p$ and $q$ are collinear.

  Let $\underline{x_1}, \underline{x_2}, \underline{x_3}$ be three real numbers. Using the coordinates in the fixed coordinate system $O, A_1, A_2$ to express points in $\mathbf{E}$, we consider $\underline{o} = (0,0), \underline{p_1} = (\underline{x_1}, 0), \underline{p_2} = (\underline{x_2}, 0)$ and $\underline{p_3} = (\underline{x_3}, 0)$. Then, the relation $\mathsf{Sum}(\underline{o}, \underline{p_1}, \underline{p_2}, \underline{p_3})$ holds if and only if $\underline{x_1} + \underline{x_2} = \underline{x_3}$ as real numbers. This allows us to translate the semi-algebraic addition into the geometric context.

- **Equal Ratio**: We consider the 5-ary relation: "$o,p$ and $q$ are collinear, $o \neq p$, $o \neq r$ and $s$ is the unique point, collinear with $o$ and $r$, that satisfies $(o : p : q) = (o : r : s)$". This is an affine-invariant geometric relation and since $\mathsf{FO}(\beta)$ is a complete language for these relations, there exists an $\mathsf{FO}(\beta)$-formula expressing it. In fact, it is expressed by the following formula:

$$\begin{aligned}\mathsf{EqualRatio}(o,p,q,r,s) := \mathsf{L}(o,p,q) \wedge \mathsf{L}(o,r,s) \wedge (o \neq p) \wedge (o \neq r) \wedge \\ \exists u \exists v[\mathsf{L}(o,u,v) \wedge \neg \mathsf{L}(o,p,u) \wedge \neg \mathsf{L}(o,r,u) \wedge (o \neq u) \wedge \\ \mathsf{P}(u,r,v,s) \wedge \mathsf{P}(u,p,v,q)).\end{aligned}$$

  That this formula expresses the desired relation is a direct consequence of Thales' theorem (also called *intercept theorem*). The formula is not quantifier free. As in the definition of $\mathsf{Sum}$, the quantifiers are needed to cover the non-generic cases, *i.e.*, when all the points are collinear.

  Let $\underline{x_1}, \underline{x_2}, \underline{x_3}$ be three real numbers. Using the coordinates in the fixed coordinate system $O, A_1, A_2$ to express points in $\mathbf{E}$, we consider $\underline{o} = (0,0), \underline{e_1} = (1,0), \underline{p_1} = (\underline{x_1}, 0), \underline{p_2} = (\underline{x_2}, 0)$ and $\underline{p_3} = (\underline{x_3}, 0)$. We have that, for $\underline{x_2} \neq 0$, $\mathsf{EqualRatio}(\underline{o}, \underline{e_1}, \underline{p_1}, \underline{p_2}, \underline{p_3})$ holds if and only if $\underline{x_1} \cdot \underline{x_2} = \underline{x_3}$

as real numbers. This allow us to translate the semi-algebraic product into the geometric context.

Constructions, similar to $\mathsf{Sum}$ and $\mathsf{EqualRatio}$, to deal with segment arithmetic can be found already in Descartes [Des37], in Hilbert's book [Hil99] and also in [WST83] (see also [Har00] for a contemporary account).

We remark that for every $\underline{o}, \underline{p}, \underline{q} \in \mathbf{E}$ there exists a unique $\underline{s}$ satisfying $\mathsf{Sum}(\underline{o}, \underline{p}, \underline{q}, \underline{s})$. On the other hand, for every $\underline{o}, \underline{p}, \underline{q}, \underline{r}$ there exists at most one $\underline{s}$ satisfying $\mathsf{EqualRatio}(\underline{o}, \underline{p}, \underline{q}, \underline{r}, \underline{s})$.

We conclude that the following two $\mathsf{FO}(\beta)$-formulas define functional relations with respect to their last variable:

- $\mathsf{Sum}(o, p, q, s)$;
- $\mathsf{EqualRatio}(o, p, q, r, s) \vee [(\neg \mathsf{L}(o, p, q) \vee \neg \mathsf{L}(o, r, s) \vee o = p \vee o = r) \wedge s = o]$.

The functions defined by these two $\mathsf{FO}(\beta)$-formulas are called the *basic segment-arithmetic functions* and are denoted by $\oplus : \mathbf{E}^3 \to \mathbf{E}$ and $\otimes : \mathbf{E}^4 \to \mathbf{E}$, respectively.

## 5.3.2 The affine projection function

We present the formula that defines the *affine projection function*.

- **Affine Projection**: We want to express the following relation: "the points $o$, $p$ and $q$ form an affine coordinate system and $s$ is the projection, parallel to $\overline{oq}$, of $r$ on the line $\overline{op}$, or $o$, $p$ and $q$ are aligned and s=o". Being an affine-invariant geometric relation, we know that the relation is expressible in $\mathsf{FO}(\beta)$. Explicitly, we can express it as:

$$\begin{aligned}\mathsf{Projection}(o, p, q, r, s) := \neg \mathsf{L}(o, p, q) \wedge [(\mathsf{L}(r, o, p) \wedge s = r) \vee \\ (\neg \mathsf{L}(r, o, p) \wedge \mathsf{L}(s, o, p) \wedge \mathsf{P}(r, s, o, q))] \vee (\mathsf{L}(o, p, q) \wedge s = o).\end{aligned}$$

We remark that the formula defines a functional relation in $s$. We call this function the *affine projection function* and denote it by $\pi : \mathbf{E}^4 \to \mathbf{E}$.

## 5.3.3 The two basic metric functions

Now, we present the two formulas that implicitly define the *basic metric functions*.

- **Segment Construction**: The axiom of segment construction states that $\exists s(\beta(p, o, s) \wedge os \equiv qr)$. This axiom appears in Tarski's axiomatization of elementary geometry [Tar59] (see also the first congruence axiom in Hilbert's text [Hil99]). We introduce the $\mathsf{FO}(\beta, \equiv)$-formula $\mathsf{SegConst}$ closely related to it:

$$\mathsf{SegConst}(o, p, q, r, s) := (o = p \wedge s = o) \vee \\ (o \neq p \wedge \beta(p, o, s) \wedge os \equiv qr).$$

We remark that this relation expressed by this formula is functional in $s$. If $\underline{o} \neq \underline{p}$, the unique $\underline{s}$ satisfying $\mathsf{SegConst}(\underline{o}, \underline{p}, \underline{q}, \underline{r}, \underline{s})$ is the point in the ray opposite to $\overrightarrow{\underline{op}}$ such that the segments $\overline{\underline{qr}}$ and $\overline{\underline{os}}$ are congruent.

- **Orthogonal Projection**: We consider the 4-ary relation expressed by

$$\mathsf{OrtProj}(o, p, q, s) := (\mathsf{L}(o, p, q) \wedge s = q) \vee \\ \vee (\neg\mathsf{L}(o, p, q) \wedge \mathsf{L}(o, p, s) \wedge (\mathsf{R}(s, q, o) \vee \mathsf{R}(s, q, p))).$$

When $o \neq p$, the last formula expresses that $s$ is the orthogonal projection of $q$ over the line passing through $o$ and $p$. We remark that this formula defines also a functional relation in $s$.

The functions implicitly defined by these $\mathsf{FO}(\beta, \equiv)$-formulas with respect to their last variable are called the *basic metric functions* and are denoted by $\kappa$ and $\pi^{\perp}$, respectively.

### 5.3.4 The language $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ and the theory $\mathcal{A}'$

We expand the language $\mathsf{FO}(\beta)$ with a new function symbol of the corresponding arity for each one of the two basic segment-arithmetic functions and for the affine projection function: $\oplus, \otimes, \pi$. As before, we use the same symbols for the function symbols in the vocabulary and for their interpretations. We also add the 0-ary relation symbol $\top$. In this way, we obtain the expanded language $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$.

Now, we define the standard interpretation of $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$. We interpret variables as points in $\mathbf{E}$, the predicate symbol $\top$ as the constant 0-ary predicate $\mathsf{true}$, the predicate symbol $\beta$ is interpreted as in $\mathsf{FO}(\beta)$ and the three new function symbols as the functions introduced above. We call the resulting theory $\mathcal{A}'$ and remark that it is a definitional extension of $\mathcal{A}$.

Being a geometric language, we shall use Definition 5.2.10 to compare the relations expressed by $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ and $\mathsf{FO}(+, \times, <, 0, 1)$-formulas. Now, the notion of *translation* given in Definition 5.2.11 can be applied, in particular, to mappings from $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ to $\mathsf{FO}(\beta)$ and from the affine-invariant fragment of $\mathsf{FO}(+, \times, <, 0, 1)$ to $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$.

Since the three $\mathsf{FO}(\beta)$-formulas stated at the end of Sections 5.3.1 and 5.3.2 implicitly define the three new functions, the expressive power of the expanded language, $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$, is the same as that of the original one, $\mathsf{FO}(\beta)$, and there exists a translation $\mathcal{B} : \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi) \rightarrow \mathsf{FO}(\beta)$ (for a proof see, e.g., Section 4.6 in Shoenfield's classic book [Sho67]). If $\varphi \in \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ happens to be a $\mathsf{FO}(\beta)$-formula, then $\mathcal{B}(\varphi)$ is just $\varphi$. Essentially, via this map, a formula in $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ is translated to an $\mathsf{FO}(\beta)$-formula replacing each occurrence of a new symbol, by its defining formula in $\mathsf{FO}(\beta)$. This is summarized in the next proposition:

**Proposition 5.3.1.** The map

$$\mathcal{B} : \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi) \rightarrow \mathsf{FO}(\beta)$$

is a translation. □

For the sake of legibility, we shall use the following suggestive notation for terms in the language $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$. We write

- $p \oplus_o q$ for $\oplus(o, p, q)$;
- $q \otimes_{o,p} r$ for $\otimes(o, p, q, r)$; and
- $\pi_{opq}(r)$ for $\pi(o, p, q, r)$.

The next lemma follows directly from the definitions.

**Lemma 5.3.2.** We consider $\underline{x_1}, \underline{x_2} \in \mathbf{R}$ and three affine-independent points $\underline{o}, \underline{e_1}, \underline{e_2} \in \mathbf{E}$. We further denote $\underline{p_1} = (\underline{x_1}, 0)$ and $\underline{p_2} = (\underline{x_2}, 0)$, where the coordinates are taken with respect to the affine coordinate system $\underline{o}, \underline{e_1}, \underline{e_2}$. Then, the standard interpretation of the term $\underline{p_1} \oplus_{\underline{o}} \underline{p_2}$ is the point with coordinates $(\underline{x_1} + \underline{x_2}, 0)$, and the standard interpretation of $\underline{p_1} \otimes_{\underline{o},\underline{e_1}} \underline{p_2}$ has coordinates $(\underline{x_1} \cdot \underline{x_2}, 0)$. □

We further define the following abbreviations:

- $\mathsf{AffCoord}^1_{o,e_1,e_2}(p) := \pi_{o,e_1,e_2}(p)$; and
- $\mathsf{AffCoord}^2_{o,e_1,e_2}(p) := \pi_{o,e_2,e_1}(p) \otimes_{o,e_2} e_1$.

  When the points $\underline{o}, \underline{e_1}, \underline{e_2}$ form an affine coordinate system, it follows immediately that the term $\mathsf{AffCoord}^1_{\underline{o},\underline{e_1},\underline{e_2}}(\underline{p})$ can be interpreted geometrically as the projection, in the direction of $\overline{\underline{oe_2}}$, of the point $\underline{p}$ onto the line $\overline{\underline{oe_1}}$. Under the same hypothesis and denoting by $\underline{p'}$ the projection parallel to $\overline{\underline{oe_1}}$ of the point $\underline{p}$ over the line $\overline{\underline{oe_2}}$, the term $\mathsf{AffCoord}^2_{\underline{o},\underline{e_1},\underline{e_2}}(\underline{p})$ represents the unique point $\underline{q}$ on the line $\overline{\underline{oe_1}}$ that satisfies $(\underline{o} : \underline{e_1} : \underline{q}) = (\underline{o} : \underline{e_2} : \underline{p'})$.

We state this result for further reference.

**Lemma 5.3.3.** We suppose that the points $\underline{o}, \underline{e_1}, \underline{e_2}$ form an affine coordinate system and that the point $\underline{p}$ has coordinates $(\underline{x}, \underline{y})$ in this coordinate system. Then, the term $\mathsf{AffCoord}^1_{\underline{o},\underline{e_1},\underline{e_2}}(\underline{p})$ is naturally interpreted as the point with coordinates $(\underline{x}, 0)$ and the term $\mathsf{AffCoord}^2_{\underline{o},\underline{e_1},\underline{e_2}}(\underline{p})$ as the point with coordinates $(\underline{y}, 0)$, always with respect to the same coordinate system. □

### 5.3.5 The language $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)$ and the theory $\mathcal{E}'$

We expand the language $\mathsf{FO}(\beta, \equiv)$ with a new function symbol of the corresponding arity for each one of the two basic segment-arithmetic function and for the two basic metric functions: $\oplus, \otimes, \pi^\perp$ and $\kappa$. As before, we use the same symbols for the function symbols in the vocabulary and for their interpretations. We also add the 0-ary relation symbol $\top$. In this way, we obtain the expanded language $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)$.

Now, we define the standard interpretation of $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)$. We interpret variables as points in $\mathbf{E}$, the predicate symbol $\top$ as the constant 0-ary predicate $\mathsf{true}$, the predicate symbols $\beta$ is interpreted as the betweenness relation and the four new function symbols as the corresponding functions introduced above. We call the resulting theory $\mathcal{E}'$ and remark that it is a definitional extension of $\mathcal{E}$.

We shall use Definition 5.2.10 to compare the relations expressed by $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)$ and $\mathsf{FO}(+, \times, <, 0, 1)$-formulas. Now, the notion of *translation* given in Definition 5.2.11 can be applied, in particular, to mappings from $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)$ to $\mathsf{FO}(\beta)$ and from the similarity-invariant fragment of $\mathsf{FO}(+, \times, <, 0, 1)$ to $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)$.

Since the four functions $\oplus, \otimes, \pi^\perp$ and $\kappa$ are implicitly definable in $\mathsf{FO}(\beta, \equiv)$ (see Sections 5.3.1 and 5.3.3), the expressive power of the expanded language, $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)$, is the same as that of the original one, $\mathsf{FO}(\beta, \equiv)$, and there exists a translation $\mathcal{M} : \mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa) \to \mathsf{FO}(\beta, \equiv)$. Again, for a proof see, e.g., Section 4.6 in Shoenfield's classic book [Sho67]. We obtain the following Proposition.

**Proposition 5.3.4.** The map

$$\mathcal{M} : \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi) \to \mathsf{FO}(\beta)$$

is a translation. □

We shall use the notation previously introduce for the symbols $\oplus$ and $\otimes$ and we denote by $\pi^\perp_{op}(q)$ the $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)$-term $\pi^\perp(o, p, q)$.

We further define the following abbreviations:

- $\mathsf{EuCoord}^1_{o,e_1,e_2}(p) := \pi^{\perp}_{o,e_1}(p)$;
- $\mathsf{EuCoord}^2_{o,e_1,e_2}(p) := \pi^{\perp}_{o,e_2}(p) \otimes_{o,e_2} e_1$; and
- $\varepsilon(o,p,q) := \kappa(o, o \oplus_{-\pi^{\perp}_{op}(q)} q, o, p)$.

The following result follows immediately from the definitions.

**Lemma 5.3.5.** We suppose that the points $\underline{o}, \underline{e_1}, \underline{e_2}$ form an Euclidean coordinate system and that the point $\underline{p}$ has coordinates $(\underline{x}, \underline{y})$ in this coordinate system. Then, the term $\mathsf{EuCoord}^1_{\underline{o},\underline{e_1},\underline{e_2}}(\underline{p})$ is naturally interpreted as the point with coordinates $(\underline{x}, 0)$ and the term $\mathsf{EuCoord}^2_{\underline{o},\underline{e_1},\underline{e_2}}(\underline{p})$ as the point with coordinates $(\underline{y}, 0)$, always with respect to the same coordinate system. □

**Lemma 5.3.6.** If the points $\underline{o}, \underline{e_1}, \underline{e_2}$ form an affine coordinate system, then the points $\underline{o}, \underline{e_1}, \varepsilon(\underline{o}, \underline{e_1}, \underline{e_2})$ form an Euclidean coordinate system.

*Proof.* Let us suppose that the three points are affine independent. The segments $\overline{\underline{o}\underline{e_1}}$ and $\overline{\underline{o}\varepsilon(\underline{o}, \underline{e_1}, \underline{e_2})}$ and congruent by construction (see the definition of $\kappa$). Since the point $\varepsilon(\underline{o}, \underline{e_1}, \underline{e_2})$ belongs to the line $\overline{\underline{o}(\underline{q} - \pi^{\perp}_{\underline{op}}(\underline{q}))}$ that is perpendicular to the line $\overline{\underline{op}}$, the three points form an Euclidean coordinate system. □

## 5.4 The translation $\mathcal{S}$ of $\mathsf{FO}(+, \times, <, 0, 1)_{QF,AI}$-formulas to $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)_{QF}$-formulas

In the present section, we define a translation from the quantifier-free affine-invariant fragment of $\mathsf{FO}(+, \times, <, 0, 1)$ into the quantifier-free fragment of $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$.

The main result of the present section is the following theorem.

**Theorem 5.4.1.** There exists a translation

$$\mathcal{S} : \mathsf{FO}(+, \times, <, 0, 1)_{QF,AI} \to \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)_{QF}.$$

### 5.4.1 A translation given an affine coordinate system for E

We assume that the variables used in $\mathsf{FO}(+, \times, <, 0, 1)$-formulas are $x_1, y_1, x_2, y_2, ...$ and we define a map (not a translation)

$$\mathcal{S}_{o,e_1,e_2} : \mathsf{FO}(+, \times, <, 0, 1)_{QF,AI} \to \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)_{QF}.$$

The image, $\mathcal{S}_{o,e_1,e_2}(\varphi)$, of an $\mathsf{FO}(+, \times, <, 0, 1)$-formula $\varphi$ in the variables $x_1, y_1, ..., x_m, y_m$, involves the variables $o, e_1, e_2, p_1, p_2, ..., p_m$.

First, we define it for $\mathsf{FO}(+, \times, <, 0, 1)$-terms, by induction on structure of the term, as follows:

- $\mathcal{S}_{o,e_1,e_2}(0) := o$,
- $\mathcal{S}_{o,e_1,e_2}(1) := e_1$,
- $\mathcal{S}_{o,e_1,e_2}(x_i) := \mathsf{AffCoord}^1_{o,e_1,e_2}(p_i)$,
- $\mathcal{S}_{o,e_1,e_2}(y_i) := \mathsf{AffCoord}^2_{o,e_1,e_2}(p_i)$,
- $\mathcal{S}_{o,e_1,e_2}(t_1 + t_2) := \mathcal{S}_{o,e_1,e_2}(t_1) \oplus_o \mathcal{S}_{o,e_1,e_2}(t_2)$ and
- $\mathcal{S}_{o,e_1,e_2}(t_1 \times t_2) := \mathcal{S}_{o,e_1,e_2}(t_1) \otimes_{o,e_1} \mathcal{S}_{o,e_1,e_2}(t_2)$, where $t_1$ and $t_2$ are $\mathsf{FO}(+, \times, <, 0, 1)$-terms.

We remark that the image of an $\mathsf{FO}(+, \times, <, 0, 1)$-term involving the variables $x_1, y_1, \ldots, x_m, y_m$, through the map $\mathcal{S}_{o,e_1,e_2}$, is an $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$-term in the variables $o, e_1, e_2$ and $p_1, \ldots, p_m$. The map $\mathcal{S}_{o,e_1,e_2}$ allows us to translate the two basic semi-algebraic operations ($+$ and $\times$) to the geometric setting, as is proved in the next proposition.

**Proposition 5.4.2.** Let us assume that $\underline{o}, \underline{e_1}, \underline{e_2}$ are three affine-independent points. Let $t$ be a $\mathsf{FO}(+, \times, <, 0, 1)$-term in the variables $x_1, y_1, \ldots, x_m, y_m$ and consider points $\underline{p_1}, \ldots, \underline{p_m}$ in $\mathbf{E}$, with coordinates $(\underline{x_1}, \underline{y_1}), \ldots, (\underline{x_m}, \underline{y_m})$ with respect to the coordinate system $\underline{o}, \underline{e_1}, \underline{e_2}$.

Then, $\mathcal{S}_{o,e_1,e_2}(t)[\underline{o}, \underline{e_1}, \underline{e_2}, \underline{p_1}, ..., \underline{p_m}]$ has coordinates $(t[\underline{x_1}, \underline{y_1}, ..., \underline{x_m}, \underline{y_m}], 0)$ in the affine coordinate system $\underline{o}, \underline{e_1}, \underline{e_2}$.

*Proof.* We prove the proposition by induction in length of the term $t$. If it is an atomic term, the conclusion follows directly from Lemma 5.3.3. It remains to prove the cases $t = r + s$ and $t = r \times s$, where $r$ and $s$ are shorter $\mathsf{FO}(+, \times, <, 0, 1)$-terms. But these cases are direct consequence of Lemma 5.3.2. $\square$

Now, we define the translation of atomic formulas. The case of the relation symbol "$<$" is based in a case analysis. We define

- $\mathcal{S}_{o,e_1,e_2}(t_1 = t_2)$ as $\mathcal{S}_{o,e_1,e_2}(t_1) = \mathcal{S}_{o,e_1,e_2}(t_2)$; and
- $\mathcal{S}_{o,e_1,e_2}(t_1 < t_2)$ as $(\mathcal{S}_{o,e_1,e_2}(t_1) \neq \mathcal{S}_{o,e_1,e_2}(t_2)) \wedge (\varphi_1 \vee \varphi_2 \vee \varphi_3)$, where

$$\begin{aligned}
&\varphi_1 := \beta(\mathcal{S}_{o,e_1,e_2}(t_2), o, e_1) \wedge \beta(\mathcal{S}_{o,e_1,e_2}(t_1), \mathcal{S}_{o,e_1,e_2}(t_2), e_1);\\
&\varphi_2 := \beta(o, e_1, \mathcal{S}_{o,e_1,e_2}(t_2)) \wedge\\
&\qquad\qquad (\beta(\mathcal{S}_{o,e_1,e_2}(t_1), o, e_1) \vee \beta(o, \mathcal{S}_{o,e_1,e_2}(t_1), \mathcal{S}_{o,e_1,e_2}(t_2))); \text{ and}\\
&\varphi_3 := \beta(o, \mathcal{S}_{o,e_1,e_2}(t_2), e_1) \wedge\\
&\qquad\qquad (\beta(\mathcal{S}_{o,e_1,e_2}(t_1), o, e_1) \vee \beta(o, \mathcal{S}_{o,e_1,e_2}(t_1), \mathcal{S}_{o,e_1,e_2}(t_2))).
\end{aligned}$$

Finally, we extend the map $\mathcal{S}_{o,e_1,e_2}$ to the whole quantifier-free fragment of $\mathsf{FO}(+,\times,<,0,1)$ in the natural way, simply translating the conjunctions as conjunctions and negations as negations. The resulting formula always has $o, e_1, e_2$ as extra free variables and one *geometric variable* for each couple of *coordinate-variables* in the original formula.

To lighten the notation, we write $\mathcal{S}_{\underline{o},\underline{e_1},\underline{e_2}}(\varphi)$ for $\mathcal{S}_{o,e_1,e_2}(\varphi)[\underline{o},\underline{e_1},\underline{e_2}]$.

**Proposition 5.4.3.** Let us suppose that $\underline{o},\underline{e_1},\underline{e_2} \in \mathbf{E}$ form an affine coordinate system, and that $\varphi$ is a quantifier-free $\mathsf{FO}(+,\times,<,0,1)$-formula in the variables $x_1, y_1, ..., x_m, y_m$. Consider points $\underline{p_1},\dots,\underline{p_m}$ in $\mathbf{E}$, with coordinates $(\underline{x_1},\underline{y_1}),\dots,(\underline{x_m},\underline{y_m})$ with respect to the coordinate system $\underline{o},\underline{e_1},\underline{e_2}$. Then, $\mathcal{A}' \models \mathcal{S}_{\underline{o},\underline{e_1},\underline{e_2}}(\varphi)[\underline{p_1},...,\underline{p_m}]$ if and only if $\mathcal{R} \models \varphi[\underline{x_1},\underline{y_1},\dots,\underline{x_m},\underline{y_m}]$.

*Proof.* It is sufficient to prove the proposition for atomic formulas. The case of a formula of the form $t_1 = t_2$ is a consequence of Proposition 5.4.2. Let us suppose that $\varphi$ is of the form $t_1 < t_2$.

Let us denote by $\underline{t_1}$ the real number $t_1[\underline{x_1},\underline{y_1},...,\underline{x_m},\underline{y_m}]$, and by $\underline{t_2}$ the real number $t_2[\underline{x_1},\underline{y_1},...,\underline{x_m},\underline{y_m}]$.

Then, the points $\mathcal{S}_{\underline{o},\underline{e_1},\underline{e_2}}(t_1)[\underline{p_1},...,\underline{p_m}]$ and $\mathcal{S}_{\underline{o},\underline{e_1},\underline{e_2}}(t_2)[\underline{p_1},...,\underline{p_m}]$ have, respectively, by Proposition 5.4.2, coordinates $(\underline{t_1},0)$ and $(\underline{t_2},0)$ in the coordinate system $\underline{o},\underline{e_1},\underline{e_2}$.

We can suppose, with out loss of generality, that $\underline{t_1} \neq \underline{t_2}$. Now, we claim that $\underline{t_1} < \underline{t_2}$ holds if and only if $(\varphi_1 \vee \varphi_2 \vee \varphi_3)[\underline{p_1},...,\underline{p_m}]$ holds. Let us suppose that $0 < \underline{t_2} < 1$, the remaining cases ($\underline{t_2} = 0, \underline{t_2} = 1, \underline{t_2} < 0$ and $\underline{t_2} > 1$) can be handled analogously.

Clearly, since $0 < \underline{t_2} < 1$, $(\varphi_1 \vee \varphi_2)[\underline{p_1},...,\underline{p_m}]$ is false. Since, under the above hypothesis, $\underline{t_1}$ is less than $\underline{t_2}$ if and only if "0 is between $\underline{t_1}$ and 1, or $\underline{t_1}$ is between 0 and $\underline{t_2}$", $\varphi_3[\underline{p_1},...,\underline{p_m}]$ holds if and only if $\underline{t_1} < \underline{t_2}$ holds. Hence, we have proved the claim and completed the proof of the proposition. □

### 5.4.2 Finding a basis

The map $\mathcal{S}_{o,e_1,e_2}$ is not a translation because it adds the three new free variables $o, e_1$ and $e_2$. We show how to use the variables $p_1, ..., p_m$ already involved in the formula, considering three different situations:

1. when all the variables represent the same point;

2. when all the variables represent points that are aligned and two are different; and

3. when there are three variables representing affine-independent points.

To distinguish these cases, we define the three $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$-formulas $\mathsf{AffBasis}$, $\mathsf{Aligned}^m$ and $\mathsf{Equal}^m$, and their $\mathsf{FO}(+, \times, <, 0, 1)$-counterparts.

First, we define

$$\begin{aligned}
&\mathsf{AffBasis}(p_1, p_2, p_3) := \neg \mathsf{L}(p_1, p_2, p_3) \quad \text{and}\\
&\mathsf{AffBasis}_{\mathsf{coord}}(x_1, y_1, x_2, y_2, x_3, y_3) :=\\
&\qquad\qquad (x_2 - x_1) \times (y_3 - y_1) - (y_2 - y_1) \times (x_3 - x_1) \neq 0.
\end{aligned}$$

We remark that these are quantifier-free formulas. Both formulas express the same affine-invariant relation: that the three points form an affine coordinate system. That the second formula expresses this relation is a consequence of the fact that the oriented area of the parallelogram with vertices at $(0,0)$, $(a,b)$, $(a+c, b+d)$, and $(c,d)$, is given by the determinant of the matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$.

Further on, we consider, for $m \in \mathbf{N}$, $m \geq 3$, the formulas

$$\begin{aligned}
&\mathsf{Aligned}^m_{\mathsf{coord}}(x_1, y_1, \ldots, x_m, y_m) :=\\
&\qquad\qquad \bigwedge_{1 \leq i < j < k \leq m} \neg \mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k);\\
&\mathsf{Aligned}^m(p_1, \ldots, p_m) := \bigwedge_{1 \leq i < j < k \leq m} \neg \mathsf{AffBasis}(p_i, p_j, p_k);\\
&\mathsf{Equal}^m_{\mathsf{coord}}(x_1, y_1, \ldots, x_m, y_m) := \bigwedge_{2 \leq i \leq m} (x_1 = x_i) \wedge (y_1 = y_i); \quad \text{and}\\
&\mathsf{Equal}^m(p_1, \ldots, p_m) := \bigwedge_{2 \leq i \leq m} (p_1 = p_i).
\end{aligned}$$

We remark that these four formulas express affine-invariant relations. The first two express the same relation, namely, that the points are aligned. The last two also express the same relation, namely, that all the points are the same.

For the remainder of this section, let us suppose that $\varphi(x_1, y_1, \ldots, x_m, y_m)$ is a quantifier-free $\mathsf{FO}(+, \times, <, 0, 1)$-formula expressing an affine-invariant relation. For $i, j, k \in \mathbf{N}$ such that $1 \leq i < j < k \leq m$, let us denote by $\varphi^{\langle i,j,k \rangle}$ the formula

$$\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k) \wedge \varphi(x_1, y_1, \ldots, x_m, y_m).$$

We remark that $\varphi^{\langle i,j,k \rangle}$ expresses an affine-invariant relation.

**Lemma 5.4.4.** The formula $\varphi^{\langle i,j,k \rangle}$ and $\mathsf{AffBasis}(p_i, p_j, p_k) \wedge \mathcal{S}_{p_i, p_j, p_k}(\varphi)$ express the same relation.

*Proof.* For any $\underline{p_1}, ..., \underline{p_m} \in \mathbf{E}$ we consider $(\underline{x_1}, \underline{y_1}), ..., (\underline{x_m}, \underline{y_m})$ to be their coordinates in some fixed affine coordinate system. We prove that

$$\mathcal{R} \models \mathsf{AffBasis}_{\mathsf{coord}}[\underline{x_i}, \underline{y_i}, \underline{x_j}, \underline{y_j}, \underline{x_k}, \underline{y_k}] \wedge \varphi[\underline{x_1}, \underline{y_1}, \ldots, \underline{x_m}, \underline{y_m}].$$

if and only if

$$\mathcal{A}' \models \mathsf{AffBasis}[\underline{p_i}, \underline{p_j}, \underline{p_k}] \wedge \mathcal{S}_{\underline{p_i},\underline{p_j},\underline{p_k}}(\varphi)[\underline{p_1}, \ldots, \underline{p_m}].$$

On the one hand, if $\underline{p_i}, \underline{p_j}, \underline{p_k}$ are affine dependent, then both formulas are clearly false. On the other hand, if $\underline{p_i}, \underline{p_j}, \underline{p_k}$ are affine independent, since $\varphi$ is affine invariant, Proposition 5.4.3 implies that $\varphi[\underline{x_1}, \underline{y_1}, \ldots, \underline{x_m}, \underline{y_m}]$ holds if and only if $\mathcal{S}_{\underline{p_i},\underline{p_j},\underline{p_k}}(\varphi)[\underline{p_1}, \ldots, \underline{p_m}]$ holds.

Hence, $\varphi^{\langle i,j,k \rangle}$ and $\mathsf{AffBasis}(p_i, p_j, p_k) \wedge \mathcal{S}_{p_i,p_j,p_k}(\varphi)$ express the same relation. □

Let us denote by $\varphi^{\langle * \rangle}$ the $\mathsf{FO}(+, \times, <, 0, 1)$-formula

$$\mathsf{Equal}^m_{\mathsf{coord}}(x_1, y_1, \ldots, x_m, y_m) \wedge \varphi(x_1, y_1, \ldots, x_m, y_m).$$

**Lemma 5.4.5.** The formula $\varphi^{\langle * \rangle}$ expresses the same relation as $\mathsf{Equal}^m(p_1, ..., p_m)$ or as $\neg\top$, and which of the two is the case is decidable.

*Proof.* Since the theory of real closed fields is recursively decidable (see, for instance, [BCR98]), it is, in particular, effectively decidable wether $\varphi^{\langle * \rangle}$ is satisfiable or not. If it is unsatisfiable, it expresses the same relation as $\neg\top$.

Suppose, on the other hand, that it its satisfiable. We prove that, under this assumption, $\varphi^{\langle * \rangle}$ expresses the same relation as $\mathsf{Equal}^m$. Clearly, if a tuple of pairs of coordinates satisfies $\varphi^{\langle * \rangle}$, then all the pairs are equal. But since $\varphi^{\langle * \rangle}$ expresses an affine-invariant relation, its truth value is invariant under translations. Hence, it is satisfied by all $m$-tuples of equal pairs of coordinates. Whence, $\varphi^{\langle * \rangle}$ expresses the same relation as $\mathsf{Equal}^m$, and the proof is complete. □

To take care of those cases where all the points are aligned and two are different, we define a new map $\mathcal{S}_{o,e_1}$, differing from $\mathcal{S}_{o,e_1,e_2}$ only in the third and fourth rules in the definition of the term-translation. We remark that the these rules are the only ones where the map $\mathcal{S}_{o,e_1,e_2}$ involves $e_2$. So, we have:

- $\mathcal{S}_{o,e_1}(0) := o$;
- $\mathcal{S}_{o,e_1}(1) := e_1$;
- $\mathcal{S}_{o,e_1}(x_i) := p_i$;

- $\mathcal{S}_{o,e_1}(y_i) := o$;
- $\mathcal{S}_{o,e_1}(t_1 + t_2) := \mathcal{S}_{o,e_1}(t_1) \oplus_o \mathcal{S}_{o,e_1}(t_2)$; and
- $\mathcal{S}_{o,e_1}(t_1 \times t_2) := \mathcal{S}_{o,e_1}(t_1) \otimes_{o,e_1} \mathcal{S}_{o,e_1}(t_2)$.

For $i \leq m$, let us denote by $\varphi^{\langle i \rangle}$ the formula

$$\mathsf{Aligned}^m_{\mathsf{coord}}(x_1, y_1, \ldots, x_m, y_m) \wedge ((x_1 \neq x_i) \vee (y_1 \neq y_i)) \wedge \varphi(x_1, y_1, \ldots, x_m, y_m).$$

Arguing as in the proofs of Proposition 5.4.2 and Lemma 5.4.4, we obtain the following result.

**Lemma 5.4.6.** The formulas, $\varphi^{\langle i \rangle}$ and

$$\mathsf{Aligned}^m(p_1, \ldots, p_m) \wedge (p_1 \neq p_i) \wedge S_{p_1,p_i}(\varphi)(p_1, \ldots, p_m)$$

express the same relation. □

The three previous lemmas motivate the following definitions. Consider the formulas

$$\alpha_m := \bigwedge_{2 \leq i \leq m} ((x_1 = x_i) \wedge (y_1 = y_i)) \vee \neg((x_1 = x_i) \wedge (y_1 = y_i));$$

$$\beta_m := \bigwedge_{1 \leq i < j < k \leq m} (\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k) \vee \neg\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k)).$$

Clearly, $\alpha_m$ and $\beta_m$ are logically valid.

We are now ready to prove Theorem 5.4.1.

*Proof of Theorem 5.4.1* Given $\varphi(x_1, y_1, \ldots, x_m, y_m)$, a quantifier-free $\mathsf{FO}(+, \times, <, 0, 1)$-formula expressing an affine-invariant relation, we define $\widetilde{\varphi}$ as the result of a first distribution of the conjunctions over the disjunctions in $\varphi \wedge \alpha_m \wedge \beta_m$. We remark that, since $\alpha_m$ and $\beta_m$ are logically valid, $\widetilde{\varphi}$ is equivalent to $\varphi$. It is also quantifier free and affine invariant. Also, and to clarify the meaning of previous distribution, we remark that any disjunct in $\widetilde{\varphi}$ contains, for any $1 \leq i < j < k \leq m$, $\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k)$ or $\neg\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k)$ as a conjunct and for any $1 < i \leq m$, it also contains $((x_1 = x_i) \wedge (y_1 = y_i))$ or $((x_1 \neq x_i) \vee (y_1 \neq y_i))$ as a conjunct.

We define the translation $\mathcal{S}(\varphi)$ as the disjunction of the translation of each disjunct $\gamma$ in $\widetilde{\varphi}$. Each disjunct is translated using the Lemmas 5.4.4, 5.4.5 and 5.4.6.

First, we consider the case where, for some $i, j, k \in \mathbf{N}$, $\gamma$ contains the formula $\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k)$ as a conjunct. Let us suppose that $\gamma$ is of the form $\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k) \wedge \delta$, where $(i, j, k)$ is the first triple, in the lexicographical order, such that $\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k)$ is a conjunct of $\gamma$. Then, we define

$$\mathcal{S}(\gamma) := \mathsf{AffBasis}(p_i, p_j, p_k) \wedge \mathcal{S}_{p_i,p_j,p_k}(\delta)(p_1, \ldots, p_m).$$

By Lemma 5.4.4, $\mathcal{S}(\gamma)$ expresses the same relation as $\gamma$.

Now, we suppose now that $\gamma$ contains no conjunct of the form $\mathsf{AffBasis}_{\mathsf{coord}}$ $(x_i, y_i, x_j, y_j, x_k, y_k)$. Hence, $\gamma$ contains $\mathsf{Aligned}^m_{\mathsf{coord}}(x_1, y_1, \ldots, x_m, y_m)$ as a conjunct. If it contains, for some $1 < i \leq m$, $\neg((x_1 = x_i) \wedge (y_1 = y_i))$ as a conjunct, let us write $\gamma = \mathsf{Aligned}^m_{\mathsf{coord}}(x_1, y_1, \ldots, x_m, y_m) \wedge \neg((x_1 = x_i) \wedge (y_1 = y_i) \wedge \delta$ for the first $i$ with this property, and define

$$\mathcal{S}(\gamma) := \mathsf{Aligned}(p_1, \ldots, p_m) \wedge (p_1 \neq p_i) \wedge S_{p_1,p_i}(\delta)(p_1, \ldots, p_m).$$

By Lemma 5.4.6, $\mathcal{S}(\gamma)$ expresses the same relation as $\gamma$.

Finally, we suppose that $\gamma$ contains no conjunct of the form $\neg((x_1 = x_i) \wedge (y_1 = y_i))$. Then, it contains $\mathsf{Equal}^m_{\mathsf{coord}}(x_1, y_1, \ldots, x_m, y_m)$ as a conjunct. We define $\mathcal{S}(\gamma) := \neg\top$ or $\mathcal{S}(\gamma) := \mathsf{Equal}^m(p_1, \ldots, p_m)$, in order to obtain a $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$-formula expressing the same relation, what is possible by Lemma 5.4.5.

We finally define

$$\mathcal{S}(\varphi) = \bigvee_{\gamma \text{ disjunct in } \widetilde{\varphi}} \mathcal{S}(\gamma)$$

Clearly, $\mathcal{S}(\varphi)$ is quantifier free and expresses the same relation as $\varphi$. Whence, $\mathcal{S} : \mathsf{FO}(+, \times, <, 0, 1)_{QF,AI} \rightarrow \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)_{QF}$ is a translation, and the proof is completed. $\square$

## 5.5 The translation $\mathcal{T}$ of $\mathsf{FO}(+, \times, <, 0, 1)_{QF,SI}$-formulas to $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)_{QF}$-formulas

We define a translation from the quantifier-free similarity-invariant fragment of $\mathsf{FO}(+, \times, <, 0, 1)$ into the quantifier-free fragment of $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^\perp, \kappa)$.

The main result of the present section is the following theorem.

**Theorem 5.5.1.** There exists a translation

$$\mathcal{T} : \mathsf{FO}(+, \times, <, 0, 1)_{QF,SI} \to \mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)_{QF}.$$

As in the last section, we assume that the variables used in $\mathsf{FO}(+, \times, <, 0,$ $1)$-formulas are $x_1, y_1, x_2, y_2, ...$ and we define a map (not a translation)

$$\mathcal{T}_{o,e_1,e_2} : \mathsf{FO}(+, \times, <, 0, 1)_{QF,AI} \to \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)_{QF}.$$

The image, $\mathcal{T}_{o,e_1,e_2}(\varphi)$, of an $\mathsf{FO}(+, \times, <, 0, 1)$-formula $\varphi$ in the variables $x_1, y_1,$ $..., x_m, y_m$, involves the variables $o, e_1, e_2, p_1, p_2, ..., p_m$.

First, we define it for $\mathsf{FO}(+, \times, <, 0, 1)$-terms, by induction in structure of the term, as follows:

- $\mathcal{T}_{o,e_1,e_2}(0) := o$,
- $\mathcal{T}_{o,e_1,e_2}(1) := e_1$,
- $\mathcal{T}_{o,e_1,e_2}(x_i) := \mathsf{EuCoord}^1_{o,e_1,e_2}(p_i)$,
- $\mathcal{T}_{o,e_1,e_2}(y_i) := \mathsf{EuCoord}^2_{o,e_1,e_2}(p_i)$,
- $\mathcal{T}_{o,e_1,e_2}(t_1 + t_2) := \mathcal{T}_{o,e_1,e_2}(t_1) \oplus_o \mathcal{T}_{o,e_1,e_2}(t_2)$ and
- $\mathcal{T}_{o,e_1,e_2}(t_1 \times t_2) := \mathcal{T}_{o,e_1,e_2}(t_1) \otimes_{o,e_1} \mathcal{T}_{o,e_1,e_2}(t_2)$, where $t_1$ and $t_2$ are $\mathsf{FO}(+,$ $\times, <, 0, 1)$-terms.

The next proposition is the Euclidean analogous to Proposition 5.4.2. Its proof is completely analogous to that of Proposition 5.4.2, using Lemma 5.3.5 instead of Lemma 5.3.3.

**Proposition 5.5.2.** Let us assume that $\underline{o}, \underline{e_1}, \underline{e_2}$ form an Euclidean coordinate system. Let $t$ be a $\mathsf{FO}(+, \times, <, 0, 1)$-term in the variables $x_1, y_1, \ldots, x_m, y_m$ and consider points $\underline{p_1}, \ldots, \underline{p_m}$ in $\mathbf{E}$, with coordinates $(\underline{x_1}, \underline{y_1}), \ldots, (\underline{x_m}, \underline{y_m})$ with respect to the coordinate system $\underline{o}, \underline{e_1}, \underline{e_2}$.

Then, $\mathcal{T}_{o,e_1,e_2}(t)[\underline{o}, \underline{e_1}, \underline{e_2}, \underline{p_1}, ..., \underline{p_m}]$ has coordinates $(t[\underline{x_1}, \underline{y_1}, ..., \underline{x_m}, \underline{y_m}], 0)$ in the Euclidean coordinate system $\underline{o}, \underline{e_1}, \underline{e_2}$. $\square$

The map $\mathcal{T}_{o,e_1,e_2}$ is defined on atomic formulas and extended to the whole quantifier-free fragment of $\mathsf{FO}(+, \times, <, 0, 1)$ in an analogous way as $\mathcal{S}_{o,e_1,e_2}$ was defined. Also, we write $\mathcal{T}_{\underline{o},\underline{e_1},\underline{e_2}}(\varphi)$ for $\mathcal{T}_{o,e_1,e_2}(\varphi)[\underline{o}, \underline{e_1}, \underline{e_2}]$.

The next proposition and its proof are the Euclidean analogous to Proposition 5.4.3.

**Proposition 5.5.3.** Let us suppose that $\underline{o}, \underline{e_1}, \underline{e_2} \in \mathbf{E}$ form an Euclidean coordinate system, and that $\varphi$ is a quantifier-free $\mathsf{FO}(+, \times, <, 0, 1)$-formula in the variables $x_1, y_1, ..., x_m, y_m$. Consider points $\underline{p_1}, \dots, \underline{p_m}$ in $\mathbf{E}$, with coordinates $(\underline{x_1}, \underline{y_1}), \dots, (\underline{x_m}, \underline{y_m})$ with respect to the coordinate system $\underline{o}, \underline{e_1}, \underline{e_2}$. Then, $\mathcal{E}' \models \mathcal{T}_{\underline{o},\underline{e_1},\underline{e_2}}(\varphi)[\underline{p_1}, ..., \underline{p_m}]$ if and only if $\mathcal{R} \models \varphi[\underline{x_1}, \underline{y_1}, \dots, \underline{x_m}, \underline{y_m}]$. □

The map $\mathcal{T}_{o,e_1,e_2}$ is not a translation because it adds the three new free variables $o, e_1$ and $e_2$. We use the same strategy as in the case of $\mathcal{S}_{o,e_1,e_2}$ to use the variables $p_1, ..., p_m$ already involved in the formula. That is, we considering the three different situations:

1. when all the variables represent the same point;
2. when all the variables represent points that are aligned and two are different; and
3. when there are three variables representing affine-independent points.

Since the Euclidean relations among aligned points coincide with the affine relation among these points, Cases (1) and (2) are translated exactly as in the affine case. The next lemma show how to manage the third case.

Let us suppose that $\varphi(x_1, y_1, \dots, x_m, y_m)$ is a quantifier-free $\mathsf{FO}(+, \times, <, 0,$ $1)$-formula expressing a similarity-invariant relation. We recall that for $i, j, k \in \mathbf{N}$ such that $1 \leq i < j < k \leq m$, we denote by $\varphi^{\langle i,j,k \rangle}$ the formula

$$\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k) \wedge \varphi(x_1, y_1, \dots, x_m, y_m).$$

We remark that $\varphi^{\langle i,j,k \rangle}$ expresses a similarity-invariant relation.

**Lemma 5.5.4.** The formulas $\varphi^{\langle i,j,k \rangle}$ and $\mathsf{AffBasis}(p_i, p_j, p_k) \wedge \mathcal{T}_{p_i,p_j,\varepsilon(p_i,p_j,p_k)}(\varphi)$ express the same relation.

*Proof.* For any $\underline{p_1}, ..., \underline{p_m} \in \mathbf{E}$ we consider $(\underline{x_1}, \underline{y_1}), ..., (\underline{x_m}, \underline{y_m})$ to be their coordinates in some fixed Euclidean coordinate system. We prove that

$$\mathcal{R} \models \mathsf{AffBasis}_{\mathsf{coord}}[\underline{x_i}, \underline{y_i}, \underline{x_j}, \underline{y_j}, \underline{x_k}, \underline{y_k}] \wedge \varphi[\underline{x_1}, \underline{y_1}, \dots, \underline{x_m}, \underline{y_m}].$$

if and only if

$$\mathcal{E}' \models \mathsf{AffBasis}[\underline{p_i}, \underline{p_j}, \underline{p_k}] \wedge \mathcal{T}_{\underline{p_i},\underline{p_j},\varepsilon(\underline{p_i},\underline{p_j},\underline{p_k})}(\varphi)[\underline{p_1}, \dots, \underline{p_m}]$$

On the one hand, if $\underline{p_i}, \underline{p_j}, \underline{p_k}$ are affine dependent, then both formulas are clearly false. On the other hand, if $\underline{p_i}, \underline{p_j}, \underline{p_k}$ are affine independent, Lemma 5.3.6 implies that $\underline{p_i}, \underline{p_j}, \varepsilon(\underline{p_i}, \underline{p_j}, \underline{p_k})$ form an Euclidean coordinate system.

Thus, since $\varphi$ is similarity invariant, Proposition 5.5.3 implies that the sentence $\varphi[\underline{x_1}, \underline{y_1}, \ldots, \underline{x_m}, \underline{y_m}]$ holds if and only if $\mathcal{T}_{\underline{p_i},\underline{p_j},\varepsilon(\underline{p_i},\underline{p_j},\underline{p_k})}(\varphi)[\underline{p_1}, \ldots, \underline{p_m}]$ holds.

Hence, $\varphi^{\langle i,j,k\rangle}$ and $\mathsf{AffBasis}(p_i, p_j, p_k) \wedge \mathcal{T}_{p_i,p_j,p_k}(\varphi)$ express the same relation, what completes the proof. $\square$

Finally, we prove Theorem 5.5.1.

*Proof of Theorem 5.5.1* Given $\varphi(x_1, y_1, \ldots, x_m, y_m)$, a quantifier-free $\mathsf{FO}(+, \times, <, 0, 1)$-formula expressing a similarity-invariant relation, we define $\widetilde{\varphi}$ as in the proof of Theorem 5.4.1. We define the translation $\mathcal{T}(\varphi)$ as the disjunction of the translation of each disjunct $\gamma$ in $\widetilde{\varphi}$. Each disjunct is translated using the Lemmas 5.5.4, 5.4.5 and 5.4.6.

Let $\gamma$ be a disjunct in the disjunction $\widetilde{\varphi}$ of the form $\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k) \wedge \delta$, where $(i, j, k)$ is the first triple, in the lexicographical order, such that $\mathsf{AffBasis}_{\mathsf{coord}}(x_i, y_i, x_j, y_j, x_k, y_k)$ is a conjunct of $\gamma$. Then, we define

$$\mathcal{T}(\gamma) := \mathsf{AffBasis}(p_i, p_j, p_k) \wedge \mathcal{T}_{p_i,p_j,\varepsilon(p_i,p_j,p_k)}(\delta)(p_1, \ldots, p_m).$$

By Lemma 5.5.4, $\mathcal{T}(\gamma)$ expresses the same relation as $\gamma$.

The other two cases ($\gamma$ contains $\mathsf{Aligned}^m_{\mathsf{coord}} \wedge (p_1 \neq p_i)$ for some $i \in \mathbf{N}$ or $\gamma$ contains $\mathsf{Equal}^m_{\mathsf{coord}}$ as conjuncts) are treated in a way completely analogous to the affine case. Since affine and Euclidean relation among aligned points coincide, the map

$$\mathcal{T}(\varphi) = \bigvee_{\gamma \text{ disjunct in } \widetilde{\varphi}} \mathcal{T}(\gamma)$$

obtained in this way $\mathcal{T} : \mathsf{FO}(+, \times, <, 0, 1)_{QF,SI} \to \mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)_{QF}$ is a translation, and the proof is completed. $\square$

## 5.6 Quantifier elimination for the theories $\mathcal{A}'$ and $\mathcal{E}'$

**Theorem 5.6.1.** The theory $\mathcal{A}'$ expresses exactly the affine-invariant geometric relations and admits effective quantifier elimination.

*Proof.* We prove that

$$\mathcal{S} \circ \mathfrak{E}_{\mathcal{R}} \circ \mathcal{C} \circ \mathcal{B} : \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi) \to \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)_{QF}$$

is an effective quantifier-elimination function.

Since $\mathcal{S}, \mathfrak{E}_{\mathcal{R}}, \mathcal{C}$ and $\mathcal{B}$ are recursive functions, their composition is recursive.

Let $\varphi$ be a $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$-formula. By Corollary 5.2.13 and Propositions 5.3.1, the $\mathsf{FO}(+, \times, <, 0, 1)$-formula $\psi := \mathcal{C}(\mathcal{B}(\varphi))$ expresses the same relation as $\varphi$. In particular, it expresses an affine-invariant relation. We recall from Section 5.2.6, that $\mathfrak{E}_{\mathcal{R}}(\psi)$ is a quantifier-free $\mathsf{FO}(+, \times, <, 0, 1)$-formula, equivalent to $\psi$. In particular, it expresses the same relation as $\varphi$. Thus, by Theorem 5.4.1, $\mathcal{S}(\mathfrak{E}_{\mathcal{R}}(\psi))$ is a quantifier-free $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$-formula expressing the same relation as $\varphi$.

Being a definitional extension of a complete theory, $\mathcal{A}'$ is complete. Thus, two $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$-formulas express the same relation (under the standard interpretation) if and only if they are equivalent in $\mathcal{A}'$.

Hence, for any $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$-formula $\varphi$, $\mathcal{S}(\mathfrak{E}_{\mathcal{R}}(\mathcal{C}(\mathcal{B}(\varphi)))) \in \mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ is quantifier free and equivalent to $\varphi$.

Whence, $\mathcal{A}'$ admits effective quantifier elimination. $\square$

In an analogous way, we obtain the following result.

**Theorem 5.6.2.** The theory $\mathcal{E}'$ expresses exactly the similarity-invariant geometric relations and admits effective quantifier elimination.

*Proof.* Arguing as in the previous proof, using Theorem 5.5.1 instead of Theorem 5.4.1, and Theorem 5.3.4 instead of 5.3.1 we conclude that

$$\mathcal{T} \circ \mathfrak{E}_{\mathcal{R}} \circ \mathcal{C} \circ \mathcal{M} : \mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa) \to \mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)_{QF}$$

is an effective quantifier-elimination function for $\mathcal{E}'$. $\square$

## 5.7 Final Remarks

### 5.7.1 Discussion on the primitive notions

We have added new function symbols and the 0-ary predicate symbol $\top$ to our vocabularies to obtain quantifier elimination. We know that the original language does not admit quantifier elimination. It is a natural question whether some of the symbols in the resulting vocabularies are dispensable.

We shall say that a symbol is *dispensable* in a vocabulary if any property expressible in the corresponding language can be expressed by a quantifier free formula not involving the symbol. We remark that, since we require the formula to be quantifier free, this notion is more subtle than what is usually understood by *independence* of the primitive notions.

We briefly argue that $\top$ and $\otimes$ are indispensable in both extended vocabularies.

The case of $\top$ is immediate. Since $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ and $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$ have no constant symbols, no quantifier free sentence can be constructed with out it. Hence, it is indispensable.

To prove that $\otimes$ is indispensable in $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$, consider the formula $z \neq o \wedge o \oplus_z o = r \otimes_{z,o} r$, expressing that the three points are collinear and that the ratio $(z : o : r)$ is equal to $\pm\sqrt{2}$. Theorem 2 in [Tar31] implies that this cannot be defined only with $\beta$ and $\oplus$ (the function $\pi$ does not add expressive power on collinear points). The proof that $\otimes$ is indispensable in $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$ is completely analogous. The dispensability of the symbols $\beta, \pi$ and $\oplus$ in $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ remains an open problem.

Consider the following two $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$-formulas:

- $\equiv (o, p, q, r) \leftrightarrow o \oplus_p o = \kappa(o, p, q, r)$;
- $\beta(p, q, r) \leftrightarrow (p = \kappa(q, r, q, p) \vee q = r)$.

The second formula is analogous to the abbreviation (3) in [Pam01]. The truth of both formulas in $\mathcal{E}'$ is easy to verify. Hence, the symbols $\beta$ and $\equiv$ can be replaced in any $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$-formula by the right side of these formulas. Thus, $\beta$ and $\equiv$ are dispensable in the language $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$. We conclude that the language $\mathsf{FO}(\top, \oplus, \otimes, \pi^{\perp}, \kappa)$ expresses exactly the similarity-invariant properties of the Euclidean plane and admits the elimination of quantifiers. The dispensability of the symbols $\oplus, \pi^{\perp}$ and $\kappa$ in $\mathsf{FO}(\top, \oplus, \otimes, \pi^{\perp}, \kappa)$ remains an open problem.

### 5.7.2 Axiom systems for the new languages $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi)$ and $\mathsf{FO}(\top, \oplus, \otimes, \pi^{\perp}, \kappa)$

Tarski's complete axiom system for elementary Euclidean geometry can be transformed to a complete axiom system for the theory $\mathcal{E}'$ in the language $\mathsf{FO}(\beta, \equiv, \top, \oplus, \otimes, \pi^{\perp}, \kappa)$ adjoining the axiom $\top$ and the implicit definitions of the new function symbols (replacing in formulas given in Section 5.3 the variable $s$ by the corresponding instantiated function symbol). Finally, replacing in the resulting axiom system, each occurrence of $\beta$ and $\equiv$ by the equivalent $\mathsf{FO}(\top, \oplus, \otimes, \pi^{\perp}, \kappa)$-formulas recently introduced, we obtain an axiom system in the language $\mathsf{FO}(\top, \oplus, \otimes, \pi^{\perp}, \kappa)$ for the corresponding theory. The resulting axioms are all universal (also called, quantifier-free) with the exception of the lower-dimensional axiom and the continuity axiom-schema. A natural question remains open: Is it possible to extend our vocabulary with finitely many new functions to obtains a purely universal axiomatization in the line of constructive analysis? We remark that an analogous procedure can be followed to axiomatize the affine case.

## Extension to higher dimensions

Our results can easily be extended to $n$-dimensional spaces for $n > 2$. We briefly indicate how.

In the affine case, we replace the projection function symbol $\pi$ by $\pi^n$ whose interpretation is defined as follows.

$$\pi^n(p, o, e_1, e_2, ..., e_n)$$

is the projection, parallel to the affine hull of $o, e_2, ...e_n$, of $p$ over $o, e_1$, if $p$ belongs to the affine hull of $o, e_1, e_2, ...e_n$ and $o$ otherwise. A direct generalization of our proofs (using $\pi^n$ to coordinate the space) shows that the language $\mathsf{FO}(\beta, \top, \oplus, \otimes, \pi^n)$, interpreted over the $n$-dimensional Euclidean space, expresses exactly the affine-invariant relations on the $n$-dimensional Euclidean space and admits quantifier elimination.

On the other hand, a straightforward generalization to dimension $n$ of our proofs shows that the language $\mathsf{FO}(\top, \oplus, \otimes, \pi^\perp, \kappa)$, interpreted over the $n$-dimensional Euclidean space (where $\pi^\perp$ is, as before, interpreted as the orthogonal projection over a line), expresses exactly the similarity-invariant relations and admits the elimination of quantifiers.

# A

# Algebraic computational models

We introduce the main computational models considered in this thesis: the *algebraic computation tree* model and the *boolean-arithmetic circuit* model.

Algebraic computation trees represent a good model of sequential algebraic computations. The notion of boolean-arithmetic circuit enables to capture the notion of *parallel complexity*.

Any boolean-arithmetic circuit can be transformed into an algebraic computation tree of depth bounded by the size of the original circuit. Thus, a lower bound for the sequential complexity in the model of algebraic computation trees implies a lower bound for the sequential complexity in the model of boolean-arithmetic circuits.

We start defining the *algebraic decision tree* model, that is a simple model used to prove lower complexity bounds (see Exercises 3.15 and 11.4 in [BCS97]).

A *tree* is a finite set $T$ of nodes such that

- there is one specially designated node called the *root* of the tree;
- the remaining nodes are partitioned into $m \geq 0$ disjoint nonempty sets $T_1, ..., T_m$, and each of these sets is in a tree. These trees are call the *subtrees* of the root.

The number $m$ of subtrees of a node $v \in T$ is called the *outdegree* of the node. The root of a tree is called the *parent* of the roots of its subtrees. The *predecessor* relation is defined as the transitive closure of the *parent* relation. Hence, a node $v_1 \in T$ is called a *predecessor* of a node $v_2 \in T$ if $v_1$ is the root of a subtree of $T$ containing $v_2$. A *ternary tree* is a tree in which each internal node has outdegree one, two or three.

## A.1 Algebraic Decision Trees over the Reals

Let $n$ be a positive integer. An *algebraic decision tree over the reals* is a ternary tree together with a function that assigns to each of its inner nodes $v$ a polynomial $F_v$ in $\mathbf{R}[X_1, ..., X_n]$, and to each of its leaves a label (for instance, these labels could be “accept" or “reject").

The semantics of algebraic decision trees is defined as follows. To any *input* $x \in \mathbf{R}^n$, we assign a unique path in the tree from the root to a leaf by continuing with the left son of a node $v$ if $F_v(x) < 0$, with the middle son if $F_v(x) = 0$, and with the right son if $F_v(x) > 0$. The *output* of the algebraic decision tree is the label of the leaf where the path ends.

The number of steps of an algebraic decision tree is defined as the depth of the underlying tree.

Algebraic decision trees are used to give lower bounds for the *branching* (also called *topological*) complexity of semi-algebraic problems.

As we show in Section 2.4, this model can be used to show that if an algorithm solves the sign condition problem for a family $\mathcal{P}$ of polynomials, evaluating only the polynomials in this family, then the algorithm might have to evaluate all the polynomials in $\mathcal{P}$ for a given input.

## A.2 Algebraic Computation Trees over the Reals

We now describe the model of *algebraic computation trees*. Our definitions are based on the formulation of [BCS97] (see also [Str81], [BO83] and [Lic90]). For convenience, we shall distinguish two types of inputs for computation trees: *variables* and *parameters*. As is also the case of physical systems, the distinction might depend on the context of application.

Analogous to Strassen [Str81], we define computation trees as consisting of *a subjacent tree*, an *instruction function* and an *inference partition* of the leaves. Since we shall consider different inference partitions for a fixed tree and a fixed instruction function, we introduce the intermediary notion of *algebraic tree* as a couple composed of a tree together with an instruction function for this tree. In this way, a computation tree is an algebraic tree together with an inference partition of the its leaves.

### A.2.1 Syntax of computation trees

**Definition A.2.1** (Algebraic trees)**.** Let $\mathcal{T}$ be a finite tree with four types of nodes: *assignment nodes* (outdegree 1), *arithmetic nodes* (outdegree 1), *test nodes* (outdegree 3) and *leaf nodes* (outdegree 0). An *algebraic tree* with variables $X_1, ..., X_n$ and parameters $U_1, ..., U_m$, is a such a tree $\mathcal{T}$ together with a function (the *instruction function*) that associates

- to each assignment node $v$, a real constant, a variable or a parameter;
- to each arithmetic node $v$, an arithmetic operation $\circ_v \in \{+, -, \times, /\}$ and two predecessor nodes, $p_1(v)$ and $p_2(v)$, of $v$ in $\mathcal{T}$;
- to each test node $v$, two predecessor nodes, $p_1(v)$ and $p_2(v)$, of $v$ in $\mathcal{T}$;
- to each leaf node $l$, an (eventually empty) *output list* of predecessor nodes $p_1(l), ..., p_{i_l}(l)$ of $l$ in $\mathcal{T}$.

The tree $\mathcal{T}$ is called the subjacent tree of the algebraic tree. When no confusion can arise, we denote algebraic trees with the same symbols as their subjacent trees.

We denote by $\mathfrak{T}^{(n)}$ the set of algebraic trees involving $n$ variables and no parameters, and by $\mathfrak{T}^{(n,m)}$ the set of algebraic trees involving $n$ variables and $m$ parameters. □

**Definition A.2.2** (Computation tree)**.** A *computation tree*, $(\mathcal{T}, \sigma)$, is an algebraic tree $\mathcal{T}$ together with a partition $\sigma$ of the set of leaves of $\mathcal{T}$ (the *inference partition*) such that the length $i_l$ of the output list is constant on $\sigma$-classes.

The (algebraic) tree $\mathcal{T}$ is called the *subjacent (algebraic) tree* of the computation tree $(\mathcal{T}, \sigma)$. The *depth* and *size* of a computation tree are the depth and the size of its subjacent tree.

If a computation tree involves parameters, we shall call it a *parametric computation tree* to emphasize this fact.

We denote by $\mathfrak{C}^{(n)}$ the set of computation trees with $n$ variables, and by $\mathfrak{C}^{(n,m)}$ the set of parametric computation trees with $n$ variables and $m$ parameters. □

The parameters of a parametric computation trees can be instantiated by different values; each instantiation defines a new computation tree.

**Definition A.2.3** (Instantiation of the parameters)**.** Let $\mathcal{T} \in \mathfrak{T}^{(n,m)}$ be a parametric algebraic tree.

For any $u = (u_1, ..., u_m) \in \mathbf{R}^m$, we define $\mathcal{T}[u]$ as the (nonparametric) algebraic tree having the same subjacent tree as $\mathcal{T}$ and whose instruction function differs from that of $\mathcal{T}$ in that it assigns the real constant $u_i$ to any node $v$ where the instruction function of $\mathcal{T}$ assigned the parameter $U_i$. □

Let $(\mathcal{T}, \sigma) \in \mathfrak{C}^{(n,m)}$ and let $u \in \mathbf{R}^m$. We remark that, since the subjacent trees of $\mathcal{T}$ and of $\mathcal{T}[u]$ are the same tree, $\sigma$ is an inference partition for $\mathcal{T}[u]$, *i.e.*, $(\mathcal{T}[u], \sigma) \in \mathfrak{C}^{(n)}$ is a computation tree.

### A.2.2 Semantics of computation trees

Let $\mathcal{T}$ be an algebraic tree with variables $X_1, ..., X_n$ and parameters $U_1, ..., U_m$. We associate to each internal node $v \in \mathcal{T}$, a rational function $comp_v \in \mathbf{R}[X_1, ..., X_n, U_1, ..., U_m]$, as follows:

- for an assignment node $v$, with an associated real constant $c$, we define $comp_v := c$;
- for an assignment node $v$, with an associated variable $X_i$, we define $comp_v := X_i$;
- for an assignment node $v$, with an associated parameter $U_i$, we define $comp_v := U_i$;
- for an arithmetic node $v$, we define $comp_v := comp_{p_1(v)} \circ_v comp_{p_2(v)}$;
- for a test node $v$, we define $comp_v := comp_{p_1(v)} - comp_{p_2(v)}$.

If $v$ is an internal node of $\mathcal{T}$, we say that it *computes* the rational function $comp_v$. For any $x = (x_1, ..., x_n) \in \mathbf{R}^n$ and any $u = (u_1, ..., u_m) \in \mathbf{R}^m$ we associate to $v$ the real value $comp_v(x, u)$ if $comp_v$ is defined on $(x, u)$. Otherwise, we say that $comp_v$ is undefined on $(x, u)$.

**Definition A.2.4** (Computation path, output)**.** The *computation path* followed in an algebraic tree $\mathcal{T}$, for parameters $(u_1, ..., u_m) \in \mathbf{R}^m$, on input $x = (x_1, ..., x_n) \in \mathbf{R}^n$, is the unique path in $\mathcal{T}$ that satisfies the following properties:

- the path starts at the root of $\mathcal{T}$;
- the successor of an assignment node $v$ in this path is its unique immediate successor in the tree $\mathcal{T}$;
- an arithmetic node $v$ has a successor in this path only if $comp_v(x, u)$ is defined: in this case the successor of $v$ in the path is its unique successor in $\mathcal{T}$; otherwise, the computation path ends in $v$;
- the successor of a test node $v$ in this path corresponds to the first, second or third immediate successor of the node $v$ in $\mathcal{T}$, according to whether $comp_v(x, u)$ is less, equal or greater than zero, respectively—*i.e.*, according to whether $comp_{p_1(v)}(x, u)$ is lower, equal or greater than $comp_{p_2(v)}(x, u)$;

- leaf nodes have no successor in a computation path.

We denote by $\mathcal{T}(x,u)$ the last node reached by this computation path. If $\mathcal{T}(x,u)$ is a leaf, say $l$, of $\mathcal{T}$, then we call it *the result of the execution* of $\mathcal{T}$ on input $x$ for parameters $u$, and we say that the algebraic tree $\mathcal{T}$ is executable on $(x,u)$. Let $p_1(l),...,p_{i_l}(l)$ be the output list of $l$. Then $(comp_{p_1(l)}(x,u),...,$ $comp_{p_{i_l}(l)}(x,u))$ is called the *output* of $\mathcal{T}$ on input $x$ for parameters $u$. If $(\mathcal{T},\sigma) \in \mathfrak{C}^{(n,m)}$, then the $\sigma$-class in which the leaf $l = \mathcal{T}(x,u)$ lies is called the *output class* of $(x,u)$. □

If $l \in \mathcal{T}$ is a leaf, we define the semi-algebraic set

$$D_l := \{(x,u) \in \mathbf{R}^n \times \mathbf{R}^m \ \mid \ \mathcal{T}(x,u) = l\}.$$

A leaf $l \in \mathcal{T}$ is called *active for parameters* $u \in \mathbf{R}^m$ if, for some $x \in \mathbf{R}^n$, it is the result of the execution of $\mathcal{T}$ on input $x$ for parameters $u$, *i.e.*, $l = \mathcal{T}(x,u)$.

**Computation trees compute collections.** Let us now state precisely what a computation tree computes. Let $(\mathcal{T},\sigma) \in \mathfrak{C}^{(n,m)}$ be a parametric computation tree and let $J \subset \mathbf{R}^n \times \mathbf{R}^m$ be a semi-algebraic set. We say that $\mathcal{T}$ is *executable* on $J$ if and only if $\mathcal{T}$ is executable on every element of $J$. Let the inference partition $\sigma$ be $\{\sigma_1,...,\sigma_t\}$ and let $k_i$ be the common output length of the nodes in $\sigma_i$. The set $J$ is partitioned into the subsets $(1 \le i \le t)$

$$J_i := \{(x,u) \in J \ \mid \ \text{the output class of } (x,u) \text{ is } \sigma_i\}.$$

We call this the *partition of inputs* of $(\mathcal{T},\sigma)$ on $J$. For each $i$, $1 \le i \le t$, we have a mapping $\varphi_i : J_i \to \mathbf{R}^{k_i}$ assigning to any $(x,u) \in J_i$ its output. The unique extension $\varphi$ of the $\varphi_i$, to a map defined on $J$ is called the *computation map* of $\mathcal{T}$ on $J$. On a given input $(x,u) \in J$ the computation tree $(\mathcal{T},\sigma)$ *decides* in which of the classes $J_i$ the element $(x,u)$ lies and computes the *output* $\varphi(x,u)$ of $\mathcal{T}$ on $(x,u)$. This motivates the following definition.

**Definition A.2.5** (Collection)**.** A *collection* for a semi-algebraic set $J \subset \mathbf{R}^n \times \mathbf{R}^m$ consists of a partition $\pi = \{J_1,...,J_t\}$ of $J$ into finitely many semi-algebraic components and of a family of semi-algebraic functions $\varphi_i : J_i \to \mathbf{R}^{k_i}$ for $1 \le i \le t$. We will denote a collection by $(\varphi,\pi)$ where $\varphi$ is the unique map defined on $J$ which extends the $\varphi_i$. □

**Definition A.2.6** (Computable in time $t$)**.** Let $(\mathcal{T},\sigma) \in \mathfrak{C}^{(n,m)}$ be a computation tree and let $J \subset \mathbf{R}^n \times \mathbf{R}^m$. We say that $(\mathcal{T},\sigma)$ *computes the collection* $(\varphi,\pi)$, consisting of the computation map and the partition of inputs of the computation tree $(\mathcal{T},\sigma)$ on $J$, *in time* $t$, where $t$ is the depth of the tree $\mathcal{T}$.

A collection $(\varphi,\pi)$ is *computable in time* $t$ if there exists a computation tree of depth bounded by $t$ that computes $(\varphi,\pi)$. □

The depth of an algebraic computation tree is also called its *total* or *algebraic complexity.* The maximum number of tests in a single path is called its *branching complexity.* Analogously, the maximum number of non-scalar products in a single path in the tree is called its *non-scalar complexity*

Let us discuss some special cases. A collection $(\varphi, \pi)$ for $J$ with $\pi = \{J\}$ describes a "pure computational problem". In this case, we omit $\pi$ and speak of the collection $\varphi$; we say that $\mathcal{T}$ computes the function $\varphi$.

On the other hand, a collection $(\varphi, \pi)$ where $\varphi$ maps every $(x, u)$ to a zero-length output, describes a decision problem. Point location problems are of this kind. If $|\pi| = 2$ then we have a membership problem. In these cases, we omit $\varphi$ and regard the partition $\pi$ as a special collection; we say that $(\mathcal{T}, \sigma)$ computes the partition $\pi$. If $\mathcal{T}$ involves parameters we say that $(\mathcal{T}, \sigma)$ computes the *parametric* partition $\pi$.

The following result is immediate and can be found, for example in [Lic96]

**Proposition A.2.7.** If a computation tree $(\mathcal{T}, \sigma)$ computes a partition $\pi$ of a set $J$ in time $t$ and $\widetilde{\pi}$ is a coarser partition of $J$, then there exists an inference partition $\widetilde{\sigma}$ such that $(\mathcal{T}, \widetilde{\sigma})$ computes the partition $\widetilde{\pi}$ in time $t$. $\square$

**Instantiation of the parameters and notation.** For any polynomial $F \in \mathbf{R}[X_1, ..., X_n, U_1, ..., U_m]$ and any $u \in \mathbf{R}^m$ let us denote by $F[u]$ the polynomial $F(X_1, ..., X_n, u) \in \mathbf{R}[X_1, ..., X_n]$. Also, for any set $J \subset \mathbf{R}^n \times \mathbf{R}^m$ and any $u \in \mathbf{R}^m$, let us denote by $J[u]$ the set $\{x \in \mathbf{R}^n \ |(x, u) \in J\}$.

A parametric partition $\pi = \{\pi_1, ..., \pi_k\}$ of a set $J \subset \mathbf{R}^n \times \mathbf{R}^m$ induces, for every $u \in \mathbf{R}^m$, a partition $\pi[u]$ on $J[u]$ given by $\pi[u] := \{\pi_1[u], ..., \pi_k[u]\}$.

The next proposition follows immediately from the definitions.

**Proposition A.2.8.** Let $(\mathcal{T}, \sigma) \in \mathfrak{C}^{(n,m)}$ be a parametric computation tree that computes a parametric partition $\pi$ of $J$ in time $t$.

If $u \in \mathbf{R}^m$, then $(\mathcal{T}[u], \sigma) \in \mathfrak{C}^{(n)}$ computes the partition $\pi[u]$ of $J[u]$ in time $t$. $\square$

## A.2.3 Pragmatics of computation trees

Algebraic computation trees provide an excellent model to prove lower bounds for the algebraic complexity of some problems. Lower bounds are usually proved for the branching complexity or the non-scalar complexity of any algebraic computation tree solving a fixed problem.

In order to prove lower bounds, the main drawback of this model is that it does not include a notion of *uniformity.* On the other hand, this allows to describe non-uniform algorithms in this model. For instance, Meyer auf der Heide [MadH84] described a non-uniform polynomial time solution to the NP-complete Knapsack problem using a similar (linear) model.

## A.3 Boolean-Arithmetic Circuits over the Reals

As said before, algebraic computation trees represent a good model of sequential algebraic computations. The notion of *boolean-arithmetic circuit* allows us to capture the notion of *parallel complexity*.

A *boolean-arithmetic circuit* (also called arithmetic networks in [vzG86a]), $C$, is a directed acyclic graph where each node of the graph is an *input node* or a *constant node* if it has in-degree zero and a *gate node* if its in-degree is positive. *Output nodes* are nodes with out-degree zero. Also, every node will be *arithmetic* or *boolean* according to the kind of information it contains (observe that input and constant nodes can be of any of both kinds). Nodes connected to gates are supposed to be of the correct type, as described bellow.

Every gate node will be labeled as one of the following:

- *Arithmetic Gate Nodes*: $\times, +, -, /$. Input: two arithmetic nodes.
- *Boolean Gate Nodes*: $\wedge, \vee$ (with two boolean nodes as input) and $\neg$ (with one boolean input node).
- *Boolean Sign Nodes.* Input: one arithmetic node. Its boolean value depends on the sign ($\{-1, 0, 1\}$) of the input. There are two kind of boolean sign nodes corresponding to the two predicates $=$ and $\leq$.
- *Arithmetic Selection Nodes.* Input: two arithmetic and one boolean node. Its values is equal to the first or second arithmetic input according to the truth value of the boolean input.

We define the size of $C$ as the number of nodes in the circuit (see also [Weg87]). We will use this number as our measure of *sequential complexity*, whereas the *parallel complexity* is determined by the *depth* of the circuit (*i.e.*, the length of the longest path in the subjacent graph, from an input node to an output node).

We will not define the semantics of boolean-arithmetic circuits. We recall that any boolean-arithmetic circuit can be transformed into an algebraic computation tree of depth bounded by the size of the original circuit. Thus, a lower bound for the sequential complexity in the model of algebraic computation trees implies a lower bound for the sequential complexity in the model of boolean-arithmetic circuits.

## A.4 The Bit Models

By *the bit model* we understand the Turing machine model. On the other hand, within an algebraic computation model, we can restrict the arithmetic operations to be performed only among integers. in this case, the bitsize of these integers involved is taken into account.

Even when we restrict our algorithms to work over the integers, they have to deal with algebraic numbers (for example, to represent sample points of semi-algebraic sets). In this context, we use the following representation for algebraic numbers.

### A.4.1 Bit Representation of Algebraic Numbers

**Definition A.4.1.** Let $T \in \mathbf{Z}[X]$ be a non-zero polynomial and let $\sigma$ be a sign condition on the derivatives of $T$. We say that $(T, \sigma)$ is a *Thom encoding* of the real algebraic number $x$ if $T(x) = 0$ and $x$ satisfies the sign condition $\sigma$. By the *degree* and *logarithmic height* of a Thom encoding we understand the degree and logarithmic height of $T$.

We remark that by Thom's Lemma (see Proposition 2.5.4 in [BCR98]) two different real numbers cannot have the same Thom encoding.

**Definition A.4.2.** A *Triangular Thom encoding* for a tuple of real algebraic numbers $(x_1, ..., x_n) \in \mathbf{R}^n$ is a pair $(\mathcal{T}, \sigma)$, where $\mathcal{T}$ is a tuple $(T_1, ..., T_n)$ of polynomials and $\sigma$ is a tuple $(\sigma_1, ..., \sigma_n)$ of sign conditions such that, $(T_1, \sigma_1)$ is a (non-zero) Thom encoding of $x_1$ and, for $2 \leq i \leq n$, $T_i \in \mathbf{Z}[X_1, ..., X_i]$ and $(T_i(x_1, ..., x_{i-1}, X_i), \sigma_i)$ is a (non-zero) Thom encoding of $x_i$.

The *degree* and *logarithmic height* of a triangular Thom encoding $(\mathcal{T}, \sigma)$ are defined, respectively, as the maximal degree and the maximal logarithmic height of the polynomials $T_1, ..., T_n$. The *size* of $(\mathcal{T}, \sigma)$ is defined as the pair $(d, \tau)$, where $d$ is its degree and $\tau$ its logarithmic height.

We remark that the triangular Thom encoding is the natural representation of the sample points obtained by the Cylindrical Algebraic Decomposition method.

To evaluate the sign of a polynomial $P \in \mathbf{Z}[X_1, ..., X_n]$ at an algebraic point, we use Algorithm 11.8 (`sign determination algorithm`) from [BPR06]. The properties of this algorithm are summarized in the following proposition.

**Proposition A.4.3.** Let $P$ be a polynomial in $\mathbf{Z}[X_1, ..., X_n]$ and let $(\mathcal{T}, \sigma)$ be a triangular Thom encoding of $x \in \mathbf{R}^n$. Assume that $P$ and $(\mathcal{T}, \sigma)$ have degrees and logarithmic height bounded by $d \geq 2$ and $\tau$, respectively. Then, the `sign determination algorithm` determines the sign of $P(x)$ performing

$d^{\mathcal{O}(n)}$ arithmetic operations. Moreover, the logarithmic height of the integers involved in these operations is bounded by $\tau d^{\mathcal{O}(n)}$.

By convention and to simplify the statement of the complexity results, we assume that the $d$, the bound on the degrees of the polynomials involved in the input of our algorithms, is at least 2.

## A.5 Representation of Polynomials

Let be given a polynomial $F \in \mathbf{R}[X_1, ..., X_n]$. We shall consider the following different representations of it. Besides the number $n$ of variables, each of these representations has associated some natural parameters measuring the complexity of the representation.

1. **Arithmetic-circuit representation**. The polynomial $F$ is represented by an arithmetic circuit $\Gamma$ over $\mathbf{R}$ that computes it (see [vzG86b, BCS97]; succinctly, an *arithmetic circuit* is a boolean-arithmetic circuit involving only arithmetic input nodes, arithmetic constant nodes and arithmetic gate nodes). We limit ourselves to division-free circuits. Let us denote by $L$ the non-scalar size of $\Gamma$ (*i.e.*, the number of non-scalar multiplication nodes in the circuit) and observe that the degree of $F$ is bounded by $2^L$. The parameters associated with this representation are $n$ and $L$.

2. **Dense arithmetic representation**. Suppose that the polynomial $F$ has degree $d$. The dense arithmetic representation of $F$ consists on the tuple in $\mathbf{R}^{\binom{d+n}{n}}$ of its coefficients in the monomial basis. The parameters associated with this representation are $n$ and $d$.

3. **Dense bit representation**. Let us assume that the polynomial $F$ has integer coefficients. If $F$ has logarithmic height $\tau$ and degree $d$, its dense bit representation is the tuple in $\mathbf{Z}^{\binom{d+n}{n}}$ of its coefficients in the monomial basis, where each integer is represented by its bit encoding (of size at most $\tau$). The parameters associated with this representation are $n, d$ and $\tau$.

4. **Sparse representations** The sparse representation of the polynomial $F$ consists of the list of all pairs $(\mu, a_\mu)$, where $a_\mu \in \mathbf{R}$ is a non-zero coefficient of $F$ corresponding to the monomial with multi-exponent $\mu \in \mathbf{N}^n$. The parameters associated with this representation are $n, d$, the number of non-zero coefficients of $F$ and, if the polynomial has integer coefficients, also their logarithmic height.

5. **Straight-line-program representation**. A *straight line program* (SLP, for short) can be defined as an algebraic computation tree without branching nodes. It is the sequential version of arithmetic circuits.

Given a family $\mathcal{F} := \{F_1, ..., F_s\}$ of polynomials in $\mathbf{R}[X_1, ..., X_n]$, the arithmetic-circuit representation the family $\mathcal{F}$ is division-free arithmetic circuit $\Gamma$ over $\mathbf{R}$ that computes all the polynomials in $\mathcal{F}$. Let us denote by $L$ the non-scalar size of $\Gamma$. The parameters associated with this representation are $s, n$ and $L$.

## A.5.1 The Relation Between the Total and the Non-Scalar Complexity of Evaluating a Family of Polynomials

Given a circuit representation of a family $\mathcal{F} := \{F_1, ..., F_s\} \subset \mathbf{R}[X_1, ..., X_n]$, we discuss the cost (total complexity) of evaluating some of these polynomials.

**Lemma A.5.1.** Let $\mathcal{F} := \{F_1, ..., F_s\} \subset \mathbf{R}[X_1, ..., X_n]$ be a family of polynomials represented by a division-free arithmetic circuit $\Gamma$ of non-scalar complexity $L$. Let $k$ be a positive integer, $1 \leq k \leq s$, and let $1 \leq i_1 \leq \cdots \leq i_k \leq s$ be integers. Then, $F_{i_1}, ..., F_{i_k}$ can be computed with total complexity $\mathcal{O}((L + k)(L + n))$.

*Proof.* Let us denote by $n_1, ..., n_L$ the non-scalar multiplication nodes in $\Gamma$ and by $P_1, ..., P_L$ the polynomials in $\mathbf{R}[X_1, ..., X_n]$ computed by these nodes. We assume, without loss of generality, that $\deg(P_1) \leq \deg(P_2) \leq \cdots \leq \deg(P_L)$.

It is easy to see that, for $1 \leq i \leq L$, the node $n_i$ computes the product of two polynomials of the form

$$\Sigma_{j=1}^{i-1} \gamma_j P_j + \Sigma_{j=1}^{n} \beta_j X_j + \beta_0,$$

where the greek letters represent real numbers. Rewriting the circuit if necessary, each of these linear combinations can be computed from the preceding non-scalar multiplication nodes and input variables without non-scalar multiplications and a total complexity of $\mathcal{O}(n + i)$. Thus, there exists a division-free arithmetic circuit, namely $\Gamma_P$, that computes the family $\{P_1, ..., P_L\}$ with total complexity $\mathcal{O}(L^2 + nL)$.

We remark that, for any $1 \leq i \leq s$,

$$F_i = \Sigma_{j=1}^{L} \gamma_j P_j + \Sigma_{j=1}^{n} \beta_j X_j + \beta_0,$$

where the greek letters represent real numbers. Hence, each $F_i$ can be computed from $\{P_1, ..., P_L\}$ performing $\mathcal{O}(L + n)$ arithmetic operations. Thus, the polynomials $F_{i_1}, ..., F_{i_k}$ can be computed with a total complexity $\mathcal{O}(L^2 + nL) + \mathcal{O}(k(L + n))$. This competes the proof. □

# Bibliography

[AE99] P. K. Agarwal and J. Erickson, *Geometric range searching and its relatives*, Advances in Discrete and Computational Geometry, American Mathematical Society, 1999, pp. 1–56.

[AHU74] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The design and analysis of computer algorithms*, Addison-Wesley, 1974.

[AKS07] M. Avendaño, T. Krick, and M. Sombra, *Factoring bivariate sparse (lacunary) polynomials*, J. Complexity **23** (2007), no. 2, 193–216.

[BCR98] J. Bochnak, M. Coste, and M. F. Roy, *Real algebraic geometry*, Modern Surveys in Mathematics, vol. 36, Springer-Verlag, 1998.

[BCS97] P. Bürgisser, M. Clausen, and M. A. Shokrollahi, *Algebraic complexity theory*, Grundlehren der mathematischen Wissenschaften, Springer, 1997.

[BCSS98] L. Blum, F. Cucker, M. Shub, and S. Smale, *Complexity and real computation*, Springer-Verlag, Secaucus, NJ, USA, 1998.

[BD07] C. W. Brown and J. H. Davenport, *The complexity of quantifier elimination and Cylindrical Algebraic Decomposition*, ISSAC '07: Proceedings of the 2007 international symposium on Symbolic and algebraic computation (New York, NY, USA), ACM, 2007, pp. 54–60.

[BE95] P. Borwein and T. Erdélyi, *Polynomials and polynomial inequalities*, Springer-Verlag, 1995.

[BGKV07] P. Balbiani, V. Goranko, R. Kellerman, and D. Vakarelov, *Handbook of spatial logics (m. aiello et al., eds.)*, ch. Logical theories for fragments of elementary geometry, pp. 343–428, Springer-Verlag, Secaucus, NJ, USA, 2007.

[BH99] W. Baur and H. Halupczok, *On lower bounds for the complexity of polynomials and their multiples*, Computational Complexity **8** (1999), no. 4, 309–315.

[BO83] M. Ben-Or, *Lower bounds for algebraic computation trees*, STOC '83: Proceedings of the fifteenth annual ACM symposium on Theory of computing (New York, NY, USA), ACM, 1983, pp. 80–86.

[BOKR84] M. Ben-Or, D. Kozen, and J. Reif, *The complexity of elementary algebra and geometry*, in STOC '84: Proceedings of the sixteenth annual ACM symposium on Theory of computing, ACM, 1984, pp. 457–464.

[BPR10] S. Basu, R. Pollack, and M .F. Roy, *An asymptotically tight bound on the number of connected components of realizable sign conditions*, Combinatorica (2009/10), To appear.

[BPR94] S. Basu, R. Pollack, and M. F. Roy, *On the combinatorial and algebraic complexity of quantifier elimination*, IEEE Symposium on Foundations of Computer Science, 1994, pp. 632–641.

[BPR96] ______, *On the combinatorial and algebraic complexity of quantifier elimination*, J. ACM **43** (1996), no. 6, 1002–1045.

[BPR99] S. Basu, R. Pollack, and M.-F. Roy, *Computing roadmaps of semialgebraic sets on a variety*, J. AMS **3** (1999), no. 1, 55–82.

[BPR06] S. Basu, R. Pollack, and M. F. Roy, *Algorithms in real algebraic geometry*, 2 ed., Algorithms and Computation in Mathematics, vol. 10, Springer-Verlag, Secaucus, NJ, USA, 2006.

[BS83] W. Baur and V. Strassen, *The complexity of partial derivatives.*, Theoretical Computer Science **22** (1983), 317–330.

[BSS89] L. Blum, M. Shub, and S. Smale, *On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines*, Bull. American Math. Soc. **21** (1989), 1–46.

[Bür01] P. Bürgisser, *Lower bounds and real algebraic geometry*, Algorithmic and Quantitative Aspects of Real Algebraic Geometry in Mathematics and Computer Science, Discrete Mathematics and Theoretical Computer Science, vol. 60, AMS, 2001, pp. 35–54.

[Can88] J. F. Canny, *Some algebraic and geometric computations in pspace*, Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing, 2-4 May 1988, Chicago, Illinois, USA, ACM, 1988, pp. 460–467.

[Can93] ______, *Computing roadmaps of general semi-algebraic sets*, Comput. J. **36** (1993), no. 5, 504–514.

[CEGS91] B. Chazelle, H. Edelsbrunner, L. J. Guibas, and M. Sharir, *A singly-exponential stratification scheme for real semi-algebraic varieties and its applications*, Theoretical Computer Science **84** (1991), no. 1, 77–105.

[CGH$^+$03] D. Castro, M. Giusti, J. Heintz, G. Matera, and L. M. Pardo, *The hardness of polynomial equation solving*, Foundations of Computational Mathematics **3** (2003), 1–74.

[CGV91] J. F. Canny, D. Grigoriev, and N. Vorobjov, *Finding connected components of a simialgebraic set in subexponential time*, Appl. Algebra Eng. Commun. Comput. **2** (1991), 217–238.

[Cla87] K. L. Clarkson, *New applications of random sampling in computational geometry*, Discrete & Computational Geometry **2** (1987), no. 2, 195–222.

[Col75] G. E. Collins, *Hauptvortrag: Quantifier elimination for real closed fields by Cylindrical Algebraic Decomposition*, Automata Theory and Formal Languages, 1975, pp. 134–183.

[CS90] B. Chazelle and M. Sharir, *An algorithm for generalized point location and its applications*, J. Symb. Computation **10** (1990), no. 3-4, 281–309.

[CW90] D. Coppersmith and S. Winograd, *Matrix multiplication via arithmetic progression*, J. Symb. Computation **9(3)** (1990), 251–280.

[Des37] R. Descartes, *La géométrie*, Jan Maire, Laiden, 1637.

[DH88] J. H. Davenport and J. Heintz, *Real quantifier elimination is doubly exponential*, J. Symb. Computation **5** (1988), 29–35.

[DL76] D. P. Dobkin and R. J. Lipton, *Multidimensional searching problems*, SIAM J. Comput. **5** (1976), no. 2, 181–186.

[DL07] C. Durvye and G. Lecerf, *A concise proof of the Kronecker polynomial system solver from scratch*, Expositiones Mathematicae (2007), 101–139.

[End00] H. Enderton, *A mathematical introduction to logic*, Harcourt-Academic Press, 2000.

[FGG] S. Figueira, D. Gorín, and R. Grimson, *On the formal semantics of IF-like logics*, To appear in the J. of Computer and System Sciences.

[FGG08] ——, *On the formal semantics of IF-like logics*, XV Workshop on Logic, Language, Information and Computation, Lecture Notes in Computer Science, vol. 5110, Springer, 2008, pp. 164–178.

[FGM90] N. Fitchas, A. Galligo, and J. Morgenstern, *Precise sequential and parallel complexity bounds for quantifier elimination over algebraically closed fields*, J. Pure and Appl. Algebra **67** (1990), 1–14.

[FR74] M. J. Fischer and M. O. Rabin, *Super-exponential complexity of presburger arithmetic*, SIAMAMS: Complexity of Computation: Proceedings of a Symposium in Applied Mathematics of the American Mathematical Society and the Society for Industrial and Applied Mathematics, vol. 7, 1974, pp. 27Ũ–41.

[GBG99] M. Gyssens, J. Van den Bussche, and D. Van Gucht, *Complete geometric query languages*, J. of Computer and System Sciences **58** (1999), no. 3, 483–511.

[GH01] M. Giusti and J. Heintz, *Kronecker's smart, little black boxes*, London Mathematical Society Lecture Notes, no. 284, pp. 69–104, Cambridge University Press, 2001.

[GHR$^+$90] D. Grigoriev, J. Heintz, M. F. Roy, P. Solernó, and N. Vorobjov, *Comptage des composantes connexes d'un ensemble semi-algébrique en temps simplement exponentiel*, C.R. Acad. Sci. Paris **311** (1990), 879–882.

[GK09] R. Grimson and B. Kuijpers, *Some lower bounds for the complexity of the linear programming feasibility problem over the reals*, J. Complexity **25** (2009), no. 1, 25–37.

[GLS01] M. Giusti, G. Lecerf, and B. Salvy, *A Gröbner free alternative for polynomial system solving*, J. of Complexity **17(1)** (2001), 154–211.

[GR93] L. Gournay and J. J. Risler, *Construction of roadmaps in semi-algebraic sets*, Appl. Algebra Eng. Commun. Comput. **4** (1993), 239–252.

[Gri88] D. Grigoriev, *Complexity of deciding tarski algebra*, J. Symb. Computation **5** (1988), no. 1/2, 65–108.

[Gri00] ———, *Topological complexity of the range searching*, J. Complexity **16** (2000), no. 1, 50–53.

[Gri07] R. Grimson, *A lower bound for the complexity of linear optimization from a quantifier-elimination point of view*, Dagstuhl Proceedings: Constraint Databases, Geometric Elimination and Geographic Information Systems, 2007.

[GV88] D. Grigoriev and N. Vorobjov, *Solving systems of polynomial inequalities in subexponential time*, J. Symb. Computation **5** (1988), no. 1/2, 37–64.

[GV92] ———, *Counting connected components of a semialgebraic set in subexponential time*, Computational Complexity **2** (1992), 133–186.

[Har00] R. Hartshorne, *Geometry: Euclid and beyond*, Springer, 2000.

[Hei83] J. Heintz, *Definability and fast quantifier elimination over algebraically closed fields*, Theoretical Computer Science **24** (1983), 239–277.

[Hil99] D. Hilbert, *Foundations of geometry*, Open Court, La Salle, 1899.

[HK04] J. Heintz and B. Kuijpers, *Constraint databases, data structures and efficient query evaluation*, Proceedings of the 1st International Symposium "Applications of Constraint Databases" (CDB'04), Lecture Notes in Computer Science, vol. 3074, Springer, 2004, pp. 1–24.

[HM93] J. Heintz and J. Morgenstern, *On the intrinsic complexity of elimination theory*, J. Complexity **9** (1993), no. 4, 471–498.

[HMPW98] J. Heintz, G. Matera, L. Pardo, and R. Wachenchauzer, *The intrinsic complexity of parametric elimination methods*, Electronic J. SADIO **1** (1998), no. 1, 37–51.

[HRS90a] J. Heintz, M. F. Roy, and P. Solernó, *Single exponential path finding in semialgebraic sets. Part 1: The case of a regular bounded hypersurface*, AAECC, 1990, pp. 180–196.

[HRS90b] ______, *Sur la complexité du principe de Tarski-Seidenberg*, Bull. SMF **118** (1990), 101–126.

[HRS93] ______, *On the theoretical and practical complexity of the existential theory of reals*, Computer J. **36** (1993), no. 5, 427–431.

[HRS94a] ______, *Description of the connected components of a semialgebraic in single exponential time*, Discrete & Computational Geometry **11** (1994), 121–140.

[HRS94b] ______, *Single exponential path finding in semi-algebraic sets II: The general case*, Algebraic geometry and its applications, collections of papers from Abhyankar's 60-th birthday conference, Purdue University, West-Lafayette, 1994.

[HSR89] J. Heintz, P. Solernó, and M. F. Roy, *On the complexity of semialgebraic sets*, IFIP Congress, 1989, pp. 293–298.

[JS00] G. Jeronimo and J. Sabia, *On the number of sets definable by polynomials*, J. of Algebra **227** (2000), no. 2, 633–644.

[Kha79] L. G. Khachiyan, *A polynomial algorithm in linear programming*, Soviet Mathematics Doklady **20** (1979), 191–194.

[KLP00] G. Kuper, L. Libkin, and J. Paredaens, *Constraint Databases*, Springer-Verlag, 2000.

[Knu98] D. E. Knuth, *Art of computer programming, volume 3: Sorting and searching*, second ed., Addison-Wesley Professional, 1998.

[KOG07] B. Kuijpers, W. Othman, and R. Grimson, *A case study of the difficulty of quantifier elimination in constraint databases: the alibi query in moving object databases*, CoRR **abs/0712.1996** (2007).

[KOG10] ______, *An analytic solution to the alibi query in the space-time prisms model for moving object data*, To appear in the International J. of Geographical Information Science, 2010.

[Koi00] P. Koiran, *Circuits versus trees in algebraic complexity*, Lecture Notes in Computer Science, vol. 1770, pp. 35–52, Springer-Verlag, London, UK, 2000.

[Laz88] D. Lazard, *Quantifier elimination: optimal solution for two classical examples*, J. Symb. Computation **5** (1988), no. 1-2, 261–266.

[LB01] M.K. Ganapathy L. Babai, L. Ranyai, *On the number of zero-patterns of a sequence of polynomials*, J. American Math. Soc. **14** (2001), 717–735.

[Lic90] T. Lickteig, *On semialgebraic decision complexity*, 1990, Technical Report TR-90-052, Int. Comput. Sc. Inst, Berkeley. Habilitationsschrift, Universitat Tubingen.

[Lic96] ______, *Semi-algebraic decision complexity, the real spectrum, and degree*, J. Pure Appl. Algebra **110** (1996), 131–184.

[LSJM40] H. G. Liddell, R. Scott, H. S. Jones, and R. Mckenzie, *A greek-english lexicon*, Clarendon Press, Oxford, 1940.

[MadH84] F. Meyer auf der Heide, *A polynomial linear search algorithm for the n-dimensional knapsack problem*, J. ACM **31** (1984), no. 3, 668–676.

[MadH88] ______, *Fast algorithms for n-dimensional restrictions of hard problems*, J. ACM **35** (1988), no. 3, 740–747.

[Mei93] S. Meiser, *Point location in arrangements of hyperplanes*, Inf. Comput. **106** (1993), no. 2, 286–303.

[Men97] E. Mendelson, *Introduction to mathematical logic*, fourth ed., Chapman & Hall, 1997.

[Mil64] J. Milnor, *On the Betti numbers of real varieties*, Proc. AMS **15** (1964), 275–280.

[Mon74] L. Monk, *An elementary-recursive decision procedure for* $th(\mathbf{R}, +, \cdot)$, U.C., Berkeley, 1974.

[MP93] J. L. Montaña and L. M. Pardo, *Lower bounds for arithmetic networks*, Appl. Algebra Eng. Commun. Comput. **4** (1993), 1–24.

[MS68] N. Moler and P. Suppes, *Quantifier-free axioms for constructive plane geometry*, Compositio Math. **20** (1968), 143–152.

[Pad99] M. Padberg, *Linear optimization and extensions*, Springer, 1999.

[Pam01] V. Pambuccian, *Constructive axiomatizations of plane absolute, euclidean and hyperbolic geometry*, Math. Logic Q. **47** (2001), no. 1, 129–136.

[Pam08] ———, *Axiomatizing geometric constructions*, J. Applied Logic **6** (2008), no. 1, 24–46.

[Par95] L. M. Pardo, *How lower and upper complexity bounds meet in elimination theory*, Proc 11th International Symposium Applied Algebra, Algebraic Algorithms and Error Correcting Codes (AAECC-11), Paris, Lecture Notes in Computer Science, vol. 948, Springer, 1995, pp. 33–69.

[Pel04] M. Pellegrini, *Ray shooting and lines in space*, Handbook of Discrete and Computational Geometry (J. E. Goodman and J. O'Rourke, eds.), CRC Press LLC, Boca Raton, FL, 2004.

[Ren88] J. Renegar, *A faster pspace algorithm for deciding the existential theory of the reals*, 29th Annual Symposium on Foundations of Computer Science, 24-26 October 1988, White Plains, New York, USA, 1988, pp. 291–295.

[Ren92a] ———, *On the computational complexity and geometry of the first-order theory of the reals, part I: Introduction. Preliminaries. The geometry of semi-algebraic sets. The decision problem for the existential theory of the reals*, J. Symb. Computation **13** (1992), no. 3, 255–300.

[Ren92b] ———, *On the computational complexity and geometry of the first-order theory of the reals, part II: The general decision problem. Preliminaries for quantifier elimination*, J. Symb. Computation **13** (1992), no. 3, 301–328.

[Ren92c] ———, *On the computational complexity and geometry of the first-order theory of the reals, part III: Quantifier elimination*, J. Symb. Computation **13** (1992), no. 3, 329–352.

[Rev02] P. Z. Revesz, *Introduction to constraint databases*, Springer, 2002.

[Sho67] J. Shoenfield, *Mathematical logic*, Addison-Wesley, 1967.

[Sma00] S. Smale, *Mathematical problems for the next century*, Mathematics: Frontiers and Perspectives (et al. V. Arnold, ed.), IMU, AMS, 2000.

[Sno04] J. Snoeyink, *Point location*, Handbook of Discrete and Computational Geometry (J. E. Goodman and J. O'Rourke, eds.), CRC Press LLC, Boca Raton, FL, 2004.

[ST79] L. Szczebra and A. Tarski, *Metamathematical discussion of some affine geometries*, Fundamenta Mathematicae **104** (1979), 155–192.

[Str69] V. Strassen, *Gaussian elimination is not optimal*, Numerische Mathematik **14(3)** (1969), 354–356.

[Str81] ______, *The computational complexity of continued fractions*, SYMSAC '81: Proceedings of the fourth ACM symposium on Symbolic and algebraic computation (New York, NY, USA), ACM, 1981, pp. 51–67.

[Str83] ______, *The computational complexity of continued fractions*, SIAM J. Complexity **12** (1983), 1–27.

[Syl42] J. J. Sylvester, *Memoir on the dialytic method of elimination. Part I*, Philosophical Magazine **XXI** (1842), 534–539.

[Szm83] W. Szmielew, *From affine to euclidean geometry. An axiomatic approach*, Kluwer Academic Publishers, The Netherlands, 1983.

[Tar31] A. Tarski, *Sur les ensembles définissables de nombres réels*, Fundamenta Mathematicae **17** (1931), 210–239.

[Tar51] ______, *A decision method for elementary algebra and geometry*, second ed., Univ. of California Press, Berkley, CA, 1951.

[Tar59] ______, *The axiomatic method (with special reference to geometry and physics)*, ch. What is elementary geometry?, pp. 16–29, North-Holland, Amsterdam, 1959.

[TG99] A. Tarski and S. Givant, *Tarski's system of geometry*, Bull. Symb. Logic **5** (1999), 175–214.

[Veb04] O. Veblen, *A system of axioms for geometry*, Transaction of the American Mathematical Society **5** (1904), 343–384.

[vzG86a] J. von zur Gathen, *Parallel arithmetic computations: a survey*, Proceedings of the 12th symposium on Mathematical foundations of computer science 1986 (New York, NY, USA), Springer-Verlag, 1986, pp. 93–112.

[vzG86b] J. von zur Gathen, *Parallel arithmetic computations: a survey*, Proceedings of the 12th Symposium on Mathematical Foundations of Computer Science, Bratislava, Czechoslovakia, August 1986 (B. R. J. Gruska and J. Wiedermann, eds.), Lecture Notes in Computer Science, vol. 233, Springer, 1986, pp. 93–112.

[War68] H. Warren, *Lower bounds for approximation of nonlinear manifolds*, Trans. AMS **133** (1968), 167–178.

[Weg87] I. Wegener, *The complexity of boolean functions*, B. G. Teubner, and J. Wiley & Sons, 1987.

[Wei88] V. Weispfenning, *The complexity of linear problems in fields*, J. Symb. Computation **5** (1988), no. 1-2, 3–27.

[Win70] S. Winograd, *On the algebraic complexity of functions*, In Actes du Congres International des Mathématiciens **3** (1970), 283–288.

[WST83] W. Szmielew W. Schwabhäuser and A. Tarski, *metamathematische Methoden in der Geometrie*, Springer-Verlag, 1983.

[Wüt76] H. R. Wüthrich, *Ein Entscheidungsverfahren für die Theorie der reellabgeschlossenen Körper*, in Komplexität von Entscheidungsproblemen, Ein Seminar (eds. E. Specker, V. Strassen) (1976), 138–162.

# Summary

This thesis is mainly dedicated to the study of upper and lower complexity bounds of some problems in the context of semi-algebraic geometry. We present a brief summary of its contents.

In Chapter 1 we analyze the algebraic complexity of the linear programming feasibility problem over the reals and prove non-trivial lower bounds for this problem. The linear programming feasibility problem can be stated as follows: given positive integers $m > n$, a matrix $H \in \mathbf{R}^{m \times n}$ and a vector $h \in \mathbf{R}^m$ decide whether there exists a column vector $x \in \mathbf{R}^n$ such that $H \cdot x \leq h$. For the case of polyhedra defined by $2n$ halfspaces in $\mathbf{R}^n$, we prove that the set $\mathcal{I}^{(2n,n)}$ of parameters describing non-empty polyhedra, has an exponential number of *limiting hypersurfaces*. From this geometric result we obtain, as a corollary, the existence of a constant $c > 1$ such that, if dense or sparse representation is used to encode polynomials, the length of any quantifier-free formula expressing the set $\mathcal{I}^{(2n,n)}$ is bounded from below by $\Omega(c^n)$. Other related complexity results are stated; in particular, a lower bound for algebraic computation trees based on the notion of *limiting hypersurface* is presented.

In Chapter 2, we study the sign condition problem for any given a family of polynomials. Essentially, the problem consists in determining the sign condition satisfied by a fixed family of polynomials at a query point, performing as little arithmetic operations as possible. After defining precisely the sign condition and the point location problems, we introduce a method called *the dialytic method* to solve the first problem efficiently. This method involves a *linearization* of the original polynomials and provides the best known algorithm to solve the sign condition problem. Finally, using a technique that resembles that of Chapter 1, we prove a lower bounds showing that the dialytic method is almost optimal.

In Chapter 3, we discuss different data structures that can be used to solve the point location problem for a given family of polynomials. This problem

asks to determine, not only the sign condition satisfied by a family of polynomials at a query point, but also the connected component of the realization of this sign condition containing the query point. After showing how to adapt the dialytic method to this problem, we introduce, In Section 3.3, a method based on an Adapted Cylindrical Algebraic Decomposition of the space that solves the point location problem for any given family. In Section 3.4, we discuss the case of polynomials with integer coefficients given in dense bit representation introducing a method that, based on diophantine geometry, solves the point location problem for generic families of polynomials. We include a brief discussion of the local ray shooting problem.

In Chapter 4, we introduce the notion of *intrinsic description* of a linearly-constructible set and study the complexity of quantifier-elimination methods in a computational model where the output is required to be an intrinsic description of the underlying set. We introduce a quantifier-elimination algorithm in this model. It turns out that our elimination algorithm has a doubly-exponential-time complexity in the worst case, when the complexity is measured in terms of syntactic parameters (number of polynomials and of quantifier alternations, dimension of the ambient space). We show that in our computational model, our algorithm is optimal, *i.e.*, we prove a doubly-exponential lower bound in the number of quantifier alternations of the input formula.

Remarkably, we obtain simply-exponential complexity bounds on intrinsic geometric parameters of the input problem. Thus, our algorithm distinguishes between well-posed and ill-posed problems and can be inscribed in the new generation of algorithms which take also into account intrinsic, semantic invariants of the input in order to measure the complexity of the procedure.

The Chapter 5 is the only chapter not related to complexity theory. Following the tradition of mathematical logic, we introduce new first-order languages for the elementary $n$-dimensional geometry and elementary $n$-dimensional affine geometry ($n \geq 2$), based on extending the traditional languages $\mathsf{FO}(\beta, \equiv)$ and $\mathsf{FO}(\beta)$, respectively, with new function symbols. Here, $\beta$ stands for the betweenness relation and $\equiv$ for the congruence relation. We show that the associated theories admit effective quantifier elimination.

A preliminary version of Chapter 1 was presented in Dagstuhl [Gri07]. A journal version is published as [GK09]. Parts of Chapters 2 and 3 have been presented in the 11th SYNASC symposium under the title “Point Location in arrangements of algebraic hypersurfaces” and a journal version of these chapters is being prepared. Chapter 5 has been recently sent for publication. Other publications in this period include [KOG07, KOG10] and [FGG08, FGG].

# Samenvatting

Deze thesis is voornamelijk gewijd aan de studie van onder- en bovengrenzen van de complexiteit van enige problemen in de context van semi-algebraïsche meetkunde. Hier geven we een korte samenvatting van de inhoud.

In hoofdstuk 1 analyseren we we de algebraïsche complexiteit van het “linear programming feasibility”-probleem over de reële getallen en we bewijzen niet-triviale ondergrenzen voor dit probleem. Het linear-programming-feasibility-probleem kan als volgt geformuleerd worden: gegeven positieve gehele getallen $m > n$, een matrix $H \in \mathbf{R}^{m\times n}$ en een vector $h \in \mathbf{R}^m$, beslis of een kolomvector $x \in \mathbf{R}^n$ bestaat zodanig dat $H \cdot x \leq h$. Voor het geval van veelvlakken gedefinieerd door $2n$ half-ruimten in $\mathbf{R}^n$, tonen we aan dat de verzameling $\mathcal{I}^{(2n,n)}$ van parameters die niet-lege veelvlakken beschrijven een exponentieel aantal begrenzende hyperoppervlakken heeft. Als een gevolg van dit meetkundig resultaat verkrijgen we het bestaan van een constante $c > 1$ zodat, als we dense of sparse representatie gebruiken om de veeltermen te beschrijven, de lengte van iedere kwantor-vrije formule die de verzameling $\mathcal{I}^{(2n,n)}$ uitdrukt, $\Omega(c^n)$ als ondergrens heeft. We geven ook andere gerelateerde complexiteitsresultaten, in het bijzonder geven we een ondergrens voor algebraïsche berekeningsbomen die gebaseerd is op de notie van begrenzend hyperoppervlak.

In hoofdstuk 2 bestuderen we het “sign condition”-probleem voor een gegeven familie van veeltermen. Dit probleem bestaat erin het teken te bepalen van een vaste familie veeltermen in een gegeven query-punt, en dit door zo weinig mogelijk rekenkundige operaties uit te voeren. Nadat we het sign-condition- en het “point location”-probleem precies gedefinieerd hebben, voeren we de zogenaamde *dialytische methode* in om het eerste probleem efficiënt op te lossen. Deze methode houdt de linearisatie in van de gegeven veeltermen en levert het beste gekende algoritme op om het sign-condition-probleem op te lossen. Tenslotte, gebruik makend van een techniek die op de techniek uit hoofdstuk 1 lijkt, bewijzen we ondergrenzen die aantonen dat de dialytische methode bijna optimaal is.

In hoofdstuk 3 bestuderen we verschillende gegevensstructuren die gebruikt kunnen worden om het point-location-probleem op te lossen voor een gegeven familie veeltermen. Dit probleem vraagt niet enkel de teken-voorwaarden te bepalen waaraan een query-punt voldoet voor de familie veeltermen, maar het vraagt bovendien naar de samenhangingscomponent van de realisatie van de teken-voorwaarden waartoe het query-punt behoort.

Nadat we aangetoond hebben hoe de dialytische methode kan aangepast worden om dit probleem op te lossen, introduceren we in sectie 3.3 een methode die gebaseerd is op Aangepaste Cilindrische Algebraïsche Decompositie van de ruimte om het point-location-probleem op te lossen voor een geven familie van veeltermen. In sectie 3.4 bestuderen we het geval van veeltermen met gehele coefficiënten die in dense bit-representatie gegeven worden en we introduceren een methode die gebaseerd is op diophantische meetkunde en die het point-location-probleem oplost voor generische families van veeltermen. We bespreken ook het "local ray shooting"-probleem.

In hoofdstuk 4 introduceren we de notie van *intrinsieke beschrijving* van een lineaire construeerbare verzameling en we bestuderen de complexiteit van kwantor-eliminatie-methodes in een berekeningsmodel waar geëist wordt dat de output een intrinsieke beschrijving van de onderliggende verzameling is. We introduceren een kwantor-eliminatie-algoritme in dit berekeningsmodel. In het slechtste geval heeft ons algoritme een dubbel-exponentiële tijdscomplexiteit, als de complexiteit gemeten wordt in termen van syntactische parameters (aantal veeltermen en kwantoren, dimensie van de omgevende ruimte). We tonen aan dat dit algoritme optimaal is in ons berekeningsmodel. We tonen een ondergrens aan die dubbel-exponetieel is in het aantal kwantoren in de formule.

Merkwaardig genoeg bekomen we enkel-exponentiële complexiteitsgrenzen op de intrinsieke meetkundige parameters van de input. Daardoor kunnen onze algoritmen het onderscheid maken tussen goed- en slecht-geformuleerde problemen en kunnen ze aldus beschouwd worden als nieuwe-generatie algoritmen die ook intrinsieke, semantische invarianten van de input in rekening brengen om de complexiteit te meten.

Hoofdstuk 5 is het enige hoofdstuk dat niet over complexiteit handelt. Volgens de traditie van wiskundige logica, introduceren we eerste-orde talen voor de elemantaire $n$-dimensionale meetkunde en de elementaire $n$-dimensionale affiene meetkunde ($n \geq 2$), gebaseerd op het uitbreiden van de klassieke talen $\mathsf{FO}(\beta, \equiv)$ en $\mathsf{FO}(\beta)$, respectievelijk, met nieuwe functie-symbolen. Hier staat $\beta$ voor de "betweenness"-relatie en $\equiv$ voor de congruentie-relatie. We tonen aan dat de geassocieerde theorieën kwantor-eliminatie toelaten.

Een werkversie van hoofdstuk 1 werd gepubliceerd als Dagstuhl-rapport [Gri07]. Een journaal-versie werd gepubliceerd als [GK09]. Delen van hoofdstukken 2 en 3 werden voorgesteld op het 11de SYNASC Symposium onder de titel “Point location in arrangements of algebraic hypersurfaces” en een journal-versie van deze hoofdstukken is in voorbereiding. Hoofdstuk 5 is naar een journaal ter publicatie gestuurd. Andere publicaties gemaakt tijdens de afgelopen jaren zijn [KOG07, KOG10] en [FGG08, FGG].