Fakultät für Mathematik

Lehrstuhl für Effiziente Algorithmen I14

Radicals of Binomial Ideals and Commutative Thue Systems

Stefan Toman

Vollständiger Abdruck der von der Fakultät für Mathematik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

genehmigten Dissertation.

Die Dissertation wurde am 28.03.2017 bei der Technischen Universität München eingereicht und durch die Fakultät für Mathematik am 14.06.2017 angenommen.

Acknowledgments

Writing this thesis would not have been possible without all the people who supported me in one way or the other or contributed to it. I would like to express my sincere appreciation to everybody who was involved.

First of all, I wish to thank my advisor Ernst W. Mayr. I consider myself very fortunate to have had an advisor who is not only a brilliant researcher, but who also always took the time to discuss any ideas with me and guided me with invaluable help and advice on all matters related to academics, be it research, teaching, financial support or anything else I needed. I also want to thank Gregor Kemper and Bruno Buchberger for completing my examination committee and reviewing this thesis.

The financial support of my work at the Technical University of Munich came from the priority project “Algorithmic and Experimental Methods in Algebra, Geometry, and Number Theory” of the German Research Foundation. I was a member of the graduate program TopMath of the Elite Network of Bavaria and the TUM Graduate School which gave me the chance to start working on the topics of this thesis early during my studies and connected me with remarkable fellow students and friends. During my studies I was also supported by scholarships of the Max Weber-Program of the State of Bavaria and the German Academic Scholarship Foundation. I am very grateful for the support of all those institutions and the people involved there.

Additionally, I want to thank the many people who helped me with the final touch of this thesis, in particular Elisabeth Ackermann for contributing the image of cyclohexane as well as Daniel Ackermann, Moritz Fuchs, Anna Maria Mathes and Sabine Toman for long hours of proofreading all the versions of this thesis and calling my attention to numerous parts that have been considerably improved since the first drafts.

Ernst Bayer fixed my computer every time I broke something and I am grateful that he always knew how to make it work again. However, I did not spend my entire time at the university writing this thesis. For all the homeworks solved and later courses taught together, lunch breaks, and gaming sessions I would like to thank my great colleagues, fellow students, and friends who made me truly enjoy my time at Garching, in particular Moritz Fuchs, Philipp Hoffmann, Christian Müller, Chris Pinkau, Hanjo Täubig, Harald Räcke, Benedikt Schamberger, Saskia Schiele and Richard Stotz to just name a few.

Last, but not least, I want to thank my family and in particular my fiancée Anna for the unconditional support during the whole course of my life, for always having an ear for any difficulties I might face, and all the wonderful time spent together.

Thank you for making this possible!

Abstract

Solving systems of polynomial equations is one of the most fundamental problems of computer algebra. The theory of Gröbner Bases allows for algorithmic solutions of many problems of polynomial ideals, but their computation takes an exponential amount of space in the worst case. Therefore, such computations are only feasible for subclasses of systems with small generators or ones that exhibit a special structure. One of the most interesting subclasses is the set of binomial ideals as they have more structure than general polynomial ideals, but still comprise the full complexity. We present a new algorithm for computing the radical of a binomial ideal which uses binomials as intermediate results of the computations only, but matches the running time of the best known algorithms. With this algorithm we can define radicals of commutative Thue systems.

Zusammenfassung

Das Lösen von Polynomgleichungssystemen gehört zu den grundlegendsten Problemen der Computeralgebra. Die Theorie der Gröbner-Basen ermöglicht algorithmische Lösungen von vielen Problemen auf Polynomgleichungssystemen, aber ihre Berechnung hat im schlechtesten Falle einen exponentiellen Speicherplatzbedarf. Daher sind diese Berechnungen nur für Teilklassen solcher Systeme möglich, die kleine Erzeuger haben oder eine besondere Struktur aufweisen. Eine der interessantesten Teilklassen ist die Menge der Binomideale, da sie mehr Struktur als allgemeine Polynomideale haben und trotzdem die volle Komplexität aufweisen. Wir präsentieren einen neuen Algorithmus um Radikale von Binomidealen zu berechnen, welcher nur Binome als Zwischenergebnisse verwendet und trotzdem die Laufzeit der besten bekannten Algorithmen aufweist. Mit diesem Algorithmus können wir Radikale von kommutativen Thue-Systemen definieren.

Contents

Part I

Introduction

1 Motivation

1.1 An Example from Chemistry

To motivate the theorems and tools used within this thesis we will discuss an application from chemistry first. Organic chemistry is the subdiscipline studying materials containing carbon atoms. Those carbon atoms have bonds to other atoms. Carbon has a valence of four, meaning that each carbon atom can enter four bonds. With these bonds they can attach to other carbon atoms or atoms of other elements. The carbon atoms in a molecule may form chains, cycles or more complicated structures. We want to discuss carbon cycles in this example.

The easiest cycle of carbon atoms is called cyclohexane. Cyclohexane molecules consist of a cycle of six carbon atoms. The bonds within the cycle occupy two bonds of each carbon atom. All remaining bonds are filled with hydrogen atoms which have a valence of one. A line angle diagram of cyclohexane is shown in Figure 1.1. These diagrams depict the bonds by lines, carbon atoms are located on the unlabeled corners of the lines. Bonds which are connected to hydrogen atoms are usually omitted. A line angle diagram with all atoms explicitly drawn and no hydrogen bonds omitted is displayed in Figure 1.2.

Carbon atoms organize themselves typically in tetrahedral structures. Such a structure is the most dense regular close-packing of equal spheres as proven by Carl Friedrich Gauß in 1831 [Gau31]. The claim that there is no irregular packing with a higher density is known as the Kepler Conjecture and was proven by Thomas Hales only much later in 1998 and published after a profound check of his computer-based proof in 2009 [Hal05].

In the line angle diagram it seems like all carbon atoms are in one plane, which is not true in

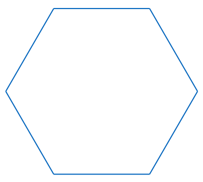


Figure 1.1: The line angle diagram of a cyclohexane molecule.

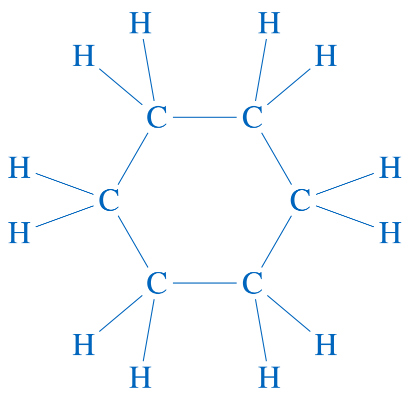


Figure 1.2: An extended line angle diagram of a cyclohexane molecule.

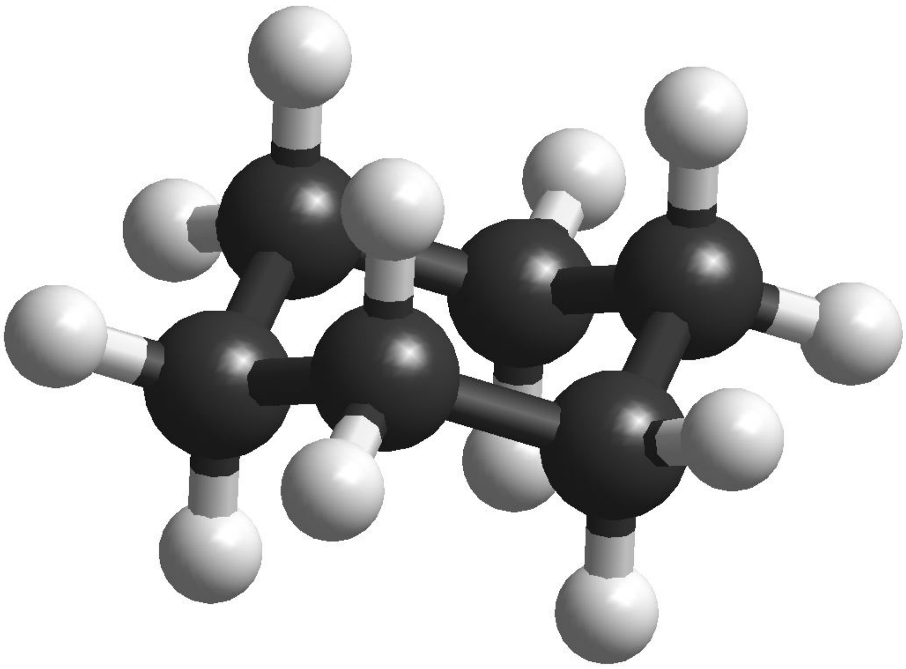


Figure 1.3: Ball-and-Stick model of a cyclohexane molecule in chair conformation. White balls represent hydrogen atoms, black balls represent carbon atoms. Image by Elisabeth Ackermann.

practice. The angles inside a tetrahedron $\arccos\left(-\frac{1}{3}\right) \approx 109.4712°$ are called the tetrahedral angle. If all carbon atoms were in a planar hexagon the angle between adjacent bonds would be 120°. To achieve the tetrahedral angle between adjacent bonds the carbon atoms rotate around their bonds and wrap around the plane. With these rotations the atoms move slightly away from the plane which is called the mean plane of the molecule. Such a three-dimensional arrangement of the carbon atoms is called a conformation of the cyclohexane molecule. The so-called chair conformation of cyclohexane is shown as an example in Figure 1.3.

The conformation of the molecule defines the relative positions of all contained atoms since the hydrogen atoms try to maximize their distance. Their positions are therefore determined by the positions of the carbon atoms. One bond to a hydrogen atom of each carbon atom is almost perpendicular to the mean plane of the carbon atoms. These atoms are alternating above or below the mean plane and are called axial. The other six hydrogen atoms are almost contained in the mean plane and are called equatorial.

In nature cyclohexane stabilizes at a certain balance of the different conformations of the molecules. The conformation has a considerable impact on the properties of the material as different conformations have different amounts of energy bound in their bonds. Thus, energy may be added or removed from cyclohexane by changing the conformation. Also, the conformation changes when heating or cooling the material. Experimental data shows that at room temperature only 0.1% of the molecules are in so-called twist-boat conformation whereas at 800° Celsius approximately 30% are in twist-boat conformation.

At room temperature cyclohexane is a colorless, flammable liquid used in the production of nylon and cleaning products. It is important to know the possible conformations of the molecules for the industrial production and usage of cyclohexane. Also, many other materials from organic chemistry feature carbon cycles. In order to understand them it is important to know about their easiest form.

To describe a conformation of cyclohexane it is enough to describe the rotations of the six carbon atoms around their bonds in the carbon cycle. Since the cycle needs to be closed it is not needed to have six indeterminates to describe the angles. In 1987 Andreas Dress described the possible conformations of cyclohexane using the following system of four equations involving three indeterminates x_1, x_2, and x_3 over the real numbers [MMN89].

$$f_1 := \det\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & \frac{8}{3} & x_1 & \frac{8}{3} \\ 1 & 1 & 0 & 1 & \frac{8}{3} & x_2 \\ 1 & \frac{8}{3} & 1 & 0 & 1 & \frac{8}{3} \\ 1 & x_1 & \frac{8}{3} & 1 & 0 & 1 \\ 1 & \frac{8}{3} & x_2 & \frac{8}{3} & 1 & 0 \end{pmatrix} = 0, \quad f_2 := \det\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & \frac{8}{3} & x_2 & \frac{8}{3} \\ 1 & 1 & 0 & 1 & \frac{8}{3} & x_3 \\ 1 & \frac{8}{3} & 1 & 0 & 1 & \frac{8}{3} \\ 1 & x_2 & \frac{8}{3} & 1 & 0 & 1 \\ 1 & \frac{8}{3} & x_3 & \frac{8}{3} & 1 & 0 \end{pmatrix} = 0$$

$$f_3 := \det\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & \frac{8}{3} & x_3 & \frac{8}{3} \\ 1 & 1 & 0 & 1 & \frac{8}{3} & x_1 \\ 1 & \frac{8}{3} & 1 & 0 & 1 & \frac{8}{3} \\ 1 & x_3 & \frac{8}{3} & 1 & 0 & 1 \\ 1 & \frac{8}{3} & x_1 & \frac{8}{3} & 1 & 0 \end{pmatrix} = 0, \quad f_4 := \det\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & \frac{8}{3} & x_1 & \frac{8}{3} & 1 \\ 1 & 1 & 0 & 1 & \frac{8}{3} & x_2 & \frac{8}{3} \\ 1 & \frac{8}{3} & 1 & 0 & 1 & \frac{8}{3} & x_3 \\ 1 & x_1 & \frac{8}{3} & 1 & 0 & 1 & \frac{8}{3} \\ 1 & \frac{8}{3} & x_2 & \frac{8}{3} & 1 & 0 & 1 \\ 1 & 1 & \frac{8}{3} & x_3 & \frac{8}{3} & 1 & 0 \end{pmatrix} = 0$$

In this formulation the indeterminates x_1, x_2, and x_3 describe distances between the carbon atoms and thus only positive solutions are of interest in this application. This system of equations has been solved by Herbert Melenk, Hans-Michael Möller and Winfried Neun in 1989 using Gröbner Basis techniques [MMN89] and all conformations of cyclohexane are well-known today.

In this thesis we will discuss Algorithms to find properties of the solutions of similar equations or solve similar equations and the complexity of those problems.

1.2 Summary of the Thesis

Gröbner Bases which are used to solve the systems of equations like the one presented in Section 1.1 were introduced in 1965 by Bruno Buchberger [Buc65] and named after his advisor Wolfgang Gröbner. They provide a powerful tool for symbolic computations with polynomial systems of equations and are the basis of many algorithms in the field of computer algebra. In particular, Gröbner Bases allow for membership tests in polynomial ideals, i.e. checking whether a given equation is an implication of other equations, and the elimination of indeterminates from these systems.

The speed of computers and the quality of the available software has increased vastly since 1989. While it was an achievement to solve the system of equations given above at that point of time, nowadays the computation of a Gröbner Basis of the given set of equations is done within seconds. An implementation of this system of equations using the computer algebra system Macaulay 2 [GS] can be found in Listing A1.1 in the appendix at Section A.1.

Cyclohexane is one of the easiest molecules that exists though and for bigger systems of equations we are not able to find solutions as fast. Known algorithms require a running time that is double exponential in the size of the input. This means that the running time squares with every additional indeterminate introduced into the system. Running times thus explode even for relatively small examples. There are examples of systems of polynomial equations with less than 100 indeterminates such that the computation of a Gröbner Basis is not doable within a reasonable time on modern computers. When solving systems of equations in practice problems of that size and also much bigger ones appear, for instance when dealing with molecules having many more atoms than cyclohexane.

This double exponential growth of the running time is inevitable as shown by Ernst W. Mayr and Albert R. Meyer in 1982 [MM82]. They have proven that the word problem for polynomial ideals is exponential space-complete by reducing the halting problem for three-counter machines, which is known to be exponential space-complete, to the uniform word problem for commutative semigroups. It is still an open question whether the same problems can be computed in exponential space and double exponential time, but this is strongly believed to be the case.

With this growth of the running time large problems will not be solvable even if the computational power continues to increase like in the last decades. Thus, it cannot be expected that we will be able to solve large systems of polynomial equations within reasonable amounts of time without fundamental changes in the computational machines available.

For this reason we will consider complexity classes and degree bounds for subclasses of polynomial ideals in this thesis that may allow for better algorithms. There are subclasses of polynomial ideals and problems that do already implicitly contain the full complexity of general polynomial ideals whereas for others we will find better complexity classes. We will also use the techniques known for polynomial ideals for solving related problems.

Two of these subclasses are the sets of binomial ideals and pure binomial ideals. We will present a new method for modeling binomial ideals with pure binomial ideals and provide new complexity bounds on the radical word problem of binomial ideals. Also, we will contribute a new algorithm for computing the radical of binomial ideals that uses binomials only. This algorithm allows the usage of specialized data structures that can only handle binomials.

Additionally, with the help of this algorithm we can define radicals of commutative Thue systems, certain term replacement systems from theoretical computer science that are equivalent to binomial ideals. With this result it is now possible to use techniques from the theory of term replacement systems for the analysis of radicals of polynomial ideals. We will also analyze the complexity of our algorithm and prove that it matches that asymptotic running time bounds of the best known algorithms for computing radicals of polynomial ideals.

2 About this Thesis

2.1 Structure

This thesis is structured into several parts, chapters and sections. In Chapter 1 we motivate the word problem and radical word problems of polynomial ideals and name the main tools needed for their solution. Chapter 2 contains this summary of the thesis and hints about the basic notation we use within this thesis.

Afterwards, we define the basic algebraic structures that we need for our main results. As we modify some very basic structures in later chapters, we present detailed definitions of the most common algebraic structures. In Chapter 3 we present groups, rings, and fields. Polynomial rings, polynomial ideals and their properties are introduced in Chapter 4. Gröbner Bases are introduced as the main tool that we need for computations with polynomial ideals in Chapter 5 together with algorithms to compute them.

We want to discuss the computational complexity of numerous Algorithms. To do so, we introduce our computational model and basic complexity theory in Chapter 6. In Chapter 7 we apply the computational model to problems in computer algebra and present relevant degree bounds and complexity results.

In the following chapters we analyze subclasses of polynomial ideals that allow for specialized algorithms or better degree and complexity bounds. Radical ideals are discussed in Chapter 8, binomial ideals and pure binomial ideals are discussed in Chapter 9, and toric ideals are discussed in Chapter 10. All those subclasses are combined in Chapter 11 to discuss the cellular decomposition, a decomposition of radical binomial ideals into toric ideals. We also present a new way of modeling binomial ideals as pure binomial ideals and provide a new complexity bound for the radical word problem of binomial ideals in this chapter.

The following chapters contain the main result of this thesis, an algorithm to compute the radical of commutative Thue systems and binomial ideals using binomials only. Those term replacement systems are introduced in Chapter 12. In Chapter 13 we prove that for every binomial contained in the radical of a binomial ideal there is a power of the binomial such that all terms of that power of the binomial are equivalent modulo the binomial ideal. This allows us to contribute a definition of radicals of commutative Thue systems in the same chapter. Degree bounds on that power are presented in Chapter 14 which allows for a complexity analysis of our algorithm to compute radicals of binomial ideals.

The findings of this thesis are summarized again in the conclusion in Chapter 15.

2.2 Fundamentals and Notation

We assume that the reader is familiar with the basics of mathematical notation and fundamental mathematics like basic set theory. We will also use elemental mathematical concepts and definitions like injective, surjective, and bijective maps or equivalence classes without reference. For a good introduction to this refer for instance to Gerd Fischer's book [Fis14]. On the other hand all basic algebraic structures will be introduced from scratch as this thesis is about algebraic topics and we want to adjust the definitions of some basic objects like polynomial rings in later sections.

We will use standard notation wherever possible. The non-negative integers will be denoted by $\mathbb{N}_0$ and the positive integers will be denoted by $\mathbb{N}_{>0}$. When working with polynomial rings, $n \in \mathbb{N}_0$ will be the number of indeterminates $x_1, \ldots, x_n$ if not stated otherwise. For vectors $u \in \mathbb{N}_0^n$ we will write $\underline{x}^u$ as a short form of $x_1^{u_1} \cdot \ldots \cdot x_n^{u_n}$.

When defining the algebraic structures used within this thesis we will often state some usual conditions on them. For instance we will always assume that the coefficient rings R of our polynomial rings are commutative. These conditions will nevertheless be mentioned in all following theorems for clarity.

We will omit punctuation marks after equations if this does not decrease the readability of the text. For example, sentences may end with a formula like

$$e^{i\pi} = -1$$

Important theorems and definitions are highlighted with gray background color. All theorems and definitions are labeled using the chapter and an index such that the labels are unique throughout the whole thesis.

Part II

Algebraic Foundations

3 Groups, Rings, and Fields

3.1 Groups

In this chapter we are going to introduce the fundamentals of the algebraic objects we want to study. We will mainly introduce the definitions and theorems needed for the work in later chapters and sketch many proofs, in particular we will often just state properties of the structures presented here without proving them if the proofs are not short. On the other hand, we state some interesting results for the sake of completeness that are not directly needed later during this thesis. Details about the algebraic structures and detailed proofs can be found in many textbooks, for instance the ones by Siegfried Bosch [Bos09], Christian Karpfinger and Kurt Meyberg [KM10], or Bartel L. van der Waerden [vdWANB43].

To construct the algebraic objects we are interested in we will always need a set of numbers or other objects M to work with. Also, we need to define an operation $\circ$ between the elements of M. Such an operation is a map $\circ : M \times M \to M$ and we usually write $a \circ b$ instead of $\circ(a, b)$ for $a, b \in M$. A set together with an associative operation is called a semigroup.

Definition 3.1 A semigroup $(M, \circ)$ consists of a non-empty set M and an operation $\circ : M \times M \to M$ on M such that $\circ$ is associative, i.e.

$$(a \circ b) \circ c = a \circ (b \circ c)$$

for all $a, b, c \in M$. Additionally, if there is also a neutral element $e \in M$ of $\circ$, i.e. an element with

$$e \circ a = a \circ e = a$$

for all $a \in M$, we call $(M, \circ, e)$ a monoid.

Since we have associativity for the operations of semigroups and monoids we will always omit brackets when using such an operation to simplify notation. We call any of these structures finite if the set M is finite. In this case, the order of the structure is the number of elements contained in M.

The neutral element is also called identity element or just identity. It is always unique in a monoid. To show this assume that there were two elements $e, e' \in M$ with

$$e \circ a = a \circ e = e' \circ a = a \circ e' = a$$

for all $a \in M$. This would imply $e \circ e' = e$ for $a = e$ and $e \circ e' = e'$ for $a = e'$ which means $e = e'$. Thus, we usually omit the neutral element in the notation $(M, \circ, e)$ of the monoid since

set	$\mathbb{N}_{>0}$	$\mathbb{N}_{>0}$	$\mathbb{N}_0$	$\mathbb{N}_0$	$\mathbb{Z}$	$\mathbb{Z}$	S_n $n \leq 2$	S_n $n \geq 3$
operation	$+$	$\cdot$	$+$	$\cdot$	$+$	$\cdot$	$\circ$	$\circ$
neutral elem.	-	1	0	1	0	1	id	id
associative	✓	✓	✓	✓	✓	✓	✓	✓
commutative	✓	✓	✓	✓	✓	✓	✓	-
semigroup	✓	✓	✓	✓	✓	✓	✓	✓
monoid	-	✓	✓	✓	✓	✓	✓	✓
group	-	-	-	-	✓	-	✓	✓

Table 3.2: Some examples of semigroups, monoids, and groups.

it can always uniquely be determined from the set and operation. Likewise, if the operation is clear from context we will also omit it and say that M is a semigroup, monoid or any of the following structures.

Examples of monoids are the integers with addition $(\mathbb{Z}, +, 0)$, the integers with multiplication $(\mathbb{Z}, \cdot, 1)$, the non-negative integers with addition $(\mathbb{N}_0, +, 0)$, the non-negative integers with multiplication $(\mathbb{N}_0, \cdot, 1)$, and the positive integers with multiplication $(\mathbb{N}_{>0}, \cdot, 1)$. All monoids are also semigroups. Note that the positive integers with addition $(\mathbb{N}_{>0}, +)$ are a semigroup but no monoid since there is no neutral element. An overview of the examples in this section can be found in Table 3.2.

Similar to these examples we usually stick to the notation of additive semigroups and monoids with operation $+$ and neutral element 0 or multiplicative semigroups and monoids with operation $\cdot$ (which is often omitted to write the shorter ab instead of $a \cdot b$ for $a, b \in M$) and neutral element 1. For additive semigroups or monoids we usually write

$$2 := 1 + 1, 3 := 1 + 1 + 1, 4 := 1 + 1 + 1 + 1, \ldots$$

The monoid $(\mathbb{Z}, +)$ also has the property that we have inverse elements for each number, namely for each $a \in \mathbb{N}_0$ we have an element $-a \in M$ with $a + (-a) = 0$. Together with this property we call $(\mathbb{Z}, +)$ a group.

Definition 3.3 A group $(M, \circ)$ is a monoid with the property that for each $a \in M$ there is an inverse element $b \in M$ with $a \circ b = b \circ a = e$ where e is the neutral element of $\circ$.

For additive monoids we usually write the inverse element as $-a := b$, i.e.

$$a + (-a) = (-a) + a = 0$$

for all $a \in M$, and for multiplicative monoids we usually write the inverse element as $a^{-1} := b$, i.e.

$$aa^{-1} = a^{-1}a = 1$$

for all $a \in M$.

The inverse element of a fixed element $a \in M$ is always unique. To prove this, assume there were two inverse elements $b, c \in M$ with

$$a + b = b + a = a + c = c + a = 0$$

In this case we have

$$b = b + 0 = b + a + c = 0 + c = c$$

which shows that both inverse elements have to be identical. This justifies our notation of the inverse element.

All examples seen above are commutative which can formally be defined as follows.

Definition 3.4 An operation $\circ : M \times M \to M$ on a set M is called commutative if and only if

$$a \circ b = b \circ a$$

for all $a, b \in M$. A semigroup, monoid or group $(M, \circ)$ is called commutative if the corresponding operation $\circ$ is commutative. A commutative group is also called an Abelian group in honor of the Norwegian mathematician Niels Abel.

For an example of a non-commutative group we will consider the symmetric group S_n for $n \in \mathbb{N}_{>0}$. A symmetric group acting on a set X consists of all permutations of the set X, i.e. all bijective maps from X to itself. The operation of the symmetric group is the composition of maps, i.e. for bijections $\mu, \tau : X \to X$ we have $\phi = \mu \circ \tau$ for a bijection $\phi : X \to X$ with $\phi(a) := \mu(\tau(a))$ for all $a \in X$. For all $n \in \mathbb{N}_{>0}$ we write S_n for the symmetric group acting on $\{1, \ldots, n\}$.

While S_1 and S_2 are rather simple Abelian groups of order one and two, respectively, the groups S_n for $n \geq 3$ are non-Abelian. To see this take $\mu, \tau : \{1, \ldots, n\} \to \{1, \ldots, n\}$ with

$$\mu(1) := 2, \mu(2) := 1, \mu(3) := 3, \tau(1) := 1, \tau(2) := 3, \tau(3) := 2$$

and

$$\mu(i) := \tau(i) := i$$

for $i \in \mathbb{N}_{>0}$, $i > 3$. In this case we have

$$(\mu \circ \tau)(1) = 2, \text{ but } (\tau \circ \mu)(1) = 3$$

and thus $\mu \circ \tau \neq \tau \circ \mu$. A visualization of these maps can be found in Figure 3.5.

The symmetric groups are in particular interesting because all finite groups are equivalent to substructures of them. To state this formally we need to define those substructures first.

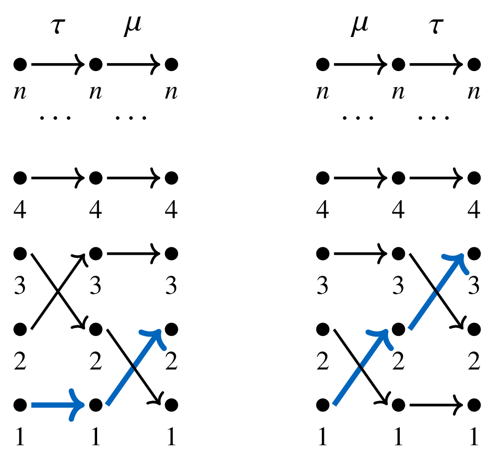


Figure 3.5: An example showing that the symmetric groups S_n for $n \in \mathbb{N}_{>0}$, $n \geq 3$ are non-Abelian.

Definition 3.6 Let N and M be two sets with $N \subseteq M$ and $\circ : M \times M \to M$ be an operation on M (and thus also on N). $(N, \circ)$ is a subsemigroup or subgroup of a semigroup or group $(M, \circ)$, respectively, if $(N, \circ)$ is a semigroup or group itself and $a \circ b \in N$ for all $a, b \in N$. $(N, \circ)$ is a submonoid of a monoid $(M, \circ)$ if $(N, \circ)$ is a monoid itself, $a \circ b \in N$ for all $a, b \in N$, and the neutral element of $\circ$ is contained in N.

Note that in contrast to submonoids for subgroups we do not need to require that the neutral element of $\circ$ is contained in N. This is because for any $a \in N$ for a subgroup N we know that the inverse element of a and its product with a, namely the neutral element, are also contained in N since N is a group itself. This argument does not work on monoids as there do not have to be inverse elements.

We have seen above that the neutral element in a monoid is always unique, so the neutral elements of N and M need to be the same. This does not work for submonoids also due to the lack of inverse elements. For example consider the monoid $(\mathbb{N}_0 \cup \{e\}, +)$ with the usual addition and $a + e := a$, $e + a := a$ for all $a \in \mathbb{N}_0 \cup \{e\}$. The neutral element of this monoid is e. $(\mathbb{N}_0, +)$ is not a submonoid of $(\mathbb{N}_0 \cup \{e\}, +)$ since it does not contain e, even though it is a monoid itself.

We know that for submonoids and subgroups the neutral element of the bigger structure is also contained in the substructure. Thus, it also has to be a neutral element there and we have seen above the neutral elements are unique. This shows that all substructures of monoids and groups have the same neutral element. Similarly, for a subgroup $(N, \circ)$ of $(M, \circ)$ and an element $a \in N$ the inverse element of a is the same in both groups.

To be able to describe similarities of these structures we need to define when two structures are essentially the same even though their elements have different names.

Definition 3.7 A map $\phi : M \to N$ between two semigroups $(M, \oplus)$ and $(N, \otimes)$ is a semigroup homomorphism if and only if $\phi(a \oplus b) = \phi(a) \otimes \phi(b)$ for all $a, b \in M$. A map between monoids is a monoid homomorphism if and only if it is a semigroup homomorphism and the neutral element of the domain is mapped to the neutral element of the codomain. A map between groups is a group homomorphism if and only if it is a semigroup homomorphism.

A bijective homomorphism is called an isomorphism, and two structures having an isomorphism between them are called isomorphic, which is denoted by $M \simeq N$.

Isomorphisms between semigroups mean that all structure can be carried over from one semigroup to another and likewise for monoids and groups. Even though it is not mentioned in the definition, the neutral element of a group is mapped to the neutral element of another group by any group homomorphism. This is because

$$\phi(a) = \phi(a \oplus e_M) = \phi(a) \otimes \phi(e_M)$$

for all $a \in M$ where e_M is the neutral element of M. We similarly get $\phi(a) = \phi(e_M) \otimes \phi(a)$ for all $a \in M$ and thus $\phi(e_M)$ is a neutral element of the image of ϕ which is a subgroup of N. We know that the neutral element of a group is unique and therefore $\phi(e_M) = e_N$ where e_N is the neutral element of N. The inverse elements are also carried over by group homomorphisms. This is true because

$$\phi(a) \otimes \phi(a^{-1}) = \phi(a \oplus a^{-1}) = \phi(e_M) = e_N \Rightarrow \phi(a)^{-1} = \phi(a^{-1})$$

for all $a \in M$.

Isomorphic groups exhibit the same structure. In particular, finite isomorphic groups have the same number of elements. We defined S_n to be the symmetric group acting on $\{1, \ldots, n\}$, because all symmetric groups acting on a set of size $n \in \mathbb{N}_{>0}$ are isomorphic. Symmetric groups are in particular interesting since all groups are isomorphic to subgroups of symmetric groups. This fact is known as Cayley's Theorem as it was published first by Arthur Cayley in 1854 [Cay54].

Theorem 3.8 (Cayley's Theorem [Cay54]) Let $(G, \oplus)$ be a group. $(G, \oplus)$ is isomorphic to a subgroup of the symmetric group operating on the set G.

Proof For each $a \in G$ we define the map

$$\phi_a : G \to G, b \mapsto a \oplus b$$

We consider the set $H := \{\phi_a \mid a \in G\}$. $(H, \circ)$ is a subgroup of the symmetric group operating on the set G. The map

$$\tau : G \to H, a \mapsto \phi_a$$

is an isomorphism between $(G, \oplus)$ and $(H, \circ)$. □

3.2 Rings

Groups are the most basic structures we use, but for most applications we will have two different operations on the same set of elements that work together. We usually call them addition and multiplication. We will call a structure having two operations with some basic properties a ring. A ring can be defined as an additive Abelian group and a monoid operating on the same set of elements such that both operations are distributive.

Definition 3.9 Let $(R, +, 0)$ be an Abelian group and $(R, \cdot, 1)$ be a monoid with $0 \neq 1$. We call $(R, +, 0, \cdot, 1)$ a ring if and only if $+$ and $\cdot$ are distributive, i.e. for all $a, b, c \in R$ we have

$$a \cdot (b + c) = (a \cdot b) + (a \cdot c) \text{ and } (b + c) \cdot a = (b \cdot a) + (c \cdot a)$$

We call the operation $+$ addition and $\cdot$ multiplication. The ring $(R, +, 0, \cdot, 1)$ is called commutative if and only if $(R, \cdot, 1)$ is commutative.

As for semigroups, monoids, and groups we will usually omit the operations and neutral elements in later sections when they are clear from context and call the set R a ring. The operation $\cdot$ is also often omitted in equations and by writing two elements next to each other we denote their product. Additionally, we usually omit brackets by computing multiplications before additions. For instance, the distributive rules can be written as $a(b + c) = ab + ac$ and $(b + c)a = ba + ca$ for all $a, b, c \in R$. For each element $a \in R$ we will denote the additive inverse element by $-a$ and multiplicative inverse elements a^{-1} if existent.

Note that there are different definitions of rings in the literature. Some authors also allow the multiplicative substructure to be a semigroup instead of a monoid and call the rings as defined here “rings with unity”. On the other hand, authors defining rings as we do often refer to rings without multiplicative identity element as pseudo-rings or rngs. The word rng is used because the missing letter i of the word ring represents the missing multiplicative identity element. Some authors also allow $1 = 0$ which results in one additional ring, namely $\{0\}$. We do not consider this as a ring to avoid some special cases without additional insight.

A typical example of a ring is the set of integers $\mathbb{Z}$ together with the usual addition and multiplication as the ring $(\mathbb{Z}, +, 0, \cdot, 1)$. More examples are collected in Table 3.10.

Even though only few conditions are contained in the definition of a ring, they do already imply many theorems that we know from the example of the ring of integers mentioned above. For instance we have $0 = 0 \cdot a$ for all $a \in R$ since

$$0 \cdot a = (0 + 0) \cdot a = 0 \cdot a + 0 \cdot a$$

and adding the additive inverse of $0 \cdot a$ to this equation results in $0 = 0 \cdot a$. Similarly, we get $0 = a \cdot 0$ for all $a \in R$.

set	$\mathbb{Z}$	$\mathbb{Q}$	$\mathbb{R}$	$\{f : \mathbb{Q} \to \mathbb{Q}\}$	$2\mathbb{Z}$	$2^{\{1,\dots,10\}}$	$\mathbb{R}^{2\times 2}$
addition	$+$	$+$	$+$	$+$ (pointwise)	$+$	$\triangle$	$+$ (pointwise)
multiplication	$\cdot$	$\cdot$	$\cdot$	$\cdot$ (pointwise)	$\cdot$	$\cap$	matrix mult.
additive identity	0	0	0	0 (pointwise)	0	$\emptyset$	$\begin{pmatrix}0 & 0\\ 0 & 0\end{pmatrix}$
mult. identity	1	1	1	1 (pointwise)	-	$\{1,\dots,10\}$	$\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$
pseudo-ring/rng	✓	✓	✓	✓	✓	✓	✓
ring	✓	✓	✓	✓	-	✓	✓
commutative ring	✓	✓	✓	✓	-	✓	-
field	-	✓	✓	-	-	-	-

Table 3.10: Some examples of rings and fields.

We can also show that we can move minus signs for general rings as we are used to from the integers and have $-(a \cdot b) = (-a) \cdot b$ for all $a, b \in R$. This holds because

$$(-a) \cdot b + a \cdot b = ((-a) + a) \cdot b = 0 \cdot b = 0$$

and therefore $-(a \cdot b) = (-a) \cdot b$. Again, we can similarly prove that $-(a \cdot b) = a \cdot (-b)$ for all $a, b \in R$.

Other properties of the ring of integers differ from general rings: As we have seen in the last section, the multiplicative monoid contained in a ring does not need to be commutative like it is for the ring of integers. An example of a non-commutative ring is contained in Table 3.10 that we want to generalize here: The set $R^{n\times n}$ of $n \times n$ matrices over some ring R for some $b \in \mathbb{N}_{>0}$ together with pointwise addition and matrix multiplication is a ring. The additive identity is the matrix filled with all zeros whereas the multiplicative identity is the identity matrix filled with zeros but ones on the diagonal. The terms "zeros" and "ones" refer to the base ring of the matrices in this context. This ring of matrices is never commutative for $n > 1$. To see this consider the matrices with all entries being zero but the top-right or bottom-left entry, respectively, being one. The products of these matrices in both orders are different. For example with $n = 3$ we have

$$\begin{pmatrix}0&0&1\\0&0&0\\0&0&0\end{pmatrix}\begin{pmatrix}0&0&0\\0&0&0\\1&0&0\end{pmatrix} = \begin{pmatrix}1&0&0\\0&0&0\\0&0&0\end{pmatrix} \text{ but } \begin{pmatrix}0&0&0\\0&0&0\\1&0&0\end{pmatrix}\begin{pmatrix}0&0&1\\0&0&0\\0&0&0\end{pmatrix} = \begin{pmatrix}0&0&0\\0&0&0\\0&0&1\end{pmatrix}$$

Despite the examples given above we will only deal with commutative rings in this thesis and assume that all rings mentioned later on are commutative.

Another property of the ring of integers that may differ from general rings is the so-called characteristic of the ring. In all rings we have that the sum of an element with its additive inverse element is zero. There may be more pairs of elements having a sum of zero in general, even though there are none for the ring of integers. We will call rings like the ring of integers rings with characteristic zero.

Definition 3.11 Let $(R, +, 0, \cdot, 1)$ be a ring. The characteristic of this ring, denoted by $\operatorname{char}(R)$, is the smallest positive integer $c \in \mathbb{N}_{>0}$ such that

$$\underbrace{1 + 1 + \cdots + 1}_{c \text{ times}} = 0$$

If there is no such $c \in \mathbb{N}_{>0}$, the characteristic of $(R, +, 0, \cdot, 1)$ is defined to be zero.

For an example of a ring with positive characteristic consider the ring $(R, \oplus, 0, \otimes, 1)$ with the set $R = \{0, 1, 2, 3, 4, 5\}$ and $a \oplus b := (a + b) \bmod 6$ and $a \otimes b := (a \cdot b) \bmod 6$ for all $a, b \in R$ where $+$ and $\cdot$ denote the usual addition and multiplication of integers and $x \bmod y$ denotes the remainder of the integer division of x by y for all $x, y \in \mathbb{Z}$. This ring is usually called $\mathbb{Z}/6\mathbb{Z}$ and has characteristic $\operatorname{char}(\mathbb{Z}/6\mathbb{Z}) = 6$ since $1 \oplus 1 = 2 \neq 0$, $1 \oplus 1 \oplus 1 = 3 \neq 0$, $1 \oplus 1 \oplus 1 \oplus 1 = 4 \neq 0$, and $1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 = 5 \neq 0$, but $1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 = 0$. Positive characteristic introduces torsion to rings and exposes interesting phenomenons. We will just consider rings with characteristic zero in this thesis since those are the most natural ones.

We have seen above that we have $a \cdot 0 = 0 \cdot a = 0$ for all $a \in R$ in all rings R. If the product of two elements is zero it is not necessary that one of the elements is also zero in general, even though this holds for the ring of integers.

Definition 3.12 Let R be a ring. An element $a \in R$ is called left zero divisor of R if and only if there is an element $b \in R \setminus \{0\}$ with $a \cdot b = 0$. Similarly, $a \in R$ is called a right zero-divisor if and only if there is a $b \in R \setminus \{0\}$ with $b \cdot a = 0$. $a \in R$ is called a zero divisor if and only if it is a left zero divisor or a right zero divisor.

A ring R having no zero divisors other than zero itself is called a domain. A commutative domain is called integral domain.

We could also have defined that R is a domain if and only if for all $a, b \in R$ with $a \cdot b = 0$ we have $a = 0$ or $b = 0$. To paraphrase it, the product of two non-zero elements is always non-zero in domains. An example of a domain is the ring of integers. The ring $\mathbb{Z}/6\mathbb{Z}$ introduced above is no domain since in this ring we have $2 \cdot 3 = 0$. In general the ring $\mathbb{Z}/n\mathbb{Z}$ is a domain if and only if $n \in \mathbb{N}_{>0}$ is a prime. We will usually deal with domains in the following.

The definition of a homomorphism can also be extended to rings. We basically want to carry over the structure on the additive group and the multiplicative monoid.

Definition 3.13 Let $(R, \oplus, 0_R, \otimes, 1_R)$ and $(S, \boxplus, 0_S, \boxtimes, 1_S)$ be rings. A map $\phi : R \to S$ is a ring homomorphism if and only if $\phi : (R, \oplus, 0_R) \to (S, \boxplus, 0_S)$ is a group homomorphism and $\phi : (R, \otimes, 1_R) \to (S, \boxtimes, 1_S)$ is a monoid homomorphism. The kernel of the ring homomorphism ϕ is the set

$$\ker(\phi) := \{a \in R \,|\, \phi(a) = 0_S\}$$

Recalling Definition 3.7 this means that ϕ should have the following properties:

a) $\phi(a \oplus b) = \phi(a) \boxplus \phi(b)$ for all $a, b \in R$

b) $\phi(a \otimes b) = \phi(a) \boxtimes \phi(b)$ for all $a, b \in R$

c) $\phi(1_R) = 1_S$

We have shown in the last section that this already implies $\phi(0_R) = 0_S$ and $\phi(-a) = -\phi(a)$ for all $a \in R$.

It is natural to also carry over the definition of substructures of groups to rings. Interestingly, the existence of two operations allows for two different definitions of substructures of rings. The more straight-forward one is to take a subgroup of the additive group and a submonoid of the multiplicative monoid. We will call this substructure a subring.

Definition 3.14 Let $(R, +, 0, \cdot, 1)$ be a ring and S be a subset of R. $(S, +, 0, \cdot, 1)$ is called a subring of $(R, +, 0, \cdot, 1)$ if and only if $(S, +, 0)$ is a subgroup of $(R, +, 0)$ and $(S, \cdot, 1)$ is a submonoid of $(R, \cdot, 1)$.

Inserting Definition 3.6 this means the following:

a) $a + b \in S$ for all $a, b \in S$

b) $a \cdot b \in S$ for all $a, b \in S$

c) $1 \in S$

We saw above that this implies $0 \in S$ and that the neutral elements of R and S need to be the same for both operations. For example $(\mathbb{Z}, +, 0, \cdot, 1)$ is a subring of $(\mathbb{Q}, +, 0, \cdot, 1)$ which is itself a subring of $(\mathbb{R}, +, 0, \cdot, 1)$.

The main structure we will consider in this thesis is another substructure of a ring, which is called an ideal. We will deal with equations as ring elements and want to describe the set of equations implied by some given set of equations. We need to adjust our definition of the substructure of a ring from subrings to ideals to represent the set of implications of some equations correctly. In particular, we can find new implied equations by adding two equations that we know to hold or by multiplying two equations, but here it is enough that at least one input equation holds to imply the output equation. Details on this construction will be explained in Section 4.3.

In comparison to subrings as mentioned above, we will adjust property b) accordingly and drop property c) since this would correspond to the equation $1 = 0$ which can never hold (remember that we required $1 \neq 0$ for all rings).

Definition 3.15 Let $(R, +, 0, \cdot, 1)$ be a ring and I be a non-empty subset of R. I is called a right ideal of R if and only if $a + b \in I$ for all $a, b \in I$ and $a \cdot b \in I$ for all $a \in I, b \in R$. Similarly, we call I a left ideal of R if and only if $a + b \in I$ for all $a, b \in I$ and $a \cdot b \in I$ for all $a \in R, b \in I$. If and only if I is both a left ideal and a right ideal of R we call it an ideal of R and denote this by writing $I \trianglelefteq R$.

Let $a_1, \ldots, a_s \in R$ be some elements of the ring for some $s \in \mathbb{N}_{>0}$. The ideal generated by $a_1, \ldots, a_s$ is the smallest ideal that contains $a_1, \ldots, a_s$ and is denoted by $\langle a_1, \ldots, a_s \rangle$.

Ideals can also be defined as the kernels of ring homomorphisms: A subset $I \subseteq R$ of a ring R is an ideal if and only if there is a ring S and a ring homomorphism $\phi : R \to S$ such that $I = \ker(\phi)$. Similarly, one can define so-called normal subgroups as the kernels of group homomorphisms.

For commutative rings all left ideals are also right ideals and vice versa. Thus, the definitions of left ideals, right ideals, and ideals are equivalent for commutative rings. We will only consider commutative rings in the remainder of this thesis.

One property that holds for all ideals $I \trianglelefteq R$ is that $0 \in I$. To show this, we can take some element $a \in I$, because I is non-empty by definition. We can now multiply a with the element $0 \in R$ and get $a \cdot 0 = 0$ as seen above. On the other hand, this argument shows that the ideal generated by 0 contains no element other than 0.

Another interesting element regarding ideal membership is 1. If we have $1 \in I$ we can prove $a \in I$ for any $a \in R$ because $a = a \cdot 1$. Thus, each ring R has at least two ideals: $\langle 0 \rangle = \{0\}$ and $\langle 1 \rangle = R$.

Note that the smallest ideal that contains $a_1, \ldots, a_n$ as used above is well-defined. This is because one can easily verify that the intersection of two ideals is an ideal again: The intersection is not empty since zero is contained in all ideals and all other properties need to hold for the intersection of ideals since they hold in both intersected ideals. Thus, the smallest ideal containing $a_1, \ldots, a_s$ can be uniquely determined as the intersection of all ideals containing $a_1, \ldots, a_s$.

It is also interesting to note that the set of generators of an ideal is not unique in general. For some examples consider the ideals $n\mathbb{Z} := \{\dots, -3n, -2n, -n, 0, n, 2n, 3n, \dots\}$ of the ring $\mathbb{Z}$ for all $n \in \mathbb{N}_0$. All those ideals can be generated by one element only, namely $n\mathbb{Z} = \langle n \rangle = \langle -n \rangle$ for all $n \in \mathbb{N}_0$. Ideals like that, which can be generated by one element only, are called principal ideals. Rings with the property that all contained ideals are principal ideals are called principal ideal rings. $\mathbb{Z}$ is an example of a principal ideal ring.

Another example in the ring $\{f : \mathbb{Q} \to \mathbb{Q}\}$ as given in Table 3.10 is the ideal

$$\{f : \mathbb{Q} \to \mathbb{Q} \mid f(1) = 0\}$$

This ideal can be generated by one element too, for instance by

$$f : \mathbb{Q} \to \mathbb{Q}, x \mapsto \begin{cases} 0 & \text{if } x = 1 \\ 1 & \text{else} \end{cases}$$

When considering an ideal of equations that hold as suggested above, we will often say that other elements are equal up to elements in the ideal because those are considered to be essentially zero in our application. Formally speaking, we will work in the quotient ring of the ideal.

Definition 3.16 Let $(R, +, 0, \cdot, 1)$ be a ring and $I \trianglelefteq R$ be an ideal. We define an equivalence relation $\sim$ on R by $a \sim b \Leftrightarrow a - b \in I$ for all $a, b \in I$. The set of equivalence classes $R/\!\sim$ of $\sim$ is denoted by R/I. The equivalence class of an element $a \in R$ is denoted by $[a]_\sim := [a]_I := \{b \in R \mid a \sim b\}$. $(R/I, [0]_I, \oplus, [1]_I, \otimes)$ is itself a ring with

$$[a]_I \oplus [b]_I := [a + b]_I \text{ and } [a]_I \otimes [b]_I := [a \cdot b]_I$$

for all $a, b \in R$. The ring R/I is called the quotient ring of R modulo I.

For proofs showing that all constructions in the definition above are well-defined we refer to a textbook [Bos09]. We did already see an example above when we considered the ring $\mathbb{Z}/6\mathbb{Z}$. In our notation from above we have $R = \mathbb{Z}$ and $I = 6\mathbb{Z} = \{\dots, -12, -6, 0, 6, 12, \dots\}$ for this example.

For the example $\{f : \mathbb{Q} \to \mathbb{Q} \mid f(1) = 0\} \trianglelefteq \{f : \mathbb{Q} \to \mathbb{Q}\}$ that we also discussed above the equivalence relation is $f \sim g \Leftrightarrow f(1) = g(1)$ for all $f, g \in \{f : \mathbb{Q} \to \mathbb{Q}\}$. The product of two equivalence classes $[f]_I$ and $[g]_I$ is the set of all functions $h \in \{f : \mathbb{Q} \to \mathbb{Q}\}$ with $h(1) = f(1)g(1)$ for all $f, g \in \{f : \mathbb{Q} \to \mathbb{Q}\}$ and similarly for sums. An equivalence class is characterized by the value of the contained functions at 1. Therefore, the quotient ring is isomorphic to $\mathbb{Q}$. This example can also be stated in a more general way.

Example 3.17 Let R be a ring, let X be a set and let $a \in X$. We have

$$\{f : X \to R\}/\{f : X \to R \mid f(a) = 0\} \simeq R$$

Note that when speaking about generators of ideals it is not clear that there are always finite generating sets of all ideals. Nevertheless, all ideals that we consider in this thesis are finitely generated. This is true for all Noetherian rings. Those rings are named after the German mathematician Emmy Noether.

Definition 3.18 A ring R is said to satisfy the ascending chain condition on ideals if and only if for every infinite chain of ideals $I_1 \subseteq I_2 \subseteq I_3 \subseteq \dots$ for some ideals $I_1, I_2, I_3, \dots \trianglelefteq R$ there is an $s \in \mathbb{N}_{>0}$ such that $I_s = I_t$ for all $t \in \mathbb{N}_{>0}$ with $t \geq s$. Rings that satisfy the ascending chain condition on ideals are called Noetherian rings.

We can show that the ascending chain condition on ideals directly implies that all ideals are finitely generated.

Theorem 3.19 Let R be a Noetherian ring and let $I \trianglelefteq R$ be an ideal. There is a finite set of generators of I, i.e. there is $s \in \mathbb{N}_{>0}$ and $a_1, \dots, a_s \in R$ such that $I = \langle a_1, \dots, a_s \rangle$.

Proof The set $M := I$ itself is a generating set of I. We reduce M, i.e. while there is an element $a \in M$ with $\langle M \rangle = \langle M \setminus \{a\} \rangle$ we remove a from M. Note that the elements to remove may not be unique. In case of several possibilities we remove an arbitrary element with the given property.

If the remaining set M is finite we have shown that there is a finite generating set. Otherwise, we can enumerate an infinite set of different $a_1, a_2, a_3, \dots \in M$ from the remaining set. The ideals

$$\langle a_1 \rangle \subsetneq \langle a_1, a_2 \rangle \subsetneq \langle a_1, a_2, a_3 \rangle \subsetneq \dots$$

form an infinite ascending chain of ideals since M is reduced which contradicts the fact that R is Noetherian. $\square$

In fact, the property of Noetherian rings that all ideals are finitely generated is an equivalent definition of Noetherian rings. All rings that we will deal with in this thesis are Noetherian as the rings of integers, rationals, reals, and the complex numbers are Noetherian rings. Also, finite rings and all fields as defined in the next section are Noetherian. Additionally, we will see that all polynomial rings and rings of formal power series over Noetherian rings are Noetherian too.

3.3 Fields

When defining rings in Definition 3.9 we only required the multiplicative substructure to be a monoid instead of a group. This means that not every element of a ring needs to have a multiplicative inverse element. Elements having a multiplicative inverse element, however, are called units.

Definition 3.20 Let R be ring. An element $a \in R$ is called a unit if it has a multiplicative inverse element, i.e. there is $b \in R$ with $ab = ba = 1$. The set of all units of a ring R is denoted by $R^\times$.

The set of units is closed under multiplication, i.e. for all $a, b \in R^\times$ we have $ab \in R^\times$. This is because $b^{-1}a^{-1}$ is an inverse element of ab since $(ab)(b^{-1}a^{-1}) = (b^{-1}a^{-1})(ab) = 1$. We also always have $1 \in R^\times$ and $-1 \in R^\times$ because 1 and -1 are both multiplicative inverses of themselves: $1 \cdot 1 = (-1) \cdot (-1) = 1$. Those properties show that $R^\times$ is never empty and $(R^\times, \cdot, 1)$ is always a group. It is thus often called group of units.

The element 0 on the other hand is never contained in the group of units. This is, because for all $a \in R$ we have $a \cdot 0 = 0 \cdot a = 0$. The group of units is therefore always a subset of $R \setminus \{0\}$. If the group of units is as large as it can possibly be we call R a field.

Definition 3.21 Let R be a commutative ring. If $R^\times = R \setminus \{0\}$ we call R a field.

That means fields are commutative rings in which we can divide by all elements but 0. We will usually denote fields by k as a shorthand for $(k, +, 0, \cdot, 1)$.

Even though fields have many properties different from general rings, we can copy many definitions from rings.

Definition 3.22 Let k and k' be fields. A map $\phi : k \to k'$ is called a field homomorphism if and only if ϕ is a ring homomorphism. k is a subfield of k' if and only if k is a subring of k'.

Considering ideals of fields is not interesting since any ideal $I \trianglelefteq k$ of a field k that contains any non-zero element $a \in k \setminus \{0\}$ also contains $aa^{-1} = 1$ and therefore $I = k$. Thus, every field has just the two ideals $\langle 0 \rangle = \{0\}$ and $\langle 1 \rangle = k$.

Fields also have the nice property that they are domains and thus have no zero divisors.

Theorem 3.23 Let k be a field. k is a domain.

Proof Let $a, b \in k$ with $a \cdot b = 0$. We will show that $a = 0$ or $b = 0$ by assuming the opposite and showing that this implies a contradiction. Thus, let $a \neq 0$ and $b \neq 0$. Since k is a field there are inverse elements $a^{-1}, b^{-1} \in k$ for a and b. Using them, the following equation holds

$$1 = 1 \cdot 1 = (a^{-1} \cdot a) \cdot (b \cdot b^{-1}) = a^{-1} \cdot (a \cdot b) \cdot b^{-1} = a^{-1} \cdot 0 \cdot b^{-1} = 0$$

We know that $1 \neq 0$ in every ring, so this equation is a contradiction to our assumption. □

When we want to use properties of fields, but only have an integral domain, we can consider the field of fractions of the ring. That is the smallest field containing the ring.

Definition 3.24 Let R be an integral domain. The field of fractions of R is the field

$$\left(\mathrm{Quot}(R), +, \frac{0}{1}, \cdot, \frac{1}{1}\right)$$

where $\mathrm{Quot}(R)$ are the equivalence classes

$$\mathrm{Quot}(R) := \left\{\frac{a}{b} \mid a, b \in R\right\} / \sim$$

modulo the equivalence relation $\sim$ defined by

$$\frac{a}{b} \sim \frac{c}{d} \Leftrightarrow ad = bc$$

and the operations are defined by

$$\left[\frac{a}{b}\right]_\sim + \left[\frac{c}{d}\right]_\sim := \left[\frac{ad + bc}{bd}\right]_\sim \text{ and } \left[\frac{a}{b}\right]_\sim \cdot \left[\frac{c}{d}\right]_\sim := \left[\frac{ac}{bd}\right]_\sim$$

for all $a, b, c, d \in R$. $\mathrm{Quot}(R)$ is also sometimes denoted by $\mathrm{Frac}(R)$.

For proofs that $\sim$ is an equivalence relation, the operations are well-defined, and that the field of fractions is actually a field we refer to algebra textbooks [Bos09, vdWANB43, KM10]. To make notation easier, we usually just write $\frac{a}{b}$ to denote the equivalence class $\left[\frac{a}{b}\right]_\sim$ for all $a, b \in R$.

Rings that are no integral domains cannot be completed to fields because all fields are integral domains. The field of fractions can be thought of as a construction similar to the rational numbers. In fact, we have $\mathrm{Quot}(\mathbb{Z}) = \mathbb{Q}$. The elements $a \in R$ are represented by the equivalence class of $\frac{a}{1} \in \mathrm{Quot}(R)$.

There is much more theory on fields and in particular field extensions, but we will not introduce more here as these topics are not needed for the main results of this thesis.

4 Polynomial Rings

4.1 Formal Power Series and Polynomials

In this section we will introduce the type rings we are using the most often in this thesis, namely polynomial rings. To do so, we consider maps from some monoid which represents the exponents, usually $\mathbb{N}_0^n$ for some $n \in \mathbb{N}_{>0}$, to a commutative ring.

Definition 4.1 Let $(M, +)$ be a monoid and R be a commutative ring. A formal power series is a map $f : M \to R$. We will denote the value of the map f evaluated at $u \in M$ by f_u. A formal power series is called a polynomial if and only if it has a finite support, i.e. there are only finitely many $u \in M$ such that $f_u \neq 0$.

For all $u \in M$ we denote by $\underline{x}^u$ the polynomial

$$\underline{x}^u : M \to R, v \mapsto \begin{cases} 1 & \text{if } u = v \\ 0 & \text{else} \end{cases} \text{ for all } v \in M$$

Those elements are collected in the set $\mathbb{M}_M := \{\underline{x}^u \mid u \in M\}$ and are called monomials. We denote the set of formal power series by R^M or $R[[M]]$ and the set of polynomials by $R^{(M)}$ or $R[M]$. We embed the elements of the ring R in the set of polynomials with the map

$$\pi : R \to R[M] \subseteq R[[M]], c \mapsto \pi(c) \text{ where } \big(\pi(c)\big)_u := \begin{cases} c & \text{if } u = 0 \\ 0 & \text{else} \end{cases}$$

for all $u \in M, c \in R$.

The embedding $\pi : R \to R[M] \subseteq R[[M]]$ will often be omitted in notation and should be applied implicitly.

The construction of formal power series and polynomials may seem highhanded at first but is useful, because both sets can be equipped with a ring structure too. Let $f, g \in R[[M]]$. We define $f + g \in R[[M]]$ and $f \cdot g \in R[[M]]$ to be formal power series with

$$(f + g)_u := f_u + g_u$$
$$(f \cdot g)_u := \sum_{\substack{v,w \in M \\ v+w=u}} f_v g_w$$

for all $u \in M$. It is easy to see that if f and g are polynomials then $f+g$ and $f \cdot g$ are polynomials too. Using the associativity, commutativity, and distributivity of the operations $+$ and $\cdot$ on R

we can show that the new operations $+$ and $\cdot$ on $R[[M]]$ are associative, commutative, and distributive too.

For the identity elements we will consider the embedding $\pi(c) \in R[M]$ of elements $c \in R$. All these $\pi(c)$ have a support of size one and are thus polynomials. The additive and multiplicative identity elements for the operations on $R[[M]]$ and $R[M]$ are $\pi(0) = 0$ and $\pi(1) = \underline{x}^0$, respectively. Now we assembled a complete ring structure on $R[[M]]$ and $R[M]$ that will be the main structures to investigate in this thesis.

Theorem 4.2 Let $(M, +)$ be a monoid and R be a commutative ring. Together with the operations $+$ and $\cdot$ as well as the embedding π as defined above the formal power series

$$\left(R[[M]], +, \pi(0), \cdot, \pi(1)\right)$$

and polynomials

$$\left(R[M], +, \pi(0), \cdot, \pi(1)\right)$$

are commutative rings. $R[M]$ is referred to as a polynomial ring having the base ring R.

Note that we use the operator $+$ for the operation on M, the additive operation on R and the addition on $R[[M]]$ and $R[M]$. Which operation to use is implied by the set the summands belong to. Similarly, we use $\cdot$ for the multiplicative operation on R, $R[[M]]$, and $R[M]$.

The embedding π is also reflected in our notation for polynomials since we write

$$\pi(c) = c\underline{x}^0 = c$$

for all $c \in R$. This notation is justified since π is a ring homomorphism with

$$\pi(a) + \pi(b) = \pi(a + b) \text{ and } \pi(a) \cdot \pi(b) = \pi(a \cdot b)$$

for all $a, b \in R$. We therefore get the same result no matter whether we understand $a + b$ or $a \cdot b$ as addition or multiplication, respectively, in R, $R[[M]]$, or $R[M]$.

We can also investigate the map of elements of M to monoids in $R[M]$ with

$$\tau : M \to R[M] \subseteq R[[M]], v \mapsto \underline{x}^v$$

$\tau : (M, +, 0) \to (R[M], \cdot, 1)$ is a monoid homomorphism which has an additive notation in its codomain and a multiplicative notation in its domain.

The definition of monomials is useful since monomials can be used to generate all other formal power series and polynomials. To see this, let $f \in R[[M]]$ be a formal power series. With the operations above we can write

$$f = \sum_{u \in M} \pi(f_u) \cdot \underline{x}^u = \sum_{u \in M} f_u \underline{x}^u$$

Splitting polynomials into sums of monomials multiplied by a constant factor each reveals the vector space structure of polynomial rings as we will discuss in Section 4.4.

We will usually use $(M, +, 0) = (\mathbb{N}_0^n, +, 0)$ for some $n \in \mathbb{N}_{>0}$ where $+$ denotes the component-wise addition of integers. In that case we will write $x_i := \underline{x}^{e_i}$ where

$$e_i := \Big(\underbrace{0, \ldots, 0}_{i-1 \text{ times}}, 1, \underbrace{0, \ldots, 0}_{n-i \text{ times}}\Big) \in \mathbb{N}_0^n$$

is the vector that contains a one at the i-th position and zeros at all other positions and denote $R[[\mathbb{N}_0^n]]$ by $R[[\underline{x}]]$ or $R[[x_1, \ldots, x_n]]$ and $R[\mathbb{N}_0^n]$ by $R[\underline{x}]$ or $R\,[x_1, \ldots, x_n]$. We will always use that monoid in later sections if not stated otherwise. In this case we can write polynomials and formal power series $f \in R[[\underline{x}]]$ also as

$$f = \sum_{u \in M} f_u \left(\prod_{i=1}^{n} x_i^{u_i} \right)$$

We may also use other names for the indeterminates sometimes to avoid a clash of notation.

Polynomials with $M = \mathbb{N}_0^n$ for $n = 1$ are called univariate if and only if $n = 1$ or multivariate otherwise. The x_i are the so-called indeterminates (also called variables or unknowns in the literature). In the case of univariate polynomials we usually write $x := x_1$ as there is only one indeterminate. The summands $f(u)\underline{x}^u$ are called terms of f while $f(u)$ is called coefficient of f and u is the exponent of the term. Terms having the coefficient zero and indeterminates having the exponent zero are usually omitted in our notation. Note that this notation is only finite for polynomials, formal power series may have infinitely many terms with non-zero coefficients.

4.2 Properties of Polynomial Rings

After having defined polynomial rings we will discuss some of their properties in this section. We will focus on algorithms and complexity results in the main part of this thesis. For both topics it is essential that the input is finite, otherwise our algorithms are not able to even read all the input in finite time, and measuring running times and complexities would not be useful in the traditional sense. We will thus focus on polynomials from here, even though some results can also be stated for formal power series.

The structure of polynomial rings we have seen so far directly implies how to compute the characteristic of $R[M]$.

Theorem 4.3 Let $R[M]$ be a polynomial ring over a commutative ring R for some monoid M. We have $\mathrm{char}(R[M]) = \mathrm{char}(R)$.

Proof The theorem is immediately clear with

$$\pi(\underbrace{1+1+\cdots+1}_{c \text{ times}}) = \underbrace{\pi(1)+\pi(1)+\cdots+\pi(1)}_{c \text{ times}} = \underbrace{1+1+\cdots+1}_{c \text{ times}}$$

for all $c \in \mathbb{N}_{>0}$. □

We will now introduce the degree of polynomials which is among others a versatile tool for proofs that algorithms terminate.

Definition 4.4 Let R be a commutative ring and $(M,+,0)$ be a monoid. Additionally, let $\phi : M \to \mathbb{N}_0$ be a monoid homomorphism from $(M,+,0)$ to $(\mathbb{N}_0,+,0)$. The degree of a monomial $\underline{x}^u$ for some $u \in M$ is

$$\deg(\underline{x}^u) := \phi(u)$$

The degree of a polynomial $f \in R[M]$ is defined as

$$\deg(f) := \max\left(\{\phi(\underline{x}^u) \mid u \in M, f_u \neq 0\}\right)$$

Since $0 \in R[M]$ has no terms we define $\deg(0) := -\infty$

We defined the degree function deg for monomials and general polynomials. Note that the usage of the same function name is justified since for all $u \in M$ we have

$$\deg(\underline{x}^u) = \max\left(\{\phi(u) \mid u \in M, f_u \neq 0\}\right) = \phi(u)$$

We could also use a general monomial $(N,+,0)$ instead of $(\mathbb{N}_0,+,0)$, but we will not need a generalization of this definition. In the general case we would just need an ordering $\leq$ on this monomial which is compatible with the monoid operation, i.e. for all $a, b \in N$ we have either $a \leq b$ or $b \leq a$, we have $a \leq b$ and $b \leq a$ if and only if $a = b$, and we also have $a \leq a + b$ and $b \leq a + b$.

As given in Definition 4.4 we will always consider $(\mathbb{N}_0,+,0)$ as a monomial for grading. The ordering on this monomial is given by the natural ordering on the non-negative integers and fulfills all given conditions. When dealing with infinite values we set $-\infty < a < \infty$ for all $a \in \mathbb{N}_0$. As mentioned above if not stated otherwise we will use $(M,+,0) = (\mathbb{N}_0^n,+,0)$ for some $n \in \mathbb{N}_{>0}$. In this case we will use the map

$$\phi : \mathbb{N}_0^n \to \mathbb{N}_0, (a_1,\ldots,a_n) \mapsto \sum_{i=1}^n a_i$$

for all $a_1,\ldots,a_n \in \mathbb{N}_0$ for the definition of the degree which is indeed a monoid homomorphism. Many properties of the degree would also hold for other monoids or ϕ, but we will restrict ourselves to this case.

In Definition 4.4 we set $\deg(0) := -\infty$. This means that deg maps polynomials not only to integers but to the extended ring $(\mathbb{Z} \cup \{\infty, -\infty\}, +, 0, \cdot, -1)$ where the operations not including ∞ or $-\infty$ work as usual and

$$a + \infty := \infty + a := \infty, \quad a + (-\infty) := (-\infty) + a := -\infty$$

while

$$a \cdot \infty := \infty \cdot a := \begin{cases} \infty & \text{if } a > 0 \\ 0 & \text{if } a = 0 \\ -\infty & \text{if } a < 0 \end{cases}$$

and

$$a \cdot (-\infty) := (-\infty) \cdot a := \begin{cases} -\infty & \text{if } a > 0 \\ 0 & \text{if } a = 0 \\ \infty & \text{if } a < 0 \end{cases}$$

for all $a \in \mathbb{Z}$.

The map $\deg : R[M] \to \mathbb{Z} \cup \{\infty, -\infty\}$ is no ring homomorphism though. Nevertheless, the degree allows for some bounds on the degree of sums and products of polynomials.

Theorem 4.5 Let $f, g \in R[M]$ be polynomials with coefficients in some commutative ring $(R, +, 0, \cdot, 1)$ and exponents in some monoid $(M, +, 0)$ for some integer $n \in \mathbb{N}_{>0}$. It holds that

$$\deg(f + g) \le \max\Big(\deg(f), \deg(g)\Big)$$
$$\deg(fg) \le \deg(f) + \deg(g)$$

Proof Recall that the degree was defined in Definition 4.4 as

$$\deg(f) := \max(\{\phi(u) \,|\, u \in M, f_u \neq 0\})$$

Since the sum of two zero elements is zero again we know that

$$\{u \in M \,|\, (f+g)_u \neq 0\} \subseteq \{u \in M \,|\, f_u \neq 0\} \cup \{u \in M \,|\, g_u \neq 0\}$$

which implies

$$\begin{aligned} \deg(f+g) &= \max\Big(\{\phi(u) \,|\, u \in M, (f+g)_u \neq 0\}\Big) \\ &\le \max\Big(\{\phi(u) \,|\, u \in M, f_u \neq 0\} \cup \{\phi(u) \mid u \in M, g_u \neq 0\}\Big) \\ &= \max\Big(\max(\{\phi(u) \,|\, u \in M, f_u \neq 0\}),\ \max(\{\phi(u) \,|\, u \in M, g_u \neq 0\})\Big) \\ &= \max\Big(\deg(f), \deg(g)\Big) \end{aligned}$$

and the first claim.

For the second claim we know that a coefficient of the term $u \in M$ of fg can only be non-zero if there is a decomposition of u into $v, w \in M$ with $u = v + w$ and $f_v \neq 0$ as well as $g_w \neq 0$. This allows for a transformation using that the degree is a monoid homomorphism on the monomials

$$\begin{aligned}
\deg(fg) &= \max\left(\phi(u) \,|\, u \in M : (fg)_u \neq 0\}\right) \\
&\leq \max\left(\phi(u) \,|\, u \in M\ \exists v, w \in M : u = v + w, f_v \neq 0, g_w \neq 0\}\right) \\
&= \max\left(\phi(v + w) \,|\, v, w \in M, f_v \neq 0, g_w \neq 0\}\right) \\
&= \max\left(\phi(v) + \phi(w) \,|\, v, w \in M, f_v \neq 0, g_w \neq 0\}\right) \\
&\leq \max\left(\{\phi(v) \,|\, v \in M, f_v \neq 0\}\right) + \max\left(\{\phi(w) \,|\, v \in M, g_w \neq 0\}\right) \\
&= \deg(f) + \deg(g)
\end{aligned}$$

which concludes the proof. $\square$

It is interesting to note that the first inequality is not an equation in general. For instance for all $f \in R[M] \setminus \{0\}$ we have

$$-\infty = \deg(0) < \underbrace{\deg(f)}_{\geq 0} + \underbrace{\deg(-f)}_{\geq 0}$$

The second inequality is in fact an equation if R is a domain.

Another property of polynomial rings that we will often use, is that we can extend monoid homomorphisms from the exponent monoid to the base ring to ring homomorphisms from the polynomial ring to the base ring.

Theorem 4.6 (Substitution homomorphism) Let $(R, +, 0, \cdot, 1)$ and $(R', +, 0, \cdot, 1)$ be commutative rings, let $\mu : R \to R'$ be a ring homomorphism, let $(M, +, 0)$ be a monoid and let $\tau : (M, +, 0) \to (R', \cdot, 1)$ be a monoid homomorphism. The map

$$\varphi : R[M] \to R', f \mapsto \sum_{u \in M} \mu(f_u)\tau(u)$$

for all $f \in R[M]$ is a ring homomorphism.

Proof As listed after Definition 3.13 we have to show the following to establish that φ is a ring homomorphism:

a)

$$\begin{aligned}
\varphi(f+g) &= \sum_{u\in M} \mu\big((f+g)_u\big)\tau(u) \\
&= \sum_{u\in M} \mu(f_u + g_u)\tau(u) \\
&= \sum_{u\in M} \big(\mu(f_u) + \mu(g_u)\big)\tau(u) \\
&= \sum_{u\in M} \mu(f_u)\tau(u) + \sum_{u\in M} \mu(g_u)\tau(u) \\
&= \varphi(f) + \varphi(g)
\end{aligned}$$

for all $f, g \in R[M]$

b)

$$\begin{aligned}
\varphi(fg) &= \sum_{u\in M} \mu\big((fg)_u\big)\tau(u) \\
&= \sum_{u\in M} \mu\left(\sum_{\substack{v,w\in M\\ v+w=u}} f_v g_w\right)\tau(u) \\
&= \sum_{v\in M}\sum_{w\in M} \mu(f_v)\mu(g_w)\tau(v+w) \\
&= \sum_{v\in M}\sum_{w\in M} \mu(f_v)\tau(v)\mu(g_w)\tau(w) \\
&= \left(\sum_{v\in M} \mu(f_v)\tau(v)\right)\left(\sum_{w\in M} \mu(g_w)\tau(w)\right) \\
&= \varphi(f)\varphi(g)
\end{aligned}$$

for all $f, g \in R[M]$

c) $\varphi(1) = \mu(1)\tau(0) = 1 \cdot 1 = 1$

□

Note that all sums mentioned above are finite and therefore well-defined.

Example 4.7 We usually consider the so-called substitution homomorphisms. In this case we take a polynomial ring $R[x_1, \dots, x_n]$ over some commutative ring R, a tuple $a \in R^n$ and $R' := R$. We define the map μ to be the identity map, and let τ be the monoid homomorphism induced by $\tau(e_i) := a_i$ for all $i \in \{1, \dots, n\}$, namely $\tau(u) = a_1^{u_1} a_2^{u_2} \dots a_n^{u_n}$. The map

$$\varphi_a : R[x_1, \dots, x_n] \to R, f \mapsto \sum_{u \in \mathbb{N}_0^n} f_u a_1^{u_1} a_2^{u_2} \dots a_n^{u_n}$$

for all $f \in R[x_1, \dots, x_n]$ is called the substitution homomorphism φ_a and we often write $f(a) := \varphi_a(f)$. A tuple $a \in R^n$ with $f(a) = 0$ is called a root of f.

Note that this notation shall not be confused with the fact, that we formally defined f to be a map from $\mathbb{N}_0^n$ to R whereas the notation implies a map from R^n to R. The map φ_a can be thought of as replacing the x_i in the notation of f by the a_i for all $i \in \{1, \dots, n\}$.

We will often use the substitution homomorphism in the following sections. One particular application of it is the definition of algebraically closed fields.

Definition 4.8 Let k be a field. We say that k is an algebraically closed field if and only if for every polynomial $f \in k[x]$ with $\deg(f) > 0$ there is an $a \in k$ such that $f(a) = 0$.

The most prominent example of an algebraically closed field is the field of complex numbers $\mathbb{C}$. Note that the fields of rational numbers $\mathbb{Q}$ and real numbers $\mathbb{R}$ are not algebraically closed as for instance the polynomial $x_1^2 + 1$ has no root over those fields. In particular, over algebraically closed fields we can factor each univariate polynomial into a product where every factor has at most degree 1. For details on algebraically closed fields we refer to algebra textbooks [Bos09, vdWANB43, KM10].

4.3 Polynomial Ideals

In Section 4.1 we discussed polynomial rings and in Section 3.2 we saw that we can define ideals as substructures of rings. In this section we will combine both. Ideals of polynomial rings are called polynomial ideals. They are the main subject of interest in this thesis. We will therefore motivate their importance and find some of their properties in this section. For proofs that are not stated in this section we again refer to a computer algebra textbook [CLO07].

There are countless real-world problems that can be modeled mathematically by polynomial equations. An easy example is a typical physics textbook problem like this one:

> You throw a ball straight up into the air. Your arms are at x meters height and you throw the ball with an initial speed of v meters per second. How long does it take the ball to be at height y? All effects of friction can be ignored in this problem.

This problem can be modeled by the polynomial equation

$$y = x + vt - 0.5 \cdot gt^2$$

where t is the time elapsed in seconds and g is the gravitational acceleration at the surface of the Earth $g \approx 9.81$ in meters per square seconds. Examples for more complicated problems can be found in Section 1.1 and there are countless more real-world problems that can be modeled by polynomial equations.

In all these problems we want to find solutions of systems of equations or just some properties of the equations. We will discuss what solutions of systems of polynomial equations formally are in detail in Section 4.5. For this section, it means that we apply a substitution homomorphism on the equations that inserts our actual values for the symbols and want the results on both sides of the equation to be the same.

The first step for solving these problems is to reshape the equations such that the right-hand side of them is always zero. To do so, we take a polynomial equation $f = g$ for some $f, g \in R[x_1, \dots, x_n]$ and consider the equivalent equation $f - g = 0$ instead. We will speak about the equation $f \in R[x_1, \dots, x_n]$ when we actually mean the equation $f = 0$.

It is important to note that some given equations imply other equations, that are automatically true if the given equations are fulfilled. For instance, consider the polynomials $f = x_1 + 1, g = x_1 + x_2^2 \in R[x_1, x_2]$. If we have $f(a) = 0$ and $g(a) = 0$ we always also have $2a_1 + 2 = (2f)(a) = 0$ and $2a_1 + a_2^2 + 1 = (f + g)(a) = 0$ for $a \in R^2$. This holds in general: If we have $f(a) = 0$ and $g(a) = 0$ for some polynomials $f, g \in R[x_1, \dots, x_n]$, $n \in \mathbb{N}_{>0}$, $a \in R^n$, and $c \in R$ we also have $(cf)(a) = 0$ and $(f + g)(a) = 0$. Note that this is exactly the definition of an ideal contained in the polynomial ring $R[x_1, \dots, x_n]$ as seen in Definition 3.15.

Using other words this means that given some equations $f_1, \dots, f_s \in R[x_1, \dots, x_n]$ for some $s \in \mathbb{N}_{>0}$ the ideal $\langle f_1, \dots, f_s \rangle \trianglelefteq R[x_1, \dots, x_n]$ is the set of equations that is already implied by the given equations. If we are given $f_1, \dots, f_s$ the other equations in $\langle f_1, \dots, f_s \rangle$ are given implicitly too. That is why we usually consider the ideal generated by the given equations.

It is important to note here again that generators of ideals are generally not unique. For example the equations above $f = x_1 + 1, g = x_1 + x_2^2 \in R[x_1, x_2]$ imply the same ideal as the equations $f = x_1 + 1, h = x_2^2 - 1 \in R[x_1, x_2]$ since $g = f + h \in \langle f, h \rangle$ and $h = g - f \in \langle f, g \rangle$ and thus $\langle f, g \rangle = \langle f, h \rangle$.

This motivation also explains why subrings are not the right structure here. Using subrings we could only multiply equations that hold by other equations that hold instead of arbitrary ones and we would miss some equations that hold. Nevertheless, an ideal does not necessarily contain all equations implied by the ideal's generators. As an easy example consider the ideal $I = \left\langle x^2 \right\rangle \trianglelefteq \mathbb{Q}[x]$. The equation $x^2 = 0$ is only fulfilled if and only if $x = 0$, but we still have $x \notin I$. We will discuss how to include all equations that are fulfilled implicitly in the Section 4.5 and Section 8.1.

For another example we refer to the system of polynomials describing conformations of cyclohexane as explained in Section 1.1. After extending the determinants and reshaping the formulas we get the following polynomial ideal: $\langle 9x_1x_2x_3 + 9x_1x_3^2 + 9x_2x_3^2 + 15x_1x_2 - 51x_1x_3 - 51x_2x_3 - 66x_3^2 - 110x_1 - 110x_2 + 253x_3 + 605, 3x_1x_2^2 + 3x_2^2x_3 - 3x_1x_3^2 - 3x_2x_3^2 - 22x_1x_2 - 22x_2^2 + 22x_1x_3 + 22x_3^2 + 121x_2 - 121x_3, 3x_1^2x_2 + 3x_1^2x_3 - 3x_1x_3^2 - 3x_2x_3^2 - 22x_1^2 - 22x_1x_2 + 22x_2x_3 + 22x_3^2 + 121x_1 - 121x_3, 81x_1x_3^3 + 81x_2x_3^3 - 891x_1x_3^2 - 891x_2x_3^2 - 594x_3^3 - 720x_1x_2 + 1683x_1x_3 + 1683x_2x_3 + 5670x_3^2 + 4455x_1 + 4455x_2 - 6774x_3 - 31790, 81x_2^2x_3^2 - 594x_2^2x_3 - 594x_2x_3^2 + 225x_2^2 + 3492x_2x_3 + 225x_3^2 + 750x_2 + 750x_3 - 14575, 81x_1^2x_3^2 - 594x_1^2x_3 - 594x_1x_3^2 + 225x_1^2 + 3492x_1x_3 + 225x_3^2 + 750x_1 + 750x_3 - 14575\rangle \trianglelefteq \mathbb{Q}[x_1, x_2, x_3, x_4]$. The set of generators given here is even a Gröbner Basis as defined in Section 5.2.

We know that for all ideals its elements can be generated by adding elements contained in the ideal or multiplying an element of the ideal with another element. Substituting the operations that resulted in an element until we reach a generator of the ideal results in the following theorem.

Theorem 4.9 Let R be a commutative ring, $s \in \mathbb{N}_{>0}$, and $f_1, \ldots, f_s \in R$. The ideal generated by $f_1, \ldots, f_s$ is the set

$$\langle f_1, \ldots, f_s \rangle = \left\{ \sum_{i=1}^{s} f_i g_i \,|\, g_1, \ldots, g_s \in R \right\}$$

For polynomial ideals we can restrict this theorem to linear combinations with monomials instead of polynomials which makes some proofs easier.

Theorem 4.10 Let R be a commutative ring, $n, s \in \mathbb{N}_{>0}$, and $f_1, \ldots, f_s \in R[x_1, \ldots, x_n]$. The ideal generated by $f_1, \ldots, f_s$ is the set

$$\langle f_1, \ldots, f_s \rangle = \left\{ \sum_{j=1}^{r} c_j m_j f_{i_j} \,|\, r \in \mathbb{N}_{>0}, i_1, \ldots, i_r \in \{1, \ldots, n\}, \right.$$
$$\left. m_1, \ldots, m_r \in \mathbb{M}_{x_1,\ldots,x_n}, c_1, \ldots, c_r \in R \right\}$$

Proof The direction "$\supseteq$" is immediately clear from the definition of a polynomial ideal (Definition 3.15). For the other direction "$\subseteq$" let $f \in \langle f_1, \ldots, f_s \rangle \trianglelefteq R[x_1, \ldots, x_n]$. With Theorem 4.9 we know that there are $g_1, \ldots, g_s \in R[x_1, \ldots, x_n]$ with $f = \sum_{i=1}^{n} f_i g_i$. Because all polynomials have finite support, there are $k_1, \ldots, k_s \in \mathbb{N}_{>0}$, monomials $m_{i,1}, \ldots, m_{i,k_i} \in R[x_1, \ldots, x_n]$, and coefficients $c_{i,1}, \ldots, c_{i,k_i} \in R$ with $g_i = \sum_{j=1}^{k_i} c_{i,j} m_{i,j}$ for all $i \in \{1, \ldots, s\}$. Summing those equations up we get

$$f = \sum_{i=1}^{s} \sum_{j=1}^{k_i} c_{i,j} m_{i,j} f_i$$

which has the same form as given in the claim. □

This theorem implies that we can restrict ourselves to monomials as coefficients in the linear combinations if we allow that generators appear several times in the linear equation.

When we do computations with polynomial ideals they will usually be given by a finite set of generators, because computers only have a finite amount of memory. This can be justified by the Hilbert Basis Theorem named after the German mathematician David Hilbert who presented it in 1890 [Hil90].

Theorem 4.11 (Hilbert Basis Theorem [Hil90]) Let R be a Noetherian ring. The polynomial ring $R[x_1]$ is Noetherian.

We can apply this theorem multiple times to prove the same theorem on multivariate polynomial rings.

Corollary 4.12 Let R be a Noetherian ring and $n \in \mathbb{N}_{>0}$. The polynomial ring $R[x_1, \ldots, x_n]$ is Noetherian.

With Theorem 3.19 this implies that all polynomial ideals over polynomial rings with Noetherian coefficient fields are finitely generated. Thus, we can always assume that polynomial ideals are given by a finite generating set. We will use the Hilbert Basis Theorem without further reference in this thesis.

A relevant special case is that polynomial ideals generated by monomials can also be generated by a finite set of monomials. This theorem is known as Dickson's Lemma named after the the American mathematician Leonard E. Dickson [Dic13]. Polynomial ideals generated by monomials are discussed more in detail in Section 9.2.

4.4 Modules, Vector Spaces, and Algebras

In the main part of this thesis we will deal with polynomial equations with arbitrary degree and number of variables. Limiting the degree of the equations or the number of variables to one makes solving the corresponding systems of equations considerably easier. We will need this special case for improved algorithms for toric ideals as discussed in Chapter 10. In this section we will discuss linear equations, i.e. equations with degree at most 1, whereas we will discuss univariate polynomials, i.e. equations involving one variable only, in Section 5.1. For proofs that we omit during this section we refer to linear algebra textbooks [Fis13, Lan87].

Polynomial ideals of linear polynomials can be considered as subspaces of a vector space. We will define the notion of vector spaces first to state this formally. As vector spaces are an easy special case of modules we will define modules too, even though we will only deal with vector spaces in the remainder of this thesis.

Definition 4.13 Let R be a ring and let $(M, +, 0)$ be an Abelian group together with an operation $\cdot : R \times M \to M$. $(M, +, 0, \cdot)$ is called an R-module if and only if

a) $1 \cdot v = v$ for all $v \in M$ where 1 is the multiplicative neutral element of R

b) $a \cdot (b \cdot v) = (a \cdot b) \cdot v$ for all $a, b \in R$ and $v \in M$

c) $a \cdot (u + v) = a \cdot u + a \cdot v$ and $(a + b) \cdot v = a \cdot v + b \cdot v$ for all $a, b \in R$ and $u, v \in M$

In the case that R is a field we call the field $k := R$, the set $V := M$, and $(V, +, 0, \cdot)$ is called a k-vector space. The elements of a vector space are called vectors.

Note that we used the operator signs $+$ and $\cdot$ for the operations of the ring R and the field k. Which operation to use is always clear from the sets the operands belong to. We will also write V for the R-module $(V, +, 0, \cdot)$ or the k-vector space $(V, +, 0, \cdot)$ if the operations are clear from context.

The most common examples of modules are the $\mathbb{Z}$-modules $\mathbb{Z}^n$, $\mathbb{Q}^n$, $\mathbb{R}^n$, and $\mathbb{C}^n$, whereas common vector spaces are the $\mathbb{Q}$-vector spaces $\mathbb{Q}^n$, $\mathbb{R}^n$, and $\mathbb{C}^n$, the $\mathbb{R}$-vector spaces $\mathbb{R}^n$, and $\mathbb{C}^n$ and the $\mathbb{C}$-vector space $\mathbb{C}^n$ for $n \in \mathbb{N}_{>0}$ each with pointwise addition and scalar multiplication as operations. Also, all polynomial rings $R[x_1, \ldots, x_n]$ over a ring R for some $n \in \mathbb{N}_{>0}$ are R-modules. To see this we use the usual addition of polynomials for the map

$$+ : R[x_1, \ldots, x_n] \times R[x_1, \ldots, x_n] \to R[x_1, \ldots, x_n]$$

and the usual multiplication of polynomials for

$$\cdot : R \times R[x_1, \ldots, x_n] \to R[x_1, \ldots, x_n]$$

where we understand elements of R as polynomials of degree zero (or degree $-\infty$ for $0 \in R$). The axioms from Definition 4.13 can easily be verified from the properties of polynomial rings. We can also restrict ourselves to the set of polynomials of degree at most one and we still get an R-module

$$\{f \in R[x_1, \ldots, x_n] \mid \deg(f) \leq 1\}$$

with the same operations as above for the R-module $R[x_1, \ldots, x_n]$. We will see another way of embedding the structure of a module into the set of polynomials during the discussion of toric ideals in Chapter 10.

As we are most interested in polynomial ideals, we need to embed an equivalent structure to polynomial ideals into modules. We will see that submodules correspond to polynomial ideals in the linear case.

Definition 4.14 Let $(M, +, 0)$ be an R-module over a ring R and let $N \subseteq M$ such that $(N, +, 0)$ is a subgroup of $(M, +, 0)$ and $a \cdot v \in N$ for all $a \in R$, $v \in N$. The R-module $(N, +, 0)$ is called a submodule of $(M, +, 0)$. In the case that $k := R$ is a field we call $(M, +, 0)$ a linear subspace of the k-vector space $(N, +, 0)$.

Note that all submodules of an R-module M over a ring R are R-modules themselves and likewise all linear subspaces of k-vector spaces over a field k are vector spaces themselves.

Returning to the R-module $R[x_1, \ldots, x_n]$ for $n \in \mathbb{N}_{>0}$ we can show that each polynomial ideal $I \trianglelefteq R[x_1, \ldots, x_n]$ in the ring $R[x_1, \ldots, x_n]$ is also a submodule of the R-module $R[x_1, \ldots, x_n]$. This implication does not hold in the other direction since for instance

$$\{f \in R[x_1, \ldots, x_n] \mid \deg(f) \leq 1\}$$

is a submodule of the R-module $R[x_1, \ldots, x_n]$, but not an ideal of the ring $R[x_1, \ldots, x_n]$.

All vector spaces and their linear subspaces can be characterized by their bases. We will now switch from general modules to vector spaces since modules do not need to have bases in general while all vector spaces have bases.

Definition 4.15 Let k be a field, let V be a k-vector space, and let $B \subseteq V$ be a subset. B is called linearly independent if and only if for all $c_b \in V$ for each $b \in B$ with $\{c_b \mid b \in B\} \neq \{0\}$ we have

$$\sum_{b \in B} c_b \cdot b \neq 0$$

The set $\operatorname{span}(\{B\})$ spanned by B is the smallest linear subspace of V that contains B. A set $B \subseteq V$ is called a generating set of V if and only if $\operatorname{span}(\{B\}) = V$. B is called a basis of V if and only if B is linearly independent and a generating set of V.

As mentioned above modules may not have bases. For instance consider the $\mathbb{Z}$-modules $\mathbb{Z}/n\mathbb{Z}$ for $n \in \mathbb{N}_{>0}$ as defined in Definition 3.16 with the usual addition and multiplication modulo n. Those modules do not have bases as all non-empty subsets of $B \subseteq \mathbb{Z}/n\mathbb{Z}$ are linearly dependent which can be shown by choosing the coefficients $c_b = n$ for all $b \in B$ which implies $\sum_{b \in B} c_b \cdot b = 0$ not matter what B is.

All vector spaces, however, have a basis and their size is unique. It was proven that this theorem is equivalent to the axiom of choice by James D. Halpern in 1966 [Hal66].

Theorem 4.16 (Dimension Theorem for Vector Spaces) Let k be a field and V be a k-vector space. V has at least one basis. All bases of V have the same cardinality. Their cardinality is called the dimension of the vector space V and denoted by $\dim(V)$.

For example the k-vector space $k[x_1, \ldots, x_n]$ over a field k for some $n \in \mathbb{N}_{>0}$ has a basis consisting of all monomials contained in $k[x_1, \ldots, x_n]$. There are infinitely many monomials in $k[x_1, \ldots, x_n]$, therefore the vector space is infinite-dimensional. The k-vector space

$$\{f \in k[x_1, \ldots, x_n] \mid \deg(f) \leq 1\}$$

also has a basis consisting of all monomials contained in it, but in this case the basis is $\{1, x_1, x_2, \ldots, x_n\}$. Thus, the dimension

$$\dim(\{f \in k[x_1, \ldots, x_n] \mid \deg(f) \leq 1\})$$

is $n + 1$.

We will discuss solving systems of equations in the k-vector space

$$\{f \in k[x_1, \ldots, x_n] \mid \deg(f) \leq 1\}$$

in Section 5.1 and will now introduce the Elimination Theorem needed there. Since the product of two linear polynomials is not linear in general it does not make sense to speak about polynomial ideals in this case. We drop the property of polynomial ideals I that for all $f \in I$ and $g \in k[x_1, \ldots, x_n]$ we have $fg \in I$ and replace it by a similar property that does not increase the degree of the polynomials: for all $f \in I$ and $g \in k$ we have $fg \in I$. Leaving all other properties the same we just changed the definition from an ideal to a linear subspace. Thus, in the case of linear equations our system of equations is modeled by a linear subspace of

$$\{f \in k[x_1, \ldots, x_n] \mid \deg(f) \leq 1\}$$

instead of a polynomial ideal of $k[x_1, \ldots, x_n]$.

The first step in solving such a system of equations is eliminating an indeterminate. Suppose we are given $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ for some $n, s \in \mathbb{N}_{>0}$ with $\deg(f_i) \leq 1$ for all $i \in \{1, \ldots, s\}$. The linear subspace spanned by those polynomials is

$$\operatorname{span}(\{f_1, \ldots, f_s\}) \subseteq \{f \in k[x_1, \ldots, x_n] \mid \deg(f) \leq 1\}$$

Eliminating indeterminate x_n means computing

$$\operatorname{span}(\{f_1,\dots,f_s\}) \cap \{f \in k[x_1,\dots,x_{n-1}] \mid \deg(f) \le 1\}$$

which is a linear subspace of

$$\{f \in k[x_1,\dots,x_n] \mid \deg(f) \le 1\}$$

again. The following theorem tells how to find a generating set of that space.

Theorem 4.17 Let $f_1,\dots,f_s \in k[x_1,\dots,x_n]$ be polynomials over a field k for some $n, s \in \mathbb{N}_{>0}$ with $\deg(f_i) \le 1$ for all $i \in \{1,\dots,s\}$. Let $a_i \in k$ be the coefficient of the term x_n in f_i for all $i \in \{1,\dots,s\}$. Without loss of generality we assume that there is a $t \in \mathbb{N}_0$ such that $a_i = 0$ if and only if $i \le t$. Furthermore, let

$$M := \{f_i \mid i \in \{1,\dots,s\}, i \le t\} \cup \{a_{t+1}f_i - a_i f_{t+1} \mid i \in \{1,\dots,s\}, i \ge t+2\}$$

We have

$$\operatorname{span}(\{f_1,\dots,f_s\}) \cap \{f \in k[x_1,\dots,x_{n-1}] \mid \deg(f) \le 1\} = \operatorname{span}(M)$$

Proof For all polynomials $f \in M$ we obviously have

$$f \in \operatorname{span}(\{f_1,\dots,f_s\}) \cap \{f \in k[x_1,\dots,x_{n-1}] \mid \deg(f) \le 1\}$$

which implies the direction "$\supseteq$" of the proof. For the other direction let

$$f \in \operatorname{span}(\{f_1,\dots,f_s\}) \cap \{f \in k[x_1,\dots,x_{n-1}] \mid \deg(f) \le 1\}$$

The containment $f \in \operatorname{span}(\{f_1,\dots,f_s\})$ means there are $c_1,\dots,c_s \in k$ with $f = \sum_{i=1}^{s} c_i f_i$ and

$$f \in \{f \in k[x_1,\dots,x_{n-1}] \mid \deg(f) \le 1\}$$

implies $\sum_{i=1}^{s} c_i a_i = 0$. Putting this together we get

$$\begin{aligned} f &= \sum_{i=1}^{s} c_i f_i \\ &= \sum_{i=1}^{s} c_i f_i - \frac{1}{a_{t+1}}\left(\sum_{i=1}^{s} c_i a_i\right) f_{t+1} \\ &= \sum_{i=1}^{s} c_i f_i - \frac{1}{a_{t+1}}\left(\sum_{i=t+1}^{s} c_i a_i\right) f_{t+1} \\ &= \sum_{i=1}^{t} c_i f_i + \sum_{i=t+2}^{s} \frac{c_i}{a_{t+1}} (a_{t+1} f_i - a_i f_{t+1}) \qquad \in \operatorname{span}(M) \end{aligned}$$

□

The polynomial f_{t+1} is called pivot of this reduction. The pivot together with the elements of M generate the original linear subspace:

$$\operatorname{span}\left(\{f_{t+1}\} \cup M\right) = \operatorname{span}\left(\{f_1, \dots, f_s\}\right)$$

We can eliminate one indeterminate after another from a linear subspace using Theorem 4.17. If we collect the pivots from each elimination of an indeterminate we get a special set of generators of the linear subspace, the so-called row-echelon form of $\operatorname{span}\left(\{f_1, \dots, f_s\}\right)$. The row-echelon form can be used to find solutions of the system of linear equations. We discuss this algorithm in Section 7.2. These findings can also be used for general vector spaces.

Another structure that we will use in Chapter 10 are algebras over commutative rings. Algebras are modules that allow for an additional multiplicative operation.

Definition 4.18 Let R be a commutative ring, $(M, +, 0, \cdot)$ be an R-module and let

$$\odot : M \times M \to M$$

be an operation. $(M, +, 0, \cdot, \odot)$ is an R-algebra if and only if $\odot$ is bilinear, i.e.

a) $u \odot (v + w) = (u \odot v) + (u \odot w)$ and $(u + v) \odot w = (u \odot w) + (v \odot w)$ for all $u, v, w \in M$

b) $a \cdot (u \odot v) = (a \cdot u) \odot v = u \odot (a \cdot v)$ for all $a \in R$ and $u, v \in M$.

Again, we also call the set M an R-algebra if the operations are clear from context and we use the same notion for all multiplicative operations if the operation to be used is clear from the operands. The most important examples of R-algebras over a commutative ring R, that we use in this thesis, are the polynomial rings $R[x_1, \dots, x_n]$ for some $n \in \mathbb{N}_{>0}$ where

$$\odot : R[x_1, \dots, x_n] \times R[x_1, \dots, x_n] \to R[x_1, \dots, x_n]$$

is the usual multiplication of polynomials.

For modules, vector spaces, and algebras we can also define homomorphisms between those structures as we did for monoids, groups, rings, and fields. Those homomorphisms need to be compatible with the respective operations on the structure.

Definition 4.19 Let R be a commutative ring and k be a field. A map $\phi : M \to N$ between two R-modules $(M, \oplus, 0_M, \cdot)$ and $(N, \boxplus, 0_N, \times)$ is a module homomorphism if and only if

$$\phi(u \oplus v) = \phi(u) \boxplus \phi(v) \text{ and } \phi(a \cdot u) = a \times \phi(u)$$

for all $a \in R$ and $u, v \in M$. A map between two k-vector spaces is a vector space homomorphism or linear map if and only if it is a module homomorphism. A map $\phi : M \to N$ between two R-algebras $(M, \oplus, 0_M, \cdot, \otimes)$ and $(N, \boxplus, 0_n, \times, \boxtimes)$ is an algebra homomorphism or linear map if and only if ϕ is a module homomorphism and additionally

$$\phi(u \otimes v) = \phi(u) \boxtimes \phi(v)$$

for all $u, v \in M$.

There are many more interesting definitions and theorems involving modules, vector spaces, and algebras that we do not need for this thesis. For instance, we can define associated algebras by changing the base ring of an algebra. We refer to algebra textbooks for details on further constructions [Fis13, Lan87, Bos09, vdWANB43].

4.5 Varieties and the Zariski topology

We did already discuss finding solutions of systems of polynomial equations in the last section without formally defining what a solution is. We will discuss the structure of those sets in this section.

Definition 4.20 Let $I \trianglelefteq R[x_1, \ldots, x_n]$ be a polynomial ideal over a commutative ring R for some $n \in \mathbb{N}_{>0}$. The variety of I is the set

$$\mathcal{V}(I) := \{(a_1, \ldots, a_n) \in R^n \mid f(a_1, \ldots, a_n) = 0 \text{ for all } f \in I\} \subseteq R^n$$

The notation $f(a_1, \ldots, a_n)$ refers to the substitution homomorphism as defined in Example 4.7. $\mathcal{V}(I)$ is the set of all points of R^n that is a solution of every equation contained in the polynomial ideal I. Equivalently, $\mathcal{V}(I)$ is the set of all points of R^n that is a solution of every equation contained in some set of generators of the polynomial ideal I. When speaking about solving a given set of equations $f_1, \ldots, f_s \in R[x_1, \ldots, x_n]$ for some $s \in \mathbb{N}_{>0}$ we usually mean finding $\mathcal{V}(\langle f_1, \ldots, f_s \rangle)$.

Vice versa, we could also define a polynomial ideal that is the largest ideal that has a given set of points as solution.

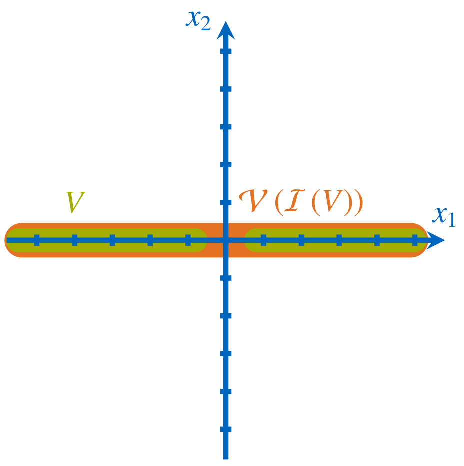


Figure 4.23: A visualization of the set $V \subseteq \mathbb{Q}^2$ from Example 4.22.

Definition 4.21 Let R be a commutative ring, $n \in \mathbb{N}_{>0}$, and let $V \subseteq R^n$ be a set of points. The vanishing ideal of V is the polynomial ideal

$$\mathcal{I}(V) := \{f \in R[x_1,\dots,x_n] \mid f(a_1,\dots,a_n) = 0 \text{ for all } (a_1,\dots,a_n) \in V\} \trianglelefteq R[x_1,\dots,x_n]$$

It is easy to verify that $\mathcal{I}(V)$ is indeed a polynomial ideal for all $V \subseteq R^n$. One could guess that the maps $\mathcal{V}$ and $\mathcal{I}$ are inverses of each other, but we have neither $I = \mathcal{I}(\mathcal{V}(I))$ nor $V = \mathcal{V}(\mathcal{I}(V))$ for polynomial ideals $I \trianglelefteq R[x_1,\dots,x_n]$ and sets $V \subseteq R^n$ in general. $I = \mathcal{I}(\mathcal{V}(I))$ only holds for radical ideals that are defined and discussed in Section 8.1. $V = \mathcal{V}(\mathcal{I}(V))$ only holds for sets V that are closed under the so-called Zariski topology, that we will define below.

Example 4.22 For an example we consider the polynomial ring $\mathbb{Q}[x_1,x_2]$ and the set

$$V := \{(a,0) \mid a \in \mathbb{Q}, a \neq 0\} \subseteq \mathbb{Q}^2$$

The set V is the x_1-axis excluding the point $(0,0)$. Since V contains infinitely many points we see that all polynomials $f \in \mathcal{I}(V) \trianglelefteq \mathbb{Q}[x_1,x_2]$ must be multiples of x_2. The polynomial ideal of all those polynomials is generated by x_2, thus $\mathcal{I}(V) = \langle x_2 \rangle$. On the other hand, the solutions of the polynomial x_2 are all points with the second component being zero which implies

$$\mathcal{V}(\mathcal{I}(V)) = \{(a,0) \mid a \in \mathbb{Q}\} \supsetneq V$$

It is interesting to note that it was indeed needed that V in this example was an infinite set, for all finite sets $V \subseteq \mathbb{Q}^2$ we get $\mathcal{V}(\mathcal{I}(V)) = V$.

For all $V \subseteq R^n$ we have $V \subseteq \mathcal{V}(\mathcal{I}(V))$, but the other direction does not need to be true. This holds because

$$\begin{aligned}\mathcal{V}(\mathcal{I}(V)) &= \left\{a \in R^n \mid f(a) = 0 \text{ for all } f \in \{g \in R[x_1, \ldots, x_n] \mid g(b) = 0 \text{ for all } b \in V\}\right\} \\ &= \left\{a \in R^n \mid f(a) = 0 \text{ for all } f \in R[x_1, \ldots, x_n] \text{ with } f(b) = 0 \text{ for all } b \in V\right\} \quad \supseteq V\end{aligned}$$

which clearly contains all points in V.

The sets $V \subseteq R^n$ with $V = \mathcal{V}(\mathcal{I}(V))$ form a so-called topology.

Definition 4.24 Let X be a set and $T \subseteq \mathcal{P}(X)$ be a set of subsets of X. T is called a topology on X if $\emptyset \in T$, $X \in T$, unions of sets in T are also contained in T and intersections of finitely many sets in T are also contained in T. Subsets of X contained in T are called open sets under T. Subsets Y of X, whose complement $X \setminus Y$ is contained in X, are called closed sets under T.

Topologies are probably best known in connection with manifolds and knot theory. This field studies the properties of spaces under continuous transformations. One famous result states that compact, connected surfaces can be continuously transformed into another if and only if they have the same orientability, number of boundary components, and genus. This is for instance the case for a cup and a torus. We will not use topologies in this thesis other than the Zariski topology.

Theorem 4.25 Let $R[x_1, \ldots, x_n]$ be the polynomial ring over a commutative ring R for some $n \in \mathbb{N}_{>0}$. The set

$$\left\{V \subseteq R^n \mid R^n \setminus V = \mathcal{V}\Big(\mathcal{I}(R^n \setminus V)\Big)\right\} \subseteq \mathcal{P}(R^n)$$

is a topology over R^n. It is called the Zariski topology in honor of the mathematician Oscar Zariski.

We refer to a textbook for the proof that the Zariski topology is in fact a topology [CLO07].

The sets $V \subseteq R^n$ with $V = \mathcal{V}(\mathcal{I}(V))$ are the sets closed under the Zariski topology. Those sets are also the image of the map $\mathcal{V}$, i.e. each set $V \subseteq R^n$, that is the variety of any polynomial ideal, has the property $V = \mathcal{V}(\mathcal{I}(V))$. Thus, for sets $V \subseteq R^n$ with $V \subsetneq \mathcal{V}(\mathcal{I}(V))$ we know that V is not the variety of any polynomial ideal $I \trianglelefteq R[x_1, \ldots, x_n]$. The set $\mathcal{V}(\mathcal{I}(V))$ is the smallest super set of V that is a variety. It is called the closure of V under the Zariski topology. In Example 4.22 we saw that the closure of $\{(a, 0) \mid a \in \mathbb{Q}, a \neq 0\}$ under the Zariski topology is $\{(a, 0) \mid a \in \mathbb{Q}\}$.

4.6 Operations on Polynomial Ideals

While working with polynomial ideals, we will use several operations that will be introduced in this section. For a survey of operations on polynomial ideals and detailed proofs of all statements in this section we refer to the Bachelor's thesis of this author [Tom12]. We just give a summary of the most important results here.

The easiest operation on a polynomial ideal is to add new elements to the ideal. The union of polynomial ideals is no polynomial ideal in general, though. For an example consider the polynomial ideals $\langle x_1 \rangle, \langle x_2 \rangle \trianglelefteq \mathbb{Q}[x_1, x_2]$. We have $x_1, x_2 \in \langle x_1 \rangle \cup \langle x_2 \rangle$ but $x_1 + x_2 \notin \langle x_1 \rangle \cup \langle x_2 \rangle$. Thus, we have to compute the ideal closure of the union.

Definition 4.26 Let $I, J \trianglelefteq R[x_1, \dots, x_n]$ be polynomial ideals over a ring R for some $n \in \mathbb{N}_{>0}$. The sum of I and J is the set $I + J := \{f + g \mid f \in I, g \in J\}$.

$I + J$ is a polynomial ideal itself and the smallest ideal containing $I \cup J$. The sum of I and J can be generated by the union of generators of I and J. As adding more polynomials to a polynomial ideal shrinks the variety of the ideal we can compute the variety of the sum of polynomial ideals as the intersection of the varieties of the original polynomial ideals.

Theorem 4.27 Let $f_1, \dots, f_s, g_1, \dots, g_r \trianglelefteq R[x_1, \dots, x_n]$ be polynomials over a ring R and let $I, J \trianglelefteq R[x_1, \dots, x_n]$ be polynomial ideals for some $n, r, s \in \mathbb{N}_{>0}$. We have

$$\langle f_1, \dots, f_s \rangle + \langle g_1, \dots, g_r \rangle = \langle f_1, \dots, f_s, g_1, \dots, g_r \rangle$$

and

$$\mathcal{V}(I + J) = \mathcal{V}(I) \cap \mathcal{V}(J)$$

Another operation, that we will often need, is the intersection of polynomial ideals. In contrast to the union of polynomial ideals the intersection of polynomial ideals contained in the same polynomial ring is always a polynomial ideal again. To compute intersections of polynomial ideals we need to define the special case of elimination ideals first.

Definition 4.28 Let $I \trianglelefteq R[x_1, \dots, x_n]$ be a polynomial ideal over a ring R for some $n \in \mathbb{N}_{>0}$. The i-th elimination ideal of I is the set

$$I \cap R[x_1, \dots, x_i] \trianglelefteq R[x_1, \dots, x_i]$$

for every $i \in \{1, \dots, n\}$.

All elimination ideals of polynomial ideals are polynomial ideals again. Note that

$$R[x_1, \dots, x_i] \subseteq R[x_1, \dots, x_n] \text{ and } I \cap R[x_1, \dots, x_i] \subseteq R[x_1, \dots, x_n]$$

are no polynomial ideals in general for $i \in \{1, \dots, n-1\}$. Elimination ideals are generally only polynomial ideals in the smaller polynomial ring $R[x_1, \dots, x_i]$. Nevertheless, we can extend polynomial ideals $J \trianglelefteq R[x_1, \dots, x_i]$ to polynomial ideals in $R[x_1, \dots, x_n]$ by taking the smallest ideal in $R[x_1, \dots, x_n]$ containing J. This extended ideal can be generated by the images of mapping a set of generators of the ideal $J \trianglelefteq R[x_1, \dots, x_i]$ to $R[x_1, \dots, x_n]$.

Computing a generating set of an elimination ideal is a computationally hard task in general. We will discuss an algorithm to compute elimination ideal in Section 5.3 and analyze its complexity in Section 7.2. For the remainder of this section we will use the computation of elimination ideals as a black box algorithm.

The computation of the general case of intersections of polynomial ideals can be reduced to elimination ideals. To do so, we will introduce a new indeterminate to our polynomial ring.

Theorem 4.29 Let $f_1, \dots, f_s, g_1, \dots, g_r \trianglelefteq R[x_1, \dots, x_n]$ be polynomials over a ring R and let $I, J \trianglelefteq R[x_1, \dots, x_n]$ be polynomial ideals for some $n, r, s \in \mathbb{N}_{>0}$. For

$$K := \langle x_{n+1} f_1, \dots, x_{n+1} f_s, (1 - x_{n+1}) g_1, \dots, (1 - x_{n+1}) g_r \rangle \trianglelefteq R[x_1, \dots, x_{n+1}]$$

we have

$$\langle f_1, \dots, f_s \rangle \cap \langle g_1, \dots, g_r \rangle = K \cap R[x_1, \dots, x_n]$$

and

$$\mathcal{V}(I \cap J) = \mathcal{V}(I) \cup \mathcal{V}(J)$$

The variety of the intersection of polynomial ideals is the union of their varieties. This is, because to get the intersection we remove all polynomials from one polynomial ideal that are not contained in the other one and thus grow the variety to also include the points from the other variety. Interestingly, there is also another operation on polynomial ideals where the variety of the result is the union of the varieties of the original ideals.

Definition 4.30 Let $I, J \trianglelefteq R[x_1, \dots, x_n]$ be polynomial ideals over a ring R for some $n \in \mathbb{N}_{>0}$. The product of I and J is the set

$$I \cdot J := \left\{ \sum_{i=1}^{s} f_i g_i \mid f_1, \dots, f_s \in I, g_1, \dots, g_s \in J, s \in \mathbb{N}_{>0} \right\}$$

Note that we have to close the product under sums as it would not be a polynomial ideal otherwise in general. Generators of the product of polynomial ideals are for instance the pairwise products of the generators of the original ideals.

Theorem 4.31 Let $f_1, \dots, f_s, g_1, \dots, g_r \trianglelefteq R[x_1, \dots, x_n]$ be polynomials over a ring R and let $I, J \trianglelefteq R[x_1, \dots, x_n]$ be polynomial ideals for some $n, r, s \in \mathbb{N}_{>0}$. We have

$$\langle f_1, \dots, f_s \rangle \cdot \langle g_1, \dots, g_r \rangle = \left\langle f_i g_j \,|\, i \in \{1, \dots, s\}, j \in \{1, \dots, r\} \right\rangle$$

and

$$\mathcal{V}(I \cdot J) = \mathcal{V}(I) \cup \mathcal{V}(J)$$

Since the product and intersection of polynomial ideals have the same variety we can exchange both operations if we only need the variety of the result to be correct. This is for instance the case for the radical word problem as defined in Section 7.1. This is useful, because the product is much faster to compute than the intersection of polynomial ideals.

To motivate the next operation consider the ideal $I := \langle x_1(x_2 - x_3), f_1, \dots, f_s \rangle \trianglelefteq \mathbb{Q}[x_1, x_2, x_3]$ for some $f_1, \dots, f_s \in \mathbb{Q}[x_1, x_2, x_3]$ and $s \in \mathbb{N}_{>0}$. Given a polynomial contained in I we can replace x_2 by x_3 and vice versa to get another polynomial contained in I because $x_1(x_2 - x_3) \in I$, but only of there is an x_1 contained in the same term as the x_2 or x_3 to replace. The x_1 behaves like a catalyst for the replacement of x_2 by x_3 or vice versa: It has to be present to execute the replacement, but it is not touched during the replacement. Removing those catalysts is called computing the quotient of a polynomial ideal.

Definition 4.32 Let $I, J \trianglelefteq R[x_1, \dots, x_n]$ be polynomial ideals over a ring R for some $n \in \mathbb{N}_{>0}$. The quotient of I by J is the set

$$I : J := \{f \in R[x_1, \dots, x_n] \mid fg \in I \text{ for all } g \in J\}$$

For a polynomial $g \in R[x_1, \dots, x_n]$ we denote the quotient $I : \langle g \rangle$ by $I : g$.

The quotient of polynomial ideals as defined above is always a polynomial ideal again. The computation of a generating set of the quotient is a computationally hard task as we will use the computation of intersections of polynomial ideals as a subroutine. The first step of the computation is to split the quotient into several subproblems for every generator of the divisor.

Theorem 4.33 Let $I, J, K \trianglelefteq R[x_1, \dots, x_n]$ be polynomial ideals over a commutative ring R for some $n \in \mathbb{N}_{>0}$. We have

$$I : (J + K) = (I : J) \cap (I : K)$$

In particular, this means for a polynomial ideal $I \trianglelefteq R[x_1, \ldots, x_n]$ over a ring R and polynomials $f_1, \ldots, f_s \in R[x_1, \ldots, x_n]$ for some $n, s \in \mathbb{N}_{>0}$ that

$$I : \langle f_1, \ldots, f_s \rangle = I : \left(\sum_{i=1}^{s} \langle f_i \rangle \right) = \bigcap_{i=1}^{s} I : f_i$$

Thus, we only have to compute the quotient with principal ideals as divisor. This can also be done using an algorithm to compute the intersection of polynomial ideals.

Theorem 4.34 Let $I \trianglelefteq R[x_1, \ldots, x_n]$ be a polynomial ideal over a commutative ring R for some $n \in \mathbb{N}_{>0}$ and let $g \in R[x_1, \ldots, x_n]$ be a polynomial. Let $f_1, \ldots, f_s \in R[x_1, \ldots, x_n]$ for some $s \in \mathbb{N}_{>0}$ such that $\langle f_1, \ldots, f_s \rangle = I \cap \langle g \rangle$. In this case, we have

$$I : f = \left\langle \frac{f_1}{g}, \ldots, \frac{f_s}{g} \right\rangle$$

Note that the fractions $\frac{f_i}{g}$ for $i \in \{1, \ldots, s\}$ are polynomials since f_i is a multiple of g. Putting those results together, this results in an algorithm to compute the quotient of two polynomial ideals.

Algorithm 4.35 Compute the quotient of one polynomial ideal by another.

Input: $f_1, \ldots, f_s, g_1, \ldots, g_r \in R[x_1, \ldots, x_n]$ polynomials over a commutative ring R for some $n, r, s \in \mathbb{N}_{>0}$

Output: a generating set of $\langle f_1, \ldots, f_s \rangle : \langle g_1, \ldots, g_r \rangle$

```
1: set $I := \langle 1 \rangle \trianglelefteq R[x_1, \ldots, x_n]$
2: for $i \in \{1, \ldots, r\}$ do
3:     compute polynomials $h_1, \ldots, h_t \in R[x_1, \ldots, x_n]$ for some $t \in \mathbb{N}_{>0}$ such that
       $\langle h_1, \ldots, h_t \rangle = \langle f_1, \ldots, f_s \rangle \cap \langle g_i \rangle$ using Theorem 4.29
4:     set $I := I \cap \left\langle \frac{h_1}{g_i}, \ldots, \frac{h_t}{g_i} \right\rangle$ using Theorem 4.29
5: end for
6: return $I$
```

Note that the quotient eliminates only one of the "catalysts" mentioned when motivating quotient of polynomial ideals. For example $\left\langle x_1^3(x_2 - x_3) \right\rangle : \langle x_1 \rangle = \left\langle x_1^2(x_2 - x_3) \right\rangle \trianglelefteq \mathbb{Q}[x_1, x_2, x_3]$. To remove all occurrences of x_1 in this example we have to compute the so-called saturation of the two polynomial ideals.

Definition 4.36 Let $I, J \trianglelefteq R[x_1, \ldots, x_n]$ be polynomial ideals over a ring R for some $n \in \mathbb{N}_{>0}$. The saturation of I by J is the set

$$I : J^\infty := \{f \in R[x_1, \ldots, x_n] \mid \text{there is an } m \in \mathbb{N}_{>0} \text{ such that for all } g \in J^m \text{ we have } fg \in I\}$$

For a polynomial $g \in R[x_1, \ldots, x_n]$ we denote the saturation $I : \langle g \rangle^\infty$ by $I : g^\infty$. The saturation of a polynomial ideal $I \trianglelefteq R[x_1, \ldots, x_n]$ is

$$I : \left(\prod_{i=1}^{n} x_i \right)^\infty$$

A polynomial ideal is said to be saturated if and only if it is equal to its saturation.

Saturations of polynomial ideals can be computed by repeatedly computing quotients. A more efficient computation is to split the saturation into saturations with principal ideals as we did for quotients.

Theorem 4.37 Let $I, J, K \trianglelefteq R[x_1, \ldots, x_n]$ be polynomial ideals over a commutative ring R for some $n \in \mathbb{N}_{>0}$. We have

$$I : (J + K)^\infty = (I : J^\infty) \cap (I : K^\infty)$$

To compute the saturation of a polynomial ideal by a principal ideal we can utilize a theorem like Theorem 4.29 for computing the intersection of polynomial ideals.

Theorem 4.38 Let $f_1, \ldots, f_s, g \trianglelefteq R[x_1, \ldots, x_n]$ be polynomials over a commutative ring R for some $n, s \in \mathbb{N}_{>0}$. For

$$K := \langle f_1, \ldots, f_s, 1 - x_{n+1} g \rangle \trianglelefteq R[x_1, \ldots, x_{n+1}]$$

we have

$$\langle f_1, \ldots, f_s \rangle : g^\infty = K \cap R[x_1, \ldots, x_n]$$

Together, this enables us to compute the saturation of two polynomial ideals.

Algorithm 4.39 Compute the saturation of one polynomial ideal by another.

```
Input: $f_1, \ldots, f_s, g_1, \ldots, g_r \in R[x_1, \ldots, x_n]$ polynomials over a commutative ring $R$ for some
    $n, r, s \in \mathbb{N}_{>0}$
Output: a generating set of $\langle f_1, \ldots, f_s \rangle : \langle g_1, \ldots, g_r \rangle^\infty$
1: set $I := \langle 1 \rangle \trianglelefteq R[x_1, \ldots, x_n]$
2: for $i \in \{1, \ldots, r\}$ do
3:     set $K := \langle f_1, \ldots, f_s, 1 - x_{n+1} g \rangle \trianglelefteq R[x_1, \ldots, x_{n+1}]$
4:     compute polynomials $h_1, \ldots, h_t \in R[x_1, \ldots, x_n]$ for some $t \in \mathbb{N}_{>0}$ such that
       $\langle h_1, \ldots, h_t \rangle = K \cap R[x_1, \ldots, x_n]$
5:     set $I := I \cap \left\langle h_1, \ldots, h_t \right\rangle$ using Theorem 4.29
6: end for
7: return $I$
```

The quotient and saturation of polynomial ideals can also be interpreted geometrically. By computing the quotient by some ideal we remove the variety of this ideal from the variety of the original polynomial ideal.

Theorem 4.40 Let $I, J \trianglelefteq R[x_1, \ldots, x_n]$ be polynomial ideals over a commutative ring R for some $n \in \mathbb{N}_{>0}$. We have

$$\mathcal{V}(I : J) = \mathcal{V}(I : J^\infty) = \overline{\mathcal{V}(I) \setminus \mathcal{V}(J)}$$

where $\overline{\mathcal{V}(I) \setminus \mathcal{V}(J)}$ denotes the Zariski closure of $\mathcal{V}(I) \setminus \mathcal{V}(J)$.

Note that the Zariski closure is crucial here since the set $\mathcal{V}(I) \setminus \mathcal{V}(J)$ may not be a variety otherwise. For an example consider $I = \langle x_2 \rangle \trianglelefteq \mathbb{Q}[x_1, x_2]$ and $J = \langle x_1, x_2 \rangle \trianglelefteq \mathbb{Q}[x_1, x_2]$. The set $\mathcal{V}(I) \setminus \mathcal{V}(J)$ is exactly the one described in Example 4.22 for an example of a set that is not closed under the Zariski topology.

We will use the operations on polynomial ideals extensively in the following sections. In particular in Section 11.1 we will use all those operations.

5 Gröbner Bases

5.1 The Univariate Case

In this section we are going to discuss computations on univariate polynomial ideals. The techniques used when dealing with univariate polynomial ideals are similar to the case of general polynomial ideals, but the computational complexity is much lower. That is what makes univariate polynomial ideals interesting objects to study. For formal proofs of all theorems presented in this chapter we refer to the book by David Cox, John Little, and Donal O'Shea [CLO07].

First of all, we will give a remark concerning the limits of computations on univariate polynomial ideals. In contrast to the case of linear equations as discussed in Section 4.4, we are not able to actually compute the variety of univariate polynomial ideals in general. The Abel-Ruffini Theorem states that there are no general solution formulas for univariate polynomials of degree 5 or higher [Abe26]. We will discuss this theorem in detail in Section 7.1.

We can still compute many properties of univariate polynomial ideals. Several properties can be seen easily after computing a suitable generating set of the ideal. To do so we define the greatest common divisor of polynomials.

Definition 5.1 Let R be a commutative ring and $n \in \mathbb{N}_{>0}$. A common divisor of polynomials $f, g \in R[x_1, \ldots, x_n]$ is a polynomial $h \in R[x_1, \ldots, x_n]$ with the property that there are polynomials $f', g' \in R[x_1, \ldots, x_n]$ such that

$$hf' = f \text{ and } hg' = g'$$

A greatest common divisor of polynomials $f, g \in R[x_1, \ldots, x_n]$ denoted by $\gcd(f, g)$ is common divisor of f and g having maximal degree.

Note that greatest common divisors can only be unique up to invertible constants contained in $R^\times$. If we have $a \in R^\times$ and $h \in R[x_1, \ldots, x_n]$ is a greatest common divisor of $f, g \in R[x_1, \ldots, x_n]$ then ah is also a greatest common divisor of f and g. When we write $\gcd(f, g)$ without further qualification we mean any greatest common divisor of f and g. There exists always a greatest common divisor of two polynomials, because $h = 1$ is a common divisor of every pair of polynomials.

In contrast to general polynomial rings univariate polynomial rings over fields always yield a unique greatest common divisor up to constant factors.

Theorem 5.2 Let k be field, $f, g \in k[x]$ be polynomials and let $h_1, h_2 \in k[x]$ be greatest common divisors of f and g. There is a $c \in k[x]^\times = k$ such that $ch_1 = h_2$.

Equivalently, we could also say that if $h \in k[x]$ is a greatest common divisor of $f, g \in k[x]$ then every other greatest common divisor of f and g is also a divisor of h. Integral domains having one of the properties of $k[x]$ presented above are also called unique factorization domains.

Similarly to the greatest common divisor we can also define the least common multiple, usually denoted by $\operatorname{lcm}(f, g)$, of polynomials $f, g \in k[x]$ as a polynomial with smallest degree that is a multiple of f and g. An analog version of Theorem 5.2 also applies for the least common multiple. The least common multiple can be computed easily from the greatest common divisor as for all $f, g \in k[x]$ we have $\gcd(f, g) \operatorname{lcm}(f, g) = fg$.

As the greatest common divisor of univariate polynomials is unique up to invertible factors, we can safely use the greatest common divisor in contexts where invertible factors do not matter, for instance when discussing generators of polynomial ideals. Let $f, g \in k[x]$ be polynomials. Because f and g are multiples of $\gcd(f, g)$, it is clear that $\langle \gcd(f, g) \rangle \subseteq \langle f, g \rangle$. In fact, both ideals are the same since every sum of multiples of f and g is also a multiple of $\langle \gcd(f, g) \rangle$.

Theorem 5.3 Let k be a field and $f, g \in k[x]$. We have

$$\Big\langle \gcd(f, g) \Big\rangle \subseteq \langle f, g \rangle$$

This theorem can also be applied multiple times. For $f_1, \ldots, f_s \in k[x]$ and some $s \in \mathbb{N}_{>0}$ we get

$$\langle f_1, \ldots, f_s \rangle = \Big\langle \gcd\left(\ldots \gcd(\gcd(f_1, f_2), f_3), \ldots, f_s \right) \Big\rangle \trianglelefteq k[x]$$

We can apply this procedure to every univariate polynomial ideal as all of them are finitely generated due to Hilbert's Basis Theorem. This implies that every univariate polynomial ideal is a principal ideal.

Corollary 5.4 Let k be a field. $k[x]$ is a principal ideal domain.

To actually compute a greatest common divisor we can use the Euclidean algorithm named after the ancient Greek mathematician Euclid of Alexandria presented in Algorithm 5.7. To do so we need to define the leading term of a monomial.

Definition 5.5 Let k be a field. The leading term of a univariate polynomial $f \in k[x]$ is the term of f with the highest degree and usually denoted by $\mathrm{LT}(f)$. The coefficient of the leading term is called the leading coefficient and denoted by $\mathrm{LC}(f)$ while the leading monomial $\mathrm{LM}(f)$ is the leading term without its coefficient. A polynomial is called monic if and only if its leading coefficient is 1.

Additionally, we need an algorithm to reduce polynomials, which is called polynomial division. This procedure tries to add multiples of a polynomial to another one in a way that the leading term of the polynomials cancel. Given two polynomials $f, g \in k[x]$, the polynomial division computes a polynomial $r \in k[x]$ with minimal degree such that f is equivalent to r modulo $\langle g \rangle$.

Algorithm 5.6 Polynomial division of univariate polynomials.

```
Input: $f, g \in k[x]$ polynomials over a field $k$, $g \neq 0$
Output: $h, r \in k[x]$ such that $f = hg + r$ and $\deg(r) < \deg(g)$
1: set $h := 0, r := f$
2: while $\deg(r) \geq \deg(g)$ do
3:     set $t := \frac{\mathrm{LT}(r)}{\mathrm{LT}(g)}$
4:     set $h := h + t, r := r - gt$
5: end while
6: return $h, r$
```

Algorithm 5.6 returns $r = 0$ if and only if f is a multiple of g or equivalently $f \in \langle g \rangle$. We call r the remainder of the division of f by g. Computing the remainder of f modulo g is also referred to as reducing f modulo g. Computing the remainders of two polynomials iteratively finally results in their greatest common divisor.

Algorithm 5.7 Euclidean algorithm: Compute the greatest common divisor of two univariate polynomials.

```
Input: $f, g \in k[x]$ polynomials over a field $k$
Output: $\gcd(f, g)$
1: while $g \neq 0$ do
2:     compute $h, r \in k[x]$ such that $f = hg + r$ and $\deg(r) < \deg(g)$ using Algorithm 5.6
3:     set $f := g, g := r$
4: end while
5: return $f$
```

The Euclidean algorithm and the polynomial division always finish after a finite amount of steps and can be used together to compute greatest common divisors.

We now want to compute whether a univariate polynomial $g \in k[x]$ is contained in a univariate polynomial ideal $I \trianglelefteq k[x]$ over a field k. First, we will find an $f \in k[x]$ such that $I = \langle f \rangle$. Such a polynomial f can be found as explained above by computing greatest common divisors of all generators of I. After computing the generator of I we still have to check whether g is a multiple of f using polynomial division.

Algorithm 5.8 Compute whether a univariate polynomial is contained in a univariate polynomial ideal.

Input: $f_1, \dots, f_s, g \in k[x]$ polynomials over a field k for some $s \in \mathbb{N}_{>0}$
Output: whether $g \in \langle f_1, \dots, f_s \rangle$ or $g \notin \langle f_1, \dots, f_s \rangle$

```
 1: set $f := f_1$
 2: for $i \in \{2, \dots, s\}$ do
 3:     compute $f := \gcd(f, f_i)$ using Algorithm 5.7
 4: end for
 5: compute $h, r \in k[x]$ such that $g = hf + r$ and $\deg(r) < \deg(f)$ using Algorithm 5.6
 6: if $r = 0$ then
 7:     return $g \in \langle f_1, \dots, f_s \rangle$
 8: else
 9:     return $g \notin \langle f_1, \dots, f_s \rangle$
10: end if
```

5.2 Definition of Gröbner Bases

We will now discuss an algorithm, that computes whether a polynomial is contained in a multivariate polynomial ideal like Algorithm 5.8 does for univariate polynomial ideals. There are some new concepts that have to be introduced for multivariate polynomial ideals. First, the leading term of a multivariate polynomial is not as easy to define as in the univariate case. Second, multivariate polynomial rings are no principal ideal domains, which means that we need another definition of the special basis of the polynomial ideal we compute. Third, polynomial division does not work as well as in the principal ideal case if we have to divide by several polynomials as their ordering can influence the result of the polynomial division.

All these issues were addressed by Bruno Buchberger who introduced Gröbner Bases in his Ph.D. thesis in 1965 [Buc65]. Gröbner Bases are special generating sets of multivariate polynomial ideals that are named after Buchberger's advisor Wolfgang Gröbner. Buchberger also presented an algorithm to compute Gröbner Bases and thus to compute whether a polynomial is contained in a multivariate polynomial ideal. Similar concepts were also independently introduced by Nikolai M. Gunther in 1913 [Gun41, RRRA03] and Fields medalist Heisuke Hironaka in 1964 [Hir64].

We will now discuss how to resolve the three obstacles listed above. First, we have to find a way to order monomials. For univariate monomials over a field k we said that a monomial

$x^a \in k[x]$ is larger than another monomial $x^b \in k[x]$ for some $a, b \in \mathbb{N}_0$ if and only if its degree is higher, i.e. $a \geq b$ or x^a is a multiple of x^b. For multivariate polynomials, it is not immediately clear which monomial should be larger than another in general. For instance $x_1 \in \mathbb{Q}[x_1, x_2]$ and $x_2 \in \mathbb{Q}[x_1, x_2]$ have the property that none of them is a multiple of the other one. We can order those monomials in an arbitrary way, but we still want multiples of monomials to be larger than the original monomials.

Definition 5.9 Let R be a ring and $n \in \mathbb{N}_{>0}$. A monomial ordering or term ordering $\geq$ on $R[x_1, \ldots, x_n]$ is an ordering of the monomials of $R[x_1, \ldots, x_n]$ such that

a) $\geq$ is a total ordering, i.e. for each two monomials $\underline{x}^u, \underline{x}^v \in R[x_1, \ldots, x_n]$ given by some $u, v \in \mathbb{N}_0^n$ we have $\underline{x}^u \preceq \underline{x}^v$ or $\underline{x}^u \succeq \underline{x}^v$, and the relations $\underline{x}^u \preceq \underline{x}^v$ and $\underline{x}^u \succeq \underline{x}^v$ together imply $u = v$

b) $\underline{x}^{u+w} \preceq \underline{x}^{v+w}$ for each $u, v, w \in \mathbb{N}_0^n$ with $\underline{x}^u \preceq \underline{x}^v$

c) $\geq$ is a well-ordering, i.e. for every set M of monomials contained in $R[x_1, \ldots, x_n]$ there is $\underline{x}^u \in M$ for some $u \in \mathbb{N}_0^n$ with $\underline{x}^u \preceq \underline{x}^v$ for all $v \in \mathbb{N}_0^n$ with $\underline{x}^v \in M$

Even though we included the ring R in the definition, the term ordering is independent of the coefficient ring. All term orderings are also term orderings on any polynomial ring over another arbitrary coefficient ring. Given a term ordering we can use the leading term, leading monomial, and leading coefficient as in the univariate case.

The definition of term orderings allows for multiple term orderings on multivariate polynomial rings. All possible term orderings were classified by Lorenzo Robbiano in 1985 [Rob85]. Though there are infinitely many term orderings on a polynomial ring in general, there are some term orderings that are used most frequently.

Example 5.10 Let R be a ring and $n \in \mathbb{N}_{>0}$. The following are term orderings on $R[x_1, \ldots, x_n]$.

a) The lexicographic term ordering $\geq_{\text{lex}}$ has $\underline{x}^u \geq_{\text{lex}} \underline{x}^v$ if and only if $u = v$ or the left-most non-zero entry of $u - v \in \mathbb{Z}^n$ is positive for all $u, v \in \mathbb{N}_0^n$.

b) The reverse lexicographic term ordering $\geq_{\text{revlex}}$ has $\underline{x}^u \geq_{\text{revlex}} \underline{x}^v$ if and only if $u = v$ or the right-most non-zero entry of $u - v \in \mathbb{Z}^n$ is negative for all $u, v \in \mathbb{N}_0^n$.

c) The graded lexicographic term ordering $\geq_{\text{grlex}}$ has $\underline{x}^u \geq_{\text{grlex}} \underline{x}^v$ if and only if $\deg(u) > \deg(v)$ or $\deg(u) = \deg(v)$ and $\underline{x}^u \geq_{\text{lex}} \underline{x}^v$ for all $u, v \in \mathbb{N}_0^n$.

d) The graded reverse lexicographic term ordering $\geq_{\text{grevlex}}$ has $\underline{x}^u \geq_{\text{grevlex}} \underline{x}^v$ if and only if $\deg(u) > \deg(v)$ or $\deg(u) = \deg(v)$ and $\underline{x}^u \geq_{\text{revlex}} \underline{x}^v$ for all $u, v \in \mathbb{N}_0^n$.

All the term orderings above have $x_1 \geq x_2 \geq \cdots \geq x_n$, but the ordering of the indeterminates may be changed. While all complexity results presented in this thesis hold for all term orderings, experiments suggest that in practice the graded reverse lexicographic term ordering often results in the fastest running times of several algorithms.

All term orderings mentioned above coincide on a univariate polynomial ring in the ordering $1 \leq x \leq x^2 \leq x^3 \dots$. In fact, this is the only term ordering on a univariate polynomial ring, which is why we did not have to worry about the term ordering in the univariate case.

If we fix a term ordering we can also execute the polynomial division presented in Algorithm 5.6 for multivariate polynomials. Note that multivariate polynomial rings are no principal ideal domains in general. Thus, we have to divide by several polynomials simultaneously.

Algorithm 5.11 Polynomial division of multivariate polynomials

```
Input: $f, g_1, \dots, g_s \in k[x_1, \dots, x_n]$ polynomials over a field $k$ for some $n, s \in \mathbb{N}_{>0}$ with $g_i \neq 0$
    for all $i \in \{1, \dots, s\}$ and a term ordering $\geq$ on $k[x_1, \dots, x_n]$
Output: $h_1, \dots, h_s, r \in k[x]$ such that $f = h_1 g_1 + \cdots + h_s g_s + r$ and no term of $r$ is divisible
    by $\mathrm{LM}(g_i)$ for all $i \in \{1, \dots, s\}$
 1: set $h_1 := 0, \dots, h_s := 0, r := 0$
 2: set $p := f$
 3: while $p \neq 0$ do
 4:     set $nodivisor$ := true
 5:     for $i \in \{1, \dots, s\}$ do
 6:         if $\mathrm{LM}(p)$ is a multiple of $\mathrm{LM}(g_i)$ then
 7:             set $t := \frac{\mathrm{LT}(p)}{\mathrm{LT}(g_i)}$
 8:             set $h_i := h_i + t, p := p - g_i t$
 9:             set $nodivisor$ := false
10:         end if
11:     end for
12:     if $nodivisor$ then
13:         set $r := r + \mathrm{LT}(p)$
14:         set $p := p - \mathrm{LT}(p)$
15:     end if
16: end while
17: return $h_1, \dots, h_s, r$
```

Note that Algorithm 5.11 returns exactly the same as Algorithm 5.6 does when invoked on a univariate polynomial ring with $s = 1$. The remainder r of a polynomial f modulo $g_1, \dots, g_s$ is also called its normal form and usually denoted by $\overline{f}^{g_1, \dots, g_s}$.

In contrast to Algorithm 5.6 on the other hand, Algorithm 5.11 does not have the property that $r = 0$ if and only if $f \in \langle g_1, \dots, g_s \rangle$. The following example by David Cox, John Little, and Donal O'Shea shows that the result of multivariate polynomial division depends on the order of the divisors [CLO07].

Example 5.12 ([CLO07]) Let $f := x_1x_2^2 - x_1 \in \mathbb{Q}[x_1, x_2]$ and consider the lexicographic term ordering $\geq_{\text{lex}}$ on $\mathbb{Q}[x_1, x_2]$ with $x_1 \geq_{\text{lex}} x_2$. Dividing f by $x_1x_2 + 1$ and $x_2^2 - 1$ using Algorithm 5.11 results in

$$\underbrace{x_1x_2^2 - x_1}_{f} = \underbrace{x_2}_{h_1}\underbrace{(x_1x_2 + 1)}_{g_1} + \underbrace{0}_{h_2}\underbrace{(x_2^2 - 1)}_{g_2} + \underbrace{(-x_1 - x_2)}_{r}$$

while exchanging the divisors results in

$$\underbrace{x_1x_2^2 - x_1}_{f} = \underbrace{x_1}_{h_1}\underbrace{(x_2^2 - 1)}_{g_1} + \underbrace{0}_{h_2}\underbrace{(x_1x_2 + 1)}_{g_2} + \underbrace{0}_{r}$$

This is a huge problem for computations using multivariate polynomial division. The remainder of a polynomial modulo the generators of a polynomial ideal should be unique to allow for membership tests. It turns out, that the remainders are only ambiguous for some generating set and there is always a generating set with unique remainders. Those generating sets are called Gröbner Bases.

Definition 5.13 Let k be a field, $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ be polynomials for some $n, s \in \mathbb{N}_{>0}$, and let $\geq$ be a term ordering on $k[x_1, \ldots, x_n]$. $\{f_1, \ldots, f_s\}$ is said to be a Gröbner Basis of $\langle f_1, \ldots, f_s\rangle \trianglelefteq k[x_1, \ldots, x_n]$ with respect to $\geq$ if and only if for all $g \in k[x_1, \ldots, x_n]$ the remainder of g modulo $f_1, \ldots, f_s$ in any order computed by Algorithm 5.11 using $\geq$ is $r = 0$ if and only if $g \in \langle f_1, \ldots, f_s\rangle$.

This definition makes one usage of Gröbner Bases immediately clear, but it is more common in the literature to define Gröbner Bases with the help of the ideal of leading terms. Both definitions are equivalent.

Theorem 5.14 Let k be a field, $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ be polynomials for some $n, s \in \mathbb{N}_{>0}$, and let $\geq$ be a term ordering on $k[x_1, \ldots, x_n]$. The ideal of leading terms of the polynomial ideal $\langle f_1, \ldots, f_s\rangle \trianglelefteq k[x_1, \ldots, x_n]$ is the ideal

$$\mathrm{LT}(\{f_1, \ldots, f_s\}) := \mathrm{LT}(\langle f_1, \ldots, f_s\rangle) := \Big\langle \mathrm{LT}(f) \mid f \in \langle f_1, \ldots, f_s\rangle \Big\rangle$$

$\{f_1, \ldots, f_s\}$ is a Gröbner Basis of $\langle f_1, \ldots, f_s\rangle \trianglelefteq k[x_1, \ldots, x_n]$ with respect to $\geq$ if and only if

$$\Big\langle \mathrm{LT}(f_1), \ldots, \mathrm{LT}(f_s) \Big\rangle = \mathrm{LT}(I)$$

Every time we refer to the leading term of a polynomial, we mean the leading term of that polynomial with respect to the term ordering $\geq$. Theorem 5.14 states that a generating set of a polynomial ideal is a Gröbner Basis if and only if the leading term of every polynomial contained in the polynomial ideal is a multiple of the leading term of one of the generators. Subsequently, when reducing a polynomial contained in the polynomial ideal modulo a Gröbner Basis we always find a generator such that the if-statement in line 6 of Algorithm 5.11 is fulfilled.

Note that Gröbner Bases as defined here are not unique. For instance, we could add additional polynomials contained in the polynomial ideal to every Gröbner Basis and the result would still be a valid Gröbner Basis. Also, we could multiply generators by constants or reduce generators modulo other generators to get other Gröbner Bases. To remove those generators, that are not needed or ambiguous, we define minimal and reduced Gröbner Bases.

Definition 5.15 Let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a polynomial ideal over a field k for some $n \in \mathbb{N}_{>0}$ and let $G \subseteq k[x_1, \ldots, x_n]$ be a Gröbner Basis of I with respect to some term ordering $\geq$ on $k[x_1, \ldots, x_n]$. G is a minimal Gröbner Basis of I with respect to $\geq$ if and only if

a) $\mathrm{LC}(f) = 1$ for every $f \in G$

b) $\mathrm{LT}(G) \neq \mathrm{LT}(G \setminus \{f\})$ for every $f \in G$

G is a reduced Gröbner Basis of I with respect to $\geq$ if and only if it is a minimal Gröbner Basis and for all $f, g \in G$ no monomial of f is a multiple of $\mathrm{LM}(g)$.

Interestingly, it turns out that it is already enough to remove unnecessary generators, scale all generators to the same leading coefficient, and reduce all generators modulo the other generators to make Gröbner Bases unique.

Theorem 5.16 Let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a polynomial ideal over a field k for some $n \in \mathbb{N}_{>0}$ and let $\geq$ be a term ordering on $k[x_1, \ldots, x_n]$. The reduced Gröbner Basis of I with respect to $\geq$ is unique.

The computation of a reduced Gröbner Basis from any other Gröbner Basis is straight-forward from the definition.

Algorithm 5.17 Compute a reduced Gröbner Basis from any Gröbner Basis

Input: $f_1, \dots, f_s \in k[x_1, \dots, x_n]$ polynomials over a field k for $n, s \in \mathbb{N}_{>0}$ and a term ordering $\geq$ on $k[x_1, \dots, x_n]$ such that $\{f_1, \dots, f_s\} \subseteq k[x_1, \dots, x_n]$ is a Gröbner Basis of $\langle f_1, \dots, f_s \rangle \trianglelefteq k[x_1, \dots, x_n]$ with respect to $\geq$

Output: the reduced Gröbner Basis of $\langle f_1, \dots, f_s \rangle \trianglelefteq k[x_1, \dots, x_n]$ with respect to $\geq$

```
1: set $G := \emptyset$
2: for $i \in \{1, \dots, s\}$ do
3:    compute the remainder $r_1 \in k[x_1, \dots, x_n]$ of $\mathrm{LM}(f_i)$ modulo $\mathrm{LT}(G)$ with respect to $\geq$
      using Algorithm 5.11
4:    if $r_1 \neq 0$ then
5:       compute the remainder $r_2 \in k[x_1, \dots, x_n]$ of $f_i$ modulo $G$ with respect to $\geq$ using
         Algorithm 5.11
6:       set $G := G \cup \left\{\frac{r_2}{\mathrm{LC}(r_2)}\right\}$
7:    end if
8: end for
9: return $G$
```

There are also other algorithms to transform Gröbner Bases. Algorithms to change the term ordering of a Gröbner Basis, for instance, include the FGLM algorithm by Jean-Charles Faugère et al. [FGLM93] and the Gröbner walk.

5.3 Buchberger's Algorithm

Bruno Buchberger also presented an algorithm to compute Gröbner Bases in his Ph.D. thesis [Buc65]. As we saw in Theorem 5.14, we need the leading monomials of a Gröbner Basis to generate the full ideal of leading terms of the polynomial ideal. To generate new leading monomials of the generators, it is therefore useful to combine them in a way that the existing leading terms cancel and new leading terms are generated. This is done by the computation of so-called subtraction polynomials or short s-polynomials.

Definition 5.18 Let $f, g \in k[x_1, \dots, x_n]$ be polynomials over a field k for some $n \in \mathbb{N}_{>0}$ and let $\geq$ be a term ordering on $k[x_1, \dots, x_n]$. The s-polynomial of f and g is the polynomial

$$\mathrm{spol}(f, g) := \frac{\mathrm{lcm}(\mathrm{LM}(f), \mathrm{LM}(g)) f}{\mathrm{LT}(f)} - \frac{\mathrm{lcm}(\mathrm{LM}(f), \mathrm{LM}(g)) g}{\mathrm{LT}(g)}$$

s-polynomials also appear while reducing polynomials. Note that the multivariate polynomial division as presented in Algorithm 5.11 just repeatedly computes s-polynomials. If the leading term of f is a multiple of the leading term of g, as it is in the situation of this algorithm, we get

$$\operatorname{spol}(f,g) = \frac{f}{\operatorname{LC}(f)} - \frac{\operatorname{LM}(f)\,g}{\operatorname{LT}(g)}$$

which are exactly the formulas computed in Algorithm 5.11.

Buchberger's Criterion implies that computing s-polynomials and reducing them modulo the existing generators is everything that needs to be done to compute a Gröbner Basis.

Theorem 5.19 (Buchberger's Criterion [Buc65]) Let $f_1, \dots, f_s \in k[x_1, \dots, x_n]$ be polynomials over a field k for some $n, s \in \mathbb{N}_{>0}$ and let $\geq$ be a term ordering on $k[x_1, \dots, x_n]$. $\{f_1, \dots, f_s\}$ is a Gröbner Basis of $\langle f_1, \dots, f_s \rangle \trianglelefteq k[x_1, \dots, x_n]$ with respect to $\geq$ if and only if for all $i, j \in \{1, \dots, s\}$ with $i \neq j$ the remainder of $\operatorname{spol}\left(f_i, f_j\right)$ modulo $\{f_1, \dots, f_s\}$ in any order with respect to $\geq$ computed using Algorithm 5.11 is 0.

With Buchberger's Criterion we can immediately state an algorithm to compute a Gröbner Basis of a polynomial ideal.

Algorithm 5.20 Buchberger's Algorithm: Compute a Gröbner Basis of a polynomial ideal

```
Input: $f_1, \dots, f_s \in k[x_1, \dots, x_n]$ polynomials over a field $k$ for $n, s \in \mathbb{N}_{>0}$, a term ordering $\geq$
    on $k[x_1, \dots, x_n]$
Output: a Gröbner Basis of $\langle f_1, \dots, f_s \rangle \trianglelefteq k[x_1, \dots, x_n]$ with respect to $\geq$
 1: set $G := \{f_1, \dots, f_s\}$, $G' := \emptyset$
 2: while $G \neq G'$ do
 3:     set $G' := G$
 4:     for $f, g \in G'$ do
 5:         compute the remainder $r \in k[x_1, \dots, x_n]$ of $\operatorname{spol}(f,g)$ modulo $G$ with respect to $\geq$
            using Algorithm 5.11
 6:         if $r \neq 0$ then
 7:             set $G := G \cup \{r\}$
 8:         end if
 9:     end for
10: end while
11: return $G$
```

We will discuss the time and space complexity of Buchberger's Algorithm and other algorithms to compute Gröbner Bases in Section 7.2 in detail.

Gröbner Bases are useful for solving the membership problem of polynomial ideals as shown above, but they can also be used to solve many other computational problems on polynomial ideals. In Section 4.6 we discussed operations on polynomial ideals and in particular elimination ideals as defined in Definition 4.28. Elimination ideals can be computed as Gröbner Bases of polynomial ideals with respect to so-called elimination orderings as term orderings.

Theorem 5.21 (Elimination Theorem) Let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a polynomial ideal over a field k for some $n \in \mathbb{N}_{>0}$. Let $G \subseteq k[x_1, \ldots, x_n]$ be a Gröbner Basis of I with respect to the lexicographic term ordering $\succeq_{\mathrm{lex}}$ with $x_1 \preceq_{\mathrm{lex}} x_2 \preceq_{\mathrm{lex}} \cdots \preceq_{\mathrm{lex}} x_n$. For every $s \in \{1, \ldots, n-1\}$ the s-th elimination ideal $I \cap k[x_1, \ldots, x_s]$ of I is generated by $G \cap k[x_1, \ldots, x_s]$.

With the help of the Elimination Theorem and Buchberger's Algorithm we can compute in particular intersections of polynomial ideals as explained in Theorem 4.29, quotients of polynomial ideals as explained in Algorithm 4.35, and saturations of polynomial ideals as explained in Algorithm 4.39.

Part III

Complexity Results

6 The Computational Model

6.1 Historical Introduction

In this chapter we will define the computational model that we use for the complexity analysis of all algorithms. When we think about computational models in practice there are numerous approaches from data centers, computers with various hardware, calculators, devices like an abacus, or even human brains that use neurons and DNA. Yet we will see that it is believed that all those ways of computing have similar computational power and there is an easy theoretical computational model, that we can use in proofs, which is mathematically equivalent to all ways of computing in nature.

For computations on large problems it is essential to execute those computations in a timely manner and without making errors. Humans experience problems with both of those properties. The creation of mechanical or electronic computation devices was therefore a huge breakthrough in the history of computation.

One of the first notable mechanical computation machines was invented by Charles Babbage. In 1822 he proposed to build the Difference Engine, a device for computing large tables of values of polynomial functions. Even though Babbage was never able to complete a physical realization of this machine, it was constructed in the 1990s to prove that his ideas and concepts actually work. Charles Babbage abandoned the project of the Difference Engine to describe the so-called Analytical Engine in 1837. This machine was never constructed too. Nevertheless, it was a milestone in the history of computation as the Analytical Engine is the first general-purpose computation device. The description of the steps, that the machine should execute, i.e. the algorithm itself, is not hardwired into the design of the device as for all known machines before, but rather part of the input. Thus, the Analytical Engine was in theory able to solve a wide range of problems instead of just computing one specific problem, like for instance the Difference Engine.

There was still no mechanical general-purpose computation device available in 1936, but in this year two theoretical computational models were introduced. Alonzo Church introduced the λ-calculus [Chu36] and Alan M. Turing proposed the Turing Machine [Tur37b]. It was shown that both computational models are mathematically equivalent [Tur37a], but – as even Church acknowledged – the Turing Machine to be the better representation of the computational model, it became the standard theoretical model of computation.

Both models are important because it has been proposed that every physically realizable computation device can be simulated by a Turing Machine. This claim is known as the (weak) Church-Turing thesis [Kle43] and generally believed to be true. The more powerful strong Church-Turing thesis states that every physically realizable computation device can be efficiently simulated by a Turing Machine. Efficiently simulated in this context means, that there is at most polynomial overhead, i.e. for every physically realizable computation device there is a constant $c \in \mathbb{N}_{>0}$ such that every computation on this device with $t \in \mathbb{N}_{>0}$ steps can be simulated on a Turing Machine in at most t^c steps. There is some debate whether the strong Church-Turing thesis is true, in particular in the light of models like quantum computers. There is currently no known way to efficiently simulate a quantum computer using a Turing Machine, but there are also no known physical realizations of quantum computers.

Turing later constructed mechanical devices to crack codes encrypted with the “Enigma” cipher during World War II. However, those were also tailored for a single purpose. During and after the war the first electronic computation devices were constructed. John von Neumann was one of the most well-known pioneers in this field and introduced the “von Neumann architecture” in 1945, which is the design principle of computers until today [vN93].

The power and technology of computers has evolved drastically since then, but the underlying principles are still similar today. In the next section we will introduce a definition of Turing Machines and some generalizations of the model.

We will not give proofs of all statements in this chapter and refer to the textbook by Sanjeev Arora and Boaz Barak for the proofs and further details [AB09].

6.2 Turing Machines

There are many equivalent definitions of Turing Machines. We will state the one with the shortest definition here. This definition requires only one tape, which is used for input and output as well as a work tape. The following definition is not the most convenient to use but formally it can be stated very easily.

Definition 6.1 A deterministic Turing Machine M is a tuple $(\Gamma, Q, q_0, q_1, \delta)$ where Γ is a finite set of tape symbols, also called the alphabet of M, Q is a set of states, $q_0 \in Q$ is the initial state of M, $q_1 \in Q$ is the halting state of M, and $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, N, R\}$ is the transition function of M.

The Turing Machine has a tape consisting of cells. Each cell of the tape can be filled with exactly one symbol from the set Γ. The cells are indexed by the set $\mathbb{N}_0$, i.e. the tape is one-dimensional, finite on one side and infinite on the other side. We assume that the tape starts with cell zero at the left and indices are growing to the right. Each Turing Machine also has a head for reading and writing, which points to one cell of the tape.

A configuration of a Turing Machine consists of its current state, the contents of the tape, and the position of the head. Initially, a Turing Machine M is in the initial state q_0, has some input as the content of the tape and the head points to cell 0. We usually assume that the input is finite and that the remainder of the tape is filled with a blank symbol $\square \in \Gamma$. To execute it, the Turing Machine takes steps and transitions to another configuration in each step. For each step it looks up $\delta(q, \gamma)$ where q is the current state of the machine and γ is the symbol at the cell the head currently points to. The triple $\delta(q, \gamma)$ defines the next state, a symbol to write in the cell the head currently points to, and a direction to move the head to afterwards: L for one cell to the left, N for no movement, and R for one cell to the right. The head is never allowed to leave the tape, i.e. to go to the left when it is at cell 0. All other contents of the tape stay the same. Now the machine is in a new configuration and ready to execute the next step. The execution stops when the machine reaches the halting state q_1. The content of the tape after the machine stops is the output of the computation. The position of the head when the machine stops is discarded. The time, that a computation needs, is the number of steps the Turing Machine executes before it stops. The space, that a computation needs, is the number of total cells read or written by the head on the tape during that computation.

To allow a formal discussion, we denote the contents of the tape as strings over the alphabet Γ ignoring the (infinitely many) blank symbols on the right. The set Γ^n contains all strings of n characters over Γ for some $n \in \mathbb{N}_0$ and we define $\Gamma^* := \bigcup_{i \in \mathbb{N}_0} \Gamma^i$ as the set of all strings of finite length over Γ. The empty word, which contains no symbols, is usually denoted by ϵ and thus we have $\Gamma^0 = \{\epsilon\}$ for all sets Γ. The elements of Γ^* are also called words over the alphabet Γ. The operation Γ^* is called the Kleene star named after the American mathematician Stephen Kleene. We can also understand the set of finite words Γ^* over an alphabet Γ as a monoid with concatenation as the operation and ϵ as the neutral element. The construction of the Kleene star can also be generalized to other monoids.

Example 6.2 As an example we construct a Turing Machine, that decides whether a number is divisible by 3. The alphabet is $\Gamma = \{0, 1, \square\}$ and the input is given as a binary encoded number on the tape where the least significant bit is in cell zero. The remainder of the tape is filled with blank symbols $\square$. We use that a binary number is divisible by 3 if and only if its alternating digit sum is divisible by 3. The machine will first compute the alternating digit sum modulo 3 while erasing the input and then print 1 if the input was divisible by 3 and 0 else. The set of states $Q = \{q_0, q_1, \ldots, q_7\}$ is described in Table 6.3, the transition function δ is defined in Table 6.4 and an example run for the input 53 is given in Table 6.5. The succession of the states can also be drawn as a graph as shown in Figure 6.6

Note that this Turing Machine does not print its output in the first cell of the tape, but in some other cell depending on the length of the input to make the description easier. The machine can easily be extended to write the output in the leftmost cell.

state	description
q_0	initial state, redirects to state q_2
q_1	halting state
q_2	even number of digits, alternating digit sum has remainder 0 modulo 3
q_3	even number of digits, alternating digit sum has remainder 1 modulo 3
q_4	even number of digits, alternating digit sum has remainder 2 modulo 3
q_5	odd number of digits, alternating digit sum has remainder 0 modulo 3
q_6	odd number of digits, alternating digit sum has remainder 1 modulo 3
q_7	odd number of digits, alternating digit sum has remainder 2 modulo 3

Table 6.3: The states of a Turing Machine to decide whether a number is divisible by 3.

Q	Γ	Q	Γ	$\{L, N, R\}$
q_0	0	q_2	0	N
q_0	1	q_2	1	N
q_2	0	q_5	$\square$	R
q_2	1	q_6	$\square$	R
q_2	$\square$	q_1	1	N
q_3	0	q_6	$\square$	R
q_3	1	q_7	$\square$	R
q_3	$\square$	q_1	0	N
q_4	0	q_7	$\square$	R
q_4	1	q_5	$\square$	R
q_4	$\square$	q_1	0	N
q_5	0	q_2	$\square$	R
q_5	1	q_4	$\square$	R
q_5	$\square$	q_1	1	N
q_6	0	q_3	$\square$	R
q_6	1	q_2	$\square$	R
q_6	$\square$	q_1	0	N
q_7	0	q_4	$\square$	R
q_7	1	q_3	$\square$	R
q_7	$\square$	q_1	0	N

Table 6.4: The transition function of a Turing Machine to decide whether a number is divisible by 3. The left two columns specify the input of $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, N, R\}$ while the right three columns specify the output. Only inputs that appear during the computation of a valid input are listed here.

state	head position	tape content
q_0	0	$1, 0, 1, 0, 1, 1, \square, \square, \ldots$
q_2	0	$1, 0, 1, 0, 1, 1, \square, \square, \ldots$
q_6	1	$\square, 0, 1, 0, 1, 1, \square, \square, \ldots$
q_3	2	$\square, \square, 1, 0, 1, 1, \square, \square, \ldots$
q_7	3	$\square, \square, \square, 0, 1, 1, \square, \square, \ldots$
q_4	4	$\square, \square, \square, \square, 1, 1, \square, \square, \ldots$
q_5	5	$\square, \square, \square, \square, \square, 1, \square, \square, \ldots$
q_4	6	$\square, \square, \square, \square, \square, \square, \square, \square, \ldots$
q_1	6	$\square, \square, \square, \square, \square, \square, 0, \square, \ldots$

Table 6.5: The configurations of a Turing Machine during a sample run with input 53.

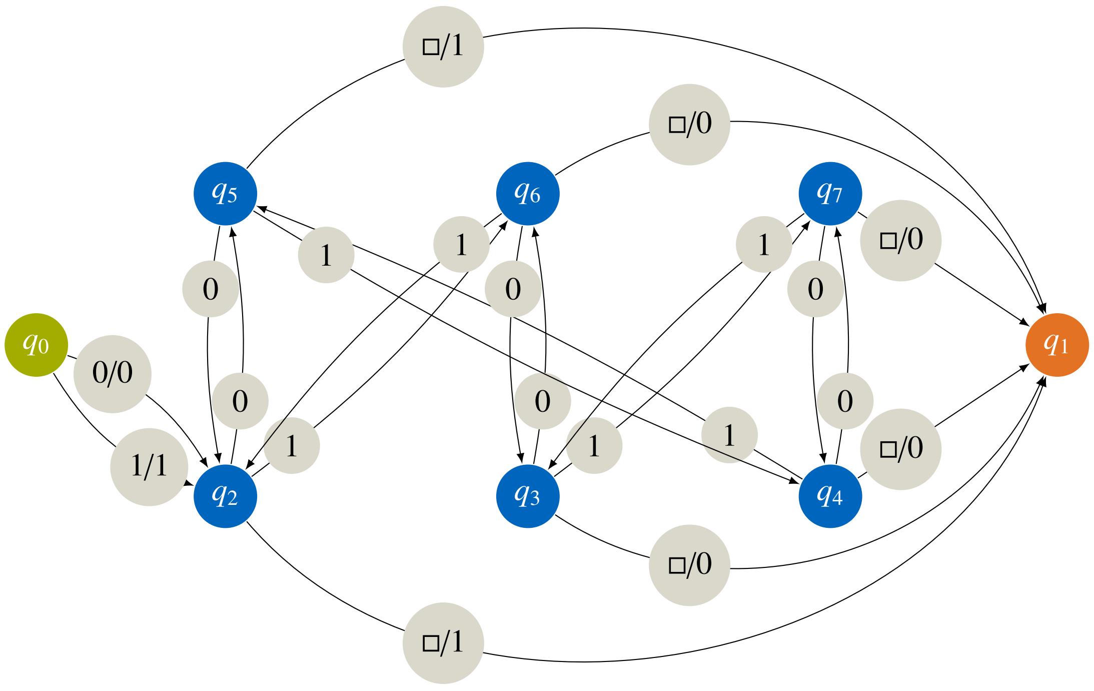


Figure 6.6: A visualization of the succession of states as given in Table 6.4. The symbols read are denoted as labels of the edges. The symbols to write are given behind a slash if not the blank symbol. The moving direction of the head is omitted to increase readability: The first and last step do not move the head, all other steps move it to the right.

When dealing with Turing Machines with finite running time we want to quantify the running time more precisely. Since the running time for most problems depends on the input size we do not use constant numbers but functions to measure running times.

Definition 6.7 Let M be a Turing Machine and let $s, t : \mathbb{N}_0 \to \mathbb{N}_0$ be functions. M is said to run in time t and space s if and only if for every input of length n, i.e. with all cells but the first n cells blank in the beginning, the execution of the machine terminates after at most $t(n)$ steps and uses at most $s(n)$ cells of the tape during the whole computation. The length of an input $x \in \{0, 1\}^*$ is the number of cells on the tape needed to describe this input and denoted by $|x|$.

We assume that the input is given in binary encoding, i.e. integers have an encoding size logarithmic in the actual number. If there are several inputs they are given one after another divided by some special tape symbol.

The machine explained above runs in time $t : \mathbb{N}_0 \to \mathbb{N}_0, n \mapsto n + 2$ and space $s : \mathbb{N}_0 \to \mathbb{N}_0, n \mapsto n + 1$. Summands like "+1" or "+2" in those examples are only a minor influence for large inputs, and in most applications only large inputs have critical running times. Thus, we usually ignore small summands by speaking about the asymptotic running time or space consumption. To do so, we use the Landau symbols named after Edmund Landau [Lan09] which is also called Big O notation. This notation orders functions by their growth instead of absolute values. For a formal definition of this notation we refer the reader to computer science textbooks [AB09, Knu68]. Using this notation we can say that our Turing Machine runs in time and space $O(n)$ or linear time and space.

The Turing Machine from Example 6.2 is an example for solving a so-called decision problem.

Definition 6.8 A decision problem or language L is defined by a subset of the set of strings $\{0, 1\}^*$ consisting of zeros and ones only. The decision problem entails, given a string $x \in \{0, 1\}^*$, to decide whether $x \in L$. A deterministic Turing Machine M is said to decide the problem L if and only if it terminates after a finite number of steps for all inputs in $x \in \{0, 1\}^*$ and writes 1 to the output tape if $x \in L$ and 0 if $x \notin L$. A decision problem $L = \{0, 1\}^*$ is called decidable if and only if there is a Turing Machine deciding L. A problem L can be decided deterministically in time $t : \mathbb{N}_0 \to \mathbb{N}_0$ and space $s : \mathbb{N}_0 \to \mathbb{N}_0$ if and only if there is a Turing Machine M that decides L and runs in time t and space s.

For some problems the input is already given as a bit string, for instance the input of Example 6.2. Other problems have to be encoded into a bit string first. We will always assume that this is done in the following without explicitly mentioning it.

If there is a Turing Machine that computes the correct result for all inputs of a problem in a finite number of steps we call this problem decidable. We will only deal with decidable problems in this thesis, but it is interesting to note that there are undecidable problems. An example of an undecidable problem is the halting problem, i.e. the problem to decide whether for a given Turing Machine there is an input such that the execution of the Turing Machine on this input takes an infinite number of steps. The existence of undecidable problems can also be used to prove Kurt Gödel's famous theorem that sound systems of axioms and inference rules are not complete, i.e. cannot be used to prove all true mathematical statements [AB09, Göd31].

There are many similar definitions of Turing Machines to the one given in Definition 6.1. A list of possible changes to the definition, that do not change the computational power of the machines, follows, i.e. the same problems are decidable with each definition and each definition can be simulated by any other definition with at most polynomial overhead.

a) The alphabet can be limited to $\Gamma = \{0, 1\}$.

b) Several heads can be allowed. During each step only one of the heads is allowed to write and read.

c) The input can be given on a separate input tape and the output can be written to a separate output tape. All other tapes are called work tapes.

d) The tape can be assumed to be infinite in both directions.

e) The machine can be allowed to use multiple work tapes instead of just one work tape.

f) Instead of a one-dimensional tape it is possible to allow two-dimensional tapes or tapes of arbitrary finite dimensions.

g) It is possible to require that the Turing Machine moves the head only dependent on the length of the input and independent of the actual input. Such a machine is called an oblivious Turing Machine.

As explained above, Turing Machines are a theoretical model of computation and not meant to be constructed as defined here. A construction of a Turing Machine is not even possible since there are only finite tapes available in practice. Nevertheless, there are several real working machines showcasing the definition of a Turing Machine like the one built by Mike Davey in 2010 shown in Figure 6.9 [Dav10]. In practice, other computation devices are used and the strong Church-Turing thesis implies that their computational power is equivalent.

Turing Machines also have the same computational power as modern programming languages like Java or C++. While simulating a Turing Machine in one of those languages efficiently is an easy task, the efficient simulation of higher programming languages using Turing Machines is very effortful, but still possible.

It is also important to note that Turing Machines are able to efficiently simulate themselves. Each Turing Machine can be binary encoded by encoding the states and the transition func-

Figure 6.9: A construction of a Turing Machine (with finite tape) built by Mike Davey [Dav10].

tion. This encoding can also be defined in a way such that every bit string of zeros and ones represents a Turing Machine.

Theorem 6.10 (universal Turing Machine) There is a so-called universal Turing Machine U, that takes the binary encoding of a Turing Machine M and an input x, and prints the result of M executed on input x within a number of steps that is polynomial in the number of steps, that M would take when executed on input x.

This simulation can even be done with a logarithmic overhead only.

6.3 Non-Determinism

Some computations are very hard, but positive results can be verified easily. For instance, this is the case when solving polynomial equations. Finding solutions is hard as we will see in this thesis. On the other hand, the verification whether a given set of values for the indeterminates indeed fulfills the equations is computationally very easy by plugging the values into the equations and checking that both sides of the equations result in the same number. We will speak about problems like that in this subsection and adjust our definition of Turing Machines accordingly.

To use a given solution we can think about Turing Machines that are able to take a hint or as Turing Machines that just guess some solution and then try to verify it. Adding the ability to guess to Definition 6.1 results in a non-deterministic Turing machine.

Definition 6.11 A non-deterministic Turing Machine M is a tuple $(\Gamma, Q, q_0, q_1, q_2, \delta_1, \delta_2)$ where Γ is a finite set of tape symbols, also called the alphabet of M, Q is a set of states, $q_0 \in Q$ is the initial state of M, $q_1 \in Q$ is the halting state of M, $q_2 \in Q$ is the accepting state of M, and $\delta_1, \delta_2 : Q \times \Gamma \to Q \times \Gamma \times \{L, N, R\}$ are the transition functions of M.

A non-deterministic Turing Machine works like a deterministic Turing machine, but it also terminates the execution when it reaches the accepting state and in each step it randomly chooses one of the transition functions δ_1 and δ_2 to apply. Since there are multiple possible outputs of non-deterministic Turing Machines, depending on which transition function was used at which step, we have to refine Definition 6.8 of a decision problem in this case.

Definition 6.12 A non-deterministic Turing Machine M is said to decide a decision problem L if and only if for all input bit strings $x \in \{0, 1\}^*$

a) there is a choice of transition functions in each step such that M terminates after a finite number of steps by reaching the accepting state if we have $x \in L$ and

b) the machine stops for all possible choices of the transition functions in each step after a finite number of steps without reaching the accepting state q_2 if we have $x \notin L$.

A problem L can be non-deterministically decided in time $t : \mathbb{N}_0 \to \mathbb{N}_0$ and space $s : \mathbb{N}_0 \to \mathbb{N}_0$ if and only if there is a Turing Machine M that decides L and runs in time t and space s for all choices of the transition functions in each step.

With this definition we can think about the non-deterministic choices of the transition function to be guesses made by the Turing Machine. Another way is to regard those choices as a certificate or witness given by an omniscient oracle.

Definition 6.13 A decision problem $L \subseteq \{0, 1\}^*$ is said to be verifiable in time $t : \mathbb{N}_0 \to \mathbb{N}_0$ and space $s : \mathbb{N}_0 \to \mathbb{N}_0$ with certificates of length $p : \mathbb{N}_0 \to \mathbb{N}_0$ if and only if there is a deterministic Turing Machine M such that

a) M runs in time t and space s and for all $x \in \{0, 1\}^*$ and

b) we have $x \in L$ if and only if there is a certificate $u \in \{0, 1\}^{p(|x|)}$, also called witness, such that M prints 1 given x and u as inputs.

If we map the non-deterministic choices of the transition function to the bits of the certificate we can find that both definitions can decide the same languages.

Theorem 6.14 A decision problem $L \subseteq \{0,1\}^*$ can be decided in polynomial time by a non-deterministic Turing Machine if and only if it can be verified in polynomial time with certificates of polynomial length.

The complexity class of all those decision problems is called **NP** as defined in the next section. Both definitions given in Theorem 6.14 can be used to define **NP**.

6.4 Complexity Classes and Reductions

In this section we will formally define complexity classes using the running time of Turing Machines as introduced above. Since there are numerous complexity classes we only define the most common complexity classes needed in this thesis. For an exhaustive list we refer to the "Complexity Zoo" [AKG+].

Definition 6.15 We define

a) $\mathbf{DTIME}(t) := \{$problems that can be decided deterministically in time $c \cdot t$ for some $c \in \mathbb{N}_{>0}\}$ for all functions $t : \mathbb{N}_0 \to \mathbb{N}_0$

b) $\mathbf{P} := \bigcup_{f \in \mathbb{Q}[n]} \mathbf{DTIME}(f(n))$

c) $\mathbf{EXP} := \bigcup_{f \in \mathbb{Q}[n]} \mathbf{DTIME}\left(2^{f(n)}\right)$

d) $\mathbf{NTIME}(t) := \{$problems that can be decided non-deterministically in time $c{\cdot}t$ for some $c \in \mathbb{N}_{>0}\}$ for all functions $t : \mathbb{N}_0 \to \mathbb{N}_0$

e) $\mathbf{NP} := \bigcup_{f \in \mathbb{Q}[n]} \mathbf{NTIME}(f(n))$

f) $\mathbf{NEXP} := \bigcup_{f \in \mathbb{Q}[n]} \mathbf{NTIME}\left(2^{f(n)}\right)$

g) $\mathbf{coNTIME}(t) := \{$problems whose complement can be decided non-deterministically in time $c \cdot t$ for some $c \in \mathbb{N}_{>0}\}$ for all functions $t : \mathbb{N}_0 \to \mathbb{N}_0$

h) $\mathbf{coNP} := \bigcup_{f \in \mathbb{Q}[n]} \mathbf{coNTIME}(f(n))$

i) $\mathbf{coNEXP} := \bigcup_{f \in \mathbb{Q}[n]} \mathbf{coNTIME}\left(2^{f(n)}\right)$

j) **DSPACE**(s) := {problems that can be decided deterministically in space $c \cdot s$ for some $c \in \mathbb{N}_{>0}$} for all functions $s : \mathbb{N}_0 \to \mathbb{N}_0$

k) **L** := $\bigcup_{f \in \mathbb{Q}[n]}$ **DSPACE** $(f(\log n))$

l) **PSPACE** := $\bigcup_{f \in \mathbb{Q}[n]}$ **DSPACE**$(f(n))$

m) **EXPSPACE** := $\bigcup_{f \in \mathbb{Q}[n]}$ **DSPACE** $\left(2^{f(n)}\right)$

n) **NSPACE**(s) := {problems that can be decided non-deterministically in space $c \cdot s$ for some $c \in \mathbb{N}_{>0}$} for all functions $s : \mathbb{N}_0 \to \mathbb{N}_0$

o) **NL** := $\bigcup_{f \in \mathbb{Q}[n]}$ **NSPACE** $(f(\log n))$

p) **NPSPACE** := $\bigcup_{f \in \mathbb{Q}[n]}$ **NSPACE**$(f(n))$

By $c \cdot t$ for $c \in \mathbb{N}_{>0}$ and $t : \mathbb{N}_0 \to \mathbb{N}_0$ we denote the function $c \cdot t : \mathbb{N}_0 \to \mathbb{N}_0, x \mapsto c \cdot t(x)$. Similarly, we define other functions like 2^f for $f \in \mathbb{Q}[n]$ as $2^f : \mathbb{Q} \to \mathbb{Q}, x \mapsto 2^{f(x)}$. We will use similar notation without further reference.

There are many relations between the complexity classes defined above, but for many pairs of complexity classes it is still unknown whether they are equivalent. For instance, we can order deterministic time bounds and space bounds as given in the following theorem.

Theorem 6.16 Let $s : \mathbb{N}_{>0} \to \mathbb{N}_{>0}$ be a function. We have

$$\textbf{DTIME}(s) \subseteq \textbf{DSPACE}(s) \subseteq \textbf{NSPACE}(s) \subseteq \textbf{DTIME}\left(2^{O(s)}\right)$$

Proof The first relation **DTIME**$(s) \subseteq$ **DSPACE**(s) is true because a deterministic Turing Machine that works in $s(|x|)$ steps for some input $x \in \{0,1\}^*$ can only write in $s(|x|)$ cells of the tape as in each step a Turing Machine can only write in one cell. The second relation **DSPACE**$(s) \subseteq$ **NSPACE**(s) holds since non-deterministic Turing Machines are obviously able to simulate deterministic Turing Machines by using the same transition function twice.

The third relation **NSPACE**$(s) \subseteq$ **DTIME** $\left(2^{O(s)}\right)$ is fulfilled since a non-deterministic Turing Machine M that uses at most space s can have at most $2^{O(s)}$ configurations. Each cell of the tape has only two possible values zero and one. The configuration can also differ by a constant number of states of the machine and s positions of the head. All in all, this results in $2^{O(s)}$ possible configurations of M. A deterministic Turing Machine N can simulate M by finding a path in the graph of configurations of M from the initial configuration to configurations with an accepting state. This simulation can be done by a breadth-first search in time $2^{O(s)}$ which concludes the proof. For a detailed proof of this theorem we refer to the textbook by Sanjeev Arora and Boaz Barak, Theorem 4.2 [AB09]. □

With this result we can order some of the complexity classes from Definition 6.15.

Corollary 6.17

$$\mathbf{L} \subseteq \mathbf{NL} \subseteq \mathbf{P} \subseteq \mathbf{PSPACE} \subseteq \mathbf{NPSPACE} \subseteq \mathbf{EXP} \subseteq \mathbf{EXPSPACE}$$

It is also known that the complexity classes characterized by deterministic time bounds and the ones characterized by deterministic space bounds, mentioned above, are in fact different. The Space Hierarchy Theorem and Time Hierarchy Theorem were proven by Juris Hartmanis, Frederick C. Hennie, Phil M. Lewis, and Richard E. Stearns.

Theorem 6.18 (Time Hierarchy Theorem [HS65, HS66]) Let $t, t' : \mathbb{N}_0 \to \mathbb{N}_0$ be functions with $t' \in o(t)$. We have

$$\mathbf{DTIME}(t') \subsetneq \mathbf{DTIME}(t)$$

Theorem 6.19 (Space Hierarchy Theorem [SHL65]) Let $s, s' : \mathbb{N}_0 \to \mathbb{N}_0$ be functions with $s' \in o(s)$. We have

$$\mathbf{DSPACE}(s') \subsetneq \mathbf{DSPACE}(s)$$

Corollary 6.20

$$\mathbf{P} \subsetneq \mathbf{EXP}, \ \mathbf{L} \subsetneq \mathbf{PSPACE} \subsetneq \mathbf{EXPSPACE}$$

This implies that many relations in Corollary 6.17 are actual subsets and no equations. It is still not known which of those relations are equations, for instance it is unknown whether **PSPACE** and **EXP** are the same sets – and thus also whether **PSPACE** and **NPSPACE** as well as **NPSPACE** and **EXP** are the same sets.

Another important open question is the relation of deterministic and non-deterministic complexity classes. Most prominently, it is an open question whether **P** and **NP** are the same set. It is widely believed that both sets are not equal, but until today there is no proof. This question was first mentioned by Kurt Gödel in a letter to John von Neumann and was selected as one of the seven Millennium Prize Problems whose solution's author will be awarded one million US-dollars by the Clay Mathematics Institute.

The membership of problems in complexity classes gives an upper bound on the complexity of the problem. Nevertheless, there might be better algorithms than the known ones, that solve problems even faster. For instance, a problem contained in **EXP** may still be contained in **P** even if currently no algorithm is known that solves this problem in polynomial time. To give lower bounds, we usually give reductions that state that a problem is at least as hard as another problem, that is known to be hard.

Definition 6.21 (Karp reductions, **NP**-completeness) Let $L, L' \subseteq \{0,1\}^*$ be decision problems. L is polynomial-time reducible or Karp reducible to L' if and only if there is a function $f : \{0,1\}^* \to \{0,1\}^*$ that can be computed in polynomial time and for every $x \in \{0,1\}^*$ we have $x \in L \Leftrightarrow f(x) \in L'$.

A decision problem $L \subseteq \{0,1\}^*$ is said to be **NP**-hard if and only if every decision problem contained in **NP** can be Karp reduced to it. A decision problem $L \subseteq \{0,1\}^*$ is said to be **NP**-complete if and only if it is contained in **NP** and **NP**-hard. Similarly, we define hardness and completeness for more complexity classes, in particular **coNP**, **PSPACE**, **EXP**, and **EXPSPACE**.

If a decision problem $L \subseteq \{0,1\}^*$ can be Karp reduced to a decision problem $L' \subseteq \{0,1\}^*$ we can think of L' to be at least as hard as L, because to solve L we could solve problem instances of L by translating them to problem instances of L' and solving those problems. An **NP**-hard decision problem is as hard as a decision problem contained in **NP** can be, since it can be used to solve all decision problems contained in **NP**.

Note that there are other definitions of reductions for certain complexity classes. For instance, complexity classes that are contained in **P** need less powerful reduction functions as otherwise the reduction function is already able to solve the problem. Also, we can extend the definition of a reduction to so-called Levin reductions or parsimonious reductions that also transform the certificates together with the problem instances. We do not define more reductions here as they are not needed for this thesis, but refer to textbooks for details [AB09].

7 Known Complexity Results for Polynomial Ideals

7.1 Problems in Algorithmic Computer Algebra

Solving systems of polynomial equations is one of the most common problems in mathematics. As we have seen in the previous chapters, the most common way to solve problems mathematically is modeling properties of objects as polynomial equations. The varieties of the resulting polynomial ideals themselves or properties of them need to be determined. These problems turn out to be inherently hard, and it is a common technique to first compute Gröbner Bases, which have properties that make solving the problems easier. In this section we will collect interesting complexity problems, that will be solved later on. Parts of this chapter have been submitted as part of an edited volume of the DFG priority project 1489 "Algorithmic and Experimental Methods in Algebra, Geometry, and Number Theory" [MT16a].

For all problems we assume that polynomial ideals are given by a list of generators, while polynomials are given as a list of exponents and their coefficients. All lists are separated by a special character of the input alphabet. All exponents, coefficients, and other numbers are encoded in the usual way using their binary representation.

The most immediate problem is to determine all solutions of a polynomial ideal.

Definition 7.1 The variety problem is given a polynomial ideal $I \trianglelefteq R[x_1, \ldots, x_n]$ over a commutative ring R for some $n \in \mathbb{N}_{>0}$ to compute the variety $\mathcal{V}(I) \subseteq R^n$.

Unfortunately, this problem cannot be solved in general. According to the Abel-Ruffini Theorem – named after the mathematicians Niels Abel and Paolo Ruffini – solutions of polynomial equations cannot even be expressed in general using simple formulas involving roots only [Abe26], even if $n = 1$.

Theorem 7.2 (Abel-Ruffini Theorem [Abe26]) Let k be a field with $\operatorname{char}(k) = 0$ and $d \in \mathbb{N}_{>0}$ with $d \geq 5$. There is a polynomial $f \in k[x_1]$ with $\deg(f) = d$ and $y \in k$ with $f(y) = 0$ such that y is not algebraic, i.e. it cannot be expressed in radicals.

While there are closed formulas for finding the zeros of univariate polynomials of degree up to 4, univariate polynomials of degree 5 or higher may have solutions not expressible using roots and the usual field operations. Thus, there is no hope of finding algorithms that compute closed solutions of the even more general variety problem of polynomial ideals.

One way to deal with this obstacle is to try to find interesting polynomials that reveal some information about the variety contained in the polynomial ideal. For instance, if we can prove for a polynomial ideal $I \trianglelefteq \mathbb{Q}[x_1, \ldots, x_n]$ for some $n \in \mathbb{N}_{>0}$ that $x_1^2 - 1 \in I$, then we know that every point in the variety $\mathcal{V}(I)$ has 1 or -1 as the first component. To prove such hypotheses we need to solve the word problem for polynomial ideals.

Definition 7.3 (word problem) The word problem of polynomial ideals entails, given a polynomial ideal $I \trianglelefteq R[x_1, \ldots, x_n]$ over a commutative ring R for some $n \in \mathbb{N}_{>0}$ and a polynomial $f \in R[x_1, \ldots, x_n]$, to compute whether $f \in I$ or $f \notin I$.

In the case that the polynomial is contained in the polynomial ideal we can represent it as a linear combination of the generators of the polynomial ideal with polynomial coefficients. Finding those coefficients is called the representation problem.

Definition 7.4 (representation problem) The representation problem of polynomial ideals entails, given polynomials $f_1, \ldots, f_s, g \in R[x_1, \ldots, x_n]$ over a commutative ring R for some $n, s \in \mathbb{N}_{>0}$, $g \in \langle f_1, \ldots, f_s \rangle$, to compute polynomials $g_1, \ldots, g_s \in R[x_1, \ldots, x_n]$ with $g = \sum_{i=1}^{s} f_i g_i$.

As we will see, the word problem for polynomial ideals is very hard to solve. Also, when motivating the word problem we wanted to determine whether a certain polynomial holds for all solutions of a given polynomial ideals. This does not imply that this polynomial is contained in the polynomial ideal. Thus, the actual question in many applications is whether $f \in \mathcal{I}(\mathcal{V}(I))$ for some polynomial ideal $I \trianglelefteq R[x_1, \ldots, x_n]$ and a polynomial $f \in R[x_1, \ldots, x_n]$. We will see a modified version of the word problem in Definition 8.5 which is called the radical word problem.

A special case of the word problem of polynomial ideals is the triviality problem of polynomial ideals. For the triviality problem we check whether $1 \in I$ for some polynomial ideal $I \trianglelefteq k[x_1, \ldots, x_n]$ over a field k for some $n \in \mathbb{N}_{>0}$. This problem is particularly important since $1 \in I$ if and only if $I = k[x_1, \ldots, x_n]$ as discussed in Section 3.2.

Definition 7.5 (triviality problem) The triviality problem of polynomial ideals entails, given a polynomial ideal $I \trianglelefteq k[x_1, \ldots, x_n]$ over a field k for some $n \in \mathbb{N}_{>0}$, to compute whether $1 \in I$.

We can solve the word problem, the representation problem of polynomial ideals, and the triviality problem of polynomial ideals using Gröbner Bases as seen in Section 5.2. Since polynomial division modulo a Gröbner Basis as presented in Algorithm 5.11 is the way we use to solve general word problems for polynomial ideals we need to find a Gröbner Basis first.

Definition 7.6 (Gröbner Basis problem) The Gröbner Basis problem of polynomial ideals entails, given a polynomial ideal $I \trianglelefteq R[x_1, \ldots, x_n]$ over a commutative ring R for some $n \in \mathbb{N}_{>0}$ and a term ordering $\succeq$, to compute the reduced Gröbner Basis of I with respect to $\succeq$.

There are many more interesting problems on polynomial ideals like the containment problem: given two polynomial ideals $I, J \trianglelefteq R[x_1, \ldots, x_n]$ compute whether $I \subseteq J$. We will concentrate on word problems as they are the most important ones.

7.2 General Gröbner Bases

The special case of linear equations is well-understood. These systems can be transformed to the row-echelon form as discussed in Theorem 4.17. This Algorithm is called Gaußian elimination named after Carl Friedrich Gauß. Using the formulas from Theorem 4.17 we can compute the row-echelon form in polynomial time. There are more detailed discussions of the running time of the Gauß algorithm available in the literature [EGG⁺06].

In case of vector spaces of the integers the row-echelon form is called Hermite normal form. The Hermite normal form can also be computed in polynomial time [SL96]. There is also a specialized algorithm for this case called the LLL algorithm [LLL82, Bre11].

Theorem 7.7 Let $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ be linear polynomials over a field k for some $n, s \in \mathbb{N}_{>0}$. The row-echelon form of

$$\operatorname{span}(\{f_1, \ldots, f_s\}) \subseteq \{f \in k[x_1, \ldots, x_n] \mid \deg(f) \leq 1\}$$

can be computed in **P**.

Several problems, like finding solutions of the system or the word problem, can be solved easily once the system is in row-echelon form. The word problem for linear systems of equations can be solved using backwards substitution.

Theorem 7.8 Let $f_1, \dots, f_s, g \in k[x_1, \dots, x_n]$ be linear polynomials over a field k for some $n, s \in \mathbb{N}_{>0}$. The word problem to decide whether

$$g \in \operatorname{span}(\{f_1, \dots, f_s\}) \text{ or } g \notin \operatorname{span}(\{f_1, \dots, f_s\})$$

can be decided in **P**.

Proof Let $f'_1, \dots, f'_{s'} \in k[x_1, \dots, x_n]$ for some $s' \in \mathbb{N}_{>0}$ be the generators of the Hermite normal form of the subspace $V := \operatorname{span}(\{f_1, \dots, f_s\})$. We have seen in Theorem 7.7 that those polynomials can be computed in **P**. To solve the word problem, we have to find the coefficients $c_1, \dots, c_{s'} \in k$ of a linear combination $g = \sum_{i=1}^{s'} c_i f'_i$ if they exist. Each f'_i for $i \in \{1, \dots, s'\}$ was selected as the pivot to eliminate one indeterminate. Starting with the pivot containing the most indeterminates, we can fix the coefficient c_i of each polynomial f'_i for $i \in \{1, \dots, s'\}$ by comparing the coefficients in front of the indeterminate x_j, that was eliminated by f'_i, for some $j \in \{1, \dots, n\}$. The coefficients for all other pivots containing x_j have already been fixed before, so we can choose c_i such that both sides of $g = \sum_{i=1}^{s'} c_i f'_i$ have the same coefficient in front of x_j. After fixing all coefficients $c_1, \dots, c_{s'}$ we can check whether indeed $g = \sum_{i=1}^{s'} c_i f'_i$. If this equation holds we have $g \in \operatorname{span}(\{f_1, \dots, f_s\})$, otherwise we have $g \notin \operatorname{span}(\{f_1, \dots, f_s\})$. This Algorithm obviously runs in polynomial time. $\square$

We can find the actual variety of a set of linear equations similarly by going through the generators of a row-echelon form of the linear subspace and fix the coordinates of all points in the variety one-by-one. For a detailed proof we refer to a linear algebra textbook [Fis13, Lan87].

Theorem 7.9 The variety problem for systems of linear equations over a polynomial ring $k[x_1, \dots, x_n]$ for some field k and some integer $n \in \mathbb{N}_{>0}$ can be solved in **P**.

The corresponding problems for non-linear systems of equations are inherently much harder. Nevertheless, a normal form of these systems also exists, analogous to the row-echelon form used above, that allows for easier computations of many problems, namely Gröbner Bases. Note that the word problem for general polynomial ideals can be solved similarly to the linear case: First we compute a basis of the system of equations with good properties and then reduce our test polynomial with this basis. To employ this algorithm we need to compute a Gröbner Basis instead of the row-echelon form.

When Gröbner Bases were introduced in 1965 by Bruno Buchberger in his PhD thesis [Buc65], Buchberger also presented an algorithm for computing Gröbner bases using the so-called Buchberger's Criterion presented in Theorem 5.19. At first, it was only known that Buchberger's Algorithm discussed in Algorithm 5.20 runs in finite time without having better space or time bounds.

The time and space requirements of algorithms computing Gröbner Bases heavily depend on the degree of the polynomials involved. For this reason it is important to have degree bounds. The first degree bounds were already given in 1926 by Grete Hermann [Her26] and corrected by Abraham Seidenberg [Sei74] and Bodo Renschuch [Ren80]. These theorems bound the degree of polynomials in the representation problem of polynomial ideals and are double exponential in the number of indeterminates.

Theorem 7.10 ([Her26, Sei74, Ren80]) Let $f_1, \dots, f_s, g \in k[x_1, \dots, x_n]$ be polynomials over a field k with $g \in \langle f_1, \dots, f_s \rangle$ for some $n, s \in \mathbb{N}_{>0}$. Let $d \in \mathbb{N}_{>0}$ such that $\deg(f_i) \leq d$ for all $i \in \{1, \dots, s\}$ and $\deg(g) \leq d$. There are $g_1, \dots, g_s \in R[x_1, \dots, x_n]$ with

$$g = \sum_{i=1}^{s} f_i g_i \text{ and } \deg(g_i) \leq (2d)^{2^n}$$

for all $i \in \{1, \dots, s\}$.

Thomas W. Dubé [Dub90] presented another double exponential degree bound. He shows a bound for the degree of generators of a Gröbner Basis.

Theorem 7.11 ([Dub90]) Let $f_1, \dots, f_s \in R[x_1, \dots, x_n]$ be polynomials with $\deg(f_i) \leq d$ for all $i \in \{1, \dots, s\}$ over a ring R for some $d, s \in \mathbb{N}_{>0}$. Every reduced Gröbner Basis of $\langle f_1, \dots, f_s \rangle$ consists of polynomials $g_1, \dots, g_r \in R[x_1, \dots, x_n]$ for some $r \in \mathbb{N}_{>0}$ with

$$\deg(g_i) \leq 2\left(\frac{d^2}{2} + d\right)^{2^{n-1}}$$

for all $i \in \{1, \dots, r\}$.

If a polynomial of double exponential degree in the number of indeterminates appears in the Gröbner Basis of a polynomial ideal then saving that one polynomial using binary encodings of the exponents already requires an exponential amount of space. Using the degree bound found by Thomas W. Dubé, the authors Klaus Kühnle and Ernst W. Mayr showed in 1996 that Gröbner Bases can be computed using exponential space in the number of indeterminates [KM96]. Thus, the word problem for polynomial ideals is contained in **EXPSPACE**.

The word problem for polynomial ideals does not require an algorithm to save an actual Gröbner Basis in its memory, even though the best known algorithms do exactly that. Nevertheless, in 1982 Ernst W. Mayr and Albert R. Meyer proved a lower bound on the worst-case space usage of each algorithm solving the word problem for polynomial ideals that is exponential in the number of variables appearing in the equations [MM82].

Theorem 7.12 ([MM82]) There is a constant $\epsilon \in \mathbb{Q}$ with $\epsilon > 0$ such that any algorithm which is able to decide the word problem for polynomial ideals contained in $\mathbb{Q}[x_1, \ldots, x_n]$ for some $n \in \mathbb{N}_{>0}$ requires space exceeding $2^{\epsilon n}$ on infinitely many instances of this problem with different sizes.

Their result was slightly improved in 1991 by Chee K. Yap who changed the constant in the exponent [Yap91]. Thus, the lower and upper bounds for the word problem for polynomial ideals coincide and the problem is proven to be **EXPSPACE**-complete in the number of indeterminates.

Theorem 7.13 The word problem for general polynomial ideals is **EXPSPACE**-complete.

Gröbner Bases can be used to solve the word problem on polynomial ideals. Given a Gröbner Basis we just need to reduce the test polynomial on this Gröbner Basis. The test polynomial is contained in the original polynomial ideal if and only if it reduces to zero modulo the Gröbner Basis. Thus, the Gröbner Basis problem of polynomial ideals is as hard as the word problem of polynomial ideals and the word problem for polynomial ideals can also be solved using exponential space in the number of indeterminates.

Corollary 7.14 The Gröbner Basis problem for general polynomial ideals is **EXPSPACE**-complete.

Regardless of these complexity bounds on worst case examples there are algorithms that compute Gröbner Bases very efficiently in practice. The aforementioned complexity bounds imply that there are inputs for every algorithm such that the space complexity grows exponentially. For instance the F4 and F5 algorithms by Jean-Charles Faugère solve most problems appearing in practice in an efficient way [Fau99, Fau02].

The lower bound on the space requirement of the word problem for polynomial ideals also implies a lower bound on the function in the degree bound by Grete Hermann. Assume that the bound given in Theorem 7.10 was $\deg(g_i) \leq h(n, d)$ for all $i \in \{1, \ldots, s\}$ for some function $h : \mathbb{N}_{>0} \times \mathbb{N}_{>0} \to \mathbb{N}_{>0}$. The number of terms of each g_i would be at most $O(h(n, d)^n)$ and therefore the space required to save some configuration of the g_i using binary encodings of the exponents would be $O(h(n, d))$. Thus, we could solve the word problem in space $O(h(n, d))$ which implies that the bound in Theorem 7.10 is sharp.

Corollary 7.15 There is a constant $\epsilon \in \mathbb{Q}$ with $\epsilon > 0$ such that there are infinitely many problems consisting of

- polynomials $f_1, \dots, f_s, g \in k[x_1, \dots, x_n]$ over a field k with $g \in \langle f_1, \dots, f_s \rangle$ for some $n, s \in \mathbb{N}_{>0}$
- $d \in \mathbb{N}_{>0}$ with $\deg(f_i) \leq d$ for all $i \in \{1, \dots, s\}$ and $\deg(g) \leq d$

such that there are only $g_1, \dots, g_s \in R[x_1, \dots, x_n]$ with

$$g = \sum_{i=1}^{s} f_i g_i$$

that have $\deg(g_i) > 2^{\epsilon n}$ for some $i \in \{1, \dots, s\}$.

There are several surveys on further complexity results for the computation of Gröbner Bases, for instance the one presented by Ernst W. Mayr [May97].

7.3 Polynomial Ideals with Low Dimension

Since the computation of Gröbner Bases is that important and hard it is a natural question to ask whether there are special subclasses of polynomial ideals, that allow for faster computations of them. One class of polynomial ideals, that allows easier computations of their Gröbner Bases, is the set of so-called zero-dimensional polynomial ideals. Other interesting subclasses of polynomial ideals will be discussed in the following chapters. To discuss this class of polynomial ideals, we need to define the dimension of a polynomial ideal.

Definition 7.16 Let $I \trianglelefteq R[x_1, \dots, x_n]$ be a polynomial ring over a commutative ring R for some $n \in \mathbb{N}_{>0}$. The dimension $\dim(I)$ of I is the maximum size of a set of indeterminates $X \subseteq \{x_1, \dots, x_n\}$ such that no leading monomial in $\mathrm{LM}(I)$ consists of these indeterminates only.

Equivalently, we can define the dimension of a polynomial ideal I as the size of the biggest set of indeterminates that is unrelated modulo the polynomial ideal I. This means, that zero-dimensional ideals have many relations between their indeterminates and that there even is a relation for each indeterminate on its own. This additional structure may be used to find improved degree bounds.

In 1983 Jean-Charles Faugère et al. presented an algorithm that is much faster in practice for zero-dimensional polynomial ideals than Buchberger's Algorithm [FGLM93]. It was also proven, that there is a single-exponential bound on the degree of Gröbner Basis elements for zero-dimensional polynomial ideals by Dickenstein et al. which enables better algorithms [DFGS91].

Theorem 7.17 ([DFGS91]) Let $f_1, \dots, f_s \in k[x_1, \dots, x_n]$ be polynomials with $\deg(f_i) \leq d$ for all $i \in \{1, \dots, s\}$ over a field k for $d, n, s \in \mathbb{N}_{>0}$ such that $\langle f_1, \dots, f_s \rangle$ has dimension zero and let $g \in \langle f_1, \dots, f_s \rangle$ be a polynomial. There are polynomials

$$g_1, \dots, g_s \in k[x_1, \dots, x_n]$$

such that $g = \sum_{i=1}^{s} f_i g_i$ and

$$\deg(f_i g_i) \leq nd^{2n} + d^n + d + \deg(f)$$

for all $i \in \{1, \dots, s\}$.

Since the special case of zero-dimensional polynomial ideals is much easier than the general problem one could expect polynomial ideals with low dimension to allow for better algorithms too.

All degree bounds given above are dependent on the number of indeterminates, as this turned out to be a very significant parameter of polynomial ideals to describe their inherent complexity. The following bound does not use the number of indeterminates as a parameter, but the degree of the polynomial ideal, which may result in better bounds for special subsets of polynomial ideals.

Using a new degree bound by Matthias Kratzer [Kra08], the authors Ernst W. Mayr and Stephan Ritscher were able to find an algorithm to compute Gröbner Bases, whose space is bounded exponentially in the dimension of the polynomial ideal [MR11].

Theorem 7.18 ([MR11]) Let $f_1, \dots, f_s \in k[x_1, \dots, x_n]$ be polynomials with $\deg(f_i) \leq d$ for all $i \in \{1, \dots, s\}$ over an infinite field k for $d, n, s \in \mathbb{N}_{>0}$. Let $m \in \mathbb{N}_0$ be the dimension of $\langle f_1, \dots, f_s \rangle$. Every reduced Gröbner Basis of $\langle f_1, \dots, f_s \rangle$ with respect to an admissible monomial ordering consists of polynomials $g_1, \dots, g_r \in R[x_1, \dots, x_n]$ for some $r \in \mathbb{N}_{>0}$ with

$$\deg(g_i) \leq 2\left(\frac{1}{2}\left(d^{2(n-m)^2} + d\right)\right)^{2^m}$$

for all $i \in \{1, \dots, r\}$.

This theorem is proven using an algorithm based on a cone decomposition of the space of polynomials. The construction of this decomposition is based on a similar decomposition presented by Thomas W. Dubé [Dub90].

Mayr and Ritscher also presented an incremental version of their algorithm, that does not use degree bounds. The space bound of this algorithm uses the degree of the actual problem instance instead of a worst-case instance. Therefore, the algorithm does not require the knowledge of any a priori degree bounds.

Both findings improved the known space bounds for polynomial ideals with low degree in comparison to the general bounds. Later, they also proved a matching lower bound [MR13], which finished the complexity analysis of computing Gröbner Bases depending on the degree of the polynomial ideal.

Part IV

Subclasses of Polynomial Ideals

8 Radical Ideals

8.1 Roots of Polynomials

As we have seen in Chapter 7, the complexity of many problems of polynomial ideals is very high. In the following chapters we will explore subclasses of polynomial ideals and discuss specialized algorithms that may allow for better results on these subclasses. We will start with radical ideals in this chapter.

In Section 4.5 we have defined varieties and vanishing ideals. We have seen there, that not for all sets $V \subseteq R^n$ for some ring R and $n \in \mathbb{N}_{>0}$ we have $V = \mathcal{V}(\mathcal{I}(V))$. We identified the sets with this property as the sets closed under the Zariski topology. It is a natural question whether we also have $I = \mathcal{I}(\mathcal{V}(I))$ for all polynomial ideals $I \trianglelefteq R[x_1, \ldots, x_n]$. We will see, that this is only the case for radical ideals which will be discussed in this chapter.

For an easy example of a polynomial ideal without the property given above, consider the polynomial ideal $I := \left\langle x^2 \right\rangle \subseteq \mathbb{Q}[x]$. We clearly have $\mathcal{V}(I) = \{0\}$ as zero is the only number whose square is zero. To compute $\mathcal{I}(\mathcal{V}(I))$ we have to find all polynomials contained in $\mathbb{Q}[x]$ that vanish at 0. The generator x^2 clearly does this, but there are more polynomial with that property: All polynomials that are a multiple of x vanish at zero. Thus, we have $\mathcal{I}(\mathcal{V}(I)) = \langle x \rangle$. The polynomial ideal $\langle x \rangle$ is a proper superset of $\left\langle x^2 \right\rangle$ as for instance x is not contained in $\left\langle x^2 \right\rangle$. Another interesting example over a multivariate polynomial ring is

$$I := \left\langle x_1^2 - x_1x_2, x_2^2 - x_1x_2 \right\rangle \trianglelefteq \mathbb{Q}[x_1, x_2]$$

For this ideal we have $\mathcal{V}(I) = \{(a, a) \mid a \in \mathbb{Q}\}$ and thus $\mathcal{I}(\mathcal{V}(I)) = \langle x_1 - x_2 \rangle$. It is interesting to note that the square of the generator of $\mathcal{I}(\mathcal{V}(I))$ is contained in I again as

$$(x_1 - x_2)^2 = x_1^2 - 2x_1x_2 + x_2^2 = (x_1^2 - x_1x_2) + (x_2^2 - x_1x_2)$$

It turns out that this is true for all polynomial ideals which motivates the definition of the radical of a polynomial ideal.

Definition 8.1 Let R be a ring. For each polynomial ideal $I \trianglelefteq R$ we define the radical of I to be

$$\sqrt{I} := \{f \in R \mid f^s \in I \text{ for some } s \in \mathbb{N}_{>0}\}$$

If $I = \sqrt{I}$ we call I a radical ideal.

We collect some easy properties of the radical first.

Theorem 8.2 Let R be an integral domain and $I \trianglelefteq R$ be an ideal. We have

$$I \subseteq \sqrt{I} \subseteq \mathcal{I}(\mathcal{V}(I))$$

and $\sqrt{I} \trianglelefteq R$ is also an ideal.

Proof The containment relation $I \subseteq \sqrt{I}$ is clear since in the definition

$$\sqrt{I} := \{f \in R \mid f^s \in I \text{ for some } s \in \mathbb{N}_{>0}\}$$

we can always choose $s = 1$. This also implies that $\sqrt{I}$ is not empty, since – by definition – I is not the empty set. For the relation $\sqrt{I} \subseteq \mathcal{I}(\mathcal{V}(I))$ assume that $f \in \sqrt{I}$. Therefore, there is an $s \in \mathbb{N}_{>0}$ such that $f^s \in I$. For each $y \in \mathcal{V}(I)$ we have that $f^s(y) = 0$ by the definition of the variety. If $f(y) \neq 0$ we have $f^{s-1}(y) = 0$, because $f(y)f^{s-1}(y) = 0$ and R is a domain. We can repeat this argument with decrementing exponent until we finally get $f(y) = 0$ and thus $\sqrt{I} \subseteq \mathcal{I}(\mathcal{V}(I))$.

According to the definition of ideals in Definition 3.15, we have to prove two more properties. First, if $f \in \sqrt{I}$ and $g \in R$ there is some $s \in \mathbb{N}_{>0}$ with $f^s \in I$ which implies $(fg)^s = (gf)^s = g^s f^s \in I$ and thus $fg, gf \in \sqrt{I}$. Second, if $f, g \in \sqrt{I}$ there are $s, t \in \mathbb{N}_{>0}$ with $f^s, g^t \in I$ which implies

$$(f+g)^{s+t} = \sum_{i=0}^{s+t} \binom{s+t}{i} f^i g^{s+t-i} \in I$$

and thus $f + g \in \sqrt{I}$. □

Over certain coefficient rings we actually have $\sqrt{I} = \mathcal{I}(\mathcal{V}(I))$, but this is not as easy to prove. This relation was discovered by the German mathematician David Hilbert in 1893 [Hil93] and is called “Hilbert’s Nullstellensatz” in honor of him.

Theorem 8.3 (Hilbert’s Nullstellensatz [Hil93]) Let k be an algebraically closed field, let $n \in \mathbb{N}_{>0}$, and let $I \trianglelefteq k[x_1, \dots, x_n]$ be a polynomial ideal. We have

$$\sqrt{I} = \mathcal{I}(\mathcal{V}(I))$$

The Nullstellensatz allows for computations on $\mathcal{I}(\mathcal{V}(I))$ without actually computing $\mathcal{V}(I)$. This is an important, fundamental finding since the Abel-Ruffini Theorem presented in Theorem 7.2 implies that there is no way to express $\mathcal{V}(I)$ explicitly in general. On the other hand,

$\sqrt{I}$ can be defined and symbolically computed without problems. Thus, Hilbert's Nullstellensatz enables us to compute $\mathcal{I}(\mathcal{V}(I))$ itself and to compute whether polynomials are contained in $\mathcal{I}(\mathcal{V}(I))$.

A proof of Hilbert's Nullstellensatz can be done using an interesting technique called the Rabinowitsch trick. This technique was presented first in a 13-line paper in 1930 [Rab30]. The author's name Rabinowitsch is a pseudonym of the American mathematician George Yuri Rainich, who was born with the name Rabinowitsch in Russia and later emigrated to the United States of America. There is an interesting anecdote about his pseudonym that can be found in the editor's end notes of *The American Mathematical Monthly* [Pal04a, Pal04b]. To use his proof technique, we need the so-called Weak Nullstellensatz which is proven in many algebra textbooks [CLO07].

Theorem 8.4 (Weak Nullstellensatz) Let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a polynomial ideal over an algebraically closed field k for some $n \in \mathbb{N}_{>0}$. If $\mathcal{V}(I) = \emptyset$ then we have $I = k[x_1, \ldots, x_n]$.

Proof (of Theorem 8.3) The direction $\subseteq$ was already shown in Theorem 8.2. For the other direction let $f \in \mathcal{I}(\mathcal{V}(I)) \trianglelefteq k[x_1, \ldots, x_n]$. We will show that $f \in \sqrt{I}$.

Let $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ be polynomials such that $\langle f_1, \ldots, f_s \rangle = I$ for some $s \in \mathbb{N}_{>0}$. Such polynomials exist due to the Hilbert Basis Theorem presented in Theorem 4.11. We consider the polynomial ideal

$$J := \langle f_1, \ldots, f_s, 1 - x_{n+1} f \rangle \trianglelefteq k[x_1, \ldots, x_{n+1}]$$

in one additional indeterminate.

We will now show that $\mathcal{V}(J) = \emptyset$. If there was $(a, b) \in \mathcal{V}(J)$ for some $a \in k^n$, $b \in k$ we also had $a \in \mathcal{V}(I)$. This would imply $f(a) = 0$ since $f \in \mathcal{I}(\mathcal{V}(I))$ and therefore

$$(1 - x_{n+1} f)(a, b) = 1 - b \cdot 0 = 1$$

which is a contradiction to $(a, b) \in \mathcal{V}(J)$. Thus, the variety $\mathcal{V}(J)$ is empty and together with the Weak Nullstellensatz from Theorem 8.4 we get $J = k[x_1, \ldots, x_{n+1}]$.

If J is the complete polynomial ring we in particular have $1 \in J$ and thus there are polynomials $g_1, \ldots, g_{s+1} \in k[x_1, \ldots, x_{n+1}]$ such that

$$1 = \sum_{i=1}^{s} g_i f_i + g_{s+1}(1 - x_{n+1} f)$$

If $f = 0$ the claim $f \in \sqrt{I}$ is obvious, otherwise we pass over to the field of fractions and substitute x_{n+1} by $\frac{1}{f}$. Doing so we get

$$1 = \sum_{i=1}^{s} g_i \left(x_1, \ldots, x_n, \frac{1}{f} \right) f_i(x_1, \ldots, x_n) \in \operatorname{Quot}(k[x_1, \ldots, x_n])$$

where we indicated the indeterminates of each polynomial for clarity. Note that the term with coefficient g_{s+1} cancels. Let $e \in \mathbb{N}_{>0}$ be the highest exponent of x_{n+1} appearing in any of the original $g_1, \ldots, g_s \in k[x_1, \ldots, x_{n+1}]$. After multiplying the equation with f^e we get

$$f^e = \sum_{i=1}^{s} h_i f_i \in k[x_1, \ldots, x_n]$$

for some $h_1, \ldots, h_s \in k[x_1, \ldots, x_n]$. As the right-hand side of this equation is contained in I, we also get $f^e \in I$ and thus $f \in \sqrt{I}$, which concludes the proof. □

We will use the Rabinowitsch trick in the next section to prove some degree bounds on radical polynomial ideals.

8.2 Degree Bounds for Radical Ideals

In this section we will discuss word problems for radicals of polynomial ideals, their complexity, and why those problems are important. When motivating the word problem in Definition 7.3, we wanted to find whether a certain polynomial holds for all solutions of a given polynomial ideals. This does not imply, that this polynomial is contained in the polynomial ideal. Instead, Hilbert's Nullstellensatz shows that over algebraically closed fields we have to check whether the polynomial is contained in the radical of the polynomial ideal. Therefore, we could relax the word problem to include more polynomials.

Definition 8.5 (radical word problem) The radical word problem of polynomial ideals is, given a polynomial ideal $I \trianglelefteq R[x_1, \ldots, x_n]$ over a commutative ring R for some $n \in \mathbb{N}_{>0}$ and a polynomial $f \in R[x_1, \ldots, x_n]$, to compute whether $f \in \sqrt{I}$ or $f \notin \sqrt{I}$.

We can also split up the radical word problem into two subproblems: finding $\mathcal{I}(\mathcal{V}(I))$ and then solving the word problem on $\mathcal{I}(\mathcal{V}(I))$. The first part of this computation is called the radical problem.

Definition 8.6 (radical problem) The radical problem of polynomial ideals is, given a polynomial ideal $I \trianglelefteq R[x_1, \ldots, x_n]$ over a commutative ring R for some $n \in \mathbb{N}_{>0}$, to compute a generating set of the polynomial ideal $\sqrt{I}$.

The radical word problem for polynomial ideals seems more complicated than the corresponding word problem since no basis of the radical ideal is given. Surprisingly, the radical word problem is in general easier to solve than the word problem for polynomial ideals. Recall that for general polynomial ideals the word problem is **EXPSPACE**-complete and the degrees appearing in the representation problem are in the worst case double exponential. It turns out that the same degree bound for radical ideals is only single exponential.

In 1987 W. Dale Brownawell showed a degree bound on the coefficients appearing in the representation problem of the constant polynomial 1 [Bro87].

Theorem 8.7 ([Bro87]) Let k be a field with $\operatorname{char}(k) = 0$ and let $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ be polynomials with $\mathcal{V}(\langle f_1, \ldots, f_s \rangle) = \emptyset$ for some $n, s \in \mathbb{N}_{>0}$. Furthermore, let $d \in \mathbb{N}_{>0}$ with $\deg(f_i) \leq d$ for all $i \in \{1, \ldots, s\}$ and $\mu := \min(n, s) \in \mathbb{N}_{>0}$. There are polynomials $g_1, \ldots, g_s \in k[x_1, \ldots, x_n]$ with

$$1 = \sum_{i=1}^{s} g_i f_i \text{ and } \deg(g_i) \leq \mu n d^{\mu} + \mu d$$

for all $i \in \{1, \ldots, s\}$.

Degree bounds on the representation problem of one can be used to find bounds for radical ideals using the Rabinowitsch trick. We can use the bound given above to find a bound for each polynomial contained in the radical of a polynomial ideal on the exponent of the polynomial such that it is contained in the original polynomial ideal.

Corollary 8.8 ([Bro87]) Let k be a field with $\operatorname{char}(k) = 0$ and let $f_1, \ldots, f_s, g \in k[x_1, \ldots, x_n]$ be polynomials with $g \in \sqrt{\langle f_1, \ldots, f_s \rangle}$ for some $n, s \in \mathbb{N}_{>0}$. Additionally, let $d \in \mathbb{N}_0$ with $\deg(f_i) \leq d$ for all $i \in \{1, \ldots, s\}$ and $\mu := \min(n, s) \in \mathbb{N}_{>0}$. For

$$e := (\mu + 1)(n + 1)(d + 1)^{\mu+1} + (\mu + 1)(d + 1) \in \mathbb{N}_{>0}$$

we have $g^e \in I$ and there are $h_1, \ldots, h_s \in k[x_1, \ldots, x_n]$ for some $s \in \mathbb{N}_{>0}$ with

$$g^e = \sum_{i=1}^{s} h_i f_i$$

and

$$\deg(h_i) \leq e(d + 1) = \Big((\mu + 1)(n + 1)(d + 1)^{\mu+1} + (\mu + 1)(d + 1)\Big)(d + 1)$$

for all $i \in \{1, \ldots, s\}$.

Proof We will use the same notation as in the proof of Hilbert's Nullstellensatz in Theorem 8.3. In that proof we showed that there are $g_1, \ldots, g_{s+1} \in k[x_1, \ldots, x_{n+1}]$ with

$$1 = \sum_{i=1}^{s} g_i f_i + g_{s_1} (1 - x_{n+1} f)$$

With Theorem 8.7 we can choose those polynomials in a way such that $\deg(g_i) \leq e$ for all $i \in \{1, \ldots, s+1\}$. After passing to the field of fractions and substituting x_{n+1} by $\frac{1}{f}$ we get

$$1 = \sum_{i=1}^{s} g_i \left(x_1, \ldots, x_n, \frac{1}{f}\right) f_i (x_1, \ldots, x_n) \in \mathrm{Quot}(k[x_1, \ldots, x_n])$$

as explained in the proof of Theorem 8.3. Multiplying this equation by g^e results in the equation to be shown. □

János Kollár improved the bound by Brownawell one year later [Kol88]. His proof also works if $\mathrm{char}(k) > 0$, but requires the additional constraint that $n \geq 3$. This constraint is no obstacle for us since we can always add new indeterminates that are not used.

Theorem 8.9 ([Kol88]) Let k be a field and let $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ be polynomials with $\mathcal{V}(\langle f_1, \ldots, f_s \rangle) = \emptyset$ for some $n, s \in \mathbb{N}_{>0}$ with $d_i := \deg(f_i) \in \mathbb{N}_0$ for all $i \in \{1, \ldots, s\}$. Without loss of generality we assume that $d_1 \leq d_2 \leq \cdots \leq d_s$. There are $g_1, \ldots, g_s \in k[x_1, \ldots, x_n]$ with

$$1 = \sum_{i=1}^{s} g_i f_i \text{ and } \deg(g_i) \leq \max(3, d_s) \prod_{i=1}^{\min(n,s)-1} \max(3, d_i)$$

for all $i \in \{1, \ldots, s\}$.

Again using the Rabinowitsch trick similarly as above we can find bounds on the representation problem for radical ideals.

Corollary 8.10 ([Kol88]) Let k be a field and let $f_1, \ldots, f_s, g \in k[x_1, \ldots, x_n]$ be polynomials with $g \in \sqrt{\langle f_1, \ldots, f_s \rangle}$ for some $n, s \in \mathbb{N}_{>0}$ with $d_i := \deg(f_i) \in \mathbb{N}_0$ for all $i \in \{1, \ldots, s\}$. Without loss of generality we assume that $d_1 \leq d_2 \leq \cdots \leq d_s$. For

$$e := \max(3, d_s) \prod_{i=1}^{\min(n,s)-1} \max(3, d_i) \in \mathbb{N}_{>0}$$

we have $g^e \in I$ and there are $h_1, \ldots, h_s \in k[x_1, \ldots, x_n]$ for some $s \in \mathbb{N}_{>0}$ with

$$g^e = \sum_{i=1}^{s} h_i f_i$$

and

$$\deg(h_i) \leq e(\deg(g)+1) = \left(\max(3,d_s) \prod_{i=1}^{\min(n,s)-1} \max(3,d_i)\right)(\deg(g)+1)$$

for all $i \in \{1,\ldots,s\}$.

Those bounds have been simplified and slightly improved for small degrees by Martín Sombra in 1999 [Som99].

Theorem 8.11 ([Som99]) Let k be a field and let $f_1,\ldots,f_s \in k[x_1,\ldots,x_n]$ be polynomials with $\mathcal{V}(\langle f_1,\ldots,f_s\rangle) = \emptyset$ for some $n,s \in \mathbb{N}_{>0}$ with $d_i := \deg(f_i) \in \mathbb{N}_0$ for all $i \in \{1,\ldots,s\}$. Without loss of generality we assume that $d_1 \leq d_2 \leq \cdots \leq d_s$. There are polynomials $g_1,\ldots,g_s \in k[x_1,\ldots,x_n]$ with

$$1 = \sum_{i=1}^{s} g_i f_i \text{ and } \deg(g_i) \leq 2d_s \prod_{i=1}^{\min(n,s)-1} d_i$$

for all $i \in \{1,\ldots,s\}$.

Corollary 8.12 ([Som99]) Let k be a field and let $f_1,\ldots,f_s,g \in k[x_1,\ldots,x_n]$ be polynomials with $g \in \sqrt{\langle f_1,\ldots,f_s\rangle}$ for some $n,s \in \mathbb{N}_{>0}$ with $d_i := \deg(f_i) \in \mathbb{N}_0$ for all $i \in \{1,\ldots,s\}$. Without loss of generality we assume that $d_1 \leq d_2 \leq \cdots \leq d_s$. For

$$e := 2d_s \prod_{i=1}^{\min(n,s)-1} d_i \in \mathbb{N}_{>0}$$

we have $g^e \in I$ and there are $h_1,\ldots,h_s \in k[x_1,\ldots,x_n]$ for some $s \in \mathbb{N}_{>0}$ with

$$g^e = \sum_{i=1}^{s} h_i f_i$$

and

$$\deg(h_i) \leq e(\deg(g)+1) = \left(2d_s \prod_{i=1}^{\min(n,s)-1} d_i\right)(\deg(g)+1)$$

for all $i \in \{1,\ldots,s\}$.

As for the word problem for general polynomial ideals we can deduce a complexity bound on the radical word problem from these degree bounds. As the degrees have only exponential size we can save the coefficients in polynomial space. Thus, we can also check all possible coefficients in polynomial space. We will see an example that the degree bounds given above are asymptotically tight in Section 14.2.

Theorem 8.13 Let k be a field. The radical word problem of polynomial ideals contained in the polynomial ring $k[x_1, \ldots, x_n]$ for $n \in \mathbb{N}_{>0}$ is contained in **PSPACE**.

We can also use the degree bounds given above to solve the triviality problem of polynomial ideals as presented in Definition 7.5. As we have single exponential bounds on the degree of coefficients appearing in the representation of 1, we can check them all in polynomial space.

Theorem 8.14 Let k be a field. The triviality word problem of polynomial ideals contained in the polynomial ring $k[x_1, \ldots, x_n]$ for $n \in \mathbb{N}_{>0}$ is contained in **PSPACE**.

Hilbert's Nullstellensatz together with the degree bounds given above is also called effective Nullstellensatz as it enables an exhaustive search for solving the radical word problem.

8.3 Computation of Radical Ideals

With the results above we also need at least polynomial space to solve the radical problem of polynomial ideals as this problem is more general than the radical word problem. The best known algorithm for solving the radical problem still takes exponential space. To find such an algorithm we consider a theorem presented by Abraham Seidenberg in 1974 [Sei74].

Theorem 8.15 (Seidenberg's Theorem [Sei74]) Let k be a perfect field, $n, s \in \mathbb{N}_{>0}$, and let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a polynomial ideal with $\dim(I) = 0$. For all $i \in \{1, \ldots, n\}$ let $g_i \in k[x_1, \ldots, x_n]$ be the square-free part of the generator of $I \cap k[x_i]$. Then we have

$$\sqrt{I} = I + \langle g_1, \ldots, g_n \rangle$$

A perfect field k in this context is a field such that every irreducible polynomial over k, i.e. every non-constant polynomial over k, that cannot be factored into the product of two non-constant polynomials, has distinct roots. In particular all finite fields, all algebraically closed fields, and all fields with $\mathrm{char}(k) = 0$ are perfect. Thus, almost all fields that appear in usual applications are perfect.

It is important to note that for all $i \in \{1, \ldots, n\}$ the ideal $I \cap k[x_i]$ is non-zero, univariate and therefore a principal ideal, because $\dim(I) = 0$. Thus, we can speak of “the” generator of $I \cap k[x_i]$. A square-free polynomial is a polynomial that is not a multiple of the square of a polynomial with degree greater than zero. Naturally, the square-free part of a polynomial is defined to be a square-free divisor of that polynomial with maximum degree. The square-free part of a polynomial has at most the degree of the polynomial itself.

Computing the square-free part of a univariate polynomial is computationally easy and can be done using Algorithm 8.16.

Algorithm 8.16 Compute the square-free part of a univariate polynomial.

```
Input: a polynomial $f \in k[x]$ over a field $k$ and $a_0, \ldots, a_r \in k$ with $f = \sum_{i=0}^{r} a_i x^i$ for some $r \in \mathbb{N}_0$
Output: the square-free part of $f$
1: compute the formal derivative $f'$ of $f$ as $f' := \sum_{i=0}^{r-1} (i + 1) a_{i+1} r^i$
2: compute $g := \gcd(f, f')$ using the Euclidean Algorithm as presented in Algorithm 5.7
3: return $\frac{f}{g}$
```

Since the Euclidean Algorithm has a linear running time in the input size the total running time of Algorithm 8.16 is also linear in the input size. The algorithm is true since factors appearing multiple times in a factorization of f would also appear as a divisor of the formal derivative f'. By dividing through the common divisors of f and f' we guarantee that the result is square-free.

We can now give an algorithm to compute the radical of a zero-dimensional polynomial ideal over a perfect field.

Algorithm 8.17 Compute the radical of a zero-dimensional polynomial ideal.

```
Input: $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ polynomials over a perfect field $k$ for some $n, s \in \mathbb{N}_0$ such that $\dim(\langle f_1, \ldots, f_s \rangle) = 0$
Output: $\sqrt{f_1, \ldots, f_s}$
1: for $i \in \{1, \ldots, n\}$ do
2:     compute a generator $h_i \in k[x_i]$ of $\langle f_1, \ldots, f_s \rangle \cap k[x_i]$ using Theorem 5.21 and Algorithm 5.20
3:     compute the square-free part $g_i \in k[x_i]$ of $h_i$ using Algorithm 8.16
4: end for
5: return $\langle f_1, \ldots, f_s, g_1, \ldots, g_n \rangle$
```

The condition $\dim(I) = 0$ can be removed from Theorem 8.15 to make the theorem more general. If $\dim(I) > 0$ there is a maximally independent set of $\dim(I)$ indeterminates x_i modulo I. Without loss of generality let those indeterminates be $x_1, \ldots, x_{\dim(I)}$. We can consider those indeterminates as part of the coefficients. In particular we can apply Theorem 8.15 to $I \trianglelefteq k'[x_{\dim(I)+1}, \ldots, x_n]$ where $k' := k(x_1, \ldots, x_{\dim(I)})$ is the quotient field of the polynomial ring $k[x_1, \ldots, x_{\dim(I)}]$. This theorem is applicable because if k has characteristic zero then k and k' are perfect. The ideal $I \trianglelefteq k'[x_{\dim(I)+1}, \ldots, x_n]$ is zero-dimensional even if I had a positive dimension and thus we can use Theorem 8.15 and Algorithm 8.17.

When moving the resulting ideal back to $k\,[x_1, \ldots, x_n]$ it is not enough to just contract the generators of the radical ideal in $k(Y)[\{x_1, \ldots, x_n\} \setminus Y]$ to $k\,[x_1, \ldots, x_n]$ by removing denominators. For instance, $\left\langle x^2 \right\rangle \trianglelefteq \mathbb{C}(x)$ is the full quotient field and therefore radical, but $\left\langle x^2 \right\rangle \trianglelefteq \mathbb{C}[x]$ is not radical. Also, extending ideals to a bigger polynomial ring and contracting them back to the original ring may add new elements instead of returning to the same ideal again. For another example consider $\langle x_1 x_2 \rangle \trianglelefteq \mathbb{C}[x_1, x_2]$. This ideal does not contain x_2, but extending the ideal to $\mathbb{C}(x_1)[x_2]$ and contracting it back to $\mathbb{C}[x_1, x_2]$ adds the monomial x_2 to the ideal since in $\mathbb{C}(x_1)[x_2]$ we can multiply by x_1^{-1}.

We present the generalized algorithm here and refer to the book by Becker and Weispfenning for a proof of the algorithm and instructions on how to implement its steps [BWK93, Chapter 8].

Algorithm 8.18 Compute the radical of a polynomial ideal.

Input: $f_1, \ldots, f_s \in k\,[x_1, \ldots, x_n]$ polynomials over a perfect field k for some $n, s \in \mathbb{N}_0$
Output: $\sqrt{f_1, \ldots, f_s}$

```
1: if $1 \in \langle f_1, \ldots, f_s \rangle$ then
2:     return $\langle 1 \rangle$
3: else
4:     compute a maximally independent set $Y$ of the indeterminates $\{x_1, \ldots, x_n\}$ modulo $\langle f_1, \ldots, f_s \rangle$
5:     compute the radical $I$ of $\langle f_1, \ldots, f_s \rangle \trianglelefteq k(Y)[\{x_1, \ldots, x_n\} \setminus Y]$ using Algorithm 8.17
6:     compute the contraction $J$ of $I$ to $k\,[x_1, \ldots, x_n]$ by multiplying each generator by the least common multiple of the denominators in its coefficients and saturating the resulting ideal at the least common multiple of the coefficients of all leading terms of the generators using Algorithm 4.39
7:     compute the radical $K$ of $\langle f_1, \ldots, f_s \rangle : J^\infty \trianglelefteq k\,[x_1, \ldots, x_n]$ by using Algorithm 4.39 and a recursive call
8:     return $J \cap K$
9: end if
```

The computation of the ideal operations is done using Gröbner Bases. Therefore, Algorithm 8.18 still takes at least exponential space and double exponential time in the worst case.

Santiago Laplagne presented an optimized version of this algorithm in 2006 [Lap06]. His Algorithm avoids components that are not needed and therefore is much faster for many practical examples. The worst case bounds of Laplagne's Algorithm are still double exponential.

Theorem 8.19 ([Lap06]) There is an algorithm that computes the generators of the radical of a polynomial ideals $I \trianglelefteq k[x_1, \ldots, x_n]$ over a perfect field k for some $n \in \mathbb{N}_{>0}$ given by a set of generators $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ for some $s \in \mathbb{N}_{>0}$ with $I = \langle f_1, \ldots, f_s \rangle$ and $\deg(f_i) \leq d$ for all $i \in \{1, \ldots, s\}$ for some $d \in \mathbb{N}_{>0}$ in time

$$(sd)^{2^{n(cn)+n-1}}$$

for a constant $c \in \mathbb{N}_{>0}$.

There are many more algorithms for computing radicals of polynomial ideals, but all known algorithms have a double exponential worst case running time in contrast to the radical word problem that can be solved in single exponential running time. We will see a specialized algorithm for computing radicals of binomial ideals in Section 11.1 and we will present a new algorithm for commutative Thue systems in Section 14.3.

All these algorithms are designed for polynomial ideals over fields with characteristic 0. There are also algorithms for polynomial ideals over fields with positive characteristic, for instance by Gregor Kemper [Kem02] and Ryutaroh Matsumoto [Mat01].

9 Binomial Ideals

9.1 Definition and Properties of Binomial Ideals

In this section we will introduce a subclass of polynomial ideals, that has more structure than general polynomial ideals, but still carries the same inherent complexity. This makes binomial ideals interesting objects to study. Also, binomial ideals occur in applications with commutative semigroups, commutative algebra, and algebraic statistics. In particular, we will discuss the relation between binomial ideals and term replacement systems in Section 12.2.

We will define binomials and binomial ideals first. Binomial ideals were examined in a paper of David Eisenbud and Bernd Sturmfels in a fundamental paper from 1996 [ES96].

Theorem 9.1 A binomial is a polynomial having at most two terms. A pure binomial is a polynomial having exactly two terms and coefficients 1 and -1, respectively. A polynomial ideal is called a binomial ideal or pure binomial ideal if there is a generating set of this ideal consisting only of binomials or pure binomials, respectively.

Note that we used only the requirement, that there is a generating set consisting of binomials only, to define binomial ideals. This means that binomial ideals can contain non-binomials and can also be generated by non-binomials in general. That is because sums and products of binomials are no binomials anymore in general.

Example 9.2 For instance, the polynomial ring $\mathbb{Q}[x_1, x_2]$ contains just one polynomial ideal that consists of binomials only, namely $\langle 0 \rangle = \{0\}$, and no polynomial ideal that consists of pure binomials only. Also, the polynomial ideal

$$I = \left\langle x_1 x_2^5 - x_1^3 x_2^2 - 6x_2^4, 2x_2^4 - 2x_1^2 x_2 - 3x_1 \right\rangle \trianglelefteq \mathbb{Q}[x_1, x_2]$$

does not seem to be a binomial ideal at first sight. Nevertheless, we also have

$$I = \left\langle x_1^2 x_2 + 2x_1, 2x_2^4 + x_1, 4x_1 x_2^3 - x_1^3, x_1^4 + 8x_1 x_2^2 \right\rangle$$

which proves that I is in fact a binomial ideal. The second set is in fact a Gröbner Basis of I with respect to a lexicographic term ordering.

We saw in this example, that – given a set of generators – one cannot directly see whether a polynomial ideal is a binomial ideal. Thus, we need an algorithm to test whether a polynomial ideal is a binomial ideal. We will see that we can read of whether a polynomial ideal is a binomial ideal from a reduced Gröbner Basis of the polynomial ideal. To prove this, we first state an observation about binomials and Buchberger's Algorithm.

Theorem 9.3 Let $f, g \in k[x_1, \ldots, x_n]$ be binomials over a field k for some $n \in \mathbb{N}_{>0}$. For every term ordering $\geq$ on $k[x_1, \ldots, x_n]$ we have that $\operatorname{spol}(f, g)$ is a binomial again. If f and g are pure binomials $\operatorname{spol}(f, g)$, is a pure binomial too, for every term ordering $\geq$ on $k[x_1, \ldots, x_n]$.

With this theorem we can prove that reduced Gröbner Bases of a binomial ideal consist of binomials only.

Theorem 9.4 Let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a binomial ideal over a coefficient field k for some $n \in \mathbb{N}_{>0}$ and let $\geq$ be a term ordering on $k[x_1, \ldots, x_n]$. The reduced Gröbner Basis of I with respect to $\geq$ consists of binomials only. Furthermore, if I is a pure binomial ideal the reduced Gröbner Basis of I with respect to $\geq$ consists of pure binomials only.

Proof For the first part of the theorem let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a binomial ideal. From the definition of binomial ideals we know that there is a generating set of I consisting of binomials only. Using Buchberger's algorithm as presented in Algorithm 5.20, we can compute a reduced Gröbner Basis of I by adding s-polynomials of polynomials in the generating set and removing elements. With Theorem 9.3 we know that this generating set needs to consist of binomials only too. From Theorem 5.16 we know that the reduced Gröbner Basis of I with respect to $\geq$ is unique, which proves the claim of the theorem.

The case of pure binomial ideals follows similarly. □

Checking whether a polynomial ideal is a binomial ideal can therefore be done by computing a reduced Gröbner Basis with respect to some arbitrary term ordering and then checking whether all polynomials in the resulting generating set are binomials as given in Algorithm 9.5. While the second part is computationally trivial, we saw in Section 7.2 that the computation of a Gröbner Basis is an **EXPSPACE**-complete problem. Nevertheless, this is the fastest known algorithm for checking whether a polynomial ideal is a binomial ideal.

In particular, we can compute elimination ideals using Gröbner Bases. The generators of an elimination ideal are just a subset of the generators of a Gröbner Basis and the Gröbner Basis of a binomial ideal consists of binomials only according to Theorem 9.4. Thus, all elimination ideals of binomial ideals are binomial ideals, again.

Algorithm 9.5 Check whether a polynomial ideal is a (pure) binomial ideal.

```
Input: $f_1, \dots, f_s \in k[x_1, \dots, x_n]$ polynomials over a field $k$ for some $n, s \in \mathbb{N}_{>0}$
Output: whether $\langle f_1, \dots, f_s \rangle$ is a (pure) binomial ideal
1: choose an arbitrary term ordering $\geq$ on $k[x_1, \dots, x_n]$
2: compute a reduced Gröbner Basis $g_1, \dots, g_t \in k[x_1, \dots, x_n]$ of $\langle f_1, \dots, f_s \rangle$ with respect to
   $\geq$ for some $t \in \mathbb{N}_{>0}$ using Algorithm 5.20 and Algorithm 5.17
3: for each $i \in \{1, \dots, t\}$ do
4:     if $g_i$ is not a (pure) binomial then
5:         return $\langle f_1, \dots, f_s \rangle$ is not a (pure) binomial ideal
6:     end if
7: end for
8: return $\langle f_1, \dots, f_s \rangle$ is a (pure) binomial ideal
```

Corollary 9.6 Let $I \trianglelefteq k[x_1, \dots, x_n]$ be a polynomial ideal over a field k for some $n \in \mathbb{N}_{>0}$. All elimination ideals of I are binomial ideals. If I is a pure binomial ideal all elimination ideals of I are pure binomial ideals.

For the following algorithms we want to use properties of the binomial generators. Since computing a reduced Gröbner Basis dominates the complexity of most following algorithms we will usually assume the binomial ideals are already given by some binomial generators, which is also often the case in practical applications.

Since pure binomial ideals are a restricted subclass of polynomial ideals one could hope that the complexity of problems of pure binomial ideals is lower than the one of the corresponding problems of general polynomial ideals. This is not the case as the worst case example presented in [MM82] is actually a pure binomial ideal.

Theorem 9.7 The word problem and the Gröbner Basis problems for binomial ideals and pure binomial ideals are **EXPSPACE**-complete.

Even though binomial ideals carry the same inherent complexity as general polynomial ideals they have more structure. The normal form of a monomial modulo a binomial ideal is a monomial again, because we can find the normal form by reducing modulo a reduced Gröbner Basis. Those Gröbner Bases consist of binomials only as seen in Theorem 9.4 and reducing a monomial modulo a binomial results in a monomial again. Thus, we can partition the set of monomials into equivalence classes of monomials having the same normal form such that this normal form is also contained in the equivalence class. A visualization of those equivalence classes can be found in Figure 9.8.

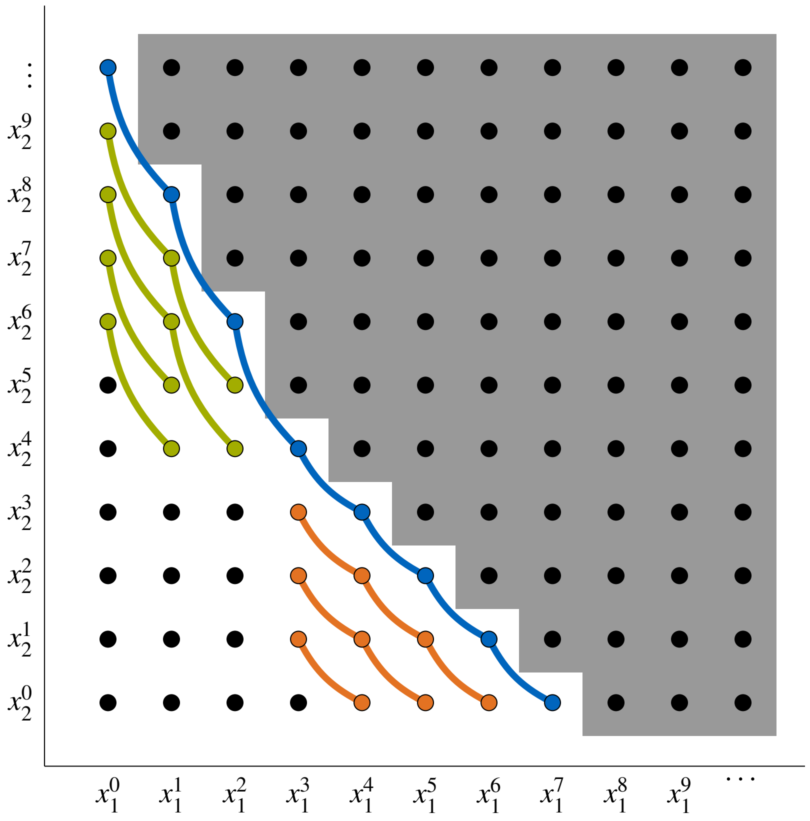


Figure 9.8: The equivalence classes of monomials modulo $I = \left\langle x_1^4 - x_1^3 x_2, x_1 x_2^4 - x_2^6 \right\rangle$. Monomials are represented by points. Two of them are in the same equivalence class if they are connected by a line or both are in an area with gray background color.

We can use this partitioning of the monomials into equivalence classes to find which polynomials are contained in a binomial ideal. We will frequently use the following theorem in Section 13.2.

Theorem 9.9 Let $I := \langle f_1, \dots, f_s \rangle \trianglelefteq R[x_1, \dots, x_n]$ be a pure binomial ideal over a field k with $\mathrm{char}(k) = 0$ for some $n, s \in \mathbb{N}_{>0}$ and pure binomials $f_1, \dots, f_s \in R[x_1, \dots, x_n]$. Furthermore, let $g \in R[x_1, \dots, x_n]$ be a polynomial with

$$g = \sum_{i=1}^{r} a_i \underline{x}^{u_i}$$

for some $r \in \mathbb{N}_{>0}$ and $a_1, \dots, a_r \in R$, $u_1, \dots, u_r \in \mathbb{N}_0^n$.

g is contained in I if and only if the sum of the coefficients of all monomials of g having the same normal form modulo I is zero, i.e. for all $\beta \in \mathbb{N}_0^n$ we have that

$$\sum_{\substack{i \in \{1,\dots,r\} \\ \overline{\underline{x}^{u_i}}^{I} = \underline{x}^{\beta}}} a_i = 0$$

Proof For the direction "⇐" assume that the coefficients of the monomials of g with the same normal form sum up to zero. The definition of the normal form implies that for each monomial of g we can add some polynomial contained in I to g such that the monomial is replaced by its normal form modulo the pure binomial ideal, which is again a monomial. Let $g' \in I \trianglelefteq R[x_1, \dots, x_n]$ be the sum of these polynomials. Since all coefficients belonging to the same normal form sum up to zero this implies that $g + g' = 0$. Because $g' \in I$ it follows that $g \in I$ too.

For the other direction "⇒" consider the generators f_i of I. They have the property, that the coefficients of their monomials with the same normal form sum up to zero. This is, because both their monomials need to have the same normal form which is implied by $f_i \in I$ and the coefficients 1 and -1 sum up to zero. For each polynomial with this property the product of this polynomial with a monomial also has that property. This can be seen since the coefficients do not change and monomials with the same normal form also have the same normal form after multiplying them with another monomial. Moreover, the sum of two polynomials with this property obviously also has this property. Since $g \in I$ can be expressed as the sum of products of the f_i with monomials, g needs to have this property too. $\square$

It is also interesting to see that binomial ideals are only closed under some operations on polynomial ideals. Using the theorems from Section 4.6 we can easily see that the sum of binomial ideals is a binomial ideal again, while intersections and products of binomial ideals are not binomial in general. Elimination ideals of binomial ideals are binomial again as seen in Corollary 9.6 and we can compute saturations and quotients of polynomial ideals using elimination ideals. Thus, quotients and saturations of binomial ideals and monomials are binomial ideals again. David Eisenbud and Bernd Sturmfels showed that radicals of binomial ideals are also binomial ideals [ES96]. All closure results mentioned in this paragraph also hold for pure binomial ideals.

9.2 Between Monomial Ideals and General Polynomial Ideals

In the last section we saw that the restriction to polynomial ideals, that can be generated by polynomials with at most two terms, does not change the complexity of the most important problems of polynomial ideals, but adds more structure. It is a natural question why we chose polynomials with at most two terms. We will discuss in this section that the restriction to polynomials with one term only results in much easier problems, but the restriction to polynomials with at most three or more terms does not add more structure to our polynomial ideals. This underlines that binomial ideals are interesting objects to study as they exactly incorporate the aspects of additional structure but still the full complexity.

We will first discuss polynomial ideals generated by monomials.

Theorem 9.10 A polynomial ideal is called a monomial ideal if there is a generating set of this ideal consisting of monomials only.

As for binomial ideals we will assume that monomial ideals are given by a generating set consisting of monomials only to allow the discussion of running times of algorithms. The s-polynomial of any two monomials vanishes, which results in a very easy structure. The following statements are simplified versions of the results on binomial ideals from the last section.

Theorem 9.11 The s-polynomial of two monomials with respect to any term order is always 0. Thus, every generating set of a monomial ideal, that consists of monomials only, is a Gröbner Basis with respect to any term ordering. Any reduced Gröbner Basis of a monomial ideal consists of monomials only.

The normal form of a monomial modulo a monomial ideal is either the monomial itself or 0. Reducing a polynomial with more terms modulo a Gröbner Basis of a monomial ideal therefore boils down to removing all terms whose monomial is a multiple of an element of the Gröbner Basis.

Theorem 9.12 Let $m_1, \ldots, m_s \in \mathbb{M}_{\{x_1,\ldots,x_n\}}$ be monomials over a field k for some $n, s \in \mathbb{N}_{>0}$ and let $f \in k[x_1, \ldots, x_n]$ be a polynomial. f is contained in $\langle m_1, \ldots, m_s\rangle$ if and only if all terms of f are a multiple of some (possibly different) m_i for some $i \in \{1, \ldots, s\}$.

Thus, we see that the most important problems for monomial ideals have polynomial complexity.

Theorem 9.13 For monomial ideals given by a generating set consisting of monomials only the Gröbner Basis problem and the word problem with arbitrary test polynomials are contained in **P**.

Polynomials with more terms, on the other hand, do not have more structure than general polynomial ideals. We could define trinomials and trinomial ideals similarly to monomials and monomial ideals or binomials and binomial ideals, respectively. There are no definitions of trinomials and trinomial ideals in the literature though, because we can simulate all polynomials using trinomials. For this simulation we need the following construction.

Theorem 9.14 Let k be a field and $n, s \in \mathbb{N}_{>0}$ be integers. Furthermore, let $t_1, \ldots, t_s \in \mathbb{N}_{>0}$, $c_{i,j} \in k$ and $m_{i,j} \in k[x_1, \ldots, x_n]$ be monomials for all $i \in \{1, \ldots, s\}$ and $j \in \{1, \ldots, t_i\}$. For

$$\begin{aligned} M = &\left\{y_{i,j} - c_{i,j}m_{i,j} - y_{i,j+1} \,|\, i \in \{1, \ldots, s\}, j \in \{1, \ldots, t_i\}\right\} \\ &\cup \left\{y_{i,1} \,|\, i \in \{1, \ldots, s\}\right\} \\ &\cup \left\{y_{i,t_i+1} \,|\, i \in \{1, \ldots, s\}\right\} \end{aligned}$$

we have

$$\langle M\rangle \cap k[x_1, \ldots, x_n] = \left\langle \sum_{j=1}^{t_1} c_{1,j}m_{1,j}, \ldots, \sum_{j=1}^{t_s} c_{s,j}m_{s,j} \right\rangle$$

where $y_{i,j}$ are new indeterminates for $i \in \{1, \ldots, s\}$ and $j \in \{1, \ldots, t_i + 1\}$.

Proof Notice that for all $i \in \{1, \ldots, s\}$ we have

$$y_{i_1} - \sum_{i=1}^{t_i}(y_{i,j} - c_{i,j}m_{i,j} - y_{i,j+1}) - y_{i,t_i+1} = \sum_{i=1}^{t_i} c_{i,j}m_{i,j}$$

This immediately shows the relation "⊇". The other direction "⊆" holds because we are not able to build polynomials containing only the variables $x_1, \ldots, x_n$ using the generators in M without adding up multiples of the equation above. □

The interesting observation about Theorem 9.14 is that $\langle M \rangle$ is what we would call a trinomial ideal while the right side

$$\left\langle \sum_{j=1}^{t_1} c_{1,j} m_{1,j}, \ldots, \sum_{j=1}^{t_s} c_{s,j} m_{s,j} \right\rangle$$

of the equation is some arbitrary polynomial ideal. This implies that we can add some new indeterminates and then reformulate our polynomial ideal as a trinomial ideal. Thus, trinomial ideals have a similar structure as general polynomial ideals, since all polynomial ideals can be transformed via new indeterminates to trinomial ideals.

10 Toric Ideals

10.1 Definition of Toric Ideals

In this section we will discuss a subclass of polynomial ideals that is a restriction of pure binomial ideals: toric ideals. Even though their definition is even more restrictive than the one of pure binomial ideals, toric ideals appear in many applications, for example in the cellular decomposition presented in Section 11.1. On the other hand, their specialized structure allows for more efficient algorithms on them. We will see for instance that the word problem of toric ideals can be decided in polynomial time. For detailed proofs of the theorems presented in this section we refer to the Master's thesis of this author [Tom15b].

Toric ideals appear that often in practice, because they can be defined as kernels of linear maps as defined in Definition 4.19.

Definition 10.1 Let k be a field, $x_1, \dots, x_n$ and $y_1, \dots y_s$ be indeterminates, and $m_1, \dots, m_n$ be monomials contained in $k(y_1, \dots, y_s)$, the quotient field of $k[y_1, \dots, y_s]$, for $n, s \in \mathbb{N}_{>0}$. Let ϕ be the k-algebra homomorphism

$$\phi : k[x_1, \dots, x_n] \to k(y_1, \dots, y_s)$$

defined by

$$\phi(x_1) = m_1, \dots, \phi(x_n) = m_n.$$

The toric ideal associated to $m_1, \dots, m_n$ is the ideal $\ker(\phi) = \{f \in k[x_1, \dots, x_n] \mid \phi(f) = 0\}$.

In Section 4.6 we discussed how to saturate a polynomial ideal. Saturating a polynomial ideal means that whenever there is a polynomial contained in the ideal, which has common indeterminates in all terms, we can remove those indeterminates and still get a polynomial contained in the ideal. The definition of toric ideals directly implies that toric ideals are saturated since $\phi(x_i f) = 0$ implies $0 = \phi(x_i f) = m_i \phi(f)$ and thus $\phi(f) = 0$ for all $i \in \{1, \dots, n\}$ and $f \in k[x_1, \dots, x_n]$.

Now consider a toric ideal $I \trianglelefteq k[x_1, \dots, x_n]$ associated to the monomials

$$m_1, \dots, m_n \in k(y_1, \dots, y_s)$$

for some $n, s \in \mathbb{N}_{>0}$. We can write I in the following way with new indeterminates $z_1, \dots, z_s$

which represent $y_1^{-1}, \ldots, y_s^{-1}$:

$$\begin{aligned}\ker(\phi) &= \overbrace{\left\langle x_1 - \underline{y}^{u_1}, \ldots, x_n - \underline{y}^{u_n} \right\rangle}^{\text{in } k[x_1,\ldots,x_n,y_1^{\pm 1},\ldots,y_s^{\pm 1}]} && \cap\, k[x_1, \ldots, x_n] \\ &= \underbrace{\left\langle x_1 - \underline{y}^{u_1^+}\underline{z}^{u_1^-}, \ldots, x_n - \underline{y}^{u_n^+}\underline{z}^{u_n^-}, 1 - y_1 z_1, \ldots, 1 - y_s z_s \right\rangle}_{\text{in } k[x_1,\ldots,x_r,y_1,\ldots,y_s,z_1,\ldots,z_s]} && \cap\, k[x_1, \ldots, x_n]\end{aligned}$$

where the i-th entries of w^+ and w^- are defined as

$$w_i^+ := \begin{cases} w_i & \text{if } w_i > 0 \\ 0 & \text{else} \end{cases} \text{ and } w_i^- := \begin{cases} -w_i & \text{if } w_i < 0 \\ 0 & \text{else} \end{cases}$$

for all $i \in \{1, \ldots, n\}$ and $w \in \mathbb{Z}^n$. As discussed in Corollary 9.6 in Section 9.1, elimination ideals of pure binomial ideals are once more pure binomial ideals. Elimination ideals can be computed as a subset of a Gröbner Basis as presented in the Elimination Theorem in Theorem 5.21 and the s-polynomials of pure binomials are pure binomials once again. Thus, all toric ideals are pure binomial ideals.

It turns out that all saturated pure binomial ideals are in fact toric ideals and thus toric ideals can be equivalently defined as saturated pure binomial ideals.

Theorem 10.2 Let k be a field and $n \in \mathbb{N}_{>0}$. $I \trianglelefteq k\,[x_1, \ldots, x_n]$ is a toric ideal if and only if it is a saturated pure binomial ideal.

We will now see that toric ideals have another useful property. Namely, toric ideals are always prime ideals and thus radical ideals. To discuss this we need to introduce prime ideals first.

Definition 10.3 Let R be a commutative ring and $I \trianglelefteq R$ be an ideal of R. I is a prime ideal if and only if for all $a, b \in R$ with $ab \in I$ we have $a \in I$ or $b \in I$.

Toric ideals are always prime ideals, which is easy to see from the definition of toric ideals. Suppose we have polynomials $f, g \in k\,[x_1, \ldots, x_n]$ over a field k for some $n \in \mathbb{N}_{>0}$ and a k-algebra homomorphism $\phi : k[x_1, \ldots, x_n] \to k(y_1, \ldots, y_s)$ as given in Definition 10.1. If $\phi(fg) = 0$ we can transform this to $\phi(f)\phi(g) = 0$ since ϕ is a k-algebra homomorphism. Because the field k is a domain, this is equivalent to $\phi(f) = 0$ or $\phi(g) = 0$, which means $f \in \ker(\phi)$ or $g \in \ker(\phi)$. As proven by David Eisenbud and Bernd Sturmfels [ES96], pure binomial prime ideals and toric ideals are actually the same.

Theorem 10.4 ([ES96]) Let k be a field with characteristic zero and $n \in \mathbb{N}_{>0}$. $I \trianglelefteq k[x_1, \ldots, x_n]$ is a toric ideal if and only if it is a pure binomial prime ideal.

Prime Ideals are in particular interesting in this context, because all prime ideals are radical ideals.

Theorem 10.5 Let R be a ring and $I \trianglelefteq R$ be a prime ideal. I is a radical ideal.

Proof Let $a \in R$ be an element such that there is an $s \in \mathbb{N}_{>0}$ with $a^s \in I$. We have to show that $a \in I$. Without loss of generality we can assume that there is $r \in \mathbb{N}_{>0}$ with $s = 2^r$. If s is no power of 2 we can increase it to the next power of 2. Since I is a prime ideal and $a^{2^{r-1}} a^{2^{r-1}} = a^{2^r} \in I$ we know that $a^{2^{r-1}} \in I$. Similarly, we can iteratively prove $a^{2^{r-2}} \in I$, $a^{2^{r-3}} \in I, \ldots, a^{2^0} = a \in I$. □

Corollary 10.6 Let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a toric ideal over a field k with characteristic zero for some $n \in \mathbb{N}_{>0}$. I is radical.

Thus, the word problem and the radical word problem of toric ideals are the same problems. Also, the radical problem of toric ideals is trivial since all toric ideals are already radical.

10.2 The Word Problem of Toric Ideals

The computational complexity of the word problem for toric ideals is much lower than for general polynomial ideals. This is interesting, because as we have seen toric ideals are just saturated pure binomial ideals, and the word problem for binomial ideals is **EXPSPACE**-complete like the word problem for general polynomial ideals.

To understand why it is so important for the running time, that the ideal is saturated, we will classify the set of saturated pure binomials contained in a polynomial ring.

Theorem 10.7 Let k be a field and $n \in \mathbb{N}_{>0}$. The map

$$\phi : \{f \in k[x_1, \ldots, x_n] \mid f \text{ saturated pure binomial}\} \to \mathbb{Z}^n, \quad \underline{x}^u - \underline{x}^v \mapsto v - u$$

for all $u, v \in \mathbb{N}_0^n$ is a bijection with inverse map

$$\begin{aligned}\varphi : \mathbb{Z}^n &\to \{f \in k[x_1, \dots, x_n] \mid f \text{ saturated pure binomial}\},\\ w &\mapsto \underline{x}^{w^-} - \underline{x}^{w^+}\end{aligned}$$

for all $w \in \mathbb{Z}^n$.

Proof It is easy to check that both maps are well-defined because $w^+, w^- \in \mathbb{N}_0^n$ for all $w \in \mathbb{Z}^n$. We also obviously have $\varphi(\phi(\underline{x}^u - \underline{x}^v)) = \underline{x}^u - \underline{x}^v$ and $\phi(\varphi(w)) = w$ for all $u, v \in \mathbb{N}_0^n$ and $w \in \mathbb{Z}^n$. □

Note that ϕ can also be used to map non-saturated pure binomials. The value of ϕ for some pure binomial is the same as the one of the saturated form of the pure binomial in that case.

With this bijection we can execute the computation of the word problem for toric ideals on $\mathbb{Z}^n$ and use the tools available for this $\mathbb{Z}$-vector space. To do so, we first have to establish that we can replace all non-saturated pure binomials by their saturated forms in the word problem for toric ideals.

Theorem 10.8 Let $I \trianglelefteq k[x_1, \dots, x_n]$ be a toric ideal over a field k for some $n \in \mathbb{N}_{>0}$ and let $u, v \in \mathbb{N}_0^n$. We have

$$\underline{x}^u - \underline{x}^v \in I \Leftrightarrow \underline{x}^{(v-u)^-} - \underline{x}^{(v-u)^+} \in I$$

Proof From Theorem 10.4 we know that I is a saturated pure binomial ideal. The theorem holds because $\underline{x}^{(v-u)^-} - \underline{x}^{(v-u)^+}$ is the saturated form of $\underline{x}^u - \underline{x}^v$. □

The last two theorems state, that we can translate the word problem of toric ideals in terms of a $\mathbb{Z}$-vector space, which makes solving the problem much easier.

Theorem 10.9 Let $g, f_1, \dots, f_s \in k[x_1, \dots, x_n]$ be pure binomials over a field k for some $n, s \in \mathbb{N}_{>0}$ such that $\langle f_1, \dots, f_s \rangle \trianglelefteq k[x_1, \dots, x_n]$ is a toric ideal. Using the notation from Theorem 10.7, we consider the $\mathbb{Z}$-vector space $\operatorname{span}(\{\phi(f_1), \dots, \phi(f_s)\}) \subseteq \mathbb{Z}^n$ and have

$$g \in \langle f_1, \dots, f_s \rangle \Leftrightarrow \phi(g) \in \operatorname{span}(\{\phi(f_1), \dots, \phi(f_s)\})$$

Proof We will just give the main idea of the proof here. For a detailed proof we refer to the Master's thesis of this author [Tom15b].

It is important to note that the s-polynomial of pure polynomials is easy to compute in terms of the corresponding exponent vectors. Let $h_1, h_2 \in k[x_1, \ldots, x_n]$ be toric ideals. It is easy to verify that $\phi(\operatorname{spol}(h_1, h_2)) = \pm\big(\phi(h_1) - \phi(h_2)\big)$. As seen in Section 5.3, we can compute whether $g \in \langle f_1, \ldots, f_s \rangle$ by computing s-polynomials of pure binomials numerous times. With s-polynomials we can compute a Gröbner Basis of $\langle f_1, \ldots, f_s \rangle \trianglelefteq k[x_1, \ldots, x_n]$ and all reductions. As s-polynomials of pure binomials can be computed as differences of the exponent vectors, all exponent vectors appearing during the reduction of g modulo $\langle f_1, \ldots, f_s \rangle$ are equivalent modulo $\operatorname{span}(\{\phi(f_1), \ldots, \phi(f_s)\})$. Conversely, all sums and differences of exponent vectors can be executed in terms of s-polynomial computations. As $0 \in \operatorname{span}(\{\phi(f_1), \ldots, \phi(f_s)\})$ we can reduce g to zero if and only if $\phi(g) \in \operatorname{span}(\{\phi(f_1), \ldots, \phi(f_s)\})$, which is exactly the claim. $\square$

In other words, computing an s-polynomial of a pure binomial $f = \underline{x}^a - \underline{x}^b \in k[x_1, \ldots, x_n]$ with $\underline{x}^u - \underline{x}^v \in k[x_1, \ldots, x_n]$ for some $a, b, u, v \in \mathbb{N}_0^n$ replaces f with a pure binomial where in the leading term of f we replaced $\underline{x}^u$ by $\underline{x}^v$. We may also multiply f by some monomial, but this can be ignored since we discuss toric ideals, i.e. saturated ideals. We now want to apply several generators of the toric ideal to f and reduce it to 0, which means finding coefficients $c_1, \ldots, c_s \in \mathbb{Z}$ such that $b - a = \sum_{i=1}^{s} c_i \phi(f_i)$. That is, we want to solve a linear system of integer equations and thus reduced our membership problem of toric ideals to a membership problem of linear equations.

This result is important since we can use tools for vector spaces to solve the word problem on the $\mathbb{Z}$-vector space $\operatorname{span}(\{\phi(f_1), \ldots, \phi(f_s)\}) \subseteq \mathbb{Z}^n$. We can use algorithms like the Gaussian algorithm explained in Section 4.4. Over $\mathbb{Z}$-vector spaces the row-echelon form is called Hermite normal form and can be computed by the Gaussian algorithm or the so-called LLL algorithm introduced by Arjen K. Lenstra, Hendrik W. Lenstra and László Lovász in 1982 [LLL82, Bre11]. As discussed in Section 7.2, both algorithms run in polynomial time [EGG+06].

Corollary 10.10 Let $g, f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ be pure binomials over a field k for some $n, s \in \mathbb{N}_{>0}$ such that $\langle f_1, \ldots, f_s \rangle \trianglelefteq k[x_1, \ldots, x_n]$ is a toric ideal. There is an algorithm that decides whether $g \in \langle f_1, \ldots, f_s \rangle$ in polynomial time.

An overview of the complexities of word problems for different types of polynomial ideals discussed in this thesis can be found in Figure 10.11.

We can also adapt the proof of Theorem 10.9 slightly to discuss the triviality problem of toric ideals. The s-polynomial of two pure binomials is always a pure binomial or zero. Thus, we can never reduce one to zero modulo a set of pure binomials. A toric ideal could therefore only be trivial if and only if one of the given generators is already a monomial, in particular $\underline{x}^0 = 1$, which is easy to check.

$$\begin{tikzcd}
& \begin{array}{c}\textbf{polynomial ideals}\\ \textbf{EXPSPACE}\\ \text{Theorem 7.13}\end{array} & \\
& & \begin{array}{c}\textbf{radical ideals}\\ \textbf{PSPACE}\\ \text{Theorem 8.13}\end{array} \arrow[ul, "\subseteq"] \\
\begin{array}{c}\textbf{binomial ideals}\\ \textbf{EXPSPACE}\\ \text{Theorem 9.7}\end{array} \arrow[uur, "\subseteq"] & & \\
& & \begin{array}{c}\textbf{radical binomial ideals}\\ \textbf{coNP}\\ \text{Corollary 11.13}\end{array} \arrow[uu, "\subseteq"] \arrow[ull, "\subseteq"] \\
\begin{array}{c}\textbf{pure binomial ideals}\\ \textbf{EXPSPACE}\\ \text{Theorem 9.7}\end{array} \arrow[uu, "\subseteq"] & & \\
& & \begin{array}{c}\textbf{radical pure binomial ideals}\\ \textbf{coNP}\\ \text{Theorem 11.8}\end{array} \arrow[uu, "\subseteq"] \arrow[ull, "\subseteq"] \\
& \begin{array}{c}\textbf{toric ideals}\\ \textbf{P}\\ \text{Corollary 10.10}\end{array} \arrow[uul, "\subseteq"] \arrow[ur, "\subseteq"] &
\end{tikzcd}$$

Figure 10.11: The complexity of the word problem for different classes of polynomial ideals.

Corollary 10.12 Let k be a field. The triviality problem for toric ideals $I \trianglelefteq k[x_1, \ldots, x_n]$ for some $n \in \mathbb{N}_{>0}$ can be solved in polynomial time.

11 Cellular Decomposition

11.1 Cellular Decomposition

The paper [ES96] also introduced a tool for binomial ideals called cellular decomposition, that we will use to analyze the complexity of the radical word problem of binomial ideals.

The main idea of cellular decomposition is to decompose the affine space into so-called cells.

Definition 11.1 Let k be a field and $n \in \mathbb{N}_{>0}$. For each $\Delta \subseteq \{x_1, \ldots, x_n\}$ the Δ-cell of $k[x_1, \ldots, x_n]$ is the set

$$k_\Delta := \left\{a \in k^n \mid a_i \neq 0 \Leftrightarrow x_i \in \Delta \text{ for all } i \in \{1, \ldots, n\}\right\} \subseteq k^n$$

For each ideal $I \trianglelefteq k[x_1, \ldots, x_n]$ and each set $\Delta \subseteq \{x_1, \ldots, x_n\}$ we define

$$\mathcal{V}_\Delta(I) := \mathcal{V}(I) \cap k_\Delta$$

There are 2^n different cells, which is an exponential number in the number of indeterminates. Visualizations of some samples of cells are shown in Figure 11.2 and Figure 11.3.

Cells have the property, that different cells are disjoint and all cells together span the full affine space. Thus, we call the following formula the cellular decomposition of k^n

$$k^n = \bigcup_{\Delta \subseteq \{x_1, \ldots, x_n\}} k_\Delta$$

Likewise, we can partition the variety of any polynomial ideal $I \trianglelefteq k[x_1, \ldots, x_n]$ into cells

$$\mathcal{V}(I) = \bigcup_{\Delta \subseteq \{x_1, \ldots, x_n\}} \mathcal{V}_\Delta(I)$$

Those cells can also be used to define the cellular decomposition of polynomial ideals instead of their varieties.

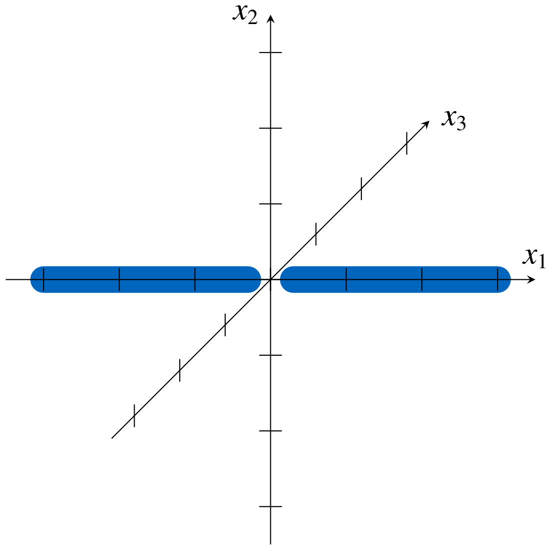


Figure 11.2: A visualization of the cell $\mathbb{Q}_{\{x_1\}} \subseteq \mathbb{Q}^3$.

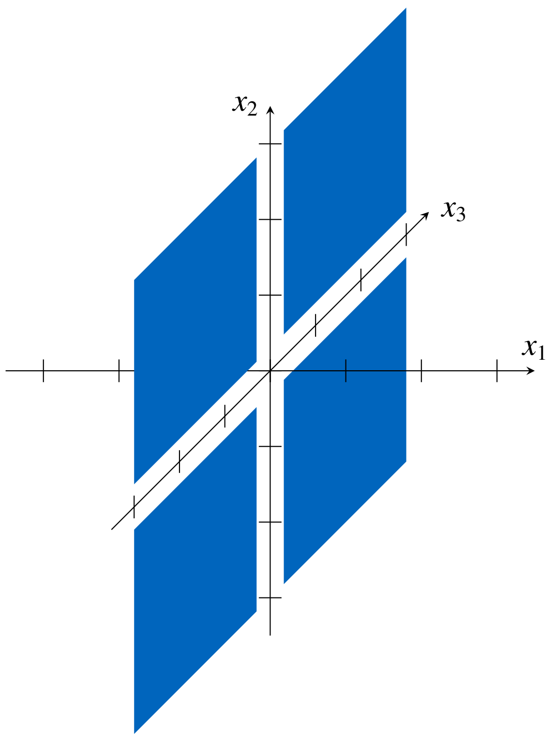


Figure 11.3: A visualization of the cell $\mathbb{Q}_{\{x_2,x_3\}} \subseteq \mathbb{Q}^3$.

Theorem 11.4 (cellular decomposition [ES96]) Let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a pure binomial ideal over an algebraically closed field k with characteristic zero for some $n \in \mathbb{N}_{>0}$. We have

$$\sqrt{I} = \bigcap_{\Delta \subseteq \{x_1,\ldots,x_n\}} \left(I_\Delta : \left(\prod_{x_i \in \Delta} x_i \right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle \right)$$

where I_Δ is the image of I under the ring homomorphism defined by

$$1 \mapsto 1, x_i \mapsto \begin{cases} x_i & \text{if } x_i \in \Delta \\ 0 & \text{else} \end{cases}$$

for all $\Delta \subseteq \{x_1, \ldots, x_n\}$. The ideals $I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle$ for some $\Delta \subseteq \{x_1, \ldots, x_n\}$ are called cell ideals of I.

Proof Consider the cellular decomposition of the variety of I.

$$\mathcal{V}(I) = \bigcup_{\Delta \subseteq \{x_1,\ldots,x_n\}} \mathcal{V}_\Delta(I)$$

Taking the vanishing ideal of both sides results in

$$\mathcal{I}(\mathcal{V}(I)) = \mathcal{I}\left(\bigcup_{\Delta\subseteq\{x_1,\dots,x_n\}} \mathcal{V}_\Delta(I)\right)$$

On the left-hand side we can apply Hilbert's Nullstellensatz from Theorem 8.3 to find that the vanishing ideal is the radical of I. On the right-hand side we can exchange the order of the computation of the vanishing ideal with the other operations as discussed in Section 4.6. This results in

$$\begin{aligned}
\sqrt{I} &= \bigcap_{\Delta\subseteq\{x_1,\dots,x_n\}} \mathcal{I}(\mathcal{V}_\Delta(I))\\
&= \bigcap_{\Delta\subseteq\{x_1,\dots,x_n\}} \mathcal{I}\Big((\mathcal{V}(I) \setminus \{a \in k^n \,|\, a_i = 0, x_i \in \Delta \text{ for some } i \in \{1,\dots,n\}\})\\
&\qquad\qquad \cap \{a \in k^n \,|\, a_i = 0 \text{ for all } i \in \{1,\dots,n\} \text{ with } x_i \notin \Delta\}\Big)\\
&= \bigcap_{\Delta\subseteq\{x_1,\dots,x_n\}} \Big((\mathcal{I}(\mathcal{V}(I)) : \mathcal{I}(\{a \in k^n \,|\, a_i = 0, x_i \in \Delta \text{ for some } i \in \{1,\dots,n\}\})^\infty)\\
&\qquad\qquad + \mathcal{I}(\{a \in k^n \,|\, a_i = 0 \text{ for all } i \in \{1,\dots,n\} \text{ with } x_i \notin \Delta\})\Big)\\
&= \bigcap_{\Delta\subseteq\{x_1,\dots,x_n\}} \sqrt{I} : \left(\prod_{x\in\Delta} x\right)^\infty + \langle x \,|\, x \notin \Delta\rangle\\
&= \bigcap_{\Delta\subseteq\{x_1,\dots,x_n\}} \sqrt{I : \left(\prod_{x\in\Delta} x\right)^\infty + \langle x \,|\, x \notin \Delta\rangle}
\end{aligned}$$

As all indeterminates not contained in Δ are added to the ideals of each cell either way, we can also remove them from the left ideal to obtain

$$\sqrt{I} = \bigcap_{\Delta\subseteq\{x_1,\dots,x_n\}} \sqrt{I_\Delta : \left(\prod_{x\in\Delta} x\right)^\infty + \langle x \,|\, x \notin \Delta\rangle}$$

Note that until here we did not use that I is a pure binomial ideal, so the above equation still holds for all polynomial ideals. If I is a pure binomial ideal, the ideal $I_\Delta : (\prod_{x\in\Delta} x)^\infty$ is a saturated pure binomial ideal and thus a toric ideal by Theorem 10.4. In case the replacement of I by I_Δ results in a monomial, the saturation is the full ideal $\langle 1\rangle = k[x_1,\dots,x_n]$ which is also toric. With Theorem 10.6 we know that therefore $I_\Delta : (\prod_{x\in\Delta} x)^\infty$ is radical as the characteristic of k is assumed to be 0. Thus, we can omit computing the radical on the right-hand side of the equation, which results in the formula to be proven. □

The cellular decomposition of pure binomial ideals is useful, because we are able to remove the radical operation from the cellular ideals in the last step of the proof. This allows for computations on the radical of a pure binomial ideal without having to compute any radical.

In return, the number of cell ideals is exponential in the number of indeterminates. David Eisenbud and Bernd Sturmfels also proved that Theorem 11.4 works for general binomial ideals (which may have coefficients).

Note that the intersection of binomial ideals is not a binomial ideal in general though. Thus, when computing the radical of a binomial ideal using cellular decomposition, some intermediate results are no binomial ideal in general. Eberhard Becker, Rudolf Grobe, and Michael Niermann showed that nevertheless the intersection operations can be computed in an order such that all intermediate results are binomial ideals [BGN97].

11.2 The Radical Word Problem for Binomial Ideals

The cellular decomposition allows for better algorithms to solve the radical word problem of binomial ideals, but not for the computation of the complete radical of binomial ideals. The problem is that to compute the radical of a pure binomial ideal using cellular decomposition, we need to compute the intersection of an exponential number of polynomial ideals, that are not necessarily toric ideals. The ideals

$$I_\Delta : \left(\prod_{x\in\Delta} x\right)^\infty$$

appearing in the cellular decomposition are toric for a pure binomial ideal $I \trianglelefteq k[x_1, \ldots, x_n]$ over a field k for some $n \in \mathbb{N}_{>0}$ and $\Delta \subseteq \{x_1, \ldots, x_n\}$, but the addition of the non-cell indeterminates makes the cell ideal

$$I_\Delta : \left(\prod_{x_i\in\Delta} x_i\right)^\infty + \langle\{x_i \mid x_i \notin \Delta\}\rangle$$

a non-toric polynomial ideal in general. Cell ideals are still binomial ideals, but there are no known better complexity bounds for the computation of the intersection of binomial ideals in comparison to the computation of the intersection of general polynomial ideals.

An important part of the intersection of toric ideals on the other hand is that they can be computed in polynomial time using Theorem 10.9. With this theorem, we just need to compute the intersection of $\mathbb{Z}$-vector spaces to describe all saturated pure binomial in the intersection of two toric ideals. Computing the intersection of $\mathbb{Z}$-vector spaces can be done for instance using the Zassenhaus Algorithm. This Algorithm is named after the German mathematician Hans J. Zassenhaus, even though there is no known formal publication by him presenting the algorithm. The Zassenhaus Algorithm needs an implementation of the Gauß algorithm and together both algorithms run in polynomial time [Fis13, Lan87]. Note that the intersection of toric ideals is not a toric ideal in general though.

Theorem 11.5 There is an algorithm that computes a generating set of a toric ideal containing exactly those pure binomials contained in the intersection of two toric ideals $I, J \trianglelefteq k[x_1, \ldots, x_n]$ over a field k for some $n \in \mathbb{N}_{>0}$ in time polynomial in n and the encoding size of I and J.

The radical word problem of pure binomial ideals on the other hand can be solved much more efficiently using cellular decomposition instead of computing a Gröbner Basis and then reducing the test polynomial modulo that basis. We will now show that the radical word problem for binomial ideals can be solved in **coNP**, which is much more efficient than **EXPSPACE** in the general case. To do so, we need to give a certificate, which makes it easy to verify that a binomial is not contained in the radical of a binomial ideal.

Theorem 11.6 There is an algorithm, that computes whether some given pure binomial $f \in k[x_1, \ldots, x_n]$ over a field k for some $n \in \mathbb{N}_{>0}$ is contained in the radical of a pure binomial ideal $I \trianglelefteq k[x_1, \ldots, x_n]$ in **coNP**.

Proof With the cellular decomposition presented in Theorem 11.4 we know that $f \in \sqrt{I}$ if and only if for all $\Delta \subseteq \{x_1, \ldots, x_n\}$ we have

$$f \in I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle$$

Negating this statement results in $f \notin \sqrt{I}$ if and only if there is $\Delta \subseteq \{x_1, \ldots, x_n\}$ such that

$$f \notin I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle$$

Thus, a certificate for $f \notin \sqrt{I}$ is such an $\Delta \subseteq \{x_1, \ldots, x_n\}$ with

$$f \notin I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle$$

The certificate clearly has at most polynomial size.

It remains to show that the certificate can be verified in polynomial time, i.e. we can decide in polynomial time whether $f \in I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle$. To do so we make a case distinction:

a) All terms of f involve indeterminates not contained in Δ: In this case we clearly have $f \in I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle$.

b) Exactly one term of f does only contain indeterminates from Δ: In this case we have that $f \in I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle$ if and only if $I_\Delta : (\prod_{x \in \Delta} x)^\infty = \langle 1 \rangle$ since $I_\Delta : (\prod_{x \in \Delta} x)^\infty$ is saturated. The triviality problem of polynomial ideals can be solved in polynomial time as discussed in Corollary 10.12.

c) Exactly two terms of f only contain indeterminates from Δ: In this case we can use Corollary 10.10 to solve the word problem of toric ideals in polynomial time.

All cases can be verified in polynomial time, which concludes the proof. □

This certificate can also be slightly extended to solve the problem for general polynomials $f \in k[x_1, \ldots, x_n]$ instead of only pure binomials.

Theorem 11.7 There is an algorithm that computes whether a polynomial $f \in k[x_1, \ldots, x_n]$ over a field k for some $n \in \mathbb{N}_{>0}$ is contained in the radical of some given pure binomial ideal $I \trianglelefteq k[x_1, \ldots, x_n]$ in **coNP**.

Proof We use the criterion given in Theorem 9.9 to extend our certificate from Theorem 11.6. The certificate now also contains one monomial such that the sum of the coefficients of monomials equivalent to the given one modulo the cell ideal is not 0. Such a monomial has to exist if and only if f is not contained in the cell ideal as proven in Theorem 9.9. Let

$$f = \sum_{i=1}^{r} a_i \underline{x}^{u_i}$$

for some $r \in \mathbb{N}_{>0}$, $a_1, \ldots, a_r \in R$, and $u_1, \ldots, u_r \in \mathbb{N}_0^n$. The full certificate consists of a $\Delta \subseteq \{x_1, \ldots, x_n\}$ and $v \in \mathbb{N}_0^n$ such that

$$\sum_{\substack{i \in \{1,\ldots,r\} \\ \overline{\underline{x}^{u_i}}^{I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle} = \underline{x}^v}} a_i \neq 0$$

Again, this certificate has polynomial size in the input length.

We now need to show that we can verify the certificate in polynomial time. Observe that we have

$$\overline{\underline{x}^{u_i}}^{I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle} = \underline{x}^v$$

if and only if

$$\underline{x}^{u_i} - \underline{x}^v \in I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle$$

Using Corollary 10.10 we can now select the coefficients to sum up in polynomial time. The summation and comparison to 0 can also clearly be done in polynomial time. □

In the Master's thesis of this author we proved that there are matching lower bounds for the radical word problem of binomial ideals by reducing the **TAUTOLOGY** problem to the radical word problem of pure binomial ideals [Tom15b]. Together, this proves that the radical word problem for pure binomial ideals is **coNP**-complete.

Theorem 11.8 ([Tom15b]) Let k be a field. The radical word problem of pure binomial ideals $I \trianglelefteq k[x_1, \ldots, x_n]$ for $n \in \mathbb{N}_{>0}$ is **coNP**-complete.

11.3 Modeling Binomial Ideals Using Pure Binomials

We will now show that the results from the previous section also hold for non-pure binomial ideals. The proof, that the radical word problem of pure binomial ideals is contained in **coNP**, has only one step that cannot be carried out with non-pure binomials: Solving the word problem of the saturated cell ideals in polynomial time by mapping the ideal to a $\mathbb{Z}$-vector space. An adjusted construction, that takes coefficients into account, will be presented here.

The first step to solve the word problem of toric ideals is to translate the pure binomials to a $\mathbb{Z}$-vector space. We can do a similar construction for saturated binomial ideals.

Theorem 11.9 Let k be a field, $n \in \mathbb{N}_{>0}$, and $\geq$ be a term ordering on $k[x_1, \ldots, x_n]$. The map

$$\begin{aligned} \phi : \{f \in k[x_1, \ldots, x_n] \mid f \text{ saturated monic binomial}\} &\to k \times \{w \in \mathbb{Z}^n \mid \underline{x}^{w+} \preceq \underline{x}^{w-}\}, \\ \underline{x}^u - a\underline{x}^v &\mapsto (a, v - u) \end{aligned}$$

for all $a \in k$ and $u, v \in \mathbb{N}_0^n$ is a bijection with inverse map

$$\begin{aligned} \varphi : k \times \{w \in \mathbb{Z}^n \mid \underline{x}^{w+} \preceq \underline{x}^{w-}\} &\to \{f \in k[x_1, \ldots, x_n] \mid f \text{ saturated monic binomial}\}, \\ (a, w) &\mapsto \begin{cases} \underline{x}^{w^-} - a\underline{x}^{w^+} & \text{if } \underline{x}^{w^-} \geq \underline{x}^{w^+} \\ \underline{x}^{w^+} - a^{-1}\underline{x}^{w^-} & \text{else} \end{cases} \end{aligned}$$

for all $a \in k$ and $w \in \mathbb{Z}^n$ with $\underline{x}^{w+} \preceq \underline{x}^{w-}$.

Proof As for toric ideals it is easy to check that both maps are well-defined and that we have $\varphi(\phi(\underline{x}^u - a\underline{x}^v)) = \underline{x}^u - a\underline{x}^v$ and $\phi(\varphi((a, w))) = (a, w)$ for all $u, v \in \mathbb{N}_0^n$ with $\underline{x}^u \geq \underline{x}^v$, $a \in k$ and $w \in \{w \in \mathbb{Z}^n \mid \underline{x}^{w+} \preceq \underline{x}^{w-}\}$. □

We use monic binomials only since all binomials can be made monic by dividing by their leading coefficients. As binomials, that differ only in a constant factor, can be considered equal for ideal membership we can map all binomials to their monic form and thus we get the generalized map

$$\begin{aligned}\phi : \{f \in k[x_1, \ldots, x_n] \mid f \text{ binomial}\} &\to k \times \mathbb{Z}^n, \\ a\underline{x}^u - b\underline{x}^v &\mapsto \left(\frac{b}{a}, v - u\right)\end{aligned}$$

for all $a, b \in k$ and $u, v \in \mathbb{N}_0^n$. Similarly, we can extend φ to

$$\begin{aligned}\varphi : k \times \mathbb{Z} &\to \{f \in k[x_1, \ldots, x_n] \mid f \text{ saturated monic binomial}\}, \\ (a, w) &\mapsto \begin{cases} \underline{x}^{w^-} - a\underline{x}^{w^+} & \text{if } \underline{x}^{w^-} \geq \underline{x}^{w^+} \\ \underline{x}^{w^+} - a^{-1}\underline{x}^{w^-} & \text{else} \end{cases}\end{aligned}$$

as multiplying by -1 just corresponds to exchanging the terms of a binomial.

We regard $k \times \mathbb{Z}^n$ as a $\mathbb{Z}$-vector space with the operations

$$(a, u) + (b, v) := (ab, u + v)$$

and

$$c \cdot (a, u) := \underbrace{(a, u) + \cdots + (a, u)}_{c \text{ times}} = (a^c, cu)$$

for all $c \in \mathbb{Z}$, $a, b \in k$, and $u, v \in \mathbb{Z}^n$. The sum operation is set up in such a way such that the computation of s-polynomials corresponds to the sum in the vector space.

Theorem 11.10 Let $a_1, a_2 \in k$, $u_1, u_2, v_1, v_2 \in \mathbb{Z}^n$, and $\geq$ be a term ordering on $k[x_1, \ldots, x_n]$ with $u_1 \geq v_1$ and $u_2 \geq v_2$ for some field k and $n \in \mathbb{N}_{>0}$. We have

$$\phi\left(\text{spol}\left(\underline{x}^{u_1} - a_1\underline{x}^{v_1}, \underline{x}^{u_2} - a_2\underline{x}^{v_2}\right)\right) = \pm\left(\phi(\underline{x}^{u_1} - a_1\underline{x}^{v_1}) - \phi(\underline{x}^{u_2} - a_2\underline{x}^{v_2})\right)$$

Proof We have

$$\text{spol}\left(\underline{x}^{u_1} - a_1\underline{x}^{v_1}, \underline{x}^{u_2} - a_2\underline{x}^{v_2}\right) = -a_1\underline{x}^{v_1 + (u_2 - u_1)+} + a_2\underline{x}^{v_2 + (u_2 - u_1)-}$$

which is mapped to $\pm\left(\frac{a_1}{a_2}, (v_1 - u_1) - (v_2 - u_2)\right)$ by ϕ. The sign depends on the order of $\underline{x}^{v_1 + (u_2 - u_1)+}$ and $\underline{x}^{v_2 + (u_2 - u_1)-}$ with respect to $\geq$. The inner part of the right-hand side evaluates to

$$\phi(\underline{x}^{u_1} - a_1\underline{x}^{v_1}) - \phi(\underline{x}^{u_2} - a_2\underline{x}^{v_2}) = (a_1, v_1 - u_1) - (a_2, v_2 - u_2) = \left(\frac{a_1}{a_2}, (v_1 - u_1) - (v_2 - u_2)\right)$$

which concludes the proof. □

For this computation we basically replaced the exponent monoid of the polynomial ring $k[x_1, \ldots, x_n] = k[\mathbb{N}_0^n]$ by the monoid $(k \times \mathbb{N}_0^n, +, (1, 0, \ldots, 0))$ where $(a, u) + (b, v) := (ab, u + v)$ for all $a, b \in k$, and $u, v \in \mathbb{Z}^n$. The coefficients are encoded into a new indeterminate and the summation of the exponents of this new indeterminate is modified. Alternatively, we could also have mapped the coefficients to logarithms in the exponent and leave the operation untouched. This approach has the disadvantage that we have to save real numbers in case of a rational base field and that a logarithm may not exist over all fields.

Encoding the coefficients into a new indeterminate may also be useful for other applications. To do so in general, one still has to identify all possible coefficients with the corresponding powers of the new indeterminate. If the new indeterminate is called z we formally carry out the computations in $R[k \times \mathbb{N}_0^n]/\langle a - z^a \mid a \in k\rangle$.

We can now use the exact same reasoning from Theorem 10.9 and Corollary 10.10 to show that the word problem for saturated binomial ideals is solvable in polynomial time.

Corollary 11.11 Let $g, f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ be binomials over a field k for some $n, s \in \mathbb{N}_{>0}$ such that $\langle f_1, \ldots, f_s\rangle \trianglelefteq k[x_1, \ldots, x_n]$ is a saturated ideal. We consider the $\mathbb{Z}$-vector space $\operatorname{span}(\{\phi(f_1), \ldots, \phi(f_s)\}) \subseteq k \times \mathbb{Z}^n$ using the notation from Theorem 11.9 with the operations

$$(a, u) + (b, v) := (ab, u + v)$$

and

$$c \cdot (a, u) := \underbrace{(a, u) + \cdots + (a, u)}_{c \text{ times}} = (a^c, cu)$$

for all $c \in \mathbb{Z}$, $a, b \in k$, and $u, v \in \mathbb{Z}^n$ and have

$$g \in \langle f_1, \ldots, f_s\rangle \Leftrightarrow \phi(g) \in \operatorname{span}(\{\phi(f_1), \ldots, \phi(f_s)\})$$

Corollary 11.12 Let $g, f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ be binomials in a polynomial ring over the field k for some $n, s \in \mathbb{N}_{>0}$ such that $\langle f_1, \ldots, f_s\rangle \trianglelefteq k[x_1, \ldots, x_n]$ is a saturated binomial ideal. There is an algorithm that decides whether $g \in \langle f_1, \ldots, f_s\rangle$ in polynomial time.

Using cellular decomposition and the certificates discussed in Section 11.1 we finally get the following corollary.

Corollary 11.13 Let k be a field. The radical word problem of non-pure binomial ideals $I \trianglelefteq k[x_1, \ldots, x_n]$ for $n \in \mathbb{N}_{>0}$ is **coNP**-complete.

All proofs may be carried out analogously to the case of pure binomial ideals by simply replacing the $\mathbb{Z}$-vector space to generalize the results from Section 11.2.

Part V

Radicals of Commutative Thue Systems

12 Term Replacement Systems

12.1 Grammars and Term Replacement Systems

Term replacement systems are a common construction in theoretical computer science. The idea of these systems is to describe instances of problems by some words and to describe when different instances of the problem have the same solution. This is done by defining that two words are equivalent if and only if they can be transformed into each other by applying a sequence of certain given basic operations. These basic operations are usually set up in a way such that it is easy to prove, that they transform problem instances to other problem instances with the same solution. Applying multiple of these basic operations allows for easily verifiable proofs that two problem instances have the same solution. We can state this concept formally by defining a term replacement system.

Definition 12.1 A term replacement system, formal grammar, or just grammar is a tuple $G = (N, \Sigma, \mathcal{P}, S)$ where N is a set of nonterminal symbols, Σ is a set of terminal symbols, $\mathcal{P}$ is a finite set of productions of the form $\alpha \to \beta$ for some $\alpha, \beta \in (N \cup \Sigma)^*$, and $S \in N$ is the start symbol.

When N, Σ, and S are clear from context we also call $\mathcal{P}$ the term replacement system instead of $G = (N, \Sigma, \mathcal{P}, S)$. To shorten notation we will usually write $\Gamma := N \cup \Sigma$. To avoid confusion we usually denote terminal symbols by digits or lowercase letters and nonterminal symbols by uppercase letters.

The descriptions of the problem instances consist of the terminal symbols Σ. The nonterminal symbols N are used as helper symbols for the productions. The start symbol S is used to define which problem instances should be accepted by our term replacement system. The productions $\mathcal{P}$ are the basic operations to transform problem descriptions into other ones with the same solution.

We can use the productions $\mathcal{P}$ to derive words from each other and define an equivalence relation between them.

Definition 12.2 A word $\alpha \in \Gamma^*$ derives a word $\beta \in \Gamma^*$ in one step modulo a term replacement system $G = (N, \Sigma, \mathcal{P}, S)$ – denoted by $\alpha \Rightarrow_{1,G} \beta$ or $\alpha \Rightarrow_{1,\mathcal{P}} \beta$ – if and only if there are $\alpha', \beta', \gamma, \delta, \in \Gamma^*$ with $\alpha = \gamma\alpha'\delta, \beta = \gamma\beta'\delta$, and $\alpha' \to \beta' \in \mathcal{P}$.

For $i \in \mathbb{N}_{>0}$ with $i \geq 2$ a word $\alpha \in \Gamma^*$ derives a word $\beta \in \Gamma^*$ in i steps modulo G – denoted by $\alpha \Rightarrow_{i,G} \beta$ or $\alpha \Rightarrow_{i,\mathcal{P}} \beta$ – if and only if there is a $\gamma \in \Gamma^*$ such that $\alpha \Rightarrow_{1,G} \gamma$ and $\gamma \Rightarrow_{i-1,G} \beta$. To make the definition complete, we say that a word $\alpha \in \Gamma^*$ derives a word $\beta \in \Gamma^*$ in zero steps modulo G – denoted by $\alpha \Rightarrow_{0,G} \beta$ or $\alpha \Rightarrow_{0,\mathcal{P}} \beta$ – if and only if $\alpha = \beta$.

A word $\alpha \in \Gamma^*$ derives a word $\beta \in \Gamma^*$ modulo G – denoted by $\alpha \Rightarrow_G \beta$ or $\alpha \Rightarrow_{\mathcal{P}} \beta$ – if and only if there is an $i \in \mathbb{N}_0$ such that $\alpha \Rightarrow_{i,G} \beta$. Two words $\alpha, \beta \in \Gamma^*$ are equivalent modulo G – denoted by $\alpha \equiv_G \beta$ or $\alpha \equiv \mathcal{P}\beta$ – if and only if $\alpha \Rightarrow_G \beta$ and $\beta \Rightarrow_G \alpha$.

A problem instance $\alpha \in \Sigma^*$ is considered to be accepted by the grammar G if it can be derived from the start symbol S. The language accepted by a grammar $G = (N, \Sigma, \mathcal{P}, S)$ is the set

$$\{\alpha \in \Sigma^* \,|\, S \Rightarrow_G \alpha\} \subseteq \Sigma^*$$

As an example consider the set M of all positive integers, that leave a remainder of 5 modulo 8. Each integer is described by its binary representation without leading zeros. Thus, we have $\Sigma = \{0, 1\}$. The grammar $G = (N, \Sigma, \mathcal{P}, S)$ with $N = \{S, T\}$ and

$$\mathcal{P} = \{S \to 101, S \to T101, T \to T0, T \to T1, T \to 1\}$$

accepts exactly the set M. The binary representation of integers contained in M is characterized by the fact that it starts with 1 and ends with 101. This is enforced by the only productions $S \to 101$ and $S \to T101$ of the start symbol S where the symbol T represents a word that starts with 1. The other productions allow the derivation of an arbitrary word starting with 1 from T. This makes all integers from M derivable from S. For instance 101101 is accepted by G because

$$S \Rightarrow_{1,G} T101 \Rightarrow_{1,G} T1101 \Rightarrow_{1,G} T01101 \Rightarrow_{1,G} 101101$$

which implies $S \Rightarrow_{4,G} 101101$ and $S \Rightarrow_G 101101$.

To decide whether a word $\alpha \in \Sigma^*$ is accepted by a grammar G, we can construct a Turing Machine that systematically tries all possible sequences of productions starting at S and stops if the word created by those productions is α. This Turing Machine would accept all words contained in G in a finite amount of time. For words, that are not created in G, the machines runs infinitely long. One can show that this is essentially the only way to decide whether a word is accepted by a general term replacement system using a deterministic Turing Machine [DSW94].

A decision problem $L \subseteq \{0, 1\}^*$, for which there is a Turing Machine, that given a word contained in L accepts after a finite amount of time and given a word not contained in L rejects after a finite amount of time or runs infinitely long, is called a semi-decidable problem and the Turing Machine is said to recognize the problem. Recall that we required a Turing Machine that decides a language $L \subseteq \{0, 1\}^*$ to stop after a finite amount of time for all inputs,

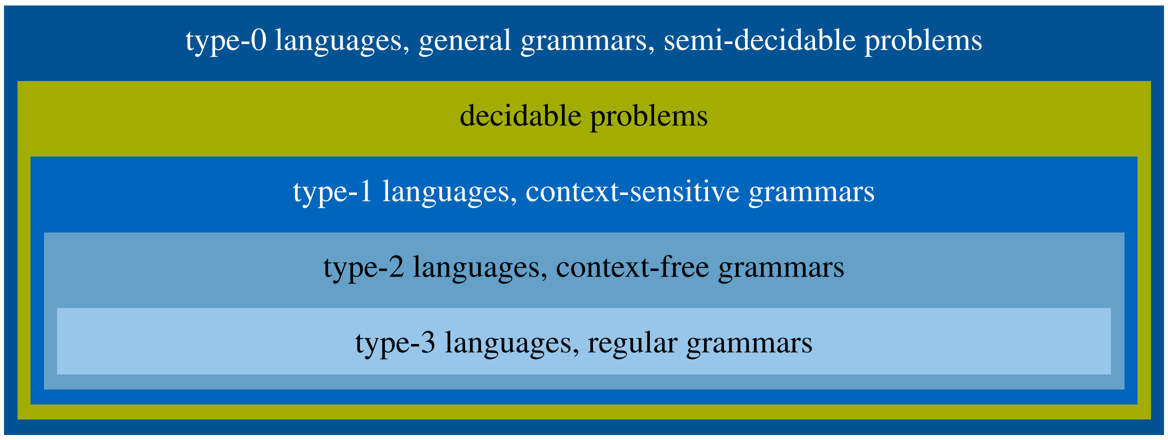

type-0 languages, general grammars, semi-decidable problems
decidable problems
type-1 languages, context-sensitive grammars
type-2 languages, context-free grammars
type-3 languages, regular grammars

Figure 12.3: A visualization of the Chomsky Hierarchy. When grammars are denoted in the diagram we refer to the languages accepted by them.

while Turing Machines semi-deciding L may run infinitely long on inputs not contained in L. Thus, the set of decision problems that are accepted by a term replacement system is called the set of recursively enumerable or semi-decidable problems and is denoted by **RE**.

The semi-decidable problems are also called type-0 languages in the Chomsky hierarchy. This hierarchy is named after the American linguist Noam Chomsky and describes different types of term replacement systems [Cho56]. A visualization of the levels of the Chomsky Hierarchy can be found in Figure 12.3.

Type-1 languages or context-sensitive grammars are described by grammars that replace only one symbol at a time, where that symbol also needs to be nonterminal. A language is context-sensitive if and only if it is is accepted by a term replacement system where all productions have the form $\alpha T\beta \to \alpha\gamma\beta$ for some $\alpha,\beta,\gamma \in \Gamma^*$ and $T \in N$. These grammars are called context-sensitive, because it is only allowed to replace T by γ in the presence of the context of α and β. The languages accepted by context-sensitive grammars can be decided by deterministic Turing Machines by starting at a word and exploring all possible productions backwards. Thus, the type-1 languages are decidable [DSW94].

Type-2 languages are accepted by so-called context-free grammars. They are a special case of context-sensitive grammars whose productions are not allowed to depend on the context. Instead of the context-sensitive productions $\alpha T\beta \to \alpha\gamma\beta$ for some $\alpha,\beta,\gamma \in \Gamma^*$ and $T \in N$, for context-free grammars all productions must have the form $T \to \gamma$ for some $\gamma \in \Gamma^*$ and $T \in N$, i.e. $\alpha = \beta = \epsilon$. The languages accepted by context-free grammars can be recognized by so-called non-deterministic pushdown automatons.

Another important special case of formal grammars are the regular grammars which accept the type-3 languages. These grammars are defined by productions of the form $T \to a$, $T \to aU$, or $T \to \epsilon$ for some $T, U \in N$ and $a \in \Sigma$. The languages accepted by these grammars can be recognized by restricted Turing Machines, that are not allowed to write to the work tapes and cannot move their head to the left. Turing Machines with this restriction are also called finite-state machines. For instance, the language of all positive integers that have a remainder 5 modulo 8 as discussed above is a regular language, because there is a regular grammar accepting this language.

12.2 Thue Systems

The productions as used above are also sometimes called semi-Thue productions in honor of the Norwegian mathematician Axel Thue. He introduced the theory of term replacement systems already in 1910 [Thu10, Thu14], but his work was not widely recognized until much later. This was mainly due to the publication in German language with a generic title in a small journal [Tho10]. Term replacement systems are therefore also called semi-Thue systems. When discussing semi-Thue systems, we generally assume that all symbols are terminal symbols, i.e. $N = \emptyset$. Also, if not stated otherwise, we assume the starting symbol to be the empty word $S = \epsilon$. Thus, a semi-Thue system is usually completely characterized by its set of productions $\mathcal{P}$. The set of terminal symbols Σ is implicitly known as the set of all symbols appearing in the productions. As in the last section we will therefore call $\mathcal{P}$ the semi-Thue system.

We will see in this section that commutative Thue systems are equivalent to pure binomial ideals as defined in Chapter 9. To achieve this, we will define commutative Thue systems first.

Definition 12.4 Let $\mathcal{P}$ be a semi-Thue system. $\mathcal{P}$ is a Thue system if and only if for each production $\alpha \to \beta \in \mathcal{P}$ for some $\alpha, \beta \in \Gamma^*$ we also have $\beta \Rightarrow_{\mathcal{P}} \alpha$. $\mathcal{P}$ is commutative if and only if for all $a, b \in \Gamma$ we have $ab \equiv_{\mathcal{P}} ba$.

When defining Thue systems we usually denote the productions as $\alpha \equiv \beta$ for some $\alpha, \beta \in \Gamma^*$ which means $\alpha \to \beta$ and $\beta \to \alpha$. For commutative Thue system we can add the productions $a \equiv b$ for all $a, b \in \Gamma$ to $\mathcal{P}$ without changing the commutative Thue system. Those productions are usually omitted and should implicitly be included when speaking about commutative Thue systems.

We will now map each commutative Thue system to a pure binomial ideal to establish an equivalence between commutative Thue systems and pure binomial ideals.

Definition 12.5 Let $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$ be a finite set with $n \in \mathbb{N}_0$ and let

$$\mathcal{P} = \left\{\alpha_i \equiv \beta_i \,|\, i \in \{1, \ldots, s\}\right\}$$

be a commutative Thue system for $s \in \mathbb{N}_{>0}$ and $\alpha_i, \beta_i \in \Sigma^*$ for $i \in \{1, \ldots, s\}$. Let $\Phi : \Sigma^* \to \mathbb{N}_0^n$ be the Parikh mapping, i.e. the i-th entry of $\Phi(\gamma)$ is the number of occurrences of σ_i in γ for all $\gamma \in \Sigma^*$ and $i \in \{1, \ldots, n\}$. For each ring R we define the polynomial ideal

$$\mathcal{I}_R(\mathcal{P}) := \left\langle \underline{x}^{\Phi(\alpha_i)} - \underline{x}^{\Phi(\beta_i)} \,\middle|\, i \in \{1, \ldots, s\} \right\rangle \trianglelefteq R\,[x_1, \ldots, x_n]$$

For each commutative Thue system $\mathcal{P}$ and for each ring R the polynomial ideal $\mathcal{I}_R(\mathcal{P})$ is obviously generated by pure binomials and therefore a pure binomial ideal. For each pure binomial ideal $I \trianglelefteq R\,[x_1, \ldots, x_n]$ for some ring R and $n \in \mathbb{N}_{>0}$ we can also find a commutative Thue system $\mathcal{P}$ over the set of terminal symbols $\Sigma = \{x_1, \ldots, x_n\}$. This is achieved by computing a generating set of I, that consists of pure binomials only, and for each generator $\underline{x}^u - \underline{x}^v \in I$ for some $u, v \in \mathbb{N}_0^n$ we add the equivalence $\underline{x}^u \equiv \underline{x}^v$ to $\mathcal{P}$, where the monomials are understood as words over Σ. For this commutative Thue system we obviously get the same set of generators when we compute $\mathcal{I}_R(\mathcal{P})$. Thus, we have maps in both directions between the set of pure binomials over $R\,[x_1, \ldots, x_n]$ and the set of commutative Thue systems over $\Sigma = \{x_1, \ldots, x_n\}$, that are inverse to each other.

We will now see that we can translate the pure binomials contained in the polynomial ring to equivalences of the corresponding commutative Thue system without losing information about their containment in the pure binomial ideal or commutative Thue system, respectively.

Theorem 12.6 ([MM82]) For all commutative Thue Systems $\mathcal{P}$ we have

$$\underline{x}^{\Phi(\gamma)} - \underline{x}^{\Phi(\delta)} \in \mathcal{I}_{\mathbb{Z}}(\mathcal{P}) \Leftrightarrow \underline{x}^{\Phi(\gamma)} - \underline{x}^{\Phi(\delta)} \in \mathcal{I}_{\mathbb{Q}}(\mathcal{P}) \Leftrightarrow \gamma \equiv_{\mathcal{P}} \delta$$

for all $\gamma, \delta \in \Sigma^*$.

Proof We will show three implications to prove this theorem. First,

$$\underline{x}^{\Phi(\gamma)} - \underline{x}^{\Phi(\delta)} \in \mathcal{I}_{\mathbb{Z}}(\mathcal{P}) \Rightarrow \underline{x}^{\Phi(\gamma)} - \underline{x}^{\Phi(\delta)} \in \mathcal{I}_{\mathbb{Q}}(\mathcal{P})$$

obviously holds.

Now assume that we have $\underline{x}^{\Phi(\gamma)} - \underline{x}^{\Phi(\delta)} \in \mathcal{I}_{\mathbb{Q}}(\mathcal{P})$. Without loss of generality let

$$\mathcal{I}_Q(\mathcal{P}) = \left\langle \underline{x}^{a_1} - \underline{x}^{b_1}, \ldots, \underline{x}^{a_s} - \underline{x}^{b_s} \right\rangle \trianglelefteq \mathbb{Q}[x_1, \ldots, x_n]$$

for some $a_1, \ldots, a_s, b_1, \ldots, b_s \in \mathbb{N}_0^n$ for $n := |\Gamma|$ and some $s \in \mathbb{N}_{>0}$. with Theorem 4.10 there are $i_1, \ldots, i_t \in \{1, \ldots, s\}$ for some $t \in \mathbb{N}_{>0}$ and $m_1, \ldots, m_t \in \mathbb{M}_{\mathbb{Q}[x_1,\ldots,x_n]}$, $c'_1, \ldots, c'_t \in \mathbb{Q}$ such that

$$\underline{x}^{\Phi(\gamma)} - \underline{x}^{\Phi(\delta)} = \sum_{j=1}^{t} c'_j m_j (\underline{x}^{a_{i_j}} - \underline{x}^{b_{i_j}})$$

By clearing denominators and switching the order of some a_i and b_i we can assume that there are $d, c_1, \ldots, c_t \in \mathbb{N}_{>0}$ with

$$d\underline{x}^{\Phi(\gamma)} - d\underline{x}^{\Phi(\delta)} = \sum_{j=1}^{t} c_j m_j (\underline{x}^{a_{i_j}} - \underline{x}^{b_{i_j}})$$

Since $\underline{x}^{\Phi(\gamma)}$ appears on the left-hand side of this equations it also has to appear on the right-hand side. Without loss of generality assume that $\underline{x}^{\Phi(\gamma)} = m_1\underline{x}^{a_{i_1}}$. We can subtract $\underline{x}^{\Phi(\gamma)}$ from the equation and get

$$m_1\underline{x}^{b_{i_1}} + (d-1)\underline{x}^{\Phi(\gamma)} - d\underline{x}^{\Phi(\delta)} = \sum_{j=1}^{t} c_j m_j (\underline{x}^{a_{i_j}} - \underline{x}^{b_{i_j}}) - m_1(\underline{x}^{a_{i_1}} - \underline{x}^{b_{i_1}})$$

In terms of the commutative Thue system, we have that $\gamma = m_1\Phi^{-1}(a_{i_1}) \equiv_{\mathcal{P}} m_1\Phi^{-1}(b_{i_1})$. Repeating this argument with $m_1\underline{x}^{b_{i_1}}$ finally leads to $\underline{x}^{\Phi(\delta)}$ as the coefficients on the right-hand side shrink in every step. In terms of the commutative Thue system $\mathcal{P}$ we get an equivalence of γ and δ. We have just shown $\underline{x}^{\Phi(\gamma)} - \underline{x}^{\Phi(\delta)} \in \mathcal{I}_{\mathbb{Q}}(\mathcal{P}) \Rightarrow \gamma \equiv_{\mathcal{P}} \delta$.

Now we show the last missing implication $\gamma \equiv_{\mathcal{P}} \delta \Rightarrow \underline{x}^{\Phi(\gamma)} - \underline{x}^{\Phi(\delta)} \in \mathcal{I}_{\mathbb{Z}}(\mathcal{P})$. Let $\mathcal{P}$ be generated by $\mathcal{P} = \{\alpha_i \equiv \beta_i \,|\, i \in \{1, \ldots, s\}\}$ for some $s \in \mathbb{N}_{>0}$. $\gamma \equiv_{\mathcal{P}} \delta$ means that there is a chain of derivations

$$\gamma = \epsilon_1\alpha_{i_1} \equiv_{\mathcal{P}} \epsilon_1\beta_{i_1} = \epsilon_2\alpha_{i_2} \equiv_{\mathcal{P}} \epsilon_2\beta_{i_2} = \cdots = \epsilon_t\beta_{i_t} = \delta$$

for some $t \in \mathbb{N}_{>0}$, $\epsilon_1, \ldots, \epsilon_t \in \Gamma^*$, and $i_1, \ldots, i_t \in \{1, \ldots, s\}$. This implies

$$\underline{x}^{\Phi(\gamma)} - \underline{x}^{\Phi(\delta)} = \sum_{j=1}^{t} \underline{x}^{\Phi(\epsilon_i)} \left(\underline{x}^{\Phi(\alpha_i)} - \underline{x}^{\Phi(\beta_i)}\right) \in \mathcal{I}_{\mathbb{Z}}(\mathcal{P})$$

which concludes the proof. □

This means that commutative Thue systems and pure binomial ideals over the integers or rationals are basically the same. This equivalence is a powerful tool since it allows to use techniques from term replacement systems for pure binomial ideals and vice versa. The result from Theorem 7.13, that the word problem for polynomial ideals is **EXPSPACE**-complete, was also done by translating a term replacement system to a pure binomial ideal [MM82].

All operations we discussed in Section 4.6, that are closed under pure binomial ideals, can be transferred to commutative Thue systems. We can map commutative Thue systems to pure binomial ideals, execute the operations, and then translate the resulting pure binomial ideals back to commutative Thue systems. This works for instance for sums, radicals, and quotients as well as saturations with monomials.

Many operations can also be interpreted in terms of commutative Thue systems. The sum of two commutative Thue systems is a system that allows for productions from both systems.

The saturation of a commutative Thue system is a system where $\alpha\beta \equiv_{\mathcal{P}} \alpha\gamma$ implies $\beta \equiv_{\mathcal{P}} \gamma$ for all $\alpha, \beta, \gamma \in \Gamma^*$. Only radicals have no immediate interpretation in terms of commutative Thue systems yet, but we will introduce one in the subsequent chapters.

13 Radicals of Term Replacement Systems

13.1 Algorithms for Computing the Radical of Pure Binomial Ideals

In this section we will present a new algorithm to compute radicals of commutative Thue systems. We will develop this algorithm in terms of pure binomial ideals since they are equivalent as seen in Theorem 12.6. We will make sure that our algorithm just uses binomials as intermediate steps since Theorem 12.6 allows for the mapping of pure binomials only.

The definition of radical ideals as presented in Definition 8.1 yields an idea of an algorithm to compute the radical of a polynomial ideal.

Algorithm 13.1 Compute the radical of a polynomial ideal using roots.

```
Input: $f_1, \ldots, f_s \in R[x_1, \ldots, x_n]$ polynomials over a Noetherian ring $R$ for some $n, s \in \mathbb{N}_{>0}$
Output: $\sqrt{\langle f_1, \ldots, f_s \rangle}$
1: $I := \langle f_1, \ldots, f_s \rangle$
2: while there are $g \in R[x_1, \ldots, x_n]$ and $r \in \mathbb{N}_{>0}$ such that $g^r \in I$ and $g \notin I$ do
3:     $I := I + \langle g \rangle$
4: end while
5: return $I$
```

Note that finding the polynomial g in Algorithm 13.1 is still a non-constructive step. All other steps can be carried out using Gröbner Basis techniques as presented earlier. There could be an infinite amount of polynomials to be added to the ideal when computing the radical ideal, but Algorithm 13.1 still terminates after a finite time.

Theorem 13.2 Algorithm 13.1 terminates after a finite number of steps for all inputs.

Proof Consider the polynomial ideals that the variable I contains during the execution time of the algorithm in order. The chain of ideals is clearly growing since we just add new generators to our ideal. As R is Noetherian, we know that $R[x_1, \ldots, x_n]$ is Noetherian too. Thus, at some point the algorithm will not enlarge the ideal anymore, since we only add polynomials to the ideal if they were not contained in the ideal yet. This means that the algorithm needs to terminate once the variable I does not change anymore. □

Likewise, Algorithms 13.3, 13.4, and 13.7 that will be presented in this section terminate after a finite amount of time for every input.

For pure binomial ideals we know that the radical is still a pure binomial ideal as discussed in Chapter 9 [ES96]. Therefore, it is enough to find roots that are pure binomials only and we can simplify our algorithm.

Algorithm 13.3 Compute the radical of a pure binomial ideal using roots.

```
Input: $f_1, \ldots, f_s \in R[x_1, \ldots, x_n]$ pure binomials over a Noetherian ring $R$ for some $n, s \in \mathbb{N}_{>0}$
Output: $\sqrt{\langle f_1, \ldots, f_s \rangle}$
1: $I := \langle f_1, \ldots, f_s \rangle$
2: while there is a pure binomial $g \in R[x_1, \ldots, x_n]$ and $r \in \mathbb{N}_{>0}$ such that $g^r \in I$ and $g \notin I$ do
3:     $I := I + \langle g \rangle$
4: end while
5: return $I$
```

We note that Algorithm 13.3 would still hold for non-pure binomial ideals if g is also chosen as a possibly non-pure binomial.

Algorithm 13.3 still involves polynomials g^r that are possibly no pure binomials. Therefore, this algorithm cannot be used when dealing with term replacement systems since we only have a map from pure binomials to productions. Other polynomials do not have known matching objects in the context of term replacement systems.

Another way to simplify Algorithm 13.1 is to limit the order of the root. Considering square roots is already enough to compute the radical ideal.

Algorithm 13.4 Compute the radical of a polynomial ideal using square roots.

```
Input: $f_1, \ldots, f_s \in R[x_1, \ldots, x_n]$ polynomials over a Noetherian ring $R$ for some $n, s \in \mathbb{N}_{>0}$
Output: $\sqrt{\langle f_1, \ldots, f_s \rangle}$
1: $I := \langle f_1, \ldots, f_s \rangle$
2: while there is $g \in R[x_1, \ldots, x_n]$ such that $g^2 \in I$ and $g \notin I$ do
3:     $I := I + \langle g \rangle$
4: end while
5: return $I$
```

Theorem 13.5 Algorithm 13.4 is correct.

Proof Algorithm 13.4 obviously returns a subset of Algorithm 13.1, given the same input, since Algorithm 13.1 may always choose $r = 2$. Since both algorithms should compute the same ideal we only need to prove that all polynomials computed by Algorithm 13.1 will also be computed by Algorithm 13.4.

Let $g \in R[x_1, \ldots, x_n]$ and $r \in \mathbb{N}_{>0}$ be chosen in the loop of Algorithm 13.1. Let $t \in \mathbb{N}_{>0}$ be the smallest positive integer with $2^t \geq r$. This implies $g^{2^t} \in I$ since g^{2^t} is a multiple of g^r. Algorithm 13.4 may now add $g^{2^{t-1}}$ to I if it is not already contained since its square is also contained in I. Similarly, it will add $g^{2^{t-2}}, g^{2^{t-3}}, \ldots, g^{2^0} = g$ to I. This means that all polynomials added to I by Algorithm 13.1 will also eventually be added by Algorithm 13.4. □

Although the square of a pure binomial is not a pure binomial, we can check whether it is contained in a pure binomial ideal using pure binomials only.

Theorem 13.6 Let $n \in \mathbb{N}_{>0}$, $\alpha, \beta \in \mathbb{N}_0^n$ and $I \trianglelefteq k[x_1, \ldots, x_n]$ be a pure binomial ideal over a field k with $\operatorname{char}(k) = 0$. Then we have

$$\left(\underline{x}^\alpha - \underline{x}^\beta\right)^2 \in I \Leftrightarrow \underline{x}^{2\alpha} - \underline{x}^{\alpha+\beta} \in I \text{ and } \underline{x}^{2\beta} - \underline{x}^{\alpha+\beta} \in I$$

Proof The direction "⇐" clearly holds by adding both polynomials.

For the other direction "⇒" we consider the normal forms of the three monomials $\underline{x}^{2\alpha}$, $\underline{x}^{\alpha+\beta}$, and $\underline{x}^{2\beta}$ modulo I. As shown in Theorem 9.9, the three terms can be partitioned into sets such that all monomials of terms in a set have the same normal form and their coefficients sum up to zero because the sum of the terms is contained in I. Since no coefficient of the terms is zero none of them can be in a singleton set, thus all of them have to be in the same set. In particular all three monomials need to have the same normal form. This directly implies the claim. □

Thus, the combination of Algorithm 13.3 and Algorithm 13.4 considering only squares of pure binomials could be executed using pure binomials only.

Algorithm 13.7 Compute the radical of a pure binomial ideal using square roots.

```
Input: $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ pure binomials over a field $k$ for some $n, s \in \mathbb{N}_{>0}$,
Output: $\sqrt{\langle f_1, \ldots, f_s \rangle}$
1: $I := \langle f_1, \ldots, f_s \rangle$
2: while there are $\alpha, \beta \in \mathbb{N}_0^n$ such that $\underline{x}^{2\alpha} - \underline{x}^{\alpha+\beta} \in I$, $\underline{x}^{2\beta} - \underline{x}^{\alpha+\beta} \in I$, and $\underline{x}^\alpha - \underline{x}^\beta \notin I$ do
3:     $I := I + \left\langle \underline{x}^\alpha - \underline{x}^\beta \right\rangle$
4: end while
5: return $I$
```

The proof of Algorithm 13.4 cannot be adopted to prove Algorithm 13.7 from Algorithm 13.3 since the intermediate results may not be pure binomials. Thus, the proof is more involved.

13.2 Proof of the Algorithm

We will need the following theorem for the proof, that gives an insight into how the equivalence classes of monomials relate in a pure binomial ideal.

Theorem 13.8 Let $I \trianglelefteq k[x_1, \dots, x_n]$ be a pure binomial ideal over a field k with $n, r \in \mathbb{N}_{>0}$ and $\gamma, \delta \in \mathbb{N}_0^n$ such that $g := \underline{x}^\gamma - \underline{x}^\delta \in R[x_1, \dots, x_n]$ is a pure binomial with $g^r \in I$ and not all monomials of g^r have the same normal form modulo I. Then there is $t \in \mathbb{N}_{>0}$ with $t > r$ such that the set of normal forms of monomials of g^t is smaller than the set of normal forms of monomials of g^r, i.e.

$$\left|\left\{\overline{\underline{x}^{j\gamma}\underline{x}^{(t-j)\delta}}^I \mid j \in \{1, \dots, t\}\right\}\right| < \left|\left\{\overline{\underline{x}^{j\gamma}\underline{x}^{(r-j)\delta}}^I \mid j \in \{1, \dots, r\}\right\}\right|$$

Proof To make notation easier we define maps

$$\begin{aligned} \nu_i : \{0, \dots, i\} &\to R[x_1, \dots, x_n] \\ j &\mapsto \underline{x}^{j\gamma}\underline{x}^{(i-j)\delta} \end{aligned}$$

for all $i \in \mathbb{N}_{>0}$ and

$$\begin{aligned} \mu : \mathbb{N}_{>0} &\to \mathbb{N}_{>0} \\ i &\mapsto \left|\left\{\overline{\nu_i(j)}^I \mid j \in \{0, \dots, i\}\right\}\right| \end{aligned}$$

With these maps the conjecture reads that if $\mu(r) > 1$ there is $t \in \mathbb{N}_{>0}$ with $t > r$ and $\mu(t) < \mu(r)$. We will think about $\nu_t(0), \dots, \nu_t(t)$ as the t-th level of monomials, where monomials having the same normal form modulo I form equivalence classes.

For $i, j \in \{0, \dots, r\}$ we know that if $\overline{\nu_r(i)}^I = \overline{\nu_r(j)}^I$ then also $\overline{\nu_{r+1}(i)}^I = \overline{\nu_{r+1}(j)}^I$, since we just multiplied both monomials by $\underline{x}^\delta$ before taking the normal form. This means that the $r+1$-th level of monomials is a copy of the r-th level with an additional monomial $\nu_{r+1}(r+1)$ and possibly more relations, which means fewer equivalence classes. Theorem 9.9 implies that $\overline{\nu_{r+1}(r+1)}^I = \overline{\nu_{r+1}(k)}^I$ for some other $k \in \{1, \dots, r\}$, since the coefficient $(-1)^{r+1}$ of the term $\nu_{r+1}(r+1)$ is not zero. Together this shows that $\mu(r+1) \le \mu(r)$.

This argument can be used multiple times to prove that $\mu(t) \le \mu(r)$ for all $t \in \mathbb{N}_{>0}$ with $t > r$, and it remains to show that there is $t \in \mathbb{N}_{>0}$ with $t > r$ such that equality is not possible.

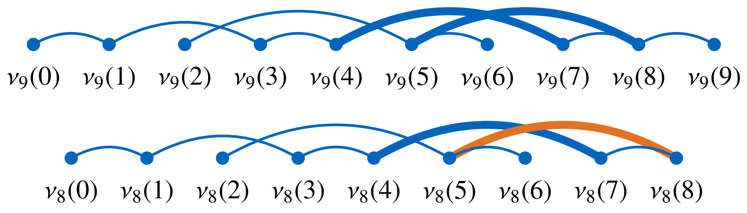

$v_9(0)$ $v_9(1)$ $v_9(2)$ $v_9(3)$ $v_9(4)$ $v_9(5)$ $v_9(6)$ $v_9(7)$ $v_9(8)$ $v_9(9)$
$v_8(0)$ $v_8(1)$ $v_8(2)$ $v_8(3)$ $v_8(4)$ $v_8(5)$ $v_8(6)$ $v_8(7)$ $v_8(8)$

Figure 13.9: A sketch for the proof of Theorem 13.8 with $r = 8$, $\mu(8) = 2$, $\mu(9) = 1$, $p = 4$, and $q = 7$. The orange line is assumed to not be contained in the pure binomial ideal.

We assume the opposite, namely $\mu(t) = \mu(r)$ for all $t \in \mathbb{N}_{>0}$ with $t > r$, and will infer a contradiction.

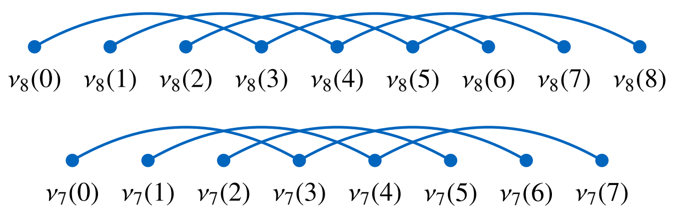
We claim that if $\overline{v_r(p)}^I = \overline{v_r(q)}^I$ for some $p, q \in \{0, \dots, r-1\}$ it is also true that $\overline{v_r(p+1)}^I = \overline{v_r(q+1)}^I$. Recall that for $i, j \in \{0, \dots, r\}$ the equation $\overline{v_r(i)}^I = \overline{v_r(j)}^I$ implied $\overline{v_{r+1}(i)}^I = \overline{v_{r+1}(j)}^I$, which was already enough to prove $\mu(r+1) \le \mu(r)$, since the $r+1$-th level of the equivalences between monomials is an exact copy of the r-th level except for the additional $v_{r+1}(r+1)$. If $\overline{v_r(p+1)}^I \ne \overline{v_r(q+1)}^I$ the equation $\overline{v_{r+1}(p+1)}^I = \overline{v_{r+1}(q+1)}^I$ was not already contained in the relations we used for this argument. But multiplying the monomials $v_r(p)$ and $v_r(q)$ by $\underline{x}^\gamma$ and taking the normal form implies exactly that equation $\overline{v_{r+1}(p+1)}^I = \overline{v_{r+1}(q+1)}^I$ which would therefore merge two equivalence classes. Therefore, we would get $\mu(r+1) < \mu(r)$ which is a contradiction to our assumption. A visualization of this argument can be found in Figure 13.9. Similarly, one can prove that if $\overline{v_r(p)}^I = \overline{v_r(q)}^I$ for some $p, q \in \{1, \dots, r\}$ we also have $\overline{v_r(p-1)}^I = \overline{v_r(q-1)}^I$.

Let $m \in \mathbb{N}_{>0}$ be the smallest positive integer such that there is $i \in \{0, \dots, r-m\}$ with $\overline{v_r(i)}^I = \overline{v_r(i+m)}^I$. We will show that this means for $i, j \in \{0, \dots, r\}$ that $\overline{v_r(i)}^I = \overline{v_r(j)}^I$ if and only if m is a divisor of $i - j$, i.e. $m \,|\, i - j$.

Without loss of generality assume $i \le j$. For the direction "$\Leftarrow$" we need to shift the relation that realizes the distance m several times as described above to establish equality between $\overline{v_r(i)}^I, \overline{v_r(i+m)}^I, \dots \overline{v_r(j)}^I$. For the other direction "$\Rightarrow$" note that a relation, that has length $m' \in \mathbb{N}_{>0}$ with $m \nmid m'$, can be combined by shifting with the relation of length m to a relation of length $\gcd(m, m') < m$, which is a contradiction to the fact that m is the length of a shortest relation. Note that $m > 1$ since we required $\mu(r) > 1$ and we have $m = \mu(r)$.

By multiplying the monomials by $\underline{x}^\gamma$ or $\underline{x}^\delta$ we get the same structure for all higher levels: For all $t \in \mathbb{N}_{>0}$ with $t > r$ and $i, j \in \{0, \dots, t\}$ we have $\overline{v_t(i)}^I = \overline{v_t(j)}^I$ if and only if $m \,|\, i - j$. The integer m may not be smaller for higher levels since we assumed $\mu(t) = \mu(r)$ for all $t \in \mathbb{N}_{>0}$ with $t > r$. A visualization of this structure is displayed in Figure 13.10.

Now let $t \in \mathbb{N}_{>0}$ be the smallest prime with $t > r$. Because $v_t(1)$ has the non-zero coefficient $-t$, Theorem 9.9 implies that it may not be in a singleton equivalence class which means $m < t$.



$v_8(0)$ $v_8(1)$ $v_8(2)$ $v_8(3)$ $v_8(4)$ $v_8(5)$ $v_8(6)$ $v_8(7)$ $v_8(8)$
$v_7(0)$ $v_7(1)$ $v_7(2)$ $v_7(3)$ $v_7(4)$ $v_7(5)$ $v_7(6)$ $v_7(7)$

Figure 13.10: An example of the structure of equivalence relations as used in the proof of Theorem 13.8 with $r = 7$ and $m = 3$.

Earlier on we showed $m > 1$, so we know that $m \in \{2, \dots, t-1\}$. In particular, this means that for the prime t we have $m \nmid t$, which is equivalent to $\overline{v_t(0)}^I \neq \overline{v_t(t)}^I$.

All coefficients $(-1)^i\binom{t}{i}$ of the monomials $v_t(i)$ for $i \in \{1, \dots, t-1\}$ are multiples of t since t is prime. Thus, the sum of the coefficients of terms of g^t having the same normal form as $v_t(0)$ is one plus a multiple of t which is obviously not zero. This is a contradiction to Theorem 9.9 and therefore our assumption $\mu(t) = \mu(r)$ was wrong. With the arguments above this proves $\mu(t) < \mu(r)$ as it was required to show. □

This theorem in particular implies that the number of equivalence classes of powers of pure binomial ideals shrinks with growing exponents and reaches one eventually.

Corollary 13.11 Let $g \in k[x_1, \dots, x_n]$ be a pure binomial over a field k with $n \in \mathbb{N}_{>0}$ and $\gamma, \delta \in \mathbb{N}_0^n$ such that $g = \underline{x}^\gamma - \underline{x}^\delta$ and let $r \in \mathbb{N}_{>0}$ with $g^r \in I$. There is a $t \in \mathbb{N}_{>0}$, $t \geq r$ such that all terms of g^t are equivalent modulo I.

Proof Theorem 13.8 states that if $\mu(r) \neq 1$ we can find $t_1 \in \mathbb{N}_{>0}$ with $t_1 > r$ such that $\mu(t_1) < \mu(r)$. If $\mu(t_1)$ is still not equal to one we can find $t_2 \in \mathbb{N}_{>0}$ with $t_2 > t_1$ such that $\mu(t_2) < \mu(t_1)$ using Theorem 13.8 again. Using this theorem multiple times we can finally conclude that there is $t \in \mathbb{N}_{>0}$ with $t \geq r$ such that $\mu(t) = 1$. □

We are now ready to prove that Algorithm 13.7 is correct.

Theorem 13.12 Algorithm 13.7 is correct.

Proof We will use the notation of the functions v and μ from the proof of Theorem 13.8 in this proof too.

Together with Theorem 13.6 it is immediately clear that every binomial computed by Algorithm 13.7 will also be computed by Algorithm 13.3 when setting $r = 2$. Therefore, we only need to show that every binomial added to I by Algorithm 13.3 will also be added by Algorithm 13.7, similar to the proof of Algorithm 13.4 in Theorem 13.5.

Let $g \in R[x_1, \dots, x_n]$ be a pure binomial with $\gamma, \delta \in \mathbb{N}_0^n$ such that $g = \underline{x}^\gamma - \underline{x}^\delta$ and $r \in \mathbb{N}_{>0}$ such that $g^r \in I$ and $f \notin I$ as chosen in the loop of Algorithm 13.3. Corollary 13.11 states that there is $t \in \mathbb{N}_{>0}$ with $t \geq r$ such that $\mu(t) = 1$. Let $s \in \mathbb{N}_{>0}$ be the smallest positive integer such that $2^s \geq t \geq r$. With the arguments above we also have $g^{2^s} \in I$ and $\mu(2^s) = 1$.

We will now describe how Algorithm 13.7 adds g to I. First, we will see how $g^{2^{s-1}}$ is added to I. Let $i \in \left\{0, \dots, 2^{s-1} - 1\right\}$. Algorithm 13.7 can choose $\alpha := i\gamma + \left(2^{s-1} - i\right)\delta$ and $\beta := (i+1)\,\gamma + \left(2^{s-1} - i - 1\right)\delta$. This is allowed since

$$\begin{aligned}
\underline{x}^{2\alpha} - \underline{x}^{\alpha+\beta} &= \underline{x}^{2i\gamma+(2^s-2i)\delta} & -\underline{x}^{(2i+1)\gamma+(2^s-2i-1)\delta} &\in I \qquad \text{and}\\
\underline{x}^{2\beta} - \underline{x}^{\alpha+\beta} &= \underline{x}^{(2i+2)\gamma+(2^s-2i-2)\delta} & -\underline{x}^{(2i+1)\gamma+(2^s-2i-1)\delta} &\in I
\end{aligned}$$

which is true because all monomials of g^{2^s} have the same normal form modulo I. Therefore, Algorithm 13.7 can add

$$\underline{x}^\alpha - \underline{x}^\beta = \nu_{2^{s-1}}(i) - \nu_{2^{s-1}}(i+1)$$

to I if it is not already contained. After these operations for all $i \in \{0, \dots, 2^s - 1\}$ we have that all monomials of $g^{2^{s-1}}$ have the same normal form modulo I. This argument can be repeated to show that $g^{2^{s-2}}, g^{2^{s-3}}, \dots, g^{2^0} = g$ will also be included in I. □

We passed to bigger exponents several times in this proof. Therefore, it is interesting to find a bound for the maximum exponent of binomials that need to be considered.

13.3 Experimental Degree Bounds

We note that we did not observe huge changes in the degree in practice. The crucial step is to find $t \in \mathbb{N}_{>0}$ with $t \geq r$ and $\mu(t) = 1$ given a binomial $g \in R\,[x_1, \dots, x_n]$ with $g^r \in I$ and $g \notin I$. We only observed examples where $t \in \{r, r+1\}$ was sufficient. This is because with $g^r \in I$ and Theorem 13.6, we need to find a partition of the monomials $\nu_r(i)$ for $i \in \{0, \dots, r\}$ into sets such that the sum of the coefficients of the elements of each set is zero. Note that we only need to consider the combinatorics of the coefficients $(-1)^i\binom{r}{i}$, which are independent of the properties of the polynomial ring like the number of indeterminates.

In particular, such partitions may occur if they are symmetric. We say that a partition is symmetric if r is odd and for all $i \in \left\{0, \dots, \frac{r-1}{2}\right\}$ the monomials $\nu_r(i)$ and $\nu_r(r-i)$ are in the same set or if r is even and all monomials are in the same set. The coefficients of such pairs of monomials sum up to zero which directly implies the conditions of Theorem 13.6. The partition, where all monomials are in the same set, is always an example of a symmetric partition.

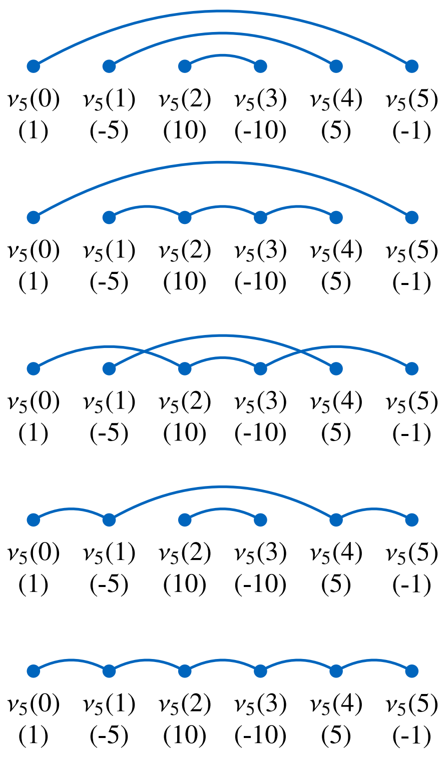

$v_5(0)$ $v_5(1)$ $v_5(2)$ $v_5(3)$ $v_5(4)$ $v_5(5)$
(1) (-5) (10) (-10) (5) (-1)
$v_5(0)$ $v_5(1)$ $v_5(2)$ $v_5(3)$ $v_5(4)$ $v_5(5)$
(1) (-5) (10) (-10) (5) (-1)
$v_5(0)$ $v_5(1)$ $v_5(2)$ $v_5(3)$ $v_5(4)$ $v_5(5)$
(1) (-5) (10) (-10) (5) (-1)
$v_5(0)$ $v_5(1)$ $v_5(2)$ $v_5(3)$ $v_5(4)$ $v_5(5)$
(1) (-5) (10) (-10) (5) (-1)
$v_5(0)$ $v_5(1)$ $v_5(2)$ $v_5(3)$ $v_5(4)$ $v_5(5)$
(1) (-5) (10) (-10) (5) (-1)

Figure 13.13: All examples found by the algorithm given in Listing A1.2 for $r = 5$. The coefficients of the monomials are given below in brackets.

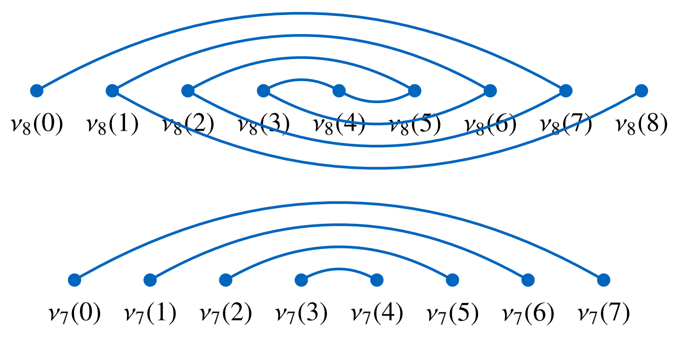


Figure 13.15: A sketch for the proof of Theorem 13.14 with $r = 7$.

For symmetric partitions we can prove that all monomials at the next level already have the same normal form modulo I.

Theorem 13.14 Let $I \trianglelefteq k[x_1, \dots, x_n]$ be a pure binomial ideal over a field k, $n \in \mathbb{N}_{>0}$, $r \in \mathbb{N}_{>0}$ odd, and $\gamma, \delta \in \mathbb{N}_0^n$ such that $g := \underline{x}^\gamma - \underline{x}^\delta \in R[x_1, \dots, x_n]$ is a pure binomial with $g^r \in I$. For all $i \in \{0, \dots, r\}$ let

$$\overline{\underline{x}^{i\gamma+(r-i)\delta}}^I = \overline{\underline{x}^{(r-i)\gamma+i\delta}}^I$$

Then all monomials of g^{r+1} have the same normal form modulo I.

Proof We will use the notation of the functions v and μ from the proof of Theorem 13.8 in this proof again. Using this notation we claim that if $\overline{v_r(i)}^I = \overline{v_r(r-i)}^I$ for all $i \in \{0, \dots, r\}$ then $\mu(r+1) = 1$.

Multiplying the monomials in the equations given above by $\underline{x}^\gamma$ results in $\overline{v_{r+1}(i)}^I = \overline{v_{r+1}(r-i)}^I$ for all $i \in \{0, \dots, r\}$ and multiplying them by $\underline{x}^\delta$ results in $\overline{v_{r+1}(i+1)}^I = \overline{v_{r+1}(r-i+1)}^I$ for all $i \in \{0, \dots, r\}$. A visualization of these relations can be found in Figure 13.15.

With these relations we have

$$\overline{v_{r+1}(0)}^I = \overline{v_{r+1}(r)}^I = \overline{v_{r+1}(2)}^I = \overline{v_{r+1}(r-2)}^I = \dots = \overline{v_{r+1}\left(\frac{r+1}{2}\right)}^I$$

and

$$\overline{v_{r+1}(r+1)}^I = \overline{v_{r+1}(1)}^I = \overline{v_{r+1}(r-1)}^I = \overline{v_{r+1}(3)}^I = \dots = \overline{v_{r+1}\left(\frac{r+1}{2}\right)}^I$$

which directly shows $\mu(r+1) = 1$. □

We tried to compute non-symmetric partitions for small r with the brute-force approach in Listing A1.2 written in C++ that can be found in Section A.2 in the appendix.

We ran the code for $r \leq 13$, but we only found few non-symmetric partitions. The output of the code is shown in Listing 13.16 and Listing 13.17. The majority of the partitions found are symmetric. Thus, we omitted printing them for $r > 6$.

Listing 13.16: Output of the algorithm at Listing A1.2 for argument 6.

```
checking level 1
0 0 (symmetric)
checking level 2
0 0 0 (symmetric)
checking level 3
0 0 0 0 (symmetric)
0 1 1 0 (symmetric)
checking level 4
0 0 0 0 0 (symmetric)
checking level 5
0 0 0 0 0 0 (symmetric)
0 0 1 1 0 0 (symmetric)
0 1 0 0 1 0 (symmetric)
0 1 1 1 1 0 (symmetric)
0 1 2 2 1 0 (symmetric)
checking level 6
0 0 0 0 0 0 0 (symmetric)
```

Listing 13.17: Output of the algorithm at Listing A1.2 for argument 13. Symmetric partitions are omitted.

```
checking level 1
checking level 2
checking level 3
checking level 4
checking level 5
checking level 6
checking level 7
checking level 8
0 0 1 1 0 0 1 0 0
0 0 1 0 0 1 1 0 0
checking level 9
checking level 10
checking level 11
checking level 12
checking level 13
0 0 0 1 1 0 0 1 1 0 0 0 0 0
0 0 0 0 0 1 1 0 0 1 1 0 0 0
0 0 0 2 2 1 1 2 2 1 1 0 0 0
0 0 1 1 1 0 0 1 1 0 0 1 0 0
0 0 1 2 2 0 0 2 2 0 0 1 0 0
0 0 1 0 0 1 1 0 0 1 1 1 0 0
```

22	0	0	1	2	2	1	1	2	2	1	1	1	0	0
23	0	0	1	0	0	2	2	0	0	2	2	1	0	0
24	0	0	1	1	1	2	2	1	1	2	2	1	0	0
25	0	0	1	3	3	2	2	3	3	2	2	1	0	0
26	0	1	0	1	1	0	0	1	1	0	0	0	1	0
27	0	1	0	2	2	0	0	2	2	0	0	0	1	0
28	0	1	0	0	0	1	1	0	0	1	1	0	1	0
29	0	1	0	2	2	1	1	2	2	1	1	0	1	0
30	0	1	0	0	0	2	2	0	0	2	2	0	1	0
31	0	1	0	1	1	2	2	1	1	2	2	0	1	0
32	0	1	0	3	3	2	2	3	3	2	2	0	1	0
33	0	1	1	1	1	0	0	1	1	0	0	1	1	0
34	0	1	1	2	2	0	0	2	2	0	0	1	1	0
35	0	1	1	0	0	1	1	0	0	1	1	1	1	0
36	0	1	1	2	2	1	1	2	2	1	1	1	1	0
37	0	1	1	0	0	2	2	0	0	2	2	1	1	0
38	0	1	1	1	1	2	2	1	1	2	2	1	1	0
39	0	1	1	3	3	2	2	3	3	2	2	1	1	0
40	0	1	2	1	1	0	0	1	1	0	0	2	1	0
41	0	1	2	2	2	0	0	2	2	0	0	2	1	0
42	0	1	2	3	3	0	0	3	3	0	0	2	1	0
43	0	1	2	0	0	1	1	0	0	1	1	2	1	0
44	0	1	2	2	2	1	1	2	2	1	1	2	1	0
45	0	1	2	3	3	1	1	3	3	1	1	2	1	0
46	0	1	2	0	0	2	2	0	0	2	2	2	1	0
47	0	1	2	1	1	2	2	1	1	2	2	2	1	0
48	0	1	2	3	3	2	2	3	3	2	2	2	1	0
49	0	1	2	0	0	3	3	0	0	3	3	2	1	0
50	0	1	2	1	1	3	3	1	1	3	3	2	1	0
51	0	1	2	2	2	3	3	2	2	3	3	2	1	0
52	0	1	2	4	4	3	3	4	4	3	3	2	1	0

The first non-symmetric partitions are found at $r = 8$. These partitions are visualized in Figure 13.18. Other non-symmetric partitions exist for $r = 13$. For all non-symmetric partitions we found, that in the next level the normal forms of all monomials are the same. This means that we do not know any examples where binomials need to be considered in Algorithm 13.7 that have substantially bigger degree than in Algorithm 13.3.

13.4 A Formal Degree Bound

We will now show a general bound for the maximum degree that needs to be considered for Algorithm 13.7.

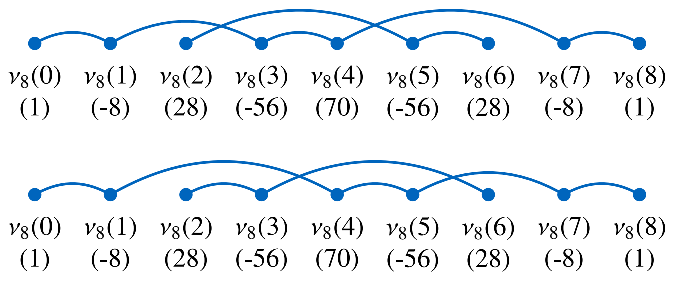

$v_8(0)$ $v_8(1)$ $v_8(2)$ $v_8(3)$ $v_8(4)$ $v_8(5)$ $v_8(6)$ $v_8(7)$ $v_8(8)$
(1) (-8) (28) (-56) (70) (-56) (28) (-8) (1)
$v_8(0)$ $v_8(1)$ $v_8(2)$ $v_8(3)$ $v_8(4)$ $v_8(5)$ $v_8(6)$ $v_8(7)$ $v_8(8)$
(1) (-8) (28) (-56) (70) (-56) (28) (-8) (1)

Figure 13.18: Both non-symmetric examples found by the algorithm given in Listing A1.2 for $r = 8$. The coefficients of the monomials are given below in brackets.

Theorem 13.19 Let k be a field, $n, s \in \mathbb{N}_{>0}$ and $f_1, \dots, f_s \in k[x_1, \dots, x_n]$ be pure binomials with $d := \max(\deg(f_1), \dots, \deg(f_s)) \in \mathbb{N}_{>0}$. Let $g \in \sqrt{\langle f_1, \dots, f_s \rangle}$ be a pure binomial. All terms of g^t with

$$t := 2d^n \Big(\log 2d^n + \log\log 2d^n \Big) \in \mathbb{N}_{>0}$$

have the same normal form modulo $\langle f_1, \dots, f_s \rangle$.

Proof Since $g \in \sqrt{\langle f_1, \dots, f_s \rangle} \subseteq k[x_1, \dots, x_n]$ there is $r \in \mathbb{N}_{>0}$ with $g^r \in \langle f_1, \dots, f_s \rangle$ and we assume that r is the smallest positive integer with this property, that is at least 3.

We discussed degree bounds in Section 8.2. The bounds mentioned there show that

$$r \leq d^n$$

for all $g \in \sqrt{\langle f_1, \dots, f_s \rangle}$ [Bro87, Kol88, Som99]. All degree bounds from this section were shown for general polynomial ideals and not only for pure binomial ideals.

Let $t' \in \mathbb{N}_{>0}$ be chosen as in the proof of Corollary 13.11 as the $(r-1)$-th prime bigger than r. Jacques Hadamard and Charles Jean de La Vallée Poussin independently proved the famous Prime Number Theorem in 1896 [Had96, dLVP96]. For this theorem let $\pi(m)$ be the number of primes in the set $\{1, \dots, m\}$. The Prime Number Theorem states that

$$\pi(m) \in \Theta\left(\frac{m}{\log m}\right)$$

In particular we have that the m-th prime is smaller than

$$m \log m + m \log\log m$$

for $m \geq 6$ [BS96]. The $2r$-th prime is obviously at least the $(r-1)$-th prime greater than r, so we can compute

$$t' < 2r(\log 2r + \log\log 2r)$$

There are also better bounds for this theorem like

$$\pi(m) < \frac{m}{\log m}\left(1 + \frac{1}{\log m} + \frac{2.51}{\log^2 m}\right)$$

for $m \geq 355991$ given by Pierre Dusart in 1998 [Dus98]. Since we are only interested in the asymptotic behavior we will state easier results that are asymptotically the same.

Putting the bounds for t' and r together we get

$$t' < 2d^n\left(\log 2d^n + \log\log 2d^n\right) = t$$

Corollary 13.11 implies that all terms of $g^{t'}$ are equivalent modulo $\langle f_1, \ldots, f_s\rangle$ and with $t' < t$ this proves the claim. □

We can now apply this degree bound to Algorithm 13.7 to find the maximum degree the algorithm needs to consider.

Corollary 13.20 Let k be a field, $n, s \in \mathbb{N}_{>0}$ and $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ be pure binomials with $d := \max(\deg(f_1), \ldots, \deg(f_s)) \in \mathbb{N}_{>0}$. Algorithm 13.7 adds every pure binomial $g \in \sqrt{\langle f_1, \ldots, f_s\rangle}$ if it considers binomials with degree up to

$$2d^n\left(\log 2d^n + \log\log 2d^n\right)\deg(g) + 1$$

Proof Let $g \in \sqrt{\langle f_1, \ldots, f_s\rangle}$ be a pure binomial as chosen by Algorithm 13.7 and let t be chosen as in Theorem 13.19. We have proven for Theorem 13.12 that Algorithm 13.7 only needs to consider levels up to $2t$ which means polynomials of degree up to $2t\deg(g)$.

The factor 2 can be removed by not going up to the next power of 2 first and halving the degree multiple times, but instead using the algorithm to halve the degree if possible and otherwise add one. This would result in a bound for the degree of the polynomials to be considered of $t\deg(g) + 1$.

Inserting the definition of t from Theorem 13.19 shows the claim. □

13.5 Radicals of Commutative Thue Systems

With the equivalence of commutative Thue systems and pure binomial ideals established in Section 12.2, the proof in Theorem 13.12 of Algorithm 13.7 and the degree bound from Theorem 13.19, we are now able to define radicals of commutative Thue systems, which we will do in this section.

We saw that commutative Thue systems are an important tool for proofs on pure binomial ideals. For instance the proof in [MM82] uses the equivalence of both constructions. There are numerous open questions on degree bounds of polynomial ideals and in particular on radical ideals. Unfortunately, the definition of radical ideals on polynomial ideals involves powers of polynomials, which are no pure binomials in general, even if the initial polynomial was a pure binomial. As binomials with more than two terms have no equivalent object in terms of commutative Thue systems it, was not possible to define radicals of commutative Thue systems before. Algorithm 13.7 involves pure binomials only, thus we can define radicals of commutative Thue systems now.

As this definition contains some order of commutative Thue systems by size we first need to tell when a commutative Thue system contains another one.

Definition 13.21 Let $\mathcal{P}$ and $\mathcal{Q}$ be commutative Thue systems over an alphabet Σ. $\mathcal{P}$ is contained in $\mathcal{Q}$ or smaller than $\mathcal{Q}$ if and only if for all $\alpha,\beta \in \Sigma^*$ with $\alpha \equiv_{\mathcal{P}} \beta$ we also have $\alpha \equiv_{\mathcal{Q}} \beta$.

Now we are finally able to define radicals of commutative Thue systems.

Definition 13.22 Let $\mathcal{P}$ be a commutative Thue system over an alphabet Σ. $\mathcal{P}$ is radical if and only if for all $\alpha,\beta \in \Sigma^*$ with $\alpha^2 \equiv_{\mathcal{P}} \alpha\beta$ and $\beta^2 \equiv_{\mathcal{P}} \alpha\beta$ we have $\alpha \equiv_{\mathcal{P}} \beta$. The radical $\sqrt{\mathcal{P}}$ of the commutative Thue system $\mathcal{P}$ is the smallest commutative Thue System $\mathcal{Q}$ that is radical and contains $\mathcal{P}$.

We will see that the radical of a commutative Thue system, i.e. the smallest radical commutative Thue system containing another one, is well-defined in the following theorem, because the pure binomial ideal corresponding to a commutative Thue system is unique.

Theorem 13.23 Let $\mathcal{P}$ be a commutative Thue system over an alphabet Σ. We have

$$\mathcal{I}_{\mathbb{Q}}\left(\sqrt{\mathcal{P}}\right) = \sqrt{\mathcal{I}_{\mathbb{Q}}(\mathcal{P})}$$

and

$$\mathcal{I}_{\mathbb{Z}}\left(\sqrt{\mathcal{P}}\right) = \sqrt{\mathcal{I}_{\mathbb{Z}}(\mathcal{P})}$$

Proof In the definition of radicals of commutative Thue systems we required that $\alpha^2 \equiv_{\mathcal{P}} \alpha\beta$ and $\beta^2 \equiv_{\mathcal{P}} \alpha\beta$ implies $\alpha \equiv_{\mathcal{P}} \beta$ for all $\alpha,\beta \in \Sigma^*$. As the equivalent implications on pure binomial ideals $\underline{x}^{2u} - \underline{x}^{u+v} \in \mathcal{I}_{Q}(\mathcal{P})$ and $\underline{x}^{2v} - \underline{x}^{u+v} \in \mathcal{I}_{Q}(\mathcal{P})$ implies $\underline{x}^{u} - \underline{x}^{v} \in \mathcal{I}_{Q}(\mathcal{P})$ for all $u, v \in \mathbb{N}_0^{|\Sigma|}$ and $\underline{x}^{2u} - \underline{x}^{u+v} \in \mathcal{I}_{Z}(\mathcal{P})$ and $\underline{x}^{2v} - \underline{x}^{u+v} \in \mathcal{I}_{Z}(\mathcal{P})$ implies $\underline{x}^{u} - \underline{x}^{v} \in \mathcal{I}_{Z}(\mathcal{P})$ for all $u, v \in \mathbb{N}_0^{|\Sigma|}$ also hold for radical ideals, as shown in Theorem 13.6, the direction "$\subseteq$" of both claims must hold.

Algorithm 13.7 and its proof in Theorem 13.12 show that this implication is already enough to compute the radical of a pure binomial ideal which proves the other direction "$\supseteq$" of both claims. □

We can also directly translate Algorithm 13.7 to commutative Thue systems to find an algorithm that computes radicals of commutative Thue systems.

Algorithm 13.24 Compute the radical of a commutative Thue system using a non-constructive step.

Input: an alphabet Σ and $\alpha_1, \ldots, \alpha_s, \beta_1, \ldots, \beta_s \in \Sigma^*$ for some $s \in \mathbb{N}_{>0}$
Output: the radical of the commutative Thue system generated by the equivalences

$$\left\{\alpha_i \equiv \beta_i \,|\, i \in \{1, \ldots, s\}\right\}$$

```
1: let $\mathcal{P}$ be the commutative Thue system generated by the equivalences
       $$\left\{\alpha_i \equiv \beta_i \,|\, i \in \{1, \ldots, s\}\right\}$$
2: while there are $\alpha, \beta \in \Sigma^*$ such that $\alpha^2 \equiv_{\mathcal{P}} \alpha\beta$, $\beta^2 \equiv_{\mathcal{P}} \alpha\beta$, but $\alpha \not\equiv_{\mathcal{P}} \beta$ do
3:     add $\alpha \equiv \beta$ to $\mathcal{P}$
4: end while
5: return $\mathcal{P}$
```

Note that during Algorithm 13.24 we always want $\mathcal{P}$ to be a valid commutative Thue system. In particular Step 3 of Algorithm 13.24 means adding $\alpha \equiv \beta$ to the set of generators of $\mathcal{P}$. This implies that in general more equivalences are added to $\mathcal{P}$, because we can add symbols to both sides of an equivalence and substitute other equivalences on one of the sides to obtain new equivalences contained in $\mathcal{P}$.

The algorithm would actually also compute the exact same result if we just added $\alpha \equiv \beta$ in Step 3 and omitted all consequences of that equivalence because those consequences would be added to $\mathcal{P}$ later on in the loop in Step 2. The problem is that the set of equivalences in a commutative Thue system is infinite in general while the set of generators is always finite. We always want to run our algorithms in finite time, because otherwise we are not able to execute them and measuring time and space complexities would not make sense. Thus, we need to work on generators of $\mathcal{P}$ to make the algorithm finish in a finite amount of time. Working on generators on the other hand makes the checks in Step 2 harder. In this line we have to solve three word problems on commutative Thue systems which is an **EXPSPACE**-complete problem as seen in Theorem 9.7.

Also, Algorithm 13.24 is still a non-constructive algorithm. The loop in Step 2 requires to find $\alpha, \beta \in \Sigma^*$ subject to some constraints without giving an explicit way to find these words. To make the algorithm constructive we will give a list of all equivalences that possibly need to be added to $\mathcal{P}$ and check them all in a suitable order. To do so, we need some degree bounds on the generators of the radical as our degree bound from Theorem 13.19 involves that degree.

In Section 14.3 we present a constructive version of the algorithm, analyze its time and space bounds, and present an improved version of it.

14 Degree Bounds for Radical Ideals

14.1 Upper Bounds

The degree bounds in Theorem 13.19 depend on the degrees of the elements in a basis of the radical of a polynomial ideal. In this chapter we will collect lower and upper complexity bounds for the degree of generators of radical ideals.

For the upper bound we will consider an algorithm to actually compute the radical ideal. Such an algorithm was presented for zero-dimensional polynomial ideals in Algorithm 8.17. This algorithm enables us to read off degree bounds of the results.

To do so, we just need a bound for the size of the elements added to the basis of I, but since these are just elements of an elimination ideal of I we can use the degree bound found by Thomas W. Dubé presented in Theorem 7.11 [Dub90]. This was already done before, for instance in Johannes Mittmann's Diploma thesis [Mit08].

Theorem 14.1 Let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a polynomial ideal over a perfect field k with $\dim(I) = 0$, $n \in \mathbb{N}_{>0}$, and let the degree of all generators of some generating set of I be at most $d \in \mathbb{N}_{>0}$. There is a generating set of $\sqrt{I}$ such that the degrees of all its generators are at most

$$2\left(\frac{d^2}{2} + d\right)^{2^{n-1}}$$

Proof With Theorem 8.15 we know that there is a generating set of $\sqrt{I}$ such that all its generators are either the given generators of I or the square-free forms of the generators of some elimination ideals of I. The former have a degree that is bounded by d as required, whereas the latter have a degree bounded by

$$2\left(\frac{d^2}{2} + d\right)^{2^{n-1}}$$

which is implied by the bound of Thomas W. Dubé presented in Theorem 7.11 [Dub90] and the fact that the degree of the square-free part of a polynomial is at most the degree of the original polynomial. □

We generalized Algorithm 8.17 for polynomial ideals with higher dimension in Chapter 8. The degree bound presented above can be transferred to polynomial ideals with higher dimensions by analyzing Algorithm 8.18 which was done by Santiago Laplagne [Lap06].

Theorem 14.2 ([Lap06]) Let $I \trianglelefteq k[x_1, \ldots, x_n]$ be a polynomial ideal over a perfect field k, $n \in \mathbb{N}_{>0}$, and let the degree of all generators of some generating set of I consisting of s polynomials be at most $d \in \mathbb{N}_{>0}$. There is a generating set of $\sqrt{I}$ such that the degrees of all its generators are at most

$$(sd)^{2^{cn^2+n-1}}$$

for some universal constant $c \in \mathbb{N}_{>0}$.

For binomial ideals a similar double exponential bound can be deduced by considering the cellular decomposition

$$\sqrt{I} = \bigcap_{\Delta \subseteq \{x_1, \ldots, x_n\}} \left(I_\Delta : \left(\prod_{x_i \in \Delta} x_i \right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle \right)$$

of I. As seen in Algorithm 4.35 the quotients can be computed using an elimination ideal each and the intersection of all ideals can be computed by exactly one elimination ideal using Lagrange polynomials. Using Dubé's bound we see that the degrees of the generators of the $I_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^\infty + \langle \{x_i \mid x_i \notin \Delta\} \rangle$ are double exponential in d. The intersection results in a degree double-exponential in something double-exponential in d, which is double-exponential in d again.

14.2 Lower Bounds

On the other hand, we can construct ideals having a Gröbner Basis with elements having exponents, that are exponential in the encoding size of a given generating set, with a simple construction. Let $s \in \mathbb{N}_{>0}$ and consider the ideal

$$I_s := \left\langle x_1 - x_2^2, x_2 - x_3^2, \ldots, x_{s-1} - x_s^2 \right\rangle \trianglelefteq k[x_1, \ldots, x_s]$$

over a field k with $\operatorname{char}(k) = 0$ and a lexicographic term ordering with $x_1 \geq x_2 \geq \cdots \geq x_s$. A pure binomial of the form $x_i - x_j^c$ for some $i, j \in \{1, \ldots, s\}$, $c \in \mathbb{N}_0$ is contained in I_s if and only if $i \leq j$ and $c = 2^{j-i}$. In particular we have

$$x_j - x_s^{2^{s-j}} \in I_s$$

for all $j \in \{1, \ldots, s-1\}$. These elements form also a Gröbner Basis of I_s with respect to the lexicographic term order defined above. The reduced Gröbner Basis of I_s is the set

$$\left\{ x_j - x_s^{2^{s-j}} \mid j \in \{1, \ldots, s-1\} \right\}$$

or $\{0\}$ for $s = 1$, which can easily be checked using for instance Buchberger's Criterion.

The exponent of $x_1 - x_s^{2^{s-1}}$ is clearly exponential in the encoding size of the generating set of I_s, which is linear in s. It is also worth mentioning that this example is even a pure binomial ideal. An implementation of this example can be found in Listing 14.3.

Listing 14.3: An implementation of I_n using Macaulay2 [GS].

```
expGBIdeal = s -> (
    R = QQ[x_1..x_s, MonomialOrder => Lex];
    ideal for i from 2 to s list x_i^2-x_(i-1)
)
```

This example has a rather easy form, but there are more complicated pure binomial ideals that even have Gröbner Bases with elements having double exponential degree in the encoding size of the generating set [MM82]. The advantage of this example is its simple structure and the fact, that the ideal is radical and saturated.

Theorem 14.4 Let $s \in \mathbb{N}_{>0}$ and

$$I_s := \left\langle x_1 - x_2^2, x_2 - x_3^2, \ldots, x_{s-1} - x_s^2 \right\rangle \trianglelefteq R\left[x_1, \ldots, x_s\right]$$

be a pure binomial ideal over a ring R. The ideal I_s is radical and saturated, i.e.

$$I_s = \sqrt{I_s} = I_s : \left(\prod_{i=1}^{s} x_i\right)^{\infty}$$

Proof For $s = 1$ we have $I_s = \{0\}$ and the claim is obvious for this case. Thus we can assume $s \geq 2$.

I_s is a pure binomial ideal and therefore we can use the cellular decomposition introduced in Section 11.1

$$\sqrt{I_s} = \bigcap_{\Delta \subseteq \{x_1,\ldots,x_s\}} \left(\left(I_s\right)_\Delta : \left(\prod_{x_i \in \Delta} x_i\right)^{\infty} + \langle \{x_i \mid x_i \notin \Delta\} \rangle \right)$$

The cell for $\Delta = \{x_1, \ldots, x_n\}$ is the saturation of I_s. Therefore, together with the fact that I_s is contained in its radical, we know that

$$I_s \subseteq \sqrt{I_s} \subseteq I_s : \left(\prod_{i=1}^{s} x_i\right)^{\infty}$$

and it remains to show that the saturation of I_s is contained in I_s.

To show this, we need to characterize the polynomials contained in I_s. As we have seen above, the reduced Gröbner Basis of I_s with respect to a lexicographic term ordering $\geq$ with

$x_1 \geq x_2 \geq \cdots \geq x_s$ is

$$\left\{x_j - x_s^{2^{s-j}} \mid j \in \{1, \ldots, s-1\}\right\}$$

Reducing polynomials using this Gröbner Basis exchanges all variables to x_s, namely it replaces terms $\underline{x}^u$ for $u \in \mathbb{N}_0^s$ by

$$\overline{\underline{x}^u}^{I_s} = x_s^t$$

where $t = \sum_{i=1}^s 2^{s-i} u_i$.

When we reduce a polynomial $f \in R[x_1, \ldots, x_s]$ using this Gröbner Basis, we exchange all indeterminates to x_s and then check whether the remaining terms sum up to zero. We are not able to reduce terms containing the indeterminate x_s only since

$$\underbrace{\mathrm{LM}(I_s)}_{\langle x_1, x_2, \ldots, x_{s-1}\rangle} \cap R[x_s] = \{0\}$$

but all other indeterminates are removed. Thus, a polynomial $f \in R$ is contained in I_s if and only if the coefficients of all its monomials that reduce to the same power of x_s sum up to zero. For some $f \in R[x_1, \ldots, x_s]$ and $\beta \in \mathbb{N}_0^s$ this conditions holds for $\underline{x}^\beta f$ if and only if it holds for f, because both polynomials have the same coefficients and the normal forms of monomials of $\underline{x}^\beta f$ are the ones of f multiplied by $\sum_{i=1}^n 2^{s-i}\beta_i$. This means $\underline{x}^\beta f \in I_s \Leftrightarrow f \in I_s$ and implies

$$I_s = I_s : \left(\prod_{i=1}^s x_i\right)^\infty$$

which concludes the proof. □

Together we get the following theorem.

Theorem 14.5 There is a family $(I_s)_{s \in \mathbb{N}_{>0}}$ of radical and saturated pure binomial ideals with $I_s \trianglelefteq R[x_1, \ldots, x_n]$ and $n = s$ for $n, s \in \mathbb{N}_{>0}$ and the maximum degree of the generators of I_s being 2 such that there is an element in the Gröbner Basis of I_s with respect to a lexicographic term ordering having degree 2^{s-1}.

Now we want to combine several of the exponential chains presented above to get a pure binomial ideal, where the Gröbner Basis of the radical has elements with exponents, that are exponential in the encoding length of the input ideal. To do this, we take three such chains $x_1, \ldots, x_s$, $y_1, \ldots, y_s$, and $z_1, \ldots, z_s$ with the corresponding pure binomials $x_1 - x_2^2, \ldots, x_{s-1} - x_s^2$, $y_1 - y_2^2, \ldots, y_{s-1} - y_s^2$, and $z_1 - z_2^2, \ldots, z_{s-1} - z_s^2$ for $s \in \mathbb{N}_{>0}$. We take three additional indeterminates o, a, and b. o serves as a connection of the chains with the pure binomials $o - x_1$, $o - y_1$, and $o - z_1$. a and b form the new element in the radical with the pure binomials $x_s - a^2$, $y_s - ab$,

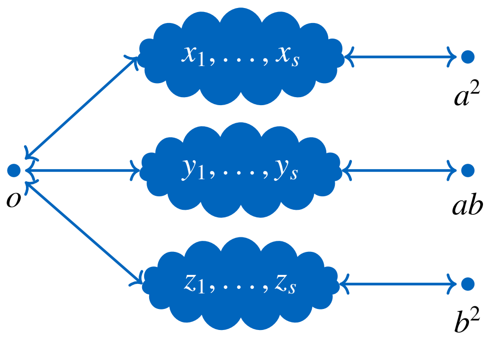


Figure 14.6: A visualization of J_s. Clouds represent ideals as used in Theorem 14.4.

and $z_s - b^2$. All in all, we consider the ideal

$$\begin{aligned} J_s := \Big\langle & x_1 - x_2^2, \dots, x_{s-1} - x_s^2, y_1 - y_2^2, \dots, y_{s-1} - y_s^2, \\ & z_1 - z_2^2, \dots, z_{s-1} - z_s^2, o - x_1, o - y_1, o - z_1, \\ & x_s - a^2, y_s - ab, z_s - b^2 \Big\rangle \\ \trianglelefteq\ & k[x_0, \dots, x_s, y_0, \dots, y_s, z_0, \dots, z_s, o, a, b] \end{aligned}$$

over a field k with $\operatorname{char}(k) = 0$ together with a lexicographical term ordering $\geq$ with

$$x_0 \geq \cdots \geq x_s \geq y_0 \geq \cdots \geq y_s \geq z_0 \geq \cdots \geq z_s \geq o \geq a \geq b$$

An implementation of these ideal can be found in Listing 14.7, a visualization in Figure 14.6.

Listing 14.7: An implementation of J_s using Macaulay2 [GS].

```
expRadIdeal = s -> (
    R := QQ[x_1..x_s, y_1..y_s, z_1..z_s, o, a, b, MonomialOrder =>
       Lex];
    L := (o-x_1, o-y_1, o-z_1, x_s-a^2, y_s-a*b, z_s-b^2);
    for i from 2 to s do (
        L = join(L, (x_i^2-x_(i-1), y_i^2-y_(i-1), z_i^2-z_(i-1)))
    );
    ideal L
);
```

Recall that with the findings above we have $x_1 - x_s^{2^{s-1}}, y_1 - y_s^{2^{s-1}}, z_1 - z_s^{2^{s-1}} \in J_s$. Together with the additional relations for o, a, and b we get

$$o - x_s^{2^{s-1}}, o - y_s^{2^{s-1}}, o - z_s^{2^{s-1}} \in J_s$$

and

$$x_s^{2^{s-1}} - a^{2^s}, y_s^{2^{s-1}} - a^{2^{s-1}} b^{2^{s-1}}, z_s^{2^{s-1}} - b^{2^s} \in J_s$$

Thus we have

$$o - a^{2^s}, o - a^{2^{s-1}} b^{2^{s-1}}, o - b^{2^s} \in J_s$$

which implies

$$a^{2^s} - 2a^{2^{s-1}} b^{2^{s-1}} + b^{2^s} = \left(a^{2^{s-1}} - b^{2^{s-1}}\right)^2 \in J_s$$

The chains $x_1, \dots, x_s$, $y_1, \dots, y_s$, and $z_1, \dots, z_s$ are radical according to Theorem 14.4. The new variables only introduce the binomial $a^{2^{s-1}} - b^{2^{s-1}}$ and its multiples and sums with other elements to the radical of J_s. Thus, we have

$$\sqrt{J_s} = J_s + \left\langle a^{2^{s-1}} - b^{2^{s-1}} \right\rangle$$

The new binomial also needs to be included in a Gröbner Basis with respect to a monomial term order with respect to $x_1 \geq x_2 \geq \cdots \geq x_s \geq y_1 \geq y_2 \geq \cdots \geq y_s \geq z_1 \geq z_2 \geq \cdots \geq z_s \geq o \geq a \geq b$ which implies the following theorem.

Theorem 14.8 There is a family $(J_s)_{s \in \mathbb{N}_{>0}}$ of pure binomial ideals with $J_s \trianglelefteq R[x_1, \dots, x_n]$ and $n = 3(s+1)$ for $n, s \in \mathbb{N}_{>0}$ and the maximum degree of the generators of J_s being 2 such that there is an element in the Gröbner Basis of $\sqrt{J_s}$ with respect to a lexicographic term order having degree 2^{s-1}.

The upper and lower bounds for the degree of generators of the radical of a polynomial ideal presented here do not match. In the next section we will see that the upper bound implies that our new algorithms computes the radical of a pure binomial ideal asymptotically as fast as the fastest known algorithms. Thus, we conjecture that the lower bound can be improved to be double exponential. A corresponding example is not known yet.

14.3 Degree Bounds for Radicals of Commutative Thue Systems

We can now discuss the running time of Algorithm 13.24 and use the degree bounds from the last sections. Recall that Step 2 of Algorithm 13.24 still requires us to choose some words from Σ^* without presenting a way to find them. Using the degree bound from Theorem 14.2 we can limit the number of words, that need to be checked, to a finite set. This makes our algorithm constructive, i.e. we can indeed implement it. Also, a constructive version of the algorithm allows for the computation of running time and space bounds.

Algorithm 14.9 Compute the radical of a commutative Thue system using degree bounds.

Input: an alphabet Σ and $\alpha_1, \dots, \alpha_s, \beta_1, \dots, \beta_s \in \Sigma^*$ for some $s \in \mathbb{N}_{>0}$
Output: $\sqrt{\mathcal{P}}$ where $\mathcal{P}$ is the commutative Thue system generated by the equivalences

$$\left\{\alpha_i \equiv \beta_i \,|\, i \in \{1, \dots, s\}\right\}$$

```
1: let $\mathcal{P}$ be the commutative Thue system generated by the equivalences
       $$\left\{\alpha_i \equiv \beta_i \,|\, i \in \{1, \dots, s\}\right\}$$
2:
       $$m := 2d^n\left(\log 2d^n + \log\log 2d^n\right)(sd)^{2^{cn^2+n-1}} + 1$$
   where $d$ is the maximum length of the words $\alpha_1, \dots, \alpha_s, \beta_1, \dots, \beta_s \in \Sigma^*$, $n = |\Sigma|$, and
   $c \in \mathbb{N}_{>0}$ is some universal constant from Theorem 14.2
3: **for** each $\alpha, \beta \in \Sigma^*$ with $|\alpha|, |\beta| \leq m$ in descending order of $|\alpha| + |\beta| \in \mathbb{N}_0$ **do**
4:   **if** $\alpha^2 \equiv_{\mathcal{P}} \alpha\beta, \beta^2 \equiv_{\mathcal{P}} \alpha\beta$, but $\alpha \not\equiv_{\mathcal{P}} \beta$ **then**
5:     add $\alpha \equiv \beta$ to $\mathcal{P}$
6:   **end if**
7: **end for**
8: **return** $\mathcal{P}$
```

First of all we need to prove, that this algorithm indeed computes the radical of a commutative Thue system.

Theorem 14.10 Algorithm 14.9 is correct.

Proof Algorithm 14.9 is almost the same as Algorithm 13.24, but checks all $\alpha, \beta \in \Sigma^*$ up to some degree bound. This bound is exactly the one from Theorem 14.2 inserted into the degree bound from Corollary 13.20, which limits the degree of generators of radical polynomial ideals, and thus also the degree of radicals of commutative Thue systems. While Theorem 14.2 states that there is one set of generators achieving this degree bound, Algorithm 14.9 finds all equivalences up to this bound, which therefore need to include the generating set from Theorem 14.2.

Additionally, we need to make sure that in case that the checks in Step 4 of Algorithm 14.9 reject an equivalence at first, it will not happen that the same checks will accept the equivalence later on when $\mathcal{P}$ is bigger. Since $|\alpha^2| + |\alpha\beta| > |\alpha| + |\beta|$ and $|\beta^2| + |\alpha\beta| > |\alpha| + |\beta|$ for $\alpha \neq \epsilon$ and $\beta \neq \epsilon$ – else the equivalences are all identical – we have checked the equivalences used in Step 4 before the one that we currently check. That is why we visit the equivalences in descending order of $|\alpha| + |\beta| \in \mathbb{N}_0$. □

Now that we have a constructive algorithm, we can analyze its time and space consumption.

Theorem 14.11 Algorithm 14.9 runs in **EXPSPACE**.

Proof To iterate through all values of α and β we only need to store the exponent vectors of the words. Instead of saving these numbers themselves we can save a binary representation of them. The encoding length of the exponents is therefore only

$$\mathcal{O}(\log(m)) = \mathcal{O}(2^{n^2}\log(sd))$$

for each pair of words $\alpha, \beta \in \Sigma^*$.

We still have to compute the result of a word problem in Step 4 of Algorithm 14.9. This problem is **EXPSPACE**-complete as explained in Theorem 9.7.

The output itself can also be saved in exponential space, as we know that there are algorithms to compute the radical of a polynomial ideal in exponential space. Note that it is required to remove redundant equivalences from the generating set of $\mathcal{P}$ to achieve this space bound.

Thus, all parts discussed above and all other steps of the algorithm can be executed in exponential space, which means that in total we can compute the radical of a pure binomial ideal or commutative Thue system in exponential space. □

EXPSPACE is also the best complexity class of known algorithms to compute the radical of a polynomial ideal. Our algorithm matches this complexity class. For random examples it is usually still much slower since it always tests all possible equivalences up to the worst possible degree bound, which is not needed for all inputs. In contrast to other algorithms, Algorithm 14.9 does not improve its speed on inputs that are not the worst case. Thus, our algorithm is not faster than known algorithms in practice, but allows a computation with binomial intermediate results and still has the same asymptotic running time and space bounds for the worst case.

14.4 Adjustments of the Closure Operation

As explained in Section 14.2 above, the known lower and upper bounds of the degree of generators of the radical given a polynomial ideal do not match. This means, that it is still possible that this bound can be drastically improved. In case of an exponential bound we would only require polynomial space to list all possible $\alpha, \beta \in \Sigma^*$ in Algorithm 14.9.

While an improved degree bound for the generators would reduce the number of equivalences to be checked, we can also try to make the check of each equivalence faster. The expensive task here is to solve the word problems in Step 4 of Algorithm 14.9. We can adjust those word problems to make the same commutative Thue system appear every time instead of different systems. This makes the algorithm run much faster in practice as the expensive step of solving a word problem over a pure binomial ideal or commutative Thue system is to compute a Gröbner Basis of the system. This Gröbner Basis can be reused if the commutative Thue system always stays the same.

To do so, we relax the closure of commutative Thue systems to check, whether words are equivalent, more easily. The problem here is that we need to make sure, that after relaxing the closure conditions, we still work on finite generating sets and thus achieve finite running times. The easiest way to relax the closure is to not compute a closure at all. We can just save a list of equivalences added to $\mathcal{P}$. The final result is still correct as the equivalences, that result from substituting new equivalences into others, would be added by the loop in Step 5 of Algorithm 14.12 later on either way. To check whether an equivalence is contained in $\mathcal{P}$, we check whether it was contained in the original $\mathcal{P}$ or whether it was added to M. Even though the set of equivalences contained in a commutative Thue system is infinite in general, the number of equivalences, that we check here, is finite due to the degree bound m. This makes the objects we save finite, but we have to split the check whether an equivalence is contained in the commutative Thue systems in two parts. First, checking whether the equivalence is contained in $\mathcal{P}$, which contains an infinite number of equivalences and is therefore given by a set of generators, and second, checking whether the equivalence is contained in our set of new equivalences.

The first part of this check is still an **EXPSPACE**-complete problem, but now we have to check every time on the same commutative Thue system, which can save some computational effort. We can now compute a Gröbner Basis of the system once and use it for all checks, that need to be done, without calculating new Gröbner Bases. The second part of the check works in time and space polynomial in the number of new equivalences added. The set of equivalences to be added can be bigger than exponential space, so this approach does not improve the asymptotic running time bounds. For instance consider a commutative Thue system that makes all words equivalent: In this case we add $O(m^2)$ elements to the generating set.

Algorithm 14.12 Compute the radical of a commutative Thue system using degree bounds, a common Gröbner Basis, and a list of added equivalences.

```
Input: an alphabet $\Sigma$ and $\alpha_1, \dots, \alpha_s, \beta_1, \dots, \beta_s \in \Sigma^*$ for some $s \in \mathbb{N}_{>0}$
Output: the radical of the commutative Thue system generated by the equivalences

        $\left\{\alpha_i \equiv \beta_i \,|\, i \in \{1, \dots, s\}\right\}$

 1: let $\mathcal{P}$ be the commutative Thue system generated by the equivalences

        $\left\{\alpha_i \equiv \beta_i \,|\, i \in \{1, \dots, s\}\right\}$

 2: choose a term ordering $\geq$ on $\mathbb{Q}[\Sigma]$, compute a Gröbner Basis $\mathcal{G}$ of $\mathcal{I}_Q(\mathcal{P})$ with respect to
    $\geq$ and use $\mathcal{G}$ for all further reductions modulo $\mathcal{P}$
 3: $M := \emptyset$
 4:
        $m := 2d^n\left(\log 2d^n + \log\log 2d^n\right)(sd)^{2^{cn^2+n-1}} + 1$

    where $d$ is the maximum length of the words $\alpha_1, \dots, \alpha_s, \beta_1, \dots, \beta_s \in \Sigma^*$, $n = |\Sigma|$, and
    $c \in \mathbb{N}_{>0}$ is some universal constant from Theorem 14.2
 5: for each $\alpha, \beta \in \Sigma^*$ with $|\alpha|, |\beta| \leq m$ in descending order of $|\alpha| + |\beta| \in \mathbb{N}_0$ do
 6:     if ($\alpha^2 \equiv_{\mathcal{P}} \alpha\beta$ or the equivalence is contained in $M$) and
        ($\beta^2 \equiv_{\mathcal{P}} \alpha\beta$ or the equivalence is contained in $M$) and
        not ($\alpha \equiv_{\mathcal{P}} \beta$ or the equivalence is contained in $M$)
         then
 7:         add $\alpha \equiv \beta$ to $M$
 8:     end if
 9: end for
10: return the commutative Thue system generated by $\{\alpha_i \equiv \beta_i \,|\, i \in \{1, \dots, s\}\} \cup M$
```

Note that this approach would not work for the non-constructive Algorithm 13.7. If we do not take a closure after each equivalence we add to $\mathcal{P}$ we could possibly add an infinite amount of equivalences to $\mathcal{P}$. The proof of Theorem 13.2 does not work anymore in this case, as there is no ascending chain condition of ideals without computing the closure as defined in Definition 3.18. The algorithm could still find a generating set of the radical and terminate, but this depends on the choices of words $\alpha, \beta \in \Sigma^*$. While "good" choices make the algorithm terminate after a finite amount of steps, "bad" choices make it run without terminating at all.

The huge list of new equivalences of the constructive Algorithm 14.12 can be compressed by making the relaxation of the closure of commutative Thue systems a little more restrictive. When we add an equivalence to $\mathcal{P}$ we will assume that just the equivalences with additional symbols on both sides should also be included in $\mathcal{P}$ automatically. Substituting other equivalences into one side will be ignored while adding equivalences to $\mathcal{P}$ during our algorithm. In this case, to check whether an equivalence $\alpha \equiv \beta$ is contained in $\mathcal{P}$, we just have to check whether it was contained in the original commutative Thue system or we are able remove some common symbols on both sides of the equivalence and obtain another equivalence that we added to $\mathcal{P}$ before.

Algorithm 14.13 Compute the radical of a commutative Thue system using degree bounds, a common Gröbner Basis, and a compressed list of added equivalences.

Input: an alphabet Σ and $\alpha_1, \dots, \alpha_s, \beta_1, \dots, \beta_s \in \Sigma^*$ for some $s \in \mathbb{N}_{>0}$
Output: the radical of the commutative Thue system generated by the equivalences

$$\left\{\alpha_i \equiv \beta_i \,|\, i \in \{1, \dots, s\}\right\}$$

```
 1: let $\mathcal{P}$ be the commutative Thue system generated by the equivalences
        $$\left\{\alpha_i \equiv \beta_i \,|\, i \in \{1, \dots, s\}\right\}$$
 2: choose a term ordering $\geq$ on $\mathbb{Q}[\Sigma]$, compute a Gröbner Basis $\mathcal{G}$ of $\mathcal{I}_{\mathbb{Q}}(\mathcal{P})$ with respect to
    $\geq$ and use $\mathcal{G}$ for all further reductions modulo $\mathcal{P}$
 3: $M := \emptyset$
 4:
        $$m := 2d^n\left(\log 2d^n + \log\log 2d^n\right)(sd)^{2^{cn^2+n-1}} + 1$$
    where $d$ is the maximum length of the words $\alpha_1, \dots, \alpha_s, \beta_1, \dots, \beta_s \in \Sigma^*$, $n = |\Sigma|$, and
    $c \in \mathbb{N}_{>0}$ is some universal constant from Theorem 14.2
 5: for each $\alpha, \beta \in \Sigma^*$ with $|\alpha|, |\beta| \leq m$ in descending order of $|\alpha| + |\beta| \in \mathbb{N}_0$ do
 6:     if ($\alpha^2 \equiv_{\mathcal{P}} \alpha\beta$ or the equivalence is a multiple of an equivalence in $M$) and
        ($\beta^2 \equiv_{\mathcal{P}} \alpha\beta$ or the equivalence is a multiple of an equivalence in $M$) and
        not ($\alpha \equiv_{\mathcal{P}} \beta$ or the equivalence is a multiple of an equivalence in $M$)
         then
 7:         remove all equivalences from $M$ that can be achieved by adding the same set of
            symbols to both sides of $\alpha \equiv \beta$
 8:         add $\alpha \equiv \beta$ to $M$
 9:     end if
10: end for
11: return the commutative Thue system generated by $\{\alpha_i \equiv \beta_i | i \in \{1, \dots, s\}\} \cup M$
```

Algorithm 14.13 still does not run in **EXPSPACE** as the compressed list M may still be larger in the worst case. Nevertheless, Algorithm 14.13 may improve the running time of actual implementations of Algorithm 13.24 for certain kinds of practical inputs, because it is only slow if the radical is much bigger than the original system.

Note that the relaxation of the closure of commutative Thue systems as presented above only works, because we know that there is only a finite set of equivalences that can possibly be needed as generators of the radical of the commutative Thue system. In general, the relaxation to only include equivalences that add the same symbols to both side of a generator of a commutative Thue system results in infinite generating sets. For instance, the commutative Thue system $\mathcal{P}$ over $\Sigma = \{0, 1\}$ generated in the common definition by $0 \equiv_{\mathcal{P}} 1$ requires the infinite set

$$\left\{0^i \equiv_{\mathcal{P}} 1^i \mid i \in \mathbb{N}_{>0}\right\}$$

as generators under this relaxed closure.

To summarize this section, Algorithm 13.24 is an algorithm for computing the radical of a pure binomial ideal. All steps run in **EXPSPACE** which is the same complexity as the best known algorithm for computing the radical of a polynomial ideal. As our algorithm involves pure binomials only, we can use it to compute radicals of pure binomial ideals and commutative Thue systems. There are various ways to improve the speed of this algorithm on many practical inputs, but these improvements have a worse space complexity in the worst case.

Part VI

Conclusion

15 Conclusion

In this thesis we discussed the complexity of several problems of polynomial ideals and subclasses of polynomial ideals. The word problem for general polynomial ideals and binomial ideals can only be solved using an exponential amount of space while radical ideals allow for algorithms using polynomial space. We found new bounds on the complexity of the word problem for radical binomial ideals whose complement can be solved in non-deterministic polynomial time.

We also introduced a new algorithm to compute radicals of pure binomial ideals. Our algorithm matches the running time of the best known algorithms to compute radicals of polynomial ideals, but uses only pure binomials as intermediate results. This allows for the usage of specialized data structures and enables us to transfer this algorithm to term replacement systems. We analyzed the structure of radical binomial ideals and used our results to define radicals of commutative Thue systems.

Part VII

Appendix

A Source Code

A.1 Cyclohexane

The following code is an implementation of the polynomial ideal modeling the conformations of cyclohexane as described in Section 1.1. We used Macaulay 2 [GS] for this implementation.

Listing A1.1: Implementation of a polynomial ideal describing the conformations of cyclohexane.

```
R = QQ[x_1,x_2,x_3]
f_1 = det matrix {
    {0, 3, 3, 3, 3, 3},
    {3, 0, 3, 8, 3*x_1, 8},
    {3, 3, 0, 3, 8, 3*x_2},
    {3, 8, 3, 0, 3, 8},
    {3, 3*x_1, 8, 3, 0, 3},
    {3, 8, 3*x_2, 8, 3, 0}
} / 3^6
f_2 = det matrix {
    {0, 3, 3, 3, 3, 3},
    {3, 0, 3, 8, 3*x_2, 8},
    {3, 3, 0, 3, 8, 3*x_3},
    {3, 8, 3, 0, 3, 8},
    {3, 3*x_2, 8, 3, 0, 3},
    {3, 8, 3*x_3, 8, 3, 0}
} / 3^6
f_3 = det matrix {
    {0, 3, 3, 3, 3, 3},
    {3, 0, 3, 8, 3*x_3, 8},
    {3, 3, 0, 3, 8, 3*x_1},
    {3, 8, 3, 0, 3, 8},
    {3, 3*x_3, 8, 3, 0, 3},
    {3, 8, 3*x_1, 8, 3, 0}
} / 3^6
f_4 = det matrix {
    {0, 3, 3, 3, 3, 3, 3},
    {3, 0, 3, 8, 3*x_1, 8, 3},
    {3, 3, 0, 3, 8, 3*x_2, 8},
    {3, 8, 3, 0, 3, 8, 3*x_3},
    {3, 3*x_1, 8, 3, 0, 3, 8},
    {3, 8, 3*x_2, 8, 3, 0, 3},
    {3, 3, 8, 3*x_3, 8, 3, 0}
} / 3^7
I = ideal(f_1, f_2, f_3, f_4)
```

```
tex gens gb I
--I == radical I
(3*x_1/11)*(3*x_3/11)*(3*x_3/11) % I

--points in the variety
for i from 1 to 4 list sub(f_i, {x_1 => 11/3, x_2 => 11/3, x_3 =>
    11/3})
for i from 1 to 4 list sub(f_i, {x_1 => 25/9, x_2 => 11/3, x_3 =>
    11/3})
for i from 1 to 4 list sub(f_i, {x_1 => 11/3, x_2 => 25/9, x_3 =>
    11/3})
for i from 1 to 4 list sub(f_i, {x_1 => 11/3, x_2 => 11/3, x_3 =>
    25/9})

--not contained in the variety
for i from 1 to 4 list sub(f_i, {x_1 => 11/3, x_2 => 11/3, x_3 =>
    25/10})
```

A.2 Experimental Degree Bound

This code finds small examples for Theorem 13.8. It is written in C++.

Listing A1.2: Find small examples for powers of pure binomials in Theorem 13.8.

```cpp
#include <cmath>
#include <iostream>
#include <sstream>
#include <vector>

using namespace std;

//precompute binomial coefficients
vector<vector<long long> > bc;
void init(int max_size) {
  bc.resize(max_size + 1);
  for(int i = 0; i <= max_size; i++) {
    bc[i].resize(i + 1);
    bc[i][0] = 1;
    for(int j = 1; j < i; j++) bc[i][j] = bc[i-1][j-1] + bc[i-1][j];
    bc[i][i] = 1;
  }
}

//check a given partition
```

```
//r is the level of the monomials, b is the bitmask of the partition
void check(int r, long long b) {
  //sum everything up
  vector<long long> sum ((r+1)/2, 0);
  vector<int> equivalence_class (r+1, 0);
  for(int k = 0; k <= r; k++) {
    equivalence_class[k] = b % ((r+1)/2);
    sum[equivalence_class[k]] += bc[r][k] * (k % 2 ? -1 : 1);
    b /= (r+1)/2;
  }

  //only take lexicographically smallest versions of each partition
     to make the
  //output easier to read
  int max_used = -1;
  for(int k = r; k >= 0; k--) {
    if(equivalence_class[k] > max_used) {
      if(equivalence_class[k] == max_used + 1)
        max_used = equivalence_class[k];
      else
        return;
    }
  }

  //check sums
  bool ok = true, symmetric = true;
  for(int i = 0; i < (r+1)/2; i++) {
    ok &= sum[i] == 0;
    symmetric &= equivalence_class[i] == equivalence_class[r - i];
  }
  if(ok) {
    for(int k = 0; k <= r; k++) cout << equivalence_class[k] << "␣";
    cout << (symmetric ? "(symmetric)" : "") << endl;
  }
}

//check all partitions for a given level r
void check(int r) {
  //find end value
  long long end = 1, pow = 1;
  for(int i = r; i >= 0; i--) {
    end += pow * min((r+1)/2 - 1, i);
    pow *= (r+1)/2;
  }
  if(r <= 2) end = 1;

  //check all possible partitions
  for(long long b = 0; b < end; b++) check(r, b);
}

//main function: check all r up to a given value
int main(int argc, char *argv[]) {
  //check arguments
  if(argc < 2) {
    cout << "Usage:␣" << argv[0] << "␣max_r" << endl;
    return 1;
  }
  stringstream convert(argv[1]);
  int max_r;
  if(!(convert >> max_r) || max_r <= 0) {
    cout << "Error:␣argument␣max_r␣(value␣'" << argv[1] << "')␣
       needs␣to␣be␣a␣positive␣integer" << endl;
    return 1;
  }

  //initialize binomial coefficients
  init(max_r);

  //check all r up to max_r
  for(int r = 1; r <= max_r; r++) {
    cout << "checking␣level␣" << r << endl;
    check(r);
  }
  return 0;
}
```

Bibliography

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[Abe26] Niels Henrik Abel. Démonstration de l'impossibilité de la résolution algébrique des équations générales qui passent le quatrieme degré. *Journal für die reine und angewandte Mathematik*, 1:65–96, 1826.

[AKG$^+$] Scott Aaronson, Greg Kuperberg, Christopher Granade, Vincent Russo, and contributors. Complexity Zoo. `https://complexityzoo.uwaterloo.ca/`. Accessed: 2016-11-30.

[BGN97] Eberhard Becker, Rudolf Grobe, and Michael Niermann. Radicals of binomial ideals. *Journal of Pure and Applied Algebra*, 117-118(0):41–79, 1997.

[Bos09] Siegfried Bosch. *Algebra*. Springer, Berlin, 7th edition, 2009.

[Bre11] Murray R. Bremner. *Lattice basis reduction: an introduction to the LLL algorithm and its applications*. CRC Press, 2011.

[Bro87] W. Dale Brownawell. Bounds for the Degrees in the Nullstellensatz. *The Annals of Mathematics*, 126(3):577, 1987.

[BS96] E. Bach and J.O. Shallit. *Algorithmic Number Theory: Efficient algorithms*. Number v. 1 in Algorithmic Number Theory. MIT Press, 1996.

[Buc65] Bruno Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal*. PhD thesis, Mathematical Institute, University of Innsbruck, Austria, 1965.

[BWK93] Thomas Becker, Volker Weispfenning, and Heinz Kredel. *Gröbner bases: A computational approach to commutative algebra*, volume 141 of *Graduate texts in mathematics*. Springer-Verlag, New York, 1993.

[Cay54] Arthur Cayley. On the theory of groups as depending on the symbolic equation $\theta^n = 1$. *Philosophical Magazine*, 7(42):40–47, 1854.

[Cho56] Noah Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113–124, September 1956.

[Chu36] Alonzo Church. An Unsolvable Problem of Elementary Number Theory. *American Journal of Mathematics*, 58(2):345–363, 1936.

[CLO07] David A. Cox, John B. Little, and Donal O'Shea. *Ideals, varieties, and algorithms: An introduction to computational algebraic geometry and commutative algebra*. Undergraduate texts in mathematics. Springer, New York, 3rd edition, 2007.

[Dav10] Mike Davey. A Turing Machine - In the Classic Style. `http://www.aturingmachine.com`, 2010. Accessed: 2015-09-10.

[DFGS91] Alicia Dickenstein, Noa Fitchas, Marc Giusti, and Carmen Sessa. The membership problem for unmixed polynomial ideals is solvable in single exponential time. *Discrete Applied Mathematics*, 33(1):73 – 94, 1991.

[Dic13] Leonard Eugene Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *American Journal of Mathematics*, 35(4):413–422, 1913.

[dLVP96] Charles Jean de La Vallée Poussin. Recherches analytiques sur la théorie des nombres premiers. *Annales de la Sociéteé scientifique de Bruxelles*, 20:183–256, 1896.

[DSW94] Martin Davis, Ron Sigal, and Elaine J. Weyuker. *Computability, Complexity, and Languages: Fundamentals of Theoretical Computer Science*. Computer science and applied mathematics. Academic Press, 1994.

[Dub90] Thomas W. Dubé. The structure of polynomial ideals and Gröbner bases. *SIAM Journal on Computing*, 19(4):750–773, 1990.

[Dus98] Pierre Dusart. *Autour de la fonction qui compte le nombre de nombres premiers*. PhD thesis, Laboratoire d'Arithmétique, de Calcul Formel et d'Optimisation, Université de Limoges, 1998.

[EGG$^+$06] Wayne Eberly, Mark Giesbrecht, Pascal Giorgi, Arne Storjohann, and Gilles Villard. Solving sparse rational linear systems. In *Proceedings of the 2006 International Symposium on Symbolic and Algebraic Computation*, ISSAC '06, pages 63–70, New York, NY, USA, 2006. ACM.

[ES96] David Eisenbud and Bernd Sturmfels. Binomial ideals. *Duke Mathematical Journal*, 84(1):1–45, 07 1996.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing grbner bases (f4). *Journal of Pure and Applied Algebra*, 139(13):61 – 88, 1999.

[Fau02] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In *International Symposium on Symbolic and Algebraic Computation Symposium - ISSAC 2002*, Villeneuve d'Ascq, France,

2002. Colloque avec actes et comité de lecture. internationale.

[FGLM93] Jean-Charles Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient computation of zero-dimensional gröbner bases by change of ordering. *Journal of Symbolic Computation*, 16(4):329–344, 1993.

[Fis13] Gerd Fischer. *Lineare Algebra: Eine Einführung für Studienanfänger*. Grundkurs Mathematik. Springer Fachmedien Wiesbaden, 2013.

[Fis14] Gerd Fischer. *Lineare Algebra*. Grundkurs Mathematik. Springer Spektrum, 18th edition, 2014.

[Gau31] Carl Friedrich Gauß. Untersuchungen über die Eigenschaften der positiven ternären quadratischen Formen von Ludwig August Seber. *Göttingische gelehrte Anzeigen*, 1831.

[Göd31] Kurt Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für Mathematik und Physik*, 38(1):173–198, 1931.

[GS] Daniel R. Grayson and Michael E. Stillman. Macaulay 2, a software system for research in algebraic geometry. Available at http://www.math.uiuc.edu/Macaulay2/.

[Gun41] Nikolai M. Gunther. Sur les modules des formes algébriques. *Trav. Inst. Math. Tbilissi [Trudy Tbiliss. Mat. Inst.]*, 9:97–206, 1941.

[Had96] Jacques Hadamard. Sur la distribution des zéros de la fonction $\zeta(s)$ et ses conséquences arithmétiques. *Bulletin de la Société Mathématique de France*, 24:199–220, 1896.

[Hal66] James D. Halpern. Bases in vector spaces and the axiom of choice. *Proceedings of the American Mathematical Society*, 17:670–673, 1966.

[Hal05] Thomas C. Hales. A proof of the Kepler conjecture. *Ann. Math. (2)*, 162(3):1065–1185, 2005.

[Her26] Grete Hermann. Die Frage der endlich vielen Schritte in der Theorie der Polynomideale. *Mathematische Annalen*, 95(1):736–788, 1926.

[Hil90] David Hilbert. Über die Theorie der algebraischen Formen. *Mathematische Annalen*, 36(4):473–534, 1890.

[Hil93] David Hilbert. Über die vollen Invariantensysteme. *Mathematische Annalen*, 42(3):313–373, 1893.

[Hir64] Heisuke Hironaka. Resolution of singularities of an algebraic variety over a field of characteristic zero: II. *Annals of Mathematics*, pages 205–326, 1964.

[HS65] Juris Hartmanis and Richard E. Stearns. On the computational complexity of algorithms. *Trans. Amer. Math. Soc.*, 117:285–306, 1965.

[HS66] Frederick C. Hennie and Richard E. Stearns. Two-Tape Simulation of Multitape Turing Machines. *J. ACM*, 13(4):533–546, October 1966.

[Kem02] Gregor Kemper. The Calculation of Radical Ideals in Positive Characteristic. *Journal of Symbolic Computation*, 34(3):229–238, 2002.

[Kle43] S. C. Kleene. Recursive predicates and quantifiers. *Transactions of the American Mathematical Society*, 53(1):41–73, 1943.

[KM96] Klaus Kühnle and Ernst W. Mayr. Exponential space computation of Gröbner bases. In *Proceedings of the 1996 international symposium on Symbolic and algebraic computation - ISSAC '96*, pages 63–71. ACM Press, 1996.

[KM10] Christian Karpfinger and Kurt Meyberg. *Algebra: Gruppen - Ringe - Körper*. Spektrum Akademischer Verlag, 2nd edition, 2010.

[Knu68] Donald E. Knuth. *The Art of Computer Programming. Vol. 1: Fundamental Algorithms*. Addison-Wesley Series in Computer Science and Information Processing. Addison-Wesley, 1968.

[Kol88] János Kollár. Sharp effective Nullstellensatz. *Journal of the American Mathematical Society*, 1(4):963, 1988.

[Kra08] Matthias Kratzer. Computing the dimension of a polynomial ideal and membership in low-dimensional ideals. Master's thesis, TU München, Ocotober 2008.

[Lan09] Edmund Landau. *Handbuch Der Lehre Von Der Verteilung Der Primzahlen*. Bd. 1. Scholarly Publishing Office, University of Michigan Library, 1909.

[Lan87] Serge Lang. *Linear Algebra*. Undergraduate texts in mathematics. Springer, New York, 1987.

[Lap06] Santiago Laplagne. An algorithm for the computation of the radical of an ideal. In *Proceedings of the 2006 international symposium on Symbolic and algebraic computation - ISSAC '06*, page 191. ACM Press, 2006.

[LLL82] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[Mat01] Ryutaroh Matsumoto. Computing the Radical of an Ideal in Positive Characteristic. *Journal of Symbolic Computation*, 32(3):263–271, 2001.

[May97] Ernst W. Mayr. Some complexity results for polynomial ideals. *Journal of Complexity*, 13(3):303–325, 1997.

[Mit08] Johannes Mittmann. Computing the Radical of Binomial Ideals. Diploma thesis, TU München, München, December 2008.

[MM82] Ernst W. Mayr and Albert R. Meyer. The complexity of the word problems for commutative semigroups and polynomial ideals. *Advances in Mathematics*, 46(3):305–329, 1982.

[MMN89] Herbert Melenk, Hans-Michael Möller, and Winfried Neun. Symbolic solution of large stationary chemical kinetics problems. *IMPACT of Computing in Science and Engineering*, 1(2):138–167, 1989.

[MR11] Ernst W. Mayr and Stephan Ritscher. Space-efficient Gröbner basis computation without degree bounds. In *Proceedings of the 36th International Symposium on Symbolic and Algebraic Computation*, ISSAC '11, pages 257–264, New York, NY, USA, 2011. ACM.

[MR13] Ernst W. Mayr and Stephan Ritscher. Dimension-dependent bounds for Gröbner bases of polynomial ideals. *Journal of Symbolic Computation*, 49:78–94, 2013. The International Symposium on Symbolic and Algebraic Computation.

[MT15] Ernst W. Mayr and Stefan Toman. The complexity of the membership problem for radical binomial ideals. In N. N. Vassiliev, editor, *International Conference Polynomial Computer Algebra '2015*, pages 61–64. Euler International Mathematical Institute, VVM Publishing, 2015.

[MT16a] Ernst W. Mayr and Stefan Toman. Complexity of membership problems of different types of polynomial ideals. *Edited volume of the DFG priority project 1489 "Algorithmic and Experimental Methods in Algebra, Geometry, and Number Theory"*, 2016. Submitted.

[MT16b] Ernst W. Mayr and Stefan Toman. Radicals of term replacement systems. Annual Conference of the DFG Priority Project SPP 1489, Kaiserslautern, October 2016.

[Pal04a] Bruce P. Palka. Editor's Endnotes. *The American Mathematical Monthly*, 111(5):456–460, 2004.

[Pal04b] Bruce P. Palka. Editor's Endnotes. *The American Mathematical Monthly*, 111(10):927–929, 2004.

[Rab30] Yuri Rabinowitsch. Zum Hilbertschen Nullstellensatz. *Mathematische Annalen*, 102(1):520, 1930.

[Ren80] Bodo Renschuch. Beiträge zur konstruktiven Theorie der Polynomideale. XVII/1. Zur Hentzelt/Noether/Hermannschen Theorie der endlich vielen Schritte. *Wiss. Z. Pädagog. Hochsch. Karl Liebknecht Potsdam*, 24(1):87–99, 1980.

[Rob85] Lorenzo Robbiano. Term orderings on the polynomial ring. In Bob F. Caviness, editor, *EUROCAL '85*, volume 204, pages 513–517. Springer Berlin / Heidelberg, 1985.

[RRRA03] Bodo Renschuch, Hartmut Roloff, Georgij G. Rasputin, and Michael Abramson. Contributions to constructive polynomial ideal theory XXIII: forgotten works of Leningrad mathematician NM Gjunter on polynomial ideal theory. *ACM SIGSAM Bulletin*, 37(2):35–48, 2003.

[Sei74] Abraham Seidenberg. Constructions in algebra. *Trans. Amer. Math. Soc.*, 197:273–313, 1974.

[SHL65] Richard E. Stearns, Juris Hartmanis, and Phil M. Lewis. Hierarchies of memory limited computations. In *Proceedings of the 6th Annual Symposium on Switching Circuit Theory and Logical Design (SWCT 1965)*, FOCS '65, pages 179–190, Washington, DC, USA, 1965. IEEE Computer Society.

[SL96] Arne Storjohann and George Labahn. Asymptotically fast computation of hermite normal forms of integer matrices. In *Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation*, ISSAC '96, pages 259–266, New York, NY, USA, 1996. ACM.

[Som99] Martín Sombra. A Sparse Effective Nullstellensatz. *Advances in Applied Mathematics*, 22(2):271 – 295, 1999.

[Tho10] Wolfgang Thomas. "when nobody else dreamed of these things“ – Axel Thue und die Termersetzung. *Informatik-Spektrum*, 33(5):504–508, 2010.

[Thu10] Axel Thue. Die Lösung eines Spezialfalles eines generellen logischen Problems. *Skrifter udg. af Videnskabs-selskabet i Christiania. I, Math.-naturv. klasse*, 1910.

[Thu14] Axel Thue. Problem über Veränderungen von Zeichenreihen nach gegebenen Regeln. *Skrifter udg. af Videnskabs-selskabet i Christiania. I, Math.-naturv. klasse*, 1914.

[Tom12] Stefan Toman. Investigation of operations on binomial ideals. Bachelor's thesis, TU München, September 2012.

[Tom15a] Stefan Toman. The complexity of the radical word problem for binomial ideals. Annual Conference of the DFG Priority Project SPP 1489, Osnabrück, September 2015.

[Tom15b] Stefan Toman. The radical word problem for binomial ideals. Master's thesis, TU München, September 2015.

[Tur37a] Alan Mathison Turing. Computability and λ-definability. *Journal of Symbolic Logic*, 2(4):153–163, 12 1937.

[Tur37b] Alan Mathison Turing. On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42(1):230–265, 1937.

[vdWANB43] Bartel L. van der Waerden, Emil Artin, Emmy Noether, and Theodore J. Benac. *Modern algebra*. 1943.

[vN93] John von Neumann. First Draft of a Report on the EDVAC. *IEEE Ann. Hist. Comput.*, 15(4):27–75, October 1993.

[Yap91] Chee K. Yap. A new lower bound construction for commutative thue systems with applications. *Journal of Symbolic Computation*, 12(1):1–27, 1991.

List of Algorithms

List of Figures

List of Tables

List of Listings

Index