JOHANNES KEPLER
UNIVERSITY LINZ

Submitted by
Dip.-Ing. Clemens Hofstadler, MSc

Submitted at
Institute for Algebra

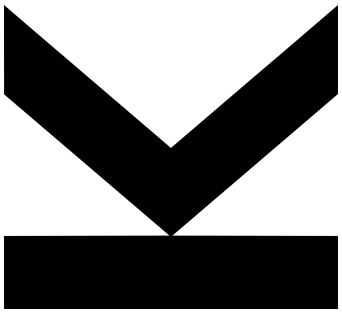
Supervisor and First Evaluator
Prof. Dr. Georg Regensburger

Second Evaluator
Cyrille Chenavier, PhD

Co-Supervisor
Dr. Clemens G. Raab

September 2023

Noncommutative Gröbner bases and automated proofs of operator statements



Doctoral Thesis

to obtain the academic degree of

Doktor der Technischen Wissenschaften

in the Doctoral Program

Technische Wissenschaften

JOHANNES KEPLER UNIVERSITY LINZ
Altenbergerstraße 69
4040 Linz, Austria
www.jku.at
DVR 0093696

Abstract

Linear operators appear in several forms in various settings all across mathematics. They can be ring elements (as in C^*-algebras), matrices, but also vector space and module homomorphisms, or more generally, morphisms in abelian categories.

In this thesis, we develop an algebraic framework for automatically proving statements about linear operators by computations with noncommutative polynomials. More specifically, arbitrary first-order statements about identities of linear operators can be treated. We present a practical semi-decision procedure for validity of such formulas based on the verification of ideal membership in a free algebra. In contrast to classical approaches for automated theorem proving, these algebraic computations automatically incorporate linearity and they benefit from efficient ideal membership procedures.

In particular, we exploit the theory of noncommutative Gröbner bases to verify ideal membership in the free algebra. In order to enhance these computations, we generalise the concept of signature Gröbner bases, originally developed for commutative polynomials, to the free setting, and more generally, to mixed algebras, allowing a mixture of commutative and noncommutative variables.

Based on Gröbner basis techniques, we also generalise existing and develop new algorithms for computing elements of specific forms in noncommutative polynomial ideals. These methods serve as one of the key steps in the aforementioned semi-decision procedure. Furthermore, we present novel methods for finding short proofs of operator statements, based on the ability to compute short certificates of ideal membership. All algorithms are implemented in various software packages for SageMath and Mathematica.

We illustrate the capabilities of our framework and software through a case study on statements about the Moore-Penrose inverse, including classical facts and recent results. Furthermore, we showcase that our approach allows to discover new theorems, and we discuss how diagram chases in abelian categories can be automated using our framework.

Kurzfassung

Lineare Operatoren treten in verschiedenen Formen in diversen mathematischen Kontexten auf. Sie können Ringelemente sein (wie in C^*-Algebren), Matrizen, aber auch Vektorraum- und Modulhomomorphismen, oder allgemeiner, Morphismen in abelschen Kategorien.

Wir entwickeln eine algebraische Theorie, um Aussagen über lineare Operatoren automatisiert durch Berechnungen mit nichtkommutativen Polynomen zu beweisen. Es können beliebige Aussagen erster Ordnung über Operatoridentitäten behandelt werden. Wir präsentieren ein Semientscheidungsverfahren zur Überprüfung der Gültigkeit solcher Formeln, basierend auf der Verifikation von Idealzugehörigkeit in freien Algebren. Im Gegensatz zu klassischen Ansätzen für automatisiertes Beweisen berücksichtigen diese algebraischen Berechnungen automatisch Linearität und profitieren von effizienten Idealzugehörigkeitsverfahren.

Insbesondere nutzen wir die Theorie der nichtkommutativen Gröbnerbasen, um Idealzugehörigkeit in der freien Algebra nachzuweisen. Um diese Berechnungen zu verbessern, verallgemeinern wir das Konzept der Signatur-Gröbnerbasen, ursprünglich für kommutative Polynome entwickelt, auf die freie Algebra und noch allgemeiner auf gemischte Algebren, welche eine Mischung aus kommutativen und nichtkommutativen Variablen erlauben.

Wir verallgemeinern auch bestehende und entwickeln neue Algorithmen zur Berechnung von Elementen bestimmter Form in nichtkommutativen Polynomidealen. Diese Methoden sind einer der Schlüsselschritte im oben genannten Semientscheidungsverfahren. Darüber hinaus präsentieren wir neue Methoden um kurze Beweise von Operatoraussagen zu finden, beruhend auf der Fähigkeit kurze Zertifikate für Idealzugehörigkeit zu berechnen. Alle Algorithmen sind in Softwarepaketen für SageMath und Mathematica implementiert.

Wir veranschaulichen das Potenzial unserer Theorie und der Software anhand einer Fallstudie zu Aussagen über die Moore-Penrose-Inverse, einschließlich klassischer Fakten und aktueller Resultate. Darüber hinaus zeigen wir, dass es unser Ansatz ermöglicht, neue Theoreme zu entdecken und Diagrammjagden in abelschen Kategorien zu automatisieren.

Acknowledgements

First and foremost, I have to express my deepest gratitude to my advisors Georg Regensburger and Clemens G. Raab for their continuous support and encouragement during my studies. Despite the challenges posed by a pandemic and several lockdowns, they always kept an open (virtual) door for insightful discussions and assistance, both in scientific and personal matters. Their guidance and expertise have been crucial in shaping my research and academic growth. At the same time, they always granted me full independence and freedom to develop and pursue my own ideas, for which I am very thankful.

Furthermore, I sincerely thank Cyrille Chenavier for agreeing to serve as the second evaluator of this thesis. I am also grateful to him and Thomas Cluzeau for their hospitality during my one-week stay at the University of Limoges. I really enjoyed my time there and our interesting discussions were a great source of motivation for me.

Many thanks also go to Thibaut Verron, who not only collaborated as an outstanding co-author on several publications but who also became a dear friend. He has taught me so many things: from mathematical concepts and the art of writing good papers and giving good talks, to even some of his LaTeX magic. I also have to thank him for showing me the ins and outs of life at conferences.

Over the past years, I have had the privilege of being a part of two different institutes: the Institute for Algebra at JKU and the Institute of Mathematics at the University of Kassel. In both places, I had the pleasure of working alongside great colleagues, many of whom have since become good friends. I have to thank all of them for making the past years such a wonderful and memorable time.

Finally, I am very grateful to my family and friends for their understanding and support. Without all of you, this endeavour would have not been possible. Thank you!

This work was supported by the Austrian Science Fund (FWF): P32301.

Contents

1 Introduction

In its section on the Moore-Penrose inverse, the Handbook of Linear Algebra [Hog13, Sec. I.5.7] lists, besides the defining identities of the Moore-Penrose inverse (see (1.1) later on page 6), a number of classical facts:

1. Every $A \in \mathbb{C}^{m\times n}$ has a unique Moore-Penrose inverse $A^\dagger$.
2. If $A \in \mathbb{R}^{m\times n}$, then $A^\dagger$ is real.
3. If $A \in \mathbb{C}^{m\times n}$ $[\dots]$ has a full rank decomposition $A = BC$ $[\dots]$, then $A^\dagger$ can be evaluated using $A^\dagger = C^*(B^*AC^*)^{-1}B^*$.
4. If $A \in \mathbb{C}^{m\times n}$ $[\dots]$ has an SVD $A = U\Sigma V^*$, then its Moore-Penrose inverse is $A^\dagger = V\Sigma^\dagger U^*$ $[\dots]$.

$\vdots$

In this context, consider the task of proving as many of these facts as possible, using only the defining identities and no additional references. This is clearly a nontrivial task for any non-expert.

In this thesis, we develop an algebraic framework for proving statements like the ones listed above based on computations with noncommutative polynomials. Our main result is a practical semi-decision procedure that allows to automatically prove any true first-order statement about identities of linear operators by verification of ideal membership in a free algebra. To enhance the polynomial computations underlying this semi-decision procedure, we also generalise existing and develop new algorithms in the realm of noncommutative Gröbner bases. Using these algorithms, which are implemented in several software packages, in combination with our framework, we are able to automatically prove the majority of the facts on the Moore-Penrose inverse in the Handbook of Linear Algebra. Moreover, we generalise recent results in operator theory and discover new theorems.

1.1 Outline and contributions

In the following, we provide an overview of this thesis and outline the main contributions of this work and the related publications. Parts of this thesis have already appeared in our published papers [Cve+21; HRR22a; HV22; BHR23; HV23b] and in our preprints [HRR22b; HV23a]. A more detailed description of our contributions as well as references to related work are given at the beginning of each chapter.

In the following Sections 1.2 and 1.3, the reader will be able to gain a practical understanding of the algebraic framework developed in this work, by learning how to translate operator statements into polynomial computations and how to use our SageMath package `operator_gb` to compute algebraic proofs. These sections also appear in our joint work [BHR23]. In Section 1.2, we also discuss previous work on using noncommutative polynomials for proving operator identities.

Chapter 2 serves as a preparation for the subsequent parts, covering basic concepts needed for our work. We start by recalling basic concepts from category theory in Section 2.1, including preadditive semicategories, which provide a very general setting for studying linear operators, and abelian categories. We also give several examples of these structures. It is noteworthy that Section 2.1.1, which is devoted to abelian categories, is only relevant for the results in Section 7.3 on automated diagram chases. Then, in Section 2.2, we discuss the construction and fundamental properties of key algebraic structures relevant for the theory and algorithms developed in this work, including free monoids, free algebras, and free (bi)modules. In Section 2.3, we review basics of term rewriting, before presenting the theory of Gröbner bases in the free algebra from the perspective of polynomial rewriting in Section 2.4. This section notably contains a novel characterisation of a large family of noncommutative monomial orders by combining totally ordered semigroups with the lexicographic order. Finally, in Section 2.5, we give an overview on many-sorted logic, which provides us with a formal language for encoding and studying statements about linear operators.

In Chapter 3, we generalise the concept of signature Gröbner bases, originally developed for commutative polynomials, to the noncommutative setting. This special kind of Gröbner bases are obtained through so-called signature-based algorithms, which compute, next to a Gröbner basis, also additional information on how the polynomials in that basis were computed. Using this information, these algorithms are not only able to predict and avoid

redundant computations, but they can also efficiently perform a number of operations on the syzygy module of a family of polynomials. The results presented in this chapter also appear in our joint work [HV22], where we develop the theory of signature Gröbner bases in the free algebra over a coefficient field, and in the joint paper [HV23b], where we extend the theory to the *mixed algebra* over commutative coefficient rings. The latter setting allows to have a mixture of commutative and noncommutative variables and arises naturally when performing different ideal theoretic operations in the free algebra. In Section 3.1, we formally introduce the mixed algebra and the central concept of signatures, based upon which we then define signature Gröbner bases in this context in Section 3.2. The following Sections 3.3 and 3.4 are devoted to the computation of signature Gröbner bases, with the former focusing on the simpler case of the free algebra over a coefficient field, and the latter presenting the theory in the mixed algebra in full generality as done in [HV23b]. Notably, Section 3.3 summarises the results from [HV22] but uses the language and techniques from [HV23b], leading not only to a simpler presentation but also to a generalisation of the results compared to [HV22]. To end this chapter, we present experimental data showing the efficiency of the developed signature-based algorithms compared to classical approaches in Section 3.5.

Chapter 4 is devoted to the development of our algebraic framework for proving statements about linear operators by polynomial computations. The results of this part also appear in our preprint [HRR22b]. In Section 4.1, we first describe how many-sorted logic allows to encode statements about identities of linear operators. Then, we recall in Section 4.2 two classical concepts from first-order logic and automated theorem proving, namely Herbrand's theorem and Ackermann's reduction, which are essential tools for translating arbitrary first-order formulas into noncommutative polynomials and their membership in appropriate ideals. Following upon that, in Section 4.3, we define one of the central new notions introduced in this work, namely the concept of *idealisation*. Idealisation is a process that allows to characterise validity of certain first-order formulas via ideal membership in the free algebra, and thus, reduces proving operator statements to computations with noncommutative polynomials. We formally prove this important connection between semantics of formulas and polynomial computations in Section 4.4. In Section 4.5, we then explain how idealisation, in combination with Herbrand's theorem and Ackermann's reduction, can be used to obtain a semi-decision procedure (Procedure 8) that allows to prove every true first-order operator statement, showing that our approach is complete in this sense. To end this chapter, in Section 4.6, we illustrate how to apply the framework to

prove a classical result about the existence of Moore-Penrose inverses in categories with involution.

While Chapter 4 lays the theoretical foundation of our framework, we take a more application-oriented point of view in Chapter 5 and present relevant aspects for effectively applying the theory in practice. We begin by giving a practical summary of the framework in Section 5.1, where we focus on the simple, yet in practice most common, case of so-called existential quasi-identities. This drastically reduces the complexity of the presentation compared to the general case of arbitrary first-order formulas handled in Chapter 4. We note that this section is self-contained and can be read independently of Chapter 4. In fact, it can even be used as an informal introduction for that chapter. Then, in Section 5.2, we discuss how frequently arising properties of matrices and linear operators can be expressed in terms of identities, and thus, treated within the framework. This section also appears in our joint work [BHR23]. Following upon that, in Section 5.3, we present several methods that allow to search for polynomials of a particular form in a given noncommutative ideal. The ability to do this is one of the key steps required in the semi-decision procedure Procedure 8. We discuss well-known and present novel algorithmic techniques based on noncommutative Gröbner basis computations to do this and illustrate them by examples. This part is mostly summarised in our paper [HRR22a], with some minor aspects like Section 5.3.2 being presented here for the first time. To end this chapter, we discuss how our algebraic perspective to proving operator statements can be leveraged to obtain short proofs. As our framework reduces proving operator statements to the verification of ideal membership in the free algebra, certificates for the latter can be considered as proofs of the former. In Section 5.4, we discuss the problem of finding minimal certificates for ideal membership by exploiting properties of signature-based Gröbner basis algorithms. This part is also presented in our joint preprint [HV23a].

Chapter 6 is dedicated to the software packages we have developed. In Section 6.1, we present our SageMath package `operator_gb`, which provides functionality for verifying ideal membership in the free algebra based on noncommutative Gröbner basis computations, with a particular focus on methods that facilitate proving statements about linear operators. While the description of the functionality of the package has also appeared as the appendix of our joint work [BHR23], the discussion of implementation details is presented here for the first time. In Section 6.2, we present the latest version of our Mathematica package `OperatorGB`, initiated in [Hof20], providing similar functionality as the SageMath package.

Finally, in Section 6.3, we give an overview on our SageMath package `signature_gb` for signature Gröbner basis computations in the free algebra. Notably, in this section, we present for the first time how to combine signature-based techniques with linear algebra style reductions in the noncommutative setting, following ideas from the commutative case and leading to a signature-based F4 algorithm for computing signature Gröbner bases in the free algebra.

Finally, in Chapter 7, we illustrate the capabilities of the framework developed in this work in combination with our software on different applications. Section 7.1 presents a case study on statements regarding the Moore-Penrose inverse published as part of our joint work [BHR23], including classical facts from the Handbook of Linear Algebra and recent results in operator theory. Moreover, in Section 7.2, we describe how our framework and software have helped to discover several improvements of Hartwig's triple reverse order law [Har86], an important result on the Moore-Penrose inverse of a product of three matrices. This part is based on the results in our joint work [Cve+21]. To end this chapter, we discuss how our framework allows to automate the process of diagram chasing in abelian categories in Section 7.3.

Notation & Conventions

Throughout this work, we follow the following conventions:

- $\mathbb{N} = \{0, 1, 2, \dots\}$ denotes the set of nonnegative integers and $\mathbb{N}_{>0} = \mathbb{N} \setminus \{0\}$ the set of positive integers.
- By a ring, we always mean a (not necessarily commutative) ring with unit element 1, except if explicitly stated otherwise.
- By an R-module, we always mean a left R-module.
- By an algebra, we always mean an associative (but not necessarily commutative) algebra with unit element 1 over a commutative ring.

1.2 From operator identities to noncommutative polynomials

In this section, we give an informal and accessible introduction to our framework, describing how to translate identities of operators into noncommutative polynomials and how computations with these polynomials allow to derive new operator identities. We also showcase how our SageMath package `operator_gb` can be used to automate these polynomial computations. We focus on basic statements about the Moore-Penrose inverse and note that various examples of automated proofs of matrix and operator identities based on computations with noncommutative polynomials are also given in [SL20; Sch21].

In 1920, E. H. Moore [Moo20] generalised the notion of the inverse of a matrix from nonsingular square matrices to all, including rectangular, matrices. This generalised inverse, by Moore also called "general reciprocal", was later rediscovered by Roger Penrose [Pen55], leading to the now commonly used name *Moore-Penrose inverse*.

Moore established, among other main properties, existence and uniqueness of his generalised inverse and justified its application to linear equations. However, Moore's work was mostly overlooked during his lifetime due to his peculiar and complicated notations, which made his results inaccessible for all but very dedicated readers. In contrast, Penrose characterised this generalised inverse by four simple identities, yielding the following definition: The *Moore-Penrose inverse* of a complex matrix A is the unique matrix B satisfying the four *Penrose identities*

$$ABA = A, \qquad BAB = B, \qquad B^*A^* = AB, \qquad A^*B^* = BA. \tag{1.1}$$

Here, T^* denotes the Hermitian adjoint of a complex matrix T, satisfying

$$(S+T)^* = S^* + T^*, \qquad (ST)^* = T^*S^*, \qquad (T^*)^* = T. \tag{1.2}$$

Typically, the Moore-Penrose inverse of A is denoted by $A^\dagger$.

Using the Penrose identities, and their adjoint versions that follow, makes basic computations involving the Moore-Penrose inverse very simple. For example, uniqueness can be

shown as follows. If B and C both satisfy (1.1), then

$$\begin{aligned} B &= BAB = BACAB = BACB^*A^* = BC^*A^*B^*A^* \\ &= BC^*A^* = BAC = A^*B^*C = A^*C^*A^*B^*C \\ &= A^*C^*BAC = CABAC = CAC = C. \end{aligned} \tag{1.3}$$

At the end of the last century, people realised that matrix identities, or more generally, identities of linear operators, can be modelled by noncommutative polynomials, and that computations like (1.3) can be automated using algebraic computations involving such polynomials [HW94; HSW98; HS99].

Noncommutative polynomials are elements in a free (associative) algebra $R\langle X\rangle$ with coefficients in a commutative ring R with unity and noncommutative indeterminates in a (typically finite) set X. Monomials are given by words over X, that is, elements in the free monoid $\langle X\rangle$, and multiplication is given by concatenation of words. In particular, indeterminates still commute with coefficients, but not with each other. We refer to Section 2.2.3 for a formal introduction of the free algebra.

Intuitively, a matrix or operator identity $B = C$, or equivalently $B-C = 0$, can be identified with the polynomial $f(b, c) = b - c$. More generally, identities of composite operators can be translated into noncommutative polynomials by introducing a noncommutative indeterminate for each basic nonzero operator, and by uniformly replacing each operator by the respective indeterminate in the difference of the left- and right-hand side of each identity. Potentially present zero operators are simply replaced by the zero in $R\langle X\rangle$.

For example, to express the Penrose identities (1.1), we introduce indeterminates a, b, a^*, b^* to represent the matrices A, B, and their adjoints, and form the polynomials

$$aba - a, \qquad bab - b, \qquad b^*a^* - ab, \qquad a^*b^* - ba. \tag{1.4}$$

With this, the computation (1.3) corresponds to the polynomial statement

$$\begin{aligned} b - c = &- (bab - b) \;-\; b(aca - a)b \;-\; bac(b^*a^* - ab) \;-\; b(c^*a^* - ac)b^*a^* \\ &+ bc^*(a^*b^*a^* - a^*) \;+\; b(c^*a^* - ac) \;-\; (a^*b^* - ba)c \;-\; (a^*c^*a^* - a^*)b^*c \\ &+ a^*c^*(a^*b^* - ba)c \;+\; (a^*c^* - ca)bac \;+\; c(aba - a)c \;+\; (cac - c). \end{aligned} \tag{1.5}$$

This shows that $b-c$ can be represented as a two-sided linear combination of the polynomials encoding that B and C satisfy the Penrose identities for A.

Remark 1.2.1. *To model the involution $*$ on the polynomial level, we introduce an additional indeterminate for the adjoint of each basic operator and simplify all operator expressions using the identities* (1.2) *before translating them into polynomials. Furthermore, whenever an identity $S = T$ holds, then so does the adjoint identity $S^* = T^*$, and these additional identities have to be translated into polynomials as well. Note that the identities* (1.2) *themselves are not translated into polynomials, but they are only used to simplify the other identities.*

Thus, to express that B is the Moore-Penrose inverse of A on the polynomial level, we have to add to (1.4) *the polynomials corresponding to the adjoint identities. Since the last two Penrose identities are self-adjoint, this yields the two additional elements $a^*b^*a^* - a^*$ and $b^*a^*b^* - b^*$. We note that these additional polynomials can be essential for proofs and also appear in* (1.5).

Algebraically, the relation (1.5) means that the polynomial $b - c$ lies in the (two-sided) ideal generated by the polynomials encoding that B and C are Moore-Penrose inverses of A. We call such a representation of an ideal element in terms of the ideal's generators a *cofactor representation* (see also Definition 2.2.29).

It is always the case, that, if an operator identity follows from given identities by arithmetic operations with operators (that is, addition, composition, and scaling), then the polynomial corresponding to this identity is contained in the ideal. However, not all polynomials that lie in the ideal correspond to valid operator identities, because, in contrast to computations with actual operators, arithmetic computations with polynomials are not restricted and all sums and products can be formed. Obviously, elements of the ideal that do not comply with the formats of the matrices (or more generally, the domains and codomains of the operators) cannot correspond to identities of operators. Thus, *a priori*, when proving an operator identity by verifying *ideal membership* like in (1.5), one has to ensure that every term appearing in a cofactor representation respects the restrictions imposed by the operators.

In the case of commutative polynomials, ideal membership can be decided algorithmically by Buchberger's algorithm [Buc65], computing a Gröbner basis of the ideal. In contrast,

ideal membership of noncommutative polynomials is only semi-decidable in general. This is a consequence of the undecidability of the word problem. More precisely, *verifying* ideal membership of noncommutative polynomials is always possible, using a noncommutative analogue of Buchberger's algorithm [Mor85; MZ98] to enumerate a (possibly infinite) Gröbner basis. However, *disproving* ideal membership is not always possible. Nevertheless, if a polynomial can be verified to lie in an ideal, then, as a byproduct, a cofactor representation of the polynomial in terms of the generators can be obtained. This representation serves as a certificate for the ideal membership and can be checked independently.

Our SageMath software package `operator_gb` allows to certify ideal membership of noncommutative polynomials by computing cofactor representations. We illustrate its usage to compute the representation given in (1.5). For a more detailed description of the functionality of the package, see Section 6.1.

To generate the polynomials encoding the Penrose identities, the package provides the command `pinv`. Furthermore, it allows to automatically add to a set of polynomials the corresponding adjoint elements, using the command `add_adj`.

```
# load the package
sage: from operator_gb import *

# create free algebra
sage: F.<a, b, c, a_adj, b_adj, c_adj> = FreeAlgebra(QQ)

# generate Penrose identities for b and c
sage: Pinv_b = pinv(a, b, a_adj, b_adj)
sage: Pinv_c = pinv(a, c, a_adj, c_adj)
# add the corresponding adjoint statements
sage: assumptions = add_adj(Pinv_b + Pinv_c)

# form a noncommutative ideal
sage: I = NCIdeal(assumptions)

# verify ideal membership of the claim
sage: proof = I.ideal_membership(b-c)
```

```
# print the found cofactor representation
sage: pretty_print_proof(proof, assumptions)

 b - c = (-c + c*a*c) + b*c_adj*(-a_adj + a_adj*b_adj*a_adj)
         - b*a*c*(-a*b + b_adj*a_adj) - b*(-a + a*c*a)*b
         + b*(-a*c + c_adj*a_adj) - b*(-a*c + c_adj*a_adj)*b_adj*a_adj
         - (-b + b*a*b) + (-c*a + a_adj*c_adj)*b*a*c
         - (-a_adj + a_adj*c_adj*a_adj)*b_adj*c + c*(-a + a*b*a)*c
         - (-b*a + a_adj*b_adj)*c + a_adj*c_adj*(-b*a + a_adj*b_adj)*c
```

Remark 1.2.2. *The computed representation is equal to* (1.5) *up to a reordering of summands.*

The correctness of a computed cofactor representation can be verified easily by expanding it, which only requires basic polynomial arithmetic. Our package allows to do this using the command `expand_cofactors`.

```
# reusing the assumptions and proof from above
sage: expand_cofactors(proof, assumptions)

b - c
```

The pioneering work mentioned above exploited the fact that the operations used in the noncommutative version of Buchberger's algorithm respect the restrictions imposed by domains and codomains of operators, see [HSW98, Thm. 25] or [SL20, Thm. 1]. Thus, using Buchberger's algorithm, proving an operator identity can be reduced to verifying ideal membership of the corresponding polynomial.

Only recently it was observed that, in fact, any verification of ideal membership, even one that does not comply with the domains and codomains of the operators, allows to deduce a correct statement about linear operators, provided that all initial polynomials correspond to actual operator identities [RRH21]. This implies that the verification of the ideal membership can be done completely independently of the operator context.

In particular, this also means that the cofactor representation given in (1.5) immediately yields the uniqueness statement of the Moore-Penrose inverse of a complex matrix. Moreover, since the polynomial computation is independent of the concrete operator context, this representation also proves a corresponding statement in every setting where it can be formulated. For example, we immediately obtain the uniqueness of the Moore-Penrose inverse for elements in arbitrary rings with involution, for bounded linear operators on Hilbert spaces, or for elements in C^*-algebras.

We can summarise the discussion of this section as follows. In the following, we identify each identity of linear operators $S = T$ with the noncommutative polynomial $s - t$ using the translation described above. An identity $P = Q$ of linear operators follows from other identities $S_1 = T_1, \ldots, S_m = T_m$ if and only if the noncommutative polynomial $p - q$ lies in the ideal generated by $s_1 - t_1, \ldots, s_m - t_m$ in the free algebra $\mathbb{Z}\langle X\rangle$. In Section 5.1, we provide with Theorem 5.1.7 a theoretical justification for this conclusion, see also Procedure 8 and Theorem 4.5.2.

Thus far, our software package only supports computations in the free algebra $\mathbb{Q}\langle X\rangle$. To ensure that the computations are also valid over $\mathbb{Z}\langle X\rangle$ as required, one has to check whether all coefficients that appear in the computed cofactor representation are in fact integers. The following routine `certify` builds a user-friendly wrapper around the ideal membership verification that also includes these checks. It raises a warning if non-integer coefficients appear in the computed cofactor representation.

```
sage: F.<a, b, c, a_adj, b_adj, c_adj> = FreeAlgebra(QQ)
sage: Pinv_b = pinv(a, b, a_adj, b_adj)
sage: Pinv_c = pinv(a, c, a_adj, c_adj)
sage: assumptions = add_adj(Pinv_b + Pinv_c)
sage: proof = certify(assumptions, b-c)
Computing a (partial) Gröbner basis and reducing the claims...

Done! Ideal membership of all claims could be verified!
```

Remark 1.2.3. *We note that the computed* `proof` *is the same as that computed by the* `ideal_membership` *routine before.*

In many situations, all involved identities are of the form $S = T$, where S and T are compositions of basic operators or zero, as, for example, in case of the Penrose identities (1.1). In such cases, all involved polynomials are binomials of the form $s - t$, where s and t are monomials in $\langle X \rangle$ or zero. For these scenarios, `certify` is guaranteed to compute a cofactor representation with integer coefficients, provided that one exists. However, for arbitrary polynomials, it could happen that `certify` only discovers a cofactor representation with rational coefficients, even if an alternative representation with integer coefficients exists. We note, however, that in all the examples we have considered thus far, this situation has never occurred.

1.3 Treating existential statements

The method described in the previous section allows to verify whether an operator identity follows from other identities by checking ideal membership of noncommutative polynomials. Although this technique is useful for proving various nontrivial statements, it still has its limitations. Specifically, it does not cover existential statements that arise, for example, when solving operator equations. This type of statement requires an extended approach and cannot be proven solely by checking ideal membership. In the following, we discuss how to treat existential statements. As an illustrative example, we consider the existence of the Moore-Penrose inverse for complex matrices. More precisely, we show that every complex matrix has a Moore-Penrose inverse, using polynomial computations.

In more general settings (for example, bounded linear operators on Hilbert spaces, elements in C^*-algebras), not every element has a Moore-Penrose inverse. Therefore, a crucial step in proving the desired statement is to characterise the fact that we are considering (complex) matrices. In particular, since the polynomial framework can only deal with identities of operators, we have to express this fact in terms of identities. One possibility to do this is via the singular value decomposition, which implies that, for every complex matrix A, there exist matrices P, Q with

$$PA^*A = A \qquad \text{and} \qquad AA^*Q = A. \tag{1.6}$$

For example, if $A = U\Sigma V^*$ is a singular value decomposition of A, then $P = Q = U\Sigma^+ V^*$ is a possible choice, where Σ^+ is obtained from Σ by replacing the nonzero diagonal entries

by their reciprocals, and thus satisfies $\Sigma\Sigma^+\Sigma^* = \Sigma^*\Sigma^+\Sigma = \Sigma$. Using this property of matrices, we can formalise the statement we consider in this section as the first-order formula

$$\forall A, P, Q\ \exists B\ :\ (PA^*A = A\ \wedge\ AA^*Q = A)\implies (1.1).$$

In the polynomial framework, the only possibility to prove such an existential statement is to derive an explicit expression for the existentially quantified objects. Once such an explicit expression is obtained, the statement can be reformulated as a basic statement concerning identities and treated like in the previous section.

For our example, this means finding an expression for B in terms of A, P, Q and their adjoints such that (1.1) holds modulo the assumptions (1.6). Algebraically, this corresponds to finding a polynomial $b = b(a, p, q, a^*, p^*, q^*)$ such that the elements (1.4), representing the Penrose identities (1.1), lie in the ideal generated by

$$pa^*a - a, \qquad aa^*q - a, \qquad a^*ap^* - a^*, \qquad q^*aa^* - a^*, \tag{1.7}$$

encoding the assumptions (1.6).

Through the use of Gröbner basis techniques, it is possible to employ a number of heuristics for finding elements of certain form in noncommutative polynomial ideals, see Section 5.3. One such approach involves introducing a dummy variable x for the desired expression b. With this dummy variable, we consider the ideal I generated by the assumptions (in our example given by (1.7)) and by the identities that b shall satisfy, but with b replaced by x (in our example these are the Penrose identities (1.4) for x). Every polynomial of the form $x - b'$ in I corresponds to a candidate expression b' for b, and by applying the elimination property of Gröbner bases (Theorem 2.4.43), we can systematically search for such candidate expressions. Our software package offers a user-friendly interface that simplifies the process of searching for expressions of this nature.

```
sage: F.<a, p, q, a_adj, p_adj, q_adj, x, x_adj> = FreeAlgebra(QQ)
sage: assumptions =  add_adj([a - p*a_adj*a, a - a*a_adj*q])
sage: Pinv_x = add_adj(pinv(a, x, a_adj, x_adj))
sage: I = NCIdeal(Pinv_x + assumptions)
```

```
sage: I.find_equivalent_expression(x)

                        [- x + a_adj*p*x, - x + a_adj*q*x,
                - x + a_adj*x_adj*x, - x + a_adj*q*p_adj]
```

Three out of the four candidate expressions for b found by the heuristic still contain the dummy variable x or its adjoint, and are thus useless. However, the last polynomial $x - a^*qp^*$ shows that $b = a^*qp^*$ is a desired representation. We use our software to show that b satisfies the Penrose identities under the assumptions (1.7).

```
sage: MP_candidate = a_adj * q * p_adj
sage: MP_candidate_adj = p * q_adj * a
sage: claims = pinv(a, MP_candidate, a_adj, MP_candidate_adj)
sage: proof = certify(assumptions, claims)
Computing a (partial) Gröbner basis and reducing the claims...

Done! Ideal membership of all claims could be verified!
```

We note that, here, `claims` is a list consisting of four polynomials, one for each of the four Penrose identities. In such cases, `certify` verifies the ideal membership of each element in the list and returns a list, here assigned to `proof`, providing a cofactor representation for each polynomial in `claims`.

Thus, we can conclude that every complex matrix A has a Moore-Penrose inverse $A^\dagger$, given by $A^\dagger = A^*QP^*$ with P, Q as in (1.6). More generally, we have proven that, for every linear operator A, if there exist linear operators P and Q satisfying (1.6), then A has a Moore-Penrose inverse $A^\dagger$, given by $A^\dagger = A^*QP^*$.

Remark 1.3.1. *Typically, under the assumptions* (1.6) *the Moore-Penrose inverse is expressed by the formula* $A^\dagger = Q^*AP^*$*, see, for example, [PR81, Lem. 3]. We note that this expression is equivalent to ours, and can be found using our software by changing the monomial order underlying the polynomial computation.*

```
sage: I.find_equivalent_expression(x,
....: order=[[q,q_adj,a,a_adj,p,p_adj],[x,x_adj]])[0]

                                - q_adj*a*p_adj + x
```

Using yet another monomial order, we can also find the element `-q_adj*p*a_adj + x` *corresponding to another equivalent solution* $A^\dagger = Q^*PA^*$. *This gets to show that the output of the polynomial heuristics depends strongly on several parameters, and in particular, on the used monomial order.*

We could prove the existence of the Moore-Penrose inverse by explicitly constructing an expression for the existentially quantified operator. This now raises the question whether this is always possible or whether we just got lucky in this example. *Herbrand's theorem* [Her30; Bus94], a fundamental result in mathematical logic, provides an answer to this question. It states that such an explicit representation always exists and can be constructed as a polynomial expression in terms of the basic operators appearing in the statement, provided that the operator statement is indeed true. We refer to Theorem 4.2.2 for the precise statement. Thus, by enumerating all such polynomial expressions, we are guaranteed to find a correct instantiation if the considered statement is correct.

Of course, naively enumerating all possible polynomial expressions quickly becomes infeasible. Therefore, it is important to have good heuristics that allow to systematically search for suitable candidate expressions. Our software package implements, apart from the heuristic described above, several such techniques for finding polynomials of special form in noncommutative ideals. We refer to Section 5.3 for further information on these methods and to Section 6.1.1 for the corresponding commands.

2 Preliminaries

For the convenience of the reader, in this chapter, we recall and discuss relevant algebraic and logical concepts required in this work. We assume a basic familiarity with fundamental algebraic notions, including abelian groups, rings, and modules over commutative rings.

We begin by revisiting some key terminology from category theory in Section 2.1. Specifically, we recall the notions of preadditive semicategories and abelian categories, and also give several classical examples of these structures. For a thorough introduction to category theory, we refer, for example, to [Mac13]. Section 2.2 is devoted to defining the algebraic structures relevant for the theory and algorithms developed in the subsequent chapters. We discuss the construction and fundamental properties of free monoids, free algebras, and free (bi)modules. For further information on these topics, see also [Row91; Coh03]. Following upon that, we review essential notions and results from the field of abstract rewriting, required for developing a theory of (signature) Gröbner bases. We give a minimal and complete survey of these results in Section 2.3. Our presentation follows [BN98, Sec. 2.1], to which we refer for further details. Building upon the concepts discussed in the previous sections, we summarise the main results of the theory of Gröbner bases for one- and two-sided ideals in the free algebra over a coefficient field in Section 2.4. Here, we provide a complete and self-contained presentation based on the concept of polynomial reduction. In particular, we combine terminology from abstract rewriting with Bergman's original presentation of noncommutative Gröbner bases [Ber78], viewing polynomial reduction as a family of linear maps. Notably, we also give a novel characterisation of a large class of noncommutative monomial orders by combining totally ordered semigroups with the lexicographic order and a result by Higman [Hig52]. This characterisation encompasses all classical orders, including, for example, (weighted) (multi-)degree orders and elimination orders. For additional resources on noncommutative Gröbner bases, we point interested readers to [Mor94; Xiu12; BGV13; Mor16], and also to [LSA20] for an overview on available software and further references. In particular, see [BGV13, Sec. 2.1, 3.2] and [Mor16, Sec. 48.7] for further

information on noncommutative monomial orders. Finally, in Section 2.5, we provide a comprehensive and self-contained introduction to many-sorted first-order logic, which serves as a framework for the developments in Chapter 4. Our presentation is based on [Man93], while also incorporating concepts and notation from [EFT21]. For further references and a historic overview of the development of many-sorted logic, we refer to [Wal87, Ch. 13].

2.1 Category theory

Category theory is a branch of mathematics that provides a powerful and abstract framework for studying mathematical structures and relationships between them. For a textbook exposition of category theory, we refer, for example, to [Mac13]. At its core, category theory explores the notion of a category, or more generally, that of a semicategory. We will consider the particular case of preadditive semicategories.

Definition 2.1.1. *A* (locally small) semicategory $\mathcal{C}$ *(also called* semigroupoid*) consists of*

- *a class* $\mathrm{Ob}(\mathcal{C})$ *of* objects*;*
- *for every two objects* $U, V \in \mathrm{Ob}(\mathcal{C})$*, a set* $\mathrm{Mor}(U, V)$ *of* morphisms *from* U *to* V*; for* $f \in \mathrm{Mor}(U, V)$*, we also write* $f\colon U \to V$*; the objects* U *and* V *are referred to as the* source *and* target *of* f*;*
- *for every three objects* $U, V, W \in \mathrm{Ob}(\mathcal{C})$*, a binary operation* $\circ\colon \mathrm{Mor}(V, W) \times \mathrm{Mor}(U, V) \to \mathrm{Mor}(U, W)$ *called* composition of morphisms*, which is associative, that is, if* $f\colon V \to W$*,* $g\colon U \to V$*,* $h\colon T \to U$*, then* $f \circ (g \circ h) = (f \circ g) \circ h$*;*

A semicategory $\mathcal{C}$ *is called* preadditive *if every set* $\mathrm{Mor}(U, V)$ *is equipped with a binary operation* $+$*, turning it into an abelian group, such that composition of morphisms is bilinear, that is,*

$$f \circ (g + h) = (f \circ g) + (f \circ h) \quad \text{and} \quad (f + g) \circ h = (f \circ h) + (g \circ h).$$

A semicategory can be thought of as a collection of objects, linked by arrows (the morphisms) that can be composed associatively. The property of being locally small refers to the fact that the objects $\mathrm{Ob}(\mathcal{C})$ can form a proper class, while, for all objects $U, V \in \mathrm{Ob}(\mathcal{C})$, the

morphisms $\mathrm{Mor}(U, V)$ constitute a set. Preadditive semicategories have the additional property that arrows with the same start and end can be added, yielding an abelian group structure that is compatible with the composition of morphisms. For further information, see, for example, [Gar05, Sec. 2] or [Til87, App. B]. We also note that the words *object* and *morphism* do not imply anything about the nature of these things. Objects can be anything from numbers and sets to more complex mathematical structures like groups, vector spaces, or topological spaces. Morphisms capture the relationships between objects and can be thought of as mappings or transformations.

Preadditive semicategories provide a natural and very general environment for studying linear operators, prescribing only linearity as a structural constraint. In particular, they encompass all the following settings.

Example 2.1.2. *In the following, R denotes a ring (not necessarily with 1).*

1. *The ring R can be considered as a preadditive semicategory with only one object, and thus, only a single set of morphisms consisting of the underlying abelian group of R. Composition of morphisms is given by the ring multiplication.*

2. *The set $\boldsymbol{Mat}(R)$ of matrices with entries in R can be considered as a preadditive category by taking as objects the sets R^n for all positive natural numbers n and letting $\mathrm{Mor}(R^n, R^m) = R^{m\times n}$. Composition is given by matrix multiplication.*

3. *The category R-**Mod** of left modules over R is a preadditive semicategory. Here, objects are left R-modules and morphisms are module homomorphisms between left R-modules. As a special case, we see that K-**Vect**, the category of vector spaces over a field K, is a preadditive semicategory. Note that the objects in these categories form proper classes and not sets.*

4. *Another example of a preadditive semicategory is the category **Ab** of abelian groups, which has all abelian groups as objects and group homomorphisms as morphisms.*

Categories are semicategories with *identity morphisms*.

Definition 2.1.3. *A semicategory $\mathcal{C}$ is a* category *if, for every object $X \in \mathrm{Ob}(\mathcal{C})$, there exists a morphism $1_X \colon X \to X$ (also denoted id_X) called the* identity morphism *for X, such that, for every $f \colon U \to V$, we have*

$$1_V \circ f = f = f \circ 1_U.$$

One can show that an identity morphism for X is unique if it exists. The proof is analogous to that of the uniqueness of the identity element in a monoid.

A *semicategory with involution* is a semicategory $\mathcal{C}$ together with a mapping *, the so-called *involution*, that sends every morphism f in $\mathcal{C}$ to a morphism f^* and satisfies the following conditions:

- if $f \colon U \to V$, then $f^* \colon V \to U$;
- $(f^*)^* = f$;
- if $f \colon V \to W$ and $g \colon U \to V$, then $(f \circ g)^* = g^* \circ f^*$;

In particular, if $\mathcal{C}$ is a category and 1_X is the identity morphism on an object $X \in \mathrm{Ob}(\mathcal{C})$, then

$$1_X^* = 1_X \circ 1_X^* = (1_X^*)^* \circ 1_X^* = (1_X \circ 1_X^*)^* = (1_X^*)^* = 1_X.$$

2.1.1 Abelian categories

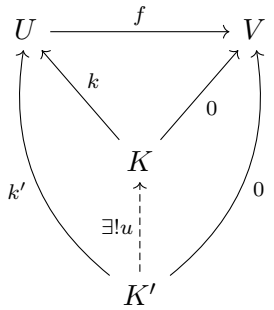
An important class of categories, particularly relevant in the field of homological algebra, is that of *abelian categories*. To introduce abelian categories, we first recall several classical notions from category theory. In what follows, $\mathcal{C}$ is a category.

A morphism $m \colon U \to V$ is *monic* (or a *monomorphism*) in $\mathcal{C}$ if, for any two morphisms $f, g \colon T \to U$, the equality $m \circ f = m \circ g$ implies $f = g$. In other words, m is monic if it can always be cancelled on the left. Dually, a morphism $e \colon U \to V$ is *epi* (or an *epimorphism*) if, for any two morphisms $f, g \colon V \to W$, the equality $f \circ e = g \circ e$ implies $f = g$. Equivalently, e is epi if it can always be cancelled on the right. A morphism $f \colon U \to V$ is *invertible* (or an *isomorphism*) if there exists a morphism $g \colon V \to U$ with $g \circ f = 1_U$ and $f \circ g = 1_V$. If such a g exists, it is unique and denoted by f^{-1}. Two objects U and V are *isomorphic* if there exists an invertible morphism $f \colon U \to V$. Every isomorphism is also a mono- and

epimorphism. The converse, however, does not hold in all categories; a morphism which is both monic and epi need not be an isomorphism.

An object $Z \in \mathrm{Ob}(\mathcal{C})$ is a *zero object* if, for every object $U \in \mathrm{Ob}(\mathcal{C})$, there is exactly one morphism $f\colon U \to Z$ and exactly one morphism $g\colon Z \to U$. It is straightforward to verify that a zero object is unique up to isomorphism, if it exists. Furthermore, if $\mathcal{C}$ has a zero object, then, for every two objects $U, V \in \mathrm{Ob}(\mathcal{C})$, the unique morphism $U \to Z \to V$ is called the *zero morphism* from U to V, denoted by $0_{U,V}$. If $\mathcal{C}$ is preadditive, the zero morphisms $0_{U,V}$ are precisely the identity elements in the abelian groups $\mathrm{Mor}(U, V)$. Composition of any morphism with a zero morphism is itself a zero morphism. In the following, for ease of notation, we use the same symbol 0 for all zero morphisms, if the sources and targets are clear from the context.

Let $\mathcal{C}$ have a zero object. A *kernel* of a morphism $f\colon U \to V$ is a morphism $k\colon K \to U$, where $K \in \mathrm{Ob}(\mathcal{C})$, such that $f \circ k = 0$ and such that, for every other morphism $k'\colon K' \to U$ with $K' \in \mathrm{Ob}(\mathcal{C})$ and $f \circ k' = 0$, there exists a unique morphism $u\colon K' \to K$ with $k \circ u = k'$. The properties of a kernel can be characterised by the following commutative diagram.



Dually, a *cokernel* of $f\colon U \to V$ is a morphism $c\colon V \to C$, where $C \in \mathrm{Ob}(\mathcal{C})$, such that $c \circ f = 0$ and such that, for every other morphism $c'\colon V \to C'$ with $C' \in \mathrm{Ob}(\mathcal{C})$ and $c' \circ f = 0$, there exists a unique morphism $u\colon C \to C'$ with $u \circ c = c'$. One can show that kernels and cokernels are unique up to isomorphism. The kernel and cokernel of f are denoted by $\ker(f)$ and $\mathrm{coker}(f)$, respectively.

Next, we consider *biproducts* in a preadditive category $\mathcal{C}$. A *(binary) biproduct* for two objects $U, V \in \mathrm{Ob}(\mathcal{C})$ is a diagram

$$U \underset{i}{\overset{p}{\leftrightarrows}} W \underset{j}{\overset{q}{\leftrightarrows}} V$$

with morphisms p, q, i, j satisfying

$$p \circ i = 1_U, \qquad q \circ j = 1_V, \qquad i \circ p + j \circ q = 1_W.$$

A biproduct need not exist for all objects $U, V \in \mathrm{Ob}(\mathcal{C})$, but if it does, it is unique up to an isomorphism of the object W.

Combining all the notions discussed above, we can finally define abelian categories.

Definition 2.1.4. *A preadditive category $\mathcal{C}$ is called* abelian *if it satisfies the following conditions.*

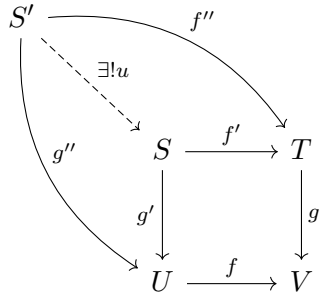
1. *$\mathcal{C}$ has a zero object;*
2. *$\mathcal{C}$ has all binary biproducts;*
3. *every morphism in $\mathcal{C}$ has a kernel and a cokernel;*
4. *every monomorphism is a kernel, and every epimorphism is a cokernel;*

Example 2.1.5.

1. *The prototypical example of an abelian category is the category **Ab** of abelian groups. The zero object of **Ab** is the trivial group $\{0\}$ and the notion of kernel in the categorical sense coincides with kernel in the algebraic sense. More precisely, the categorical kernel of a morphism $f\colon U \to V$ is the subgroup K of U defined by $K = \{x \in U : f(x) = 0\}$ together with the inclusion homomorphism $k\colon K \to U$. The cokernel of f is the quotient group $C = V/f(U)$ together with the natural projection $c\colon V \to C$. Furthermore, the biproduct is given by the direct sum with natural projections and inclusions of each factor.*
2. *The category R-**Mod** of left modules over a ring R is an abelian category. The zero object is given by the trivial module, defined as the trivial group equipped with the trivial R-action. Kernels and cokernels are given by the corresponding algebraic*

objects, like in the case of ***Ab****. As a special case, we see that also the category* K*-****Vect*** *of vector spaces over a field* K *is an abelian category.*

It follows from the first three conditions of Definition 2.1.4 that an abelian category has all *pullbacks*. More generally, an abelian category has all *finite limits*. In the following, however, we only care about pullbacks and refer to [Mac13, Sec. VIII.3] for further details. The *pullback* of two morphisms $f\colon U \to V$ and $g\colon T \to V$ in a category $\mathcal{C}$ is an object $S \in \mathrm{Ob}(\mathcal{C})$ together with two morphisms $f'\colon S \to T$ and $g'\colon S \to U$ such that $f \circ g' = g \circ f'$. Furthermore, for every other object $S' \in \mathrm{Ob}(\mathcal{C})$ and pair of morphisms $f''\colon S' \to T$ and $g''\colon S' \to U$ with $f \circ g'' = g \circ f''$, there exists a unique $u\colon S' \to S$ such that $f'' = f' \circ u$ and $g'' = g' \circ u$. The properties of a pullback can be summarised by the following commutative diagram.



In any category $\mathcal{C}$, the morphism f' is monic if f is, and, in an abelian category, f' is epi if f is.

Condition 4 in the definition of abelian categories also implies, among other things, that any morphism that is both monic and epi is an isomorphism. Furthermore, in an abelian category, every morphism f has a *natural factorisation* as $f = m \circ e$ with m monic and e epi. Moreover,

$$m = \ker(\mathrm{coker}(f)), \qquad e = \mathrm{coker}(\ker(f)).$$

Based on this factorisation, the image and coimage of f are defined uniquely up to isomorphism as

$$\mathrm{im}(f) = \ker(\mathrm{coker}(f)), \qquad \mathrm{coim}(f) = \mathrm{coker}(\ker(f)).$$

Note that $\mathrm{im}(f)$ is a monomorphism. More generally, every kernel is necessarily monic. This follows from the unique factorisation requirement in the definition of kernels.

One of the central notions in abelian categories is that of an *exact sequence*, which is based on an equivalence of monomorphisms. If $f\colon U \to V$ and $g\colon T \to V$ are two monomorphisms with a common target V, we write $f \leqq g$ if $f = g \circ h$ for some $h\colon U \to T$. When both $f \leqq g$ and $g \leqq f$, we write $f \equiv g$. This defines an equivalence relation on monomorphisms with target V, and the equivalence classes are called *subobjects* of V.

Definition 2.1.6. *Let $\mathcal{C}$ be an abelian category. A sequence of morphisms*

$$U \xrightarrow{\quad f \quad} V \xrightarrow{\quad g \quad} W$$

is exact *(at V) if* $\operatorname{im}(f) \equiv \ker(g)$ *(equivalence as subobjects of V).*

2.2 Algebraic structures

In this section, we recall basic algebraic notions and constructions relevant for the theory developed in the subsequent chapters. In particular, we discuss the construction and properties of free monoids, free algebras and free (bi)modules.

2.2.1 Free monoid

First, we study the free monoid on a set X, consisting of all finite words over the alphabet X. To this end, we first recall the definition of a monoid.

Definition 2.2.1. *A* semigroup *is a nonempty set M equipped with an associative binary operation $*\colon M \times M \to M$ (typically called* multiplication*). If, additionally, there exists an element $e \in M$ satisfying $e * x = x * e = x$ for all $x \in M$, then M is called a* monoid*.*

Remark 2.2.2. *To emphasise the binary operation $*$ on the set M, we also say that the pair $(M, *)$ is a semigroup or a monoid. This emphasis is important when a set can be equipped with different operations, leading to possibly different structures.*

The element e in Definition 2.2.1 is called the *identity element* or the *neutral element* of the monoid, and sometimes also denoted by 1. One can show that it is unique. In the following, we write xy for the product $x * y$ in a semigroup or monoid, and, for $n \in \mathbb{N}$, we abbreviate the n-fold product of x with itself by x^n, that is, $x^n \coloneqq \underbrace{x * \cdots * x}_{n \text{ times}}$, with the special case $x^0 \coloneqq e$.

The second notion needed to define free monoids is that of a word over an alphabet X.

Definition 2.2.3. *An* alphabet *is a set X and a* word *over X is a finite sequence $w = x_1 \dots x_d$ with $d \in \mathbb{N}$ and $x_i \in X$ for $i = 1, \dots, d$. For $d = 0$, we obtain the empty sequence, also called the* empty word*, which we denote by* 1*. The quantity d is called the* length *of w and denoted by $|w|$. The set of all words over X is denoted by $\langle X \rangle$.*

A basic binary operation on $\langle X \rangle$ is that of concatenation,

$$(x_1 \dots x_d, x'_1 \dots x'_{d'}) \mapsto x_1 \dots x_d x'_1 \dots x'_{d'}.$$

It is straightforward to check that this operation is associative and that the empty word 1 acts as an identity element. Consequently, the set $\langle X \rangle$ equipped with the operation of concatenation is a monoid. More precisely, we arrive at the following definition.

Definition 2.2.4. *Let X be a set. The set $\langle X \rangle$ together with the binary operation of concatenation is called the* free monoid *or* word monoid *over X.*

Example 2.2.5. *Let $X = \{x, y, z\}$. The elements of $\langle X \rangle$ are of the form*

$$1, x, y, z, xx, yy, zz, xy, yx, xz, zx, yz, zy, xxx, \dots$$

In $\langle X \rangle$, we have, for example, $xy \cdot z = xyz \neq zxy = z \cdot xy$.

Remark 2.2.6. *If X is a singleton $X = \{x\}$, then $\langle X \rangle = \{x^n \mid n \in \mathbb{N}\}$ consists of all nonnegative powers of x and since $x^i x^j = x^{i+j} = x^{j+i} = x^j x^i$ for all $i, j \in \mathbb{N}$, we end up with a commutative monoid. If $|X| > 1$, then $\langle X \rangle$ is noncommutative.*

If the elements of X are given explicitly, say $X = \{x_1, \ldots, x_n\}$, we omit the set parentheses and simply write $\langle x_1, \ldots, x_n \rangle$ for $\langle \{x_1, \ldots, x_n\} \rangle$. Furthermore, for sets $X_1, \ldots, X_k$, let $\langle X_1, \ldots, X_k \rangle = \langle X_1 \cup \cdots \cup X_k \rangle$.

The multiplication in $\langle X \rangle$ naturally induces a notion of divisibility of words.

Definition 2.2.7. *Let X be a set and $w, w' \in \langle X \rangle$. We say that w'* divides *w, or w* is divisible by *w', if there exist $a, b \in \langle X \rangle$ such that*

$$w = aw'b.$$

In this case, we call a, b the cofactors *of the division. If w' divides w, we also equivalently say that w is a* multiple *of w'. If $a = 1$, then w' is a* prefix *of w, and if $b = 1$, then w' is a* suffix *of w. Furthermore, w' is a* proper *prefix (resp. suffix) of w if it is a prefix (resp. suffix) of w and $w' \neq w$.*

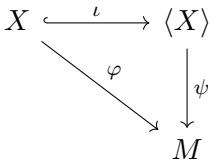
Example 2.2.8. *Let $X = \{x, y, z\}$ and $w = xyxz$, $w' = yx \in \langle X \rangle$. Then w' divides w, since $aw'b = w$ with cofactors $a = x$ and $b = z$. Also $w'' = xz \in \langle X \rangle$ divides w, and, in fact, w'' is a proper suffix of w.*

Remark 2.2.9. *Note that, unlike in the commutative case, a word w' can divide a word w in several ways, causing the* cofactors *a and b to be non-unique in general. For example, consider the words $w = xyxyx$ and $w' = xy$ in $\langle x, y \rangle$. Clearly w' divides w, but we have $w = w'xyx = xyw'x$.*

Definition 2.2.10. *Let M, N be semigroups. A function $\psi \colon M \to N$ is a* semigroup homomorphism *if $\psi(xy) = \psi(x)\psi(y)$ for all $x, y \in M$. If M and N are monoids and ψ additionally satisfies $\psi(e_M) = e_N$, where e_M and e_N are the identity elements in M and N respectively, then ψ is a* monoid homomorphism.

The free monoid $\langle X \rangle$ over X earns the attribute *free* because it is the "generic" monoid over X, where elements of X do not satisfy any nontrivial relations. In particular, it satisfies the following *universal property.*

Theorem 2.2.11 (Universal property of free monoids)**.** *Let X be a set, M be a monoid, and ι be the canonical injection from X into $\langle X \rangle$. For any map $\varphi\colon X \to M$, there exists a unique monoid homomorphism $\psi\colon \langle X \rangle \to M$ such that the following diagram commutes.*



In particular, we have $\psi(x_1 \dots x_d) = \varphi(x_1) \dots \varphi(x_d)$.

2.2.2 Free module

In this section, we recall the notion of a free module. In the following, R is a (not necessarily commutative) ring. We note that we only consider left R-modules in the following.

Definition 2.2.12. *Let M be an R-module. A set $B \subseteq M$ is a* basis *of M if it satisfies the following two conditions:*

1. *B is a* generating set *of M, that is, every element in M is a finite R-linear combination of elements of B;*
2. *B is* linearly independent*, that is, for every subset $\{b_1, \dots, b_n\} \subseteq B$ of distinct elements and $r_1, \dots, r_n \in R$, the R-linear combination $\sum_{i=1}^{n} r_i b_i$ is zero only if $r_i = 0$ for all $i = 1, \dots, n$;*

A module M is called free *if it has a basis.*

An important example of a free module is *the free module* on a set X.

Definition 2.2.13. *Let X be a set. The* free R-module *on the set X is the set of finite formal sums*

$$RX = \left\{ \sum_{x \in X} r_x x \;\middle|\; r_x \in R \text{ and } r_x = 0 \text{ for almost all } x \right\}$$

together with the addition and scalar multiplication

$$\sum_{x\in X} r_x x + \sum_{x\in X} s_x x = \sum_{x\in X} (r_x + s_x)\, x \qquad\qquad r \sum_{x\in X} r_x x = \sum_{x\in X} (r r_x)\, x.$$

Remark 2.2.14. *Alternatively, the free R-module on X can be constructed as the set of all functions from X to R with finite support, that is,*

$$RX = \{ f \colon X \to R \mid f(x) = 0 \text{ for almost all } x \},$$

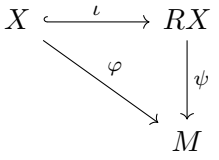
with point-wise addition $(f+g)(x) = f(x)+g(x)$ and scalar multiplication $(rf)(x) = rf(x)$.

One can verify by a direct computation that RX is indeed an R-module and that it is free with basis X. In fact, one can show that RX is – up to isomorphism – the only free R-module with basis X.

Proposition 2.2.15. *For a set X, any free R-module with basis X is isomorphic to RX.*

This is a consequence of the following universal property of the free R-module.

Theorem 2.2.16 (Universal property of free modules)**.** *Let X be a set, M be an R-module, and ι be the canonical injection from X into RX. For any map $\varphi \colon X \to M$, there exists a unique R-module homomorphism $\psi \colon RX \to M$ such that the following diagram commutes:*



In particular, we have

$$\psi\Big(\sum_{x\in X} r_x x\Big) = \sum_{x\in X} r_x \varphi(x).$$

2.2.3 Free algebra

In the following, we recall how to construct, starting from the free monoid on a set X, a ring, the *free (associative) algebra* on X. We also discuss several properties of the free algebra. In the following, R denotes a commutative ring (with unity).

Definition 2.2.17. *Let X be a set. The* free (associative) algebra *on X over R (or the* ring of noncommutative polynomials *in the indeterminates X with coefficients in R) is the free R-module $R\langle X\rangle$ on the free monoid $\langle X\rangle$ together with the multiplication*

$$\left(\sum_{u\in\langle X\rangle} c_u u\right)\left(\sum_{v\in\langle X\rangle} c'_v v\right) = \sum_{w\in\langle X\rangle}\sum_{uv=w} (c_u c'_v)\, w.$$

Elements in $R\langle X\rangle$ are called (noncommutative) polynomials.

Remark 2.2.18. *If X is a singleton $X = \{x\}$, we have already seen that $\langle X\rangle$ is a commutative monoid consisting of all nonnegative powers of x. In this case, $R\langle X\rangle$ is the usual univariate polynomial ring $R[x]$, and in particular, commutative. If $|X| > 1$, then $R\langle X\rangle$ is a noncommutative ring.*

Example 2.2.19. *Let $R = \mathbb{Z}$ and $X = \{x, y\}$. We consider $R\langle X\rangle = \mathbb{Z}\langle x, y\rangle$. For $f_1 = xy + x$, $f_2 = xy - 2y \in \mathbb{Z}\langle x, y\rangle$, we can compute*

$$\begin{aligned} f_1 + f_2 &= 2xy + x - 2y, \\ f_1 f_2 &= (xy + x)(xy - 2y) = xyxy + xxy - 2xyy - 2xy, \\ f_2 f_1 &= (xy - 2y)(xy + x) = xyxy + xyx - 2yxy - 2yx. \end{aligned}$$

Note that $f_1 f_2 \neq f_2 f_1$.

Definition 2.2.20. *Let $f = \sum_{w\in\langle X\rangle} c_w w \in R\langle X\rangle$ be a polynomial. The* coefficient $c_w \in R$ *of the* monomial w *in f is denoted by* $\operatorname{coeff}(f, w)$. *Furthermore, the* support *of f is the set* $\operatorname{supp}(f) = \{w \in \langle X\rangle \mid \operatorname{coeff}(f, w) \neq 0\}$.

One can verify that the free algebra is indeed a ring. Moreover, the multiplication in $R\langle X\rangle$ is R-bilinear, that is,

$$r \cdot (fg) = (r \cdot f)g = f(r \cdot g),$$

for all $r \in R$ and $f, g \in R\langle X\rangle$. This shows that the free algebra $R\langle X\rangle$ is in fact not only a ring, but an R-algebra as defined below.

Definition 2.2.21. *An* (associative) R-algebra A *is a ring that is also an R-module satisfying*

$$r \cdot (ab) = (r \cdot a)b = a(r \cdot b)$$

for all $r \in R$ and $a, b \in A$. The algebra A is commutative *if its multiplication is commutative.*

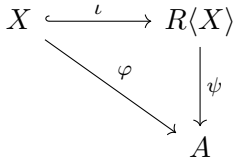
Definition 2.2.22. *Let A, B be R-algebras. A function $\psi\colon A \to B$ is an R-*algebra homomorphism *if it satisfies the following conditions:*

1. $\psi(ra + sb) = r\psi(a) + s\psi(b)$ *for all $r, s \in R$, $a, b \in A$;*
2. $\psi(ab) = \psi(a)\psi(b)$ *for all $a, b \in A$;*
3. $\psi(1) = 1$*;*

Remark 2.2.23. *The first condition of Definition 2.2.22 says that ψ is an R-module homomorphism (or an R-linear map), and the other two conditions say that ψ is a ring homomorphism.*

Just like the free monoid is the generic monoid over a set X, the free algebra is the generic R-algebra on X, satisfying an analogous universal property.

Theorem 2.2.24 (Universal property of free algebras)**.** *Let X be a set, A be an R-algebra, and ι be the canonical injection from X into $R\langle X\rangle$. For any map $\varphi\colon X \to A$, there exists a unique R-algebra homomorphism $\psi\colon R\langle X\rangle \to A$ such that the following diagram commutes:*



In particular, if I is an index set and $X = \{x_i \mid i \in I\}$, then

$$\psi\Big(\sum_{\substack{d\in\mathbb{N}\\ i_1,\dots,i_d\in I}} c_{i_1,\dots,i_d} x_{i_1}\dots x_{i_d}\Big) = \sum_{\substack{d\in\mathbb{N}\\ i_1,\dots,i_d\in I}} c_{i_1,\dots,i_d}\varphi(x_{i_1})\dots\varphi(x_{i_d}).$$

The main algebraic objects of interest in the subsequent chapters are one- and two-sided ideals. We recall their definition here. In the following, R still denotes a commutative ring and S is a (not necessarily commutative) ring.

Definition 2.2.25. *A subset $I \subseteq S$ is a* left *(resp.* right*)* ideal *of S if it satisfies the following conditions.*

1. *I is nonempty;*
2. *I is closed under addition, that is, $f + g \in I$, for all $f, g \in I$;*
3. *I is closed under left (resp. right) multiplication by arbitrary ring elements, that is, $sf \in I$, (resp. $fs \in I$) for all $s \in S$ and $f \in I$;*

If I is a right (resp. left) ideal of S, we write $I \trianglelefteq_r S$ (resp. $I \trianglelefteq_l S$). Furthermore, I is a (two-sided) ideal *of S, denoted by $I \trianglelefteq S$, if it is a left and right ideal.*

It is straightforward to check that every left/right/two-sided ideal of S forms an additive subgroup of S.

Example 2.2.26. *Every ring S has two trivial ideals, namely the* trivial ideal $\{0\}$ *and S itself.*

For the rest of this thesis, when working with one-sided ideals, we restrict ourselves to right ideals, since the situation for left ideals is completely symmetric and all theorems about right ideals also hold, *mutatis mutandis*, for left ideals. In order to help with the distinction between two-sided ideals and right ideals, we denote the former by capital letters, for example, $I, J, \dots$, and the latter by capital letters with an additional subscript ρ, such as, $I_\rho, J_\rho, \dots$.

Any subset J of a (right) ideal I that is itself a (right) ideal, is called a *subideal* of I.

For a set $F \subseteq S$, we denote by (F) and $(F)_\rho$ the two-sided ideal, respectively the right ideal, generated by F, that is

$$(F) \coloneqq \left\{ \sum_{i=1}^{d} r_i f_i s_i \;\middle|\; d \in \mathbb{N}, r_i, s_i \in S, f_i \in F \right\},$$
$$(F)_\rho \coloneqq \left\{ \sum_{i=1}^{d} f_i s_i \;\middle|\; d \in \mathbb{N}, s_i \in S, f_i \in F \right\}.$$

A set $F \subseteq S$ is a *generating set* of an ideal $I \trianglelefteq S$ if $I = (F)$, and a *right generating set* of I if $I = (F)_\rho$. Analogously, F is a *generating set* of a right ideal $I_\rho \trianglelefteq_r S$ if $I_\rho = (F)_\rho$. We agree upon the convention to write $(f_1, \ldots, f_m)$ and $(f_1, \ldots, f_m)_\rho$ for $(\{f_1, \ldots, f_m\})$ and $(\{f_1, \ldots, f_m\})_\rho$ respectively, if the elements of $F = \{f_1, \ldots, f_m\}$ are given explicitly.

A (two-sided or right) ideal I of S is *finitely generated* if it has a finite generating set.

Recall that a ring S is *Noetherian* if it satisfies the ascending chain condition, meaning that every ascending sequence $I_0 \subseteq I_1 \subseteq \ldots$ of ideals $I_0, I_1, \ldots \trianglelefteq S$ eventually stabilises, or equivalently, that every ideal of S is finitely generated.

For example, every field is Noetherian. Furthermore, if R is a Noetherian ring, then, by Hilbert's basis theorem, also the univariate polynomial ring $R[x]$ is Noetherian, and thus, by induction, so is $R[x_1, \ldots, x_n]$ for every $n \in \mathbb{N}$. Unfortunately, this is not true for the free algebra. If $|X| > 1$, then the free algebra $R\langle X \rangle$ is not Noetherian. Consequently, there exist ideals in $R\langle X \rangle$ which are not finitely generated. A classical example is the following.

Example 2.2.27. *Let R be a commutative ring (or even a field). The ideal*

$$I = (xy^n x \mid n \in \mathbb{N}) \subseteq R\langle x, y \rangle$$

has no finite generating set, and thus, is not finitely generated. The proof of this fact relies on the fact that I is a monomial ideal and generated by a so-called minimal infinite generating set. For further details, we refer to [Xiu12, Rem. 2.1.25].

In the following, we restrict ourselves to finitely generated ideals. A central problem when working with ideals is the following *ideal membership problem.*

Problem 2.2.28 (Ideal membership)**.**
Input: $r, r_1, \ldots, r_m \in S$
Output: *True if* $r \in (r_1, \ldots, r_m)$ *and* *False otherwise*

In many settings, most notably in the case of commutative polynomials, this problem is decidable. For the free algebra, however, the ideal membership problem is undecidable, as the famous word problem for semigroups, which is known to be undecidable [Dav58, Thm. 4.5], can be reduced to ideal membership in $R\langle X\rangle$, see, for example, [Xiu12, Rem. 2.2.12]. More precisely, we will see in Section 2.4.3 that the theory of Gröbner bases allows to *verify* ideal membership in $R\langle X\rangle$, that is, to give an affirmative answer to Problem 2.2.28, thus rendering the problem in fact semi-decidable. However, *disproving* ideal membership in the free algebra is not always possible.

One way to certify the ideal membership of an element $f \in R\langle X\rangle$ in the ideal generated by a set $F \subseteq R\langle X\rangle$ is to provide a representation of f in terms of the generators F. Such a representation is called a *cofactor representation*.

Definition 2.2.29. *Let $F \subseteq R\langle X\rangle$. A representation of an element $f \in (F)$ of the form*

$$f = \sum_{i=1}^{d} a_i f_i b_i, \tag{2.1}$$

with $d \in \mathbb{N}$, $a_i, b_i \in R\langle X\rangle$, and $f_i \in F$ is called a cofactor representation *of f with respect to F. We refer to the elements a_i and b_i as the* cofactors *of f with respect to the representation* (2.1).

Note that, in general, the cofactors a_i and b_i from several summands with the same f_i cannot be collected on both sides of f_i simultaneously. However, collecting cofactors only on the right-hand side of summands with the same f_i is possible if the left-hand side cofactors are all equal. For example, we can perform the simplification

$$\sum_{j=1}^{k} a_i f_i b_j = a_i f_i \left(\sum_{j=1}^{k} b_j \right).$$

Similarly, we can collect the cofactors on the left-hand side if the cofactors on the right-hand side of the summands are equal.

Remark 2.2.30. *Given a cofactor representation of the form* (2.1)*, we can expand the cofactors* a_i *and* b_i *into monomials in* $\langle X \rangle$*. Hence, we can write every* $f \in (F)$ *as*

$$f = \sum_{i=1}^{d'} c_i v_i f_i w_i,$$

with $c_i \in R$*,* $v_i, w_i \in \langle X \rangle$ *and* $f_i \in F$*.*

Example 2.2.31. *Consider the ideal* $(f_1, f_2) \trianglelefteq \mathbb{Z}\langle x, y \rangle$ *generated by* $f_1 = yx - xy$ *and* $f_2 = xy - x$ *and let* $f = xyyx - xx \in (f_1, f_2)$*. A cofactor representation of* f *is given by*

$$f = xyf_1 + xyf_2 + f_2x.$$

Note that the two summands xyf_2 *and* f_2x *cannot be merged together. This example also shows that a cofactor representation is not necessarily unique, since we also have*

$$f = f_2yx + f_2x. \tag{2.2}$$

Note that, in (2.2), we can collect the cofactors on the right-hand side of the summands to obtain $f = f_2(yx + x)$*.*

We also recall the concept of a quotient ring. As before, S denotes a (not necessarily commutative) ring.

Definition 2.2.32. *The* quotient ring *of* S *by an ideal* $I \trianglelefteq S$ *is the set*

$$S/I \coloneqq \{[r] \mid r \in S\}$$

together with the addition and multiplication

$$[r] + [s] = [r + s] \qquad [r][s] = [rs],$$

where $[r] = r + I = \{r + i \mid i \in I\}$ *denotes the equivalence class of* r *modulo* I*.*

For inline text, we will also write S/I instead of S/I.

It is straightforward to check that the operations in Definition 2.2.32 are well-defined and that S/I indeed forms a ring.

To end this section, we recall the concept of a graded ring. We refer to [Row06, Ch. 7] and [Coh03, Ch. 6.1] for further information.

Definition 2.2.33. *Let M be a monoid. A ring S is an* M-graded ring *if there exists a decomposition*

$$S = \bigoplus_{m \in M} S_m$$

of S into a direct sum of abelian subgroups S_m such that $S_m S_n \subseteq S_{mn}$ for all $m, n \in M$.

Given an M-graded ring S, an element $s \in S$ is called *homogeneous* if $s \in S_m$ for some $m \in M$. If $s \neq 0$, then m is called the *degree* of s, denoted by $\deg(s) = m$. By definition, zero is a homogeneous element and contained in every S_m. We leave its degree undefined. Every nonzero $s \in S$ has a unique decomposition as

$$s = s_{m_1} + \cdots + s_{m_d}$$

with $m_i \in M$ and $0 \neq s_{m_i} \in S_{m_i}$ for $i = 1, \ldots, d$. The elements s_{m_i} are referred to as the *homogeneous components* of s.

We list some classical examples of gradings of the free algebra $R\langle X \rangle$.

Example 2.2.34. *For each $w \in \langle X \rangle$, the set $Rw = \{rw \mid r \in R\} \subseteq R\langle X \rangle$ forms an abelian subgroup. Furthermore, we have*

$$R\langle X \rangle = \bigoplus_{w \in \langle X \rangle} Rw,$$

and since also $RwRw' = Rww'$ for all $w, w' \in \langle X \rangle$, we see that $R\langle X \rangle$ is a $\langle X \rangle$-graded ring. In this case, the homogeneous elements are all the terms of the form rw for $r \in R$ and $w \in \langle X \rangle$.

Example 2.2.35. *For $n \in \mathbb{N}$, let $R\langle X \rangle_n = \{f \in R\langle X \rangle \mid |w| = n \text{ for all } w \in \operatorname{supp}(f)\}$. These abelian subgroups define an $\mathbb{N}$-graded structure on $R\langle X \rangle$, called the* standard grading *of $R\langle X \rangle$. If we decompose a nonzero $f \in R\langle X \rangle$ as*

$$f = f_1 + \cdots + f_d$$

with $0 \neq f_i \in R\langle X\rangle_i$ *for* $i = 1, \ldots, d$*, then the number* $d \in \mathbb{N}$ *is called the* (standard) degree *of* f*, denoted by* $\deg(f)$*. Clearly, we have* $\deg(w) = |w|$ *for* $w \in \langle X\rangle$ *and* $\deg(c) = 0$ *for* $c \in R \setminus \{0\}$*.*

Example 2.2.36. *Let* $X = \{x_1, \ldots, x_n\}$*. A matrix* $A \in \mathbb{R}^{n\times m}$ *with rows* $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n \in \mathbb{R}^m$ *defines a grading of the free algebra* $R\langle X\rangle$ *by the monoid* $(\mathbb{R}^m, +)$*. The grading can be specified by the* (weighted) degree function $\deg_A \colon \langle X\rangle \to \mathbb{R}^m$ *associated to* A*, defined on* X *to be* $\deg_A(x_i) = \boldsymbol{a}_i$*, for* $i = 1, \ldots, n$*, and extended to be a monoid homomorphism. Such a degree function decomposes* $R\langle X\rangle$ *into a direct sum of the abelian subgroups*

$$R\langle X\rangle_{\boldsymbol{a}} = \{f \in R\langle X\rangle \mid \deg_A(w) = \boldsymbol{a} \text{ for all } w \in \operatorname{supp}(f)\},$$

for $\boldsymbol{a} \in \mathbb{R}^m$*. The standard grading of* $R\langle X\rangle$ *considered in Example 2.2.35 can be obtained as a special case by choosing* $A = (1, 1, \ldots, 1)^T \in \mathbb{R}^{n\times 1}$*.*

Definition 2.2.37. *An ideal* $I \trianglelefteq S$ *of an* M*-graded ring* S *is called* homogeneous *(or* M*-graded) if* $I = \bigoplus_{m\in M}(I \cap S_m)$*.*

The condition in Definition 2.2.37 can also be expressed by saying that the ideal I can be generated by homogeneous elements, or equivalently, that, with every nonzero element $s \in I$, also all homogeneous components of s are contained in I. An M-graded ideal I, induces an M-grading on the quotient ring S/I.

2.2.4 Free bimodule

In this section, we recall constructions and properties of *bimodules*. In particular, we discuss the direct product and sum as well as the tensor product of bimodules. These constructions allow us to describe the free bimodule on a set, which we use in subsequent chapters to express and study relations of elements in the free algebra. For further information and proofs of the claimed statements, we refer to [Row91, Ch. 1.7]. In the following, R, S, T are (not necessarily commutative) rings.

Definition 2.2.38. *An* (R,S)-bimodule *is an abelian group* $(M,+)$ *satisfying the following two conditions:*

1. *M is a left R-module and a right S-module;*
2. *we have $(r \cdot m) \cdot s = r \cdot (m \cdot s)$ for all $r \in R$, $s \in S$ and $m \in M$;*

An (R,R)-bimodule is simply called an R-bimodule.

Definition 2.2.39. *Let M be an (R,S)-bimodule. A subset $N \subseteq M$ is called an* (R,S)-subbimodule *of M if it is an additive subgroup of M and satisfies $r \cdot n \cdot s \in N$ for all $r \in R$, $s \in S$ and $n \in N$.*

For a subset $B \subseteq M$ of an (R,S)-bimodule M, the (R,S)-subbimodule of M generated by B is the set

$$\left\{ \sum_{i=1}^{d} r_i b_i s_i \;\middle|\; d \in \mathbb{N}, r_i \in R, s_i \in S, b_i \in B \right\}.$$

It is the smallest (R,S)-subbimodule of M containing B.

Definition 2.2.40. *Let M and N be (R,S)-bimodules. A function $\psi\colon M \to N$ is an* (R,S)-bimodule homomorphism *if it satisfies the following conditions:*

1. *$\psi(m+n) = \psi(m) + \psi(n)$ for all $m, n \in M$;*
2. *$\psi(rms) = r\psi(m)s$ for all $r \in R$, $s \in S$, $m \in M$;*

We recall some standard constructions of bimodules.

Definition 2.2.41. *Let I be an index set and $(M_i)_{i\in I}$ be a family of (R,S)-bimodules. The* direct product *of $(M_i)_{i\in I}$ is the Cartesian product, that is, the set*

$$\prod_{i\in I} M_i = \{(m_i)_{i\in I} \mid m_i \in M_i \text{ for all } i \in I\}$$

with component-wise addition and scalar multiplication:

$$(m_i)_{i\in I} + (m_i')_{i\in I} = (m_i + m_i')_{i\in I}, \qquad r(m_i)_{i\in I} = (rm_i)_{i\in I}, \qquad (m_i)_{i\in I} r = (m_i r)_{i\in I},$$

where $r \in R$ and $s \in S$.

It is straightforward to verify that the direct product $\prod_{i\in I} M_i$ is an (R,S)-bimodule. In particular, if $M_i = M$ for all $i \in I$, we obtain the *direct power* of M, denoted by M^I.

Definition 2.2.42. *Let I be an index set and $(M_i)_{i\in I}$ be a family of (R,S)-bimodules. The* (external) direct sum *of $(M_i)_{i\in I}$ is the (R,S)-subbimodule*

$$\bigoplus_{i\in I} M_i = \left\{ (m_i)_{i\in I} \in \prod_{i\in I} M_i \;\middle|\; m_i = 0 \text{ for almost all } i \in I \right\}$$

of $\prod_{i\in I} M_i$. An element $(m_i)_{i\in I} \in \bigoplus_{i\in I} M_i$ is denoted by the formal sum $\sum_{i\in I} m_i i$.

If $M_i = M$ for all $i \in I$, we obtain the direct sum of $|I|$ copies of M, denoted by $M^{(I)}$. Note that, for a finite index set I, the direct product $\prod_{i\in I} M_i$ and the direct sum $\bigoplus_{i\in I} M_i$ coincide, but for infinite I, the direct sum is a strict subset of the direct product.

Another way to construct (R,S)-bimodules is via the tensor product.

Definition 2.2.43. *Let M be an (S,R)-bimodule and let N be an (R,T)-bimodule. Furthermore, let G be the free $\mathbb{Z}$-module on the Cartesian product $M \times N$ and let H be the subgroup of G generated by all elements of the form*

$$\begin{gathered}(m+m',n)-(m,n)-(m',n)\\(m,n+n')-(m,n)-(m,n')\\(mr,n)-(m,rn)\end{gathered}$$

for $m,m' \in M$, $n,n' \in N$, and $r \in R$. The tensor product *of M and N over R is the abelian group*

$$M \otimes_R N \coloneqq {}^{G}\!/_{H},$$

The equivalence class of (m,n) in $M \otimes_R N$ is denoted by $m \otimes n$.

In other words, the tensor product $M \otimes_R N$ is the abelian group generated by the set of *pure tensors* $\{m \otimes n \mid m \in M, n \in N\}$ with relations

$$\begin{aligned}(m+m')\otimes n &= m\otimes n + m'\otimes n,\\ m\otimes(n+n') &= m\otimes n + m\otimes n',\\ mr\otimes n &= m\otimes rn.\end{aligned}$$

Every element in $M \otimes_R N$ can be written as a finite sum of pure tensors, that is

$$M \otimes_R N = \left\{ \sum_{i=1}^{d} m_i \otimes n_i \;\middle|\; d \in \mathbb{N}, m_i \in M, n_i \in N \right\}.$$

Furthermore, the tensor product $M \otimes_R N$ admits an (S,T)-bimodule structure with scalar multiplication

$$s \sum_{i=1}^{d} (m_i \otimes n_i) = \sum_{i=1}^{d} sm_i \otimes n_i \qquad \sum_{i=1}^{d} (m_i \otimes n_i)\, t = \sum_{i=1}^{d} m_i \otimes n_i t.$$

The tensor product is a very useful construction. It allows, for example, to consider R-bimodules as left modules over a larger ring. We recall that R^{op} denotes the *opposite ring* of R where multiplication is defined as $r \cdot s \coloneqq sr$ for all $r, s \in R$. Then we consider the tensor product

$$R \otimes_{\mathbb{Z}} R^{\mathrm{op}} = \left\{ \sum_{i=1}^{d} r_i \otimes s_i \;\middle|\; d \in \mathbb{N}, r_i, s_i \in R \right\},$$

which becomes a ring with the multiplication defined on pure tensors to be

$$(r \otimes s)(r' \otimes s') = (rr' \otimes s's)$$

and extended linearly. The ring $R \otimes_{\mathbb{Z}} R^{\mathrm{op}}$ is called the *enveloping ring* of R. With this, we arrive at the following result, see also [Row91, Prop. 1.7.31].

Proposition 2.2.44. *Every R-bimodule can be considered as an $R \otimes_{\mathbb{Z}} R^{op}$-module with scalar multiplication*

$$\left(\sum_{i=1}^{d} r_i \otimes s_i \right) m = \sum_{i=1}^{d} r_i m s_i.$$

Conversely, every $R \otimes_{\mathbb{Z}} R^{op}$-module can be considered as an R-bimodule with scalar multiplication

$$rm = (r \otimes 1)m \qquad mr = (1 \otimes r)m.$$

Combining the tensor product with direct sums, we obtain the main bimodule construction relevant for this work. To this end, recall that the *center* $Z(R)$ of a ring R is the commutative subring

$$Z(R) = \{r \in R \mid rs = sr \text{ for all } s \in R\}.$$

Definition 2.2.45. *Let $S \subseteq Z(R)$ be a commutative subring of R and $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_n\}$ be a finite set. The* free R-bimodule on the set $\mathcal{E}$ centralising S *is the R-bimodule*

$$\Sigma = (R \otimes_S R)^{(\mathcal{E})}.$$

By Proposition 2.2.44, the free R-bimodule Σ can be considered as a $R \otimes_{\mathbb{Z}} R^{\mathrm{op}}$-module. In this setting, Σ is free with basis $\mathcal{E}$, and can, in this sense, be regarded as a *free* R-bimodule, see also [Coh85, Sec. 0.11].

Remark 2.2.46. *More generally, instead of a commutative subring S, an arbitrary (possibly noncommutative) subring of R can be used in the construction above. Moreover, the finite set $\mathcal{E}$ can be replaced by an arbitrary (possibly infinite) set X. We refer to [Coh85, Sec. 0.11] for further information.*

Elements in Σ are of the form

$$\sum_{i=1}^{n} \left(\sum_{j=1}^{d_i} a_{i,j} \otimes b_{i,j} \right) \varepsilon_i,$$

with $a_{i,j}, b_{i,j} \in R$. For easier readability, we write them as

$$\sum_{i=1}^{n} \sum_{j=1}^{d_i} a_{i,j} \varepsilon_i b_{i,j}.$$

Note that, for all $s \in S$,

$$s \left(\sum_{i=1}^{n} \sum_{j=1}^{d_i} a_{i,j} \varepsilon_i b_{i,j} \right) = \sum_{i=1}^{n} \sum_{j=1}^{d_i} a_{i,j} s \varepsilon_i b_{i,j} = \sum_{i=1}^{n} \sum_{j=1}^{d_i} a_{i,j} \varepsilon_i s b_{i,j} = \left(\sum_{i=1}^{n} \sum_{j=1}^{d_i} a_{i,j} \varepsilon_i b_{i,j} \right) s.$$

The first and last identities can be derived from the fact that $s \in S \subseteq Z(R)$ is central, and the middle identity follows from the fact that the tensor product in Definition 2.2.45 is taken over S, which motivates the use of the term "centralising" in the definition.

Remark 2.2.47. *The free R-bimodule centralising a commutative subring S will be required in Chapter 3 for introducing noncommutative signature Gröbner bases. In particular, for dealing with the so-called* mixed algebra *(see Sections 3.1.1 and 3.1.2), which contains a mixture of commutative and noncommutative indeterminates.*

2.3 Basics of abstract rewriting

Abstract rewriting describes the concept of traversing some directed graph, or more concretely, of manipulating some object (for example, a term or formula) in a stepwise manner, often via the repeated application of certain simplification rules. Mathematically, this process can be described by a binary relation $\to$ on a set A.

Definition 2.3.1. *Let A be a set and $\to \subseteq A \times A$ be a binary relation on A. The pair $(A, \to)$ is called an* abstract reduction system*, and $\to$ is a* reduction relation *or simply a* reduction*. We write $x \to y$ for $(x, y) \in \to$.*

Remark 2.3.2. *The term* reduction *is coined by the fact that in many applications some quantity decreases with each reduction step.*

The repeated application of reductions corresponds to a composition of relations. Recall that the composition of two relations $R \subseteq A \times B$ and $S \subseteq B \times C$ is the relation

$$R \circ S \coloneqq \{(x, z) \in A \times C \mid \exists y \in B : (x, y) \in R \wedge (y, z) \in S\}.$$

Based on this, we introduce some basic notions of the composition of a reduction with itself. To this end, we fix an abstract reduction system $(A, \to)$ for the rest of this section.

Definition 2.3.3. *We define the following notions:*

$\xrightarrow{0}$	$:= \{(x,x) \mid x \in A\}$	identity
$\xrightarrow{n+1}$	$:= \xrightarrow{n} \circ \rightarrow$	$(n+1)$-fold composition, $n \in \mathbb{N}$
$\xrightarrow{*}$	$:= \bigcup_{n\in\mathbb{N}} \xrightarrow{n}$	reflexive transitive closure
$\leftarrow$	$:= \{(y,x) \mid x \rightarrow y\}$	inverse
$\leftrightarrow$	$:= \rightarrow \cup \leftarrow$	symmetric closure
$\overset{*}{\leftrightarrow}$	$:= (\leftrightarrow)^*$	reflexive transitive symmetric closure

Remark 2.3.4. *Notations like $\leftarrow$ or $\leftrightarrow$ only make sense for arrow-like symbols. In case of an arbitrary relation $R \subseteq A \times A$ one could write R^{-1} for the inverse relation, for example.*

One can show that $\overset{*}{\leftrightarrow}$ is the least equivalence relation containing $\rightarrow$. More generally, the P *closure* of a binary relation $\rightarrow \subseteq A \times A$ is the least set with predicate P containing $\rightarrow$. While, for an arbitrary predicate P, the P closure of $\rightarrow$ need not always exist, for the cases defined above it does. The reason is that reflexivity, transitivity and symmetry are closed under arbitrary intersections, and in such cases, the P closure is simply the intersection of all sets with predicate P containing $\rightarrow$.

The following definitions extend the notation from above.

Definition 2.3.5. *Let $x, y, z \in A$.*

1. *x is* reducible *if there exists y such that $x \rightarrow y$;*
2. *x is* irreducible *or* in normal form *if it is not reducible;*
3. *y is* a normal form *of x if $x \xrightarrow{*} y$ and y is in normal form; if x has a unique normal form, the latter is denoted by $x{\downarrow}$;*
4. *x and y are* joinable*, denoted by $x \downarrow y$, if there exists z such that $x \xrightarrow{*} z \xleftarrow{*} y$;*

To illustrate these notions we consider and slightly extend [BN98, Ex. 2.1.2].

Example 2.3.6. *For $A = \mathbb{N}_{>0} \setminus \{1\}$ and $\to = \{(m,n) \mid m > n \ \wedge \ n \mid m\}$, the following hold:*

1. *m is in normal form if and only if m is prime.*
2. *p is a normal form of m if and only if p is a prime factor of m. Consequently, all elements of A have a normal form, which however is not always unique. In fact, the only elements that have a unique normal form are prime powers.*
3. *$m \downarrow n$ if and only if m and n are not relatively prime.*
4. *For the inverse relation $\leftarrow = \{(n,m) \mid \exists k \in \mathbb{N}_{>0} \setminus \{1\} : m = kn\}$, no element has a normal form.*
5. *$\overset{*}{\leftrightarrow} = A \times A$.*

Example 2.3.7. *For $A = \langle a, b \rangle$ and $\to = \{(ubav, uabv) \mid u, v \in A\}$, the following hold:*

1. *w is in normal form if and only if w is sorted, that is, w is of the form $a^m b^n$ for some $m, n \in \mathbb{N}$.*
2. *Every w has a unique normal form $w{\downarrow}$, obtained by sorting w.*
3. *$w \downarrow w'$ if and only if $w \overset{*}{\leftrightarrow} w'$ if and only if w and w' contain the same number of as and bs.*

One important application of reduction systems, and the one that we are interested in, is deciding the equivalence of two elements x and y of A with respect to the equivalence induced by $\to$, that is, to solve the following *word problem*.

Problem 2.3.8 (Word problem)**.**
Input: $(A, \to)$ abstract reduction system, $x, y \in A$
Output: True if $x \overset{*}{\leftrightarrow} y$ and False otherwise

One approach to solve this problem is to reduce x and y to normal form x' and y' respectively, and then check x' and y' for syntactic equality. However, as easy as this method sounds in theory, in practice two problems can arise. First of all, there might exist elements in A that do not admit a normal form, as seen, for example, for the inverse relation in

Example 2.3.6. Moreover, normal forms need not be unique; as an example, consider the relation from Example 2.3.6. These problems render the word problem undecidable in general. In order to discuss existence and uniqueness of normal forms, we recall the following central properties of reduction relations.

Definition 2.3.9. *A reduction $\to$ is*

1. Church-Rosser *if $x \overset{*}{\leftrightarrow} y$ implies $x \downarrow y$;*
2. confluent *if $y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2$ implies $y_1 \downarrow y_2$;*
3. normalising *if every element has a normal form;*
4. terminating *if there is no infinite descending chain $a_0 \to a_1 \to \ldots$;*

Example 2.3.10. *The reductions in Example 2.3.6 and 2.3.7 are both terminating but only the second one is Church-Rosser and confluent. The inverse relation in Example 2.3.6 is not even terminating.*

The following result relates the Church-Rosser property to confluence.

Theorem 2.3.11. *A reduction $\to$ is Church-Rosser if and only if it is confluent.*

Proof. If $\to$ is Church-Rosser and $y_1 \overset{*}{\leftarrow} x \overset{*}{\rightarrow} y_2$, then $y_1 \overset{*}{\leftrightarrow} y_2$, and hence, $y_1 \downarrow y_2$ by the Church-Rosser property. So $\to$ is confluent.

Conversely, if $\to$ is confluent and $x \overset{*}{\leftrightarrow} y$, then we show $x \downarrow y$ by induction on the length of the chain $x \overset{*}{\leftrightarrow} y$. If $x = y$, then trivially $x \downarrow y$. If $x \overset{*}{\leftrightarrow} y' \leftrightarrow y$, then the induction hypothesis implies $x \downarrow y'$, that is, $x \overset{*}{\rightarrow} z \overset{*}{\leftarrow} y'$ for some z. Now, we distinguish between two cases:

- $y' \leftarrow y$: in this case, $x \downarrow y$ follows immediately from $x \downarrow y'$;
- $y' \to y$: in this case, $z \overset{*}{\leftarrow} y' \to y$, and so confluence implies $z \downarrow y$, which in turn yields $x \downarrow y$; □

The previous theorem has the following consequence.

Corollary 2.3.12. *If $\to$ is confluent and $x \overset{*}{\leftrightarrow} y$, then*

1. *$x \overset{*}{\to} y$ if y is in normal form;*
2. *$x = y$ if both x and y are in normal form;*

Proof. By Theorem 2.3.11, $\to$ is Church-Rosser and therefore $x \overset{*}{\leftrightarrow} y$ implies $x \downarrow y$, that is, $x \overset{*}{\to} z \overset{*}{\leftarrow} y$ for some z. If y is normal form, then $z = y$, and so $x \overset{*}{\to} y$. If x is also in normal form, then also $z = x$, and $x = y$ follows. □

We see that the confluence of a reduction implies that every element has at most one normal form. Thus, if two elements can be reduced to normal form, then their equivalence can be decided by syntactically comparing the normal forms. However, normal forms need not exist for all elements as we can end up with infinite reduction sequences.

To ensure the existence of normal forms, a reduction has to be normalising. A normalising and confluent reduction has unique normal forms and allows to solve the word problem by computing these normal forms. This is the result of the following theorem.

Theorem 2.3.13. *If $\to$ is normalising and confluent, then every element has a unique normal form. Furthermore, $x \overset{*}{\leftrightarrow} y$ if and only if $x{\downarrow} = y{\downarrow}$.*

Proof. The uniqueness of the normal forms follows from the definition of normalising and from Corollary 2.3.12. For the second assertion, the "if"-direction is trivial. For the "only if"-direction, note that $x \overset{*}{\leftrightarrow} y$ implies $x{\downarrow} \overset{*}{\leftrightarrow} y{\downarrow}$, and Corollary 2.3.12 yields $x{\downarrow} = y{\downarrow}$. □

Thus, for normalising and confluent reductions, we obtain a systematic way to solve the word problem: simply reduce both elements to normal form and check if the normal forms are equal. This renders Problem 2.3.8 decidable for this kind of reductions, provided that normal forms are computable and that identity is decidable.

To end this section, we note that proving normalisation for a concrete reduction is often a nontrivial task. However, it is easy to see that termination implies normalisation, and for many practical applications the former is easier to verify. Furthermore, while normalisation is enough for finding normal forms, it often requires a (typically more costly) breadth-first search rather than a depth-first search which is sufficient for terminating

reductions. Therefore, in practice, we seek terminating and confluent reductions rather than normalising and confluent ones.

2.4 Gröbner bases in the free algebra

Gröbner bases have become a fundamental and multipurpose tool in computational algebra. They are best known in algebras of commutative polynomials over fields [Buc65], where they can answer questions ranging from solving systems of polynomial equations, over performing ideal arithmetic operations to examining algebraic varieties. In the last decades, Gröbner bases have also been extended to various other settings, including noncommutative polynomials in the free algebra over fields [Bok76; Ber78; Mor85] or over rings [Pri96; MZ98; Mor15; LMA23]. For recent surveys on the theory of noncommutative Gröbner bases, we refer to [Nor01; Xiu12; BGV13; Mor16], and for an overview on available software packages to compute them, see [LSA20] and references therein.

In this section, we recall the main results of the theory of Gröbner bases for one- and two-sided ideals in the free algebra over a coefficient field. Our presentation is self-contained and based on the concept of polynomial reduction, which we discuss in Section 2.4.2. Before doing this, we first give an overview over several noncommutative monomial orders in Section 2.4.1 and discuss their properties. Here, we also present a novel characterisation of a large class of such orders by combining totally ordered semigroups with the lexicographic order as a tiebreaker. Our characterisation also makes use of a result by Higman [Hig52] and covers many classical orders, such as (weighted) (multi-)degree orders and elimination orders. Gröbner bases of two-sided ideals in the free algebra are defined in Section 2.4.3, where we also give several equivalent characterisations, including Bergman's famous diamond lemma [Ber78], which provides an algorithmic test to verify whether a given set of noncommutative polynomials forms a Gröbner basis. Based on this characterisation, we then provide a noncommutative version of Buchberger's algorithm for the free algebra. However, as the free algebra is not Noetherian, some ideals do not admit finite Gröbner bases, and consequently, we cannot expect Buchberger's algorithm to terminate for any input. Nevertheless, we show that the algorithm always correctly enumerates a (possibly infinite) Gröbner basis. Finally, in Section 2.4.4, we recall the main results of the theory of Gröbner bases for one-sided ideals. We restrict our presentation to right ideals and note that all results also apply symmetrically to left ideals.

2.4.1 Monomial orders

Recall that a total order $\preceq$ on a set A is a binary relation on A that is antisymmetric, transitive, and total. We follow the usual convention to write $a \preceq b$ instead of $(a, b) \in \preceq$. When working with monomials, it is often crucial that every collection of them possesses a least element. A total order satisfying this property is called a *well-order*. If we additionally require that the order respects the multiplication in $\langle X \rangle$, then we obtain a *monomial order*.

Definition 2.4.1. *A total order $\preceq$ on $\langle X \rangle$ is called a* monomial order *or an* admissible order *on $\langle X \rangle$ if it satisfies the following two conditions:*

1. *$w \preceq w'$ implies $awb \preceq aw'b$ for all $a, b, w, w' \in \langle X \rangle$* (compatibility with multiplication)*;*
2. *every nonempty subset of $\langle X \rangle$ has a least element* (well-order)*;*

The following result characterises well-orders via the non-existence of infinite strictly decreasing sequences of elements.

Lemma 2.4.2. *Let $\preceq$ be a total order on a set A. Every nonempty subset of A has a least element if and only if every decreasing sequence of elements in A eventually stabilises, that is, if*

$$a_0 \succeq a_1 \succeq \cdots$$

is an infinite decreasing sequence of elements $a_0, a_1, \ldots \in A$, then there exists $n \in \mathbb{N}$ such that $a_n = a_{n+k}$ for all $k \in \mathbb{N}$.

Proof. For the first implication, assume that every nonempty subset of A has a least element and let $a_0, a_1, \ldots \in A$ be an infinite decreasing sequence. By assumption, the set $B = \{a_i \mid i \in \mathbb{N}\}$ has a least element, say a_n for some $n \in \mathbb{N}$. By definition of a least element, $a_n \preceq a_i$ for all $i \in \mathbb{N}$, and so, in particular, $a_n \preceq a_{n+k}$ for all $k \in \mathbb{N}$. Since $a_0 \succeq a_1 \succeq \cdots$ is a decreasing sequence, we additionally have $a_n \succeq a_{n+k}$ for all $k \in \mathbb{N}$. Now the antisymmetry of $\preceq$ implies that $a_n = a_{n+k}$ for all $k \in \mathbb{N}$ and so the sequence stabilises.

For the second implication, assume that every decreasing sequence stabilises and assume, for contradiction, that there exists a nonempty subset of A without a least element. Let

$B \subseteq A$ be such a set. Then an infinite strictly deceasing sequence can be constructed as follows: since B is nonempty, choose $a_0 \in B$ arbitrary. Now, for $n \in \mathbb{N}$, choose $a_{n+1} \in B$ such that $a_n \succ a_{n+1}$. Note that such a choice is always possible because no a_n is a least element of B. This yields the infinite strictly decreasing sequence $a_0 \succ a_1 \succ \cdots$, contradicting the assumption. □

The compatibility of a monomial order with the multiplication in $\langle X \rangle$ together with the property of being a well-order implies that 1 must be the least element.

Proposition 2.4.3. *Let $\preceq$ be a monomial order on $\langle X \rangle$. Then $1 \preceq w$ for all $w \in \langle X \rangle$.*

Proof. Assume, for contradiction, that there exists $w \in \langle X \rangle$ such that $1 \succ w$. Then the first condition of Definition 2.4.1 yields $w^n = w^n \cdot 1 \succ w^n \cdot w = w^{n+1}$ for all $n \in \mathbb{N}$. This implies the existence of the infinite strictly decreasing sequence $1 \succ w \succ w^2 \succ \cdots$, which is a contradiction to Proposition 2.4.2. □

Conversely, it can be shown that a total order on $\langle X \rangle$ that is compatible with the multiplication in $\langle X \rangle$ and has 1 as its least element, must be a well-order, provided that X is finite.

Proposition 2.4.4. *Let X be a finite set and let $\preceq$ be a total order on $\langle X \rangle$ such that the following conditions hold:*

1. *$w \preceq w'$ implies $awb \preceq aw'b$ for all $a, b, w, w' \in \langle X \rangle$;*
2. *$1 \preceq w$ for all $w \in \langle X \rangle$;*

Then $\preceq$ is a well-order, and consequently, a monomial order.

This result can be seen a consequence of a theorem by Higman [Hig52]. Here, we give a direct and more elementary proof taken from [Con71], which is based on an idea in [Nas63].

Proof of Proposition 2.4.4. Assume, for contradiction, that there exist infinite decreasing sequences $w_0 \succ w_1 \succ \cdots$ of elements $w_0, w_1, \ldots \in \langle X \rangle$. We now construct the "smallest" such sequence. In particular, we choose $w_0 \in \langle X \rangle$ such that its length $|w_0|$ is minimal among all elements starting such an infinite sequence. Next, for $n \in \mathbb{N}$, we choose inductively $w_{n+1} \in \langle X \rangle$ so that its length $|w_{n+1}|$ is minimal among all elements in $\{w \in \langle X \rangle \mid w_0, \ldots, w_n, w \text{ starts an infinite strictly decreasing sequence}\}$.

Since X is finite, there exist infinitely many elements w_n in the previously constructed sequence that begin with the same variable, say $w_{n_0} = xv_0, w_{n_1} = xv_1, \ldots$ with $x \in X$, $v_i \in \langle X \rangle$ for all $i \in \mathbb{N}$ and $n_0 < n_1 < \cdots$. We record the following two important properties of this subsequence:

1. Since $1 \prec x$, we also have $v_0 = 1 \cdot v_0 \prec x \cdot v_0 = w_{n_0}$.
2. We must have $v_i \succ v_{i+1}$ for all $i \in \mathbb{N}$, as otherwise $w_{n_i} = x \cdot v_i \preceq x \cdot v_{i+1} = w_{n_{i+1}}$, which contradicts the fact that the w_n form a strictly decreasing sequence.

The two observations above imply that $w_0 \succ \cdots \succ w_{n_0-1} \succ v_0 \succ v_1 \succ \cdots$ is an infinite strictly decreasing sequence. More precisely, this sequence is "smaller" than the one we constructed previously because $|v_0| < |w_{n_0}|$. This is a contradiction. □

The first concrete order on $\langle X \rangle$ that we consider is the *lexicographic order* $\preceq_{\text{lex}}$, which is probably best known from how the words in a dictionary are ordered. This is why it is sometimes also referred to as the *dictionary order*. It is adapted to the free monoid $\langle X \rangle$ as follows. From now on, we assume that $X = \{x_1, \ldots, x_n\}$ is finite.

Definition 2.4.5. *Let $X = \{x_1, x_2, \ldots, x_n\}$. Order the elements of X as*

$$x_1 \prec_{\text{lex}} x_2 \prec_{\text{lex}} \cdots \prec_{\text{lex}} x_n.$$

Then $w \preceq_{\text{lex}} w'$ for $w, w' \in \langle X \rangle$ if one of the following conditions holds:

1. *there exist $l, r, r' \in \langle X \rangle$ and $x_i, x_j \in X$ such that $w = lx_ir$, $w' = lx_jr'$ and $x_i \prec_{\text{lex}} x_j$;*
2. *there exists $r \in \langle X \rangle$ such that $w' = wr$;*

Example 2.4.6. *Let $X = \{x, y, z\}$ and order the indeterminates as $x \prec_{\text{lex}} y \prec_{\text{lex}} z$. Then $xyx \prec_{\text{lex}} xyz$ by the first condition of Definition 2.4.5 and $xy \prec_{\text{lex}} xyx$ by the second condition of Definition 2.4.5.*

Remark 2.4.7. *By reordering the elements in $X = \{x_1, \ldots, x_n\}$, one can obtain different orders. Formally, this means that, for a permutation π of $\{1, \ldots, n\}$, we consider $\preceq_{\text{lex},\pi}$ where the indeterminates are ordered as $x_{\pi(1)} \prec_{\text{lex},\pi} x_{\pi(2)} \prec_{\text{lex},\pi} \cdots \prec_{\text{lex},\pi} x_{\pi(n)}$ and where $w \preceq_{lex,\pi} w'$ for $w, w' \in \langle X \rangle$ if one of the following conditions holds:*

1. *there exist $l, r, r' \in \langle X \rangle, x_i, x_j \in X$ such that $w = lx_ir$, $w' = lx_jr'$ and $x_i \prec_{\text{lex},\pi} x_j$;*
2. *there exists $r \in \langle X \rangle$ such that $w' = wr$;*

The lexicographic order from Definition 2.4.5 can be obtained as a special case by setting $\pi = \text{id}$. To keep the notation uncluttered, we only use the "classical" lexicographic order $\preceq_{\text{lex}}$ for all future definitions and examples in this thesis.

The lexicographic order is a total order, but it is not a monomial order since it is neither a well-order nor compatible with the multiplication in $\langle X \rangle$. This can easily be seen by taking $X = \{x, y\}$ and letting $x \prec_{\text{lex}} y$. Then $y \succ_{\text{lex}} xy \succ_{\text{lex}} xxy \succ_{\text{lex}} \cdots$ is an infinite strictly decreasing sequence in $\langle X \rangle$. Furthermore, we have $x \prec_{\text{lex}} xx$ but $xy \succ_{\text{lex}} xxy$.

Remark 2.4.8. *In case of commutative monomials, the lexicographic order is a monomial order. This order can be obtained from Definition 2.4.5 by writing commutative monomials $x_1^{a_1} \ldots x_n^{a_n}$ in the form*

$$\underbrace{x_n \ldots x_n}_{a_n \text{ times}} \underbrace{x_{n-1} \ldots x_{n-1}}_{a_{n-1} \text{ times}} \cdots \underbrace{x_1 \ldots x_1}_{a_1 \text{ times}}.$$

Although, as seen above, the lexicographic order is not multiplicative in general, there are elements in $\langle X \rangle$ for which it is. This fact is captured by the following lemma.

Lemma 2.4.9. *Let $w, w' \in \langle X \rangle$ be such that neither is a prefix of the other. Then $w \prec_{\text{lex}} w'$ implies $awb \prec_{\text{lex}} aw'b$ for all $a, b \in \langle X \rangle$.*

Proof. The second condition of Definition 2.4.5 would imply that one of w and w' is a prefix of the other. Since this is not the case here, there must exist $l, r, r' \in \langle X \rangle$ and $x_i, x_j \in X$ such that $w = lx_ir, w' = lx_jr'$ and $x_i \prec_{\text{lex}} x_j$. Consequently, we can write $awb = alx_irb$ and $aw'b = alx_jr'b$. These two words share the same prefix al and at index $|al| + 1$ it still holds that $x_i \prec_{\text{lex}} x_j$. Thus, the first condition of Definition 2.4.5 implies $awb \prec_{\text{lex}} aw'b$. □

In other words, as long as we can rely only on the first condition of Definition 2.4.5, the lexicographic order is compatible with the multiplication in $\langle X \rangle$. Based on this observation, we present the following construction, which uses the lexicographic order as a building block for monomial orders on $\langle X \rangle$. Recall that $X = \{x_1, \ldots, x_n\}$ is assumed to be finite.

Definition 2.4.10. *Let S be a semigroup equipped with a total order $\leq$ that is compatible with the multiplication in S and let $\varphi\colon \langle X \rangle \to S$ be a semigroup homomorphism such that $\varphi(1) < \varphi(w)$ for all $w \in \langle X \rangle \setminus \{1\}$. The* φ-lexicographic order $\preceq_\varphi$ *on $\langle X \rangle$ is defined by $w \preceq_\varphi w'$ for $w, w' \in \langle X \rangle$ if one of the following conditions holds:*

1. *$\varphi(w) < \varphi(w')$;*
2. *$\varphi(w) = \varphi(w')$ and $w \preceq_{\text{lex}} w'$;*

Remark 2.4.11. *Strictly speaking, the φ-lexicographic order $\preceq_\varphi$ does not only depend on φ but also on S and $\leq$. Since S and $\leq$ are usually clear from the context, we omit these dependencies in the notation.*

In contrast to the lexicographic order, the φ-lexicographic order is a monomial order. To show this, the following key observation is needed.

Lemma 2.4.12. *Let $S, \leq$, and φ be as in Definition 2.4.10 and let $w, w' \in \langle X \rangle$. If $\varphi(w) = \varphi(w')$, then neither w nor w' is a proper prefix of the other.*

Proof. Assume, for contradiction, that one is a proper prefix of the other, say, w is a proper prefix of w'. Then there exists $r \in \langle X \rangle \setminus \{1\}$ such that $w' = wr$. By assumption on φ, we have $\varphi(1) < \varphi(r)$. This now leads to the following contradiction to the assumption $\varphi(w) = \varphi(w')$:

$$\varphi(w) = \varphi(w \cdot 1) = \varphi(w)\varphi(1) < \varphi(w)\varphi(r) = \varphi(wr) = \varphi(w').$$ □

Using this lemma, we can now prove the following result.

Theorem 2.4.13. *The φ-lexicographic order $\preceq_\varphi$ is a monomial order.*

Proof. The φ-lexicographic order is a total order, since the order $\leq$ on S and $\preceq_{\text{lex}}$ are both total orders. To show that $\preceq_\varphi$ is compatible with the multiplication in $\langle X \rangle$, let $a, b, w, w' \in \langle X \rangle$ such that $w \preceq_\varphi w'$. If $w = w'$, then clearly also $awb = aw'b$. Hence, we can assume that $w \neq w'$, and consequently $w \prec_\varphi w'$. Then either $\varphi(w) < \varphi(w')$ or $\varphi(w) = \varphi(w')$ and $w \prec_{\text{lex}} w'$. In the first case, we can use the fact that φ is a homomorphism and that $\leq$ is compatible with the multiplication in S to obtain

$$\varphi(awb) = \varphi(a)\varphi(w)\varphi(b) < \varphi(a)\varphi(w')\varphi(b) = \varphi(aw'b),$$

and thus $awb \prec_\varphi aw'b$. For the second case, Lemma 2.4.12 implies that neither w nor w' is a proper prefix of the other. Since also $w \neq w'$, Lemma 2.4.9 yields the desired result. To see that $\preceq_\varphi$ is also a well-order, note that $1 \preceq_\varphi w$ for all $w \in \langle X \rangle$ as $\varphi(1) < \varphi(w)$ for all $w \in \langle X \rangle \setminus \{1\}$ by assumption. With this, the result follows from Proposition 2.4.4. □

In the following, we present several well-known monomial orders as special instances of φ-lexicographic orders.

Example 2.4.14. *Recall that $X = \{x_1, \ldots, x_n\}$.*

1. *For $S = (\mathbb{N}, +)$ with the usual order and $\varphi(w) = |w|$, we obtain the* degree lexicographic order $\preceq_{\text{deglex}}$ *(also called* graded lexicographic order*).*
2. *Let $\boldsymbol{a} \in \mathbb{R}^n_{>0}$ and let $\deg_{\boldsymbol{a}} \colon \langle X \rangle \to \mathbb{R}$ be the weighted degree function (see Example 2.2.36) associated to $\boldsymbol{a}$ when $\boldsymbol{a}$ is considered as a matrix consisting of one column. For $S = (\mathbb{R}_{\geq 0}, +)$ with the usual order and $\varphi(w) = \deg_{\boldsymbol{a}}(w)$, we obtain the* weighted degree lexicographic order $\preceq_{\boldsymbol{a}}$ *(also called* weighted graded lexicographic order*).*
3. *Let $A \in \mathbb{R}^{n \times m}_{\geq 0}$ be a matrix without zero rows and let $\deg_A \colon \langle X \rangle \to \mathbb{R}^m$ be the associated weighted degree function. For $S = (\mathbb{R}^m_{\geq 0}, +)$ with the usual lexicographic order and $\varphi(w) = \deg_A(w)$, we obtain the* weighted multidegree lexicographic order $\preceq_A$*.*

4. *Let* $[X] = \{x_1^{a_1} \dots x_n^{a_n} \mid (a_1, \dots, a_n) \in \mathbb{N}^n\}$ *be the set of commutative monomials in the indeterminates* X *and let* $\cdot$ *be the multiplication of commutative monomials. We can set* $S = ([X], \cdot)$ *equipped with any commutative monomial order and use* $\varphi\colon \langle X \rangle \to [X]$ *as the canonical projection from noncommutative to commutative monomials.*

It is easy to see that in all example above φ is a semigroup homomorphism satisfying $\varphi(1) < \varphi(w)$ for all $w \in \langle X \rangle \setminus \{1\}$. Consequently, Theorem 2.4.13 ensures that the respective orders are indeed all monomial orders.

We make some remarks on the orders presented above:

1. The weighted degree lexicographic order allows to value the appearance of certain indeterminates in a monomial more than the appearance of others. The usual degree lexicographic ordering can be obtained as a special case by setting $\boldsymbol{a} = (1, \dots, 1)$.

2. Note that the vector $\boldsymbol{a}$ used in the weighted degree lexicographic order must consist of strictly positive numbers. It is not sufficient to demand nonnegative numbers as the following example shows. Let $X = \{x, y\}$ and order the indeterminates as $x \prec_{\text{lex}} y$. Furthermore, let $\boldsymbol{a} = (0, 1)$. Then $\deg_{\boldsymbol{a}}(x^n y) = 1$ for all $n \in \mathbb{N}$, and thus, $x^{n+1} y \prec_{\boldsymbol{a}} x^n y$ for all $n \in \mathbb{N}$, since $x \prec_{\text{lex}} y$. Hence, we end up with the infinite strictly decreasing sequence $y \succ_{\boldsymbol{a}} xy \succ_{\boldsymbol{a}} xxy \succ_{\boldsymbol{a}} \cdots$ of monomials in $\langle X \rangle$, showing that $\preceq_{\boldsymbol{a}}$ is not a well-order in this case, and consequently, also not a monomial order.

   For the same reason, the matrix A used to construct a weighted multidegree lexicographic order must not contain zero rows.

3. The lexicographic order on $\mathbb{R}^m_{\geq 0}$ is given by $\boldsymbol{a} \leq \boldsymbol{b}$, for $\boldsymbol{a} = (a_1, \dots, a_m)$ and $\boldsymbol{b} = (b_1, \dots, b_m)$, if $a_i = b_i$ for all $i = 1, \dots, m$ or if $a_i < b_i$ for the smallest index $1 \leq i \leq m$ where a_i and b_i differ.

4. All orderings presented above can be considered as special instances of the last case.

Choosing certain block structures for the matrix A when constructing a weighted multidegree lexicographic order, yields so-called elimination orders.

Definition 2.4.15. *Let X and Y be disjoint sets of indeterminates. A monomial order $\preceq$ on $\langle X, Y\rangle$ is an* elimination order *for Y if*

$$(w \prec w' \text{ and } w' \in \langle X\rangle) \implies w \in \langle X\rangle,$$

for all $w, w' \in \langle X, Y\rangle$.

Example 2.4.16. *Let $X = \{x_1, \dots, x_n\}$. Using a weighted multidegree lexicographic order $\preceq_{I_n}$ on $\langle X\rangle$ with the $n \times n$ identity matrix I_n yields an elimination order for $\{x_1, \dots, x_k\}$ for every $k = 1, \dots, n$.*

Elimination orders are very useful when simplifying expressions (and, in particular, when solving systems), as they allow to eliminate the larger indeterminates in Y from a system of polynomials. Using Gröbner bases, we will be able to express this property formally. We refer to Theorem 2.4.43 for the precise statement.

To end this section, we recall several classical definitions relevant for studying noncommutative Gröbner bases. Since many of them depend on a monomial order, we fix a monomial order $\preceq$ on $\langle X\rangle$ for the rest of this section. If a particular order is used in an example, we denote it by the corresponding subscript. In the following, R is a commutative ring.

Definition 2.4.17. *Let $f \in R\langle X\rangle \setminus \{0\}$. The* leading monomial $\operatorname{lm}(f)$ *of f is the $\preceq$-maximal element in* $\operatorname{supp}(f)$*, and the* leading coefficient $\operatorname{lc}(f)$ *of f is the coefficient of* $\operatorname{lm}(f)$*. The* leading term $\operatorname{lt}(f)$ *of f is* $\operatorname{lt}(f) = \operatorname{lc}(f) \cdot \operatorname{lm}(f)$*. Furthermore, we define* $\operatorname{lc}(0) = \operatorname{lt}(0) = 0$*. A polynomial with leading coefficient* 1 *is called* monic*.*

Note that, so far, the leading monomial of the zero polynomial is undefined. To avoid exceptions for the zero polynomial, it is convenient to extend a monomial order $\preceq$ to $\langle X\rangle \cup \{0\}$ and set $0 \prec w$ for all $w \in \langle X\rangle$. In the following, we consider such an extension, and with this in mind, we define $\operatorname{lm}(0) = 0$. Furthermore, by a slight abuse of notation, for sets $F \subseteq R\langle X\rangle$, we let $\operatorname{lm}(F) = \{\operatorname{lm}(f) \mid f \in F\} \subseteq \langle X\rangle \cup \{0\}$.

Remark 2.4.18. *To be precise, we should only speak of a leading monomial/coefficient/term with respect to a monomial order $\preceq$. In the following, we omit this additional information when it is clear from the context which monomial order is used.*

The following proposition, lists some basic facts about the notions introduced above. They follow essentially directly from the definition.

Lemma 2.4.19. *Let R be an integral domain and $f, g \in R\langle X\rangle$.*

1. $\mathrm{lm}(f+g) \preceq \max_{\preceq}\{\mathrm{lm}(f), \mathrm{lm}(g)\}$ *with equality if and only if* $\mathrm{lt}(f)+\mathrm{lt}(g) \neq 0$*;*
2. $\mathrm{lm}(fg) = \mathrm{lm}(f)\mathrm{lm}(g)$*,* $\mathrm{lc}(fg) = \mathrm{lc}(f)\mathrm{lc}(g)$*, and thus,* $\mathrm{lt}(fg) = \mathrm{lt}(f)\mathrm{lt}(g)$*;*

The "smaller" terms of a polynomial are all its terms except the leading term. They are called the *tail* of the polynomial.

Definition 2.4.20. *The* tail *of $f \in R\langle X\rangle$ is* $\mathrm{tail}(f) = f - \mathrm{lt}(f)$.

2.4.2 Polynomial reduction

One central notion for the theory of Gröbner bases in the free algebra is *(noncommutative) polynomial reduction*. For a simpler presentation, we restrict ourselves to coefficient fields in the following. A theory over coefficient rings can also be developed, see, for example, [Pri96; LMA23] for further information. In the following, K is a field and $X = \{x_1, \ldots, x_n\}$ is a finite set of indeterminates. Furthermore, we fix a monomial order $\preceq$ on $\langle X\rangle$. We note that a generalisation of polynomial reduction can also be introduced without the need for a monomial order, see [GHM19]. We define polynomial reduction as a reduction relation on $K\langle X\rangle$ in three steps.

Definition 2.4.21. *Let $a, b \in \langle X\rangle$, $g \in K\langle X\rangle \setminus \{0\}$, and $G \subseteq K\langle X\rangle$. We define the following reduction relations on $K\langle X\rangle$:*

$$
\begin{aligned}
f \to_{a,g,b} f' \quad &:\Longleftrightarrow \quad \mathrm{lm}(agb) \in \mathrm{supp}(f) \text{ and } f' = f - \frac{\mathrm{coeff}(f, \mathrm{lm}(agb))}{\mathrm{lc}(g)} agb \\
\to_g \quad &:= \bigcup_{a,b\in\langle X\rangle} \to_{a,g,b} \\
\to_G \quad &:= \bigcup_{g\in G\setminus\{0\}} \to_g
\end{aligned}
$$

These relations are called the polynomial reduction relation *with respect to a, g, b, with respect to g, and with respect to G respectively.*

Note that polynomial reduction cannot increase the leading monomial. More precisely, the following result follows immediately from the definition.

Lemma 2.4.22. *If $f \to_G f'$, then $\mathrm{lm}(f) \succeq \mathrm{lm}(f')$. Moreover, if $f \to_{a,g,b} f'$, then also $\mathrm{lm}(f) \succeq \mathrm{lm}(agb)$.*

As a consequence, we also see that, for every nonzero $g \in K\langle X\rangle$, the polynomial $\mathrm{tail}(g)$ is always irreducible with respect to $\to_g$.

A reduction $f \to_{a,g,b} f'$ is a *top reduction* if $\mathrm{lm}(f) \succ \mathrm{lm}(f')$ and a *tail reduction* otherwise. Top and tail reductions are defined analogously for $\to_g$ and $\to_G$.

The reduction $\to_{a,g,b}$ is obviously terminating, because if f is reducible, say $f \to_{a,g,b} f'$, then f' is in normal form. Furthermore, every element has a unique normal form with respect to $\to_{a,g,b}$. This is not necessarily the case for $\to_g$ and $\to_G$. However, as an important consequence of the properties of monomial orders, we see that $\to_G$ (and as a special case $\to_g$) is also terminating. To prove this, we extend the monomial order $\preceq$ to a strict partial order $\lll$ on polynomials as done in [Win96]. Recall that $\mathrm{lm}(0) = 0 \prec w$ for all $w \in \langle X\rangle$.

Definition 2.4.23. *Let $f, g \in R\langle X\rangle$. Then $f \lll g$ if one of the following conditions holds:*

1. $\mathrm{lm}(f) \prec \mathrm{lm}(g)$*;*
2. $\mathrm{lm}(f) = \mathrm{lm}(g)$ *and* $\mathrm{tail}(f) \lll \mathrm{tail}(g)$*;*

Remark 2.4.24. *The order $\lll$ coincides with the multiset order $<_{\mathrm{mul}}$ discussed in [BN98, Sec. 2.5] when $<_{\mathrm{mul}}$ is induced by $\prec$ and restricted to finite sets.*

Several properties of the monomial order $\preceq$ carry over to $\lll$. In particular, the property that infinite strictly decreasing sequences of elements do not exist.

Proposition 2.4.25. *There exist no infinite strictly decreasing sequences of elements in $K\langle X\rangle$ with respect to $\lll$.*

Proof. Assume, for contradiction, that there exists an infinite strictly decreasing sequence

$$f_0 \gg f_1 \gg \dots$$

of elements $f_0, f_1, \dots \in K\langle X\rangle$. Choose this sequence so that $\operatorname{lm}(f_0)$ is minimal with respect to $\preceq$ among all $f \in K\langle X\rangle$ starting an infinite sequence. Choosing such f_0 is possible because $\preceq$ is a well-order. Note that then $\operatorname{lm}(f_0) = \operatorname{lm}(f_n)$ for all $n \in \mathbb{N}$ because $\operatorname{lm}(f_0) \succ \operatorname{lm}(f_n)$ would violate the minimality of $\operatorname{lm}(f_0)$ as f_n also starts an infinite strictly decreasing sequence $f_n \gg f_{n+1} \gg \dots$. Thus, by definition of $\ll$, we must have

$$\operatorname{tail}(f_0) \gg \operatorname{tail}(f_1) \gg \dots,$$

which is still an infinite sequence. Note that $f_0 \neq 0$ as 0 is minimal with respect to $\ll$. But then $\operatorname{lm}(\operatorname{tail}(f_0)) \prec \operatorname{lm}(f_0)$, which is a contradiction to the minimality of $\operatorname{lm}(f_0)$. □

Corollary 2.4.26. *The reduction $\to_G$ is terminating.*

Proof. By the definition of $\ll$, any reduction $f \to_G f'$ implies $f \gg f'$. Thus, an infinite sequence of reductions $f_0 \to_G f_1 \to_G \dots$ yields an infinite sequence $f_0 \gg f_1 \gg \dots$ of elements in $K\langle X\rangle$, which cannot happen by Proposition 2.4.25. □

Recall from Definition 2.3.3, that $\xrightarrow{*}$ denotes the reflexive transitive closure of a reduction $\to$. For $\to_G$, this means that $f \xrightarrow{*}_G f'$ if either $f = f'$ or if there exist finitely many $h_0, \dots, h_d \in K\langle X\rangle$ such that

$$f = h_0 \ \to_G \ h_1 \ \to_G \ \dots \ \to_G \ h_d = f'.$$

Note that, despite yielding unique normal forms, the reduction $\to_{a,g,b}$ is not a function, as it is only defined for polynomials with $\operatorname{lm}(agb)$ in their support. This causes proofs involving polynomial reduction to be a bit cumbersome at times, as one has to make case distinctions depending on whether an element is reducible or not. To avoid these case distinctions, it is convenient to associate a function to this reduction.

Therefore, we associate to $\to_{a,g,b}$ the K-vector space endomorphism

$$r_{a,g,b} \colon K\langle X\rangle \to K\langle X\rangle$$

defined on the basis $\langle X \rangle$ to be

$$r_{a,g,b}(w) \coloneqq \begin{cases} f' & \text{if } w \to_{a,g,b} f' \\ w & \text{otherwise} \end{cases}$$

and extended K-linearly. In [Ber78], such a function is called a *reduction homomorphism*. It relates to our notion of polynomial reduction via the following result.

Lemma 2.4.27. *Let $G \subseteq K\langle X \rangle$ and $f, f' \in K\langle X \rangle$ such that $f \xrightarrow{*}_G f'$. There exists a K-vector space endomorphism $r \colon K\langle X \rangle \to K\langle X \rangle$ satisfying*

1. *$r(f) = f'$;*
2. *$g \xrightarrow{*}_G r(g)$ for all $g \in K\langle X \rangle$;*
3. *$r(g) = g$ for all irreducible $g \in K\langle X \rangle$;*

Furthermore, r is a composition of reduction homomorphisms.

Proof. If $f = f'$, we can set $r = \mathrm{id}$ to be the empty composition of reduction homomorphisms, which satisfies all the required properties. Otherwise, f' is obtained by a sequence of reductions

$$f \ \to_{a_1,g_1,b_1} \ \cdots \ \to_{a_d,g_d,b_d} \ f',$$

with $a_i, b_i \in \langle X \rangle$ and $g_i \in G$. To each reduction, we consider the associated reduction homomorphism r_{a_i,g_i,b_i} and define r as the composition $r = r_{a_d,g_d,b_d} \circ \cdots \circ r_{a_1,g_1,b_1}$. This is again a K-vector space endomorphism. We show the three claimed properties for the case $d = 1$, that is, $r = r_{a,g,b}$. The general case $d \geq 1$ follows by induction.

Claim 1 follows immediately from the fact that both $\to_{a,g,b}$ and $r_{a,g,b}$ act in the same way on $\mathrm{lm}(agb)$ and leave all other words unchanged. Claim 2 holds trivially if $g = r(g)$. Otherwise, by definition of reduction homomorphisms, $\mathrm{lm}(agb)$ appears in g and we have $g \to_{a,g,b} r(g)$, showing that also $g \xrightarrow{*}_G r(g)$. Finally, 3 follows from 2. □

Using reduction homomorphisms, we can easily prove the following result about polynomial reduction that will be needed later.

Lemma 2.4.28. *Let $G \subseteq K\langle X\rangle$ and $f, f' \in K\langle X\rangle$. Then the following hold:*

1. *$f \xrightarrow{*}_G f'$ implies $cafb \xrightarrow{*}_G caf'b$ for all $c \in K$ and $a, b \in \langle X\rangle$;*
2. *$f - f' \xrightarrow{*}_G 0$ implies $f \downarrow_G f'$;*

Proof. For $c = 0$, the first part holds trivially. For nonzero c, the first part follows from the definition, since $f \to_{u,g,v} f'$ implies $cafb \to_{au,g,vb} caf'b$. For the second part, let r be the K-vector space endomorphism associated to the reduction $f - f' \xrightarrow{*}_G 0$ by Lemma 2.4.27. Then $r(f) - r(f') = r(f - f') = 0$ and Lemma 2.4.27 yields $f \xrightarrow{*}_G r(f) = r(f') \ {}_G\xleftarrow{*} f'$, showing that $f \downarrow_G f'$. □

If a sequence of reductions yields zero, we can obtain a cofactor representation of the reduced polynomial with respect to the reducers. More precisely, for a nonzero $f \in K\langle X\rangle$, the fact $f \xrightarrow{*}_G 0$ corresponds to the existence of a reduction sequence

$$f = h_0 \ \to_{a_1,g_1,b_1} \ h_1 \ \to_{a_2,g_2,b_2} \ \dots \ \to_{a_d,g_d,b_d} \ h_d = 0.$$

Expanding this sequence yields the following cofactor representation of f with respect to G:

$$f = \sum_{i=1}^{d} c_i a_i g_i b_i, \tag{2.3}$$

where $c_i = \operatorname{coeff}(h_{i-1}, \operatorname{lm}(a_i g_i b_i))/\operatorname{lc}(g_i)$. This immediately implies the following result.

Lemma 2.4.29. *If $f \xrightarrow{*}_G 0$, then $f \in (G)$. More generally, $f \xrightarrow{*}_G f'$ implies $f - f' \in (G)$.*

An interesting aspect of cofactor representations obtained by reductions to zero is that they are *bounded* as defined below.

Definition 2.4.30. *Let $G \subseteq K\langle X\rangle$. For a nonzero $f \in (G)$, a cofactor representation*

$$f = \sum_{i=1}^{d} a_i g_i b_i,$$

with $a_i, b_i \in K\langle X\rangle$ and $g_i \in G$, of f with respect to G is called bounded *if $\operatorname{lm}(a_i g_i b_i) \preceq \operatorname{lm}(f)$ for all $i = 1, \dots, d$.*

Corollary 2.4.31. *If $f \overset{*}{\to}_G 0$, then f has a bounded cofactor representation with respect to G. More precisely, the cofactor representation* (2.3) *is bounded.*

Proof. Equation (2.3) is clearly a cofactor representation of f with respect to G. The fact that it is bounded follows from Lemma 2.4.22. □

Lemma 2.4.29 provides a first connection between polynomial reduction and the ideal membership problem in $K\langle X\rangle$: Reductions to zero imply ideal membership. This condition, however, is only sufficient and generally not necessary as the following example shows.

Example 2.4.32. *Consider the ideal $I = (f_1, f_2) \trianglelefteq \mathbb{Q}\langle x, y\rangle$ with $f_1 = xy+1$, $f_2 = xyx-1$. Then*

$$x + 1 = f_1 x - f_2 \in I,$$

but this element is irreducible with respect to $\to_{\{f_1,f_2\}}$ (with respect to any monomial order on $\langle x, y\rangle$).

By allowing also inverse reductions, we obtain a full characterisation of ideal membership via polynomial reduction.

Theorem 2.4.33. *Let $G \subseteq K\langle X\rangle$ and $f, f' \in K\langle X\rangle$. Then $f \in (G)$ if and only if $f \overset{*}{\leftrightarrow}_G 0$. More generally, $f - f' \in (G)$ if and only if $f \overset{*}{\leftrightarrow}_G f'$.*

Proof. The first assertion clearly follows from the second one. Thus, we show the latter.

For the "if"-direction, define a relation on $K\langle X\rangle$ by $f \sim_G f'$ if $f - f' \in (G)$. This is an equivalence relation. Furthermore, note that $\overset{*}{\leftrightarrow}_G$ is the smallest equivalence relation containing $\to_G$. Thus, $\to_G \subseteq \sim_G$ implies $\overset{*}{\leftrightarrow}_G \subseteq \sim_G$, which in turn yields the desired result. But $\to_G \subseteq \sim_G$ follows from Lemma 2.4.29 and the fact that $\to_G \subseteq \overset{*}{\to}_G$.

For the "only if"-direction, write $f - f'$ as $f - f' = \sum_{i=1}^{d} c_i a_i g_i b_i$ with nonzero $c_i \in K$, $a_i, b_i \in \langle X\rangle$ and $g_i \in G$. We show that $f \overset{*}{\leftrightarrow}_G f'$ holds for the case $d = 1$. Then the general statement for $d \geq 1$ follows by induction on d. Clearly $g_1 \to_G 0$, and thus, by the first part of Lemma 2.4.28, also $f - f' = c_1 a_1 g_1 b_1 \to_G 0$. With this, the second part of Lemma 2.4.28 yields $f \downarrow_G f'$, and consequently, $f \overset{*}{\leftrightarrow}_G f'$. □

From a computational point of view, transitioning from $\overset{*}{\to}_G$ to $\overset{*}{\leftrightarrow}_G$ is problematic because the inverse relation is not terminating. However, if $\to_G$ is confluent, then Theorem 2.4.33 yields an algorithmic way to solve the ideal membership problem.

Corollary 2.4.34. *If $\to_G$ is confluent, then $f \in (G)$ if and only if $f \overset{*}{\to}_G 0$.*

Proof. By Theorem 2.4.33, $f \in (G)$ if and only if $f \overset{*}{\leftrightarrow}_G 0$. Note that $\to_G$ is terminating by Corollary 2.4.26, and thus, normalising. Hence, if $\to_G$ is additionally confluent, Theorem 2.3.13 is applicable and yields $f \overset{*}{\leftrightarrow}_G 0$ if and only if f and 0 have the same unique normal form, and the result follows since 0 is in normal form. □

Hence, ideal membership in an ideal (G) can be decided by polynomial reduction provided that $\to_G$ is confluent. Unfortunately, this is not the case for all sets G. Such distinguished sets for which $\to_G$ is confluent are called *Gröbner bases* of the ideal (G).

2.4.3 Gröbner bases of two-sided ideals

Recall that we have fixed a monomial order $\preceq$ on $\langle X \rangle$.

Definition 2.4.35. *Let $I \trianglelefteq K\langle X \rangle$. A subset $G \subseteq I$ is a* Gröbner basis *of I if $(G) = I$ and $\to_G$ is confluent.*

By Theorem 2.3.13, every element $f \in K\langle X \rangle$ has a unique normal form $f{\downarrow_G}$ with respect to $\to_G$ if G is a Gröbner basis. Conversely, if every element has a unique normal form, then $\to_G$ is clearly confluent, and thus, G is a Gröbner basis. So we arrive at the following alternative characterisation of Gröbner bases.

Corollary 2.4.36. *Let $I \trianglelefteq K\langle X \rangle$. A subset $G \subseteq I$ is a Gröbner basis of I if and only if $(G) = I$ and every $f \in K\langle X \rangle$ has a unique normal form $f{\downarrow_G}$ with respect to $\to_G$.*

In fact, there exist many other characterisations of Gröbner bases. In the following, we list a few common ones.

Theorem 2.4.37. *Let $I \trianglelefteq K\langle X\rangle$ and $G \subseteq I$. The following conditions are equivalent:*

1. *G is a Gröbner basis of I;*
2. *$f \xrightarrow{*}_G 0$ for all $f \in I$;*
3. *$(\mathrm{lm}(I)) = (\mathrm{lm}(G))$;*
4. *for all nonzero $f \in I$, there exists $g \in G$ such that $\mathrm{lm}(g)$ divides $\mathrm{lm}(f)$;*

Proof. We show $1 \iff 2 \implies 3 \implies 4 \implies 2$.

$1 \implies 2$ Follows from Corollary 2.4.34.

$2 \implies 1$ Note that $(G) = I$ follows from Lemma 2.4.29. For the confluence, let $f, f_1, f_2 \in K\langle X\rangle$ be such that $f_1 \,{}_G\!\xleftarrow{*} f \xrightarrow{*}_G f_2$. Then clearly $f_1 \xleftrightarrow{*}_G f_2$, and thus, $f_1 - f_2 \in I$ by Theorem 2.4.33. Our assumption implies $f_1 - f_2 \xrightarrow{*}_G 0$. With this, Lemma 2.4.28 yields $f_1 \downarrow_G f_2$, showing that $\to_G$ is confluent.

$2 \implies 3$ The inclusion $(\mathrm{lm}(G)) \subseteq (\mathrm{lm}(I))$ is clear because $G \subseteq I$. For the other inclusion, it suffices to show that $\mathrm{lm}(I) \subseteq (\mathrm{lm}(G))$ because $(\mathrm{lm}(I))$ is the smallest ideal containing $\mathrm{lm}(I)$. To this end, let $f \in I$. For $f = 0$, note that $\mathrm{lm}(f) = 0 \in (\mathrm{lm}(G))$. Thus, we can assume $f \neq 0$. Then, by assumption, $f \xrightarrow{*}_G 0$, but this is only possible if f is top reducible by G, that is, if there exist $a, b \in \langle X\rangle$ and $g \in G$ such that $\mathrm{lm}(f) = \mathrm{lm}(agb)$. This yields $\mathrm{lm}(f) = \mathrm{lm}(agb) = a\mathrm{lm}(g)b \in (\mathrm{lm}(G))$.

$3 \implies 4$ Let $f \in I$ be nonzero. By assumption, $\mathrm{lm}(f) \in (\mathrm{lm}(I))$ can be written as

$$\mathrm{lm}(f) = \sum_{i=1}^{d} c_i a_i \mathrm{lm}(g_i) b_i,$$

with $c_i \in K$, $a_i, b_i \in \langle X\rangle$ and $g_i \in G$. Since $\mathrm{lm}(f)$ is a monomial, all summands in this linear combination except for one must cancel, so that $\mathrm{lm}(f) = a_j \mathrm{lm}(g_j) b_j$ for some $1 \leq j \leq d$, showing that $\mathrm{lm}(f)$ is divisible by $\mathrm{lm}(g_j)$.

$4 \implies 2$ Assume, for contradiction, that not all elements in I can be reduced to zero with respect to G. Let $f \in I$ be such an element. Without loss of generality, we can assume that f is irreducible with respect to $\to_G$. By assumption, there exists $g \in G$ such that $\mathrm{lm}(g)$ divides $\mathrm{lm}(f)$, but then f is (top) reducible by g – a contradiction. □

We make some remarks about the characterisations in Theorem 2.4.37 and their consequences:

- While Definition 2.4.35 has to require explicitly that G is a generating set of the ideal I, we get this property for free with all conditions of Theorem 2.4.37.
- As a consequence of condition 3, we see that every ideal I has a Gröbner basis. For example, taking $G = I$ satisfies this condition.
- We can also see that a Gröbner basis of I is by no means unique. In fact, if G is a Gröbner basis of I, then so is $G \cup \{f\}$ for every $f \in I$.
- Condition 2 yields an algorithm for solving the ideal membership problem provided that we know a Gröbner basis for the ideal I of interest. To determine whether f lies in I, simplify reduce f to normal form and see if the result is zero.
- Condition 3 is commonly used as the definition of Gröbner bases. For example, the commutative standard reference [CLO15] as well as the noncommutative introductory text [Mor94] use this condition.
- The (polynomial) ideals in condition 3 can be replaced by monoid ideals, leading to the statement that the monoid ideal $\mathrm{lm}(I)$ is equal to the monoid ideal generated by $\mathrm{lm}(G)$, which is essentially condition 4. We refer, for example, to [Xiu12] for the definition of monoid ideals and for further information.

Another common characterisation of Gröbner bases is the following.

Proposition 2.4.38. *Let $I \trianglelefteq K\langle X\rangle$. A subset $G \subseteq I$ is a Gröbner basis of I if and only if all nonzero $f \in I$ have a bounded cofactor representation with respect to G.*

Proof. The "only if"-part of the statement follows from Corollary 2.4.31 and condition 2 of Theorem 2.4.37. For the "if"-part, we prove that the assumption implies condition 4 of Theorem 2.4.37. To this end, choose an arbitrary nonzero $f \in I$. By assumption, f can be written as

$$f = \sum_{i=1}^{d} a_i g_i b_i,$$

with $a_i, b_i \in K\langle X\rangle$ and $g_i \in G$ such that $\mathrm{lm}(a_i g_i b_i) \preceq \mathrm{lm}(f)$ for all $i = 1, \ldots, d$. In particular, for at least one $1 \leq j \leq d$, we must have $\mathrm{lm}(f) = \mathrm{lm}(a_j g_j b_j)$, showing that $\mathrm{lm}(g_j)$ divides $\mathrm{lm}(f)$. □

Although a Gröbner basis of an ideal is not unique in general, we can demand certain additional properties in order to obtain a distinguished and unique Gröbner basis (with respect to a fixed monomial order), called the *reduced Gröbner basis*. To introduce this notion, we first define an *interreduced* set of polynomials.

Definition 2.4.39. *A set $G \subseteq K\langle X\rangle$ is* interreduced *if every $g \in G$ is irreducible with respect to $\to_{G\setminus\{g\}}$.*

Definition 2.4.40. *A Gröbner basis $G \subseteq I$ of an ideal $I \trianglelefteq K\langle X\rangle$ is* reduced *if G is interreduced and all polynomials in G are monic.*

The following result establishes existence and uniqueness of reduced Gröbner bases.

Proposition 2.4.41. *Every ideal in $K\langle X\rangle$ has a unique reduced Gröbner basis (with respect to a fixed monomial order).*

Proof. Let $I \trianglelefteq K\langle X\rangle$ be an ideal. To show existence, let $G \subseteq I$ be a Gröbner basis and set

$$H = \{g \in G \setminus \{0\} \mid \mathrm{lm}(g) \text{ is not divisible by any word in } \mathrm{lm}(G \setminus \{g\})\}.$$

By condition 3 of Theorem 2.4.37, the set H is also a Gröbner basis of I because $(\mathrm{lm}(H)) = (\mathrm{lm}(G)) = (\mathrm{lm}(I))$. Next, set

$$G' = \{\mathrm{lm}(g) - (\mathrm{lm}(g)\downarrow_H) \mid g \in H\}.$$

We show that G' is a reduced Gröbner basis of I. By Lemma 2.4.29, we see that $G' \subseteq I$. Furthermore, $(\mathrm{lm}(G')) = (\mathrm{lm}(H)) = (\mathrm{lm}(G)) = (\mathrm{lm}(I))$, and so G' is a Gröbner basis of I. Finally, by construction, G' is interreduced and all its elements are monic.

To show uniqueness, let $G, H \subseteq I$ be reduced Gröbner bases of I. We show that $G \subseteq H$, and the other inclusion will follow from the symmetry of our argument. Let $g \in G$. Since H is a Gröbner basis, there exists $h \in H$ such that $\mathrm{lm}(h)$ divides $\mathrm{lm}(g)$. Since G is also a

Gröbner basis, there exists $g' \in G$ such that $\mathrm{lm}(g')$ divides $\mathrm{lm}(h)$. This is only possible if $g = g'$ because G is interreduced. Consequently, $\mathrm{lm}(g) = \mathrm{lm}(h)$ and, more precisely, $\mathrm{lt}(g) = \mathrm{lt}(h)$ because both elements are monic. We show that, in fact, $g = h$. To this end, consider the difference $g - h \in I$ and note that $\mathrm{lt}(g) = \mathrm{lt}(h)$ cancel in $g - h$. Since G and H are interreduced, none of the remaining terms can be reducible. So $g - h$ is in normal form, showing that $g - h = 0$ by condition 2 of Theorem 2.4.37. □

Given a finite Gröbner basis of an ideal $I \trianglelefteq K\langle X\rangle$, the proof of Proposition 2.4.41 shows how to construct the (finite) reduced Gröbner basis of I. As a direct consequence of this construction, we obtain the following result.

Corollary 2.4.42. *An ideal in $K\langle X\rangle$ has a finite Gröbner basis if and only if its reduced Gröbner basis is finite.*

Proof. The "if"-direction is clear. For the "only if"-direction, let G be a finite Gröbner basis of an ideal I. Following the proof of Proposition 2.4.41, the reduced Gröbner basis G' of I can be constructed from a subset of the leading monomials in G, implying that $|G'| \leq |\mathrm{lm}(G)| \leq |G| < \infty$. □

However, note that the reduced Gröbner basis of an ideal need not be finite, not even for principal ideals (see Example 2.4.55). This implies that there exist finitely generated ideals in $K\langle X\rangle$ that do not admit a finite Gröbner basis.

An important property of Gröbner bases is that they possess the so-called *elimination property* allowing to compute elimination ideals [BB98]. If $I \trianglelefteq K\langle X, Y\rangle$ is an ideal, then, analogous to the commutative case (see [CLO15, Thm. 3.1.2]), the elimination property of noncommutative Gröbner bases allows to obtain a Gröbner basis of the *elimination ideal* $I \cap K\langle X\rangle$ from a suitable Gröbner basis of I. A Gröbner basis is suitable to do this, if it is computed with respect to an elimination order for Y (see Definition 2.4.15). The following theorem states the elimination property of noncommutative Gröbner bases, see also [BB98, Thm. 3.2].

Theorem 2.4.43. *Let $I \trianglelefteq K\langle X, Y\rangle$ and let G be a Gröbner basis of I with respect to an elimination order for Y. Then $G \cap K\langle X\rangle$ is a Gröbner basis of the elimination ideal $I \cap K\langle X\rangle$.*

Proof. Let $G_X = G \cap K\langle X\rangle$. Clearly, $G_X \subseteq I \cap K\langle X\rangle$. We use condition 4 of Theorem 2.4.37 to show that G_X is a Gröbner basis of $I \cap K\langle X\rangle$. To this end, let $f \in I \cap K\langle X\rangle$ be nonzero. Since G is a Gröbner basis of I, there exists $g \in G$ such that $\mathrm{lm}(g)$ divides $\mathrm{lm}(f)$. Since $f \in K\langle X\rangle$, we have $\mathrm{lm}(f) \in \langle X\rangle$, and thus, also $\mathrm{lm}(g) \in \langle X\rangle$. Since G is computed with respect to an elimination order for Y, this implies $g \in K\langle X\rangle$, showing that $g \in G_X$. □

A Gröbner basis of an ideal I also allows to perform computations in the quotient ring $K\langle X\rangle/I$. To make this precise, for $G \subseteq K\langle X\rangle$, we denote by $K\langle X\rangle_{\mathrm{irr,G}} \subseteq K\langle X\rangle$ the set of all irreducible elements with respect to $\to_G$, that is,

$$K\langle X\rangle_{\mathrm{irr,G}} = \{f \in K\langle X\rangle \mid f \text{ is irreducible w.r.t. } \to_G\}.$$

Note that this set is a K-vector space with basis consisting of all irreducible words. Furthermore, we consider the set

$$K\langle X\rangle_{\mathrm{un,G}} = \{f \in K\langle X\rangle \mid f \text{ has a unique normal form } f{\downarrow_G} \text{ w.r.t. } \to_G\}.$$

of all elements with a unique normal form. The following lemma states that this is also a K-vector space.

Lemma 2.4.44. *Let $G \subseteq K\langle X\rangle$. Then the following hold:*

1. *$K\langle X\rangle_{\mathrm{un,G}}$ is a K-vector space;*
2. *the map $\psi\colon K\langle X\rangle_{un,G} \to K\langle X\rangle_{\mathrm{irr,G}}, f \mapsto f{\downarrow_G}$ is K-linear;*
3. *for $f, g, h \in K\langle X\rangle$, if $\mathrm{supp}(fgh) \subseteq K\langle X\rangle_{\mathrm{un,G}}$, then $fgh \in K\langle X\rangle_{\mathrm{un,G}}$. Moreover, if $g \xrightarrow{*}_G g'$, then $fg'h \in K\langle X\rangle_{\mathrm{un,G}}$ and $(fg'h){\downarrow_G} = (fgh){\downarrow_G}$.*

Proof. We show 1 and 2 using reduction homomorphisms. To this end, let $c \in K$ and $f, g \in K\langle X\rangle_{\mathrm{un,G}}$. Let h be a normal form of $cf + g$ with respect to $\to_G$. By Lemma 2.4.27, there exists a K-vector space endomorphism r satisfying $r(cf + g) = h$ and, since h is irreducible, $r(h) = h$. Furthermore, also $f \xrightarrow{*}_G r(f)$, and therefore $r(f) \xrightarrow{*}_G f{\downarrow_G}$. Now, again by Lemma 2.4.27, we obtain a K-vector space endomorphism r' so that $r'(r(f)) = f{\downarrow_G}$ and $r'(h) = h$. With this, we now have $g \xrightarrow{*}_G r'(r(g)) \xrightarrow{*}_G g{\downarrow_G}$, yielding

yet another K-vector space endomorphism r'' with $r''(r'(r(g))) = g{\downarrow_G}$ and $r''(h) = h$ as well as $r''(f{\downarrow_G}) = f{\downarrow_G}$. Combining all of this yields

$$h = r''(r'(r(h))) = r''(r'(r(cf+g))) = cr''(r'(r(f))) + r''(r'(r(g))) = cf{\downarrow_G} + g{\downarrow_G},$$

showing that $cf{\downarrow_G} + g{\downarrow_G}$ is the unique normal form of $cf + g$. This also shows that $\psi(cf+g) = c\psi(f) + \psi(g)$, and thus that ψ is K-linear.

For 3, note that $fgh \in K\langle X\rangle_{\mathrm{un,G}}$ follows from 1. To show that $fg'h$ has a unique normal form, we first assume that $f, g, h \in \langle X\rangle$ are monomials. In this case, $fgh \xrightarrow{*}_G fg'h$ by Lemma 2.4.28, and since the former has a unique normal form, so does the latter, with the same normal form. For the general case, write $f = \sum_{A_i \in \langle X\rangle} a_i A_i$, $g = \sum_{B_j \in \langle X\rangle} b_j B_j$ and $h = \sum_{C_k \in \langle X\rangle} c_k C_k$. Furthermore, let r be the map that Lemma 2.4.27 associates to the reduction $g \xrightarrow{*}_G g'$. With this, we have

$$fg'h = fr(g)h = \sum_{i,j,k} a_i b_j c_k A_i r(B_j) C_k.$$

From the case for monomials, we know that $A_i r(B_j) C_k$ has a unique normal form for all i, j, k and $(A_i B_j C_k){\downarrow_G} = (A_i r(B_j) C_k){\downarrow_G}$. Now the result follows from part 1 and 2. □

Note that, by Corollary 2.4.36, the set $K\langle X\rangle_{\mathrm{un,G}}$ is all of $K\langle X\rangle$ if and only if G is a Gröbner basis of (G). With this, we can state yet another characterisation of Gröbner bases.

Theorem 2.4.45. *Let $I \trianglelefteq K\langle X\rangle$. A subset $G \subseteq I$ is a Gröbner basis of I if and only if $K\langle X\rangle = I \oplus K\langle X\rangle_{\mathrm{irr,G}}$.*

Proof. For the "if"-direction, we show condition 2 of Theorem 2.4.37. To this end, let $f \in I$ and let $f' \in K\langle X\rangle_{\mathrm{irr,G}}$ be a normal form of f with respect to $\to_G$. By Lemma 2.4.29, $f - f' \in I$, and since $f \in I$, we get that also $f' \in I$. Thus $f' \in I \cap K\langle X\rangle_{\mathrm{irr,G}} = \{0\}$.

For the "only if"-direction, consider the K-linear map

$$\psi\colon K\langle X\rangle_{\mathrm{un,G}} \to K\langle X\rangle_{\mathrm{irr,G}}, \quad f \mapsto f{\downarrow_G}.$$

from Lemma 2.4.44. Note that ψ is idempotent. Hence, ψ is a projection and therefore $K\langle X\rangle = K\langle X\rangle_{\mathrm{un,G}} = \ker(\psi) \oplus \mathrm{im}(\psi)$, where the first equality follows since G is a Gröbner basis. To finish the proof, note that $\ker(\psi) = I$ and $\mathrm{im}(\psi) = K\langle X\rangle_{\mathrm{irr,G}}$. □

As a consequence of this theorem, we see that $K\langle X\rangle/I$ and $K\langle X\rangle_{\mathrm{irr,G}}$ are isomorphic as K-vector spaces provided that G is a Gröbner basis of I. Thus, the normal forms with respect to $\to_G$ form a system of representatives of the quotient ring. This yields an algorithmic way to perform computation in $K\langle X\rangle/I$:

$$[f] + [g] = [f{\downarrow_G} + g{\downarrow_G}] \qquad [f] \cdot [g] = [(fg){\downarrow_G}]$$

Thus far, none of the characterisations of Gröbner bases is constructive in the sense that it allows to algorithmically verify whether a given set is a Gröbner basis of an ideal. In the commutative case, Buchberger's Criterion [CLO15, Thm. 2.6.6] provides precisely such a constructive characterisation. In the following, we present a noncommutative version, typically referred to as *Bergman's diamond lemma* [Ber78]. As a first step towards phrasing this result, we define the notion of *ambiguities* as done in [Ber78].

Definition 2.4.46. *Let $p, q \in \langle X\rangle$. If there exist $a, b \in \langle X\rangle \setminus \{1\}$ with $|a| < |q|$ and $|b| < |p|$ such that $ap = qb$ (resp. $pa = bq$), then we call the tuple*

$$(a \otimes 1, 1 \otimes b, p, q) \qquad (\textit{resp. } (1 \otimes a, b \otimes 1, p, q))$$

an overlap ambiguity *of p and q.*

If there exist $a, b \in \langle X\rangle$ such that $p = aqb$ (resp. $apb = q$), then we call the tuple

$$(1 \otimes 1, a \otimes b, p, q) \qquad (\textit{resp. } (a \otimes b, 1 \otimes 1, p, q)),$$

an inclusion ambiguity *of p and q.*

For $f, g \in K\langle X\rangle \setminus \{0\}$, an (overlap/inclusion) ambiguity *of f and g is*

$$(a \otimes b, c \otimes d, f, g),$$

where $(a \otimes b, c \otimes d, \mathrm{lm}(f), \mathrm{lm}(g))$ is an (overlap/inclusion) ambiguity of $\mathrm{lm}(f)$ and $\mathrm{lm}(g)$ and, in case $f = g$, at least one of a, b, c, d is not 1. We denote by $\mathrm{amb}(f, g)$ the set of all

ambiguities of f and g. When clear by context, we drop the polynomials f and g from an ambiguity and simply write $(a \otimes b, c \otimes d) \in \operatorname{amb}(f, g)$.

Remark 2.4.47. *Ambiguities correspond to compositions in [Bok76] and to obstructions in [Mor94; Xiu12].*

Note that two elements can give rise to several ambiguities but only to finitely many. Furthermore, a polynomial can also form overlap ambiguities with itself. We record the following property which follows directly from the definition.

Lemma 2.4.48. *If $(a \otimes b, c \otimes d)$ is an ambiguity of f and g, then $\operatorname{lm}(afb) = \operatorname{lm}(cgd)$.*

Based on this result, we define the *leading monomial* of an ambiguity.

Definition 2.4.49. *Let $f, g \in K\langle X\rangle \setminus \{0\}$ and $\mathfrak{a} = (a \otimes b, c \otimes d) \in \operatorname{amb}(f, g)$. The* leading monomial *of $\mathfrak{a}$ is $\operatorname{LM}(\mathfrak{a}) \coloneqq \operatorname{lm}(afb) = \operatorname{lm}(cgd)$.*

An ambiguity $\mathfrak{a}$ of two polynomials f and g captures the fact that the term $\operatorname{LM}(\mathfrak{a})$ can be reduced by f and g in two (possibly different) ways. A set G containing f and g can only be a Gröbner basis if these two reductions lead to a canonical normal form. To check whether this is the case, one can look at the *S-polynomial* of the ambiguity. In the following, for a set $G \subseteq K\langle X\rangle$, let

$$\operatorname{amb}(G) \coloneqq \bigcup_{f,g \in G\setminus\{0\}} \operatorname{amb}(f, g).$$

Definition 2.4.50. *Let $G \subseteq K\langle X\rangle$ and $\mathfrak{a} = (a \otimes b, c \otimes d, f, g) \in \operatorname{amb}(G)$. The* S-polynomial *of $\mathfrak{a}$ is*

$$\text{S-Pol}(\mathfrak{a}) \coloneqq \frac{1}{\operatorname{lc}(f)} afb - \frac{1}{\operatorname{lc}(g)} cgd.$$

The ambiguity $\mathfrak{a}$ is resolvable *if $\text{S-Pol}(\mathfrak{a}) \xrightarrow{*}_G 0$.*

The following result is an immediate consequence of the definition.

Lemma 2.4.51. *For $\mathfrak{a} \in \operatorname{amb}(G)$, we have $\operatorname{lm}(\text{S-Pol}(\mathfrak{a})) \prec \operatorname{LM}(\mathfrak{a})$ and $\text{S-Pol}(\mathfrak{a}) \in (G)$.*

If an ambiguity $\mathfrak{a}$ is resolvable, then, by Lemma 2.4.28, the two different reductions of $\mathrm{LM}(\mathfrak{a})$ are joinable, thus yielding the confluence property for these reductions. Furthermore, by Corollary 2.4.31, the resolvability of an ambiguity implies that its S-polynomial has a bounded cofactor representation with respect to G.

We note that our definition of resolvability differs slightly from the one in [Ber78]. The original definition is in general weaker than Definition 2.4.50, requiring only that the two polynomials that form the S-polynomial are joinable. In fact, [Ber78] also uses a second, even weaker, notion of resolvability, called *resolvable relative to* a monomial order $\preceq$, which we discuss below. In Theorem 2.4.54, we show that, for a Gröbner basis G, the different notions of resolvability are all equivalent.

In the following, for $G \subseteq K\langle X\rangle$ and $w \in \langle X\rangle$, let

$$I_{G,w} \coloneqq \left\{ \sum_{i=1}^{d} a_i g_i b_i \;\middle|\; d \in \mathbb{N}, a_i, b_i \in K\langle X\rangle, g_i \in G, \mathrm{lm}(a_i g_i b_i) \prec w \right\}.$$

Definition 2.4.52. *Let $G \subseteq K\langle X\rangle$. An ambiguity $\mathfrak{a} \in \mathrm{amb}(G)$ is* resolvable relative to $\preceq$ *if* $\text{S-Pol}(\mathfrak{a}) \in I_{G,\mathrm{LM}(\mathfrak{a})}$.

Definition 2.4.52 can also be phrased equivalently in terms of an analogue of the commutative notion of *lcm representations* [CLO15, Def. 2.9.5]. A cofactor representation of an S-polynomial $\text{S-Pol}(\mathfrak{a})$ is called an *lcm representation* if it is of the form

$$\text{S-Pol}(\mathfrak{a}) = \sum_{i=1}^{d} a_i g_i b_i,$$

with $\mathrm{lm}(a_i g_i b_i) \prec \mathrm{LM}(\mathfrak{a})$ for all $i = 1, \ldots, d$. With this, an ambiguity is resolvable relative to $\preceq$ if and only if $\text{S-Pol}(\mathfrak{a})$ has an lcm representation.

As described above, resolvability of an ambiguity is related to the existence of a bounded cofactor representation of the S-polynomial. Analogously, resolvability relative to $\preceq$ corresponds to the existence of lcm representations. In the following, we describe how lcm representations are related to bounded cofactor representations, and thus, how the two

notions of resolvability are related to each other. To this end, write $\text{S-Pol}(\mathfrak{a}) = \sum_{i=1}^{d} c_i a_i g_i b_i$ and consider the inequalities:

$$\operatorname{lm}(\text{S-Pol}(\mathfrak{a})) \succeq \operatorname{lm}(a_i g_i b_i) \quad \text{for all } i = 1, \ldots, d, \tag{2.4}$$

$$\operatorname{LM}(\mathfrak{a}) \succ \operatorname{lm}(a_i g_i b_i) \quad \text{for all } i = 1, \ldots, d. \tag{2.5}$$

Note that (2.4) says that the cofactor representation is bounded, while (2.5) says that it is an lcm representation. We can see that (2.4) implies (2.5) because $\operatorname{LM}(\mathfrak{a}) \succ \operatorname{lm}(\text{S-Pol}(\mathfrak{a}))$ by the definition of S-polynomials. This shows that every bounded cofactor representation of an S-polynomial is also an lcm representation. The converse, however, need not hold, as, in an lcm representation, we make no assumptions on how $\operatorname{lm}(\text{S-Pol}(\mathfrak{a}))$ and $\operatorname{lm}(a_i g_i b_i)$ relate to each other. Thus, lcm representations are strictly weaker than bounded cofactor representations. This is also witnessed by the following example.

Example 2.4.53. *Consider the elements*

$$f_1 = xy + 1, \qquad f_2 = yz + 1, \qquad f_3 = xy - x + z + 1 \in \mathbb{Q}\langle x, y, z\rangle.$$

Using $\preceq_{\text{deglex}}$ *as a monomial order with* $x \prec_{\text{lex}} y \prec_{\text{lex}} z$*, we have an overlap ambiguity*

$$\mathfrak{a} = (1 \otimes z, x \otimes 1, f_1, f_2)$$

between f_1 *and* f_2*. The cofactor representation*

$$z - x = \text{S-Pol}(\mathfrak{a}) = (-1) \cdot f_1 + 1 \cdot f_3$$

is an lcm representation because $\operatorname{LM}(\mathfrak{a}) = xyz \succ \operatorname{lm}(f_1), \operatorname{lm}(f_3)$*, but it is not bounded as* $\operatorname{lm}(\text{S-Pol}(\mathfrak{a})) = z \prec \operatorname{lm}(f_1)$*.*

In Proposition 2.4.38, we have seen that a set G is a Gröbner basis of an ideal I if and only if all nonzero elements in I have a bounded cofactor representation with respect to G. The following *diamond lemma* [Ber78, Thm. 1.2] weakens this condition in two ways: First, not all elements in the ideal have to be considered but it is enough to only look at S-polynomials of G. Moreover, bounded cofactor representations can be replaced by the weaker lcm representations.

Theorem 2.4.54 (Bergman's diamond lemma)**.** *Let $G \subseteq K\langle X\rangle$. The following conditions are equivalent:*

1. *G is a Gröbner basis of (G);*
2. *all ambiguities of G are resolvable;*
3. *all ambiguities of G are resolvable relative to $\preceq$;*

Proof. The implications $1 \implies 2 \implies 3$ are clear. To finish the proof, we show $3 \implies 1$. More precisely, we show the condition of Corollary 2.4.36, that is, that every element in $K\langle X\rangle$ has a unique normal form. By Lemma 2.4.44, the set $K\langle X\rangle_{\mathrm{un,G}}$ is a K-vector space. Thus, it suffices to prove $\langle X\rangle \subseteq K\langle X\rangle_{\mathrm{un,G}}$ to show that $K\langle X\rangle = K\langle X\rangle_{\mathrm{un,G}}$. We show the former by well-founded induction on words $W \in \langle X\rangle$ with respect to the well-order $\preceq$.

Assume inductively that all words $\prec W$ have a unique normal form, which implies that all polynomials $f \in K\langle X\rangle$ with $\mathrm{lm}(f) \prec W$ have a unique normal form. In particular, $cagb{\downarrow_G} = 0$ for all $c \in K$, $a, b \in \langle X\rangle$ and $g \in G$ such that $\mathrm{lm}(agb) \prec W$, and so $f{\downarrow_G} = 0$ for all $f \in I_{G,W} \subseteq K\langle X\rangle_{\mathrm{un,G}}$ by Lemma 2.4.44.

We can assume that W is reducible, as otherwise it is its own unique normal form. Let $f_1, f_2 \in K\langle X\rangle_{\mathrm{irr,G}}$ be normal forms of W with respect to $\to_G$. Since W is reducible, we have $f_1, f_2 \neq W$, and therefore we can expand the first reduction of W, yielding:

$$f_1 \overset{*}{\leftarrow}_G h_1 \leftarrow_{a_1,g_1,b_1} W \to_{a_2,g_2,b_2} h_2 \overset{*}{\to}_G f_2$$

for some $h_1, h_2 \in K\langle X\rangle$. By the definition of polynomial reduction, $\mathrm{lm}(h_i) \prec W$, and thus $h_i \in K\langle X\rangle_{\mathrm{un,G}}$, for $i = 1, 2$, by induction hypothesis. So we have $f_i = h_i{\downarrow_G}$. We show that $h_1 \downarrow_G h_2$, which gives $f_1 = h_1{\downarrow_G} = h_2{\downarrow_G} = f_2$. To this end, we note that we can assume without loss of generality that g_1 and g_2 are monic. Then

$$h_2 - h_1 = (W - a_2g_2b_2) - (W - a_1g_1b_1) = a_1g_1b_1 - a_2g_2b_2.$$

We distinguish between different cases depending on the positions of $\mathrm{lm}(g_1)$ and $\mathrm{lm}(g_2)$ relative to each other in $W = \mathrm{lm}(a_1g_1b_1) = \mathrm{lm}(a_2g_2b_2)$. Without loss of generality, we can assume that $\mathrm{lm}(g_1)$ begins no later than $\mathrm{lm}(g_2)$ in W.

Case 1 $\mathrm{lm}(g_2)$ is fully contained in $\mathrm{lm}(g_1)$, that is, $\mathrm{lm}(g_1) = a\mathrm{lm}(g_2)b$ for some $a, b \in \langle X\rangle$. Then there exists the inclusion ambiguity $\mathfrak{a} = (1 \otimes 1, a \otimes b, g_1, g_2) \in \mathrm{amb}(G)$. The respective S-polynomial is $\text{S-Pol}(\mathfrak{a}) = g_1 - ag_2b$. Because $\mathfrak{a}$ is resolvable relative to $\preceq$, we have $\text{S-Pol}(\mathfrak{a}) \in I_{G,\mathrm{lm}(g_1)}$. Consequently, $h_2 - h_1 = a_1\text{S-Pol}(\mathfrak{a})b_1 \in I_{G,a_1\mathrm{lm}(g_1)b_1} = I_{G,W}$, showing that $h_2 - h_1 \xrightarrow{*}_G 0$, and thus, $h_1 \downarrow_G h_2$ by Lemma 2.4.28.

Case 2 $\mathrm{lm}(g_2)$ and $\mathrm{lm}(g_1)$ overlap in W, that is, $\mathrm{lm}(g_1)a = b\mathrm{lm}(g_2)$. Then there exists the overlap ambiguity $\mathfrak{a} = (1\otimes a, b\otimes 1, g_1, g_2) \in \mathrm{amb}(G)$. The same reasoning as in the previous case shows that $h_2 - h_1 = a_1\text{S-Pol}(\mathfrak{a})b_2 \in I_{G,a_1\mathrm{lm}(g_1)b_2} = I_{G,W}$, and thus, $h_2 - h_1 \xrightarrow{*}_G 0$.

Case 3 $\mathrm{lm}(g_1)$ is fully contained in a_2, that is, $W = a_1\mathrm{lm}(g_1)w\mathrm{lm}(g_2)b_2$ for some $w \in \langle X\rangle$. Then

$$h_1 = W - a_1g_1b_1 = -a_1\mathrm{tail}(g_1)w\mathrm{lm}(g_2)b_2.$$

Note that all terms of h_1 are $\prec W$, and thus contained in $K\langle X\rangle_{\mathrm{un,G}}$. So Lemma 2.4.44 yields that h_1 has the unique normal form $h_1{\downarrow_G} = (a_1\mathrm{tail}(g_1)w\mathrm{tail}(g_2)b_2){\downarrow_G}$. Analogously, we see that $h_2{\downarrow_G} = (a_1\mathrm{tail}(g_1)w\mathrm{tail}(g_2)b_2){\downarrow_G}$, showing that $h_1{\downarrow_G} = h_2{\downarrow_G}$. □

Resolvability relative to $\preceq$ is interesting from a theoretical point of view as it is the weakest notion that allows to characterise Gröbner bases. In practical applications, however, it is hard to verify algorithmically. Resolvability as defined in Definition 2.4.50, on the other hand, can be checked by reducing the S-polynomial to (a not necessarily unique) normal form. If this normal form is zero, the ambiguity is resolvable. Thus, Theorem 2.4.54 finally provides an algorithmic test to verify whether a set G is a Gröbner basis: Check if all ambiguities of G are resolvable by reducing the corresponding S-polynomials to normal form.

Example 2.4.55. *Let K be any field and consider $I = (xyx - xy) \trianglelefteq K\langle x, y\rangle$. We use $\preceq_{\mathrm{deglex}}$ as a monomial order with $x \prec_{\mathrm{lex}} y$ and show that the reduced Gröbner basis of I is given by*

$$G = \{xy^nx - xy^n \mid n \in \mathbb{N}_{>0}\}.$$

Clearly, G is interreduced and all its elements are monic. Furthermore, for $g_n = xy^nx - xy^n$, we have

$$g_{n+1} = xyg_n + g_1y^n(1 - x)$$

and since $g_1 \in I$, this yields inductively $G \subseteq I$.

It remains to check that G is a Gröbner basis of I. Using the diamond lemma, we can do this. Note that all ambiguities of G are overlap ambiguities of the form

$$\mathfrak{a}_{m,n} = (1 \otimes y^n x, xy^m \otimes 1, g_m, g_n),$$

for $m, n \in \mathbb{N}_{>0}$. The corresponding S-polynomials are

$$\text{S-Pol}(\mathfrak{a}_{m,n}) = xy^{m+n}x - xy^m xy^n,$$

which can be reduced to zero as follows

$$xy^{m+n}x - xy^m xy^n \ \to_{1,g_m,y^n}\ xy^{m+n}x - xy^{m+n} \ \to_{1,g_{m+n},1}\ 0.$$

This shows that all ambiguities of G are resolvable, and consequently that G is a Gröbner basis of I.

Remark 2.4.56. *By introducing a new indeterminate z that abbreviates the product xy and by adding the corresponding generator $z - xy$, one can extend the ideal I from Example 2.4.55 to a new (in some sense equivalent) ideal I' in $K\langle x, y, z\rangle$, which has a finite Gröbner basis. We note that the same observation has also been made in [KN85]. More precisely, using $\preceq_{\text{deglex}}$ with $z \prec_{\text{lex}} x \prec_{\text{lex}} y$, the ideal $I' = (xyx - xy, z - xy) \trianglelefteq K\langle x, y, z\rangle$ has the finite reduced Gröbner basis*

$$G' = \{zx - z, xy - z, zy - z^2\}.$$

Using this finite Gröbner basis, one can solve the ideal membership problem (Problem 2.2.28) for I, because, for any $f \in K\langle x, y\rangle$, we have $f \in I$ if and only if $f \in I'$.

It is clear, however, that this trick cannot work for all finitely generated ideals, as otherwise the word problem would be decidable. It would be interesting to find conditions on the generators of an ideal I for this technique to work, allowing to solve the ideal membership problem for I by finding a finite Gröbner basis of an equivalent ideal I'. We note that we are not aware of any such conditions.

Note that, if an ambiguity $\mathfrak{a}$ of a set G is not resolvable, we can make it resolvable by adding the nonzero normal form of S-Pol($\mathfrak{a}$) to G. Then, to test if the enlarged set is

a Gröbner basis, we, of course, also have to check whether all ambiguities involving the newly added element are resolvable. If all ambiguities of the enlarged set, including those involving the newly added element, are resolvable, then we have successfully computed a Gröbner basis. If not, then we choose an unresolvable ambiguity and repeat this procedure. These instructions basically describe Buchberger's algorithm in the free algebra, given formally in Algorithm 1. We note that the algorithm contains the notion of a *fair* selection strategy, which is defined below in Definition 2.4.57.

Algorithm 1: Buchberger's algorithm

Input: $f_1, \dots, f_r \in K\langle X\rangle$ generating an ideal $I = (f_1, \dots, f_r)$
Output (if the algorithm terminates): $G \subseteq I$ a Gröbner basis of I

```
1 G ← {f_1, ..., f_r};
2 amb ← amb(G);
3 while amb ≠ ∅ :
4 |   select an ambiguity 𝔞 ∈ amb using a fair strategy and remove it;
5 |   f' ← result of reducing S-Pol(𝔞) with respect to →_G;
6 |   if f' ≠ 0 :
7 |   |   G ← G ∪ {f'};
8 |   |   amb ← amb ∪ ⋃_{g∈G} amb(f', g);
9 return G;
```

The noncommutative version of Buchberger's algorithm is essentially identical to the commutative one, with one crucial difference: termination is not guaranteed. Since $K\langle X\rangle$ contains finitely generated ideals that do not admit a finite Gröbner basis (see Example 2.4.55), we cannot expect to obtain a terminating algorithm. Instead, we have to content ourselves with an enumeration procedure satisfying:

- termination if and only if the input ideal I admits a finite Gröbner basis (with respect to the chosen monomial order) and returning such a finite Gröbner basis;
- otherwise, enumeration of an infinite sequence $(g_n)_{n\in\mathbb{N}}$ such that the infinite set $\{g_n \mid n \in \mathbb{N}\}$ forms a Gröbner basis of I;

To ensure this behaviour, the selection strategy used in line 4 is crucial. While in the commutative case, the choice of the selection strategy can have a huge impact on the performance of the algorithm, it has no influence on its correctness. In the noncommutative

case, this is different: only a *fair* selection strategy, selecting every ambiguity that is formed eventually, ensures correctness of the algorithm.

Definition 2.4.57. *A selection strategy, choosing from an ever-changing set S an element at a time, is called* fair *if every element that is present in S at some point is selected eventually.*

Example 2.4.58. *Important examples of fair selection strategies are:*

- *A first-in first-out selection strategy, always choosing the ambiguity that has been in the set* `amb` *the longest, is fair.*
- *More generally, a selection strategy that chooses the elements in* `amb` *in generations, always selecting all ambiguities from one generation before proceeding to the next one, is fair.*
- *A monomial order $\preceq$ is called* fair *if, for any $w \in \langle X \rangle$, the set $\{w' \in \langle X \rangle \mid w' \prec w\}$ is finite. A selection strategy that always chooses $\mathfrak{a} \in$ `amb` with $\mathrm{LM}(\mathfrak{a})$ minimal is fair if the used monomial order is fair.*

Using a non-fair selection strategy in Algorithm 1 can have drastic consequences, resulting in the algorithm running indefinitely in the worst case, regardless of whether the ideal possesses a finite Gröbner basis (under the chosen monomial order). Furthermore, in such cases, the infinite set produced by the algorithm fails to be a Gröbner basis, see also Example 47.6.27 and the subsequent discussion in [Mor16]. For a more in-depth discussion on fair selection strategies, we refer to [Mor16, Sec. 47.6.3]. However, if Algorithm 1 is used as described, employing a fair selection strategy, then it is a correct enumeration procedure for Gröbner bases.

Theorem 2.4.59. *Let $f_1, \ldots, f_r \in K\langle X \rangle$ and, for $n \in \mathbb{N}$, let G_n be the value of G in Algorithm 1 after n iterations of the "*while*" loop given $f_1, \ldots, f_r$ as input. Then $G = \bigcup_{n \in \mathbb{N}} G_n$ is a Gröbner basis of $I = (f_1, \ldots, f_r)$. In this sense, Algorithm 1 enumerates a Gröbner basis G of I.*

Proof. First, note that $(G) = I$ because $f_1, \ldots, f_r \in G$ and all normal forms f' which are added to G are elements of I by Lemma 2.4.29 and 2.4.51. With Theorem 2.4.54, it remains to show that all ambiguities of G are resolvable. To this end, let $\mathfrak{a} \in \mathrm{amb}(G)$ be an ambiguity of G. Since the selection strategy in line 4 is fair, the ambiguity $\mathfrak{a}$ is chosen eventually from `amb`, say, at the $(n+1)$th iteration. Then its S-polynomial is reduced with respect to $\to_{G_n}$. If the normal form f' of this reduction is zero, $\mathfrak{a}$ is resolvable with respect to G_n, and thus also with respect to G since $G_n \subseteq G$. Otherwise, the normal form is added to G_n, so that $G_{n+1} = G_n \cup \{f'\}$, and $\mathfrak{a}$ becomes resolvable with respect to G_{n+1}, implying again that $\mathfrak{a}$ is also resolvable with respect to G since $G_{n+1} \subseteq G$. □

We call the sets G_n produced as intermediate results in Algorithm 1 *partial Gröbner bases* of the ideal I. We can also show that Algorithm 1 terminates whenever the ideal admits a finite Gröbner basis.

Proposition 2.4.60. *Let $f_1, \ldots, f_r \in K\langle X\rangle$. Algorithm 1 terminates given $f_1, \ldots, f_r$ as input if and only if the ideal $I = (f_1, \ldots, f_r)$ admits a finite Gröbner basis (with respect to the chosen monomial order). If the algorithm terminates, the output is a finite Gröbner basis of I.*

Proof. The "only if"-direction of the statement follows immediately from Theorem 2.4.59. This also implies the last statement. For the "if"-direction, let $H = \{h_1, \ldots, h_d\} \subseteq I$ be a finite Gröbner basis of I. Furthermore, for $n \in \mathbb{N}$, let G_n be as in Theorem 2.4.59. Then $G = \bigcup_{n\in\mathbb{N}} G_n$ is also a Gröbner basis of I. Therefore, for every $i = 1, \ldots, d$, there exists $g_i \in G$ such that $\mathrm{lm}(g_i)$ divides $\mathrm{lm}(h_i)$. But then the finite set $G' = \{g_1, \ldots, g_d\} \subseteq G$ satisfies condition 4 of Theorem 2.4.37 and is consequently also a Gröbner basis of I. Since $g_i \in G_{n_i}$ for some $n_i \in \mathbb{N}$, it follows that G_N with $N = \max_i n_i$ is a Gröbner basis of I. Note that G_N can be computed in finite time by Algorithm 1. Furthermore, the set `amb` in Algorithm 1 is finite at all times, containing after N iterations, say, M ambiguities. Since all the respective S-polynomials will reduce to zero with respect to $\to_{G_N}$ by Theorem 2.4.37, neither G_N nor `amb` will be enlarged anymore. Thus, after M more iterations `amb` becomes empty and the algorithm returns the finite Gröbner basis $G_N = G_{N+M}$. □

In general, it is undecidable whether an ideal $(f_1, \ldots, f_r)$ admits a finite Gröbner basis, and thus, whether Algorithm 1 terminates given $f_1, \ldots, f_r$ as input. In practice, we therefore

add additional constraints to Algorithm 1 to guarantee termination. One way of doing this is by keeping track of the number of iterations of the **while** loop and stopping the algorithm after a prescribed number of iterations. Another way is to impose an upper bound on the degree of the leading monomials of ambiguities that are considered. If, during the execution of the algorithm, an ambiguity arises whose leading monomial has a larger degree than the designated bound, this ambiguity is discarded. Both approaches ensure termination but only yield a partial Gröbner basis of I. This might even be the case if I has a finite Gröbner basis but we stop the algorithm too early or impose a degree bound which is too low.

Thus, even though Gröbner bases theoretically solve the ideal membership problem in $K\langle X\rangle$, in practice we still face the problem that we cannot always obtain a complete (possibly infinite) Gröbner basis. This causes the ideal membership problem to be undecidable in general. More precisely, Algorithm 1 renders the ideal membership problem in $K\langle X\rangle$ semi-decidable. This is a consequence of the following result.

Lemma 2.4.61. *Let $f, f_1, \ldots, f_r \in K\langle X\rangle$. If $f \in (f_1, \ldots, f_r)$, then this fact can be verified in finite time.*

Proof. If $f \in I = (f_1, \ldots, f_r)$, then f can be reduced to zero by any complete (possibly infinite) Gröbner basis of I. For this reduction to zero, however, only finitely many elements are needed. Thus, we can use Algorithm 1 to enumerate a Gröbner basis G of I, and whenever a new element is added to the set G, we compute a normal form of f with respect to $\to_G$. Because $f \in I$ this will eventually yield zero, verifying the ideal membership. □

Corollary 2.4.62. *The ideal membership problem (Problem 2.2.28) in $K\langle X\rangle$ is semi-decidable.*

2.4.4 Gröbner bases of right ideals

A theory of Gröbner bases for one-sided ideals can be developed analogously to the two-sided case. In this section, we recall the most important results of this theory for right ideals. As noted before, all results also apply symmetrically to left ideals. Like in the previous section, K is a field, $X = \{x_1, \ldots, x_n\}$ is a finite set of indeterminates, and $\preceq$ is a monomial order on $\langle X\rangle$.

Remark 2.4.63. *When dealing with right ideals instead of two-sided ideals, the compatibility condition of a monomial order can be weakened to right compatibility with the multiplication in $\langle X \rangle$, so that $w \preceq w'$ implies $wb \preceq w'b$ for all $b, w, w' \in \langle X \rangle$.*

To deal with right ideals, we adapt the definition of polynomial reduction from Definition 2.4.21.

Definition 2.4.64. *Let $b \in \langle X \rangle$, $g \in K\langle X \rangle \setminus \{0\}$, and $G \subseteq K\langle X \rangle$. We define the following reduction relations on $K\langle X \rangle$:*

$$\begin{aligned} \to_{\rho,g,b} &:= \to_{1,g,b} \\ \to_{\rho,g} &:= \bigcup_{b \in \langle X \rangle} \to_{\rho,g,b} \\ \to_{\rho,G} &:= \bigcup_{g \in G \setminus \{0\}} \to_{\rho,g} \end{aligned}$$

These relations are called the polynomial right reduction relation *with respect to g, b, with respect to g, and with respect to G respectively.*

By replacing two-sided ideals by right ideals, all results about polynomial reduction carry over to polynomial right reduction. In particular, $\to_{\rho,G}$ is terminating and we have the following result relating right reduction to ideal membership in right ideals.

Lemma 2.4.65. *If $f \to_{\rho,G} f'$, then $f - f' \in (G)_\rho$. Moreover, if $\to_{\rho,G}$ is confluent, then $f \in (G)_\rho$ if and only if $f \overset{*}{\to}_{\rho,G} 0$.*

Gröbner bases of right ideals can be defined analogous to the two-sided case.

Definition 2.4.66. *Let $I_\rho \trianglelefteq_r K\langle X \rangle$. A subset $G \subseteq I_\rho$ is a* right Gröbner basis *of I if $(G)_\rho = I_\rho$ and $\to_{\rho,G}$ is confluent.*

Also Theorem 2.4.37 carries over to the one-sided case, with an analogous proof.

Theorem 2.4.67. *Let $I_\rho \trianglelefteq_r K\langle X \rangle$ and $G \subseteq I_\rho$. The following conditions are equivalent:*

1. *G is a right Gröbner basis of I_ρ;*

2. $f \xrightarrow{*}_{\rho,G} 0$ *for all* $f \in I_\rho$;

3. $(\mathrm{lm}(I_\rho))_\rho = (\mathrm{lm}(G))_\rho$;

4. *for all nonzero* $f \in I_\rho$, *there exists* $g \in G$ *such that* $\mathrm{lm}(g)$ *is a prefix of* $\mathrm{lm}(f)$;

Many properties of two-sided Gröbner bases also have a one-sided analogue. In particular, the elimination property also holds for right ideals and right Gröbner bases. Again, the proof is analogous to the two-sided case (see Theorem 2.4.43).

Proposition 2.4.68. *Let* $I_\rho \trianglelefteq_r K\langle X, Y\rangle$ *and let* G *be a right Gröbner basis of* I_ρ *with respect to an elimination order for* Y. *Then* $G \cap K\langle X\rangle$ *is a right Gröbner basis of the elimination ideal* $I_\rho \cap K\langle X\rangle$.

We shall see that right Gröbner bases are computationally a lot simpler than two-sided Gröbner bases. To this end, we first introduce the notion of a *right interreduced* set and a reduced right Gröbner basis.

Definition 2.4.69. *A set* $G \subseteq K\langle X\rangle$ *is* right interreduced *if every* $g \in G$ *is irreducible with respect to* $\rightarrow_{\rho,G\setminus\{g\}}$.

Definition 2.4.70. *A right Gröbner basis* $G \subseteq I_\rho$ *of a right ideal* $I_\rho \trianglelefteq_r K\langle X\rangle$ *is* reduced *if* G *is right interreduced and all polynomials in* G *are monic.*

Existence and uniqueness of reduced right Gröbner bases can be shown just like in the two-sided case. However, in contrast to two-sided ideals, we can easily obtain the reduced right Gröbner basis of a finitely generated right ideal from any finite generating set. This is a consequence of the following two results.

Proposition 2.4.71. *Let* $G \subseteq K\langle X\rangle$ *be a right interreduced set that consists only of monic polynomials. Then* G *is the reduced right Gröbner basis of* $(G)_\rho$.

Proof. We only have to show that G is a Gröbner basis. We do this by proving condition 4 of Theorem 2.4.67. To this end, let $f \in (G)_\rho$ be nonzero. Then f can be written as $f = \sum_{i=1}^{d} c_i g_i b_i$ with nonzero $c_i \in K$, $b_i \in \langle X \rangle$, and $g_i \in G$. Because G is interreduced, no $\mathrm{lm}(g_i)$ is a prefix of $\mathrm{lm}(g_j)$ for $i \neq j$. Therefore, $\mathrm{lm}(g_i b_i) \neq \mathrm{lm}(g_j b_j)$ for all $i \neq j$, and thus, by Lemma 2.4.19, $\mathrm{lm}(f) = \max_i \mathrm{lm}(g_i b_i)$, showing that $\mathrm{lm}(g_j)$ with $j = \operatorname{argmax}_i \mathrm{lm}(g_i b_i)$ is a prefix of $\mathrm{lm}(f)$. □

The following lemma follows from [Xiu12, Thm. 3.2.8].

Lemma 2.4.72. *Let $G \subseteq K\langle X \rangle$ be finite. There exists an algorithm to turn G into a finite (right) interreduced set G' such that $(G) = (G')$ (resp. $(G)_\rho = (G')_\rho$).*

Consequently, we see that every finitely generated right ideal I_ρ has a finite right Gröbner basis. In particular, the reduced right Gröbner basis of I_ρ can be computed by right interreducing any finite generating set of I_ρ. Since right Gröbner bases allow to decide ideal membership for right ideals, we arrive at the following result.

Corollary 2.4.73. *The ideal membership problem for finitely generated right ideals in $K\langle X \rangle$ is decidable.*

To end this section, we mention a different approach for computing right Gröbner bases that relies on Gröbner bases for two-sided ideals. This approach is described in [Hey01] and has the advantage that the (more commonly implemented) algorithms for two-sided ideals (like Algorithm 1) can be reused without any adaptations. The following result follows from [Hey01, Alg. 4.9] and the discussion afterwards.

Proposition 2.4.74. *Let $f_1, \ldots, f_r \in K\langle X \rangle$ and let $y \notin X$ be a new indeterminate. Let G be a Gröbner basis of the two-sided ideal $(yf_1, \ldots, yf_r) \subseteq K\langle X, y \rangle$. Then the set $\{g \mid yg \in G\}$ is a right Gröbner basis of the right ideal $(f_1, \ldots, f_r)_\rho \subseteq K\langle X \rangle$.*

More generally, the approach described in [Hey01] allows to compute right Gröbner bases of right ideals in quotients $K\langle X \rangle / I$ of the free algebra by a two-sided ideal I.

2.5 Many-sorted logic

Classical first-order logic is an established logic, satisfying many important theorems such as the Löwenheim-Skolem theorem [EFT21, Thm. VI.1.1] or the compactness theorem [EFT21, Thm. VI.2.1]. It also possesses a sound and complete calculus, see, for example, [EFT21, Ch. IV]. Classical first-order logic can be considered unsorted, as the structures used only consist of a single universe or domain of objects. In mathematics or computer science, however, we often formalise statements involving different kinds of objects. The goal of *many-sorted first-order logic* is to provide a natural setting for modelling such situations.

In the following sections, we recall the syntax and semantics of many-sorted first-order logic, and we present a sound and complete calculus for it. Our main reference for this part is [Man93], but we also include notation from [EFT21].

2.5.1 Syntax

The syntax of a logic describes the allowed symbols of the language and how they can be composed to construct (well-formed) terms and formulas. In case of many-sorted first-order logic, we fix the following basic sets that form our alphabet. We assume that all these sets are pairwise disjoint.

We fix an enumerable set $\mathbf{Sort} = \{s_1, s_2, \dots\}$ of *sorts*. Each sort can describe one type of object we want to model. We also fix enumerable sets $\mathbf{Con} = \{c_1, c_2, \dots\}$ and $\mathbf{Fun} = \{f_1, f_2, \dots\}$ of *constant symbols* and *function symbols* respectively. These symbols allow to model dedicated constants in structures as well as functions on them. Note that we do not fix any predicate symbols as we will only work with one special predicate symbol $\approx$ for equality, which we will interpret as identity. For a description of many-sorted first-order logic including arbitrary predicate symbols, we refer to [Man93]. Finally, we fix an enumerable set $\mathbf{Var} = \{x_1, x_2, \dots\}$ of *variables*.

A *signature* lists and describes the non-logical symbols of a logic that are relevant in a particular context. In many-sorted logic, a signature is defined as follows.

Definition 2.5.1. *A* signature *is a tuple* $\Sigma = (S, C, F, \sigma)$ *consisting of*

1. *a set of sorts* $S \subseteq \mathbf{Sort}$*;*

2. *a set of constant symbols* $C \subseteq \mathbf{Con}$*;*
3. *a set of function symbols* $F \subseteq \mathbf{Fun}$*;*
4. *a* sort function $\sigma\colon \mathbf{Var} \cup C \cup F \to \bigcup_{n\geq 1} S^n$ *satisfying the following conditions:*
   a) $\sigma(x) \in S$ *for all* $x \in \mathbf{Var} \cup C$*;*
   b) $\sigma(f) \in \bigcup_{n>1} S^n$ *for all* $f \in F$*;*

The set S defines the relevant sorts of the signature. The sets C and F determine which constant and function symbols the signature contains. The sort function σ assigns to every variable and to all relevant symbols their *sort*. Conditions (4a) and (4b) ensure that variables and constant symbols get mapped to a single sort $s \in S$, while function symbols get mapped to a tuple of sorts $(s_1, \ldots, s_n, s) \in S^{n+1}$.

Remark 2.5.2. *For a function symbol* f*, we write* $\sigma(f) = s_1 \times \cdots \times s_n \to s$ *instead of* $\sigma(f) = (s_1, \ldots, s_n, s)$*. The quantity* n *is called the* arity *of* f*.*

In the following, we fix a signature $\Sigma = (S, C, F, \sigma)$. Within this signature, we can construct *terms* and *formulas*.

Definition 2.5.3. *The set* $\mathbf{Term}(\Sigma, s)$ *of* terms of sort $s \in S$ *is the smallest set satisfying the following conditions:*

1. $\{x \in \mathbf{Var} \cup C \mid \sigma(x) = s\} \subseteq \mathbf{Term}(\Sigma, s)$*;*
2. *if* $f \in F$ *with* $\sigma(f) = s_1 \times \cdots \times s_n \to s$ *and* $t_i \in \mathbf{Term}(\Sigma, s_i)$ *for* $i = 1, \ldots, n$*, then* $f(t_1, \ldots, t_n) \in \mathbf{Term}(\Sigma, s)$*;*

For a term $t \in \mathbf{Term}(\Sigma, s)$*, its* sort *is* $\sigma(t) \coloneqq s$*. The set* $\mathbf{Term}(\Sigma)$ *of* terms *is given by*

$$\mathbf{Term}(\Sigma) \coloneqq \bigcup_{s \in S} \mathbf{Term}(\Sigma, s).$$

A term is called *ground* if it does not contain any variables. The set of all ground terms is denoted by $\mathbf{Ground}(\Sigma)$. Note that this set is countable.

Example 2.5.4. *Consider the signature* $\Sigma = (S, C, F, \sigma)$ *with* $S = \{s_1, s_2\}$, $C = \{c, d\}$, $F = \{f, g\}$, *and* σ *such that*

$$\sigma(c) = s_1, \qquad \sigma(d) = s_2, \qquad \sigma(f) = s_1 \times s_1 \to s_2, \qquad \sigma(g) = s_2 \to s_1$$

Furthermore, let $x, y \in \mathbf{Var}$ *with* $\sigma(x) = s_1$ *and* $\sigma(y) = s_2$.

Within this signature, the expression $f(x, y)$ *is* not *a well-formed term because* y *has the wrong sort. Examples of well-formed terms are all the basic symbols* c, d, x, y, *but also* $f(x, x)$ *or* $g(f(g(d), c))$. *Note that* c, d *as well as* $g(f(g(d), c)$ *are ground terms, while* $f(x, x)$ *is not.*

Definition 2.5.5. *The set* $\mathbf{Form}(\Sigma)$ *of* formulas *is the smallest set satisfying the following conditions:*

1. *if* $t, t' \in \mathbf{Term}(\Sigma, s)$ *for some* $s \in S$, *then* $t \approx t' \in \mathbf{Form}(\Sigma)$;
2. *if* $\varphi \in \mathbf{Form}(\Sigma)$, *then* $\neg\varphi \in \mathbf{Form}(\Sigma)$;
3. *if* $\varphi, \psi \in \mathbf{Form}(\Sigma)$, *then* $(\varphi * \psi) \in \mathbf{Form}(\Sigma)$ *with* $* \in \{\vee, \wedge, \to\}$;
4. *if* $\varphi \in \mathbf{Form}(\Sigma)$ *and* $x \in \mathbf{Var}$, *then* $Qx : \varphi \in \mathbf{Form}(\Sigma)$ *with* $Q \in \{\exists, \forall\}$;

For convenience in notation, we follow the common convention to drop the outermost parentheses of formulas and we assume the following precedence of the logical operators: $\neg, \vee, \wedge, \to, \exists, \forall$. Furthermore, we abbreviate a block of consecutive equally quantified variables $Qx_1 Qx_2 \dots Qx_k$ with $Q \in \{\exists, \forall\}$ by $Qx_1, x_2, \dots, x_k$, or simply by $Q\boldsymbol{x}$. Furthermore, to indicate the scope of a quantifier, we also write $Q\boldsymbol{x} : \varphi(\boldsymbol{x})$.

We recall some standard definitions from (many-sorted) first-order logic. Formulas constructed in part 1 of Definition 2.5.5 are called *atomic formulas*. A *literal* is an atomic formula or its negation. In the following, we write $t \not\approx t'$ for the literal $\neg(t \approx t')$. In the last part of Definition 2.5.5, Q is called the *quantifier* of x and φ is the *scope* of Qx. Any occurrence of the variable x in the scope φ is called *bound*. All non-quantified occurrences of a variable, that is, all occurrences which are not in the scope of a quantifier, are called *free* occurrences. A variable x is *bound* (resp. *free*) in a formula φ if there is a bound (resp. free) occurrence of x in φ. We denote the set of all free variables in a formula φ by $\mathrm{Free}(\varphi)$. If $\mathrm{Free}(\varphi) = \emptyset$, then φ is a *sentence*. The set of all sentences is denoted by

Sent(Σ). Furthermore, a formula without any quantifiers is called *quantifier-free*, and a quantifier-free sentence is called *ground*. Note that a ground sentence does not contain any variables.

Example 2.5.6. *We reconsider the signature* $\Sigma = (S, C, F, \sigma)$ *from Example 2.5.4, that is,* $S = \{s_1, s_2\}$, $C = \{c, d\}$, $F = \{f, g\}$, *and* σ *such that*

$$\sigma(c) = s_1, \qquad \sigma(d) = s_2, \qquad \sigma(f) = s_1 \times s_1 \to s_2, \qquad \sigma(g) = s_2 \to s_1$$

Furthermore, we let $x, y \in \mathbf{Var}$ *with* $\sigma(x) = s_1$ *and* $\sigma(y) = s_2$.

Within this signature, the expression $f(x, x) \approx c$ *is* not *a well-formed formula because* $f(x, x)$ *and* c *have different sorts. Examples of well-formed formulas are* $f(x, x) \approx f(g(d), c)$ *or* $\forall x : (x \not\approx c \vee \exists y : g(y) \approx c)$. *The first formula is a quantifier-free atomic formula. Note that it contains the free variable* x, *and is therefore not a sentence. The second formula is a sentence because both variables that appear in it are bound, however it is not ground as it is not quantifier-free. An example of a ground sentence is* $c \approx g(d) \to f(c, g(d)) \approx f(g(d), c)$.

We denote by $\varphi[x \mapsto t]$ the substitution of the variable x by the term t in the formula φ. Such a substitution is only possible if x and t are of the same sort. The result of this substitution is the formula obtained by the replacement of all free occurrences of x by t, with renaming, if necessary, those bound variables in φ that coincide with a variable appearing in t. More precisely, we have the following formal definition of the substitution process, which we first consider for terms and then extend to formulas.

Definition 2.5.7. *Let* $x \in \mathbf{Var}$ *and* $t \in \mathbf{Term}(\Sigma)$ *such that* $\sigma(x) = \sigma(t)$. *The* substitution *of* x *by* t *in a term is recursively defined as follows, where* $y \in \mathbf{Var}$, $c \in C$, *and* $f \in F$:

$$\begin{aligned} y[x \mapsto t] &:= \begin{cases} t & \text{if } x = y \\ y & \text{otherwise} \end{cases} \\ c[x \mapsto t] &:= c \\ f(t_1, \ldots, t_n)[x \mapsto t] &:= f(t_1[x \mapsto t], \ldots, t_n[x \mapsto t]) \end{aligned}$$

The substitution *of x by t in a formula is recursively defined as:*

$$
\begin{aligned}
(s \approx s')[x \mapsto t] &\coloneqq s[x \mapsto t] \approx s'[x \mapsto t] \\
(\neg\varphi)[x \mapsto t] &\coloneqq \neg(\varphi[x \mapsto t]) \\
(\varphi * \psi)[x \mapsto t] &\coloneqq \varphi[x \mapsto t] * \psi[x \mapsto t] \\
(Qy : \varphi)[x \mapsto t] &\coloneqq \begin{cases} Qy : \varphi & \text{if } x \notin \mathrm{Free}(Qy : \varphi) \\ Qy : \varphi[x \mapsto t] & \text{if } x \in \mathrm{Free}(Qy : \varphi), y \notin \mathrm{Free}(t) \\ Qz : (\varphi[y \mapsto z])[x \mapsto t] & \text{if } x \in \mathrm{Free}(Qy : \varphi), y \in \mathrm{Free}(t), \\ & \text{and } z \text{ is a new variable with} \\ & \sigma(z) = \sigma(y) \end{cases}
\end{aligned}
$$

Here, $t_1, \ldots, t_n, s, s' \in \mathbf{Term}(\Sigma)$, $\varphi, \psi \in \mathbf{Form}(\Sigma)$, $* \in \{\vee, \wedge, \to\}$, and $Q \in \{\exists, \forall\}$.

Example 2.5.8. *Consider a signature Σ containing a function symbol $f : s_1 \times s_2 \to s_1$. Furthermore, let $x, y, z \in \mathbf{Var}$ be such that $\sigma(x) = \sigma(y) = s_1$ and $\sigma(z) = s_2$. We have, for example,*

1. $(x \approx f(y,z))[x \mapsto f(x,z)] = f(x,z) \approx f(y,z)$;
2. $(\exists x : x \approx f(y,z))\,[x \mapsto f(x,z)] = \exists x : x \approx f(y,z)$;
3. $(\exists y : x \approx f(y,z))\,[x \mapsto f(x,z)] = \exists y : f(x,z) \approx f(y,z)$;
4. $(\exists z : x \approx f(y,z))\,[x \mapsto f(x,z)] = \exists z' : f(x,z) \approx f(y,z')$, *with z' such that* $\sigma(z') = \sigma(z) = s_2$;

2.5.2 Semantics

In ordinary first-order logic, formulas are assigned truth values by an *interpretation* in a universe A. In many-sorted logic, we no longer have a single universe but a dedicated domain A_s for each sort $s \in S$. These domains, together with constants and functions that interpret the constant and function symbols, form a *structure*. Recall that we fixed a signature $\Sigma = (S, C, F, \sigma)$.

Definition 2.5.9. *A* structure *A of signature Σ is a tuple*

$$A = \Big((A_s)_{s\in S}, (c^A)_{c\in C}, (f^A)_{f\in F}\Big)$$

satisfying the following conditions:

1. *$(A_s)_{s\in S}$ is a family of nonempty sets;*
2. *$(c^A)_{c\in C}$ is a family of elements such that $\sigma(c) = s$ implies $c^A \in A_s$;*
3. *$(f^A)_{f\in F}$ is a family of functions such that $\sigma(f) = s_1 \times \cdots \times s_n \to s$ implies*

$$f^A \colon A_{s_1} \times \cdots \times A_{s_n} \to A_s;$$

Given a structure A of signature Σ, an *assignment* on A is a function

$$\mathfrak{a} \colon \mathbf{Var} \to \bigcup_{s\in S} A_s$$

such that $\mathfrak{a}(x) \in A_s$ if and only if $\sigma(x) = s$ for all $x \in \mathbf{Var}$.

Definition 2.5.10. *Let A be a structure of signature Σ and let $\mathfrak{a}$ be an assignment on A. The pair $\mathcal{I} = (A, \mathfrak{a})$ is called an* interpretation.

An interpretation allows to map terms to elements in the different universes. More precisely, the interpretation $\mathcal{I}(t)$ of a term $t \in \mathbf{Term}(\Sigma)$ is defined as

$$\mathcal{I}(t) \coloneqq \begin{cases} \mathfrak{a}(x) & \text{if } t = x \in \mathbf{Var} \\ c^A & \text{if } t = c \in C \\ f^A(\mathcal{I}(t_1), \ldots, \mathcal{I}(t_n)) & \text{if } t = f(t_1, \ldots, t_n) \text{ with } f \in F \end{cases}$$

Note that all expressions above are well-defined. In particular, if t is of sort s, then $\mathcal{I}(t)$ is an element in A_s.

Using interpretations of terms, we can interpret formulas. To do this, we introduce the following notation. If $\mathfrak{a}$ is an assignment on a structure A, $x \in \mathbf{Var}$ is a variable of sort s and $a \in A_s$, then $\mathfrak{a}[x \mapsto a]$ denotes the function

$$\mathfrak{a}[x \mapsto a] \colon \mathbf{Var} \to \bigcup_{s \in S} A_s,$$

that maps every $y \neq x$ to $\mathfrak{a}(y)$ and x to a. This function is clearly also an assignment on A. Furthermore, for $\mathcal{I} = (A, \mathfrak{a})$, let $\mathcal{I}[x \mapsto a] = (A, \mathfrak{a}[x \mapsto a])$.

With this, the interpretation of a formula $\varphi \in \mathbf{Form}(\Sigma)$ is the truth value $\mathcal{I}(\varphi) \in \{\top, \bot\}$ defined as follows:

$$\begin{aligned}
\mathcal{I}(t \approx t') &:= \top \quad \text{iff} \quad \mathcal{I}(t) = \mathcal{I}(t') \text{ as elements in } A_{\sigma(t)} \\
\mathcal{I}(\neg\varphi) &:= \top \quad \text{iff} \quad \mathcal{I}(\varphi) = \bot \\
\mathcal{I}(\varphi \vee \psi) &:= \top \quad \text{iff} \quad \mathcal{I}(\varphi) = \top \text{ or } \mathcal{I}(\psi) = \top \\
\mathcal{I}(\varphi \wedge \psi) &:= \top \quad \text{iff} \quad \mathcal{I}(\varphi) = \top \text{ and } \mathcal{I}(\psi) = \top \\
\mathcal{I}(\varphi \to \psi) &:= \top \quad \text{iff} \quad \mathcal{I}(\varphi) = \bot \text{ or } \mathcal{I}(\psi) = \top \\
\mathcal{I}(\forall x : \varphi) &:= \top \quad \text{iff} \quad \text{for all } a \in A_{\sigma(x)} \text{ it holds that } \mathcal{I}[x \mapsto a](\varphi) = \top \\
\mathcal{I}(\exists x : \varphi) &:= \top \quad \text{iff} \quad \text{there exists some } a \in A_{\sigma(x)} \text{ such that } \mathcal{I}[x \mapsto a](\varphi) = \top
\end{aligned}$$

Remark 2.5.11. *Note that the equality symbol $\approx$ is always interpreted as the identity in the structure A. Such structures that interpret equality as identity are called* normal. *We only consider normal structures in this work.*

An interpretation $\mathcal{I}$ is a *model* of a formula φ if $\mathcal{I}(\varphi) = \top$. Furthermore, φ is *valid* (resp. *unsatisfiable*) if every (resp. no) interpretation is a model of φ. Two formulas φ and ψ are *logically equivalent* if they have the same models.

Similarly, an interpretation is a model of a set of formulas $\Phi \subseteq \mathbf{Form}(\Sigma)$ if it is a model of every formula in Φ, and Φ is *valid* (resp. *unsatisfiable*) if every formula in Φ is valid (resp. unsatisfiable).

A formula φ is a *semantic consequence* of a set of formulas Φ, if each model of Φ is also a model of φ. In this case, we write $\Phi \models \varphi$. If φ is not a semantic consequence of Φ, we write $\Phi \not\models \varphi$. Two formulas φ and ψ are Φ-*equivalid* if $\Phi \models \varphi$ if and only if $\Phi \models \psi$.

The following classical result relates the semantic consequence relation to the notion of unsatisfiability. The first equivalence follows immediately from the definitions and the second equivalence is the *Compactness theorem* for many-sorted first-order logic [Man93, Sec. 2.5].

Proposition 2.5.12. *Let* $\Phi \subseteq \mathbf{Form}(\Sigma)$ *and* $\varphi \in \mathbf{Form}(\Sigma)$*. The following are equivalent:*

1. $\Phi \models \varphi$*;*
2. $\Phi \cup \{\neg\varphi\}$ *is unsatisfiable;*
3. *there exist* $\varphi_1, \ldots, \varphi_r \in \Phi$ *such that* $\varphi_1 \wedge \cdots \wedge \varphi_r \wedge \neg\varphi$ *is unsatisfiable;*

2.5.3 Formal computations

In this section, we recall how to validate that a formula φ is a semantic consequence of a set of formulas Φ by performing syntactic operations. Such a syntactic computation is called a *formal computation* and one way to perform it is by applying certain syntactic rules, so-called *sequent rules*, to the formulas at hand. Several sequent rules form a *sequent calculus*.

We recall the sequent calculus $\mathrm{LK}^{=}$ for many-sorted first-order logic with equality. This calculus – for ordinary first-order logic without equality – was first introduced by Gentzen [Gen35] and has subsequently been extended to include equality, see for example [DV01] for further information in the unsorted case. As illustrated in [Man93, Sec. 2], the adaptation to the many-sorted case is straightforward.

We note that there also exist other inference systems for (many-sorted) first-order logic. We focus on the sequent calculus $\mathrm{LK}^{=}$ in this work as this inference system allows a natural translation of formal computations with formulas into polynomial computations. We refer to Chapter 4 for further information.

A *sequent* is an expression of the form $\Gamma \vdash \Delta$ where Γ and Δ are finite (possibly empty) multisets of formulas. A sequent can be considered as an assertion of the form "whenever *all* $\varphi \in \Gamma$ are true, then *at least one* $\psi \in \Delta$ is also true". An expression of the form Γ, φ in a sequent represents the multiset $\Gamma \cup \{\varphi\}$. In particular, $\varphi_1, \ldots, \varphi_m \vdash \psi_1, \ldots, \psi_n$ means $\{\varphi_1, \ldots, \varphi_m\} \vdash \{\psi_1, \ldots, \psi_n\}$.

Sequent rules allow us to pass from one sequent to another. The sequent calculus $\mathrm{LK}^{=}$ comprises the following sequent rules. In the following, α denotes an atomic formula, φ, ψ are arbitrary formulas, and t, t' are terms.

Axioms

$$\text{(Ax)}\quad \frac{}{\Gamma, \alpha \vdash \Delta, \alpha} \qquad \text{(Ref)}\quad \frac{}{\Gamma \vdash \Delta, t \approx t}$$

Structural rules

$$(\mathrm{W}\vdash)\quad \frac{\Gamma \vdash \Delta}{\Gamma, \varphi \vdash \Delta} \qquad (\vdash \mathrm{W})\quad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \varphi}$$

$$(\mathrm{C}\vdash)\quad \frac{\Gamma, \varphi, \varphi \vdash \Delta}{\Gamma, \varphi \vdash \Delta} \qquad (\vdash \mathrm{C})\quad \frac{\Gamma \vdash \Delta, \varphi, \varphi}{\Gamma \vdash \Delta, \varphi}$$

Propositional rules

$$(\neg\vdash)\quad \frac{\Gamma \vdash \Delta, \varphi}{\Gamma, \neg\varphi \vdash \Delta} \qquad (\vdash \neg)\quad \frac{\Gamma, \varphi \vdash \Delta}{\Gamma \vdash \Delta, \neg\varphi}$$

$$(\vee\vdash)\quad \frac{\Gamma, \varphi \vdash \Delta \qquad \Gamma, \psi \vdash \Delta}{\Gamma, \varphi \vee \psi \vdash \Delta} \qquad (\vdash \vee)\quad \frac{\Gamma \vdash \Delta, \varphi, \psi}{\Gamma \vdash \Delta, \varphi \vee \psi}$$

$$(\wedge\vdash)\quad \frac{\Gamma, \varphi, \psi \vdash \Delta}{\Gamma, \varphi \wedge \psi \vdash \Delta} \qquad (\vdash \wedge)\quad \frac{\Gamma \vdash \Delta, \varphi \qquad \Gamma \vdash \Delta, \psi}{\Gamma \vdash \Delta, \varphi \wedge \psi}$$

$$(\to\vdash)\quad \frac{\Gamma \vdash \Delta, \varphi \qquad \Gamma, \psi \vdash \Delta}{\Gamma, \varphi \to \psi \vdash \Delta} \qquad (\vdash \to)\quad \frac{\Gamma, \varphi \vdash \Delta, \psi}{\Gamma \vdash \Delta, \varphi \to \psi}$$

Quantification rules

$$(\exists\ \vdash)\quad \frac{\Gamma, \varphi[x \mapsto y]\ \vdash\ \Delta}{\Gamma, \exists x : \varphi\ \vdash\ \Delta} \text{ if } y \text{ is a new variable with } \sigma(y) = \sigma(x)$$

$$(\vdash\ \exists)\quad \frac{\Gamma\ \vdash\ \Delta, \varphi[x \mapsto t]}{\Gamma\ \vdash\ \Delta, \exists x : \varphi} \text{ if } t \text{ is a term with } \sigma(t) = \sigma(x)$$

$$(\forall\ \vdash)\quad \frac{\Gamma, \varphi[x \mapsto t]\ \vdash\ \Delta}{\Gamma, \forall x : \varphi\ \vdash\ \Delta} \text{ if } t \text{ is a term with } \sigma(t) = \sigma(x)$$

$$(\vdash\ \forall)\quad \frac{\Gamma\ \vdash\ \Delta, \varphi[x \mapsto y]}{\Gamma\ \vdash\ \Delta, \forall x : \varphi} \text{ if } y \text{ is a new variable with } \sigma(y) = \sigma(x)$$

Equational rule

$$(\text{Sub})\quad \frac{\Gamma, \varphi[x \mapsto t'], t \approx t'\ \vdash\ \Delta}{\Gamma, \varphi[x \mapsto t], t \approx t'\ \vdash\ \Delta} \text{ if } \sigma(x) = \sigma(s)$$

A rule without a sequent on top is called an *axiom*. Any statement that can be derived from the axioms using the rules stated above, is *provable by a formal computation* (cf. [EFT21, Def. IV.1.1]).

Definition 2.5.13. *A formula φ is* provable by a formal computation *from a set Φ of formulas if there exists a finite subset $\{\varphi_1, \ldots, \varphi_n\} \subseteq \Phi$ such that the sequent $\varphi_1, \ldots, \varphi_n \vdash \varphi$ is derivable by the rules of the sequent calculus $\mathrm{LK}^{=}$ stated above starting from axioms. In this case, we write $\Phi \vdash \varphi$.*

The following theorem recalls that $\mathrm{LK}^{=}$ is correct and complete. Correctness refers to the fact that every formal computation yields a semantic consequence and completeness states that every semantic consequence is provable by a formal computation. While the former follows easily from the definition of the sequent rules, the latter is essentially the result of *Gödel's completeness theorem* [Göd30], see also [Man93, Sec. 2] for the many-sorted case.

Theorem 2.5.14. *Let $\Phi \subseteq \mathbf{Form}(\Sigma)$ and $\varphi \in \mathbf{Form}(\Sigma)$. Then $\Phi \models \varphi$ if and only if $\Phi \vdash \varphi$.*

3 Noncommutative signature Gröbner bases

For commutative polynomials, the latest generation of Gröbner basis algorithms are so-called signature-based algorithms, heralded by the F5 algorithm [Fau02]. This class of algorithms was the subject of extensive research in the past 20 years, a survey of which can be found in [EF17]. By using the concept of *signatures*, these algorithms compute, in addition to a Gröbner basis, some information on how the polynomials in that basis were computed. Using this information, the algorithms are able to identify relations between the computed polynomials, and use them to predict and avoid reductions to zero, leading to a significant performance improvement. This approach, which improves upon the earlier idea of tracing syzygies [MMT92], has had a drastic impact on the field, leading to advances also in other algorithmic tools such as staggered linear bases [HJ20; HM23], initially introduced in [GM86].

Beyond their original purpose of optimising the algorithms, it was more recently observed that the data of signatures also has direct applications. For instance, they have been used in computational geometry [Ede+23], and they allow to perform a number of operations on the syzygy module of a family of polynomials, without the computational overhead of module Gröbner basis computations. Specifically, the data encoded in signatures allows to reconstruct cofactor representations of the Gröbner basis elements in terms of the input polynomials as well as a Gröbner basis of the input's syzygy module [GVW16].

This potential has led to signature Gröbner bases being generalised beyond commutative polynomials over fields, for instance to polynomial ideals over rings [EPP17; FV21] or to one-sided ideals in solvable noncommutative algebras [Sun+12]. In [Kin14], a noncommutative version of the F5 algorithm for right modules over quotients of path algebras was described and used to efficiently compute bases of Loewy layers. Additionally, recent independent work [CLV21] introduced analogous definitions to those in Section 3.2 for the free algebra and established lower bounds for the complexity of the set of leading monomials of the module of syzygies, in the sense of the Chomsky hierarchy.

In this chapter, we extend the notion of signature Gröbner bases to noncommutative polynomials in the free algebra, and more generally, to *mixed polynomials* that involve both commutative and noncommutative variables. More precisely, we consider the *mixed algebra* $R[X]\langle Y\rangle = R[x_1, \ldots, x_k]\langle y_1, \ldots, y_n\rangle$ over a commutative principal ideal domain R. This structure was first studied in [MZ98] and arises, for example, when introducing auxiliary variables to a free algebra, such as homogenisation variables or tag variables for computing the intersection of ideals or the homogeneous part of an ideal (see Section 5.3.1, 5.3.3). Furthermore, the mixed algebra provides a natural setting for studying many finitely presented structures, such as Iwahori-Hecke algebras [Hum90; LMA23] or (discrete) Heisenberg groups [LS15]. Moreover, by considering coefficient rings instead of fields, computations over $\mathbb{Z}$ become possible, which can be considered universal as they remain valid over any ring.

A theory of classical Gröbner bases in the mixed algebra can be developed analogously to the free algebra. However, since the coefficient domain is now a ring, different notions of Gröbner bases exist, namely weak and strong ones, corresponding to different notions of reductions. Of the two, strong bases and reductions are the most similar to fields and the ones we will focus on in the following. Weak Gröbner bases in the free algebra were studied in [Pri96], for example. For the theory of strong Gröbner bases in the mixed algebra over the ring of integers $\mathbb{Z}$ or a field, we refer to [MZ98].

After formally introducing the mixed algebra and the concept of signatures in Section 3.1, we define signature Gröbner bases in this setting in Section 3.2. To facilitate understanding, we initially focus, for their computation, on the simpler case of signature Gröbner bases in the free algebra over a coefficient field. This approach allows us to become familiar with the relevant notions and algorithms in a less complex setting before transitioning to full generality. Additionally, the free algebra over a coefficient field enjoys significant relevance, justifying a separate discussion of this case.

Therefore, we present in Section 3.3 a generalisation of the commutative signature cover criterion [GVW16] to the free setting. Building upon this criterion, we state Algorithm 3 for computing signature Gröbner bases in the free algebra. We also extend classical signature-based elimination criteria such as the syzygy criterion, the F5 criterion, and the singular criterion, in order to use signatures to enhance the efficiency of the algorithms. Moreover, we show how, starting from a signature Gröbner basis, cofactor representations of the basis elements can be reconstructed, and how a basis of the syzygy module of the

generators can be obtained. We note that, while the results of this section are published in [HV22], the presentation here is new, using the language and techniques from our joint work [HV23b]. In particular, in [HV22], signature Gröbner bases are characterised essentially via reductions to zero. In contrast, here we use the cover criterion for that, which not only simplifies the presentation but also leads to a more general algorithm compared to [HV22] (see Remark 3.3.15).

More precisely, for introductory purposes, we initially present Algorithm 2 for computing *labelled Gröbner bases*. However, this algorithm is impractical due to its expensive module computations. Replacing those computations with signature manipulations naturally leads to Algorithm 3, which also includes the signature-based criteria. This algorithm can be considered as a generic template for signature-based algorithms in the free algebra. Finally, we introduce Algorithms 4 and 5, which allow to reconstruct the output of Algorithm 2 using only the signatures.

In Section 3.4, we then present the theory of signature Gröbner bases in the mixed algebra in full generality. As in the free case, our main result is a generalisation of the cover criterion, which allows us to state Algorithm 6 for computing signature Gröbner bases in the mixed algebra. More precisely, Algorithm 6 allows to compute labelled Gröbner bases in the mixed algebra, but an adaptation to signature Gröbner bases is straightforward and works analogous to the free case, which is discussed in detail in Section 3.3.3. Notably, the cover criterion has also been adapted in the commutative setting to coefficient rings [FV21]. The results of this section are published in [HV23b].

A difficulty specific to the case of noncommutative polynomials is that some ideals do not possess a finite signature Gröbner basis, even if they have a finite Gröbner basis. Additionally, the module of syzygies of the generators typically does not admit a finite Gröbner basis. While this is an inherent limitation, we prove that our algorithms nevertheless correctly enumerate a signature Gröbner basis, providing a Gröbner basis for the input ideal, as well as an effective description of the module of syzygies of the input polynomials.

We also provide (prototype) implementations of the algorithms presented in this chapter for SageMath. More precisely, the algorithms for the free algebra presented in Section 3.3 are implemented in the package `signature_gb`, which is described in more detail in Section 6.3, and a prototype implementation of Algorithm 6 in the mixed algebra is available at

`https://clemenshofstadler.com/software/`,

but, so far, only supports computations over coefficient fields.

In Section 3.5, we show empirically that the use of signatures allows to drastically reduce the number of S-polynomials considered and reduced to zero. First timings also indicate that, as in the commutative case, noncommutative signature-based algorithms can lead to an acceleration of Gröbner basis computations in the free algebra. Additionally, we provide experimental data indicating that the mixed algebra setting provides a clear advantage over other (more naive) approaches for computing noncommutative (signature) Gröbner bases involving some commutative variables.

3.1 Basics

In this section, we recall the definition of the mixed algebra and the free bimodule over the mixed algebra as well as of Gröbner bases in both of these settings. Additionally, we introduce the central concept of *signatures*.

3.1.1 The mixed algebra

In the following, R is a commutative ring (with unity) and $X = \{x_1, \ldots, x_k\}$, $Y = \{y_1, \ldots, y_n\}$ are disjoint sets of indeterminates. First, we define the main algebraic structure of interest in this chapter, the *mixed algebra*. To this end, we first introduce *mixed monomials*.

Definition 3.1.1. *The set of (commutative) monomials in X is denoted by $[X]$, that is,*

$$[X] = \{\boldsymbol{x}^{\boldsymbol{a}} = x_1^{a_1} \ldots x_k^{a_k} \mid \boldsymbol{a} = (a_1, \ldots, a_k) \in \mathbb{N}^k\}.$$

A mixed monomial *is a product $\boldsymbol{x}^{\boldsymbol{a}} w$ with $\boldsymbol{x}^{\boldsymbol{a}} \in [X]$ a (commutative) monomial in X and $w \in \langle Y \rangle$ a (noncommutative) word in Y. The set of all mixed monomials is denoted by $[X]\langle Y \rangle$.*

The set of mixed monomials becomes a (noncommutative) monoid with a component-wise multiplication of the commutative and noncommutative parts, that is, for $\boldsymbol{x}^{\boldsymbol{a}}v, \boldsymbol{x}^{\boldsymbol{b}}w \in [X]\langle Y\rangle$, we have

$$\left(\boldsymbol{x}^{\boldsymbol{a}}v\right) \cdot \left(\boldsymbol{x}^{\boldsymbol{b}}w\right) = \boldsymbol{x}^{\boldsymbol{a}+\boldsymbol{b}}vw.$$

With this, the mixed algebra can be defined as the monoid ring of the monoid of mixed monomials over the ring R.

Definition 3.1.2. *The monoid ring $\mathcal{A} = R[X]\langle Y\rangle$ of $[X]\langle Y\rangle$ over R is called the* mixed algebra *on X and Y over R. Elements in $\mathcal{A}$ are called* (mixed) polynomials*.*

Remark 3.1.3. *The mixed algebra can also be considered as the quotient*

$$\mathcal{A} = {}^{R\langle X, Y\rangle}\!\!\diagup\!\!_{([X, X \cup Y])},$$

where $[U, V] = \{uv - vu \mid u \in U, v \in V\}$ is the set of commutator relations between two sets U and V.

Note that, if $X = \emptyset$, we simply recover the definition of the free algebra on Y over R. Thus, the mixed algebra can be considered as a generalisation of the free algebra. Furthermore, there are canonical monomorphisms from the commutative polynomial ring $R[X]$ as well as from the free algebra $R\langle Y\rangle$ into $\mathcal{A}$. So we can consider these sets as subrings of $\mathcal{A}$.

Each element $f \in \mathcal{A}$ is of the form

$$f = \sum_{\boldsymbol{x}^{\boldsymbol{a}}w \in [X]\langle Y\rangle} c_{\boldsymbol{a},w}\boldsymbol{x}^{\boldsymbol{a}}w,$$

with only finite many nonzero $c_{\boldsymbol{a},w} \in R$. Note that, in this algebra, $x_i f = f x_i$ for all $f \in \mathcal{A}$, but in general $y_i f \neq f y_i$.

Example 3.1.4. *Consider the mixed algebra* $\mathcal{A} = \mathbb{Z}[s,t]\langle x,y\rangle$. *For* $f_1 = s^2xy + styx$, $f_2 = s^2xy - 2t^2yx \in \mathcal{A}$, *we can compute*

$$\begin{aligned} f_1 + f_2 &= 2s^2xy + styx - 2t^2yx \\ f_1f_2 &= (s^2xy + styx)(s^2xy - 2t^2yx) = s^4xyxy - 2s^2t^2xyyx + s^3tyxxy - 2st^3yxyx \\ f_2f_1 &= (s^2xy - 2t^2yx)(s^2xy + styx) = s^4xyxy + s^3txyyx - 2s^2t^2yxxy - 2st^3yxyx \end{aligned}$$

Note that $f_1f_2 \neq f_2f_1$.

A term in $\mathcal{A}$ is the product of a nonzero coefficient in R and a mixed monomial. We denote by $\mathrm{T}(\mathcal{A})$ the set of all terms of $\mathcal{A}$. Divisibility of terms in $\mathcal{A}$ is defined component-wise: given nonzero $c, d \in R$ and $\boldsymbol{x}^{\boldsymbol{a}}v, \boldsymbol{x}^{\boldsymbol{b}}w \in [X]\langle Y\rangle$, we say

$$c\boldsymbol{x}^{\boldsymbol{a}}v \text{ divides } d\boldsymbol{x}^{\boldsymbol{b}}w \quad :\Longleftrightarrow \quad c \text{ divides } d,\ \boldsymbol{x}^{\boldsymbol{a}} \text{ divides } \boldsymbol{x}^{\boldsymbol{b}},\ \text{and } v \text{ is a subword of } w.$$

Monomial orders on mixed monomials can be defined analogously to the case of noncommutative monomials (see Definition 2.4.1).

Definition 3.1.5. *A total order* $\preceq$ *on* $[X]\langle Y\rangle$ *is called a* monomial order *if it satisfies the following two conditions:*

1. $m \preceq m'$ *implies* $amb \preceq am'b$ *for all* $a, b, m, m' \in [X]\langle Y\rangle$;
2. *every nonempty subset of* $[X]\langle Y\rangle$ *has a least element;*

Remark 3.1.6. *In condition 1 of Definition 3.1.5, it is, in fact, enough to consider* $b \in \langle Y\rangle$.

For ease of notations, we extend the definition of monomial and term to contain 0, which is assumed to be smaller than all other elements. Furthermore, the notions of leading monomial, leading coefficient, and leading term are defined analogously to the free case (see Definition 2.4.17).

Example 3.1.7. *Let $\preceq_X$ be a monomial order on $[X]$ and $\preceq_Y$ be a monomial order on $\langle Y\rangle$. The following are examples of monomial orders on $[X]\langle Y\rangle$:*

- $\boldsymbol{x}^{\boldsymbol{a}} v \preceq \boldsymbol{x}^{\boldsymbol{b}} w \quad :\Longleftrightarrow \quad \left(\boldsymbol{x}^{\boldsymbol{a}} \prec_X \boldsymbol{x}^{\boldsymbol{b}}\right)$ *or* $\left(\boldsymbol{x}^{\boldsymbol{a}} = \boldsymbol{x}^{\boldsymbol{b}} \text{ and } v \preceq_Y w\right)$;
- $\boldsymbol{x}^{\boldsymbol{a}} v \preceq \boldsymbol{x}^{\boldsymbol{b}} w \quad :\Longleftrightarrow \quad (v \prec_Y w)$ *or* $\left(v = w \text{ and } \boldsymbol{x}^{\boldsymbol{a}} \preceq_X \boldsymbol{x}^{\boldsymbol{b}}\right)$;

Additionally, any degree function $\deg\colon [X]\langle Y\rangle \to \mathbb{R}^m$ *on* $[X]\langle Y\rangle$ *can be used as a first comparison criterion before using any of the orders above.*

In the following, we gather the most important results about (strong) Gröbner bases in the mixed algebra over a principal ideal domain R. We assume that R is computable, in the sense that all arithmetic operations, including gcd-computations and the computation of Bézout coefficients, can be performed effectively. Classical examples of such rings are the integers $\mathbb{Z}$ or the univariate polynomial ring $K[x]$ over a field K, with the extended Euclidean algorithm.

We start by adapting the notion of polynomial reduction to $\mathcal{A}$. To this end, we fix a monomial order $\preceq$ on $[X]\langle Y\rangle$.

Definition 3.1.8. *Let $a, b \in [X]\langle Y\rangle$, $g \in \mathcal{A} \setminus \{0\}$, and $G \subseteq \mathcal{A}$. We define the following reduction relations on $\mathcal{A}$:*

$$
\begin{aligned}
f \to_{a,g,b} f' \quad &:\Longleftrightarrow \quad \textit{all the following conditions hold:} \\
&\quad \bullet\ \operatorname{lm}(agb) \in \operatorname{supp}(f); \\
&\quad \bullet\ \operatorname{lc}(g) \textit{ divides } \operatorname{coeff}(f, \operatorname{lm}(agb)); \\
&\quad \bullet\ f' = f - \frac{\operatorname{coeff}(f, \operatorname{lm}(agb))}{\operatorname{lc}(g)} agb; \\
\to_g \quad &:= \quad \bigcup_{a,b \in \langle X\rangle} \to_{a,g,b} \\
\to_G \quad &:= \quad \bigcup_{g \in G \setminus \{0\}} \to_g
\end{aligned}
$$

The relations introduced above differ in two ways from the polynomial reduction relations in Definition 2.4.21. First, noncommutative polynomials and monomials are replaced by mixed polynomials and monomials. Moreover, the second condition in the definition

of $\to_{a,g,b}$ is new. This condition arises due to the fact that $\mathcal{A} = R[X]\langle Y\rangle$ is defined over a coefficient ring, while Definition 2.4.21 considers the free algebra over a coefficient field. In the latter case, divisibility of coefficients is trivially fulfilled in all cases. Thus, if $X = \emptyset$ and if R is a field, we recover Definition 2.4.21.

Remark 3.1.9. *When working with (commutative, noncommutative, or mixed) polynomials over coefficient rings, several notions of reductions exist (weak, strong, and also modular reductions by the coefficients), see, for example, [AL94, Ch. 4] or [Mor16, Ch. 46]. In this work, we focus on strong reductions requiring divisibility of the leading coefficients.*

Like in the free case, a reduction $f \to_G f'$ is a *top reduction* if $\mathrm{lm}(f) \succ \mathrm{lm}(f')$ and a *tail reduction* otherwise. Furthermore, recall that $\overset{*}{\to}_G$ denotes the reflexive, transitive closure of $\to_G$. We also note that termination of $\to_G$ can be proven like in the free case (see Proposition 2.4.25 and Corollary 2.4.26).

In the following, we introduce Gröbner bases in the mixed algebra via reducibility to zero. We note that a Gröbner theory could also be developed analogous to Section 2.4.3 using confluence of the reduction relation as the defining property. The two definitions are equivalent, and the proof is the same as in the classical case. Here, however, we stick to the following definition, which shall prove more useful.

Definition 3.1.10. *Let $I \trianglelefteq \mathcal{A}$. A subset $G \subseteq I$ is a* (strong) Gröbner basis *of I if $f \overset{*}{\to}_G 0$ for all $f \in I$.*

Proposition 3.1.11. *Let $I \trianglelefteq \mathcal{A}$. A subset $G \subseteq I$ is a Gröbner basis of I if and only if, for all nonzero $f \in I$, there exists $g \in G$ such that $\mathrm{lt}(g)$ divides $\mathrm{lt}(f)$.*

The proof of Proposition 3.1.11 is analogous to that of the corresponding statements in Theorem 2.4.37. We nevertheless include it here for the convenience of the reader.

Proof. For the "if"-direction, assume, for contradiction, that there exists $f \in I \setminus \{0\}$ which cannot be reduced to zero by G. Without loss of generality, we can assume that f is irreducible with respect to $\to_G$. By assumption, there exists $g \in G$ such that $\mathrm{lt}(g)$ divides $\mathrm{lt}(f)$, but then f is (top) reducible by g – a contradiction.

For the "only if"-direction, let $f \in I \setminus \{0\}$ be arbitrary. By assumption, $f \xrightarrow{*}_G 0$, but this is only possible if f is top reducible by G, that is, if there exist $c \in R$, $a, b \in [X]\langle Y\rangle$, and $g \in G$ such that $\mathrm{lt}(f) = \mathrm{lt}(cagb)$, showing that $\mathrm{lt}(f)$ is divisible by $\mathrm{lt}(g)$. □

3.1.2 Free bimodule over the mixed algebra

Let $\mathcal{A} = R[X]\langle Y\rangle$ be the mixed algebra on $X = \{x_1, \ldots, x_k\}$ and $Y = \{y_1, \ldots, y_n\}$ over the commutative principal ideal domain R. For $r \in \mathbb{N}_{>0}$, we considered the free $\mathcal{A}$-bimodule (see Definition 2.2.45)

$$\Sigma = (\mathcal{A} \otimes_{Z(\mathcal{A})} \mathcal{A})^{(\mathcal{E})}$$

on the set $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_r\}$ centralising the center $Z(\mathcal{A})$ of $\mathcal{A}$. Note that $Z(\mathcal{A}) = R[X]$ if $|Y| \neq 1$ and, if $Y = \{y\}$, then $Z(\mathcal{A}) = \mathcal{A}$, as, in this case, $\mathcal{A} = R[X]\langle y\rangle = R[X, y]$.

Recall that elements in this bimodule are of the form

$$\alpha = \sum_{i=1}^{r} \left(\sum_{j=1}^{d_i} p_{i,j} \otimes q_{i,j} \right) \varepsilon_i,$$

with $p_{i,j}, q_{i,j} \in \mathcal{A}$, which we also write as

$$\alpha = \sum_{i=1}^{r} \sum_{j=1}^{d_i} p_{i,j} \varepsilon_i q_{i,j}.$$

Note that Σ is a free R-module with basis

$$M(\Sigma) \coloneqq \{a\varepsilon_i b \mid a, b \in [X]\langle Y\rangle, 1 \leq i \leq r\}. \tag{3.1}$$

Remark 3.1.12. *The representation* (3.1) *of* $M(\Sigma)$ *lists elements more than once, as*

$$\boldsymbol{x}^{\boldsymbol{a}} v \varepsilon_i \boldsymbol{x}^{\boldsymbol{b}} w = \boldsymbol{x}^{\boldsymbol{a}+\boldsymbol{b}} v \varepsilon_i w = v \varepsilon_i \boldsymbol{x}^{\boldsymbol{a}+\boldsymbol{b}} w = \boldsymbol{x}^{\boldsymbol{b}} v \varepsilon_i \boldsymbol{x}^{\boldsymbol{a}} w.$$

If $|Y| \neq 1$, *a representation of* $M(\Sigma)$ *in which every element occurs precisely once, is given by*

$$M(\Sigma) = \{\boldsymbol{x}^{\boldsymbol{a}} v \varepsilon_i w \mid \boldsymbol{x}^{\boldsymbol{a}} \in [X], v, w \in \langle Y\rangle, 1 \leq i \leq r\}.$$

Every nonzero $\alpha \in \Sigma$ can be written uniquely as $\alpha = \sum_{i=1}^{d} c_i \mu_i$ with nonzero $c_i \in R$ and pairwise different $\mu_i \in M(\Sigma)$. We call $M(\Sigma)$ the set of *(bi)module monomials* of Σ. Furthermore, the set of *(bi)module terms* of Σ is defined as

$$T(\Sigma) \coloneqq \{c\mu \mid c \in R, \mu \in M(\Sigma)\}.$$

Divisibility of nonzero terms in Σ is defined as follows: let $c, d \in R$ be nonzero and $a\varepsilon_i b, a'\varepsilon_j b' \in M(\Sigma)$ be module monomials. We assume that all commutative variables have been collected on the left-hand sides, that is, $b, b' \in \langle Y \rangle$ if $|Y| > 1$ and $b, b' = 1$ otherwise. Then we say

$$ca\varepsilon_i b \text{ divides } da'\varepsilon_j b' \quad :\Longleftrightarrow \quad c \text{ divides } d,\ i = j,\ \exists a'', b'' \in [X]\langle Y \rangle : a' = a''a \text{ and } b' = bb'',$$

Just like a monomial order allows to compare monomials, a *(bi)module order* on $M(\Sigma)$ allows to compare module monomials.

Definition 3.1.13. *A total order $\preceq_\Sigma$ on $M(\Sigma)$ is called a* (bi)module order *if it satisfies the following two conditions:*

1. *$\mu \preceq \sigma$ implies $a\mu b \preceq_\Sigma a\sigma b$ for all $\mu, \sigma \in M(\Sigma)$ and $a, b \in [X]\langle Y \rangle$;*
2. *every nonempty subset of $M(\Sigma)$ has a least element;*

Many classical module orders arise as extensions of a monomial order. We mention a few important ones.

Example 3.1.14. *In the following, tuples $\boldsymbol{p} = (p_1, \ldots, p_m)$ and $\boldsymbol{q} = (q_1, \ldots, q_m)$ are compared lexicographically from left to right, that is, $\boldsymbol{p} \leq \boldsymbol{q}$ if $p_i = q_i$ for all $i = 1, \ldots, m$ or $p_i < q_i$ for the smallest index $1 \leq i \leq m$ where p_i and q_i differ. In this process, if p_i and q_i are monomials, they are compared using a fixed monomial order. Let $a\varepsilon_i b, a'\varepsilon_j b' \in M(\Sigma)$ be module monomials. We assume that all commutative variables have been collected on the left-hand sides, that is, $b, b' \in \langle Y \rangle$ if $|Y| > 1$ and $b, b' = 1$ otherwise.*

1. *The* position-over-term *order $\preceq_{\mathrm{PoT}}$ is defined by:*

$$a\varepsilon_i b \preceq_{\mathrm{PoT}} a'\varepsilon_j b' \quad :\Longleftrightarrow \quad (i, a, b) \leq (j, a', b').$$

2. The term-over-position *order* $\preceq_{\mathrm{ToP}}$ *is defined by:*

$$a\varepsilon_i b \preceq_{\mathrm{ToP}} a'\varepsilon_j b' \quad :\Longleftrightarrow \quad (a,b,i) \leq (a',b',j).$$

For the following orders, we fix, for each basis element ε_i*, a nonzero polynomial* $f_i \in \mathcal{A}$*.*

3. The degree-over-position-over-term *order* $\preceq_{\mathrm{DoPoT}}$ *is defined by:*

$$a\varepsilon_i b \preceq_{\mathrm{DoPoT}} a'\varepsilon_j b' \quad :\Longleftrightarrow \quad \big(\deg(af_ib), i, a, b\big) \leq \big(\deg(a'f_jb'), j, a', b'\big).$$

4. The degree-over-term-over-position *order* $\preceq_{\mathrm{DoToP}}$ *is defined by:*

$$a\varepsilon_i b \preceq_{\mathrm{DoToP}} a'\varepsilon_j b' \quad :\Longleftrightarrow \quad \big(\deg(af_ib), a, b, i\big) \leq \big(\deg(a'f_jb'), a', b', j\big).$$

We fix a module order $\preceq_\Sigma$ on $M(\Sigma)$. For ease of notation, we extend $\preceq_\Sigma$ to $M(\Sigma) \cup \{0\}$ and set $0 \prec_\Sigma \mu$ for all $\mu \in M(\Sigma)$.

Definition 3.1.15. *Let* $\alpha = \sum_{i=1}^{d} c_i\mu_i$ *with nonzero* $c_i \in R$ *and pairwise different* $\mu_i \in M(\Sigma)$*. The support of* α *is* $\mathrm{supp}(\alpha) = \{\mu_1, \ldots, \mu_d\}$*. For* $\alpha \neq 0$*, the* signature monomial $\mathrm{sm}(\alpha)$ *of* α *is the* $\preceq_\Sigma$*-maximal element in* $\mathrm{supp}(\alpha)$ *and the* signature coefficient $\mathrm{sc}(\alpha)$ *of* α *is the coefficient of* $\mathrm{sm}(\alpha)$*. For* $\alpha = 0$*, we define* $\mathrm{sc}(0) = \mathrm{sm}(0) = 0$*. The* signature $\mathrm{sig}(\alpha)$ *of* α *is* $\mathrm{sig}(\alpha) = \mathrm{sc}(\alpha) \cdot \mathrm{sm}(\alpha)$*.*

Remark 3.1.16. *We define the signature* $\mathrm{sig}(\alpha)$ of a module element $\alpha \in \Sigma$*, which is different to the original (commutative) definition of a signature. Initially, signatures were looked at from a polynomial point of view, and therefore, signatures* of polynomials *were defined [Fau02]. We refer to Remark 3.1.22 for further information on the original definition of signatures, see also [EF17, Rem. 2.1]. Our notion of signature is a noncommutative adaptation of the concept first introduced in [GGV10]. Furthermore, we note that our signatures are* terms*, and thus, also include a coefficient. This inclusion is motivated by our consideration of signature Gröbner bases over coefficient rings, and was also done in the commutative setting, see [FV21].*

One immediate consequence of our definition of signatures is the following lemma.

Lemma 3.1.17. *If $\alpha \in \Sigma$, $a, b \in [X]\langle Y\rangle$, and $c \in R$, then* $\operatorname{sig}(ca\alpha b) = ca\operatorname{sig}(\alpha)b$.

Note that the signature consists of a coefficient and a module monomial. Thus, to easily compare signatures, we extend $\preceq_\Sigma$ to a total preorder on nonzero terms.

Definition 3.1.18. *Let $\mu, \sigma \in M(\Sigma)$ and $c, d \in R \setminus \{0\}$. The terms $c\mu$ and $d\sigma$ are* similar, *denoted by $c\mu \simeq d\sigma$, if $\mu = \sigma$. Furthermore, we write $c\mu \preceq_\Sigma d\sigma$ if $\mu \prec_\Sigma \sigma$ or $c\mu \simeq d\sigma$.*

To define Gröbner bases in Σ, we adapt the reduction relation to this setting.

Definition 3.1.19. *Let $a, b \in [X]\langle Y\rangle$, $\beta \in \Sigma \setminus \{0\}$, and $H \subseteq \Sigma$. We define the following reduction relations on Σ:*

$$
\begin{aligned}
\alpha \to_{a,\beta,b} \alpha' \quad &:\Longleftrightarrow \quad \textit{all the following conditions hold:} \\
&\quad \bullet\ \operatorname{sm}(a\beta b) \in \operatorname{supp}(\alpha); \\
&\quad \bullet\ \operatorname{sc}(\beta)\ \textit{divides}\ \operatorname{coeff}(\alpha, \operatorname{sm}(a\beta b)); \\
&\quad \bullet\ \alpha' = \alpha - \frac{\operatorname{coeff}(\alpha, \operatorname{sm}(a\beta b))}{\operatorname{sc}(\beta)} a\beta b; \\
\to_\beta \quad &:= \quad \bigcup_{a,b \in [X]\langle Y\rangle} \to_{a,\beta,b} \\
\to_H \quad &:= \quad \bigcup_{\beta \in H \setminus \{0\}} \to_\beta
\end{aligned}
$$

The relations introduced above can be compared to those of Definition 3.1.8; the main difference is that mixed polynomials are replaced by bimodule elements, which also requires to exchange notions like leading coefficient and monomial by the corresponding signature analogues.

Analogous to the polynomial case, a reduction $\alpha \to_H \alpha'$ is called a *top reduction* if $\operatorname{sig}(\alpha) \succ_\Sigma \operatorname{sig}(\alpha')$ and a *tail reduction* otherwise. Termination of $\to_H$ can be proven like in the polynomial case (see Proposition 2.4.25 and Corollary 2.4.26).

Gröbner bases in Σ can be defined just like in the mixed algebra via reducibility to zero. As noted in the previous section, a definition using confluence of the reduction relation is also possible and equivalent to ours, however, for the present application less practical.

Furthermore, since signature-based algorithms allow to compute Gröbner bases of the syzygy module in a very structured way, we also introduce the notion of Gröbner bases *up to a signature* σ.

Definition 3.1.20. *Let $M \subseteq \Sigma$ be an $\mathcal{A}$-subbimodule. A subset $H \subseteq M$ is a* Gröbner basis of M up to signature $\sigma \in T(\Sigma)$ *if $\alpha \xrightarrow{*}_H 0$ for all $\alpha \in M$ with $\operatorname{sig}(\alpha) \prec_\Sigma \sigma$. Furthermore, H is a* Gröbner basis of M *if $\alpha \xrightarrow{*}_H 0$ for all $\alpha \in M$.*

Proposition 3.1.21. *Let $M \subseteq \Sigma$ be an $\mathcal{A}$-subbimodule. A subset $H \subseteq M$ is a Gröbner basis of M if and only if, for all nonzero $\alpha \in M$, there exists $\beta \in H$ such that $\operatorname{sig}(\beta)$ divides $\operatorname{sig}(\alpha)$.*

The proof of Proposition 3.1.21 is a direct adaptation of that of Proposition 3.1.11. We nevertheless include it here for the convenience of the reader.

Proof. For the "if"-direction, assume, for contradiction, that there exists $\alpha \in M \setminus \{0\}$ which cannot be reduced to zero by H. Without loss of generality, we can assume that α is irreducible with respect to $\to_H$. By assumption, there exists $\beta \in H$ such that $\operatorname{sig}(\beta)$ divides $\operatorname{sig}(\alpha)$, but then α is (top) reducible by β – a contradiction.

For the "only if"-direction, let $\alpha \in M \setminus \{0\}$ be arbitrary. By assumption, $\alpha \xrightarrow{*}_H 0$, but this is only possible if α is top reducible by H, that is, if there exist $c \in R$, $a, b \in [X]\langle Y\rangle$, and $\beta \in H$ such that $\operatorname{sig}(\alpha) = \operatorname{sig}(ca\beta b)$, showing that $\operatorname{sig}(\alpha)$ is divisible by $\operatorname{sig}(\beta)$. □

3.1.3 Signature and labelled polynomials

In this section, we relate the mixed algebra $\mathcal{A}$ to the free $\mathcal{A}$-bimodule Σ introduced in the previous section. More precisely, in order to encode relations between a family $(f_1, \dots, f_r) \in \mathcal{A}^r$ of polynomials in $\mathcal{A}$ generating an ideal $I = (f_1, \dots, f_r)$, we consider the map

$$\overline{\cdot}\colon \Sigma \to I, \quad \alpha = \sum_{i=1}^{r} \sum_{j=1}^{d_i} p_{i,j} \varepsilon_i q_{i,j} \quad \mapsto \quad \overline{\alpha} \coloneqq \sum_{i=1}^{r} \sum_{j=1}^{d_i} p_{i,j} f_i q_{i,j}.$$

Remark 3.1.22. *The original definition of signatures given in [Fau02] uses the map* $\overline{\cdot}$ *and looks as follows: given a nonzero polynomial* $f \in I$*, its signature is uniquely defined as* $\min\{\mathrm{sm}(\alpha) \mid \overline{\alpha} = f\}$*. Note that signatures were originally defined for* polynomials*, while we use signatures of* module elements*. Furthermore, originally, signatures did not contain a coefficient, which was due to the fact that only coefficient fields were considered.*

The following lemma follows immediately from the definition.

Lemma 3.1.23. *The map* $\overline{\cdot}$ *is a surjective* $\mathcal{A}$*-bimodule homomorphism.*

Using the map $\overline{\cdot}$, we can relate the elements in an ideal $I = (f_1, \ldots, f_r) \trianglelefteq \mathcal{A}$ to their module representations in Σ. To this end, we adapt notation from [Sun+12].

Definition 3.1.24. *We denote by* $f^{[\alpha]}$ *a pair* $(f, \alpha) \in I \times \Sigma$ *with* $f = \overline{\alpha}$ *and refer to it as a* labelled polynomial*. Furthermore, we denote by* $f^{(\sigma)}$ *a pair* $(f, \sigma) \in I \times T(\Sigma)$ *such that there exists* $f^{[\alpha]}$ *with* $\mathrm{sig}(\alpha) = \sigma$ *and refer to it as a* signature polynomial*.*

We denote by $I^{[\Sigma]}$ *and* $I^{(\Sigma)}$ *the set of all labelled polynomials and the set of all signature polynomials respectively, that is,*

$$\begin{aligned} I^{[\Sigma]} &:= \left\{ f^{[\alpha]} \mid \alpha \in \Sigma,\ f = \overline{\alpha} \right\} \subseteq I \times \Sigma, \\ I^{(\Sigma)} &:= \left\{ f^{(\sigma)} \mid \exists f^{[\alpha]} \in I^{[\Sigma]} : \mathrm{sig}(\alpha) = \sigma \right\} \subseteq I \times T(\Sigma). \end{aligned}$$

We refer to $I^{[\Sigma]}$ *and* $I^{(\Sigma)}$ *as the* labelled module *and the* signature module *respectively generated by* $f_1, \ldots, f_r$*.*

Remark 3.1.25. *Different families of generators of the same ideal* $I \trianglelefteq \mathcal{A}$ *lead to different sets* $I^{[\Sigma]}$*. More precisely, given* $I^{[\Sigma]}$*, the family of generators of* I *used in the construction can be recovered:* f_i *is the polynomial part of the element* $f_i^{[\varepsilon_i]}$ *in* $I^{[\Sigma]}$*.*

Lemma 3.1.23 implies that $I^{[\Sigma]}$ is indeed an $\mathcal{A}$-bimodule with component-wise addition and scalar multiplication, that is, for $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$, $c \in R$, and $a, b \in [X]\langle Y \rangle$ we have

$$f^{[\alpha]} + g^{[\beta]} = (f+g)^{[\alpha+\beta]}, \qquad caf^{[\alpha]}b = (cafb)^{[ca\alpha b]}.$$

Also, by definition, $f^{[\alpha]} = g^{[\beta]}$ if and only if $f = g$ and $\alpha = \beta$.

We extend the definition of signature to labelled and signature polynomials and define, for $f^{[\alpha]} \in I^{[\Sigma]}$ and $f^{(\sigma)} \in I^{(\Sigma)}$, $\operatorname{sig}(f^{[\alpha]}) = \operatorname{sig}(\alpha)$ and $\operatorname{sig}(f^{(\sigma)}) = \sigma$ respectively.

The motivation for using the notation $f^{[\alpha]}$ (resp. $f^{(\sigma)}$) is that the polynomial f, rather than the module element α (resp. the module term σ), is the main object of interest. Furthermore, the reason for introducing both labelled polynomials and signature polynomials is that the former allow to present the theory in a simpler fashion and lead to simpler proofs. However, in an actual implementation of a signature-based algorithm, keeping track of the full module representation stored in a labelled polynomial causes a significant overhead in terms of memory consumption and overall computation time. Fortunately, we will see that all theoretical results only depend on information encoded in signature polynomials. Consequently, when implementing a signature-based algorithm, one would only work with signature polynomials. This reduces the computational overhead to a minimum. Additionally, we note that the reconstruction techniques discussed in Section 3.3.3 allow to efficiently recover all information encoded in labelled polynomials from the signature polynomials computed by signature-based algorithms.

In what follows, we will sometimes work with sets of polynomials, sometimes with sets of labelled polynomials and sometimes with sets of signature polynomials. To be able to better distinguish between these cases, we denote subsets of $\mathcal{A}$ by capital letters (for example, $G \subseteq \mathcal{A}$) and subsets of $I^{[\Sigma]}$ and $I^{(\Sigma)}$ by capital letters with the additional exponent $^{[\Sigma]}$ and $^{(\Sigma)}$ respectively (for example, $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ and $G^{(\Sigma)} \subseteq I^{(\Sigma)}$).

To end this section, we introduce the notion of *syzygies* of a labelled module $I^{[\Sigma]}$. To this end, we fix a family of polynomials $(f_1, \ldots, f_r) \in \mathcal{A}^r$ and let $I^{[\Sigma]}$ be the labelled module generated by $f_1, \ldots, f_r$.

Definition 3.1.26. *A* syzygy *of $I^{[\Sigma]}$ is any element $\gamma \in \Sigma$ such that $0^{[\gamma]} \in I^{[\Sigma]}$. The set of all syzygies of $I^{[\Sigma]}$ is denoted by* $\operatorname{Syz}(I^{[\Sigma]})$.

In other words, a syzygy of $I^{[\Sigma]}$ is a bimodule element γ such that $\overline{\gamma} = 0$. Thus, any syzygy of $I^{[\Sigma]}$ encodes a relation between the generators $f_1, \ldots, f_r$. Some of these relations are trivial, originating from the fact that $fmg - fmg = 0$ for all $f, g \in \mathcal{A}$ and $m \in \langle Y \rangle$. This leads to the following definition.

Definition 3.1.27. *Let $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$. For all $m \in \langle Y \rangle$, the syzygy $\alpha m g - f m \beta$ is called a* trivial syzygy *between $f^{[\alpha]}$ and $g^{[\beta]}$. For $G^{[\Sigma]} \subseteq I^{[\Sigma]}$, the set of all* trivial syzygies of $G^{[\Sigma]}$ *is*

$$\operatorname{Triv}(G^{[\Sigma]}) \coloneqq \left\{ \alpha m g - f m \beta \mid f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]}, m \in \langle Y \rangle \right\}.$$

We note that $\operatorname{Syz}(I^{[\Sigma]})$ forms an $\mathcal{A}$-subbimodule of Σ.

Remark 3.1.28. *Note that trivial syzygies are defined only for purely noncommutative middle parts $m \in \langle Y \rangle$. While they could also be defined more generally for mixed monomials $m = \boldsymbol{x}^{\boldsymbol{a}} w \in [X]\langle Y \rangle$, any such element is always a multiple of a trivial syzygy with purely noncommutative middle part, since*

$$\alpha m g - f m \beta = \alpha \boldsymbol{x}^{\boldsymbol{a}} w g - f \boldsymbol{x}^{\boldsymbol{a}} w \beta = \boldsymbol{x}^{\boldsymbol{a}} \left(\alpha w g - f w \beta \right).$$

Therefore, it suffices to focus solely on trivial syzygies as defined in Definition 3.1.27.

3.2 Signature and labelled Gröbner bases

In this section, we introduce the notion of *signature Gröbner bases* of ideals in the mixed algebra $\mathcal{A}$. As an intermediate notion, we define the concept of *labelled Gröbner bases*, which are Gröbner bases keeping track of cofactor representations of each element with respect to the generators. As in the commutative case, signature Gröbner bases are defined using a more restrictive notion of polynomial reduction, called sig-*reduction*. Moreover, we also define and characterise noncommutative *minimal* signature/labelled Gröbner bases. We note that all notions introduced here are straightforward generalisations of the same notions for commutative polynomials. The key differences between the commutative and noncommutative case will become apparent in Sections 3.3 and 3.4.

We fix $(f_1, \ldots, f_r) \in \mathcal{A}^r$ and let $I^{[\Sigma]}$ be the labelled module generated by $f_1, \ldots, f_r$. Furthermore, we fix a monomial order on $[X]\langle Y \rangle$ and a module order on $M(\Sigma)$, both of which will be denoted by the same symbol $\preceq$. It will be clear from the context which order is meant, as we will denote elements from Σ by Greek letters and elements from $\mathcal{A}$ by Roman letters.

In order to discuss signature Gröbner bases, we adapt the notion of polynomial reduction to labelled polynomials. Recall that we focus on strong reductions requiring divisibility of the leading coefficients. This leads to the following definition of (strong) sig-*reduction*.

Definition 3.2.1. *Let* $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ *with* $g \neq 0$. *We say that* $f^{[\alpha]}$ *is* (strongly) sig-reducible *by* $g^{[\beta]}$ *if there exist* $a, b \in [X]\langle Y\rangle$ *such that the following conditions hold:*

- $\mathrm{lm}(agb) \in \mathrm{supp}(f)$*;*
- $\mathrm{lc}(g)$ *divides* $\mathrm{coeff}(f, \mathrm{lm}(agb))$*;*
- $\mathrm{sig}(a\beta b) \preceq \mathrm{sig}(\alpha)$*;*

The corresponding sig-reduction *is*

$$f'^{[\alpha']} = f^{[\alpha]} - \frac{\mathrm{coeff}(f, \mathrm{lm}(agb))}{\mathrm{lc}(g)} a g^{[\beta]} b,$$

which we denote by $f^{[\alpha]} \to_{g^{[\beta]}} f'^{[\alpha']}$.

A sig-*reduction is a* top sig-reduction *if* $\mathrm{lm}(f') \prec \mathrm{lm}(f)$*, and a* tail sig-reduction *otherwise. Furthermore, a* sig-*reduction is called* singular *if* $\mathrm{sig}(\alpha') \prec \mathrm{sig}(\alpha)$*, and* regular *if* $\mathrm{sig}(\alpha') = \mathrm{sig}(\alpha)$.

Remark 3.2.2. *The first two conditions in Definition 3.2.1 mean that we perform classical polynomial reduction, which implies that* $\mathrm{lm}(f') \preceq \mathrm{lm}(f)$*. The last condition ensures that this inequality also transfers over to* Σ*, so that* $\mathrm{sig}(\alpha') \preceq \mathrm{sig}(\alpha)$*, with equality if and only if the* sig-*reduction is regular. Note that there are also* sig-*reductions which are neither regular nor singular; in this case* $\mathrm{sig}(\alpha') \simeq \mathrm{sig}(\alpha)$*. Furthermore, if* R *is a field, then the second condition is trivially fulfilled in all cases.*

Those observations allow to generalise the definition to signature polynomials. More precisely, given $f^{(\sigma)}, g^{(\mu)} \in I^{(\Sigma)}$*, it is possible to test whether* $f^{(\sigma)}$ *is* sig-*reducible by* $g^{(\mu)}$*. Furthermore, if the* sig-*reduction is regular, it is possible to compute its remainder as a signature polynomial.*

We extend the notions from Definition 3.2.1 to sets of labelled polynomials $G^{[\Sigma]} \subseteq I^{[\Sigma]}$. In particular, $f^{[\alpha]}$ is sig-*reducible* by $G^{[\Sigma]}$ if there exists $g^{[\beta]} \in G^{[\Sigma]}$ such that $f^{[\alpha]}$ is

sig-reducible by $g^{[\beta]}$. Furthermore, $f^{[\alpha]} \to_{G^{[\Sigma]}} f'^{[\alpha']}$ if there exists $g^{[\beta]} \in G^{[\Sigma]}$ such that $f^{[\alpha]} \to_{g^{[\beta]}} f'^{[\alpha']}$. Recall that $\xrightarrow{*}_{G^{[\Sigma]}}$ denotes the reflexive transitive closure of $\to_{G^{[\Sigma]}}$.

If $f^{[\alpha]}$ sig-reduces to $0^{[\alpha']}$ (in zero or more steps), we say that $f^{[\alpha]}$ sig-*reduces to zero*.

We capture some useful facts about sig-reductions that will be needed later. This first lemma follows immediately from the definition (in particular, from the first sentence in Remark 3.2.2).

Lemma 3.2.3. *If $f^{[\alpha]} \xrightarrow{*}_{G^{[\Sigma]}} f'^{[\alpha']}$, then $f \xrightarrow{*}_{G} f'$, where $G = \{g \mid g^{[\beta]} \in G^{[\Sigma]}\}$.*

Using the notion of sig-reducibility, we now define *(strong) labelled Gröbner bases* of the module $I^{[\Sigma]}$, extending the definition from the commutative case [FV21, Def. 2.4] in a straightforward way.

Definition 3.2.4. *A set $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ is a* (strong) labelled Gröbner basis of $I^{[\Sigma]}$ up to signature $\sigma \in T(\Sigma)$ *if all $f^{[\alpha]} \in I^{[\Sigma]}$ with* $\operatorname{sig}(\alpha) \prec \sigma$ sig-*reduce to zero by $G^{[\Sigma]}$. Furthermore, $G^{[\Sigma]}$ is a* (strong) labelled Gröbner basis of $I^{[\Sigma]}$ *if all $f^{[\alpha]} \in I^{[\Sigma]}$* sig-*reduce to zero by $G^{[\Sigma]}$.*

Remark 3.2.5. *Since different families of generators of the ideal I lead to different labelled modules $I^{[\Sigma]}$, they also lead to different labelled Gröbner bases (see also Example 3.2.16). Additionally, the labelled Gröbner bases also depend on the used monomial and module orders.*

Disregarding the module labelling from Definition 3.2.4 recovers the definition of classical Gröbner bases in $\mathcal{A}$ (see Definition 3.1.10). The following result follows immediately from Lemma 3.2.3.

Lemma 3.2.6. *If $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ is a labelled Gröbner basis of $I^{[\Sigma]}$, then $\{g \mid g^{[\beta]} \in G^{[\Sigma]}\}$ is a Gröbner basis of $I \subseteq \mathcal{A}$.*

We also provide the following equivalent characterisation of labelled Gröbner bases that will turn out useful later.

Lemma 3.2.7. *A set $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ is a labelled Gröbner basis of $I^{[\Sigma]}$ (up to signature $\sigma \in T(\Sigma)$) if and only if every $f^{[\alpha]} \in I^{[\Sigma]}$ with $f \neq 0$ (and $\mathrm{sig}(\alpha) \prec \sigma$) is top* sig*-reducible by $G^{[\Sigma]}$.*

Proof. The "only if"-direction is clear. For the "if"-direction, assume, for contradiction, that $G^{[\Sigma]}$ is not a labelled Gröbner basis (up to signature σ). Then there exists $f^{[\alpha]} \in I^{[\Sigma]}$ with $f \neq 0$ (and $\mathrm{sig}(\alpha) \prec \sigma$) which does not sig-reduce to zero by $G^{[\Sigma]}$. Choose such an element with minimal leading monomial $\mathrm{lm}(f)$. By assumption, $f^{[\alpha]}$ is top sig-reducible by $G^{[\Sigma]}$. Let $f'^{[\alpha']}$ be the result of this sig-reduction. By the minimality of $\mathrm{lm}(f)$, the element $f'^{[\alpha']}$ can be sig-reduced to zero by $G^{[\Sigma]}$, but then so can be $f^{[\alpha]}$ – a contradiction. □

Like classical Gröbner bases of an ideal, a labelled Gröbner basis of $I^{[\Sigma]}$ is not unique. In fact, if $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ is a labelled Gröbner basis of $I^{[\Sigma]}$, then so is $G^{[\Sigma]} \cup \{f^{[\alpha]}\}$ for every $f^{[\alpha]} \in I^{[\Sigma]}$. Furthermore, the set $I^{[\Sigma]}$ is always a labelled Gröbner basis of $I^{[\Sigma]}$.

Lemma 3.2.8. *Let $(f_1, \ldots, f_r) \in \mathcal{A}^r$. The labelled module $I^{[\Sigma]}$ generated by $f_1, \ldots, f_r$ has a (possibly infinite) labelled Gröbner basis.*

We remind the reader that we present all the relevant theory for our signature-based algorithm in terms of labelled polynomials, keeping in mind that in an actual implementation one would work only with signature polynomials. Consequently, such an implementation would not compute a labelled Gröbner basis but instead a signature Gröbner basis as defined below.

Definition 3.2.9. *A set $G^{(\Sigma)} \subseteq I^{(\Sigma)}$ is a* (strong) signature Gröbner basis of $I^{[\Sigma]}$ (up to signature $\sigma \in T(\Sigma)$) *if there exists a labelled Gröbner basis $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ (up to signature σ) such that*

$$G^{(\Sigma)} = \left\{ g^{(\mathrm{sig}(\beta))} \mid g^{[\beta]} \in G^{[\Sigma]} \right\}.$$

In the theoretical sections, which are this one and Sections 3.3.1 and 3.3.2, we focus on the notion of labelled Gröbner bases and present all theoretical results in terms of this concept. However, one should always keep in mind that all relevant results also transfer over to signature Gröbner bases as they only depend on information that is also

available in signature polynomials. In Section 3.3.3, we then shift our focus to a more application-oriented point of view and present our signature-based algorithm in terms of signature polynomials. Additionally, we show how to reconstruct a labelled Gröbner basis from a signature Gröbner basis.

The outcome of classical polynomial reduction depends on more than just the leading term of the polynomial that is reduced. Polynomials which share the same leading term can still reduce to different elements, even under reduction by a partial Gröbner basis. In case of regular sig-reductions by a partial labelled Gröbner basis over a coefficient field, all labelled polynomials with similar signatures yield the same regular sig-reduced normal form (up to multiplication by a constant). This fact follows from the following lemma, which is an analogue of [RS12, Lem. 2] in the mixed algebra.

Lemma 3.2.10. *Let $\mathcal{A} = K[X]\langle Y\rangle$ with a field K and let $I^{[\Sigma]}$ be the labelled module generated by $(f_1, \ldots, f_r) \in \mathcal{A}^r$. Furthermore, let $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ be such that $\operatorname{sig}(\alpha) \simeq \operatorname{sig}(\beta)$ and let $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ be a labelled Gröbner basis up to signature $\operatorname{sig}(\alpha)$. Then, with $c \in K$ such that $\operatorname{sig}(c\alpha) = \operatorname{sig}(\beta)$, the following hold:*

- *If $f^{[\alpha]}$ and $g^{[\beta]}$ are regular* sig*-reduced by $G^{[\Sigma]}$, then $cf = g$.*
- *If $f^{[\alpha]}$ and $g^{[\beta]}$ are regular top* sig*-reduced by $G^{[\Sigma]}$, then $\operatorname{lt}(cf) = \operatorname{lt}(g)$.*

Proof. We first prove that $cf = g$ if $f^{[\alpha]}$ and $g^{[\beta]}$ are regular sig-reduced. Consider $h^{[\gamma]} = cf^{[\alpha]} - g^{[\beta]}$ and assume, for contradiction, that $h \neq 0$. Note that, since $\operatorname{sig}(c\alpha) = \operatorname{sig}(\beta)$, we have $\operatorname{sig}(\gamma) \prec \operatorname{sig}(\alpha)$. Hence, by assumption, $h^{[\gamma]}$ sig-reduces to zero by $G^{[\Sigma]}$. In particular, there exists $p^{[\delta]} \in G^{[\Sigma]}$ with $\operatorname{sig}(\delta) \preceq \operatorname{sig}(\gamma)$ such that $p^{[\delta]}$ top sig-reduces $h^{[\gamma]}$. Without loss of generality, we may assume that $\operatorname{lm}(h)$ comes from a term in $f^{[\alpha]}$. But then $f^{[\alpha]}$ is regular sig-reducible by $p^{[\delta]}$, which is a contradiction.

The proof of the second statement is similar: consider $h^{[\gamma]} = cf^{[\alpha]} - g^{[\beta]}$ and assume, for contradiction, that $\operatorname{lm}(h) = \max\{\operatorname{lm}(f), \operatorname{lm}(g)\}$. Without loss of generality, we may assume that $\operatorname{lm}(h) = \operatorname{lm}(f)$. The rest of the proof is identical: $\operatorname{sig}(\gamma) \prec \operatorname{sig}(\alpha)$, so $h^{[\gamma]}$ sig-reduces to zero, and in particular, it is top sig-reducible. This is a contradiction to $f^{[\alpha]}$ being regular top sig-reduced. □

Although a labelled Gröbner basis $G^{[\Sigma]}$ of $I^{[\Sigma]}$ is not unique in general, we can impose certain additional conditions on $G^{[\Sigma]}$ in order to at least obtain one that is as small as possible. We call such a labelled Gröbner basis *minimal*.

Definition 3.2.11. *A labelled Gröbner basis $G^{[\Sigma]}$ of $I^{[\Sigma]}$ (up to signature $\sigma \in T(\Sigma)$) is called* minimal *if no $g^{[\beta]} \in G^{[\Sigma]}$ is top* sig*-reducible by $G^{[\Sigma]} \setminus \{g^{[\beta]}\}$.*

We also extend this definition to signature Gröbner bases.

Definition 3.2.12. *A signature Gröbner basis $G^{(\Sigma)}$ of $I^{[\Sigma]}$ (up to signature $\sigma \in T(\Sigma)$) is called* minimal *if there exists a minimal labelled Gröbner basis $G^{[\Sigma]}$ of $I^{[\Sigma]}$ (up to signature σ) such that*

$$G^{(\Sigma)} = \left\{ g^{(\mathrm{sig}(\beta))} \mid g^{[\beta]} \in G^{[\Sigma]} \right\}.$$

Note that the definition does not depend on the particular choice of the labelled Gröbner basis, as, in fact, all relevant information to check whether $G^{(\Sigma)}$ is minimal is already contained in $G^{(\Sigma)}$ itself. This follows from the fact that sig-reducibility can be tested for signature polynomials, see also Remark 3.2.2.

A minimal labelled Gröbner basis is minimal in the following sense. In the following, R^* denotes the group of invertible elements of the commutative ring R.

Proposition 3.2.13. *Let $G^{[\Sigma]}$ be a minimal labelled Gröbner basis and $H^{[\Sigma]}$ be a labelled Gröbner basis of $I^{[\Sigma]}$. Then, for every $g^{[\beta]} \in G^{[\Sigma]}$ there exists $h^{[\gamma]} \in H^{[\Sigma]}$ such that*

$$\exists c \in R^* : \mathrm{lt}(g) = \mathrm{lt}(ch) \quad \textit{and} \quad \mathrm{sig}(\beta) \simeq \mathrm{sig}(\gamma).$$

Proof. Let $g^{[\beta]} \in G^{[\Sigma]}$. Since $H^{[\Sigma]}$ is a labelled Gröbner basis, $g^{[\beta]}$ can be sig-reduced to zero by $H^{[\Sigma]}$. In particular, this means that $g^{[\beta]}$ is top sig-reducible by $H^{[\Sigma]}$, that is, there exist $h^{[\gamma]} \in H^{[\Sigma]}$ and $a, b \in [X]\langle Y \rangle$, $c \in R$ such that

$$\mathrm{lt}(g) = \mathrm{lt}(cahb) \quad \text{and} \quad \mathrm{sig}(\beta) \succeq \mathrm{sig}(a\gamma b).$$

Similarly, since $G^{[\Sigma]}$ is also a labelled Gröbner basis, there exist $g'^{[\beta']} \in G^{[\Sigma]}$ and $a', b' \in [X]\langle Y\rangle$, $c' \in R$ such that

$$\mathrm{lt}(h) = \mathrm{lt}(c'a'g'b') \quad \text{and} \quad \mathrm{sig}(\gamma) \succeq \mathrm{sig}(a'\beta'b').$$

Combining these two statements yields

$$\mathrm{lt}(g) = \mathrm{lt}(cahb) = \mathrm{lt}(cc'aa'g'b'b) \quad \text{and} \quad \mathrm{sig}(\beta) \succeq \mathrm{sig}(a\gamma b) \succeq \mathrm{sig}(aa'\beta'b'b).$$

Now, if $g^{[\beta]} \neq g'^{[\beta']}$, then $g'^{[\beta']}$ could be used to top sig-reduce $g^{[\beta]}$, which contradicts the minimality of $G^{[\Sigma]}$. Thus $g^{[\beta]} = g'^{[\beta']}$, which implies that $cc' = a = a' = b = b' = 1$, and therefore,

$$\mathrm{lt}(g) = \mathrm{lt}(ch) \quad \text{and} \quad \mathrm{sig}(\beta) \simeq \mathrm{sig}(\gamma). \qquad \square$$

We note that a labelled module $I^{[\Sigma]}$ always has a minimal labelled Gröbner basis, which is finite if and only if $I^{[\Sigma]}$ has a finite labelled Gröbner basis. This follows from the following proposition, which tells us that we can obtain a minimal basis by removing top sig-reducible elements.

Proposition 3.2.14. *Let $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ be a labelled Gröbner basis of $I^{[\Sigma]}$ such that there exists $g^{[\beta]} \in G^{[\Sigma]}$ which is top* sig*-reducible by $G^{[\Sigma]} \setminus \{g^{[\beta]}\}$. Then $G^{[\Sigma]} \setminus \{g^{[\beta]}\}$ is also a labelled Gröbner basis of $I^{[\Sigma]}$.*

Proof. By assumption, there exists $g'^{[\beta']} \in G^{[\Sigma]} \setminus \{g^{[\beta]}\}$ such that $g^{[\beta]}$ is top sig-reducible by $g'^{[\beta']}$. Then every element which is top sig-reducible by $g^{[\beta]}$ is also top sig-reducible by $g'^{[\beta']}$. Consequently, it follows from Lemma 3.2.7 that $G^{[\Sigma]} \setminus \{g^{[\beta]}\}$ is also a labelled Gröbner basis of $I^{[\Sigma]}$. $\square$

Corollary 3.2.15. *The labelled module $I^{[\Sigma]}$ has a finite labelled Gröbner basis if and only if $I^{[\Sigma]}$ has a finite minimal labelled Gröbner basis. Furthermore, all minimal labelled Gröbner bases of $I^{[\Sigma]}$ are either infinite or have the same finite cardinality.*

Proof. If $I^{[\Sigma]}$ has a finite labelled Gröbner basis $G^{[\Sigma]}$, then, applying Proposition 3.2.14 repeatedly, $G^{[\Sigma]}$ contains a finite minimal labelled Gröbner basis as a subset. The converse is clear.

For the last statement, assume that $G_1^{[\Sigma]}$ and $G_2^{[\Sigma]}$ are minimal labelled Gröbner bases of $I^{[\Sigma]}$. If they are both infinite, there is nothing to show. Otherwise, assume that $G_2^{[\Sigma]}$ is finite. We show that $|G_1^{[\Sigma]}| \leq |G_2^{[\Sigma]}|$. To this end, assume, for contradiction, that $|G_1^{[\Sigma]}| > |G_2^{[\Sigma]}|$. Then, by Proposition 3.2.13 and the pigeonhole principle, there exist distinct $g_1^{[\beta_1]}, g_2^{[\beta_2]} \in G_1^{[\Sigma]}$ and $h^{[\gamma]} \in G_2^{[\Sigma]}$ such that

$$\mathrm{lt}(g_1) = \mathrm{lt}(c_1 h), \qquad \mathrm{lt}(g_2) = \mathrm{lt}(c_2 h), \qquad \mathrm{sig}(\beta_1) \simeq \mathrm{sig}(\gamma) \simeq \mathrm{sig}(\beta_2),$$

with invertible $c_1, c_2 \in R$. Rearranging these equations shows that $\mathrm{lt}(g_1) = \mathrm{lt}(c_1 c_2^{-1} g_2)$ and $\mathrm{sig}(\beta_1) \simeq \mathrm{sig}(\beta_2)$, and consequently, that $g_1^{[\beta_1]}$ is top sig-reducible by $g_2^{[\beta_2]}$, which contradicts the minimality of $G_1^{[\Sigma]}$. Analogously, one can show that $|G_2^{[\Sigma]}| \leq |G_1^{[\Sigma]}|$, and the result follows. □

Assuming that $|Y| > 1$, we cannot expect all labelled modules to have a finite (minimal) labelled Gröbner basis, as there are finitely generated ideals in $\mathcal{A}$ that simply do not have a finite Gröbner basis, and, consequently, also no finite labelled Gröbner basis. Unfortunately, the condition that an ideal $I = (f_1, \ldots, f_r)$ has a finite Gröbner basis is also not sufficient to ensure that the labelled module generated by $f_1, \ldots, f_r$ has a finite labelled Gröbner basis as the following example shows.

Example 3.2.16. *We give an example of a family of generators whose ideal admits a finite Gröbner basis, but whose labelled module does not admit a finite labelled Gröbner basis. The construction and the proof of the claims rely on notions introduced in Section 3.3, and will be deferred until that point.*

Let K be a field. We consider the family $(f, g_0) \in K\langle x, y\rangle^2$ with

$$f = yx - xy, \qquad g_0 = xy - 1.$$

Using $\preceq_{\mathrm{deglex}}$ as a monomial order with $x \prec_{\mathrm{lex}} y$ and $\preceq_{\mathrm{DoPoT}}$ as a module order, the set $G = \{f, g_0\}$ is already a Gröbner basis of the ideal $I = (f, g_0)$.

A minimal labelled Gröbner basis of the module $I^{[\Sigma]}$ generated by f, g_0 is given by the infinite set

$$G^{[\Sigma]} = \left\{ f^{[\varepsilon_1]}, g_0^{[\varepsilon_2]} \right\} \cup \left\{ g_n^{[\beta_n]} \mid n \in \mathbb{N}_{>0} \right\},$$

with $g_n = x^{n+1}y - x^n$ and certain $\beta_n \in \Sigma$ with $\mathrm{sig}(\beta_n) = \varepsilon_2 x^n$ for $n > 0$.

So, Corollary 3.2.15 implies that $I^{[\Sigma]}$ does not have a finite labelled Gröbner basis. The obstruction to having a finite labelled Gröbner basis is that $g_0^{[\varepsilon_2]}$ cannot be used to sig*-reduce any of the elements $g_n^{[\beta_n]}$ for $n > 0$. Indeed, since $x^n\varepsilon_2 \succ \varepsilon_2 x^n = \mathrm{sig}(\beta_n)$, the reductions would cause the signatures to increase, and would not be* sig*-reductions.*

If we exchange the order of f and g_0, and consider the family of generators g_0, f generating the module $I^{[\Sigma']}$, where $\Sigma' = (K\langle X\rangle \otimes_K K\langle X\rangle)^{(\mathcal{D})}$ denotes the free $K\langle X\rangle$-bimodule on the set $\mathcal{D} = \{\delta_1, \delta_2\}$, then the finite set

$$G^{[\Sigma']} = \left\{g_0^{[\delta_1]}, f^{[\delta_2]}\right\}$$

is a minimal labelled Gröbner basis of $I^{[\Sigma']}$. The difference is that $g_0^{[\delta_1]}$ has now signature δ_1 instead of ε_2, which causes the elements $g_n^{[\beta_n]}$, for $n > 0$, to have signature $\mathrm{sig}(\beta_n) = x^n\delta_2$ and renders them all top sig*-reducible by $g_0^{[\delta_1]}$.*

Remark 3.2.17. *For a fixed monomial order, having a finite Gröbner basis is a property of an ideal, independently of its generators and their order, see also [Mor94, Sec. 6] for further information. By contrast, exchanging the order of the generators or using different generating sets for the same ideal can affect the finiteness of its (minimal) labelled Gröbner bases. Here, the ideal spanned is the same, but the underlying module is different. In general, different choices of generators of an ideal I can lead to drastically different module structures for $I^{[\Sigma]}$, and thus, to different (minimal) labelled Gröbner bases.*

3.3 Computations in the free algebra

Before discussing the computation of labelled Gröbner bases in the more general setting of the mixed algebra, we first present the theory in the easier case of the free algebra over a coefficient field. This shall help, on the one hand, to get familiar with the relevant notions and the algorithms in a simpler setting, before transitioning to full generality. Furthermore, the free algebra over a coefficient field is probably also the most relevant setting, making it worth discussing this case separately.

We note that all results in this section only depend on *similarity* of signatures, that is, only equality up to coefficients is required. This is a consequence of the fact that we are working over a coefficient field here. Consequently, in this setting, one could define

signatures without coefficients, as done in [HV22], for example. For the sake of consistency, we, however, stick to Definition 3.1.15 and consider signatures with coefficients.

In this section, we fix a monomial order $\preceq$ on $\langle X\rangle$ and a module order $\preceq_\Sigma$ on $M(\Sigma)$. We assume that the two orders are *compatible* in the sense that, for all $a, b \in \langle X\rangle$ and $i \in \{1, \dots, r\}$, we have

$$a \prec b \iff a\varepsilon_i \prec_\Sigma b\varepsilon_i \iff \varepsilon_i a \prec_\Sigma \varepsilon_i b.$$

We will denote both orders by the same symbol $\preceq$. As before, this shall cause no confusion as module elements will be denoted by Greek letters and polynomials by Roman letters.

Example 3.3.1. *All the module orders in Example 3.1.14 are compatible with the underlying monomial order that is used for the monomial comparisons.*

3.3.1 Computation of labelled Gröbner bases

The objective of this section is to state an adaptation of the noncommutative version of Buchberger's algorithm (Algorithm 1) to include signatures. To this end, we need to adapt the notion of ambiguities and S-polynomials to the case of labelled polynomials.

Regular ambiguities

We first extend the notion of ambiguities from Definition 2.4.46 from polynomials to labelled polynomials. To this end, we fix a family of polynomials $(f_1, \dots, f_r) \in K\langle X\rangle^r$ generating a labelled module $I^{[\Sigma]}$ and note that, in this setting, the free bimodule is simply

$$\Sigma = (K\langle X\rangle \otimes_K K\langle X\rangle)^{(\mathcal{E})},$$

with basis $\mathcal{E} = \{\varepsilon_1, \dots, \varepsilon_r\}$.

Definition 3.3.2. *For $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ with $f, g \neq 0$ an* (overlap/inclusion) ambiguity *of $f^{[\alpha]}$ and $g^{[\beta]}$ is*

$$(a \otimes b, c \otimes d, f^{[\alpha]}, g^{[\beta]}),$$

where $(a \otimes b, c \otimes d, f, g)$ *is an (overlap/inclusion) ambiguity of the polynomials* f *and* g*. We denote by* $\operatorname{amb}(f^{[\alpha]}, g^{[\beta]})$ *the set of all ambiguities of* $f^{[\alpha]}$ *and* $g^{[\beta]}$*. Furthermore, for* $G^{[\Sigma]} \subseteq I^{[\Sigma]}$*, let*

$$\operatorname{amb}(G^{[\Sigma]}) \coloneqq \bigcup_{\substack{f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]} \\ f,g \neq 0}} \operatorname{amb}(f^{[\alpha]}, g^{[\beta]}).$$

Like for polynomials, when clear by context, we drop $f^{[\alpha]}$ and $g^{[\beta]}$ from an ambiguity and simply write $(a \otimes b, c \otimes d) \in \operatorname{amb}(f^{[\alpha]}, g^{[\beta]})$. As ambiguities of labelled polynomials are the same as ambiguities of classical polynomials, we have the same relationship $\operatorname{lm}(afb) = \operatorname{lm}(cgb)$ for ambiguities $(a \otimes b, c \otimes d, f^{[\alpha]}, g^{[\beta]})$ of labelled polynomials that we also have for classical polynomials (see Lemma 2.4.48). Hence, we define the leading monomial of an ambiguity analogously, and additionally, we now also define the *signature* of an ambiguity.

Definition 3.3.3. *Let* $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ *be such that* $f, g \neq 0$ *and let* $\mathfrak{a} = (a \otimes b, c \otimes d) \in \operatorname{amb}(f^{[\alpha]}, g^{[\beta]})$*. The* leading monomial *of* $\mathfrak{a}$ *is* $\operatorname{LM}(\mathfrak{a}) \coloneqq \operatorname{lm}(afb)$ *and the* signature *of* $\mathfrak{a}$ *is*

$$\operatorname{SIG}(\mathfrak{a}) \coloneqq \max\left\{\operatorname{sig}(a\alpha b), \operatorname{sig}(c\beta d)\right\},$$

choosing the first if $\operatorname{sig}(a\alpha b) \simeq \operatorname{sig}(c\beta d)$*. Furthermore,* $\mathfrak{a}$ *is called* regular *if* $\operatorname{sig}(a\alpha b) \not\simeq \operatorname{sig}(c\beta d)$ *and* singular *otherwise, that is, if* $\operatorname{sig}(a\alpha b) \simeq \operatorname{sig}(c\beta d)$*.*

Example 3.3.4. *Let* K *be a field. We consider the labelled module* $I^{[\Sigma]}$ *generated by the following family of polynomials in* $K\langle a, b, c, d\rangle$*:*

$$f_1 = aba - a, \qquad f_2 = ab - cd, \qquad f_3 = ba - dc.$$

We use $\preceq_{\text{deglex}}$ *as a monomial order with* $a \prec_{\text{lex}} b \prec_{\text{lex}} c \prec_{\text{lex}} d$ *and* $\preceq_{\text{DoPoT}}$ *as a module order.*

The labelled polynomial $f_1^{[\varepsilon_1]}$ *forms an (overlap) ambiguity with itself:*

$$\mathfrak{a} = (1 \otimes ba, ab \otimes 1, f_1^{[\varepsilon_1]}, f_1^{[\varepsilon_1]}).$$

Note that $\operatorname{LM}(\mathfrak{a}) = ababa$ *and* $\operatorname{SIG}(\mathfrak{a}) = ab\varepsilon_1 \succ \varepsilon_1 ba$*, showing that* $\mathfrak{a}$ *is regular.*

Furthermore, $I^{[\Sigma]}$ *contains the labelled polynomials*

$$f^{[\alpha]} = cba - abc^{[c\varepsilon_3 - \varepsilon_2 c]} \qquad g^{[\beta]} = bada - da^{[\varepsilon_3 da - d\varepsilon_2 a + d\varepsilon_1]},$$

which form the (overlap) ambiguity

$$\mathfrak{a}' = (1 \otimes da, c \otimes 1, f^{[\alpha]}, g^{[\beta]}).$$

Note that $\mathfrak{a}'$ *is singular since* $\mathrm{SIG}(\mathfrak{a}') = \alpha da = c\beta = c\varepsilon_3 da$.

Definition 3.3.5. *Let* $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ *and* $\mathfrak{a} = (a \otimes b, c \otimes d, f^{[\alpha]}, g^{[\beta]}) \in \mathrm{amb}(G^{[\Sigma]})$. *The* S-polynomial *of* $\mathfrak{a}$ *is*

$$\text{S-Pol}(\mathfrak{a}) \coloneqq \frac{1}{\mathrm{lc}(f)} a f^{[\alpha]} b - \frac{1}{\mathrm{lc}(g)} c g^{[\beta]} d.$$

Remark 3.3.6. *Ambiguities of signature polynomials can be defined analogously to ambiguities of labelled polynomials. Furthermore, for regular ambiguities, it is possible to compute the S-polynomial as a signature polynomial (in particular, it is possible to compute its signature).*

Recall that the signature of a labelled polynomial is simply the signature of its module part. With this, note that, if $\mathfrak{a}$ is a regular ambiguity, then $\mathrm{SIG}(\mathfrak{a}) \simeq \mathrm{sig}(\text{S-Pol}(\mathfrak{a}))$.

The following lemma asserts that any situation, where multiples of two labelled polynomials share a common leading monomial but differ in their signatures, can be characterised by a regular ambiguity or a trivial syzygy between the two elements. This lemma is technical and useful for the rest of the proofs. It ensures that it suffices to consider regular ambiguities.

Lemma 3.3.7. *Let* $g_1^{[\beta_1]}, g_2^{[\beta_2]} \in I^{[\Sigma]}$ *be such that* $g_1, g_2 \neq 0$ *and let* $a_i, b_i \in \langle X \rangle$, $i = 1, 2$, *such that*

$$\mathrm{lm}(a_1 g_1 b_1) = \mathrm{lm}(a_2 g_2 b_2) \quad \textit{and} \quad \mathrm{sig}(a_1 \beta_1 b_1) \succ \mathrm{sig}(a_2 \beta_2 b_2).$$

Then there exist $a_3, b_3 \in \langle X \rangle$ *such that one of the following conditions holds:*

1. *there exists a trivial syzygy* π *between* $g_1^{[\beta_1]}$ *and* $g_2^{[\beta_2]}$ *such that* $\mathrm{sig}(a_3 \pi b_3) = \mathrm{sig}(a_1 \beta_1 b_1)$*;*
2. *there exists a regular ambiguity* $\mathfrak{a} \in \mathrm{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$ *with* $a_3 \mathrm{LM}(\mathfrak{a}) b_3 = \mathrm{lm}(a_1 g_1 b_1)$ *and* $a_3 \mathrm{SIG}(\mathfrak{a}) b_3 = \mathrm{sig}(a_1 \beta_1 b_1)$*;*

Proof. We distinguish between different cases (see Figure 3.1), depending on the position of $\mathrm{lm}(g_1)$ and $\mathrm{lm}(g_2)$ relative to each other in $W = \mathrm{lm}(a_1 g_1 b_1) = \mathrm{lm}(a_2 g_2 b_2)$.

Case 1 $\mathrm{lm}(g_1)$ is fully contained in a_2 or b_2, or equivalently, $\mathrm{lm}(g_1)$ and $\mathrm{lm}(g_2)$ do not overlap in W. We first consider the case where $\mathrm{lm}(g_1)$ is contained in a_2, that is, where $W = A\mathrm{lm}(g_1)B\mathrm{lm}(g_2)C$ for some $A, B, C \in \langle X \rangle$. Then $\pi = \beta_1 B g_2 - g_1 B \beta_2$ is a trivial syzygy between $g_1^{[\beta_1]}$ and $g_2^{[\beta_2]}$. To prove the assertion regarding the signatures, note that $\mathrm{sig}(\pi) = \mathrm{sig}(\beta_1 B g_2) \succ \mathrm{sig}(g_1 B \beta_2)$ as $A\mathrm{sig}(\beta_1 B g_2)C = \mathrm{sig}(A\beta_1 B\mathrm{lm}(g_2)C) = \mathrm{sig}(a_1\beta_1 b_1) \succ \mathrm{sig}(a_2\beta_2 b_2) = \mathrm{sig}(A\mathrm{lm}(g_1)B\beta_2 C) = A\mathrm{sig}(g_1 B\beta_2)C$. Hence, with $a_3 = A$ and $b_3 = C$, we obtain $\mathrm{sig}(a_3 \pi b_3) = \mathrm{sig}(A\pi C) = \mathrm{sig}(A\beta_1 B g_2 C) = \mathrm{sig}(a_1\beta_1 b_1)$.

The other case, where $\mathrm{lm}(g_1)$ is contained in b_2, works along the same lines using the trivial syzygy $\pi = g_2 B\beta_1 - \beta_2 B g_1$.

Case 2 $\mathrm{lm}(g_1)$ is fully contained in $\mathrm{lm}(g_2)$, that is, $A\mathrm{lm}(g_1)C = \mathrm{lm}(g_2)$ for some $A, C \in \langle X \rangle$. In this case, there exists an inclusion ambiguity $\mathfrak{a} = (A \otimes C, 1 \otimes 1) \in \mathrm{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$. Note that $\mathrm{SIG}(\mathfrak{a}) = \mathrm{sig}(A\beta_1 C) \succ \mathrm{sig}(\beta_2)$ as $a_2\mathrm{sig}(A\beta_1 C)b_2 = \mathrm{sig}(a_1\beta_1 b_1) \succ \mathrm{sig}(a_2\beta_2 b_2) = a_2\mathrm{sig}(\beta_2)b_2$. Hence, $\mathfrak{a}$ is regular and, with $a_3 = a_2$ and $b_3 = b_2$, we have $a_3\mathrm{SIG}(\mathfrak{a})b_3 = \mathrm{sig}(a_2 A\beta_1 C b_2) = \mathrm{sig}(a_1\beta_1 b_1)$ and $a_3\mathrm{LM}(\mathfrak{a})b_3 = \mathrm{lm}(a_2 g_2 b_2) = \mathrm{lm}(a_1 g_1 b_1)$.

Case 3 $\mathrm{lm}(g_2)$ is fully contained in $\mathrm{lm}(g_1)$, that is, $\mathrm{lm}(g_1) = A\mathrm{lm}(g_2)C$ for some $A, C \in \langle X \rangle$. In this case, there exists an inclusion ambiguity $\mathfrak{a} = (1 \otimes 1, A \otimes C) \in \mathrm{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$. Note that $\mathrm{SIG}(\mathfrak{a}) = \mathrm{sig}(\beta_1) \succ \mathrm{sig}(A\beta_2 C)$ as $a_1\mathrm{sig}(\beta_1)b_1 = \mathrm{sig}(a_1\beta_1 b_1) \succ \mathrm{sig}(a_2\beta_2 b_2) = a_1\mathrm{sig}(A\beta_2 C)b_1$. Hence, $\mathfrak{a}$ is regular and, with $a_3 = a_1$ and $b_3 = b_1$, we have $a_3\mathrm{SIG}(\mathfrak{a})b_3 = \mathrm{sig}(a_1\beta_1 b_1)$ and $a_3\mathrm{LM}(\mathfrak{a})b_3 = \mathrm{lm}(a_1 g_1 b_1)$.

Case 4 $\mathrm{lm}(g_1)$ and $\mathrm{lm}(g_2)$ overlap but are not fully contained in one another and $\mathrm{lm}(g_1)$ begins before $\mathrm{lm}(g_2)$, that is, $\mathrm{lm}(g_1)C = A\mathrm{lm}(g_2)$ for some $A, C \in \langle X \rangle$. In this case, there exists an overlap ambiguity $\mathfrak{a} = (1 \otimes C, A \otimes 1) \in \mathrm{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$. Note that $\mathrm{SIG}(\mathfrak{a}) = \mathrm{sig}(\beta_1 C) \succ \mathrm{sig}(A\beta_2)$ as $a_1\mathrm{sig}(\beta_1 C)b_2 = \mathrm{sig}(a_1\beta_1 b_1) \succ \mathrm{sig}(a_2\beta_2 b_2) = a_1\mathrm{sig}(A\beta_2)b_2$. Hence, $\mathfrak{a}$ is regular and, with $a_3 = a_1$ and $b_3 = b_2$, we have $a_3\mathrm{SIG}(\mathfrak{a})b_3 = \mathrm{sig}(a_1\beta_1 C b_2) = \mathrm{sig}(a_1\beta_1 b_1)$ and $a_3\mathrm{LM}(\mathfrak{a})b_3 = \mathrm{lm}(a_1 g_1 C b_2) = \mathrm{lm}(a_1 g_1 b_1)$.

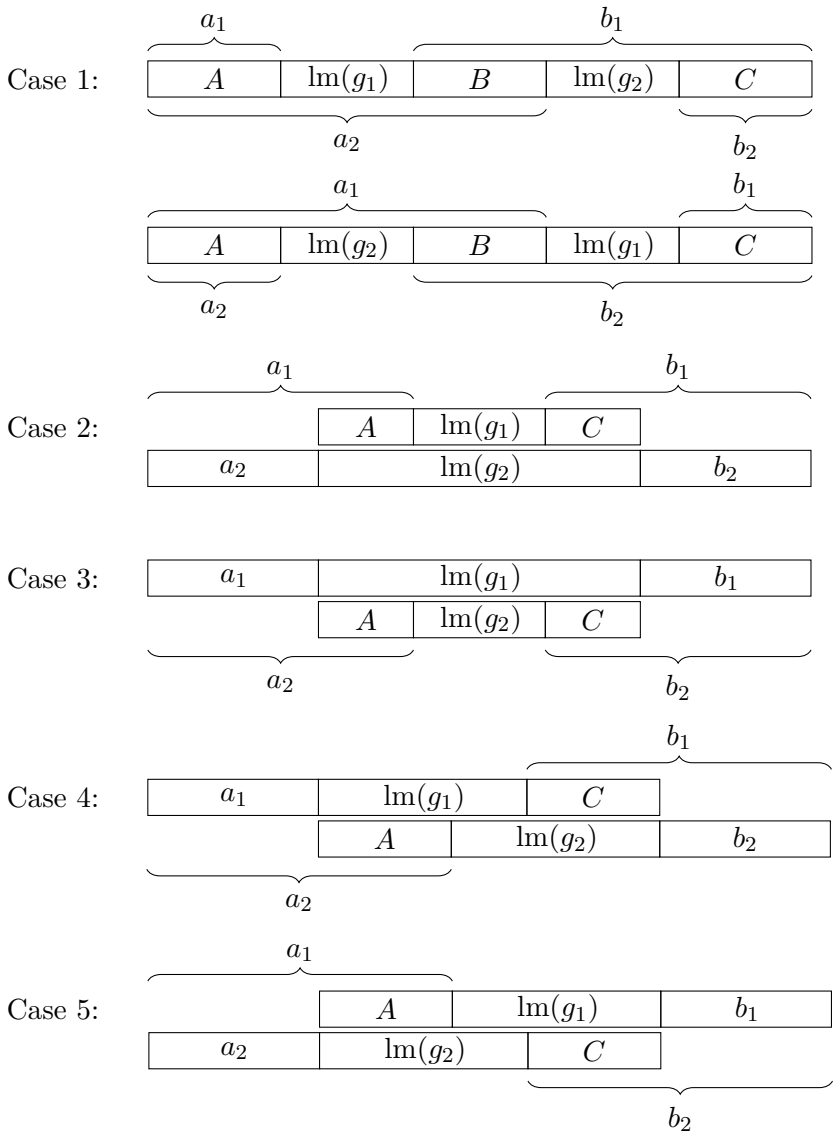


Figure 3.1: Relative position of $\mathrm{lm}(g_1)$ and $\mathrm{lm}(g_2)$ in the proof of Lemma 3.3.7.

Case 5 $\mathrm{lm}(g_1)$ and $\mathrm{lm}(g_2)$ overlap but are not fully contained in one another and $\mathrm{lm}(g_1)$ begins after $\mathrm{lm}(g_2)$, that is, $A\mathrm{lm}(g_1) = \mathrm{lm}(g_2)C$ for some $A, C \in \langle X\rangle$. In this case, there exists an overlap ambiguity $\mathfrak{a} = (A \otimes 1, 1 \otimes C) \in \mathrm{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$. Note that $\mathrm{SIG}(\mathfrak{a}) = \mathrm{sig}(A\beta_1) \succ \mathrm{sig}(\beta_2 C)$ as $a_2\mathrm{sig}(A\beta_1)b_1 = \mathrm{sig}(a_1\beta_1 b_1) \succ \mathrm{sig}(a_2\beta_2 b_2) = a_2\mathrm{sig}(\beta_2 C)b_1$. Hence, $\mathfrak{a}$ is regular and, with $a_3 = a_2$ and $b_3 = b_1$, we have $a_3\mathrm{SIG}(\mathfrak{a})b_3 = \mathrm{sig}(a_2 A\beta_1 b_1) = \mathrm{sig}(a_1\beta_1 b_1)$ and $a_3\mathrm{LM}(\mathfrak{a})b_3 = \mathrm{lm}(a_2 A g_1 b_1) = \mathrm{lm}(a_1 g_1 b_1)$. □

Remark 3.3.8. *The case distinction in the proof of Lemma 3.3.7 can be compared to the one in the proof of Theorem 2.4.54; both have to consider the same cases. The only difference is that in the latter we can rely more on symmetries eliminating some cases immediately. This does not work in the former because of the (asymmetric) assertion regarding the signatures.*

Cover criterion

In Section 2.4.3, we have seen that the weakest property that allows to characterise classical Gröbner bases in the free algebra is resolvability relative to a monomial order $\preceq$ (see Definition 2.4.52). In the setting with signatures, a similar characterisation of labelled Gröbner bases can be given by the notion of *covered* ambiguities. We define this property as a straightforward adaptation of the notion introduced in [GVW16].

Definition 3.3.9. *Let $G^{[\Sigma]} \subseteq I^{[\Sigma]}$, $H \subseteq \mathrm{Syz}(I^{[\Sigma]})$, and $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ with $f, g \neq 0$. An ambiguity $\mathfrak{a} \in \mathrm{amb}(f^{[\alpha]}, g^{[\beta]})$ is* covered *by $(G^{[\Sigma]}, H)$ if there exist $h^{[\delta]} \in G^{[\Sigma]} \cup \{0^{[\gamma]} \mid \gamma \in H\}$ and $a, b \in \langle X\rangle$ such that the following conditions hold:*

- $\mathrm{LM}(\mathfrak{a}) \succ \mathrm{lm}(ahb)$*;*
- $\mathrm{SIG}(\mathfrak{a}) \simeq \mathrm{sig}(a\delta b)$*;*

In the definition, h can also be zero, in which case the first condition is always fulfilled since $\mathrm{lm}(0) = 0 \prec w$ for all $w \in \langle X\rangle$. Also, note that only similarity of signatures is required.

Remark 3.3.10. *The property of being covered only depends on information encoded in the polynomial part and the signature of the labelled polynomials, and thus, can be generalised directly to signature polynomials.*

The following theorem extends the characterisation of signature Gröbner bases provided in [HV22, Thm. 39] (item 2 below) by a noncommutative version of the GVW *cover criterion* [GVW16, Thm. 2.4] (item 3 below).

In the following, we say that an element $f^{[\alpha]} \in I^{[\Sigma]}$ is *singular top* sig-*reducible* by a syzygy $\gamma \in \mathrm{Syz}(I^{[\Sigma]})$ if there exist $a, b \in \langle X \rangle$ such that $\mathrm{sig}(\alpha) \simeq \mathrm{sig}(a\gamma b)$.

Theorem 3.3.11. *Let $\sigma \in T(\Sigma)$, $G^{[\Sigma]} \subseteq I^{[\Sigma]}$, and $H \subseteq \mathrm{Syz}(I^{[\Sigma]})$ be such that the following conditions hold:*

- *for all $g^{[\beta]} \in G^{[\Sigma]} : g \neq 0$;*
- *for all $\varepsilon_i \prec \sigma$, there exists $\beta_i \in H \cup \{\beta \mid g^{[\beta]} \in G^{[\Sigma]}\}$ with $\mathrm{sig}(\beta_i) \simeq \varepsilon_i$;*
- *$\{\gamma \in \mathrm{Triv}(G^{[\Sigma]}) \mid \mathrm{sig}(\gamma) \prec \sigma\} \subseteq H$;*

Then the following are equivalent:

1. *$G^{[\Sigma]}$ is a labelled Gröbner basis of $I^{[\Sigma]}$ up to signature σ and H is a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ up to signature σ;*
2. *the S-polynomials of all regular ambiguities $\mathfrak{a}$ of $G^{[\Sigma]}$ with $\mathrm{SIG}(\mathfrak{a}) \prec \sigma$ regular* sig-*reduce by $G^{[\Sigma]}$ to elements which are singular top* sig-*reducible by $G^{[\Sigma]}$ or by H;*
3. *all regular ambiguities $\mathfrak{a}$ of $G^{[\Sigma]}$ with $\mathrm{SIG}(\mathfrak{a}) \prec \sigma$ are covered by $(G^{[\Sigma]}, H)$;*

Remark 3.3.12. *The notion of being singular top* sig-*reducible is equivalent to what is in the (commutative) literature also called* sig-redundant *(see [EP11]) and included in the concept of* super top reductions *in [GVW16]. Additionally, a regular* sig-*reduced element being singular top* sig-*reducible corresponds to the notion of not being* primitive sig-irreducible *in [AP11].*

Proof of Theorem 3.3.11. $1 \implies 2$ Let $\mathfrak{a} \in \mathrm{amb}(G^{[\Sigma]})$ be regular with $\mathrm{SIG}(\mathfrak{a}) \prec \sigma$ and let $p^{[\pi]}$ be the result of regular sig-reducing S-Pol$(\mathfrak{a})$ by $G^{[\Sigma]}$. Note that $\mathrm{sig}(\pi) \simeq \mathrm{SIG}(\mathfrak{a}) \prec \sigma$. If $p = 0$, then π is a syzygy of $I^{[\Sigma]}$. Since H is a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ up to signature σ by assumption, there exist $\gamma \in H$ and $a, b \in \langle X \rangle$ such that $\mathrm{sig}(\pi) \simeq \mathrm{sig}(a\gamma b)$, showing that $p^{[\pi]}$ is singular top sig-reducible by H. If $p \neq 0$, then $p^{[\pi]}$ is singular top sig-reducible by $G^{[\Sigma]}$ as $G^{[\Sigma]}$ is a labelled Gröbner basis up to signature σ.

$2 \implies 3$ Let $\mathfrak{a} \in \operatorname{amb}(G^{[\Sigma]})$ be regular with $\operatorname{SIG}(\mathfrak{a}) \prec \sigma$ and let $p^{[\pi]}$ be the result of regular sig-reducing S-Pol($\mathfrak{a}$) by $G^{[\Sigma]}$. Note that $\operatorname{sig}(\pi) \simeq \operatorname{SIG}(\mathfrak{a}) \prec \sigma$. By assumption $p^{[\pi]}$ is singular top sig-reducible by some element in $G^{[\Sigma]}$ or H but then $\mathfrak{a}$ is covered by the same element as $\operatorname{LM}(\mathfrak{a}) \succ \operatorname{lm}(p)$.

$3 \implies 1$ The proof follows the same structure as the proof of [GVW16, Thm. 2.4].

Assume, for contradiction, that 1 is wrong. Then there exists $f^{[\alpha]} \in I^{[\Sigma]}$ with $\operatorname{sig}(\alpha) \prec \sigma$ such that either $f \neq 0$ and $f^{[\alpha]}$ is not sig-reducible by $G^{[\Sigma]}$ or $f = 0$ and α is not reducible by H. Choose such $f^{[\alpha]}$ with minimal signature $\operatorname{sig}(\alpha)$. Note that this means that $G^{[\Sigma]}$ is a labelled Gröbner basis up to signature $\operatorname{sig}(\alpha)$.

By the second assumption, there exist $a_1, b_1 \in \langle X \rangle$ and $g_1^{[\beta_1]} \in G^{[\Sigma]} \cup \{0^{[\gamma]} \mid \gamma \in H\}$ with $\operatorname{sig}(a_1\beta_1 b_1) \simeq \operatorname{sig}(\alpha)$. We select these elements so that $\operatorname{lm}(a_1 g_1 b_1)$ is minimal and claim that $a_1 g_1^{[\beta_1]} b_1$ is not regular top sig-reducible by $G^{[\Sigma]}$.

To prove this claim, suppose that $a_1 g_1^{[\beta_1]} b_1$ is regular top sig-reducible by $g_2^{[\beta_2]} \in G^{[\Sigma]}$. Note that this implies that both elements have nonzero polynomial part. By Lemma 3.3.7, there exist $a_3, b_3 \in \langle X \rangle$ and either a trivial syzygy or a regular ambiguity between $g_1^{[\beta_1]}$ and $g_2^{[\beta_2]}$ satisfying certain conditions. We distinguish between the two possible cases.

Trivial syzygy Let π be the trivial syzygy such that $\operatorname{sig}(a_3 \pi b_3) = \operatorname{sig}(a_1\beta_1 b_1)$. As $\operatorname{sig}(\alpha) \simeq \operatorname{sig}(a_1\beta_1 b_1)$, this implies that α is reducible by $\pi \in \{\gamma \in \operatorname{Triv}(G^{[\Sigma]}) \mid \operatorname{sig}(\gamma) \prec \sigma\} \subseteq H$ – a contradiction.

Regular ambiguity Let $\mathfrak{a} \in \operatorname{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$ be regular such that

$$a_3 \operatorname{LM}(\mathfrak{a}) b_3 = \operatorname{lm}(a_1 g_1 b_1) \quad \text{and} \quad a_3 \operatorname{SIG}(\mathfrak{a}) b_3 = \operatorname{sig}(a_1 \beta_1 b_1). \tag{3.2}$$

Since $\operatorname{SIG}(\mathfrak{a}) \preceq \operatorname{sig}(a_1\beta_1 b_1) \prec \sigma$, the ambiguity $\mathfrak{a}$ is covered by $(G^{[\Sigma]}, H)$. So there exist $h^{[\delta]} \in G^{[\Sigma]} \cup \{0^{[\gamma]} \mid \gamma \in H\}$ and $a, b \in \langle X \rangle$ such that $\operatorname{LM}(\mathfrak{a}) \succ \operatorname{lm}(ahb)$ and $\operatorname{SIG}(\mathfrak{a}) \simeq \operatorname{sig}(a\delta b)$. Combining this with (3.2) yields the following contradiction to the minimality of $\operatorname{lm}(a_1 g_1 b_1)$:

$$\begin{aligned} &\operatorname{lm}(a_3 a h b b_3) \prec a_3 \operatorname{LM}(\mathfrak{a}) b_3 = \operatorname{lm}(a_1 g_1 b_1), \\ &\operatorname{sig}(a_3 a \delta b b_3) \simeq a_3 \operatorname{SIG}(\mathfrak{a}) b_3 = \operatorname{sig}(a_1 \beta_1 b_1) \simeq \operatorname{sig}(\alpha). \end{aligned}$$

Consequently, both $f^{[\alpha]}$ and $a_1 g_1^{[\beta]} b_1$ are not regular top sig-reducible by $G^{[\Sigma]}$. Then Lemma 3.2.10 yields $\mathrm{lm}(f) = \mathrm{lm}(a_1 g_1 b_1)$, showing that $f^{[\alpha]}$ is (singular) top sig-reducible by $g_1^{[\beta_1]}$ – a contradiction. □

If the conditions of Theorem 3.3.11 hold for all signatures, we have a complete labelled Gröbner basis and a complete Gröbner basis of the syzygy module.

Corollary 3.3.13. *Let $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ and $H \subseteq \mathrm{Syz}(I^{[\Sigma]})$ be such that the following conditions hold:*

- *for all $g^{[\beta]} \in G^{[\Sigma]} : g \neq 0$;*
- *for all ε_i, there exists $\beta_i \in H \cup \{\beta \mid g^{[\beta]} \in G^{[\Sigma]}\}$ with $\mathrm{sig}(\beta_i) \simeq \varepsilon_i$;*
- *$\mathrm{Triv}(G^{[\Sigma]}) \subseteq H$;*

Then $G^{[\Sigma]}$ is a labelled Gröbner basis of $I^{[\Sigma]}$ and H is a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ if and only if all regular ambiguities of $G^{[\Sigma]}$ are covered by $(G^{[\Sigma]}, H)$.

Proof. Note that $G^{[\Sigma]}$ is a labelled Gröbner basis of $I^{[\Sigma]}$ and H is a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ if and only if they are a (labelled) Gröbner basis up to signature σ for all $\sigma \in T(\Sigma)$. With this, the result follows from Theorem 3.3.11. □

Example 3.2.16 (continuing from p. 113)**.** *Recall that in Example 3.2.16 we considered the family $(f, g_0) \in K\langle x, y\rangle^2$ with*

$$f = yx - xy, \qquad g_0 = xy - 1.$$

over a field K. We used $\preceq_{\mathrm{deglex}}$, where $x \prec_{\mathrm{lex}} y$, as a monomial order and $\preceq_{\mathrm{DoPoT}}$ as a module order.

We claimed that a minimal labelled Gröbner basis of the labelled module $I^{[\Sigma]}$ generated by f, g_0, is given by

$$G^{[\Sigma]} = \left\{ f^{[\varepsilon_1]}, g_0^{[\varepsilon_2]} \right\} \cup \left\{ g_n^{[\beta_n]} \mid n \in \mathbb{N}_{>0} \right\},$$

with $g_n = x^{n+1}y - x^n$ and certain $\beta_n \in \Sigma$ with $\mathrm{sig}(\beta_n) = \varepsilon_2 x^n$ for $n > 0$. We now prove this claim. To this end, we let $\beta_0 = \varepsilon_2$ in the following.

First, for the minimality of $G^{[\Sigma]}$*, observe that the only possible top reductions would be using* $g_m^{[\beta_m]}$ *to reduce* $g_n^{[\beta_n]}$ *for* $m < n$*. But as* $x^{m-n}\mathrm{sig}(\beta_n) = x^{m-n}\varepsilon_2 x^n \succ \varepsilon_2 x^m = \mathrm{sig}(\beta_m)$*, those reductions would not be* sig*-reductions. Therefore, none of the elements of* $G^{[\Sigma]}$ *is top* sig*-reducible by the others, and* $G^{[\Sigma]}$ *is minimal.*

Now, we prove that $G^{[\Sigma]}$ *is indeed a labelled Gröbner basis of* $I^{[\Sigma]}$ *using Corollary 3.3.13. The first two hypotheses of the theorem are clearly satisfied. Then we verify that all regular ambiguities of* $G^{[\Sigma]}$ *are covered by* $G^{[\Sigma]}$ *and the set* $H = \{\gamma\} \cup \mathrm{Triv}(G^{[\Sigma]})$ *of syzygies, where* $\gamma = y\varepsilon_2 - \varepsilon_2 y - \varepsilon_1 y$*.*

We have, for all $n \in \mathbb{N}$*, the following regular ambiguities* $\mathfrak{a}_{f,n}$ *and* $\mathfrak{a}_{n,f}$ *between* f *and* g_n*:*

$$\mathfrak{a}_{f,n} = \left(1 \otimes x^n y,\, y \otimes 1,\, f_1^{[\varepsilon_1]},\, g_n^{[\beta_n]}\right),$$
$$\mathfrak{a}_{n,f} = \left(1 \otimes x,\, x^{n+1} \otimes 1,\, g_n^{[\beta_n]},\, f_1^{[\varepsilon_1]}\right).$$

Note that $\mathrm{SIG}(\mathfrak{a}_{f,n}) = y\mathrm{sig}(\beta_n) = y\varepsilon_2 x^n$*. Thus, each of these ambiguities is covered by the syzygy* $\gamma \in H$ *as* $\mathrm{sig}(\gamma) = y\varepsilon_2$*. Furthermore, we have* $\mathrm{SIG}(\mathfrak{a}_{n,f}) = \mathrm{sig}(\beta_n)x = \mathrm{sig}(\beta_{n+1})$ *and* $\mathrm{LM}(\mathfrak{a}_{n,f}) = \mathrm{lm}(g_n)x$*. Since* $\mathrm{lm}(g_{n+1}) \prec \mathrm{lm}(g_n)x$*, each ambiguity* $\mathfrak{a}_{n,f}$ *is covered by* $g_{n+1}^{[\beta_{n+1}]}$*.*

Additionally, for each pair $m, n \in \mathbb{N}$ *with* $m > n$*, we have a regular ambiguity between* $g_m^{[\beta_m]}$ *and* $g_n^{[\beta_n]}$ *given by*

$$\mathfrak{a}_{m,n} = \left(1 \otimes 1,\, x^{m-n} \otimes 1,\, g_m^{[\beta_m]},\, g_n^{[\beta_n]}\right).$$

As $\mathrm{SIG}(\mathfrak{a}_{m,n}) = x^{m-n}\mathrm{sig}(\beta_n)$*, these ambiguities are all covered by the trivial syzygies*

$$x^{m-n}\beta_n - \beta_m \in \mathrm{Triv}(G^{[\Sigma]}) \subseteq H.$$

Thus all regular ambiguities of $G^{[\Sigma]}$ *are covered by* $(G^{[\Sigma]}, H)$*, showing that* $G^{[\Sigma]}$ *is a minimal labelled Gröbner basis of* $I^{[\Sigma]}$ *and that* H *is a Gröbner basis of the syzygy module.*

We also claimed that the finite set $G^{[\Sigma']} = \{g_0^{[\delta_1]}, f^{[\delta_2]}\}$ *forms a minimal labelled Gröbner basis of the labelled module* $I^{[\Sigma']}$ *generated by* $g_0^{[\delta_1]}, f^{[\delta_2]}$*. The minimality of* $G^{[\Sigma']}$ *is clear. To show that* $G^{[\Sigma']}$ *is also a labelled Gröbner basis, we note that* $G^{[\Sigma']}$ *only has two regular ambiguities, given by*

$$\mathfrak{a}_{f,0} = \left(1 \otimes y,\, y \otimes 1,\, f^{[\delta_2]},\, g_0^{[\delta_1]}\right), \qquad \mathfrak{a}_{0,f} = \left(1 \otimes x,\, x \otimes 1,\, g_0^{[\delta_1]},\, f^{[\delta_2]}\right),$$

which are covered by the syzygies

$$\gamma_{f,0} = \delta_2 y - y\delta_1 + \delta_1 y, \qquad \gamma_{0,f} = x\delta_2 - \delta_1 x + x\delta_1.$$

Similarly to Bergman's diamond lemma characterising classical Gröbner bases, Theorem 3.3.11 allows us to state a first non-optimised version of a signature-based algorithm for noncommutative polynomials by ensuring that all regular ambiguities are covered. Additionally, the theorem also allows us to describe more precisely Gröbner bases of the syzygy module $\mathrm{Syz}(I^{[\Sigma]})$. Consider the set $\mathrm{H}_{\mathrm{triv}}$ of trivial syzygies of $G^{[\Sigma]} \subseteq I^{[\Sigma]}$, that is,

$$\mathrm{H}_{\mathrm{triv}} = \left\{ \alpha m g - f m \beta \mid f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]}, m \in \langle X \rangle \right\}. \tag{3.3}$$

Note that, for all $f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]}$ and $m \in \langle X \rangle$ for which $\mathrm{sig}(\alpha m \mathrm{lt}(g)) \not\simeq \mathrm{sig}(\mathrm{lt}(f) m \beta)$, the set of signatures of $\mathrm{H}_{\mathrm{triv}}$ contains

$$\max\left\{\mathrm{sig}(\alpha m \mathrm{lt}(g)), -\mathrm{sig}(\mathrm{lt}(f) m \beta)\right\}.$$

Because of this, it may happen that this set contains infinitely many signatures that do not divide each other, and indeed this will be the case for all sufficiently nontrivial ideals. This observation implies that, for all such ideals, $\mathrm{Syz}(I^{[\Sigma]})$ does not admit a finite Gröbner basis.

However, Theorem 3.3.11 shows that a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ is given by adding to $\mathrm{H}_{\mathrm{triv}}$ all the syzygies found by regular sig-reducing to zero all S-polynomials coming from regular ambiguities of $G^{[\Sigma]}$. Furthermore, if $G^{[\Sigma]}$ is finite, the set of trivial syzygies in $\mathrm{H}_{\mathrm{triv}}$ can be enumerated using the description (3.3), and altogether we obtain an effective description of the syzygy module of $f_1, \ldots, f_r$.

Algorithm

The algorithm, incorporating both the computation of the labelled Gröbner basis and of the aforementioned description of the syzygy module of $f_1, \ldots, f_r$, is given in Algorithm 2. We note that we state this algorithm only for theoretical consideration. In an actual implementation, one would replace all computations with labelled polynomials in Algorithm 2

by computations with signature polynomials. In Section 3.3.3, we state with Algorithm 3 an optimised version of Algorithm 2 incorporating this.

Algorithm 2 follows the same basic structure as Algorithm 1. It runs on a loop, processing ambiguities from a queue. At each step, it selects an ambiguity using a *fair* selection strategy. Like in the classical case without signatures (see Definition 2.4.57 and line 4 in Algorithm 1), such a selection strategy has to ensure that every ambiguity that is formed is eventually processed.

Algorithm 2: Labelled Gröbner basis algorithm

Input: $(f_1, \dots, f_r) \in K\langle X\rangle^r$

Output (if the algorithm terminates):

- $G^{[\Sigma]}$ a labelled Gröbner basis of the labelled module $I^{[\Sigma]}$ generated by $f_1, \dots, f_r$;
- $H \subseteq \mathrm{Syz}(I^{[\Sigma]})$ such that $H \cup \mathrm{Triv}(G^{[\Sigma]})$ is a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$;

```
1  G^[Σ] ← {f_1^[ε_1], ..., f_r^[ε_r]};
2  H ← ∅;
3  amb ← amb(G^[Σ]);
4  while amb ≠ ∅ :
5      select an ambiguity 𝔞 ∈ amb using a fair strategy and remove it;
6      if 𝔞 is regular and 𝔞 is not covered by (G^[Σ], H) :
7          f'^[α'] ← result of regular sig-reducing S-Pol(𝔞) by G^[Σ];
8          if f' = 0 :
9              H ← H ∪ {α'};
10         else:
11             G^[Σ] ← G^[Σ] ∪ {f'^[α']};
12             amb ← amb ∪ ⋃_{g^[β] ∈ G^[Σ]} amb(f'^[α'], g^[β]);
13 return G^[Σ], H;
```

Example 3.3.14. *Important examples of fair selection strategies are the following. One can compare them to the fair strategies for Buchberger's algorithm in the free algebra mentioned in Example 2.4.58.*

- *A first-in first-out selection strategy, always choosing the ambiguity that has been in the set* `amb` *the longest, is fair.*

- *More generally, a selection strategy that chooses the elements in* amb *in generations, always selecting all ambiguities from one generation before proceeding to the next one, is fair.*
- *A module order* $\preceq$ *is called* fair *if, for any* $\mu \in M(\Sigma)$*, the set* $\{\mu' \in M(\Sigma) \mid \mu' \prec \mu\}$ *is finite. A selection strategy that always chooses* $\mathfrak{a} \in \mathtt{amb}$ *with* $\mathrm{SIG}(\mathfrak{a})$ *minimal is fair if the used module order is fair.*

Theorem 3.3.11 implies that any ambiguity that is not regular or that is already covered by $(G^{[\Sigma]}, H)$ can be discarded without having to perform any sig-reductions. This fact is incorporated into Algorithm 2 in line 6.

Remark 3.3.15. *Compared to [HV22, Algo. 1] for computing labelled Gröbner bases in the free algebra, Algorithm 2 provides two main advantages: the stronger cover criterion and additional flexibility by allowing any fair selection strategy. Especially the latter is an important improvement. The algorithm in [HV22] could only process ambiguities by increasing signatures, which restricted the allowed module orders to only fair orders, excluding, for example, any kind of elimination order.*

If an ambiguity is not redundant, its S-polynomial is regular sig-reduced and its normal form is either added to the partial Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ (if the normal form is zero) or to the partial labelled Gröbner basis $G^{[\Sigma]}$ (if the normal form is nonzero). In the latter case, also the set of ambiguities to process is updated. The algorithm ensures that for every regular ambiguity of $G^{[\Sigma]}$, an element is eventually added to $G^{[\Sigma]}$ or to H which covers it.

Like in the case of noncommutative Gröbner bases without signatures, we cannot expect Algorithm 2 to always terminate. However, the following theorem ensures that the algorithm correctly enumerates a labelled Gröbner basis of the labelled module $I^{[\Sigma]}$ generated by the input $(f_1, \dots, f_r) \in K\langle X\rangle^r$ and a Gröbner basis of the syzygy module $\mathrm{Syz}(I^{[\Sigma]})$. It can be compared to Theorem 2.4.59.

Theorem 3.3.16. *Let* $(f_1, \dots, f_r) \in K\langle X\rangle^r$ *and, for* $n \in \mathbb{N}$*, let* $G_n^{[\Sigma]}$*,* H_n*, and* $\mathtt{amb}_n$ *be the value of* $G^{[\Sigma]}$*,* H*, and* amb *respectively, in Algorithm 2 after* n *iterations of the "*while*" loop given the family* $(f_1, \dots, f_r)$ *as input. Furthermore, let* $\sigma_n = \min\{\mathrm{SIG}(\mathfrak{a}) \mid \mathfrak{a} \in \mathtt{amb}_n\}$*, choosing an arbitrary element in case of similarity.*

Then, for $n \in \mathbb{N}$, the following hold:

1. *$G_n^{[\Sigma]}$ is a labelled Gröbner basis of the labelled module $I^{[\Sigma]}$ generated by $f_1, \dots, f_r$ up to signature σ_n;*
2. *$H_n \cup \{\gamma \in \mathrm{Triv}(G_n^{[\Sigma]}) \mid \mathrm{sig}(\gamma) \prec \sigma_n\}$ is a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ up to signature σ_n;*

Furthermore,

1. *$G^{[\Sigma]} = \bigcup_{n\in\mathbb{N}} G_n^{[\Sigma]}$ is a labelled Gröbner basis of $I^{[\Sigma]}$;*
2. *$H \cup \mathrm{Triv}(G^{[\Sigma]}) = \bigcup_{n\in\mathbb{N}} \left(H_n \cup \mathrm{Triv}(G_n^{[\Sigma]})\right)$ is a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$;*

In this sense, Algorithm 2 enumerates a labelled Gröbner basis of $I^{[\Sigma]}$ and a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$.

Proof. For the first part, fix $n \in \mathbb{N}$. We show that the algorithm enforces the requirements of Theorem 3.3.11. By construction, all elements in $G_n^{[\Sigma]}$ have nonzero polynomial part, and, for all signatures ε_i, the element $f_i^{[\varepsilon_i]}$ lies in $G_0^{[\Sigma]} \subseteq G_n^{[\Sigma]}$. Finally, the algorithm ensures that all regular ambiguities it considers are covered: either an ambiguity $\mathfrak{a}$ is already covered, or an element is added to either $G^{[\Sigma]}$ or H with signature similar to $\mathrm{SIG}(\mathfrak{a})$ and leading monomial strictly smaller than $\mathrm{LM}(\mathfrak{a})$. Either way, this element covers the ambiguity $\mathfrak{a}$. To finish the proof of the first part, we note that if σ_n is a minimal signature left in $\mathtt{amb}_n$, then all ambiguities of $G_n^{[\Sigma]}$ with signature $\prec \sigma_n$ have already been processed. With this, Theorem 3.3.11 implies the result.

The second part follows analogously to the first part from Corollary 3.3.13 and the fact that, due to the fair selection strategy, every ambiguity of $G^{[\Sigma]}$ is processed eventually. □

Remark 3.3.17. *Theorem 3.3.16 implies that, if a fair module order is used in Algorithm 2 and the selection of the ambiguities is done by increasing signature, then the algorithm allows to compute, for any signature $\sigma \in M(\Sigma)$, a labelled Gröbner basis up to signature σ and a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ up to signature σ* in finite time.

Just like Algorithm 1, we can show that Algorithm 2 terminates if and only if the labelled module admits a finite labelled Gröbner basis.

Proposition 3.3.18. *Let $(f_1, \dots, f_r) \in K\langle X\rangle^r$. Algorithm 2 terminates given the family $(f_1, \dots, f_r)$ as input if and only if the labelled module generated by $f_1, \dots, f_r$ admits a finite labelled Gröbner basis (with respect to the chosen orders).*

Proof. Analogous to Proposition 2.4.60. □

3.3.2 Elimination criteria

In the commutative case, it is well known that signature-based algorithms can be equipped with strong elimination criteria, allowing to detect and avoid sig-reductions to zero. So far, we have already seen that we can immediately discard all singular ambiguities and remove a regular ambiguity if it is covered. In this section, we adapt some other classical techniques from the commutative case to our setting, namely the *syzygy criterion*, the *singular criterion* and the *F5 criterion*. In Algorithm 3, we include these criteria to show how to use them in practice.

Proposition 3.3.19 (Syzygy criterion)**.** *Let $f^{[\alpha]} \in I^{[\Sigma]}$ and let $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ be a labelled Gröbner basis up to signature* $\mathrm{sig}(\alpha)$*. If there exists a syzygy $\gamma \in \mathrm{Syz}(I^{[\Sigma]})$ such that* $\mathrm{sig}(\gamma)$ *divides* $\mathrm{sig}(\alpha)$*, then $f^{[\alpha]}$ can be regular* sig*-reduced to zero by $G^{[\Sigma]}$.*

Proof. Let $c \in K$ and $a, b \in \langle X\rangle$ such that $\mathrm{sig}(\alpha) = \mathrm{sig}(ca\gamma b)$. Consider $f^{[\beta]} = f^{[\alpha]} - ca0^{[\gamma]}b$ and note that $\mathrm{sig}(\beta) \prec \mathrm{sig}(\alpha)$. Since $G^{[\Sigma]}$ is a labelled Gröbner basis up to signature $\mathrm{sig}(\alpha)$, the labelled polynomial $f^{[\beta]}$ sig-reduces to zero by $G^{[\Sigma]}$. Using the same reductions, we see that $f^{[\alpha]}$ regular sig-reduces to zero by $G^{[\Sigma]}$. □

Hence, we can discard any regular ambiguity during the computation of a labelled Gröbner basis whose signature is divisible by the signature of a syzygy. In particular, all elements stored in the set H are known syzygies and can be used directly to remove redundant ambiguities. We note that excluding covered ambiguities already encapsulates this part of the syzygy criterion. Another way to leverage the syzygy criterion is to exploit the fact that signatures of trivial syzygies can be predicted in advance without having to perform any reductions. The aim of the *F5 criterion* is to detect these trivial syzygies. It follows immediately from the cover criterion (Theorem 3.3.11) but is not directly included in Algorithm 2.

Corollary 3.3.20 (F5 criterion)**.** *In Algorithm 2, let* $\mathfrak{a} \in \mathtt{amb}$ *be regular such that there exist* $f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]}$ *and* $m \in M(\Sigma)$ *with*

- $\operatorname{sig}(\alpha m g) \not\simeq \operatorname{sig}(f m \beta)$*, and*
- $\operatorname{SIG}(\mathfrak{a})$ *is divisible by* $\max\{\operatorname{sig}(\alpha m g), \operatorname{sig}(f m \beta)\}$.

Then $\mathfrak{a}$ *is covered by the trivial syzygy* $\alpha m g - f m \beta \in \operatorname{Triv}(G^{[\Sigma]})$ *and can be discarded.*

The F5 criterion as phrased above also includes Buchberger's coprime criterion for eliminating S-polynomials coming from elements with coprime leading monomials, see [LMA23, Lem. 22] for a noncommutative version without signatures over coefficient rings.

Unlike in the commutative case, it is not possible to simply add all trivial syzygies to H whenever a new element is added to the labelled Gröbner basis, as there are infinitely many. Indeed, each pair of polynomials gives rise to infinitely many trivial syzygies, one for each choice of $m \in \langle X \rangle$. However, to check the F5 criterion, there are only finitely many syzygies which can possibly apply, and we can construct them on demand. This renders the complexity of applying Corollary 3.3.20 essentially quadratic in the size of $G^{[\Sigma]}$. Analogous to the commutative case, this cost can be reduced to linear for homogeneous polynomials under certain module orders. We refer to Section 3.4.3 for further details.

To end this section, we mention with the singular criterion another important elimination criterion. If the selection strategy in Algorithm 2 chooses ambiguities with the same signature by increasing leading monomial, it is also fully encapsulated by the cover criterion.

Corollary 3.3.21 (Singular criterion)**.** *If, at any point in Algorithm 2, there are regular* $\mathfrak{a}, \mathfrak{b} \in \mathtt{amb}$ *with* $\operatorname{SIG}(\mathfrak{a}) \simeq \operatorname{SIG}(\mathfrak{b})$ *and* $\operatorname{LM}(\mathfrak{a}) \preceq \operatorname{LM}(\mathfrak{b})$*, then* $\mathfrak{b}$ *can be removed after processing* $\mathfrak{a}$*.*

Proof. The assumptions imply that $\mathfrak{b}$ is covered by S-Pol($\mathfrak{a}$). So, after processing $\mathfrak{a}$ (and thus reducing S-Pol($\mathfrak{a}$)) both ambiguities are covered. □

3.3.3 Computation of signature Gröbner bases and reconstruction

So far, Algorithm 2 keeps but does not exploit all the information encoded in the full module representation contained in labelled polynomials. As indicated earlier, keeping track of the full module representation, however, causes a significant overhead in terms of memory consumption and overall computation time. Consequently, in an actual implementation of Algorithm 2, one would only keep track of the signature of each polynomial, and thereby, work with signature polynomials. In doing so, instead of computing a labelled Gröbner basis and a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$, the algorithm only computes a signature Gröbner basis (see Definition 3.2.9) of $I^{[\Sigma]}$ and a Gröbner basis of the $K\langle X\rangle$-subbimodule of Σ generated by all signatures of syzygies, that is, the $K\langle X\rangle$-subbimodule generated by $\mathrm{sig}(\mathrm{Syz}(I^{[\Sigma]}))$. Here, by a slight abuse of notation, we let $\mathrm{sig}(H) = \{\mathrm{sig}(\alpha) \mid \alpha \in H\} \subseteq T(\Sigma)$ for sets $H \subseteq \Sigma$. Additionally, to obtain an efficient implementation, one would also exploit the elimination criteria discussed in the previous section. Note that these criteria only depend on information encoded in signature polynomials. Incorporating these changes leads to Algorithm 3, which is an optimised version of Algorithm 2.

Theorem 3.3.22. *Algorithm 3 is correct.*

Proof. Follows from the correctness of Algorithm 2 and from Corollary 3.3.20 and 3.3.21. □

In the following, we discuss how to recover the information that is lost when Algorithm 3 is used instead of Algorithm 2. In particular, this means reconstructing a labelled Gröbner basis from a signature Gröbner basis and reconstructing a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ from one of the bimodule generated by $\mathrm{sig}(\mathrm{Syz}(I^{[\Sigma]}))$. To this end, we adapt the reconstruction methods described in [GVW16] to recover module representations of elements of the ideal and of syzygies from signatures to our noncommutative setting.

We let $G^{(\Sigma)} \subseteq I^{(\Sigma)}$ and $H \subseteq \mathrm{sig}(\mathrm{Syz}(I^{[\Sigma]}))$ be the output of Algorithm 3 when given the family of generators $(f_1, \ldots, f_r) \in K\langle X\rangle^r$ as input. Recall that the algorithm does not necessarily terminate. As such, $G^{(\Sigma)}$ will either be the full output of Algorithm 3 assuming termination, or the partial output after interrupting the computation. In the latter case, the set $G^{(\Sigma)}$ is only a signature Gröbner basis up to a certain signature $\sigma \in T(\Sigma)$ and H together with the signatures of the trivial syzygies only forms a partial Gröbner basis of the bimodule generated by $\mathrm{sig}(\mathrm{Syz}(I^{[\Sigma]}))$.

Algorithm 3: Signature Gröbner basis algorithm

Input: $(f_1, \dots, f_r) \in K\langle X\rangle^r$

Output (if the algorithm terminates):

- $G^{(\Sigma)}$ a signature Gröbner basis of the labelled module $I^{[\Sigma]}$ generated by $f_1, \dots, f_r$;
- $H \subseteq \operatorname{sig}(\operatorname{Syz}(I^{[\Sigma]}))$ such that $H \cup \operatorname{sig}(\operatorname{Triv}(G^{(\Sigma)}))$ is a Gröbner basis of the $K\langle X\rangle$-subbimodule generated by $\operatorname{sig}(\operatorname{Syz}(I^{[\Sigma]}))$;

```
1  G^(Σ) ← {f_1^(ε_1), ..., f_r^(ε_r)};
2  H ← ∅;
3  amb ← amb(G^(Σ));
4  while amb ≠ ∅ :
5      select an ambiguity 𝔞 ∈ amb using a fair strategy and remove it;
6      if 𝔞 is regular and 𝔞 is not covered by (G^(Σ), H) :
7          remove all regular 𝔟 ∈ amb with SIG(𝔞) ≃ SIG(𝔟) and LM(𝔞) ⪯ LM(𝔟) (singular
            criterion);
8          if 𝔞 does not satisfy the hypotheses of the F5 criterion (Corollary 3.3.20) :
9              f'^(σ) ← result of regular sig-reducing S-Pol(𝔞) by G^(Σ);
10             if f' = 0 :
11                 H ← H ∪ {σ};
12             else:
13                 G^(Σ) ← G^(Σ) ∪ {f'^(σ)};
14                 amb ← amb ∪ ⋃_{g^(μ) ∈ G^(Σ)} amb(f'^(σ), g^(μ));
15 return G^(Σ), H;
```

In this general setting, the goal of this section is twofold. First of all, starting from $G^{(\Sigma)}$ we want to reconstruct a labelled Gröbner basis $G^{[\Sigma]}$ (up to signature σ). Secondly, for each element $\mu \in H$, we want to find a syzygy $\gamma \in \operatorname{Syz}(I^{[\Sigma]})$ such that $\operatorname{sig}(\gamma) \simeq \mu$.

In situations where Algorithm 3 terminates, that is, when $G^{(\Sigma)}$ is a signature Gröbner basis and H together with the signatures of the trivial syzygies forms a Gröbner basis of the bimodule generated by $\operatorname{sig}(\operatorname{Syz}(I^{[\Sigma]}))$, achieving both of these goals allows us to also recover a Gröbner basis of $\operatorname{Syz}(I^{[\Sigma]})$. The algorithms which we describe in this section are a direct adaptation of the ones outlined in [GVW16].

Our first goal can be achieved by the following Algorithm 4. We note that no matter whether Algorithm 3 terminates by itself or whether we interrupt the computation, the sets $G^{(\Sigma)}$ and H are always finite. Also note that Algorithm 4 requires a *minimal* signature

Gröbner basis as input, but the output of Algorithm 3 is not necessarily minimal. Thus, before calling Algorithm 4, redundant elements have to be removed from the signature basis, using, for example, Proposition 3.2.14.

Algorithm 4: Labelled Gröbner basis reconstruction

Input: $G^{(\Sigma)}$ a finite minimal signature Gröbner basis of $I^{[\Sigma]}$ (up to some signature $\sigma \in T(\Sigma)$)
Output: $G^{[\Sigma]}$ a finite minimal labelled Gröbner basis of $I^{[\Sigma]}$ (up to signature σ)

1 $G^{[\Sigma]} \leftarrow \emptyset$;
2 $H^{(\Sigma)} \leftarrow G^{(\Sigma)}$; `/* make a copy so that we do not alter` $G^{(\Sigma)}$ `*/`
3 **while** $H^{(\Sigma)} \neq \emptyset$ **:**
4 choose $f^{(\mu)} \in H^{(\Sigma)}$ such that $\mu \simeq \min\{\mu' \mid f'^{(\mu')} \in H^{(\Sigma)}\}$;
5 $H^{(\Sigma)} \leftarrow H^{(\Sigma)} \setminus \{f^{(\mu)}\}$;
6 choose $a, b \in \langle X \rangle, g^{[\beta]} \in G^{[\Sigma]} \cup \{f_1^{[\varepsilon_1]}, \ldots, f_r^{[\varepsilon_r]}\}$ such that $\mathrm{sig}(a\beta b) \simeq \mu$ and $\mathrm{lm}(agb)$ is minimal;
7 $g'^{[\beta']} \leftarrow$ result of regular top sig-reducing $ag^{[\beta]}b$ by $G^{[\Sigma]}$;
8 $G^{[\Sigma]} \leftarrow G^{[\Sigma]} \cup \{g'^{[\beta']}\}$;
9 **return** $G^{[\Sigma]}$;

Remark 3.3.23. *As will be clear from the proof of Proposition 3.3.24, the minimality condition in line 6 of Algorithm 4 is not required for the correctness of the algorithm. It is included purely for efficiency reasons with the hope of having to do less* sig*-reductions if* $\mathrm{lm}(agb)$ *is minimal. We note that the same also holds for the minimality condition in line 3 of Algorithm 5.*

Proposition 3.3.24. *Algorithm 4 is correct.*

Proof. Let $\tilde{G}^{[\Sigma]} \subseteq I^{[\Sigma]}$ be a minimal labelled Gröbner basis of $I^{[\Sigma]}$ (up to signature σ) such that $G^{(\Sigma)} = \{g^{(\mathrm{sig}(\beta))} \mid g^{[\beta]} \in \tilde{G}^{[\Sigma]}\}$. Furthermore, let $G^{[\Sigma]}$ be the output of Algorithm 4 given $G^{(\Sigma)}$ as input. To prove the correctness of the algorithm, we show that

$$\left\{(\mathrm{lm}(g), \mathrm{sm}(\beta)) \mid g^{[\beta]} \in G^{[\Sigma]}\right\} = \left\{(\mathrm{lm}(g), \mathrm{sm}(\beta)) \mid g^{[\beta]} \in \tilde{G}^{[\Sigma]}\right\}. \tag{3.4}$$

In other words, we show that the labelled polynomials in $G^{[\Sigma]}$ have the same leading monomials and similar signatures as the elements in the labelled Gröbner basis $\tilde{G}^{[\Sigma]}$. Then, every element in $I^{[\Sigma]}$ is sig-reducible by $G^{[\Sigma]}$ if and only if it is sig-reducible by $\tilde{G}^{[\Sigma]}$, and

Lemma 3.2.7 yields that $G^{[\Sigma]}$ is a labelled Gröbner basis (up to signature σ). Furthermore, the minimality of $\tilde{G}^{[\Sigma]}$ implies the minimality of $G^{[\Sigma]}$.

To prove (3.4), we show that the following loop invariant holds whenever the algorithm reaches line 3:

$$\left\{(\mathrm{lm}(g), \mathrm{sm}(\mu)) \mid g^{(\mu)} \in G^{(\Sigma)} \setminus H^{(\Sigma)}\right\} = \left\{(\mathrm{lm}(g), \mathrm{sm}(\beta)) \mid g^{[\beta]} \in G^{[\Sigma]}\right\}. \tag{3.5}$$

Once the algorithm terminates and $H^{(\Sigma)} = \emptyset$, this implies (3.4) since the leading monomials and signatures of the elements in $G^{(\Sigma)}$ are equal to those of $\tilde{G}^{[\Sigma]}$ by definition of $\tilde{G}^{[\Sigma]}$.

Obviously (3.5) holds in the very beginning when $H^{(\Sigma)} = G^{(\Sigma)}$. So, now assume that (3.5) holds at some point when the algorithm reaches line 3 and let $f^{(\mu)} \in H^{(\Sigma)}$ be the signature polynomial that is chosen in line 4. Furthermore, let $\alpha \in \Sigma$ be such that $f^{[\alpha]} \in \tilde{G}^{[\Sigma]}$ with $\mathrm{sig}(\alpha) = \mu$. Then, let $a, b \in \langle X \rangle$ and $g^{[\beta]} \in G^{[\Sigma]} \cup \{f_1^{[\varepsilon_1]}, \ldots, f_r^{[\varepsilon_r]}\}$ be as chosen in line 6. Due to the presence of the generators $f_1^{[\varepsilon_1]}, \ldots, f_r^{[\varepsilon_r]}$, such a choice of a, b and $g^{[\beta]}$ is always possible. Let $g'^{[\beta']}$ be the result of the computation in line 7. By construction, $g'^{[\beta']}$ is regular top sig-reduced by $G^{[\Sigma]}$. Furthermore, note that $f^{[\alpha]}$ is regular top sig-reduced by $\tilde{G}^{[\Sigma]}$ because $\tilde{G}^{[\Sigma]}$ is minimal. Consequently, the loop invariant implies that $f^{[\alpha]}$ is also regular top sig-reduced by $G^{[\Sigma]}$. Note that, since we only care about *regular* top sig-reducibility, it is irrelevant whether we consider $G^{[\Sigma]}$ before or after adding $g'^{[\beta']}$ as $\mathrm{sig}(\alpha) \simeq \mathrm{sig}(\beta')$. Also, note that the loop invariant, together with the fact that μ was chosen to be minimal among all signatures in $H^{(\Sigma)}$, implies that $G^{[\Sigma]}$ is a labelled Gröbner basis up to signature $\mu \simeq \mathrm{sig}(\alpha)$. Hence, Lemma 3.2.10 is applicable to $f^{[\alpha]}$ and $g'^{[\beta']}$, yielding $\mathrm{lm}(f) = \mathrm{lm}(g')$. Since also $\mu \simeq \mathrm{sig}(a\beta b) = \mathrm{sig}(\beta')$, the loop invariant still holds after removing $f^{(\mu)}$ from $H^{(\Sigma)}$ and adding $g'^{[\beta']}$ to $G^{[\Sigma]}$. □

After recovering a labelled Gröbner basis, we can proceed with the following Algorithm 5 to also recover the syzygies whose signatures are saved in H.

Proposition 3.3.25. *Algorithm 5 is correct.*

Proof. To see the correctness of the algorithm, the only problematic lines are line 3 and 4. Note that, due to the presence of the generators $f_1^{[\varepsilon_1]}, \ldots, f_r^{[\varepsilon_r]}$ in line 3, a choice of a, b and $g^{[\beta]}$ as required is always possible. It remains to show that $ag^{[\beta]}b$ really regular sig-reduces

to zero by $G^{[\Sigma]}$, but this follows from Proposition 3.3.19 and the fact that the elements in H are signatures of syzygies. □

Algorithm 5: Syzygy reconstruction

Input:

- $G^{[\Sigma]}$ a labelled Gröbner basis of $I^{[\Sigma]}$ (up to some signature $\sigma \in T(\Sigma)$);
- $H \subseteq \mathrm{sig}(\mathrm{Syz}(I^{[\Sigma]}))$ (such that $\max H \prec \sigma$);

Output: $\tilde{H} \subseteq \mathrm{Syz}(I^{[\Sigma]})$ such that $\mathrm{sig}(\tilde{H}) = H$

1 $\tilde{H} \leftarrow \emptyset$;
2 **for** $\mu \in H$ **:**
3 choose $a, b \in \langle X \rangle, g^{[\beta]} \in G^{[\Sigma]} \cup \{f_1^{[\varepsilon_1]}, \ldots, f_r^{[\varepsilon_r]}\}$ such that $\mathrm{sig}(a\beta b) \simeq \mu$ and $\mathrm{lm}(agb)$ is minimal;
4 $0^{[\beta']} \leftarrow$ result of regular sig-reducing $ag^{[\beta]}b$ by $G^{[\Sigma]}$;
5 $\tilde{H} \leftarrow \tilde{H} \cup \{c\beta'\}$ with $c = \mathrm{sc}(\mu)/\mathrm{sc}(\beta')$;
6 **return** $\tilde{H}$

To conclude this section, we note that if Algorithm 3 terminates without interruption, a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ can be obtained as follows: First, apply Algorithm 4 to obtain a labelled Gröbner basis of $I^{[\Sigma]}$. Next, use Algorithm 5 to get the set $\tilde{H}$ containing the recovered syzygies. Finally, a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ is given by $\tilde{H} \cup \mathrm{Triv}(G^{[\Sigma]})$.

3.4 Computations in the mixed algebra

In this section, we only present the theory in terms of labelled Gröbner bases and leave all adaptations for signature polynomials to the reader. The required changes are completely analogous to the free algebra, and also the reconstruction techniques from Section 3.3.3 can be adapted to the more general setting of the mixed algebra.

We fix a family of polynomials $(f_1, \ldots, f_r) \in \mathcal{A}^r$ generating a labelled module $I^{[\Sigma]}$, as well as a monomial order $\preceq$ on $[X]\langle Y \rangle$ and a module order $\preceq_\Sigma$ on $M(\Sigma)$. As in the previous section, we assume that the two orders are *compatible*, which, in this setting now, means that, for all $a, b \in [X]\langle Y \rangle$ and $i \in \{1, \ldots, r\}$, we have

$$a \prec b \iff a\varepsilon_i \prec_\Sigma b\varepsilon_i \iff \varepsilon_i a \prec_\Sigma \varepsilon_i b.$$

As before, we will denote both orders by the same symbol $\preceq$ from now on.

3.4.1 Computation of labelled Gröbner bases

Regular ambiguities

In order to define S- (and G-)polynomials in our setting, we first adapt the notion of ambiguities to the mixed algebra. Since we now work over a coefficient ring and deal with mixed monomials, the adaptation is not as straightforward as in the free case. In particular, we now have to define a new type of ambiguity, called *external ambiguity*. These ambiguities encode situations where a term can be reduced by $f^{[\alpha]}$ and $g^{[\beta]}$ in two (possibly different) *non-overlapping* ways. More precisely, they ensure that all terms of the form $\mathrm{lm}(f)m\mathrm{lm}(g)$ with $m \in \langle Y \rangle$ have canonical normal forms. In the free algebra over a coefficient field, such ambiguities are redundant, which is reflected by Case 3 in the proof of Theorem 2.4.54 and by Case 1 in the proof of Lemma 3.3.7 respectively. Intuitively, a noncommutative analogue of Buchberger's coprime criterion [Mor94, Cor. 5.8] ensures that the respective reductions yield a canonical normal form, making external ambiguities useless in that setting. Now, when dealing with coefficient rings and mixed monomials, Buchberger's coprime criterion (Corollary 3.4.29) is weaker, and we cannot discard all external ambiguities *a priori*. We note that external ambiguities also have to be considered when computing Gröbner bases in the mixed algebra without signatures [MZ98], or when computing Gröbner bases in the free algebra over coefficient rings [LMA23].

We first define ambiguities for mixed monomials and then extend them to labelled mixed polynomials.

Definition 3.4.1. *Let $\boldsymbol{x}^{\boldsymbol{a}}p, \boldsymbol{x}^{\boldsymbol{b}}q \in [X]\langle Y \rangle$ and $(a \otimes b, c \otimes d, p, q)$ be an (overlap/inclusion) ambiguity of the words p and q. An* (overlap/inclusion) ambiguity *of $\boldsymbol{x}^{\boldsymbol{a}}p$ and $\boldsymbol{x}^{\boldsymbol{b}}q$ is given by*

$$(ua \otimes b, vc \otimes d, \boldsymbol{x}^{\boldsymbol{a}}p, \boldsymbol{x}^{\boldsymbol{b}}q),$$

where $u = \mathrm{lcm}(\boldsymbol{x}^{\boldsymbol{a}}, \boldsymbol{x}^{\boldsymbol{b}})/\boldsymbol{x}^{\boldsymbol{a}}$ and $v = \mathrm{lcm}(\boldsymbol{x}^{\boldsymbol{a}}, \boldsymbol{x}^{\boldsymbol{b}})/\boldsymbol{x}^{\boldsymbol{b}}$.

Furthermore, for every $m \in \langle Y \rangle$, we call the tuples

$$(u \otimes mq, vpm \otimes 1, \boldsymbol{x}^{\boldsymbol{a}}p, \boldsymbol{x}^{\boldsymbol{b}}q) \quad \text{and} \quad (uqm \otimes 1, v \otimes mp, \boldsymbol{x}^{\boldsymbol{a}}p, \boldsymbol{x}^{\boldsymbol{b}}q)$$

external ambiguities *of $\boldsymbol{x}^{\boldsymbol{a}}p$ and $\boldsymbol{x}^{\boldsymbol{b}}q$.*

For $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ *with* $f, g \neq 0$*, an* (overlap/inclusion/external) ambiguity *of* $f^{[\alpha]}$ *and* $g^{[\beta]}$ *is*

$$(ua \otimes b, vc \otimes d, f^{[\alpha]}, g^{[\beta]})$$

where $(ua \otimes b, vc \otimes d, \mathrm{lm}(f), \mathrm{lm}(g))$ *is an (overlap/inclusion/external) ambiguity of the mixed monomials* $\mathrm{lm}(f)$ *and* $\mathrm{lm}(g)$*. We denote by* $\mathrm{amb}(f^{[\alpha]}, g^{[\beta]})$ *the set of all ambiguities of* $f^{[\alpha]}$ *and* $g^{[\beta]}$*. Also, for* $G^{[\Sigma]} \subseteq I^{[\Sigma]}$*, let*

$$\mathrm{amb}(G^{[\Sigma]}) \coloneqq \bigcup_{\substack{f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]} \\ f,g \neq 0}} \mathrm{amb}(f^{[\alpha]}, g^{[\beta]}).$$

Remark 3.4.2. *Note that two elements* $f^{[\alpha]}$ *and* $g^{[\beta]}$ *can only have finitely many overlap and inclusion ambiguities, but they always have infinitely many external ambiguities.*

As in the previous cases, when clear by context, we drop $f^{[\alpha]}$ and $g^{[\beta]}$ from an ambiguity. Ambiguities of labelled mixed polynomials differ in two ways from ambiguities of (labelled) polynomials in the free algebra. First, since we now deal with mixed monomials, they also have to match the commutative part of the leading monomials. Furthermore, the more general setting of the mixed algebra requires the consideration of external ambiguities.

Nevertheless, also in this setting, ambiguities of labelled polynomials still satisfy the following relation regarding their leading monomial.

Lemma 3.4.3. *If* $(a \otimes b, c \otimes d)$ *is an ambiguity of* $f^{[\alpha]}$ *and* $g^{[\beta]}$*, then* $\mathrm{lm}(afb) = \mathrm{lm}(cgd)$.

We define the leading monomial and the signature of an ambiguity analogously to the free case (Definition 3.3.3). In the following, for a pair of nonzero $f, g \in \mathcal{A}$, let $\mathrm{lcmlc}(f, g)$ be the least common multiple of $\mathrm{lc}(f)$ and $\mathrm{lc}(g)$.

Definition 3.4.4. *Let* $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ *be such that* $f, g \neq 0$ *and let* $\mathfrak{a} = (a \otimes b, c \otimes d) \in \mathrm{amb}(f^{[\alpha]}, g^{[\beta]})$*. The* leading monomial *of* $\mathfrak{a}$ *is* $\textsc{lm}(\mathfrak{a}) \coloneqq \mathrm{lm}(afb)$ *and, with* $c_f = \frac{\mathrm{lcmlc}(f,g)}{\mathrm{lc}(f)}$*,* $c_g = \frac{\mathrm{lcmlc}(f,g)}{\mathrm{lc}(g)}$*, the* signature *of* $\mathfrak{a}$ *is*

$$\textsc{sig}(\mathfrak{a}) \coloneqq \max\left\{\mathrm{sig}(c_f a \alpha b), -\mathrm{sig}(c_g c \beta d)\right\},$$

choosing the first if $\operatorname{sig}(a\alpha b) \simeq \operatorname{sig}(c\beta d)$. *Furthermore,* $\mathfrak{a}$ *is called* regular *if* $\operatorname{sig}(a\alpha b) \not\simeq \operatorname{sig}(c\beta d)$ *and* singular *if* $\operatorname{sig}(c_f a\alpha b) = \operatorname{sig}(c_g c\beta d)$.

Compared to Definition 3.3.3, the notion of singularity is now more restrictive. This implies that it can happen that an ambiguity is neither regular nor singular. Also note that, in comparison to Definition 3.3.3, the signature of an ambiguity now also incorporates information about the leading coefficients of the polynomials. In the free case over a coefficient field, this information was redundant, but now, when working over a coefficient ring, it is essential.

The following lemma is an analogue of Lemma 3.3.7.

Lemma 3.4.5. *Let* $g_1^{[\beta_1]}, g_2^{[\beta_2]} \in I^{[\Sigma]}$ *be such that* $g_1, g_2 \neq 0$ *and let* $t_i \in T(\mathcal{A}), b_i \in \langle Y\rangle$, $i = 1, 2$, *such that*

$$\operatorname{lm}(t_1 g_1 b_1) = \operatorname{lm}(t_2 g_2 b_2) \quad \textit{and} \quad \operatorname{sig}(t_1\beta_1 b_1) \succ \operatorname{sig}(t_2\beta_2 b_2).$$

Then there exist a regular ambiguity $\mathfrak{a} \in \operatorname{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$, $t_3 \in T(\mathcal{A})$, *and* $b_3 \in \langle Y\rangle$ *with*

$$t_3 \textsc{lm}(\mathfrak{a}) b_3 = \operatorname{lm}(t_i g_i b_i) \quad \textit{and} \quad t_3 \textsc{sig}(\mathfrak{a}) b_3 \simeq \operatorname{sig}(t_1\beta_1 b_1),$$

with equality of signatures if $\operatorname{lc}(t_1 g_1 b_1) = \operatorname{lc}(t_2 g_2 b_2)$.

Proof. Write $t_i = c_i \boldsymbol{x}^{\boldsymbol{a}_i} v_i \in T(\mathcal{A})$ and $\operatorname{lm}(g_i) = \boldsymbol{x}^{\boldsymbol{b}_i} w_i \in [X]\langle Y\rangle$. Then, by assumption, $\boldsymbol{x}^{\boldsymbol{a}_1+\boldsymbol{b}_1} = \boldsymbol{x}^{\boldsymbol{a}_2+\boldsymbol{b}_2}$ and $v_1 w_1 b_1 = v_2 w_2 b_2 =: W$. If, in W, either of w_1 and w_2 is completely contained in the other, then there exists an inclusion ambiguity of w_1 and w_2 characterising this situation (see Case 2 and 3 in Figure 3.1), otherwise one of them starts earlier in W and the other finishes later, in which case there is an overlap (see Case 4 and 5 in Figure 3.1) or external ambiguity (see Case 1 in Figure 3.1) of w_1 and w_2 characterising this situation. Hence, in any case, there exists an ambiguity $(p_1 \otimes q_1, p_2 \otimes q_2)$ of w_1 and w_2 such that $p_1 w_1 q_1 = p_2 w_2 q_2$ is a subword of W. Thus, there exist $l, r \in \langle Y\rangle$ such that $l p_1 w_1 q_1 r = l p_2 w_2 q_2 r = W$. By definition, $\mathfrak{a} = (u_1 p_1 \otimes q_1, u_2 p_2 \otimes q_2)$, with $u_i = \operatorname{lcm}(\boldsymbol{x}^{\boldsymbol{b}_1}, \boldsymbol{x}^{\boldsymbol{b}_2})/\boldsymbol{x}^{\boldsymbol{b}_i}$, is an ambiguity in $\operatorname{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$. We claim that $\mathfrak{a}$ satisfies the conditions of the lemma with $t_3 = m_3 l \in T(\mathcal{A})$, where $m_3 = \boldsymbol{x}^{\boldsymbol{a}_1+\boldsymbol{b}_1}/\operatorname{lcm}(\boldsymbol{x}^{\boldsymbol{b}_1}, \boldsymbol{x}^{\boldsymbol{b}_2}) = \boldsymbol{x}^{\boldsymbol{a}_2+\boldsymbol{b}_2}/\operatorname{lcm}(\boldsymbol{x}^{\boldsymbol{b}_1}, \boldsymbol{x}^{\boldsymbol{b}_2})$ and $b_3 = r$. The condition on the leading monomials is clear by construction. For the claim concerning

the signatures, we note that $m_3 u_i = \boldsymbol{x}^{\boldsymbol{a}_i}$ and, by the choice of l, r, we have $lp_i = v_i$ and $q_i r = b_i$. Thus, for $i = 1, 2$,

$$\operatorname{sig}(t_3 u_i p_i \beta_i q_i b_3) = \operatorname{sig}(m_3 u_i l p_i \beta_i q_i r) = \operatorname{sig}(\boldsymbol{x}^{\boldsymbol{a}_i} v_i \beta_i b_i) \simeq \operatorname{sig}(t_i \beta_i b_i),$$

showing that $\mathfrak{a}$ is regular since $\operatorname{sig}(t_1 \beta_1 b_1) \succ \operatorname{sig}(t_2 \beta_2 b_2)$ and that

$$t_3 \textsc{sig}(\mathfrak{a}) b_3 = \frac{\operatorname{lcmlc}(g_1, g_2)}{\operatorname{lc}(g_1)} \operatorname{sig}(t_3 u_1 p_1 \beta_1 q_1 b_3) \simeq \operatorname{sig}(t_1 \beta_1 b_1). \tag{3.6}$$

For the final part, assume that also $\operatorname{lc}(t_1 g_1 b_1) = \operatorname{lc}(t_2 g_2 b_2)$, that is, $c_1 \operatorname{lc}(g_1) = c_2 \operatorname{lc}(g_2)$. Then, multiplying t_3 by $c_1 \operatorname{lc}(g_1)/\operatorname{lcmlc}(g_1, g_2)$ turns the similarity in (3.6) into equality. □

Remark 3.4.6. *Lemma 3.4.5 contains, in contrast to Lemma 3.3.7, no special case for trivial syzygies. This is because the trivial syzygy constructed in Lemma 3.3.7 might no longer satisfy all the required conditions in the general setting of the mixed algebra. Instead, such situations are now characterised by the existence of external ambiguities, and are thus included in the stated version. Furthermore, since the signature of an ambiguity now also incorporates information about the leading coefficients of the polynomials, we only obtain equality of signatures in the conclusion of Lemma 3.4.5 in case of equality of leading terms. This is also why the left cofactors t_i now hold a coefficient as well, which is in contrast to Lemma 3.3.7 where they are just (noncommutative) monomials.*

Next, we extend the definition of S-polynomials to labelled mixed polynomials.

Definition 3.4.7. *Let $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ be such that $f, g \neq 0$ and let $\mathfrak{a} = (a \otimes b, c \otimes d) \in \operatorname{amb}(f^{[\alpha]}, g^{[\beta]})$. The* S-polynomial *of $\mathfrak{a}$ is*

$$\text{S-Pol}(\mathfrak{a}) \coloneqq \frac{\operatorname{lcmlc}(f, g)}{\operatorname{lc}(f)} a f^{[\alpha]} b - \frac{\operatorname{lcmlc}(f, g)}{\operatorname{lc}(g)} c g^{[\beta]} d.$$

As usual, S-polynomials are defined so that leading terms cancel, that is, if $h^{[\delta]} = \text{S-Pol}(\mathfrak{a})$, then $\operatorname{lm}(h) \prec \textsc{lm}(\mathfrak{a})$. Furthermore, $\operatorname{sig}(\delta) \preceq \textsc{sig}(\mathfrak{a})$, with equality if and only if the ambiguity is regular and strict inequality if and only if the ambiguity is singular.

In the setting of coefficient rings, only considering S-polynomials is not enough to compute Gröbner bases. This has been observed in the commutative case, see, for example, [BW93,

Sec. 10.1] for a textbook exposition of the setting without signature and [FV21] for the case with signatures, as well as in the noncommutative case, see, for example, [LMA23]. In addition to S-polynomials, we now also need G-polynomials, which do not aim at cancelling leading terms but at obtaining minimal leading coefficients.

Definition 3.4.8. *Let* $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ *be such that* $f, g \neq 0$ *and let* $\mathfrak{a} = (a \otimes b, c \otimes d) \in \operatorname{amb}(f^{[\alpha]}, g^{[\beta]})$. *Furthermore, let* $r, s \in R$ *be Bézout coefficients of* $\operatorname{lc}(f)$ *and* $\operatorname{lc}(g)$, *that is,* $r\operatorname{lc}(f) + s\operatorname{lc}(g) = \gcd(\operatorname{lc}(f), \operatorname{lc}(g))$. *The* G-polynomial *of* $\mathfrak{a}$ *with respect to* r, s *is*

$$\text{G-Pol}_{r,s}(\mathfrak{a}) \coloneqq raf^{[\alpha]}b + scg^{[\beta]}d.$$

The coefficients r, s in the definition of G-polynomials are not unique, and the leading term is only unique up to multiplication by an invertible element. More precisely, the leading monomial of $\text{G-Pol}_{r,s}(\mathfrak{a})$ is $\operatorname{LM}(\mathfrak{a})$ and the leading coefficient is $\gcd(\operatorname{lc}(f), \operatorname{lc}(g))$. The signature of the G-polynomial also depends on the choice of r, s. A crucial observation [FV21, Prop. 2.13] is that these coefficients can be chosen so that G-polynomials are *never* singular, that is, so that the signatures of the two summands do not cancel each other.

Lemma 3.4.9. *Let* $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ *be such that* $f, g \neq 0$ *and let* $\mathfrak{a} \in \operatorname{amb}(f^{[\alpha]}, g^{[\beta]})$. *There exist* $r, s \in R$ *such that* $\operatorname{sig}(\text{G-Pol}_{r,s}(\mathfrak{a})) \simeq \operatorname{SIG}(\mathfrak{a})$.

Proof. The proof of [FV21, Prop. 2.13] only relies on properties of the leading coefficients and carries over directly to our setting. We repeat it here for completeness. If the ambiguity $\mathfrak{a}$ is regular, any pair of Bézout coefficients works. Otherwise, let $a = \operatorname{lc}(f)$, $b = \operatorname{lc}(g)$, $c = \operatorname{sc}(\alpha)$, $d = \operatorname{sc}(\beta)$. The goal is to show that there are Bézout coefficients $r, s \in R$ of a, b so that

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} r \\ s \end{pmatrix} = \begin{pmatrix} \gcd(a,b) \\ h \end{pmatrix},$$

with $h \neq 0$. Assume, for contradiction, that $h = 0$ for all choices of r, s. Now, if $ad - bc = 0$, then

$$ah = a(cr + ds) = c(ar + bs) = c\gcd(a,b) \neq 0,$$

which is a contradiction. Otherwise, if $ad - bc \neq 0$, we consider

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} r - b \\ s + a \end{pmatrix} = \begin{pmatrix} \gcd(a, b) \\ ad - bc \end{pmatrix},$$

yielding again a contradiction. □

We assume that, for every potential pair of leading coefficients, a set of Bézout coefficients r, s, as described in Lemma 3.4.9, has been fixed. We then refer to the corresponding G-polynomial as *the* G-polynomial of $\mathfrak{a}$, denoted by G-Pol($\mathfrak{a}$). Note that if $h^{[\delta]} = \text{G-Pol}(\mathfrak{a})$, then $\text{sig}(\delta) \simeq \text{SIG}(\mathfrak{a})$ and $\text{lm}(h) = \text{LM}(\mathfrak{a})$.

The following lemma captures the significance of G-polynomials for our computations. It states that, if a leading term can be written as a sum of two other leading terms, then it is divisible by the leading term of a G-polynomial.

Lemma 3.4.10. *Let $f^{[\alpha]}, g_1^{[\beta_1]}, g_2^{[\beta_2]} \in I^{[\Sigma]}$ be such that $f, g_1, g_2 \neq 0$ and such that there exist $t_i \in T(\mathcal{A}), b_i \in \langle Y \rangle$, $i = 1, 2$, with*

$$\text{lt}(f) = \text{lt}(t_1 g_1 b_1) + \text{lt}(t_2 g_2 b_2) \quad \textit{and} \quad \text{sig}(t_1 \beta_1 b_1) \succ \text{sig}(t_2 \beta_2 b_2).$$

Then there exists $g_3^{[\beta_3]} = \text{G-Pol}(\mathfrak{a})$ for some $\mathfrak{a} \in \text{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$ and $t_3 \in T(\mathcal{A}), b_3 \in \langle Y \rangle$ such that $\text{lt}(t_3 g_3 b_3) = \text{lt}(f)$ and $\text{sig}(t_3 \beta_3 b_3) \simeq \text{sig}(t_1 \beta_1 b_1)$.

Proof. Note that $\text{lm}(t_1 g_1 b_1) = \text{lm}(t_2 g_2 b_2)$. Thus, the result follows from Lemma 3.4.5, 3.4.9, and the properties of G-polynomials. □

Definition 3.4.11. *A set $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ is* complete *if the G-polynomials of all ambiguities of $G^{[\Sigma]}$ are top* sig*-reducible by $G^{[\Sigma]}$.*

A set can be completed by adding G-polynomials to it. The following definition extends the idea of G-polynomials to syzygies. To this end, we define the least common multiple of two module monomials as follows. For $i = 1, 2$, let $\sigma_i = \boldsymbol{x}^{\boldsymbol{a}_i} v_i \varepsilon_j w_i \in M(\Sigma)$ with commutative part $\boldsymbol{x}^{\boldsymbol{a}_i} \in [X]$, noncommutative parts $v_i, w_i \in \langle Y \rangle$, and some $1 \leq j \leq r$. In the special case $Y = \{y\}$, we assume that all appearances of y have been collected in v_i, so that $w_i = 1$.

Now, if $v_{k'}$ is a suffix of v_k and $w_{l'}$ is a prefix of w_l, where $\{k, k'\} = \{l, l'\} = \{1, 2\}$, then $\operatorname{lcm}(\sigma_1, \sigma_2) \coloneqq \operatorname{lcm}(\boldsymbol{x}^{\boldsymbol{a}_1}, \boldsymbol{x}^{\boldsymbol{a}_2}) v_k \varepsilon_j w_l$.

Definition 3.4.12. *Let $\gamma_1, \gamma_2 \in \Sigma$ be such that $\operatorname{sig}(\gamma_i) = c_i \sigma_i$ with $c_i \in R$, $\sigma_i \in M(\Sigma)$ for $i = 1, 2$. Assume that $\operatorname{lcm}(\sigma_1, \sigma_2)$ is defined and, for $i = 1, 2$, let $a_i, b_i \in [X]\langle Y\rangle$ be such that $\operatorname{lcm}(\sigma_1, \sigma_2) = a_i \sigma_i b_i$. Also, let r, s be Bézout coefficients of $\gcd(c_1, c_2)$. The* sig-Combination *of γ_1 and γ_2 is* sig-Comb$(\gamma_1, \gamma_2) \coloneqq r a_1 \gamma_1 b_1 + s a_2 \gamma_2 b_2$.

A set $H \subseteq \operatorname{Syz}(I^{[\Sigma]})$ is sig-complete *if any sig-Combination of elements in H is top reducible by H.*

Example 3.4.13. *Consider the mixed algebra $\mathcal{A} = \mathbb{Z}[s, t]\langle x, y\rangle$ and the free $\mathcal{A}$-bimodule $\Sigma = (\mathcal{A} \otimes_{\mathbb{Z}[s,t]} \mathcal{A})^{(\mathcal{E})}$ on $\mathcal{E} = \{\varepsilon_1, \varepsilon_2\}$.*

For $\sigma_1 = stxy\varepsilon_1 y$ and $\sigma_2 = t^2 y \varepsilon_1 yx$, we have

$$\operatorname{lcm}(\sigma_1, \sigma_2) = st^2 xy\varepsilon_1 yx.$$

Assuming that, for $i = 1, 2$, σ_i is the signature monomial of

$$\gamma_1 = 2stxy\varepsilon_1 y + tx\varepsilon_2, \qquad \gamma_2 = 3t^2 y\varepsilon_1 yx - s\varepsilon_1,$$

the sig-Combination of γ_1 and γ_2 is

$$\text{sig-Comb}(\gamma_1, \gamma_2) = 2t\gamma_1 x + (-1)sx\gamma_2 = st^2 xy\varepsilon_1 yx + 2t^2 x\varepsilon_2 x + s^2 x\varepsilon_1.$$

To end this section, we introduce a concept needed later.

Definition 3.4.14. *Let $f^{[\alpha]} \in I^{[\Sigma]}$ and $G^{[\Sigma]} \subseteq I^{[\Sigma]}$. We say that $f^{[\alpha]}$ is* super reducible *by $G^{[\Sigma]}$ if there exist $g^{[\beta]} \in G^{[\Sigma]}$ and $t \in T(\mathcal{A})$, $b \in \langle Y\rangle$ such that $\operatorname{lm}(f) = \operatorname{lm}(tgb)$ and $\operatorname{sig}(\alpha) = \operatorname{sig}(t\beta b)$.*

Note that super reducibility need not imply sig-reducibility as the former only requires equality of the leading monomials, without considering the leading coefficients. However, if a set of reducers is complete, a super reducible element is also sig-reducible. This is the result of Proposition 3.4.15 below, which is an adaptation of [FV21, Prop. 2.17].

Proposition 3.4.15. *Let $f^{[\alpha]} \in I^{[\Sigma]}$ and $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ be complete and a labelled Gröbner basis up to signature* $\mathrm{sig}(\alpha)$*. If $f^{[\alpha]}$ is super reducible by $G^{[\Sigma]}$, then it is also top* sig*-reducible by $G^{[\Sigma]}$.*

Proof. We essentially follow the proof of [FV21, Prop. 2.17]. Super reducibility implies the existence of $g_1^{[\beta_1]} \in G^{[\Sigma]}$ and $t_1 \in T(\mathcal{A})$, $b_1 \in \langle Y \rangle$ such that $\mathrm{lm}(f) = \mathrm{lm}(t_1 g_1 b_1)$ and $\mathrm{sig}(\alpha) = \mathrm{sig}(t_1 \beta_1 b_1)$. If, in fact, $\mathrm{lt}(f) = \mathrm{lt}(t_1 g_1 b_1)$, then $f^{[\alpha]}$ is top sig-reducible by $g_1^{[\beta_1]}$. Otherwise, with $h^{[\delta]} = f^{[\alpha]} - t_1 g_1^{[\beta_1]} b_1$, we have $\mathrm{lm}(h) = \mathrm{lm}(f)$ and $\mathrm{sig}(\delta) \prec \mathrm{sig}(\alpha)$. By assumption, $h^{[\delta]}$ is top sig-reducible by $G^{[\Sigma]}$. Let $g_2^{[\beta_2]}$ be such a reducer with $t_2 \in T(\mathcal{A})$, $b_2 \in \langle Y \rangle$ such that $\mathrm{lt}(h) = \mathrm{lt}(t_2 g_2 b_2)$ and $\mathrm{sig}(\delta) \succeq \mathrm{sig}(t_2 \beta_2 b_2)$. Consequently, we have

$$\mathrm{lt}(f) = \mathrm{lt}(t_1 g_1 b_1) + \mathrm{lt}(t_2 g_2 b_2) \quad \text{and} \quad \mathrm{sig}(\alpha) \succ \mathrm{sig}(\delta) \succeq \mathrm{sig}(t_2 \beta_2 b_2).$$

By Lemma 3.4.10, there exists $g_3^{[\beta_3]} = \text{G-Pol}(\mathfrak{a})$ for some $\mathfrak{a} \in \mathrm{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$ and $t_3 \in T(\mathcal{A}), b_3 \in \langle Y \rangle$ such that $\mathrm{lt}(t_3 g_3 b_3) = \mathrm{lt}(f)$ and $\mathrm{sig}(t_3 \beta_3 b_3) \simeq \mathrm{sig}(t_1 \beta_1 b_1) = \mathrm{sig}(\alpha)$. Since $G^{[\Sigma]}$ is complete, $g_3^{[\beta_3]}$ is top sig-reducible by $G^{[\Sigma]}$, and any reducer of $g_3^{[\beta_3]}$ can be used to top sig-reduce $f^{[\alpha]}$. □

Cover criterion

In the following, we adapt the cover criterion (Theorem 3.3.11) to the mixed algebra. To this end, we first generalise the notion of an ambiguity being covered (Definition 3.3.9).

Definition 3.4.16. *Let $G^{[\Sigma]} \subseteq I^{[\Sigma]}$, $H \subseteq \mathrm{Syz}(I^{[\Sigma]})$, and $f^{[\alpha]}, g^{[\beta]} \in I^{[\Sigma]}$ with $f, g \neq 0$. An ambiguity $\mathfrak{a} \in \mathrm{amb}(f^{[\alpha]}, g^{[\beta]})$ is* covered *by $(G^{[\Sigma]}, H)$ if there exist $h^{[\delta]} \in G^{[\Sigma]}$, $\gamma \in H$, and $t, t' \in T(\mathcal{A})$, $b, b' \in \langle Y \rangle$ such that the following conditions hold:*

- $\mathrm{LM}(\mathfrak{a}) \succ \mathrm{lm}(thb)$*;*
- $\mathrm{SIG}(\mathfrak{a}) = \mathrm{sig}(t\delta b) + \mathrm{sig}(t'\gamma b')$*;*

Remark 3.4.17. *In Definition 3.4.16, either of t and t' can also be 0. If $t = 0$, then $\mathrm{lm}(thb) = 0$ and the first condition is trivially fulfilled.*

The property of being covered defined above differs in one main point from Definition 3.3.9 for labelled polynomials over coefficient fields: We now need to consider linear combinations to form the signature. Also, now equality of signatures is required whereas in the free case similarity was sufficient. This requirement comes from the fact that we deal with coefficient rings now and is also necessary in the commutative case, see [FV21, Def. 2.18].

The following theorem is a generalisation of Theorem 3.3.11 and an adaptation of [FV21, Thm. 3.1]. We note that, while all implications of Theorem 3.3.11 could be generalised, in the following we only state the relevant characterisation.

Theorem 3.4.18. *Let $\sigma \in T(\Sigma)$, $G^{[\Sigma]} \subseteq I^{[\Sigma]}$, and $H \subseteq \mathrm{Syz}(I^{[\Sigma]})$ be such that the following conditions hold:*

- *for all $g^{[\beta]} \in G^{[\Sigma]} : g \neq 0$;*
- *for all $\varepsilon_i \prec \sigma$, there exists $\beta_i \in H \cup \{\beta \mid g^{[\beta]} \in G^{[\Sigma]}\}$ with $\mathrm{sig}(\beta_i) = \varepsilon_i$;*
- *$G^{[\Sigma]}$ is complete and H is* sig*-complete;*

If all regular ambiguities $\mathfrak{a}$ of $G^{[\Sigma]}$ with $\mathrm{SIG}(\mathfrak{a}) \prec \sigma$ are covered by $(G^{[\Sigma]}, H)$, then $G^{[\Sigma]}$ is a labelled Gröbner basis of $I^{[\Sigma]}$ up to signature σ and H is a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ up to signature σ.

Remark 3.4.19. *Comparing Theorem 3.4.18 with the corresponding Theorem 3.3.11 in the free setting, we see that in the second hypothesis of the theorem, we now have to require equality of signatures whereas in Theorem 3.3.11 similarity was sufficient.*

Proof of Theorem 3.4.18. Assume, for contradiction, that the theorem is wrong. Then there exists $f^{[\alpha]} \in I^{[\Sigma]}$ with $\mathrm{sig}(\alpha) \prec \sigma$ such that either $f \neq 0$ and $f^{[\alpha]}$ is not sig-reducible by $G^{[\Sigma]}$ or $f = 0$ and α is not reducible by H. Pick such $f^{[\alpha]}$ with minimal signature. Note that this means that $G^{[\Sigma]}$ is a labelled Gröbner basis up to signature $\mathrm{sig}(\alpha)$.

Let $g_1^{[\beta_1]} \in G^{[\Sigma]}$, $\gamma_1 \in H$ and $t_1, t_1' \in T(\mathcal{A})$, $b_1, b_1' \in \langle Y \rangle$ such that

$$\mathrm{sig}(\alpha) = \mathrm{sig}(t_1 \beta_1 b_1) + \mathrm{sig}(t_1' \gamma_1 b_1'). \tag{3.7}$$

By assumption, such a decomposition exists (in fact, with either $t_1 = 0$ or $t_1' = 0$ but we do not require this for (3.7)). We select these elements so that $\mathrm{lm}(t_1 g_1 b_1)$ is minimal and claim that $t_1 g_1^{[\beta_1]} b_1$ is not regular top sig-reducible by $G^{[\Sigma]}$.

To prove this claim, suppose that $t_1 g_1^{[\beta_1]} b_1$ is regular top sig-reducible by $g_2^{[\beta_2]}$. Then there exist $t_2 \in T(\mathcal{A}), b_2 \in \langle Y \rangle$ such that $\mathrm{lt}(t_1 g_1 b_1) = \mathrm{lt}(t_2 g_2 b_2)$ and $\mathrm{sig}(t_1 \beta_1 b_1) \succ \mathrm{sig}(t_2 \beta_2 b_2)$. By Lemma 3.4.5, there exists a regular ambiguity $\mathfrak{a} \in \mathrm{amb}(g_1^{[\beta_1]}, g_2^{[\beta_2]})$ and $t_3 \in T(\mathcal{A}), b_3 \in \langle Y \rangle$ such that

$$t_3 \textsc{lm}(\mathfrak{a}) b_3 = \mathrm{lm}(t_i g_i b_i) \quad \text{and} \quad t_3 \textsc{sig}(\mathfrak{a}) b_3 = \mathrm{sig}(t_1 \beta_1 b_1). \tag{3.8}$$

Since $\textsc{sig}(\mathfrak{a}) \preceq \mathrm{sig}(t_1 \beta_1 b_1) \prec \sigma$, the ambiguity $\mathfrak{a}$ is covered by $(G^{[\Sigma]}, H)$. So there exist $h^{[\delta]} \in G^{[\Sigma]}$, $\gamma \in H$ and $t, t' \in T(\mathcal{A})$, $b, b' \in \langle Y \rangle$ such that $\textsc{lm}(\mathfrak{a}) \succ \mathrm{lm}(thb)$ and $\textsc{sig}(\mathfrak{a}) = \mathrm{sig}(t\delta b) + \mathrm{sig}(t'\gamma b')$. Combining this with (3.7) and (3.8) yields

$$\begin{aligned} &\mathrm{lm}(t_3 thbb_3) \prec t_3 \textsc{lm}(\mathfrak{a}) b_3 = \mathrm{lm}(t_1 g_1 b_1), \\ &\mathrm{sig}(\alpha) = \mathrm{sig}(t_3 t \delta b b_3) + \mathrm{sig}(t_3 t' \gamma b' b_3) + \mathrm{sig}(t_1' \gamma_1 b_1'). \end{aligned}$$

Let $\pi = \text{sig-Comb}(\gamma_1, \gamma)$, which is well-defined and contained in H because H is sig-complete. By definition, $\mathrm{sig}(\pi)$ divides the sum $\mathrm{sig}(t_3 t' \gamma b' b_3) + \mathrm{sig}(t_1' \gamma_1 b_1')$. Therefore, the pair $(h^{[\delta]}, \pi) \in G^{[\Sigma]} \times H$ yields a decomposition of $\mathrm{sig}(\alpha)$ with smaller leading monomial than $\mathrm{lm}(t_1 g_1 b_1)$; a contradiction to the minimality of $\mathrm{lm}(t_1 g_1 b_1)$.

Thus, $t_1 g_1^{[\beta_1]} b_1$ is not regular top sig-reducible. Now, we distinguish between two cases depending on whether $f = 0$ or not.

Case 1 First, we consider the case $f \neq 0$. Observe that $\mathrm{lm}(f) \neq \mathrm{lm}(t_1 g_1 b_1)$ as otherwise $f^{[\alpha]} - t_1' 0^{[\gamma_1]} b_1'$ would be super reducible by $g_1^{[\beta_1]}$, and thus, by Proposition 3.4.15, top sig-reducible by $G^{[\Sigma]}$. But then also $f^{[\alpha]}$ would be top sig-reducible – a contradiction. Now, let $f'^{[\alpha']} = f^{[\alpha]} - t_1 g_1^{[\beta_1]} b_1 - t_1' 0^{[\gamma_1]} b_1'$. Then $\mathrm{sig}(\alpha') \prec \mathrm{sig}(\alpha)$ and $\mathrm{lt}(f') = \max\{\mathrm{lt}(f), \mathrm{lt}(t_1 g_1 b_1)\} \neq 0$. By minimality of $\mathrm{sig}(\alpha)$, $f'^{[\alpha']}$ is top sig-reducible by $G^{[\Sigma]}$. But this implies that $f^{[\alpha]}$ or $t_1 g_1^{[\beta_1]} b_1$ is regular top sig-reducible by $G^{[\Sigma]}$, which is a contradiction.

Case 2 Now, we consider the case $f = 0$. To this end, let $f'^{[\alpha']}$ be like before, and note that $\mathrm{lt}(f') = -\mathrm{lt}(t_1 g_1 b_1)$. If $f' \neq 0$, then $f'^{[\alpha']}$ being top sig-reducible implies that $t_1 g_1^{[\beta_1]} b_1$

is regular top sig-reducible by $G^{[\Sigma]}$, which is a contradiction. Thus, $f' = -t_1 g_1 b_1 = 0$, and by assumption on $G^{[\Sigma]}$, we can conclude $t_1 = 0$, showing that α is reducible by $\gamma_1 \in H$. □

Like in the free case (see Corollary 3.3.13), Theorem 3.4.18 can be extended to characterise full labelled Gröbner bases.

Corollary 3.4.20. *Let $G^{[\Sigma]} \subseteq I^{[\Sigma]}$ and $H \subseteq \mathrm{Syz}(I^{[\Sigma]})$ be such that the following conditions hold:*

- *for all $g^{[\beta]} \in G^{[\Sigma]} : g \neq 0$;*
- *for all ε_i, there exists $\beta_i \in H \cup \{\beta \mid g^{[\beta]} \in G^{[\Sigma]}\}$ with $\mathrm{sig}(\beta_i) = \varepsilon_i$;*
- *$G^{[\Sigma]}$ is complete and H is* sig*-complete;*

If all regular ambiguities of $G^{[\Sigma]}$ are covered by $(G^{[\Sigma]}, H)$, then $G^{[\Sigma]}$ is a labelled Gröbner basis of $I^{[\Sigma]}$ and H is a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$.

Algorithm

Equipped with Theorem 3.4.18, we can describe with Algorithm 6 an algorithm for enumerating labelled Gröbner bases and Gröbner bases of syzygy modules in the mixed algebra. This algorithm combines key elements of Kandri-Rody and Kapur's algorithm for commutative Gröbner bases over $\mathbb{Z}$ with signatures [FV21] and Algorithm 2 for computing labelled Gröbner bases in free algebras over fields. The difference between the free algebra and the mixed algebra is apparent in the fact that now, in addition to S-polynomials, also G-polynomials have to be computed. Additionally, the computation of ambiguities and the reductions are done using the definitions from the mixed algebra.

Like the previous Gröbner algorithms, also Algorithm 6 runs on a loop, processing elements from a queue. However, now, instead of ambiguities, the queue contains directly the S- and G-polynomials. This is due to the fact that S- and G-polynomials have to be treated differently. At each step, the algorithm selects a polynomial using a fair selection strategy, and, if not redundant, reduces it and forms new polynomials from it, adding them to the queue. Just like Algorithm 2, also this algorithm ensures that, for every regular ambiguity of $G^{[\Sigma]}$, an element is eventually added which covers it.

Algorithm 6: Labelled Gröbner basis algorithm

Input: $(f_1, \ldots, f_r) \in \mathcal{A}^r$

Output (if the algorithm terminates):

- $G^{[\Sigma]}$ a labelled Gröbner basis of the labelled module $I^{[\Sigma]}$ generated by $f_1, \ldots, f_r$;
- $H \subseteq \mathrm{Syz}(I^{[\Sigma]})$ a Gröbner basis of $I^{[\Sigma]}$;

$G^{[\Sigma]} \leftarrow \emptyset$;
$H \leftarrow \emptyset$;
$\mathtt{poly} \leftarrow \left\{(f_1^{[\varepsilon_1]}, \mathtt{N}), \ldots, (f_r^{[\varepsilon_r]}, \mathtt{N})\right\}$;
while $\mathtt{poly} \neq \emptyset$ **:**
    select an element $(f^{[\alpha]}, \mathtt{type})$ from $\mathtt{poly}$ using a fair strategy and remove it;
    if $\mathtt{type}$ is not $\mathsf{S}(\mathfrak{a})$ or $\mathfrak{a}$ is not covered by $(G^{[\Sigma]}, H)$ **:**
        $f'^{[\alpha']} \leftarrow$ result of regular sig-reducing $f^{[\alpha]}$ by $G^{[\Sigma]}$;
        if $f' = 0$ **:**
            $H \leftarrow H \cup \{\alpha'\}$, and make H sig-complete;
        else:
            $G^{[\Sigma]} \leftarrow G^{[\Sigma]} \cup \{f'^{[\alpha']}\}$;
            for $g^{[\beta]} \in G^{[\Sigma]}$ and $\mathfrak{a} \in \mathrm{amb}(f'^{[\alpha']}, g^{[\beta]})$ **:**
                add $(\text{S-Pol}(\mathfrak{a}), \mathsf{S}(\mathfrak{a}))$ to $\mathtt{poly}$ if $\mathfrak{a}$ is regular;
                add $(\text{G-Pol}(\mathfrak{a})), \mathsf{G}(\mathfrak{a}))$ to $\mathtt{poly}$;
return $G^{[\Sigma]}, H$

As in the classical case, S- and G-polynomials are added for different purposes: S-polynomials create new leading monomials and G-polynomials create new leading coefficients for existing leading monomials. The cover criterion concerns the existence of leading monomials in the basis, and can be used to skip over S-polynomials that correspond to ambiguities which are already covered. The situation is different for G-polynomials: even if the corresponding ambiguity is covered, it is necessary to process and add the G-polynomial to complete the set $G^{[\Sigma]}$.

This requires keeping track of the construction of each labelled polynomial in the algorithm. For this purpose, the main queue $\mathtt{poly}$ contains pairs $(f^{[\alpha]}, \mathtt{type})$, where $\mathtt{type}$ is either the symbol N indicating that f is one of the input polynomials, or the symbol $\mathsf{S}(\mathfrak{a})$, respectively $\mathsf{G}(\mathfrak{a})$, indicating that $f^{[\alpha]}$ is the S-polynomial, respectively the G-polynomial, of the ambiguity $\mathfrak{a}$.

We note that Algorithm 6 is stated using labelled polynomials, carrying their entire module representation. However, like in the free case, all relevant properties only require considering the signature of the module representation, and the algorithm can be adapted to work with signature polynomials instead of labelled polynomials. In that case, the full module representations can be recovered *a posteriori* using the techniques from Section 3.3.3.

For brevity, the presentation of the algorithm is simplified concerning the handling of the queue of pairs. Unlike in the commutative case or the free case over fields, the inserting from line 12 to 14 generally involves infinitely many elements. In order for the algorithm to correctly enumerate a labelled Gröbner basis, it needs to process all possible S- and G-polynomials, so it cannot enter an infinite loop inside the main loop (necessarily infinite).

Instead, Algorithm 6 should be modified to add a spooling machinery ensuring that all pairs are processed. We detail a possible implementation of such a mechanism. The main idea is that the algorithm must ensure that `poly` is finite at all times. The spooling mechanism ensures that by only adding a finite number of pairs to `poly` at each iteration. The additional mechanism would then run in parallel for all sources of pairs, ensuring that the pairs are processed in a fair order.

More precisely, we assume that we know a way to enumerate ambiguities, that is, we are given two functions:

- `firstamb` taking as input two labelled polynomials $f^{[\alpha]}$ and $g^{[\beta]}$ and returning an ambiguity $\mathfrak{a}_0$ between them;
- `nextamb` taking as input an ambiguity $\mathfrak{a}$ of two labelled polynomials $f^{[\alpha]}$ and $g^{[\beta]}$, and returning another ambiguity $n(\mathfrak{a})$ between them;

with the property that $\{\mathfrak{a}_0, n(\mathfrak{a}_0), n(n(\mathfrak{a}_0)), \dots\} = \mathrm{amb}(f^{[\alpha]}, g^{[\beta]})$.

The algorithm would maintain an additional variable `spool`, which is a finite set of ambiguities and initialised to the empty set before the main loop starting in line 4. The lines 12 to 14 would be replaced by population of that list:

Algorithm 6a: Add the ambiguities to the spooling

```
12a for g^[β] ∈ G^[Σ] :
12b     a_0 ← firstamb(f'^[α'], g^[β]);
12c     add a_0 to spool;
```

The construction of the pairs would be done closer to their actual use, by adding the following lines at the very end of the main loop, outside the "else"-block and just prior to selection of the next pair:

Algorithm 6b: Construct the pairs and update `spool`

```
14a for 𝔞 ∈ spool :
14b     add (S-Pol(𝔞), S(𝔞)) to poly if 𝔞 is regular;
14c     add (G-Pol(𝔞), G(𝔞)) to poly;
14d spool ← {nextamb(𝔞) | 𝔞 ∈ spool};
```

This has the effect of adding finitely many new elements to `poly`, namely at most 2 for each element of `spool`, and updating `spool` to generate new elements the next time the code is evaluated.

The main property that the machinery must satisfy is that it yields an overall fair selection strategy, correctly enumerating all ambiguities of pairs of polynomials.

Proposition 3.4.21. *In Algorithm 6, with the structure described above, if the selection of ambiguities in* `poly` *in line 5 is done following a fair strategy, for all ambiguities of* $G^{[\Sigma]}$*, the corresponding S- and G-polynomials are eventually selected and processed.*

Proof. Let $f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]}$ and let $\mathfrak{a}$ be an ambiguity between them. Since the function `nextamb` enumerates all ambiguities of $f^{[\alpha]}, g^{[\beta]}$, eventually the insertion mechanism reaches the ambiguity $\mathfrak{a}$, at which point the corresponding S- and G-polynomials are either inserted into `poly` or discarded. Since selection in `poly` is done following a fair strategy, eventually both the S- and the G-polynomial of $\mathfrak{a}$ will be selected and processed. □

In the following, we assume that Algorithm 6 is adapted to include the spooling mechanism described above. Then we can show the following correctness result analogous to Theorem 3.3.16.

Theorem 3.4.22. *Algorithm 6 correctly enumerates a labelled Gröbner basis of* $I^{[\Sigma]}$ *and a Gröbner basis of* $\mathrm{Syz}(I^{[\Sigma]})$.

Proof. We prove that the algorithm enforces the requirements of Theorem 3.4.18. First, by construction, all elements added to $G^{[\Sigma]}$ have nonzero polynomial part. All signatures ε_i are processed, and result in either an element in H or in $G^{[\Sigma]}$, depending on whether $f_i^{[\varepsilon_i]}$ regular sig-reduces to 0 or not. The bases $G^{[\Sigma]}$ and H are complete and sig-complete respectively, because all G-polynomials and all sig-Combinations are added to them. Finally, the algorithm ensures that all regular ambiguities it considers are covered: either the ambiguity $\mathfrak{a}$ is already covered, or an element is added to either $G^{[\Sigma]}$ or H with signature $\textsc{sig}(\mathfrak{a})$. Either way, this element covers the ambiguity $\mathfrak{a}$. Finally, by Proposition 3.4.21, it processes all ambiguities, covering all of them eventually. □

Remark 3.4.23. *If the used module order is fair, the functions* `firstamb` *and* `nextamb` *enumerate ambiguities by increasing signature, and if the selection strategy processes elements by increasing signatures, then, whenever Algorithm 6 considers a labelled polynomial $f^{[\alpha]}$, the sets $G^{[\Sigma]}$ and H are a labelled Gröbner basis and a Gröbner basis respectively up to signature* $\mathrm{sig}(\alpha)$.

To end this section, we note that Algorithm 6 provides a semi-decision procedure for ideal membership in the mixed algebra, analogous to Buchberger's algorithm (Algorithm 1) in the free algebra over a field.

Lemma 3.4.24. *Let $f, f_1, \ldots, f_r \in \mathcal{A}$. If $f \in (f_1, \ldots, f_r)$, then this fact can be verified in finite time.*

Proof. Using Algorithm 6 and exploiting the fact that any labelled Gröbner basis is also a Gröbner basis, the proof is analogous to that of Lemma 2.4.61. □

Corollary 3.4.25. *The ideal membership problem (Problem 2.2.28) in $\mathcal{A}$ is semi-decidable. In particular, the ideal membership problem in $R\langle X\rangle$ is semi-decidable, where R is a commutative principal ideal domain.*

3.4.2 Elimination criteria

For simplicity, we presented Algorithm 6 stripped to its main loop, with only the cover criterion necessary for the loop invariant. But just like Algorithm 2 for the free algebra, also this algorithm can be equipped with powerful elimination criteria to skip redundant ambiguities and polynomials. In this section, we list a few of these additional criteria that can be added to the algorithm. They all work similarly to their counterpart in the commutative case, ensuring that elements are covered (for S-polynomials) or sig-reducible (for G-polynomials). First, we focus on G-polynomials.

Corollary 3.4.26. *Let* $f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]}$ *be such that* $f, g \neq 0$ *and* $\mathfrak{a} \in \operatorname{amb}(f^{[\alpha]}, g^{[\beta]})$. *If* $\text{G-Pol}(\mathfrak{a})$ *is top* sig*-reducible by* $G^{[\Sigma]}$*, then it can be discarded.*

Proof. It is a direct consequence of Theorem 3.4.18: if $\text{G-Pol}(\mathfrak{a})$ is top sig-reducible by $G^{[\Sigma]}$, it is not required for making $G^{[\Sigma]}$ complete, and thus, can be discarded. □

The criterion allows to avoid all reductions of G-polynomials to zero. In fact, it avoids all top reductions of G-polynomials entirely. Nevertheless, it still makes sense to tail reduce G-polynomials in practice. As a special case of Corollary 3.4.26, we see that, if $\operatorname{lc}(f) \mid \operatorname{lc}(g)$, then all their G-polynomials can be discarded as they are all top sig-reducible by $f^{[\alpha]}$.

Corollary 3.4.27. *Let* $f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]}$ *be such that* $f, g \neq 0$ *and* $\operatorname{lc}(f) \mid \operatorname{lc}(g)$. *Then all G-polynomials of* $f^{[\alpha]}$ *and* $g^{[\beta]}$ *can be discarded.*

Proof. In this situation, $\gcd(\operatorname{lc}(f), \operatorname{lc}(g)) = \operatorname{lc}(f)$, and thus, all G-polynomials of $f^{[\alpha]}$ and $g^{[\beta]}$ are top sig-reducible by $f^{[\alpha]}$. □

As a consequence of Corollary 3.4.27, we see that all G-polynomials are redundant if we work over a coefficient field.

We now move to S-polynomials. Like explained in Section 3.3.2, excluding covered ambiguities encapsulates part of the syzygy criterion (Proposition 3.3.19), stating that any regular ambiguity whose signature is reducible by the signature of a syzygy in H can be discarded, and the singular criterion (Corollary 3.3.21), which says that, for each signature, at most one regular ambiguity (the one with minimal leading monomial) has to be considered.

Additionally, like in the free case, we can make use of trivial syzygies to detect redundant elements, yielding the F5 criterion in this setting.

Corollary 3.4.28 (F5 criterion)**.** *In Algorithm 6, let* $\mathfrak{a} \in \texttt{amb}$ *be regular such that there exist* $f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]}$ *and* $m \in \langle Y \rangle$ *with*

- $\operatorname{sig}(\alpha m g) \not\simeq \operatorname{sig}(f m \beta)$*, and*
- $\operatorname{SIG}(\mathfrak{a})$ *is divisible by* $\max\{\operatorname{sig}(\alpha m g), \operatorname{sig}(f m \beta)\}$.

Then $\mathfrak{a}$ *is covered by the trivial syzygy* $\alpha m g - f m \beta$ *and* S-Pol($\mathfrak{a}$) *can be discarded after adding this trivial syzygy to the set* H*.*

This version of the F5 criterion fully includes Buchberger's coprime criterion for eliminating S-polynomials coming from elements with coprime leading terms, see [LMA23, Lem. 22] for a noncommutative version without signatures. In particular, if $f^{[\alpha]}, g^{[\beta]}$ are such that $\operatorname{lc}(f)$ and $\operatorname{lc}(g)$ as well as the commutative parts of $\operatorname{lm}(f)$ and $\operatorname{lm}(g)$ are coprime, then, for every regular external ambiguity of these elements, the S-polynomial can be discarded after adding a suitable trivial syzygy to H.

Corollary 3.4.29. *Let* $f^{[\alpha]}, g^{[\beta]} \in G^{[\Sigma]}$ *with* $\operatorname{lm}(f) = \boldsymbol{x}^{\boldsymbol{a}} v$ *and* $\operatorname{lm}(g) = \boldsymbol{x}^{\boldsymbol{b}} w$*. If* $\gcd(\operatorname{lc}(f), \operatorname{lc}(g)) = \gcd(\boldsymbol{x}^{\boldsymbol{a}}, \boldsymbol{x}^{\boldsymbol{b}}) = 1$*, then all S-polynomials coming from regular external ambiguities of* $f^{[\alpha]}$ *and* $g^{[\beta]}$ *can be discarded after adding the set of trivial syzygies* $\bigcup_{m \in \langle Y \rangle} \{\alpha m g - f m \beta, \beta m f - g m \alpha\}$ *to the set* H*.*

Proof. By definition of external ambiguities and the assumption on $\boldsymbol{x}^{\boldsymbol{a}}, \boldsymbol{x}^{\boldsymbol{b}}$, any external ambiguity $\mathfrak{a}$ of $f^{[\alpha]}$ and $g^{[\beta]}$ is of the form $\mathfrak{a} = (\boldsymbol{x}^{\boldsymbol{b}} \otimes mw, \boldsymbol{x}^{\boldsymbol{a}} v m \otimes 1)$ or $\mathfrak{a} = (\boldsymbol{x}^{\boldsymbol{b}} w m \otimes 1, \boldsymbol{x}^{\boldsymbol{a}} \otimes mv)$. For the first case, the assumption on the leading coefficients implies

$$\begin{aligned} \operatorname{SIG}(\mathfrak{a}) &= \max\left\{\operatorname{sig}(\operatorname{lc}(g)\boldsymbol{x}^{\boldsymbol{b}} \alpha m w), -\operatorname{sig}(\operatorname{lc}(f)\boldsymbol{x}^{\boldsymbol{a}} v m \beta)\right\} \\ &= \max\left\{\operatorname{sig}(\alpha m \operatorname{lt}(g)), -\operatorname{sig}(\operatorname{lt}(f) m \beta\right\} \\ &= \max\left\{\operatorname{sig}(\alpha m g), -\operatorname{sig}(f m \beta)\right\}. \end{aligned}$$

Since $\mathfrak{a}$ is regular, $\operatorname{sig}(\alpha m g) \not\simeq \operatorname{sig}(f m \beta)$, and thus, Corollary 3.4.28 implies that S-Pol($\mathfrak{a}$) can be discarded after adding $\alpha m g - f m \beta$ to H. The second case follows along the same lines by adding $\beta m f - g m \alpha$ to H. □

Remark 3.4.30. *Buchberger's coprime criterion allows to eliminate, with finitely many checks, an infinite number of S-polynomials. It would be interesting to investigate whether this behaviour can affect the termination of Algorithm 6. Or more generally, can the elimination criteria presented in this section be used to eliminate infinitely many S- and G-polynomials, using only a finite number of operations, and in this way cause the algorithm to terminate? We have not tried to answer this question yet.*

Note that, in contrast to the F5 criterion in the free case (Corollary 3.3.20), we now have to explicitly add the trivial syzygy used in the criterion to the set H. This is because Algorithm 2 explicitly adds all trivial syzygies of $G^{[\Sigma]}$ to the set H at the end of the algorithm. Algorithm 6 does not have to do this because the information of the trivial syzygies is now contained in the external ambiguities of the polynomials. Compare this also with Remark 3.4.6.

We note that, just like in the free case, checking the F5 criterion can be done by constructing on the fly the finitely many syzygies that can possibly apply. This causes the complexity of applying Corollary 3.3.20 in general to be quadratic in the size of $G^{[\Sigma]}$. In the following Section 3.4.3, we discuss how to reduce this cost to linear for homogeneous polynomials and certain module orders.

3.4.3 Application of the mixed algebra: Homogenisation

It is frequently preferable to work with homogeneous polynomials when computing Gröbner bases, in both the commutative and noncommutative case. Specifically in the context of signatures, they open the possibility to use more efficient orders for the F5 criterion. However, often the considered systems consist of inhomogeneous polynomials. In this case, the process of *homogenisation* can be used to turn such an inhomogeneous system into a homogeneous one. This process of homogenisation naturally leads to elements in a mixed algebra. Thus, in this section, we discuss one important application of the mixed algebra, namely computations with homogenised polynomials.

The notion of (standard) degree extends in a straightforward way from polynomials (Example 2.2.35) to mixed polynomials. In particular, given a mixed polynomial

$$f = \sum_{i=1}^{d} c_i \boldsymbol{x}^{\boldsymbol{a}_i} w_i \in R[X]\langle Y\rangle,$$

its degree $\deg(f)$ is the maximum degree of the monomials $\boldsymbol{x}^{\boldsymbol{a}_i} w_i$ appearing in f, that is, $\deg(f) = \max_i \deg(\boldsymbol{x}^{\boldsymbol{a}_i}) + \deg(w_i)$. It is called homogeneous if all its terms have the same degree. The *homogenisation* f^h of f is the polynomial

$$f^h = \sum_{i=1}^{d} c_i h^{\deg(f)-\deg(\boldsymbol{x}^{\boldsymbol{a}_i} w_i)} \boldsymbol{x}^{\boldsymbol{a}_i} w_i \in R[h, X]\langle Y\rangle,$$

where h is a new *homogenisation variable*. Note that the homogenisation f^h is a homogeneous polynomial, and evaluating it at $h = 1$ yields back f. In that construction, the homogenisation variable h commutes with all other indeterminates. Hence, in particular, if $f_1, \dots, f_r$ are elements in a free algebra $R\langle X\rangle$, then their homogenisations $f_1^h, \dots, f_r^h$ are mixed polynomials in $R[h]\langle X\rangle$.

When working with inhomogeneous polynomials, applying the F5 criterion is essentially quadratic in the size of the partially computed signature Gröbner basis. The reason for this is that one does not know in advance which signature realises the maximum in the second condition of Corollary 3.3.20. This problem can be partially remedied by considering module orders which make this comparison easy, such as the position-over-term order $\preceq_{\mathrm{PoT}}$, first comparing the index $\mathrm{ind}(\sigma) \coloneqq i$ of a signature $\sigma = a\varepsilon_i b$ before comparing the terms a and b (see Example 3.1.14). However, to fully utilise the F5 criterion in this case, elements have to be processed by increasing signatures, which does not constitute a fair selection strategy in the noncommutative setting if $\preceq_{\mathrm{PoT}}$ is used. An alternative can be to decouple the selection strategy from the module order, but then one cannot expect that all elements necessary to use the F5 criterion will be present in time for its use.

Another possibility, when dealing with homogeneous polynomials is to use the degree-over-position-over-term order $\preceq_{\mathrm{DoPoT}}$. This order first considers the degree $\deg(\sigma) \coloneqq \deg(af_ib)$ of a signature $\sigma = a\varepsilon_i b$. In the homogeneous case, we have $\deg(f) = \deg(\mathrm{sig}(\alpha))$ for all labelled polynomials $f^{[\alpha]}$ computed during the execution of a signature-based algorithm, while in the inhomogeneous case, it only holds that $\deg(f) \leq \deg(\mathrm{sig}(\alpha))$, with the latter sometimes being called the *sugar degree* of f [Gio+91], see also [Ede13, Sec. 4] for a discussion of the sugar degree in relation to signatures. Recall from Example 3.1.14 that $\preceq_{\mathrm{DoPoT}}$ first compares the degree of the signatures, then their indices, and finally the monomials a and b. This order is fair, and can thus be used as a selection strategy in Algorithm 6, by always picking the element with smallest signature in the queue `poly` in line 5. Moreover, this order makes it possible to verify the conditions of the F5 criterion

easily, and the incremental calculation ensures that all the required signatures are available when applying the criterion.

Corollary 3.4.31 (F5 criterion optimised)**.** *In Algorithm 6, assume that the input* $(f_1, \dots, f_r) \in \mathcal{A}^r$ *is homogeneous and that* $\preceq_{DoPoT}$ *is used as a module order with a graded monomial order.*

Let $g^{[\beta]} \in G^{[\Sigma]}$*. For all* $m \in \langle Y \rangle$ *and* $j > \mathrm{ind}(\beta)$*, the module terms* $\mathrm{lt}(g)m\varepsilon_j$ *and* $\varepsilon_j m\mathrm{lt}(g)$ *are signatures of trivial syzygies between* $g^{[\beta]}$ *and* $f_j^{[\varepsilon_j]}$*.*

If, additionally, elements in Algorithm 6 are processed by increasing signature and all ambiguities with signature divisible by a signature as above are discarded, then all ambiguities whose signature is divisible by the signature of a trivial syzygy $\sigma = g_2 m\beta_1 - \beta_2 m g_1$*, with* $\mathrm{ind}(\beta_1) \neq \mathrm{ind}(\beta_2)$*, are discarded in this way.*

Proof. They are the signature of the trivial syzygies $gm\varepsilon_j - \beta m f_j$ and $\varepsilon_j mg - f_j m\beta$ respectively, observing that both members of these trivial syzygies must have the same degree. Note that $\deg(g) = \deg(\mathrm{sig}(\beta))$ as $g^{[\beta]} \in G^{[\Sigma]}$ and all elements in $G^{[\Sigma]}$ satisfy this property. This follows from the fact that this property holds for the basis elements $f_i^{[\varepsilon_i]}$, $i = 1, \dots, r$, and, since the input is homogeneous, it is also preserved by all operations done in Algorithm 6 (S/G-polynomial formation, sig-reductions).

For the second part, the signature of the trivial syzygy σ is $\mathrm{sig}(g_2 m\beta_1)$ if $\mathrm{ind}(\beta_1) > \mathrm{ind}(\beta_2)$ and $-\mathrm{sig}(\beta_2 m g_1)$ otherwise. Both cases are handled similarly, so assume that we are in the first one. Then, the signature of σ is a discarded signature, obtained from $g_2^{[\beta_2]}$. It remains to prove that $\mathrm{sig}(\beta_2) \prec \mathrm{SIG}(\mathfrak{a})$, to ensure that $g_2^{[\beta_2]}$ was computed in time for discarding $\mathfrak{a}$. Since $\mathrm{SIG}(\mathfrak{a})$ is divisible by $\mathrm{sig}(\sigma) \succ \mathrm{sig}(\beta_2 m g_1)$, we have $\deg(\mathrm{SIG}(\mathfrak{a})) \geq \deg(\mathrm{sig}(\beta_2))$. With $\mathrm{ind}(\mathrm{SIG}(\mathfrak{a})) = \mathrm{ind}(\beta_1) > \mathrm{ind}(\beta_2)$, this yields $\mathrm{sig}(\beta_2) \prec \mathrm{SIG}(\mathfrak{a})$. □

3.5 Experiments

In this section, we make two comparisons. First, we compare a noncommutative signature-based algorithm to a classical Gröbner basis algorithm in the free algebra. More precisely, we compare a signature-based F4 algorithm, which is an adaptation of Algorithm 3 and described in more detail in Section 6.3.2, to the classical noncommutative F4

algorithm [Xiu12, Sec. 5.4]. We have implemented both algorithms in our software packages using the same underlying data structures. This makes for a fair comparison, allowing to actually compare the effect of signatures and not the effect of different implementations. We give data about the number of S-polynomials computed and reduced as well as about the number of reductions to zero and the overall computation time when computing (signature) Gröbner bases for certain benchmark examples.

Secondly, we compare the mixed algebra setting to other (more naive) approaches for computing noncommutative (signature) Gröbner bases involving some commutative variables. More precisely, we compare Algorithm 6 to the following two approaches:

1. classical Gröbner basis computations in the free algebra where commutator relations are added explicitly to the generators of an ideal;

2. signature Gröbner basis computations in the free algebra where commutator relations are added explicitly to the generators of an ideal but are given a trivial signature 0 so that sig-reductions by these relations are always possible;

For all classical Gröbner basis computations, we use our SageMath package `operator_gb` and, for the signature-based computations in the free algebra, we use our package `signature_gb`. The two packages are described in more detail in Section 6.1 and 6.3 respectively. For the signature-based computations in the mixed-algebra, we use our prototype implementation of Algorithm 6 over coefficient fields for SageMath, including the criteria for S-polynomial elimination discussed in Section 3.4.2 (G-polynomials are redundant over fields).

All computations are performed over the coefficient field $\mathbb{Q}$ and a degree-lexicographic monomial order is used in combination with $\preceq_{\mathrm{DoPoT}}$ for the signature-based computations.

For the first comparison, we consider the benchmark examples listed in Table 3.1. The first three examples are homogeneous and taken from [LL09]. As done in [LL09], we compute truncated (signature) Gröbner bases of these homogeneous ideals up to certain degree bounds. The designated degree bounds are indicated by the number after the "`-`" in the name of each example in Table 3.2. For example, `lp1-12` means that we compute a Gröbner basis of the example `lp1` up to degree 12. Additionally, we also consider two inhomogeneous ideals derived from finite generalised triangular groups taken from [RS02, Thm. 2.12] as also done in [Xiu12]. Both of these ideals have finite (signature) Gröbner bases.

Example	Generators of the ideal
`braid3`	$yxy - zyz, xyx - zxy, zxz - yzx, x^3 + y^3 + z^3 + xyz$
`lp1`	$z^4 + yxyx - xy^2x - 3zyxz, x^3yxy - xyx, zyx - xyz + zxz$
`lv2`	$xy + yz, x^2 + xy - yx - y^2$
`tri1`	$x^3 - 1, y^2 - 1, (yxyxyx^2yx^2)^2 - 1$
`tri3`	$x^3 - 1, y^3 - 1, (yxyx^2)^2 - 1$

Table 3.1: Benchmark examples for the comparison between signature-based computations and classical Gröbner basis computations.

Table 3.2 compares the number of S-polynomials computed and reduced and the number of reductions to zero that occur while computing (truncated) (signature) Gröbner bases for the examples in Table 3.1. We also list the computation times (in seconds). As evidenced by the results, the signature-based algorithm can provide a drastic advantage over conventional Gröbner basis computations, requiring significantly fewer S-polynomials and reductions to zero. Notably, for the homogeneous examples, the signature-based algorithm consistently surpasses the standard method across all metrics, yielding a speed-up of up to 40 times. For two of the considered examples (`lp1` and `lv2`), there are even no zero reductions at all. However, for the inhomogeneous examples, the scenario is reversed, with the signature-based approach needing more S-polynomials and a greater number of reductions to zero. Nevertheless, it is noteworthy that, for all examples, the proportion of reductions to zero is consistently lower in the signature-based algorithm, indicating that this algorithm spends proportionally less time on these redundant computations. While this behaviour is intriguing, we have not investigated it in more detail yet.

For the second comparison concerning the mixed algebra, we report in Table 3.3 on the number of S-polynomials reduced and the number of zero reductions that appear during the computation of (signature) Gröbner bases for the following benchmark examples. We do not list computation times here, as our mixed algebra implementation uses substantially more naive data structures and routines than the other implementations, not allowing for a fair comparison. Nevertheless, we note that, despite this, the mixed algebra approach still remains the most efficient for certain examples. For instance, in the case of `heis-9`, the mixed algebra approach yields a computation time of 2.3 seconds, while the classical and signature-based approaches in the free algebra require 118 seconds and more than 10 minutes, respectively.

Example	classical Gröbner basis			signature-based algorithm		
	S-poly	red. to 0	time	S-poly	red. to 0	time
`braid3-10`	844	496	2.6	422	7	0.3
`braid3-11`	1896	1215	66.6	1005	70	1.6
`lp1-12`	204	136	9.9	97	0	2.2
`lp1-13`	307	213	121.6	144	0	12.2
`lv2-30`	8850	8038	26.8	814	0	1.1
`lv2-40`	21 146	19 664	176.5	1484	0	4.5
`tri1`	261	199	0.2	573	399	0.7
`tri3`	197	150	0.1	327	241	0.1

Table 3.2: Number of S-polynomials and reductions to zero during the computation of (truncated) (signature) Gröbner bases for different benchmark examples as well as computation times (in sec).

We consider the benchmark examples listed below. Note that all considered ideals are homogeneous.

- The example `ufn1h` is taken from [LL09] and concerns the ideal

$$\begin{aligned}\texttt{ufn1h} = \Big(&a^2 - ah, b^2 - bh, c^2 - ch, d^2 - dh, aba - abh,\\ &bab - abh, aca - ach, cac - ach, ada - adh, dad - adh,\\ &bcb - bch, cbc - bch, bdb - bdh, dbd - bdh, cdc - cdh, dcd - cdh\Big)\end{aligned}$$

  in $\mathbb{Q}[h]\langle a, b, c, d\rangle$.

- The example `ih` is taken from [LMA23, Ex. 31] and is related to Iwahori-Hecke algebras [Hum90], which are deformations of the group algebra of a Coxeter group. For example, the Iwahori-Hecke algebra of type A_3, is finitely presented as the quotient of $\mathbb{Z}[q, q^{-1}]\langle x, y, z\rangle$ by the ideal

$$\left(x^2 + x - qx - q, y^2 + y - qy - q, z^2 + z - qz - q, zx - xz, yxy - xyx, zyz - yzy\right).$$

Example	classical Gröbner basis		naive signature basis		Algorithm 6	
	S-poly	red. to 0	S-poly	red. to 0	S-poly	red. to 0
`ufn1h-8`	740	604	9075	716	270	120
`ufn1h-9`	1218	1022	28 027	4785	455	243
`ih-8`	456	414	8179	116	35	8
`ih-9`	520	475	22 302	212	37	9
`heis-8`	3917	2871	15 521	87	22	4
`heis-9`	11 555	8520	53 781	222	48	14

Table 3.3: Comparison of different approaches for computing (signature) Gröbner bases in the mixed algebra.

Here, we consider the homogenisation of that ideal in $\mathbb{Q}[q, q^{-1}, h]\langle x, y, z\rangle$, that is,

$$\texttt{ih} = \Big(x^2 + hx - qx - hq, y^2 + hy - qy - hq, z^2 + hz - qz - hq,$$
$$zx - xz, yxy - xyx, zyz - yzy, h^2 - qq^{-1}\Big).$$

- The example `heis` is related to the discrete Heisenberg group $\langle x, y, z \mid z = xyx^{-1}y^{-1}, xz = zx, yz = zy\rangle$. We consider the homogenisation of the ideal describing these relations in $\mathbb{Q}[z, z^{-1}, h]\langle x, x^{-1}, y, y^{-1}\rangle$, that is,

$$\texttt{heis} = \left(h^3 z - xyx^{-1}y^{-1}, h^2 - zz^{-1}, h^2 - xx^{-1}, h^2 - x^{-1}x, h^2 - yy^{-1}, h^2 - y^{-1}y\right).$$

As before, we compute truncated (signature) Gröbner bases up to fixed degrees for each of these homogeneous ideals, with the used degree bounds indicated by the number after the "`-`" in Table 3.3.

As Table 3.3 shows, Algorithm 6 has to consider significantly fewer S-polynomials. It is also worth noting that this approach yields substantially smaller outputs. For example, the basis computed for `heis-9` in the mixed algebra consists of 34 elements, compared to 3056 elements for the classical approach in the free algebra and more than 50 000 for the naive signature-based algorithm. The main reason for this behaviour is that, in the classical approaches, the commutator relations become oriented reduction rules, which makes them less flexible. This causes a lot of computations that are avoided in the mixed algebra. Additionally, the naive signature-based computation suffers from the fact that the

commutator relations are only "visible" on the polynomial level but not on the signature level. Hence, the elimination criteria cannot be exploited fully because they miss this crucial information. In contrast to this, in the mixed algebra setting, the information about the commutative variables is also directly propagated to the signatures.

4 Theoretical framework for verifying operator statements

Linear operators are found in various areas of mathematics, appearing as ring elements (as in C^*-algebras), matrices, and more generally, as vector space and module homomorphisms. Also many statements in homological algebra, typically expressed in the language of abelian categories, describe properties of linear operators. In this chapter, we develop a framework to efficiently prove the validity of first-order statements about operator identities phrased in any of these settings by verifying ideal membership of noncommutative polynomials in a free algebra. All results of this chapter also appear in our preprint [HRR22b].

Translating operator identities into noncommutative polynomials provides several computational advantages. Naturally, polynomial computations respect linearity. Furthermore, as our framework shows, restrictions imposed by domains and codomains of operators can be ignored when performing polynomial arithmetic. Finally, using the theory of noncommutative Gröbner bases, one can exploit efficient computer algebra implementations and heuristics for polynomial computations, in particular for verifying ideal membership. These routines can also compute cofactor representations, which serve as certificates for ideal membership that can be verified easily and independently.

Noncommutative polynomials have been used previously to model and analyse operator identities. Starting from the pioneering work [HW94; HSW98], where polynomial techniques are used to simplify matrix identities in linear systems theory, over [HS99], where similar methods are used to discover operator identities and to solve matrix equations, until [SL20; Sch21], where proving operator identities via polynomial computations and related questions are addressed. Recently, also a framework was developed that allows to infer the validity of a statement about operators from ideal membership of noncommutative polynomials [RRH21]. This framework can treat propositional statements where several identities (the assumptions) imply another identity (the claim).

Our approach extends this framework, which was the motivation for this work and builds the foundation for our results, in several ways. First of all, we allow more general first-order statements that can include quantifiers, function symbols, and all boolean connectives. Secondly, while [RRH21] only gives a sufficient algebraic condition for an operator statement to hold, we provide with our main result (Theorem 4.4.1) an equivalence between universal truth of operator statements and ideal membership of noncommutative polynomials. This allows us to state with Procedure 8 a semi-decision procedure for verifying the validity of operator statements based on polynomial computations, showing that our approach is complete in that sense.

In our framework, operator statements are treated as statements about morphisms in preadditive semicategories (Definition 2.1.1). These structures provide a natural and very general environment for our application, prescribing only linearity as a structural constraint. In particular, semicategories encompass all the aforementioned settings (rings, matrices, homomorphisms, abelian categories).

To model statements in semicategories, we use many-sorted first-order logic. It extends classical first-order logic by assigning a sort to each symbol. These sorts allow to model objects from different universes and restrict which expressions can be formed. In our context, they are used to represent domains and codomains of operators. We slightly adapt the syntax of many-sorted first-order logic (see Section 2.5) to our application in Section 4.1. More precisely, in our setting, the logic consists of ordinary first-order formulas, with equality as the only predicate symbol. This allows us, in particular, to express the axioms of preadditive semicategories. We can then specify universal truth of operator statements via the logical notion of semantic consequences of these axioms.

Gödel's famous completeness theorem [Göd30] for first-order logic states that semantic consequences can be validated by formal computations. A formal computation is a purely syntactic procedure performed on the formulas using a deductive system, like the sequent calculus presented in Section 2.5.3. Such deductive systems have to treat linearity axioms separately and ensure that all formed expressions respect the restrictions imposed by the sorts.

In Section 4.3, we explain how a formal computation can be replaced by a computation with noncommutative polynomials. In contrast to classical deductive systems, these polynomial computations naturally include linearity and, as will be shown, restrictions induced by

sorts can be neglected. When translating formulas into polynomial statements, quantifiers and function symbols have to be treated separately. To this end, we first recall two important concepts from the field of first-order logic and automated theorem proving, namely Herbrand's theorem [Her30] and Ackermann's reduction [Ack54] in Section 4.2. For modern textbook expositions of these concepts in the unsorted case, we refer to [EFT21, Sec. XI.1] and [KS16, Sec. 3.3.1] respectively. These procedures allow to reduce arbitrary formulas to *arithmetic operator statements*, containing no quantifiers and only arithmetic function symbols.

To characterise validity of arithmetic operator statements, we introduce the process of *idealisation* (Section 4.3), which assigns to every such formula a predicate about polynomial ideal membership. We then show in Section 4.4 that this idealisation is true if and only if the corresponding formula is a semantic consequence of the axioms of preadditive semicategories. In this way, universal truth of operator statements is reduced to ideal membership of noncommutative polynomials.

Recall that ideal membership of noncommutative polynomials in the free algebra is only semi-decidable. More precisely, verifying ideal membership in this setting is always possible using, for example, Gröbner bases, but disproving it is not possible in general. Since also first-order logic is only semi-decidable, we cannot hope to obtain a terminating algorithm for deciding validity of first-order operator statements. Nevertheless, our results allow to present a semi-decision procedure (Procedure 8) that terminates if and only if an operator statement is universally true. Our procedure is an adaptation of Gilmore's algorithm [Gil60] interleaved with ideal membership verifications.

In Section 4.5.1, we discuss computational aspects that are relevant for treating large, nontrivial examples, and in Section 4.6, we illustrate our methods on an introductory example about the existence of Moore-Penrose inverses in categories with involution.

4.1 Modelling operator statements using many-sorted logic

In this section, we adapt the general concept of many-sorted logic discussed in Section 2.5 to our setting of linear operators. In particular, we use variables and constant symbols to represent linear operators, and we exploit the concept of sorts to represent the domains and codomains of these operators. To this end, we fix an enumerable set $\mathbf{Ob} = \{v_1, v_2, \dots\}$ of

object symbols, and we consider the pairs $(u, v) \in \mathbf{Ob} \times \mathbf{Ob}$ as sorts, that is, we specialise $\mathbf{Sort} = \mathbf{Ob} \times \mathbf{Ob}$. Furthermore, for each sort (u, v), we identify a distinguished *zero constant symbol* $0_{u,v} \in \mathbf{Con}$. We collect them in the set $\mathbf{Zero} = \{0_{u,v} \mid u, v \in \mathbf{Ob}\}$. Similarly, we also distinguish certain special function symbols and consider the set of *arithmetic function symbols* $\mathbf{Arith} = \bigcup_{u,v,w \in \mathbf{Ob}} \{-_{u,v}, +_{u,v}, \cdot_{u,v,w}\} \subseteq \mathbf{Fun}$.

Recall that a signature lists and describes the non-logical symbols of a logic that are relevant in a particular context. For our setting, we consider signatures of the form $\Sigma = (O \times O, C, F, \sigma)$, where

1. $O \subseteq \mathbf{Ob}$ is a set of object symbols;
2. $C \subseteq \mathbf{Con}$ such that $0_{u,v} \in C$ for all $u, v \in O$;
3. $F \subseteq \mathbf{Fun}$ such that $-_{u,v}, +_{u,v}, \cdot_{u,v,w} \in F$ for all $u, v, w \in O$;
4. the sort function σ satisfies the following conditions:
   a) $\sigma(0_{u,v}) = (u, v)$ for all $0_{u,v} \in C$;
   b) $\sigma(-_{u,v}) = (u, v) \to (u, v)$ for all $-_{u,v} \in F$;
   c) $\sigma(+_{u,v}) = (u, v) \times (u, v) \to (u, v)$ for all $+_{u,v} \in F$;
   d) $\sigma(\cdot_{u,v,w}) = (v, w) \times (u, v) \to (u, w)$ for all $\cdot_{u,v,w} \in F$;

We give some remarks on how signatures now specialise compared to the general Definition 2.5.1.

- As also noted before Definition 2.5.1, in our setting, signatures do not contain any predicate symbols. The only predicate that we will work with is equality $\approx$, which will be interpreted as identity.
- The additional conditions on the sets C and F of relevant constant and function symbols ensure that they contain at least the zero constants and the arithmetic function symbols.

- The additional conditions on the sort function σ ensure that the dedicated zero constants and the arithmetic function symbols have the sorts one would expect, mirroring the operations in preadditive semicategories (see Definition 2.1.1). Note that $-_{u,v}$ is a unary function symbol that represents additive inversion. We used the notation from Remark 2.5.2 to represent these sorts.

In the following, we simply write $\Sigma = (O, C, F, \sigma)$ for a signature $\Sigma = (O \times O, C, F, \sigma)$ satisfying the conditions above. Using this special kind of signature, we can define *operator statements*. To this end, we fix a signature $\Sigma = (O, C, F, \sigma)$ for the rest of this section.

Definition 4.1.1. *Any formula* $\varphi \in \mathbf{Form}(\Sigma)$ *is called an* operator statement.

As a special class of operator statements, we can formulate the *axioms of preadditive semicategories*. We denote the set of these axioms by $\mathcal{A}$ and to define it we introduce the auxiliary sets $\mathcal{A}_{t,u,v,w}$, with $t, u, v, w \in O$, containing the sentences listed below. To make the sorts of the variables used better visible, we write $x^{(u,v)}$ in the prefix if x is a variable of sort (u, v). Furthermore, to simplify the notation, we use the same symbols $-$, $+$, and $\cdot$ for all arithmetic function symbols $-_{u,v}$, $+_{u,v}$, and $\cdot_{u,v,w}$. We also use the usual infix notation for $+$ and $\cdot$, and we assume the natural precedence of these operations.

$$\mathcal{A}_{t,u,v,w} = \left\{ \begin{array}{lcl} \forall x^{(v,w)}, y^{(u,v)}, z^{(t,u)} & : & x \cdot (y \cdot z) \approx (x \cdot y) \cdot z, \\ \forall x^{(u,v)}, y^{(u,v)}, z^{(u,v)} & : & x + (y + z) \approx (x + y) + z, \\ \forall x^{(u,v)} & : & x + 0_{u,v} \approx x, \\ \forall x^{(u,v)} & : & x + (-x) \approx 0_{u,v}, \\ \forall x^{(u,v)}, y^{(u,v)} & : & x + y \approx y + x, \\ \forall x^{(v,w)}, y^{(u,v)}, z^{(u,v)} & : & x \cdot (y + z) \approx x \cdot y + x \cdot z, \\ \forall x^{(v,w)}, y^{(v,w)}, z^{(u,v)} & : & (x + y) \cdot z \approx x \cdot z + y \cdot z \end{array} \right\}$$

Note that the sorts of the used arithmetic function symbols are determined implicitly by the sorts of the variables. For example, adding the sorts of the function symbols explicitly to the last axiom listed above yields

$$\forall x^{(v,w)}, y^{(v,w)}, z^{(u,v)} \ : \ (x +_{v,w} y) \cdot_{u,v,w} z \ \approx \ x \cdot_{u,v,w} z +_{u,w} y \cdot_{u,v,w} z.$$

Definition 4.1.2. *Let $\Sigma = (O, C, F, \sigma)$ be a signature and $\mathcal{A}_{t,u,v,w}$ be as above. The set $\mathcal{A}$ of* axioms of preadditive semicategories in signature Σ *is given by*

$$\mathcal{A} \coloneqq \bigcup_{t,u,v,w \in O} \mathcal{A}_{t,u,v,w}.$$

Note that $\mathcal{A}$ is finite if and only if O is. If Σ is clear from the context, we refer to $\mathcal{A}$ simply as the set of axioms of preadditive semicategories and omit the dependency on Σ.

We make some remarks about the nature of models $\mathcal{I} = (A, \mathfrak{a})$ of the axioms $\mathcal{A}$ in signature $\Sigma = (O, C, F, \sigma)$. The structure $A = (A_{u,v})_{u,v \in O}$ can be considered as a preadditive semicategory $\mathcal{C}$ with objects $\mathrm{Ob}(\mathcal{C}) = O$ and morphism sets $\mathrm{Mor}(u, v) = A_{u,v}$. The arithmetic operations in $\mathcal{C}$, that is, composition of morphisms, addition, and additive inversion, are given by the interpretations of the arithmetic function symbols. Furthermore, the axioms $\mathcal{A}$ ensure that each set $A_{u,v}$ is indeed an abelian group with neutral element given by the interpretation of the constant symbol $0_{u,v}$ and that the composition of morphisms is bilinear.

We are interested in the first-order theory axiomatised by the set $\mathcal{A}$, that is, all statements that follow from the axioms of preadditive semicategories. Since the axioms $\mathcal{A}$ only prescribe basic linearity assumptions, we can consider any such statement as a universally true operator statement. Formally, we arrive at the following definition.

Definition 4.1.3. *An operator statement $\varphi \in \mathbf{Form}(\Sigma)$ is* universally true *if $\mathcal{A} \models \varphi$.*

Based on Theorem 2.5.14 and the sequent calculus $\mathrm{LK}^{=}$ presented in Section 2.5.3, the universal truth of an operator statement φ can be proven by a formal computation deriving $\mathcal{A} \vdash \varphi$. In the following sections, we describe how such a formal computation can be replaced by a computation with noncommutative polynomials. Arithmetic function symbols have a natural translation into the polynomial setting. To be able to also handle other function symbols and quantifiers, we recall two important concepts in the following, namely Herbrand's theorem [Her30] and Ackermann's reduction [Ack54].

4.2 Tools from first-order logic

4.2.1 Herbrand's theorem

We state a simplified version of the original theorem which asserts that unsatisfiability of a certain class of formulas can be reduced to (propositional) unsatisfiability of a ground sentence. As a consequence, we see that certain semantic consequences can be verified by a formal computation involving only ground instances of the original formulas. This allows to systematically eliminate quantifiers. For a modern textbook exposition of Herbrand's theorem in the unsorted case, see, for example, [EFT21, Sec. XI.1].

Before we proceed to state Herbrand's theorem, we first recall some normal forms of formulas as well as the notion of *Herbrand expansion*. In the following, we fix a signature $\Sigma = (O, C, F, \sigma)$.

A sentence of the form $\varphi = Q_1 x_1 \dots Q_n x_n : \varphi^*$ with φ^* quantifier-free and $Q_i \in \{\exists, \forall\}$ is in *prenex normal form* with *prefix* $Q_1 x_1 \dots Q_n x_n$. If the prefix consists only of existential (universal) quantifiers, that is, $Q_i = \exists$ ($Q_i = \forall$) for all i, then φ is in *Herbrand normal form* (*Skolem normal form*).

We note that any formula φ can be transformed into prenex normal form by applying the following rewrite rules to φ until a normal form is reached:

$$\begin{aligned}
\neg Qx : \psi &\quad\rightsquigarrow\quad \bar{Q}x : \neg\psi \\
(Qx : \psi_1) * \psi_2 &\quad\rightsquigarrow\quad Qy : (\psi_1[x \mapsto y] * \psi_2) \text{ for } * \in \{\wedge, \vee\} \\
(Qx : \psi_1) \to \psi_2 &\quad\rightsquigarrow\quad \bar{Q}y : (\psi_1[x \mapsto y] \to \psi_2) \\
\psi_1 * (Qx : \psi_2) &\quad\rightsquigarrow\quad Qy : (\psi_1 * \psi_2[x \mapsto y]) \text{ for } * \in \{\wedge, \vee, \to\}
\end{aligned}$$

Here, y is a new variable with $\sigma(y) = \sigma(x)$ and $\bar{Q}$ denotes the quantifier dual to Q, that is, $\bar{\exists} = \forall$ and $\bar{\forall} = \exists$. A formula is logically equivalent to the prenex normal form produced by these rewrite rules.

The *Herbrand expansion* $H(\varphi)$ of a sentence $\varphi = Q_1 x_1 \dots Q_n x_n : \varphi^*$ in prenex normal form is the set of all ground instances of φ, that is, if $n = 0$, then $H(\varphi) \coloneqq \{\varphi\}$, and otherwise

$$H(\varphi) \coloneqq \{\varphi^*[x_1 \mapsto t_1, \dots, x_n \mapsto t_n] \mid t_i \in \mathbf{Ground}(\Sigma), \sigma(t_i) = \sigma(x_i)\}\,.$$

Remark 4.2.1. *The Herbrand expansion depends on the signature Σ, and in particular, on the sort function σ. Different signatures and sort functions lead to different expansions.*

Note that $H(\varphi)$ is either a singleton (in case φ is ground), or infinite. This follows from the presence of the arithmetic function symbols and the zero constant symbols in all our signatures, which can be nested arbitrarily deep. All formulas in $H(\varphi)$ are ground, meaning that they do not contain any variables. We extend the Herbrand expansion also to sets of sentences $\Phi \subseteq \mathbf{Sent}(\Sigma)$ in prenex normal form, by setting $H(\Phi) \coloneqq \bigcup_{\varphi \in \Phi} H(\varphi)$. For any countable set Φ, also $H(\Phi)$ is countable. This follows from the fact that $\mathbf{Ground}(\Sigma)$ is countable.

Theorem 4.2.2 (Herbrand's theorem)**.** *Let $\varphi \in \mathbf{Sent}(\Sigma)$ be in Skolem normal form. Then φ is unsatisfiable if and only if there exist $\varphi_1, \ldots, \varphi_l \in H(\varphi)$ such that $\varphi_1 \wedge \cdots \wedge \varphi_l$ is unsatisfiable.*

Proof. Follows from [Nel+10, Thm. 30], where the statement is proven for order-sorted logic. Classical (unordered) many-sorted logic can be considered as a special case of order-sorted logic. □

Below we state a corollary of Theorem 4.2.2, providing a form of Herbrand's theorem tailored towards our application. This form is a typical formulation of Herbrand's theorem in ordinary first-order logic, see, for example, [EFT21, Thm. XI.1.4], but we were not able to find it for many-sorted logic in the literature. Although the proof carries over in a straightforward way from the unsorted case, we still include it here for completeness.

Corollary 4.2.3. *Let $\Phi \subseteq \mathbf{Sent}(\Sigma)$ be a set of sentences in Skolem normal form and let $\varphi \in \mathbf{Sent}(\Sigma)$ be in Herbrand normal form. The following are equivalent:*

1. *$\Phi \models \varphi$;*
2. *there exist $\psi_1, \ldots, \psi_k \in H(\Phi)$, $\varphi_1, \ldots, \varphi_l \in H(\varphi)$ such that $\psi_1 \wedge \cdots \wedge \psi_k \vdash \varphi_1 \vee \cdots \vee \varphi_l$;*

Proof. By Proposition 2.5.12, $\Phi \models \varphi$ is equivalent to $\pi_1 \wedge \cdots \wedge \pi_r \wedge \neg\varphi$ being unsatisfiable for some $\pi_1, \ldots, \pi_r \in \Phi$. By assumption, each π_i is of the form $\pi_i = \forall \boldsymbol{x}_i : \pi_i^*$ where $\boldsymbol{x}_i$ is a sequence of variables and π_i^* is quantifier-free. Analogously, φ can be written as

$\varphi = \exists \boldsymbol{y} : \varphi^*$ where $\boldsymbol{y}$ is a sequence of variables and φ^* is quantifier-free. Thus, a prenex normal form of $\pi_1 \wedge \cdots \wedge \pi_r \wedge \neg\varphi$ is given by

$$\alpha \coloneqq \forall \boldsymbol{z} : \bigwedge_{i=1}^{r} \pi_i^* \wedge \neg\varphi^*,$$

for a certain sequence of variables $\boldsymbol{z}$. Note that α is in Skolem normal form. Consequently, by Theorem 4.2.2, α being unsatisfiable is equivalent to the existence of ground instances $\alpha_1, \ldots, \alpha_l \in H(\alpha)$ such that $\alpha_1 \wedge \cdots \wedge \alpha_l$ is unsatisfiable. We note that each α_j can be written as $\alpha_j = \bigwedge_{i=1}^{r} \psi_{i,j} \wedge \neg\varphi_j$ with $\psi_{i,j} \in H(\pi_i) \subseteq H(\Phi)$ and $\varphi_j \in H(\varphi)$. So, condition 1 is equivalent to $\bigwedge_{j=1}^{l} \bigwedge_{i=1}^{r} \psi_{i,j} \wedge \bigwedge_{j=1}^{l} \neg\varphi_j$ being unsatisfiable. By Proposition 2.5.12 and Theorem 2.5.14, the latter is equivalent to condition 2. □

The axioms $\mathcal{A}$ of preadditive semicategories form a set of sentences in Skolem normal form. Hence, Corollary 4.2.3 can be used to reduce semantic consequences $\mathcal{A} \models \varphi$ to syntactic consequences of ground sentences. However, it might seem that the version of Herbrand's theorem stated above has limited applicability since it requires φ to be in Herbrand normal form. Fortunately, for any set $\Phi \subseteq \mathbf{Sent}(\Sigma)$ of axioms, every formula can be transformed into a Φ-equivalid sentence in Herbrand normal form.

Starting from an arbitrary formula φ, first any variables not in the scope of a quantifier are bound by a universal quantifier, that is, if $\boldsymbol{x} = x_1, \ldots, x_n$ are not in the scope of a quantifier, then φ is transformed into the sentence $\forall \boldsymbol{x} : \varphi$. The latter formula is called the *universal closure* of φ. By the semantics of the universal quantifier, φ and $\forall \boldsymbol{x} : \varphi$ are Φ-equivalid.

Next, $\forall \boldsymbol{x} : \varphi$ is transformed into prenex normal form. After this process, the resulting formula is almost in Herbrand normal form. However, the prefix of a formula in prenex normal form can still contain universal quantifiers. The goal of *Herbrandisation* is to remove these universal quantifiers in such a way that validity is preserved. More precisely, given a formula φ in prenex normal form, applying the following rules exhaustively is called *Herbrandisation*.

$$\begin{aligned} \forall y : \psi \quad &\rightsquigarrow \quad \psi[y \mapsto c] \\ \exists x_1, \ldots, x_n \forall y : \psi \quad &\rightsquigarrow \quad \exists x_1, \ldots, x_n : \psi[y \mapsto f(x_1, \ldots, x_n)] \end{aligned}$$

In these rules, c is a new constant symbol with sort $\sigma(c) = \sigma(y)$ and f is a new function symbol with sort $\sigma(f) = \sigma(x_1) \times \cdots \times \sigma(x_n) \to \sigma(y)$. Note that, whenever a Herbrandisation rule is applied, the signature $\Sigma = (O, C, F, \sigma)$ has to be extended, yielding a new signature. In case of applying the first rule, c has to be added to C, and for the second rule, f has to be added to F. In both cases, σ has to be extended as well.

It is clear that Herbrandisation transforms well-formed formulas φ into well-formed formulas in Herbrand normal form. Additionally, the result of this transformation is also Φ-equivalid to φ. We refer to [Wal87, Ch. 11], where the dual concept of *Skolemisation* is discussed, for further details.

Combining the steps described above, we obtain the following result. We omit its proof as it is completely dual to the one presented in [Wal87, Ch. 11] for Skolemisation.

Proposition 4.2.4. *Let $\Phi \subseteq \mathbf{Sent}(\Sigma)$ be a set of sentences. Every formula $\varphi \in \mathbf{Form}(\Sigma)$ can be transformed into a Φ-equivalid sentence in Herbrand normal form.*

4.2.2 Ackermann's reduction

When translating formulas into polynomial statements, it is natural to interpret the arithmetic function symbols as the arithmetic operations on polynomials. However, other function symbols – such as those introduced by Herbrandisation – cannot be translated directly. To treat such function symbols, we utilise a technique from the theory of uninterpreted functions with equality called *Ackermann's reduction*.

The goal of Ackermann's reduction is to remove function symbols from a formula in such a way that validity is preserved. The reduction requires replacing function symbols by new constant symbols and adding a set of constraints to enforce *functional consistency*, which encodes that two function instantiations are equal if instantiated with equal arguments.

Ackermann's reduction is described formally in Algorithm 7. We present this procedure fairly detailed since we could not find a suitable reference for the setting of many-sorted first-order logic. For a discussion in the unsorted case, we refer to [KS16, Sec. 3.3.1]. Although the proofs carry over in a straightforward way to our setting, we decided to still add them here for the convenience of the reader.

Algorithm 7 is presented for the elimination of a single function symbol. By repeated application, all unwanted function symbols can be removed from a given formula. Note that this reduction introduces new constants, which – analogous to the Herbrandisation – requires to extend the signature.

Algorithm 7: Ackermann's reduction

Input: Quantifier-free formula φ with m different instances of function symbol f of sort $\sigma(f) = (u_1, v_1) \times \cdots \times (u_n, v_n) \to (u, v)$

Output: Quantifier-free formula φ^{Ack} without instances of f s.t. φ is valid iff φ^{Ack} is

1 Assign indices to the m instances of f. Denote by f_i the instance of f with index i and by $t_1^{(i)}, \ldots, t_n^{(i)}$ the arguments of f_i.

2 Let $\mathcal{T}$ be the function that maps every instance $f_i(t_1^{(i)}, \ldots, t_n^{(i)})$ to a new constant symbol c_i with sort $\sigma(c_i) = (u, v)$ and that leaves all other terms unchanged. In case of nested appearances of f, the function $\mathcal{T}$ is only applied to the outermost instance.

3 $\varphi^{\mathrm{flat}} \leftarrow \mathcal{T}(\varphi)$;

4 Let φ^{FC} denote the following formula

$$\bigwedge_{i=1}^{m-1} \bigwedge_{j=i+1}^{m} \Big(\mathcal{T}(t_1^{(i)}) \approx \mathcal{T}(t_1^{(j)}) \wedge \cdots \wedge \mathcal{T}(t_n^{(i)}) \approx \mathcal{T}(t_n^{(j)})\Big) \to c_i \approx c_j;$$

5 $\varphi^{\mathrm{Ack}} \leftarrow \Big(\varphi^{\mathrm{FC}} \to \varphi^{\mathrm{flat}}\Big)$;

6 **return** φ^{Ack};

Proposition 4.2.5. *Algorithm 7 is correct.*

Proof. The resulting formula is clearly a quantifier-free formula without instances of f. It remains to show that φ is valid if and only if φ^{Ack} is, but this follows from the more general Proposition 4.2.7 below by setting $\Phi = \emptyset$. □

Example 4.2.6. *Consider the formula*

$$\varphi = (x \not\approx y) \wedge f(x) \approx f(z) \wedge (x \not\approx f(x) \vee f(z) \not\approx f(f(y))).$$

with variables x, y, z of sort (u, v) for some $u, v \in \mathbf{Ob}$ and function symbol f with $\sigma(f) = (u, v) \to (u, v)$. We apply Algorithm 7 to remove f from φ.

After assigning indices to the instances of f (say, from left to right and outside-in), we compute φ^{flat} and φ^{FC} accordingly:

$$\begin{aligned}\varphi^{flat} &= (x \not\approx y) \wedge c_1 \approx c_2 \wedge (x \not\approx c_1 \vee c_2 \not\approx c_3),\\ \varphi^{FC} &= (x \approx z \to c_1 \approx c_2) \ \wedge \ (x \approx c_4 \to c_1 \approx c_3)\\ &\quad \wedge (x \approx y \to c_1 \approx c_4) \ \wedge \ (z \approx c_4 \to c_2 \approx c_3)\\ &\quad \wedge (z \approx y \to c_2 \approx c_4) \ \wedge \ (c_4 \approx y \to c_3 \approx c_4),\end{aligned}$$

where c_1, c_2, c_3, c_4 are new constant symbols of sort (u, v) replacing $f(x), f(z), f(f(y)), f(y)$ respectively. Then φ is valid if and only if $\varphi^{Ack} = \varphi^{FC} \to \varphi^{flat}$ is, and in fact, one can show that φ^{Ack} is not valid, implying that neither is φ.

In our application, we are interested in semantic consequences $\Phi \models \varphi$. The following result asserts that Ackermann's reduction also preserves this property for quantifier-free formulas φ if the removed function symbol does not appear in Φ.

Proposition 4.2.7. *Let $\varphi \in \mathbf{Form}(\Sigma)$ be a quantifier-free formula containing a function symbol f. Let $\Phi \subseteq \mathbf{Form}(\Sigma)$ be a set of formulas not containing f. Then φ is Φ-equivalid to φ^{Ack} when f is removed.*

Proof. We prove $\Phi \not\models \varphi$ if and only if $\Phi \not\models \varphi^{\mathrm{Ack}}$. To this end, we consider all interpretations within the signature that contains both f and the new constant symbols c_i.

For one implication, assume that $\Phi \not\models \varphi$. Let $\mathcal{I} = (A, \mathfrak{a})$ be a model of Φ such that $\mathcal{I}(\varphi) = \bot$. Consider the interpretation $\mathcal{I}' = (A', \mathfrak{a})$ where A' is obtained from A by setting $c_i^A = \mathcal{I}(f_i(t_1^{(i)}, \ldots, t_n^{(i)}))$, for all i, and leaving everything else unchanged. Since c_i^A are newly introduced constant symbols, they do not appear in Φ. As the interpretations of all other terms remain unchanged, $\mathcal{I}$ and $\mathcal{I}'$ agree on all terms in Φ, and therefore, $\mathcal{I}'(\Phi) = \mathcal{I}(\Phi) = \top$. Furthermore, $\mathcal{I}'(\varphi^{\mathrm{flat}}) = \mathcal{I}(\varphi) = \bot$ and $\mathcal{I}'(\varphi^{\mathrm{FC}}) = \top$. Consequently $\mathcal{I}(\varphi^{\mathrm{Ack}}) = \bot$, showing that φ^{Ack} is not a semantic consequence of Φ.

For the other implication, assume that $\Phi \not\models \varphi^{\mathrm{Ack}}$. Let $\mathcal{I}' = (A', \mathfrak{a}')$ be a model of Φ such that $\mathcal{I}'(\varphi^{\mathrm{Ack}}) = \bot$. Then, in particular, $\mathcal{I}'(\varphi^{\mathrm{FC}}) = \top$ and $\mathcal{I}'(\varphi^{\mathrm{flat}}) = \bot$. Now, consider the interpretation $\mathcal{I} = (A, \mathfrak{a}')$ where A is obtained from A' by replacing the function $f^{A'}$ by the function f^A that agrees with $f^{A'}$ except for $f^A(\mathcal{I}'(\tau_1^{(i)}), \ldots, \mathcal{I}'(\tau_n^{(i)})) = \mathcal{I}'(c_i)$, for

all i, where $\tau_k^{(i)} = \mathcal{T}(t_k^{(i)})$. Note that this function is well-defined since $\mathcal{I}'$ is a model of φ^{FC}. Since the symbol f does not appear in Φ, the interpretations of all terms in Φ remain unchanged, and thus, $\mathcal{I}(\Phi) = \mathcal{I}'(\Phi) = \top$. Furthermore, $\mathcal{I}(\varphi) = \mathcal{I}'(\varphi^{\mathrm{flat}}) = \bot$, and consequently $\Phi \not\models \varphi$. □

For the special case of the axioms $\mathcal{A}$ of preadditive semicategories, we obtain the following corollary of Proposition 4.2.7.

Corollary 4.2.8. *A quantifier-free operator statement φ is universally true if and only if φ^{Ack} is, when a non-arithmetic function symbol is removed from φ.*

4.3 Idealisation

Idealisation is a process that allows to characterise universal truth of certain operator statements via ideal membership in the free algebra $\mathbb{Z}\langle X\rangle$. To define this concept, we first specify a certain class of operator statements, so-called *arithmetic operator statements*.

Definition 4.3.1. *Let Σ be a signature. An operator statement $\varphi \in \mathbf{Form}(\Sigma)$ is called* arithmetic *if it is quantifier-free and all function symbols appearing in φ are arithmetic function symbols.*

Example 4.3.2. *Consider a signature $\Sigma = (O, C, F, \sigma)$ with $O = \{u, v\}$ and non-arithmetic function symbols $f, g \in F$ such that*

$$\sigma(f) = (u, v) \to (u, v) \qquad \sigma(g) = (v, u) \times (u, v) \to (u, u).$$

Furthermore, let $x, y \in \mathbf{Var}$ with $\sigma(x) = (u, v)$ and $\sigma(y) = (v, u)$.

The quantifier-free operator statement $x \cdot y \cdot x + f(x) \approx x \vee y \cdot x \approx g(y, x)$ is not arithmetic as it contains the function symbols f and g. An example of an arithmetic operator statement would be $x \cdot y \cdot x \approx x \wedge y \cdot x \cdot y \not\approx y$.

All terms appearing in an arithmetic operator statement are essentially polynomial expressions in some basic symbols (the variables and constant symbols). Thus, it is straightforward to translate terms appearing in such formulas into noncommutative polynomials, simply by translating variables and constant symbols into indeterminates and the arithmetic function symbols into the arithmetic operations in $\mathbb{Z}\langle X\rangle$. Note that, since any formula is a finite string of symbols, it is clear that the polynomials obtained in this way consist of finitely many indeterminates. We formally describe how this is done in the following section.

4.3.1 From terms to polynomials

Fix a signature $\Sigma = (O, C, F, \sigma)$ and let $\varphi \in \mathbf{Form}(\Sigma)$ be an arithmetic operator statement. We associate to every variable and nonzero constant symbol appearing in φ an indeterminate, that is, we consider the set

$$X_\varphi = \{x \in \mathbf{Var} \cup (C \setminus \mathbf{Zero}) \mid x \text{ appears in } \varphi\}$$

of indeterminates. Note that we do not assign indeterminates to the distinguished zero constants $0_{u,v}$. With this, we define a translation function T_φ, that maps every term in φ to a polynomial in $\mathbb{Z}\langle X_\varphi\rangle$, as follows:

$$\begin{aligned}
T_\varphi(0_{u,v}) &= 0 \\
T_\varphi(x) &= x \quad \text{if } x \in \mathbf{Var} \cup \mathbf{Con} \\
T_\varphi(-t) &= -T_\varphi(t) \\
T_\varphi(s+t) &= T_\varphi(s) + T_\varphi(t) \\
T_\varphi(s \cdot t) &= T_\varphi(s) \cdot T_\varphi(t)
\end{aligned}$$

Remark 4.3.3. *While the zero constant symbols $0_{u,v}$ are translated into the additive identity $0 \in \mathbb{Z}\langle X_\varphi\rangle$, constant symbols representing identity morphisms cannot be translated into the multiplicative identity 1 in the free algebra as this would constitute a many-to-one mapping and a loss of information. We note that this is not an issue when mapping all zero operators to the zero polynomial, as the zero polynomial does not affect any polynomial computations. Instead, identity operators have to be treated like any other nonzero constant symbol.*

In order to simplify the notation, we identify a term t appearing in φ with its image $T_\varphi(t)$ under the translation map. Furthermore, if $s \approx t$ or $s \not\approx t$ is a literal in φ, we write $s - t$ for the polynomial $T_\varphi(s) - T_\varphi(t)$.

Example 4.3.4. *Consider a signature $\Sigma = (O, C, F, \sigma)$ with $O = \{u, v\}$ and σ such that*

$$\sigma(a) = (u, v), \qquad \sigma(b) = (v, u), \qquad \sigma(c) = (u, v), \qquad \sigma(d) = (v, u),$$

where $a, b, c, d \in \mathbf{Var}$.

Let $\varphi \in \mathbf{Form}(\Sigma)$ be the following arithmetic operator statement:

$$\varphi = (a \cdot b \cdot a + c \approx 0_{u,v} + a \wedge b \cdot a \cdot b \not\approx -b) \implies a \cdot b + c \cdot d \approx 0_{v,v}.$$

Using the indeterminates $X_\varphi = \{a, b, c, d\}$, the three literals appearing in φ are translated into the following polynomials $\mathbb{Z}\langle a, b, c, d\rangle$:

$$aba + c - a, \qquad bab + b, \qquad ab + cd.$$

4.3.2 From formulas to ideal membership

Next, we describe the translation of arithmetic operator statements into ideal theoretic statements. In what follows, we consider $\wedge$ and $\vee$ as associative and commutative operations, that is, $\varphi \wedge \psi = \psi \wedge \varphi$ and $\varphi \wedge (\psi \wedge \rho) = (\varphi \wedge \psi) \wedge \rho$, and analogously for $\vee$. We recall that every quantifier-free formula φ can be transformed into a logically equivalent formula of the form

$$\bigwedge_{i=1}^{m} \left(\bigvee_{j=1}^{n_i} s_{i,j} \not\approx t_{i,j} \vee \bigvee_{k=1}^{n_i'} p_{i,k} \approx q_{i,k} \right), \tag{4.1}$$

with terms $s_{i,j}, t_{i,j}, p_{i,k}, q_{i,k}$. In the above formula, either of the two disjunctions can also be empty, that is, it is possible that either $n_i = 0$ or $n_i' = 0$, but not both.

We recall that a formula of the form (4.1) is called in *conjunctive normal form (CNF)*. It is a conjunction of *clauses*, where a clause is a disjunction of literals. A formula can have several CNFs.

One way to obtain a CNF of a quantifier-free formula φ is to apply to φ exhaustively each of the following sets of rewrite rules [RN10, Sec. 7.5.2], in the given order:

1. Eliminate implications:
$$\psi_1 \to \psi_2 \;\rightsquigarrow\; \neg\psi_1 \vee \psi_2 \tag{4.2}$$

2. Move $\neg$ inwards (i.e., compute a negation normal form):
$$\neg\neg\psi \;\rightsquigarrow\; \psi \qquad \neg(\psi_1 \wedge \psi_2) \;\rightsquigarrow\; \neg\psi_1 \vee \neg\psi_2 \qquad \neg(\psi_1 \vee \psi_2) \;\rightsquigarrow\; \neg\psi_1 \wedge \neg\psi_2 \tag{4.3}$$

3. Distribute $\vee$ over $\wedge$:
$$\psi \vee (\psi_1 \wedge \psi_2) \;\rightsquigarrow\; (\psi \vee \psi_1) \wedge (\psi \vee \psi_2) \tag{4.4}$$

We note that the above rules apply modulo associativity and commutativity of $\wedge, \vee$. This process yields a unique result, which is a CNF of φ. We denote it by $\mathrm{CNF}(\varphi)$. Also note that this transformation preserves the semantics of φ.

Lemma 4.3.5. *Any quantifier-free formula φ is logically equivalent to* $\mathrm{CNF}(\varphi)$.

Based on the conjunctive normal form, we define the idealisation. We first discuss the special case of arithmetic clauses.

Definition 4.3.6. *Let $C = \bigvee_{j=1}^{n} s_j \not\approx t_j \vee \bigvee_{k=1}^{n'} p_k \approx q_k \in \mathbf{Form}(\Sigma)$ be an arithmetic clause. The* idealisation $\mathrm{I}(C)$ *of C is the following predicate considered as a statement in the free algebra $\mathbb{Z}\langle X_C\rangle$:*

$$\mathrm{I}(C) \;:\equiv\; \textit{there exists } 1 \le k \le n' \textit{ such that } p_k - q_k \in (s_1 - t_1, \ldots, s_n - t_n). \tag{4.5}$$

To motivate this definition, write C in the equivalent form $\bigwedge_j s_j \approx t_j \to \bigvee_k p_k \approx q_k$. This shows that C is a semantic consequence of $\mathcal{A}$ if and only if at least one $p_k \approx q_k$ can be derived from all $s_j \approx t_j$ and from the linearity axioms encoded in $\mathcal{A}$. This fact is described by $\mathrm{I}(C)$. Thus, this predicate being true is equivalent to C being universally true.

To extend this definition to arbitrary arithmetic operator statements φ, we consider the normal form $\mathrm{CNF}(\varphi)$ and use the fact that a conjunction of clauses is satisfied if and only if each clause is satisfied individually. Based on this, the idealisation of φ is given by conjunctively combining the idealisations of all clauses C in $\mathrm{CNF}(\varphi)$.

Definition 4.3.7. *Let $\varphi \in \mathbf{Form}(\Sigma)$ be an arithmetic operator statement. The* idealisation $\mathrm{I}(\varphi)$ *of φ is the predicate*

$$\mathrm{I}(\varphi) :\equiv \bigwedge_{\substack{C \text{ clause} \\ \text{of } \mathrm{CNF}(\varphi)}} \mathrm{I}(C).$$

In the succeeding section, we will prove that the idealisation of φ is true if and only if φ is universally true. To this end, we state the following lemma, which relates the semantics of the basic logical connectives $\wedge$ and $\vee$ to the notion of idealisation.

Lemma 4.3.8. *Let $\varphi, \psi \in \mathbf{Form}(\Sigma)$ be arithmetic operator statements. Then the following hold:*

1. $\mathrm{I}(\varphi) = \top$ *implies* $\mathrm{I}(\varphi \vee \psi) = \top$*;*
2. $\mathrm{I}(\varphi) = \top$ *and* $\mathrm{I}(\psi) = \top$ *implies* $\mathrm{I}(\varphi \wedge \psi) = \top$*;*

Proof. 1. Let $\mathrm{CNF}(\varphi) = \bigwedge_i C_i$ and $\mathrm{CNF}(\psi) = \bigwedge_j D_j$ with clauses C_i, D_j. Each clause of $\mathrm{CNF}(\varphi \vee \psi)$ is of the form $C_i \vee D_j$. By assumption $\mathrm{I}(C_i) = \top$ for all i. Then the statement follows from the fact that the ideal corresponding to $\mathrm{I}(C_i)$ is a subideal of the one corresponding to $\mathrm{I}(C_i \vee D_j)$ and the set of candidate members for $\mathrm{I}(C_i)$ is a subset of those for $\mathrm{I}(C_i \vee D_j)$.

2. By assumption, the idealisation of each clause of $\mathrm{CNF}(\varphi)$ and $\mathrm{CNF}(\psi)$ is true. Then the statement follows from the fact that each clause of $\mathrm{CNF}(\varphi \wedge \psi)$ is a clause from $\mathrm{CNF}(\varphi)$ or $\mathrm{CNF}(\psi)$. □

4.4 Characterising universal truth via ideal membership

In this section, we relate the semantic notion of universal truth of an arithmetic operator statement φ to the ideal theoretic statement $\mathrm{I}(\varphi) = \top$. The main result is the following theorem. Recall that we have fixed a signature $\Sigma = (O, C, F, \sigma)$.

Theorem 4.4.1. *An arithmetic operator statement $\varphi \in \mathbf{Form}(\Sigma)$ is universally true if and only if $\mathrm{I}(\varphi) = \top$.*

Proof. The "if"-part follows from Lemma 4.4.5 below and the "only if"-part from Lemma 4.4.8. □

Remark 4.4.2. *We make some remarks about Theorem 4.4.1.*

1. *Theorem 4.4.1 reduces universal truth to the verification of finitely many ideal memberships in $\mathbb{Z}\langle X_\varphi\rangle$. Since ideal membership in $\mathbb{Z}\langle X_\varphi\rangle$ is semi-decidable (Corollary 3.4.25) this immediately yields a semi-decision procedure for verifying universal truth of arithmetic operator statements. In Section 4.5, we generalise this procedure to all operator statements.*

2. *The idealisation $\mathrm{I}(\varphi)$ is independent of the signature Σ. The corresponding polynomial computations are the same for all signatures Σ in which φ can be formulated. In particular, they are independent of the sorts appearing in φ. Consequently, Theorem 4.4.1 tells us that a single computation with polynomials shows the universal truth of φ in* all *signatures in which φ can be formulated.*

3. *The idealisation $\mathrm{I}(\varphi)$ is defined with respect to a specific CNF of φ. Theorem 4.4.1 implies that $\mathrm{I}(\varphi)$ is in fact independent of the concrete CNF used. Thus, any CNF of φ can be used to compute $\mathrm{I}(\varphi)$. This can provide computational advantages.*

Theorem 4.4.1 is a generalisation of [RRH21, Thm. 32]. We state a corollary of this result below that will play a crucial role in proving the sufficiency part of Theorem 4.4.1. The language of [RRH21] differs from the language of this work in many points. Consequently, the following result, which we formulate in language of this work, looks very different to the original in [RRH21]. To facilitate the comparison between the two versions, we state the main differences in Remark 4.4.4.

Corollary 4.4.3. *Let $C = s_1 \not\approx t_1 \vee \cdots \vee s_n \not\approx t_n \vee p \approx q \in \mathbf{Form}(\Sigma)$ be an arithmetic clause. If $\mathrm{I}(C) = \top$, then C is universally true.*

Proof. Follows from [RRH21, Thm. 32]. We note that [RRH21, Thm. 32] is phrased for preadditive categories but by examining the proof it becomes clear that it also holds for preadditive semicategories. □

Remark 4.4.4.

1. *In this work, domains and codomains of operators are encoded in form of sorts via the sort function σ. In contrast to this, [RRH21] uses a directed multigraph, also known as* quiver*, for this. A quiver is a natural choice for representing operators between different spaces, however, it is unsuited for representing functions that use those operators as arguments. In the language of this work, quivers are good for representing the sorts of variables and constant symbols but not for representing the sorts of function symbols. We note that [RRH21] does not consider function symbols other than the arithmetic function symbols, whose sorts are predetermined anyway.*

2. *The syntactic objects of interest in [RRH21] are* compatible polynomials*. Being compatible means that the polynomials respect the restrictions imposed by the domains and codomains of the operators they model. In the terminology of this work, one can say that compatible polynomials are precisely those elements $s - t$ that originate from translating formulas $s \approx t$.*

3. *Another difference between [RRH21] and this work is how a semantic meaning is assigned to syntactic objects. Here we use the concept of interpretations for this, while [RRH21] uses* representations *of a quiver. It turns out that the latter can be considered as a special case of the former.*

4. *A closer adaptation of [RRH21, Thm. 32] would be given if the clause C in Corollary 4.4.3 was replaced by the logically equivalent formula $(s_1 \approx t_1 \wedge \cdots \wedge s_n \approx t_n) \to p \approx q$. We decided for the formulation above as it makes the subsequent proofs easier.*

The remainder of this section is dedicated to proving Theorem 4.4.1. More precisely, Lemma 4.4.5 proves the sufficiency part of Theorem 4.4.1 by reducing the case of arbitrary arithmetic operator statements to the case of clauses and using Corollary 4.4.3. For the

other implication, we use that fact that the sequent calculus $\mathrm{LK}^{=}$ is correct and complete, meaning that $\mathcal{A} \models \varphi$ if and only if $\mathcal{A} \vdash \varphi$. In particular, we consider the idealisation of certain sequents and show that all relevant sequent rules of $\mathrm{LK}^{=}$ preserve the property of these idealisations being true. This is done in Lemma 4.4.6. In order to handle the presence of the axioms $\mathcal{A}$ in sequents, we prove Lemma 4.4.7, which captures one of the main advantages of our approach – the axioms $\mathcal{A}$ can be neglected when performing idealisations. Finally, we combine Lemma 4.4.6 and 4.4.7 with Herbrand's theorem to prove the necessity part of Theorem 4.4.1 in Lemma 4.4.8.

Lemma 4.4.5. *Let $\varphi \in \mathbf{Form}(\Sigma)$ be an arithmetic operator statement. If $\mathrm{I}(\varphi) = \top$, then φ is universally true.*

Proof. Recall that φ being universally true means that $\mathcal{A} \models \varphi$.

First, we consider the case where φ is a clause $C = \bigvee_{j=1}^{n} s_j \not\approx t_j \vee \bigvee_{k=1}^{n'} p_k \approx q_k$. By definition of the idealisation, $\mathrm{I}(C) = \top$ if and only if there exists $1 \leq k \leq n'$ such that $p_k - q_k \in (s_1 - t_1, \ldots, s_n - t_n)$. Without loss of generality, assume that $k = 1$. Let $D = \bigvee_{j=1}^{n} s_j \not\approx t_j \vee p_1 \approx q_1$ and note that $\mathrm{I}(D) = \top$. So, by Corollary 4.4.3, $\mathcal{A} \models D$ but then also $\mathcal{A} \models D \vee \bigvee_{k=2}^{n'} p_k \approx q_k$, and the result follows.

For arbitrary φ, let $\mathrm{CNF}(\varphi) = \bigwedge_{i=1}^{m} C_i$ with clauses C_i. Then $\mathrm{I}(\varphi) = \top$ implies $\mathrm{I}(C_i) = \top$ for all $i = 1, \ldots, m$. By the previous discussion, $\mathcal{A} \models C_i$ for all $i = 1, \ldots, m$. Now, the semantics of $\wedge$ implies $\mathcal{A} \models \bigwedge_{i=1}^{m} C_i$, and the result follows from Lemma 4.3.5. □

Let $\Gamma, \Delta \subseteq \mathbf{Form}(\Sigma)$ be finite multisets of arithmetic operator statements. In the following, we define an idealisation of sequents $\Gamma \vdash \Delta$. More precisely, we denote

$$\mathrm{I}\left(\Gamma \vdash \Delta\right) \coloneqq \mathrm{I}\left(\bigvee_{\gamma \in \Gamma} \neg\gamma \vee \bigvee_{\delta \in \Delta} \delta\right).$$

We say that an axiom sequent rule $\dfrac{}{\Gamma \vdash \Delta}$ *preserves validity* if $\mathrm{I}(\Gamma \vdash \Delta) = \top$. Similarly, we say that a sequent rule of the form $\dfrac{\Gamma' \vdash \Delta'}{\Gamma \vdash \Delta}$ *preserves validity* if $\mathrm{I}(\Gamma' \vdash \Delta') = \top$ implies $\mathrm{I}(\Gamma \vdash \Delta) = \top$. Finally, a sequent rule of the form $\dfrac{\Gamma_1 \vdash \Delta_1 \qquad \Gamma_2 \vdash \Delta_2}{\Gamma \vdash \Delta}$ *preserves validity* if $\mathrm{I}(\Gamma_1 \vdash \Delta_1) = \top$ and $\mathrm{I}(\Gamma_2 \vdash \Delta_2) = \top$ implies $\mathrm{I}(\Gamma \vdash \Delta) = \top$.

Lemma 4.4.6. *All sequent rules of the sequent calculus* $\mathrm{LK}^{=}$ *(Section 2.5.3) not involving quantifiers preserve validity when applied to arithmetic operator statements.*

Proof. All sequent rules in $\mathrm{LK}^{=}$ contain multisets Γ and Δ which are comprised of all formulas that are irrelevant for the particular rule. To take care of these formulas, we denote in the following $\Phi = \bigvee_{\gamma\in\Gamma} \neg\gamma \vee \bigvee_{\delta\in\Delta} \delta$.

Axioms.

For (Ax), we have to show that $\mathrm{I}(\Phi \vee \varphi) = \top$ with $\varphi = s \not\approx t \vee s \approx t$. The idealisation of φ corresponds to the polynomial assertion $s - t \in (s - t)$, which is clearly true. Thus, $\mathrm{I}(\varphi) = \top$, and the result follows from the first part of Lemma 4.3.8.

For (Ref), we have to show that $\mathrm{I}(\Phi \vee t \approx t) = \top$. Note that $\mathrm{I}(t \approx t) = \top$ because it corresponds to the predicate $t - t = 0 \in (\emptyset) = \{0\}$. Then the result follows from the first part of Lemma 4.3.8.

Structural rules.

Preserving validity for the rules (W $\vdash$) and ($\vdash$ W) follows from the first part of Lemma 4.3.8. The other two rules (C $\vdash$) and ($\vdash$ C) only remove duplicates of formulas. This leaves the ideals and the sets of candidate members generated during the idealisation unchanged, and thus also these rules preserve validity.

Propositional rules.

For the $\neg$-rules, preserving validity translates into the trivial statements "$\mathrm{I}(\Phi \vee \varphi) = \top$ implies $\mathrm{I}(\Phi \vee \varphi) = \top$" and "$\mathrm{I}(\Phi \vee \neg\varphi) = \top$ implies $\mathrm{I}(\Phi \vee \neg\varphi) = \top$".

For ($\vee$ $\vdash$), we denote $\Psi = \Phi \vee \neg(\varphi \vee \psi)$, $\Phi_1 = \Phi \vee \neg\varphi$ and $\Phi_2 = \Phi \vee \neg\psi$. We have to show that $\mathrm{I}(\Psi) = \top$ assuming $\mathrm{I}(\Phi_1) = \top$ and $\mathrm{I}(\Phi_2) = \top$. Note that applying the rewrite rules (4.2) and (4.3) for the CNF computation to Ψ yields

$$\Phi' \vee (\varphi' \wedge \psi'), \tag{4.6}$$

where Φ' is a negation normal form of Φ and φ', ψ' are negation normal forms of $\neg\varphi$ and $\neg\psi$ respectively. Analogously, applying (4.2) and (4.3) to $\Phi_1 \wedge \Phi_2$ yields

$$(\Phi' \vee \varphi') \wedge (\Phi' \vee \psi'). \tag{4.7}$$

Applying now the distributivity rule (4.4) to (4.6) gives (4.7), showing that $\mathrm{CNF}(\Psi) = \mathrm{CNF}(\Phi_1 \wedge \Phi_2)$. Thus also $\mathrm{I}(\Psi) = \mathrm{I}(\Phi_1 \wedge \Phi_2)$, and the result follows from the second part of Lemma 4.3.8. For $(\vdash \vee)$, we obtain the trivial statement that $\mathrm{I}(\Phi \vee \varphi \vee \psi) = \top$ implies $\mathrm{I}(\Phi \vee \varphi \vee \psi) = \top$.

The proofs for $\wedge$ are dual to the proofs for $\vee$.

Using the fact that $\mathrm{CNF}(\varphi \to \psi) = \mathrm{CNF}(\neg\varphi \vee \psi)$ the proofs for $\to$ are analogous to the proofs for $\vee$.

Equational rule.

Denote $\Psi = \Phi \vee \neg\varphi[x \mapsto t] \vee t \not\approx t'$ and $\Psi' = \Phi \vee \neg\varphi[x \mapsto t'] \vee t \not\approx t'$. We have to show that $\mathrm{I}(\Psi') = \top$ implies $\mathrm{I}(\Psi) = \top$. Note that, for each clause C in $\mathrm{CNF}(\Psi)$, there is a corresponding clause C' in $\mathrm{CNF}(\Psi')$ which differs from C only by the fact that all occurrences of t that stem from x are replaced by t'. Now, fix an arbitrary clause C in $\mathrm{CNF}(\Psi)$ and the corresponding clause C' in $\mathrm{CNF}(\Psi')$. By definition of the idealisation, it suffices to show that $\mathrm{I}(C') = \top$ implies $\mathrm{I}(C) = \top$ to prove that $\mathrm{I}(\Psi') = \top$ implies $\mathrm{I}(\Psi) = \top$.

By definition of the CNF-transformation, both clauses C and C' contain the literal $t \not\approx t'$. Thus, if we denote the ideal obtained during the idealisation of C by J and that of C' by J', then, by definition of the idealisation, the polynomial $t - t'$ is a generator of both J and J'. Furthermore, since all formulas are assumed to be arithmetic operator statements, every literal $a \approx b$ or $a \not\approx b$ in C involving the term t gets translated into a polynomial $a(t) - b(t)$ depending on t. Analogously, every literal in C' involving the term t' gets translated into a polynomial $a(t') - b(t')$ depending on t'. Now, since $t - t' \in J, J'$, we have that, for any polynomial f, the element $f(t)$ lies in any of these ideals if and only if $f(t')$ does. Therefore, $a(t) - b(t) \in J$ if and only if $a(t') - b(t') \in J$, and analogously for J'. This shows that, in fact, $J = J'$, and implies that $\mathrm{I}(C) = \top$ if and only if $\mathrm{I}(C') = \top$. □

The following lemma captures a main advantage of our approach. It asserts that instantiations of the axioms $\mathcal{A}$ can be neglected when performing the idealisation.

Lemma 4.4.7. *Let $\varphi \in \mathbf{Form}(\Sigma)$ be an arithmetic operator statement. Then $\mathrm{I}(\neg\alpha \vee \varphi) = \mathrm{I}(\varphi)$ for all ground instances $\alpha \in H(\mathcal{A})$.*

Proof. Note that $\neg\alpha = s \not\approx t$ for some ground terms $s, t \in \mathbf{Ground}(\Sigma)$ and that

$$\mathrm{CNF}(\neg\alpha \vee \varphi) = \mathrm{CNF}(s \not\approx t \vee \varphi) = \bigwedge_i (s \not\approx t \vee C_i),$$

where $\mathrm{CNF}(\varphi) = \bigwedge_i C_i$. The ideal generated during the idealisation of each clause $s \not\approx t \vee C_i$ is the ideal obtained from C_i plus the additional generator $s - t$. The statement now follows from the fact that $s - t = 0$ since the formulas in $\mathcal{A}$ only describe basic linearity properties that hold in any ring. □

Lemma 4.4.8. *Let $\varphi \in \mathbf{Form}(\Sigma)$ be an arithmetic operator statement. If φ is universally true, then $\mathrm{I}(\varphi) = \top$.*

Proof. Recall that φ being universally true means that $\mathcal{A} \models \varphi$. Since φ is a quantifier-free, it is in Herbrand normal form. Using Herbrandisation and replacing all (free) variables in φ by new constant symbols, we can, without loss of generality, assume that φ is in fact a ground sentence. Note that this preserves universal truth. The axioms $\mathcal{A}$ form a set of sentences in Skolem normal form. Consequently, Corollary 4.2.3 yields the existence of $\alpha_1, \ldots, \alpha_k \in H(\mathcal{A})$ such that

$$\alpha_1 \wedge \cdots \wedge \alpha_k \vdash \varphi. \tag{4.8}$$

Here we used the fact that φ does not contain any variables, and therefore $H(\varphi) = \{\varphi\}$. Since all formulas in (4.8) are quantifier-free, there exists a formal proof in the sequent calculus $\mathrm{LK}^{=}$ that does not use any of the quantification rules. As, by Lemma 4.4.6, all other rules preserve validity, we have that $\top = \mathrm{I}(\alpha_1 \wedge \cdots \wedge \alpha_k \vdash \varphi) = \mathrm{I}(\neg\alpha_1 \vee \cdots \vee \neg\alpha_n \vee \varphi)$. Then repeated application of Lemma 4.4.7 yields the desired result. □

4.5 Semi-decision procedure

Using Herbrand's theorem and Ackermann's reduction (Section 4.2) we can easily reduce the case of verifying that an arbitrary operator statement φ is universally true to the previously discussed case of arithmetic operator statements. The following steps give an overview on how this can be done. They can be considered as an adaptation of Gilmore's algorithm [Gil60].

1. Bring φ into Herbrand normal form, denoted by φ^H.
2. Let $\varphi_1, \varphi_2, \ldots$ be an enumeration of $H(\varphi^H)$.
3. Let $n = 1$.
4. Form the formula $\psi_n = \bigvee_{i=1}^{n} \varphi_i$.
5. Remove all non-arithmetic function symbols from ψ_n using Ackermann's reduction. Denote the obtained formula by ψ_n^{Ack} (note that ψ_n^{Ack} is an arithmetic operator statement).
6. If $I(\psi_n^{\mathrm{Ack}}) = \top$, then φ is universally true. Otherwise, increase n by 1 and go to step 4.

Since first-order logic is only semi-decidable, we cannot expect to obtain an algorithm that terminates on any input. The best we can hope for is a semi-decision procedure that terminates if and only if a formula φ is indeed universally true. However, the steps above, as phrased now, still have a subtle flaw that stops them from even being a semi-decision procedure.

The conditional check in step 6 requires to decide certain ideal memberships. While *verifying* ideal membership of noncommutative polynomials is always possible in finite time, *disproving* it is generally not. Consequently, verifying that the condition in step 6 is false is generally not possible in finite time. In cases where this is required, the procedure cannot terminate – even if φ is indeed universally true.

To overcome this flaw and to obtain a true semi-decision procedure, we have to interleave the computations done for different values of n. Procedure 8 shows one way how this can be done. It essentially follows the steps described above, except that it only performs finitely many operations to check if $I(\psi_n^{\mathrm{Ack}}) = \top$ for each n.

Remark 4.5.1. *We make some remarks about Procedure 8.*

1. *The procedure treats the case where the Herbrand normal form of the input formula is a ground sentence separately. In this case, the computation can be simplified since the Herbrand expansion is a singleton. Then $\psi_n^{Ack} = \varphi_1^{Ack} = (\varphi^H)^{Ack}$ for all n since φ^H is the only element in $H(\varphi^H)$.*

Procedure 8: Semi-decision procedure for verifying universal truth

Input: a signature Σ and an operator statement $\varphi \in \mathbf{Form}(\Sigma)$

Output: $\top$ if and only if φ is universally true; otherwise infinite loop

```
1  φ^H ← Herbrand normal form of φ;
2  φ_1, φ_2, ... ← an enumeration of H(φ^H);
3  ψ_1 ← φ_1;
4  ψ_1^Ack ← Ackermann's reduction of ψ_1 removing all non-arith. function symbols;
5  for n ← 1, 2, ... :
6      for k ← 1, ..., n :
7          if I(ψ_k^Ack) = ⊤ can be verified with n operations of an ideal membership
             semi-decision procedure :
8              return ⊤;
9      if φ^H is a ground sentence :          // in this case H(φ^H) = {φ^H} = {φ_1}
10         ψ_{n+1}^Ack ← ψ_n^Ack;
11     else:
12         ψ_{n+1} ← ψ_n ∨ φ_{n+1};
13         ψ_{n+1}^Ack ← Ackermann's reduction of ψ_{n+1} removing all non-arith. function
             symbols;
```

2. *Line 7 contains the term* operation of a procedure. *Thereby we mean any (high- or low-level) set of instructions of the procedure that can be executed in* finite time.

3. *If φ is not universally true, then the procedure runs into an infinite loop. However, it can be modified to also terminate in some cases when φ is not universally true, see the discussion below the proof of Theorem 4.5.2.*

Theorem 4.5.2. *Procedure 8 terminates and outputs $\top$ if and only if φ is universally true.*

Proof. Recall that φ being universally true means that $\mathcal{A} \models \varphi$. Also, note that $\mathcal{A}$ is a set of sentences in Skolem normal form. Thus, by Proposition 4.2.4 and Corollary 4.2.3, $\mathcal{A} \models \varphi$ if and only if $\mathcal{A} \models \psi_N$ for some large enough $N \in \mathbb{N}$. Also, note that ψ_N is a quantifier-free operator statement. Hence, Corollary 4.2.8 yields that $\mathcal{A} \models \varphi$ is equivalent to $\mathcal{A} \models \psi_N^{\mathrm{Ack}}$. Finally, since ψ_N^{Ack} is an arithmetic operator statement, Theorem 4.4.1 implies that $\mathcal{A} \models \varphi$ is equivalent to $I(\psi_N^{\mathrm{Ack}}) = \top$. In other words, $\mathcal{A} \models \varphi$ is equivalent to the condition in line 7 being satisfied at some point. This shows that, if the algorithm terminates, then $\mathcal{A} \models \varphi$.

The other implication follows from Corollary 3.4.25 and the fact that the computations for each n can be executed in a finite amount of time. □

To end this section, we give a partial answer to the question "When can we prove that φ is *not* universally true?" or in other words "When does φ *not* follow from the axioms of preadditive semicategories?". We assume, without loss of generality, that φ is in Herbrand normal form. First, we note that if $H(\varphi)$ is infinite, then it is not possible to deduce that φ is not universally true using our approach as this would require to argue over all (infinitely many) finite subsets of $H(\varphi)$. If $H(\varphi)$ is not infinite, then $H(\varphi) = \{\varphi\}$. In this case, proving that φ is not universally true boils down to verifying that $I(\varphi^{\mathrm{Ack}}) = \bot$, which is the case if and only if certain ideal memberships do not hold. Although generally impossible, there are cases in which ideal membership can be disproven (for example, when the ideal is homogeneous or admits a finite Gröbner basis). In these situations, based on Theorem 4.4.1, we can prove that φ is not universally true.

4.5.1 Computational aspects

The performance of Procedure 8 strongly depends on the order in which the elements in the Herbrand expansion are enumerated. In the following, we present a few basic techniques to find the right instances needed for a verification that an operator statement φ is universally true that turned out to be useful in practice. Some of these techniques deviate quite strongly from how Procedure 8 is phrased, but they can have a drastic impact on the efficiency of the procedure.

In practical applications, the formula φ under consideration is typically an implication of the form $\varphi = \bigwedge_i \alpha_i \to \gamma$, where α_i represent different assumptions and γ forms a claimed property. In the following, we assume that φ is of this form.

Universal identities are sentences of the form $\forall \boldsymbol{x} : s(\boldsymbol{x}) \approx t(\boldsymbol{x})$ where s, t are terms depending on the variables $\boldsymbol{x} = x_1, \ldots, x_n$. Such formulas in the assumptions often describe very basic properties (such as linearity/multiplicativity of a function, commutativity of certain operators, etc.). Formally, each application of such an assumption in a proof requires a new instantiation of the formula φ. This increases the size of the resulting formula, making further computations more involved. To avoid this problem, we can remove the assumption $\forall \boldsymbol{x} : s(\boldsymbol{x}) \approx t(\boldsymbol{x})$ from φ and treat it implicitly by considering a universal rewrite rule

$s \mapsto t$ that maps every instantiation $s(t_1, \ldots, t_n)$ to $t(t_1, \ldots, t_n)$. This rewrite rule is then applied exhaustively to all formulas that occur during the computation.

Furthermore, when φ contains a unary non-arithmetic function symbol f, practical experience has shown that it can be beneficial to extend assumptions of the form

$$\bigwedge_i s_i \approx t_i \to \bigvee_j p_j \approx q_j$$

to

$$\bigwedge_i s_i \approx t_i \to \bigvee_j (p_j \approx q_j \wedge f(p_j) \approx f(q_j)),$$

where f is applied to all terms of the correct sort. In particular, plain identities $p \approx q$ can be extended to $p \approx q \wedge f(p) \approx f(q)$. This process is clearly sound and often helps to automatically generate the right function instantiations.

Next, we discuss the presence of existential identities, that is, sentences of the form $\exists \boldsymbol{x} : p(\boldsymbol{x}) \approx q(\boldsymbol{x})$, in the claim γ. Such formulas can describe, for instance, the existence of a solution to an equation and finding suitable instantiations for them is usually a hard task. Often, however, good candidates for solutions can be found automatically if enough "good" instantiations of the assumptions can be obtained somehow. In such cases, we can form the polynomial ideal $(s_1 - t_1, \ldots, s_m - t_m)$ consisting of all instantiated assumptions $s_i \approx t_i$ and search for an instantiation of $p - q$ in this ideal. In Section 5.3, we describe several techniques that allow to search for polynomials of a specific form in ideals.

Once suitable instantiations have been found, they are combined disjunctively into one formula and Ackermann's reduction is performed to remove non-arithmetic function symbols. When forming the idealisation of the resulting arithmetic operator statement φ, the formula has to be transformed into CNF. This transformation can lead to an exponential increase in size and to exponentially many ideal membership verifications during the idealisation. Classically in automated theorem proving, this exponential blow-up is avoided by a so-called Tseitin transformation, see, for example, [KS16, Sec. 1.3], which transforms a formula into a new formula in CNF by introducing new variables, causing only a linear increase in size. While this process preserves satisfiability, it need not preserve validity, the notion we are interested in. Instead of trying to adapt the Tseitin transformation to our setting, we

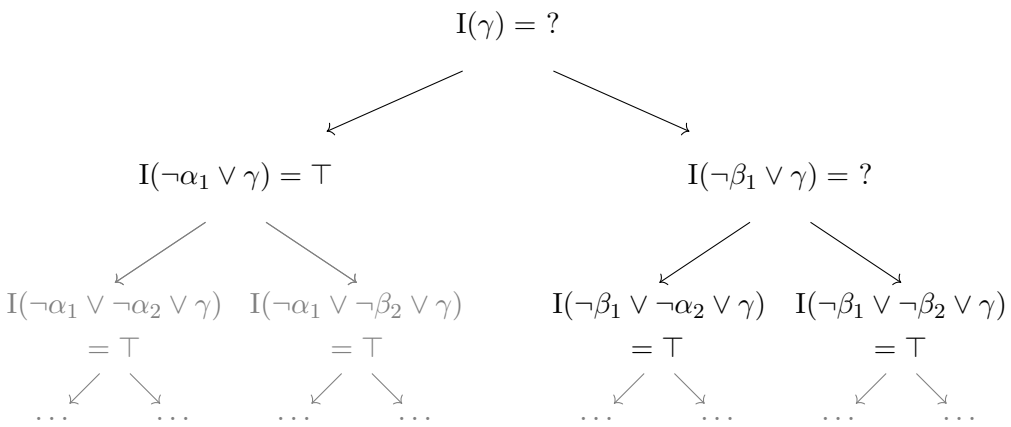


Figure 4.1: Illustration on how to incrementally compute the idealisation of $((\alpha_1 \vee \beta_1) \wedge (\alpha_2 \vee \beta_2) \wedge (\alpha_3 \vee \beta_3)) \to \gamma$. Grey coloured computations are avoided. The question mark "?" indicates that the idealisation could not be verified to be true within some fixed time frame.

instead explain an approach based on an incremental computation of the idealisation. It follows from the following observation. If $\varphi = (\alpha \vee \beta) \to \gamma$, then

$$\mathrm{I}(\varphi) = \mathrm{I}(\neg\alpha \vee \gamma) \wedge \mathrm{I}(\neg\beta \vee \gamma).$$

In such situations, it can make sense to compute the idealisation incrementally. First, we form $\mathrm{I}(\gamma)$ and if $\mathrm{I}(\gamma) = \top$, then also $\mathrm{I}(\varphi) = \top$ (by Lemma 4.3.8), and we are done. Only if $\mathrm{I}(\gamma) = \top$ cannot be verified fast enough (that is, within some fixed time frame), we compute $\mathrm{I}(\neg\alpha \vee \gamma)$ and $\mathrm{I}(\neg\beta \vee \gamma)$. The main advantage of this idea is that it can be applied recursively in case of several assumptions (see Figure 4.1 for an example). This is because $\bigwedge_i \alpha_i \to \gamma$ and $(\alpha_1 \to (\alpha_2 \to \cdots \to (\alpha_m \to \gamma)\dots))$ are logically equivalent, and thus, lead to the same idealisation. While, in the worst case, this technique still requires exponentially many ideal membership tests, it, in practice, usually allows to drastically reduce the number of polynomial computations.

No matter whether the optimisation discussed above is used or not, the crucial step of Procedure 8 requires the verification of finitely many ideal memberships. We note that this polynomial computation is independent of the sorts of the involved terms, providing a clear advantage over other verification approaches (such as sequent calculi) that require additional checks to ensure that the sort restrictions are respected at every step. Additionally, ideal membership $f \in (f_1, \dots, f_r)$ can be certified easily by providing a cofactor representation

of f with respect to the generators $f_1, \ldots, f_r$. Recall that such a cofactor representation can be computed with the help of Gröbner bases and serves as a proof that can be checked easily and independently of how it was obtained.

4.6 Fully worked example

As an example to illustrate the workings of the framework, we consider a classical statement from [PR81] about the existence of Moore-Penrose inverses in categories with involution.

Let $\mathcal{C}$ be a category with involution. A morphism $X\colon U \to V$ in $\mathcal{C}$ is *$*$-cancellable* if $X^*XP = X^*XQ$ implies $XP = XQ$ and $RXX^* = SXX^*$ implies $RX = SX$ for all morphisms P, Q, R, S. Moreover, X is *regular* if there is $A\colon V \to U$ with $XAX = X$, and X has a *Moore-Penrose inverse* if there is $X^\dagger\colon V \to U$ satisfying

$$XX^\dagger X = X, \qquad X^\dagger XX^\dagger = X^\dagger, \qquad (XX^\dagger)^* = XX^\dagger, \qquad (X^\dagger X)^* = X^\dagger X.$$

A morphism need not have a Moore-Penrose inverse. The following part of [PR81, Lem. 3] provides a sufficient condition for its existence.

Lemma 4.6.1. *Let $X\colon U \to V$ be a morphism in a preadditive category with involution. If X is $*$-cancellable and both X^*X and XX^* are regular, then X has a Moore-Penrose inverse.*

In the following, we describe how to prove Lemma 4.6.1 using the framework introduced in this chapter and Procedure 8.

Step 1: Choosing a signature As a first step, we have to choose a suitable signature. To this end, we fix $u, v \in \mathbf{Ob}$ and consider $\Sigma = (O, C, F, \sigma)$ with

1. $O = \{u, v\}$;
2. $C = \{i_u, i_v, 0_{u,u}, 0_{u,v}, 0_{v,u}, 0_{v,v}\}$;
3. $F = \{{}^*_{u,u}, {}^*_{u,v}, {}^*_{v,u}, {}^*_{v,v}\} \cup \{-_{\alpha,\beta}, +_{\alpha,\beta}, \cdot_{\alpha,\beta,\gamma} \mid \alpha, \beta, \gamma \in O\}$;
4. σ satisfies $\sigma(i_u) = (u, u)$, $\sigma(i_v) = (v, v)$, and $\sigma({}^*_{\alpha,\beta}) = (\alpha, \beta) \to (\beta, \alpha)$ for $\alpha, \beta \in O$;

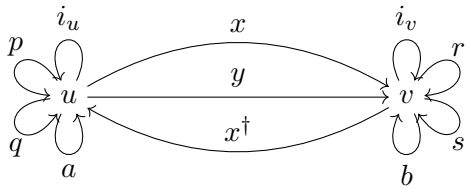


Figure 4.2: Sorts of the variables and constant symbols needed to translate Lemma 4.6.1.

Besides the required zero constant and arithmetic function symbols, the signature contains constant symbols i_u, i_v for the identity morphisms I_U, I_V and function symbols ${}^*_{u,u}, {}^*_{u,v}, {}^*_{v,u}, {}^*_{v,v}$ representing the involution. To simplify the notation, we will denote these function symbols all by the same symbol * in the following. Finally, we require σ to assign the expected sorts to these constant and function symbols.

Step 2: Translating the statement After fixing a signature, we can translate the statement about morphisms into an operator statement. To keep the notation uncluttered, we omit the sorts of the variables and constant symbols in the formula. Instead, we visualise the sorts via the directed multigraph in Figure 4.2, where each symbol s with $\sigma(s) = (\alpha, \beta)$ is represented by an edge with label s from vertex α to β. In the following, we write st for $s \cdot t$ to keep the expressions shorter. With this, Lemma 4.6.1 can be translated into the formula

$$\varphi = \forall x, a, b\, \exists p, q, r, s, y, x^\dagger : \varphi_{\mathrm{Id}} \wedge \varphi_{\mathrm{Reg}} \wedge \varphi_{\mathrm{Cancel}} \to \varphi_{\mathrm{MP}},$$

where

$$\begin{aligned}
\varphi_{\mathrm{Id}} &= yi_u \approx y \ \wedge\ i_v y \approx y \\
\varphi_{\mathrm{Reg}} &= x^*xax^*x \approx x^*x \ \wedge\ xx^*bxx^* \approx xx^* \\
\varphi_{\mathrm{Cancel}} &= (x^*xp \approx x^*xq \to xp \approx xq) \ \wedge\ (rxx^* \approx sxx^* \to rx \approx sx) \\
\varphi_{\mathrm{MP}} &= xx^\dagger x \approx x \ \wedge\ x^\dagger xx^\dagger \approx x^\dagger \ \wedge\ (xx^\dagger)^* \approx xx^\dagger \ \wedge\ (x^\dagger x)^* \approx x^\dagger x.
\end{aligned}$$

The formula consists of three assumptions φ_{Id}, φ_{Reg}, $\varphi_{\mathrm{Cancel}}$, capturing basic properties of the identity morphisms, the regularity of X^*X and XX^*, and the $*$-cancellability of X respectively. We have not encoded all properties of the identity morphisms in φ_{Id} but only those that will turn out relevant for the proof. The same also holds for the $*$-cancellability

assumption φ_{Cancel}. We do this to keep the formulas shorter and better readable. Of course, usually one does not know *a priori* which properties are relevant before finding a proof, and thus, one would encode all properties. The claimed property, the existence of a Moore-Penrose inverse of X, is translated into φ_{MP}. Formally, the assumptions should also include the axioms for the involution function symbols. Since these axioms are all very basic universal identities, we, as described in Section 4.5.1, omit them and instead consider the universal rewrite rules $(st)^* \mapsto t^*s^*$, $(t^*)^* \mapsto t$ that we use to simplify all occurring terms. Additionally, we apply another technique mentioned in Section 4.5.1 and add to every assumption the corresponding adjoint statement, that is, we replace φ_{Id}, φ_{Reg}, φ_{Cancel} by

$$\begin{aligned}
\varphi^*_{\text{Id}} &= \varphi_{\text{Id}} \ \wedge\ (yi_u)^* \approx y^* \ \wedge\ (i_v y)^* \approx y^* \\
\varphi^*_{\text{Reg}} &= \varphi_{\text{Reg}} \ \wedge\ (x^*xax^*x)^* \approx (x^*x)^* \ \wedge\ (xx^*bxx^*)^* \approx (xx^*)^* \\
\varphi^*_{\text{Cancel}} &= (x^*xp \approx x^*xq \to (xp \approx xq \wedge (xp)^* \approx (xq)^*)) \\
&\quad \wedge\ (rxx^* \approx sxx^* \to (rx \approx sx \wedge (rx)^* \approx (sx)^*))\,.
\end{aligned}$$

Step 3: Applying Procedure 8 Since φ is already a sentence in prenex normal form, we can immediately start with the Herbrandisation to remove the universal quantifier. In this process, we replace the variables x, a, b by new constant symbols, which we denote for simplicity also by x, a, b. By a slight abuse of notation, we keep the same notations $\varphi^*_{\text{Id}}, \varphi^*_{\text{Reg}}, \varphi^*_{\text{Cancel}}$, and φ_{MP} for the thereby obtained formulas. This yields the Herbrand normal form

$$\varphi^H = \exists p, q, r, s, y, x^\dagger : \varphi^*_{\text{Id}} \wedge \varphi^*_{\text{Reg}} \wedge \varphi^*_{\text{Cancel}} \to \varphi_{\text{MP}}.$$

Then we consider an enumeration $\varphi_1, \varphi_2, \ldots$ of the Herbrand expansion $H(\varphi^H)$ where

$$\varphi_1 = \varphi^H[p \mapsto ax^*x, q \mapsto i_u, r \mapsto xx^*b, s \mapsto i_v, y \mapsto x, x^\dagger \mapsto x^*bxax^*]$$

and set $\psi_1 = \varphi_1$.

Remark 4.6.2. *Without any prior knowledge, one would most likely choose a very different enumeration of $H(\varphi^H)$, maybe instantiating all variables first only with constants and then successively use more complex terms. In order to avoid unnecessary computations, we use an enumeration that starts with the correct instantiations. In this particular example,*

the instantiations of the assumptions can be guessed relatively easily from the available information, in particular, from φ_{Reg}. The instantiation x^*bxax^* of $x^\dagger$ can then be found automatically. Using, for example, the technique explained in Section 1.3, one can set $x^\dagger$ to a dummy variable d and compute a Gröbner basis of the ideal generated by all identities appearing in φ_1 with respect to an elimination order for d. This Gröbner basis then contains an element of the form $d - x^*bxax^*$, indicating that $x^\dagger \mapsto x^*bxax^*$ is the correct instantiation.

Now Ackermann's reduction is used to remove the function symbol * from ψ_1. After applying the rewrite rules $(st)^* \mapsto t^*s^*$, $(t^*)^* \mapsto t$ exhaustively, the only instances of this function symbol are $x^*, a^*, b^*, i_u^*, i_v^*$. We replace these terms by new constants, which we denote by the same symbols. This yields a new formula ψ_1^{flat}. Furthermore, we add the functional consistency constraints

$$\psi_1^{\mathrm{FC}} = \ (a \approx i_u \to a^* \approx i_u^*) \wedge (b \approx i_v \to b^* \approx i_v^*).$$

Note that other consistency constraints cannot be formed due to the restrictions imposed by the sorts of the symbols. The resulting formula is $\psi_1^{\mathrm{Ack}} = \psi_1^{\mathrm{FC}} \to \psi_1^{\mathrm{flat}}$.

Directly computing the idealisation of ψ_1^{Ack} would require 256 ideal membership tests. To reduce the number of polynomial computations, we apply the optimisation discussed in Section 4.5.1 and compute $\mathrm{I}(\psi_1^{\mathrm{Ack}})$ incrementally. Every implication $\alpha \to \beta$ corresponds to a disjunction $\neg\alpha \vee \beta$. So we remove all implications from the assumptions (including ψ_1^{FC}), that is, we remove ψ_1^{FC} and the instantiation of $\varphi_{\mathrm{Cancel}}^*$ from ψ_1^{Ack}. Then we form the CNF of the simplified formula, which yields only 4 clauses. Since they all share a common core corresponding to $\varphi_{\mathrm{Id}}^* \wedge \varphi_{\mathrm{Reg}}^*$, we denote

$$\begin{aligned} C_{\mathrm{core}} = {} & xi_u \not\approx x \ \vee \ i_v x \not\approx x \ \vee \ i_u^* x^* \not\approx x^* \ \vee \ x^* i_v^* \not\approx x^* \\ & \vee \ x^*xax^*x \not\approx x^*x \ \vee \ xx^*bxx^* \not\approx xx^* \ \vee \ x^*xa^*x^*x \not\approx x^*x \ \vee \ xx^*b^*xx^* \not\approx xx^*. \end{aligned}$$

With this, the 4 clauses are

$$\begin{aligned} C_1 &= C_{\text{core}} \vee xx^*bxax^*x \approx x, \\ C_2 &= C_{\text{core}} \vee x^*bxax^*xx^*bxax^* \approx x^*bxax^*, \\ C_3 &= C_{\text{core}} \vee xa^*x^*b^*xx^* \approx xx^*bxax^*, \\ C_4 &= C_{\text{core}} \vee x^*xa^*x^*b^* \approx x^*bxax^*x. \end{aligned}$$

As can be seen, we have one clause corresponding to each of the 4 identities in φ_{MP}.

None of the idealisations of $C_1, \ldots, C_4$ is true. So we include the first implication

$$x^*xax^*x \approx x^*xi_u \to (xax^*x \approx xi_u \wedge x^*xa^*x^* \approx i_u^*x^*) \tag{4.9}$$

from the instantiation of $\varphi^*_{\text{Cancel}}$ and consider

$$C_i' = C_i \vee x^*xax^*x \approx x^*xi_u \qquad \text{and} \qquad C_i'' = C_i \vee xax^*x \not\approx xi_u \vee x^*xa^*x^* \not\approx i_u^*x^*$$

for $i = 1, \ldots, 4$. Note that (4.9) is the first implication of $\varphi^*_{\text{Cancel}}$ with p and q substituted by ax^*x and i_u respectively.

A polynomial computation shows that $\mathrm{I}(C_i') = \top$ but $\mathrm{I}(C_i'') = \bot$ for all i. We note that, in this situation, we can indeed verify that the idealisations $\mathrm{I}(C_i'')$ are false as all ideals involved have finite Gröbner bases. So we continue with the 4 clauses $C_1'', \ldots, C_4''$ and keep adding in the remaining implications. After integrating the second implication

$$xx^*bxx^* \approx i_vxx^* \to (xx^*bx \approx i_vx \wedge x^*b^*xx^* \approx x^*i_v^*)$$

from $\varphi^*_{\text{Cancel}}$, all remaining idealisations become true. For example, the following representation certifies that $\mathrm{I}(C_1'' \vee xx^*bx \not\approx i_vx \vee x^*b^*xx^* \not\approx x^*i_v^*) = \top$:

$$xx^*bxax^*x - x \;=\; xx^*b(xax^*x - xi_u) \;+\; xx^*b(xi_u - x) \;+\; (xx^*bx - i_vx) \;+\; (i_vx - x).$$

This shows that the idealisations of all clauses of ψ_1^{Ack} are true, thus also $\mathrm{I}(\psi_1^{\text{Ack}}) = \top$. This finishes the proof of Lemma 4.6.1. In total, this approach required only 20 ideal membership tests compared to 256 if we had translated ψ_1^{Ack} directly. We note that all required polynomial computations are rather simple and lead to ideal membership certificates of similar size and complexity as the one shown above.

5 Practical aspects of applying the framework

The semi-decision procedure for verifying universal truth (Procedure 8) and the concepts required to state it (Herbrandisation, Ackermann's reduction, idealisation,...) are presented in the previous chapter in a very general setting, allowing to treat *arbitrary* operator statements. This makes the theory very powerful, however, as witnessed by the illustrative example in Section 4.6, also a bit cumbersome to apply in practice. In this chapter, we discuss different aspects relevant for applying the framework in practice, showing that, for most examples, the actual computations can be simplified a lot compared to the presentation in the previous chapter.

In particular, we first give a practical summary of the framework in Section 5.1, focusing on the simple, yet in practice most relevant, case of *existential quasi-identities*. This drastically reduces the complexity of the presentation compared to the general case of arbitrary first-order formulas. A self-contained description of the framework for the larger class of $\forall\exists$-statements is given in our work [BHR23, Sec. 5].

In Section 5.2, we then present how several common properties of matrices and linear operators can be expressed in terms of identities, and thus, treated within the framework. In particular, we discuss how to encode matrices with real entries, how to treat identity operators, and how to handle injectivity and surjectivity of operators. Leveraging the latter, we can also express the property of matrices having full row or column rank. Finally, we discuss the translation of range inclusions of linear operators into identities using Douglas' lemma [Dou66]. Each of these properties is accompanied by examples and insights into the practical application of our software package `operator_gb` (see also Section 6.1). This section is published in our work [BHR23].

An important aspect concerning the effectiveness of Procedure 8 is the order in which the elements in the Herbrand expansion are enumerated. Naively enumerating all possible expressions quickly becomes infeasible. Therefore, practical implementations of the procedure require techniques for identifying suitable candidate expressions. Translated to the setting of polynomials, this leads to a desire for methods that allow to search for elements of a particular form in a given ideal. In Section 5.3, we discuss well-known and present novel algorithmic techniques based on noncommutative Gröbner basis computations to do this and illustrate them by examples. In particular, we discuss classical ideal intersection techniques [Nor98; Mor16] and we present a new algorithm to compute generators of the intersection of a two-sided ideal with a right ideal. Furthermore, we discuss how an ideal can be intersected with a subalgebra of the free algebra. We also generalise a method to compute monomials in a commutative ideal [SST13; Mil16], leading to new algorithms that allow to compute homogeneous polynomials in a two-sided ideal and monomials in a right ideal. This section mostly appears in our work [HRR22a], with some minor aspects like Section 5.3.2 being presented here for the first time.

To end this chapter, we discuss how our algebraic perspective to proving operator statements can be leveraged to obtain short proofs. As our framework reduces the universal truth of operator statements to the verification of ideal membership, cofactor representations that certify the latter can be considered as a proof of the former. In Section 5.4, we discuss the problem of finding cofactor representations with a minimal number of terms. The results of this section also appear in our preprint [HV23a].

5.1 Practical summary

In the following, we describe the theory developed in Chapter 4 from a practical point of view, focusing on *existential quasi-identities*. Additionally, we limit our scope to arithmetic function symbols (addition, negation, and multiplication), excluding more general function symbols. Other (especially unary) function symbols can often be handled *ad hoc*, as described, for example, in Remark 1.2.1 for the adjoint operator or in Section 5.2.1 for the matrix transpose and entry-wise complex conjugation. We note that this restricted approach allows for a drastically simpler presentation, rendering it more accessible and efficiently implementable. Moreover, in practice, most statements can be effectively cast into this specific format.

To model statements about linear operators, or more generally about morphisms in preadditive semicategories, we consider a subset of many-sorted first-order logic. More precisely, we fix an enumerable set of *object symbols* $\mathbf{Ob} = \{v_1, v_2, \dots\}$ and call a pair $(u, v) \in \mathbf{Ob} \times \mathbf{Ob}$ a *sort*. We also fix an enumerable set of variables $\{x_1, x_2, \dots\}$ as well as, for each sort (u, v), a *zero constant* $0_{u,v}$. Furthermore, we fix a *sort function* σ mapping each variable x to a sort $\sigma(x) \in \mathbf{Ob} \times \mathbf{Ob}$ and each zero constant $0_{u,v}$ to $\sigma(0_{u,v}) = (u, v)$. Intuitively, variables correspond to basic operators and the zero constants model distinguished zero operators. The images of these symbols under the sort function σ represent their domains and codomains.

Using these basic symbols, we can construct terms, and building upon that, operator statements. Note that the following definition also extends the sort function from variables and constants to terms.

Definition 5.1.1. *A* term *is any expression that can be built up inductively using the following rules:*

1. *each variable x is a term of sort $\sigma(x)$;*
2. *each zero constant $0_{u,v}$ is a term of sort (u, v);*
3. *if t is a term, then $-t$ is a term of sort $\sigma(-t) \coloneqq \sigma(t)$;*
4. *if s, t are terms of sort $\sigma(s) = \sigma(t)$, then $s + t$ is a term of sort $\sigma(s + t) \coloneqq \sigma(s)$;*
5. *if s, t are terms of sort $\sigma(s) = (v, w)$, $\sigma(t) = (u, v)$, then st is a term of sort $\sigma(st) \coloneqq (u, w)$;*

Terms are simply all noncommutative polynomial expressions that can be formed from the variables and the zero constants under the restrictions imposed by the sort function. They correspond to all operators that can be formed from the basic operators with the arithmetic operations of addition, negation, and composition.

Definition 5.1.2. *An* operator statement *is a first-order formula that can be built up inductively using the following rules:*

1. *if s, t are terms of sort $\sigma(s) = \sigma(t)$, then $s \approx t$ is an operator statement;*

2. *if φ is an operator statement, then so is $\neg\varphi$;*

3. *if φ, ψ are operator statements, then so is $\varphi * \psi$ for $* \in \{\vee, \wedge, \rightarrow\}$;*

4. *if φ is an operator statement, then so is $Qx : \varphi$ for any variable x and $Q \in \{\exists, \forall\}$;*

We abbreviate a block of consecutive equally quantified variables $Qx_1Qx_2 \dots Qx_k$, with $Q \in \{\exists, \forall\}$, by $Qx_1 \dots x_k$, or simply by $Q\boldsymbol{x}$. If a term t or an operator statement φ depends on variables $\boldsymbol{x} = x_1, \dots, x_k$, we also write $t(\boldsymbol{x})$ and $\varphi(\boldsymbol{x})$ to emphasise this dependency.

An *interpretation* $\mathcal{I}$ allows to interpret an operator statement φ as a statement about morphisms in a preadditive semicategory $\mathcal{C}$. It assigns to each object symbol $u \in \mathbf{Ob}$ an object $\mathcal{I}(u) \in \mathrm{Ob}(\mathcal{C})$ and to each variable x of sort $\sigma(x) = (u, v)$ a morphism $\mathcal{I}(x) \colon \mathcal{I}(u) \rightarrow \mathcal{I}(v)$. Each zero constant $0_{u,v}$ is mapped to the zero morphism in the abelian group $\mathrm{Mor}(\mathcal{I}(u), \mathcal{I}(v))$. This ensures that the terms in φ are translated into well-formed morphisms in $\mathcal{C}$. Then φ can be evaluated to a truth value by interpreting the boolean connectives and the quantifiers like in classical first-order logic, interpreting $\approx$ as the identity in $\mathcal{C}$.

Definition 5.1.3. *An operator statement φ is* universally true *if φ evaluates to true under all possible interpretations in every preadditive semicategory $\mathcal{C}$.*

Remark 5.1.4. *For a formal definition and more in-depth discussion of interpretations and universal truth of operator statements, we refer to Sections 2.5.2 and 4.1.*

Note that an interpretation of φ depends implicitly on the sort function σ, and thus, so does the semantic evaluation of φ. An operator statement may be universally true with respect to one sort function but not with respect to another. For instance, statements that hold for square matrices may not hold for rectangular matrices. Therefore, we should only refer to universal truth with respect to a specific sort function. For the sake of brevity, we assume a fixed sort function σ and disregard this dependency in the following.

Definition 5.1.5. *An* existential quasi-identity *is an operator statement of the form*

$$\forall \boldsymbol{x} \exists \boldsymbol{y} : \bigwedge_{i=1}^{m} s_i(\boldsymbol{x}) \approx t_i(\boldsymbol{x}) \rightarrow p(\boldsymbol{x}, \boldsymbol{y}) \approx q(\boldsymbol{x}, \boldsymbol{y}).$$

Remark 5.1.6. *In the definition of existential quasi-identities, we also allow degenerate cases without universally or existentially quantified variables, that is, where $\boldsymbol{x} = \emptyset$ or $\boldsymbol{y} = \emptyset$. However, note that an existential quasi-identity has to be a sentence, meaning that all variables that appear have to be quantified by a quantifier.*

In the following theorem, we show that universal truth of existential quasi-identities can be easily characterised by ideal membership of noncommutative polynomials. As the proof will show, this result arises by specialising the semi-decision Procedure 8 for verifying universal truth to existential quasi-identities. In the following, we identify each term $t(\boldsymbol{x})$ as a noncommutative polynomial in $\mathbb{Z}\langle\boldsymbol{x}\rangle$ and translate each equality $s(\boldsymbol{x}) \approx t(\boldsymbol{x})$ into the noncommutative polynomial $s(\boldsymbol{x}) - t(\boldsymbol{x}) \in \mathbb{Z}\langle\boldsymbol{x}\rangle$.

Theorem 5.1.7. *An existential quasi-identity*

$$\forall \boldsymbol{x} \exists \boldsymbol{y} : \bigwedge_{i=1}^{m} s_i(\boldsymbol{x}) \approx t_i(\boldsymbol{x}) \to p(\boldsymbol{x}, \boldsymbol{y}) \approx q(\boldsymbol{x}, \boldsymbol{y}),$$

where $\boldsymbol{y} = y_1, \ldots, y_k$, is universally true if and only if there exist terms $\boldsymbol{z} = z_1(\boldsymbol{x}), \ldots, z_k(\boldsymbol{x})$ depending only on $\boldsymbol{x}$ such that $\sigma(z_j) = \sigma(y_j)$, for all $j = 1, \ldots, k$, and such that the ideal membership

$$p(\boldsymbol{x}, \boldsymbol{z}) - q(\boldsymbol{x}, \boldsymbol{z}) \in \big(s_1(\boldsymbol{x}) - t_1(\boldsymbol{x}), \ldots, s_m(\boldsymbol{x}) - t_m(\boldsymbol{x})\big)$$

holds in the free algebra $\mathbb{Z}\langle\boldsymbol{x}\rangle$.

Proof. We show that applying Procedure 8 to an existential quasi-identity corresponds precisely the computation described in the theorem. First, note that the Herbrand normal form of an existential quasi-identity, is the formula itself, only with the universal quantifier removed – here we assume, without loss of generality, that the new constant symbols are given the same names as the variables they replace. Furthermore, the Herbrand expansion of this normal form is the set of all formulas of the form

$$\bigwedge_{i=1}^{m} s_i(\boldsymbol{x}) \approx t_i(\boldsymbol{x}) \to p(\boldsymbol{x}, \boldsymbol{z}) \approx q(\boldsymbol{x}, \boldsymbol{z})$$

with terms $\boldsymbol{z} = z_1(\boldsymbol{x}), \ldots, z_n(\boldsymbol{x})$ that depend only on $\boldsymbol{x}$ and satisfy $\sigma(z_j) = \sigma(y_j)$ for all $j = 1, \ldots, k$. Note that the symbols $\boldsymbol{x}$ now have to be considered as constant symbols,

as the variables have been replaced during the Herbrandisation, and thus, the formulas in the Herbrand expansion are indeed ground. Then, Procedure 8 has to verify whether the idealisation $I(\varphi_1 \vee \cdots \vee \varphi_n)$ is true for some $n \in \mathbb{N}$ and elements φ_l in the Herbrand expansion, $l = 1, \ldots, n$. Note that Ackermann's reduction is not required because existential quasi-identities do not contain any non-arithmetic function symbols. If we write each $\varphi_l = \bigwedge_{i=1}^{m} s_i(\boldsymbol{x}) \approx t_i(\boldsymbol{x}) \to p(\boldsymbol{x}, \boldsymbol{z_l}) \approx q(\boldsymbol{x}, \boldsymbol{z_l})$, then a conjunctive normal form of $\varphi_1 \vee \cdots \vee \varphi_n$ is given by the single clause

$$\bigvee_{i=1}^{m} s_i(\boldsymbol{x}) \not\approx t_i(\boldsymbol{x}) \vee \bigvee_{l=1}^{n} p(\boldsymbol{x}, \boldsymbol{z_l}) \approx q(\boldsymbol{x}, \boldsymbol{z_l}).$$

Finally, this idealisation is true if and only if the ideal membership

$$p(\boldsymbol{x}, \boldsymbol{z_l}) - q(\boldsymbol{x}, \boldsymbol{z_l}) \in (s_1(\boldsymbol{x}) - t_1(\boldsymbol{x}), \ldots, s_m(\boldsymbol{x}) - t_m(\boldsymbol{x}))$$

holds for some $1 \leq l \leq n$. □

If an existential quasi-identity φ contains no existentially quantified variables, Theorem 5.1.7 reduces the universal truth of φ to the verification of the single ideal membership

$$p(\boldsymbol{x}) - q(\boldsymbol{x}) \in (s_1(\boldsymbol{x}) - t_1(\boldsymbol{x}), \ldots, s_m(\boldsymbol{x}) - t_m(\boldsymbol{x})).$$

Since the latter problem is semi-decidable, this immediately yields a semi-decision procedure for universal truth for this kind of statements.

If φ also contains existentially quantified variables, we can proceed as follows: To find suitable terms $\boldsymbol{z} = z_1(\boldsymbol{x}), \ldots, z_k(\boldsymbol{x})$ as required by the theorem, we first search for elements of the form $p(\boldsymbol{x}, \boldsymbol{z}) - q(\boldsymbol{x}, \boldsymbol{z})$ with arbitrary $\boldsymbol{z}$ in the ideal $(s_1(\boldsymbol{x}) - t_1(\boldsymbol{x}), \ldots, s_m(\boldsymbol{x}) - t_m(\boldsymbol{x}))$ and then check *a posteriori* if any of these elements corresponds to a formula $p(\boldsymbol{x}, \boldsymbol{z}) \approx q(\boldsymbol{x}, \boldsymbol{z})$ with terms $\boldsymbol{z}$ as required by the theorem. We note that this is also the technique we have employed in Section 1.3 to prove the existence of the Moore-Penrose inverse for matrices. In Section 5.3, we present several methods that allow to search for polynomials of a specific form in an ideal and illustrate how they can be used to prove existential quasi-identities.

5.2 Treating common properties

5.2.1 Real matrices

A property that appears regularly in matrix statements, especially in combination with the Hermitian adjoint A^*, is that of having matrices over the reals. It can be encoded by decomposing the Hermitian adjoint A^* into an entry-wise complex conjugation, denoted by A^{C}, followed by a transposition, that is, $A^* = (A^{\mathsf{C}})^{\mathsf{T}}$. With this, a matrix A being real can be expressed algebraically by the identity $A = A^{\mathsf{C}}$, exploiting the fact that the conjugate of a real number is the number itself.

To model the complex conjugation and the transposition on the polynomial level, we proceed analogous to modelling the involution (see Remark 1.2.1). We introduce additional variables a^{C} and a^{T} for the complex conjugate and the transpose, respectively, of each basic operator A. Additionally, for every assumption $S = T$, we have to translate, next to the corresponding adjoint identity $S^* = T^*$, now also the transposed identity $S^{\mathsf{T}} = T^{\mathsf{T}}$ as well as the conjugated identity $S^{\mathsf{C}} = T^{\mathsf{C}}$ into polynomials. These additional identities first have to be simplified using the following rules that relate the different function symbols to each other:

$$(S+T)^{\alpha} = S^{\alpha} + T^{\alpha}, \qquad (T^{\alpha})^{\beta} = \begin{cases} T & \text{if } \alpha = \beta \\ T^{\gamma} & \text{if } \alpha \neq \beta \end{cases},$$
$$(ST)^{\mathsf{C}} = S^{\mathsf{C}}T^{\mathsf{C}}, \qquad (ST)^{\delta} = T^{\delta}S^{\delta},$$

with $\alpha, \beta, \gamma, \delta \in \{*, \mathsf{C}, \mathsf{T}\}$ such that $\gamma \neq \alpha, \beta$ and $\delta \neq \mathsf{C}$.

As an illustrative example, we consider the statement that the Moore-Penrose inverse B of a real matrix A is real as well. With the help of our software package `operator_gb`, it can be proven as follows. We refer to Section 6.1 for further information on the corresponding commands.

```
sage: from operator_gb import *

sage: F.<a, a_tr, a_c, a_adj, b, b_tr, b_c, b_adj> = FreeAlgebra(QQ)
```

```
# the basic assumptions
sage: Pinv_b = add_adj(pinv(a, b, a_adj, b_adj))
# the transposed and conjugated assumptions
sage: Pinv_b_tr = [a_tr*b_tr*a_tr - a_tr, b_tr*a_tr*b_tr - b_tr,
....: a_tr*b_tr - b_c*a_c, b_tr*a_tr - a_c*b_c]
sage: Pinv_b_c = [a_c*b_c*a_c - a_c, b_c*a_c*b_c - b_c]
# assumption that a is real
sage: a_real = [a - a_c, a_tr - a_adj]

sage: assumptions = Pinv_b + Pinv_b_tr + Pinv_b_c + a_real
sage: proof = certify(assumptions, b - b_c)
Computing a (partial) Gröbner basis and reducing the claims...

Done! Ideal membership of all claims could be verified!
```

5.2.2 Identity operators

Next, we discuss how to handle identity matrices and operators. While zero operators have a natural translation into the zero polynomial, identity operators cannot be directly mapped to the multiplicative identity 1 in the free algebra, as this would constitute a many-to-one mapping and a loss of information. We note that this is not an issue when mapping all zero operators to the zero polynomial, as the zero polynomial does not affect any polynomial computations.

Instead, identity operators have to be treated like any other basic operator, which means introducing a new indeterminate i_u for every identity operator I_U and explicitly adding the identities satisfied by I_U to the assumptions. In particular, these are the idempotency of I_U, the fact that I_U is self-adjoint, and the identities $AI_U = A$ and $I_U B = B$ for all basic operators A, B for which these expressions are well-defined.

We illustrate the handling of identity operators in the next section.

5.2.3 Injectivity, surjectivity, and full matrix ranks

Injectivity and surjectivity of operators appear regularly as properties in statements. They can be encoded by exploiting the following classical fact.

Lemma 5.2.1. *Let U, V be nonempty sets. A function $A\colon U \to V$ is*

1. *injective if and only if A has a left inverse $B\colon V \to U$;*
2. *surjective if and only if A has a right inverse $C\colon V \to U$;*

Thus, an assumption of injectivity of an operator A can be encoded via the identity $BA = I_U$, where B is a new operator that does not satisfy any additional hypotheses and I_U is the identity on U. Analogously, surjectivity of A corresponds to the identity $AC = I_V$. For proving injectivity or surjectivity of an operator in our setting, we have to show the existence of a left or right inverse by finding an explicit expression for such an operator.

As a special case of the discussion above, we also obtain a way to encode the property of a matrix having full row or column rank. This follows from the fact that a matrix A has full row rank if and only if the associated linear function is surjective, which, by Lemma 5.2.1, is the case if and only if A has a right inverse. Dually, A has full column rank if and only if A has a left inverse.

To illustrate the handling of full rank assumptions as well as of identity matrices, we consider the statement: If $A = BC$ is a full rank decomposition of a matrix A, that is, B has full column rank and C has full row rank, then $A^\dagger = C^\dagger B^\dagger$. Using our software `operator_gb`, it can be proven as follows.

```
sage: from operator_gb import *

sage: F.<a, b, c, i, u, v, x, y, z, a_adj, b_adj, c_adj, i_adj,
....: u_adj, v_adj, x_adj, y_adj, z_adj> = FreeAlgebra(QQ)

sage: Pinv_a = pinv(a, x, a_adj, x_adj)
sage: Pinv_b = pinv(b, y, b_adj, y_adj)
```

```
sage: Pinv_c = pinv(c, z, c_adj, z_adj)
# full ranks encoded via one-sided inverses
sage: rank_decomp = [a - b*c, u*b - i, c*v - i]
# encode identity i
sage: id = [i*i - i, i - i_adj, b*i - b, i*y - y, i*c - c,
....: z*i - z, i*u - u, v*i - v]

sage: assumptions = add_adj(Pinv_a + Pinv_b + Pinv_c +
....: rank_decomp + id)
sage: claim = x - z*y
sage: proof = certify(assumptions, claim)
Computing a (partial) Groebner basis and reducing the claims...

Starting iteration 5...
Done! Ideal membership of all claims could be verified!
```

Remark 5.2.2. *We note that the* `certify` *routine (more precisely, the Gröbner basis computation underlying this command) is an iterative procedure. By default, the package informs about the computational progress of this procedure by printing an update message* `Starting iteration n...` *every fifth iteration, see also Section 6.1.*

5.2.4 Range inclusions

Another common class of properties are conditions on ranges and kernels, like the inclusion of ranges $\mathcal{R}(A) \subseteq \mathcal{R}(B)$ of operators A, B. In case of linear operators over a field, such a range inclusion can be translated into the existence of a factorisation $A = BX$ for some operator X. We note that, in Hilbert and Banach spaces, this is the well-known factorisation property in Douglas' lemma [Dou66].

Thus, also facts like $\mathcal{R}(A^\dagger) = \mathcal{R}(A^*)$ can be treated within the framework by finding explicit factorisations of $A^\dagger$ and A^* in terms of the other. Using our software package `operator_gb`, such factorisations can be found easily. We refer to Section 6.1 for the implemented methods to do this and for the corresponding commands, and to the following Section 5.3 for an explanation of these techniques.

```
sage: from operator_gb import *

sage: F.<a, a_adj, a_dag, a_dag_adj> = FreeAlgebra(QQ)
sage: Pinv_a = add_adj(pinv(a, a_dag, a_adj, a_dag_adj))
sage: I = NCIdeal(Pinv_a)
# R(A^\dag) \subseteq R(A^*)
sage: I.find_equivalent_expression(a_dag, prefix=a_adj,
....: heuristic='naive')

                    [a_dag - a_adj*a_dag_adj*a_dag]

# R(A^*) \subseteq R(A^\dag)
sage: I.find_equivalent_expression(a_adj, prefix=a_dag,
....: heuristic='naive')

                          [a_adj - a_dag*a*a_adj]
```

5.3 Heuristics for computing elements of certain form

In this section, we present classical as well as novel algorithms for finding elements of a particular form in a given ideal in the free algebra by exploiting Gröbner basis techniques. The ability to do this is one of the crucial steps in the semi-decision Procedure 8 for verifying universal truth, as well as in the simplified Theorem 5.1.7 for treating existential quasi-identities. Our techniques are also accompanied by illustrative examples.

The general methodology for all techniques presented in this section is the same: Starting from an ideal in which we want to search for specific polynomials, we first specify a certain subideal (for example, by ideal intersections), which has to contain all polynomials of the desired form. Then we enumerate generators of this subideal and search for a polynomial of the desired form. Note that, in general, there is no guarantee that we can find such a polynomial among the generators of the subideal. However, in practice, this was the case for almost all algebraic proofs of operator statements we considered so far.

In particular, the first Section 5.3.1 is devoted to exploiting ideal intersections for finding elements of which only certain factors in a factorisation are known. We discuss well-known ideal intersection techniques [Nor98; Mor16] and we present a new algorithm to compute generators of the intersection of a two-sided ideal with a right ideal. We also characterise when a two-sided ideal is finitely generated as a right ideal. Then, in Section 5.3.2, we describe how a result from [Nor98] can be used to compute elements in the intersection of an ideal with a subalgebra of the free algebra. Following upon that, we present an algorithm for computing homogeneous polynomials in noncommutative ideals and discuss how it can be used to find positive factorisations of linear operators. This method is a generalisation of the techniques presented in [SST13; Mil16] for computing monomials in a commutative polynomial ideal. Finally, in Section 5.3.4, we show how an adaptation of this procedure also allows to find monomials in right ideals.

We note that the methods presented in this chapter are not exhaustive. There also exist other techniques that allow to compute elements of certain form in noncommutative polynomial ideals. For example, variable elimination [Sch21, Sec. 4.2] or employing an ansatz [Clu+18] are simple yet effective methods commonly used in practice. Additionally, many problems can be reformulated as factorisation problems in a suitable quotient of the free algebra. In [BHL17], it was shown that this quotient is a finite factorisation domain if it admits a finite dimensional filtration and a terminating algorithm was given that allows to compute all such factorisations in the affirmative case. Unfortunately, however, in the context of problems related to operator statements, we usually do not have access to such a finite dimensional filtration (in fact, most of the time we do not even know whether it exists) as this would require a description of a complete Gröbner basis.

To simplify the considered setting, we work in the free algebra $K\langle X\rangle$ over a field K for the rest of this section.

5.3.1 Ideal intersections

Ideal intersections provide a useful tool for finding elements of certain forms in an ideal $I \trianglelefteq K\langle X\rangle$. For example, they allow to systematically search for elements of the form $lar \in I$ with known $a \in K\langle X\rangle$ but unknown $l, r \in K\langle X\rangle$ by intersecting I with the ideal (a). More specifically, by intersecting I with the right ideal $(a)_\rho$, we can compute polynomials of the form $ar \in I$, which allows to search for elements with a known prefix.

As an example where ideal intersections can help to prove an operator statement, we consider the following characterisation of the solvability of the operator equation $AYB = C$ in Hilbert spaces from [AG10, Prop 3.3]. In the following, $P^\dagger$ denotes the Moore-Penrose inverse of a bounded linear operator P and P^* denotes the adjoint operator. Furthermore, $\mathcal{R}(P)$ denotes the range of P.

Theorem 5.3.1. *Let $A\colon \mathcal{H}_4 \to \mathcal{H}_2$, $B\colon \mathcal{H}_1 \to \mathcal{H}_3$, $C\colon \mathcal{H}_1 \to \mathcal{H}_2$ be bounded linear operators on complex Hilbert spaces. There exists a bounded linear operator $Y\colon \mathcal{H}_3 \to \mathcal{H}_4$ such that $AYB = C$ if and only if $\mathcal{R}(C) \subseteq \mathcal{R}(A)$ and $\mathcal{R}((A^\dagger C)^*) \subseteq \mathcal{R}(B^*)$.*

We discuss how to prove the sufficiency of the range inclusions in the theorem above by applying Theorem 5.1.7. First, all properties appearing in the statement have to be phrased in terms of identities. Using Douglas' lemma (see Section 5.2.4), the postulated range inclusions can be translated into the existence of operators P, Q such that $C = AP$ and $(A^\dagger C)^* = B^*Q$. The existence of the Moore-Penrose inverse $A^\dagger$ is encoded in terms of identities by adding the four Penrose identities for $A^\dagger$ to our assumptions. With this, Theorem 5.3.1 can be translated into the existential quasi-identity

$$\psi \coloneqq \forall \boldsymbol{x} \exists y : (\varphi \to ayb \approx c)\,,$$

where $\boldsymbol{x}$ abbreviates a sequence of 12 variables representing $A, A^\dagger, B, C, P, Q$ and their respective adjoint operators, and φ is a conjunction of the defining identities of $A^\dagger$, the two identities for the range inclusions, and the corresponding adjoint statements. By Theorem 5.1.7, proving ψ to be universally true corresponds to verifying that the polynomial $ayb - c$, for a suitable choice of y, is contained in the ideal generated by the polynomials F corresponding to the identities in φ. The sorts of the involved symbols are depicted in Figure 5.1. Each symbol s of sort (α, β) is represented by an edge with label s from vertex α to β.

Thus, finding a solution Y of $AYB = C$ corresponds to finding a suitable polynomial $ayb - c \in (F)$, where a, b, c are known but y is an unknown polynomial. To find such an element, we intersect (F) with the right ideal $(a, c)_\rho$ and enumerate generators of this intersection. Among these generators, we then search for a polynomial of the form $ar - c$ with the additional requirement that the free part r has to end with the variable b. If we can find such a polynomial and show that it is compatible with the sorts depicted in

Figure 5.1, then this shows that ψ is universally true, thus proving the sufficiency part of Theorem 5.3.1. This approach is carried out successfully in Example 5.3.17 at the end of this section.

In the following, we discuss how generating sets of several different kinds of ideal intersections can be computed. Recall that $I, J, \ldots$ denote two-sided ideals, while $I_\rho, J_\rho, \ldots$ denote right ideals.

In the case of commutative polynomials, two ideals $I, J \subseteq K[X]$ can be intersected by computing the elimination ideal $tI + (1-t)J \cap K[X]$, where t denotes a new tag variable. It is well-known that the same also works for intersecting one-sided ideals in the free algebra (see, for example, [Mor16, Rem. 48.8.6]).

Theorem 5.3.2. *Let $I_\rho = (f_1, \ldots, f_r)_\rho$, $J_\rho = (g_1, \ldots, g_s)_\rho \trianglelefteq_r K\langle X\rangle$ be two right ideals. Furthermore, let $t \notin X$ be a new indeterminate. Consider the right ideal*

$$H_\rho = \big(tf_i, (1-t)g_j \mid 1 \leq i \leq r, 1 \leq j \leq s\big)_\rho.$$

Then $I_\rho \cap J_\rho = H_\rho \cap K\langle X\rangle$.

Consequently, making use of the elimination property of right Gröbner bases (Proposition 2.4.68), a right Gröbner basis of $I_\rho \cap J_\rho$ can be obtained by computing a right Gröbner basis G of H_ρ with respect to an elimination order for t and selecting all polynomials in G not involving the variable t. Proposition 2.4.71 and Lemma 2.4.72 ensure that the Gröbner basis computed in this way is always finite.

In [Nor98, Thm. 2], it was shown that the elimination property of Gröbner bases also allows to compute the intersection of two-sided ideals provided that we introduce the commutator relations $tx_k - x_k t$, for all $k = 1, \ldots, n$, between the new tag variable t and the elements of X.

Theorem 5.3.3. *Let $I = (f_1, \ldots, f_r)$, $J = (g_1, \ldots, g_s) \trianglelefteq K\langle X\rangle$ be two ideals. Furthermore, let $t \notin X$ be a new indeterminate. Consider the ideal*

$$H = \big(tf_i, (1-t)g_j, tx_k - x_k t \mid 1 \leq i \leq r, 1 \leq j \leq s, 1 \leq k \leq n\big).$$

Then $I \cap J = H \cap K\langle X\rangle$.

Hence, as for right ideals, a Gröbner basis of the intersection can be obtained by intersecting a Gröbner basis G of H, computed with respect to an elimination order for t, with $K\langle X\rangle$. However, in contrast to the one-sided case, we cannot expect this Gröbner basis to be finite in general.

Next, we consider the third and for our application most relevant type of ideal intersection – the intersection of a two-sided ideal I with a right ideal J_ρ. The basic idea to compute the intersection $I \cap J_\rho$ is to consider I as a right ideal and to intersect two right ideals. As Theorem 5.3.2 provides an algorithmic way to compute the intersection of two right ideals, it remains to discuss how to obtain a right generating set of an ideal I. The following proposition is a specialisation of a result in [Gre00] for path algebras and tells us how the reduced Gröbner basis of an ideal I can be used to obtain a right Gröbner basis of I. In particular, this provides a way to obtain a right generating set of a two-sided ideal.

Proposition 5.3.4. *Let $I \trianglelefteq K\langle X\rangle$ and let G be its reduced Gröbner basis. The set*

$$\begin{aligned}\rho(I) \coloneqq \{wg \mid\ & w \in \langle X\rangle, g \in G,\\ & p \text{ is irreducible w.r.t. } \to_G \text{ for any proper prefix } p \text{ of } \operatorname{lm}(wg)\}\end{aligned}$$

is a right Gröbner basis of I. Furthermore, $\operatorname{lm}(f)$ is irreducible with respect to $\to_{\rho,\rho(I)\setminus\{f\}}$ for all $f \in \rho(I)$.

Proof. Follows from [Gre00, Prop 7.1] since $K\langle X\rangle$ can be viewed as a path algebra with only one vertex. □

Remark 5.3.5. *Equivalently, $\rho(I)$ can be written as*

$$\rho(I) = \{wg \mid w \in \langle X\rangle_{\mathrm{irr,I}}, g \in G, p \in \langle X\rangle_{\mathrm{irr,I}} \text{ for any proper prefix } p \text{ of } \operatorname{lm}(wg)\}, \quad (5.1)$$

where G is the reduced Gröbner basis of I and $\langle X\rangle_{\mathrm{irr,I}} \coloneqq \langle X\rangle \setminus \operatorname{lm}(I)$ is the set of all irreducible words with respect to $\to_G$.

It follows from the uniqueness of the reduced Gröbner basis (Proposition 2.4.41), that, for a fixed monomial order, the set $\rho(I)$ only depends on I. In the following, we discuss under which conditions a two-sided ideal I is finitely generated as a right ideal. The following result tells us that this is the case if and only if the set $\rho(I)$ is finite.

Proposition 5.3.6. *For $I \trianglelefteq K\langle X\rangle$, the following are equivalent:*

1. *I is finitely generated as a right ideal;*
2. *I has a finite right Gröbner basis;*
3. *the set $\rho(I)$ is finite;*

Proof. The implications $3 \implies 2 \implies 1$ are clear, and $1 \implies 2$ follows from Proposition 2.4.71 and Lemma 2.4.72. For $2 \implies 3$, assume, for contradiction, that I has a finite right Gröbner basis, say $H = \{h_1, \ldots, h_m\}$, but that $\rho(I)$ is infinite. Since $\rho(I)$ is a right Gröbner basis of I, there exist $g_i \in \rho(I)$ such that $\mathrm{lm}(g_i)$ is a prefix of $\mathrm{lm}(h_i)$ for all $i = 1, \ldots, m$. Now let $g \in \rho(I) \setminus \{g_1, \ldots, g_m\}$ be arbitrary but fixed. Such an element exists since $\rho(I)$ is infinite. Because H is a right Gröbner basis of I and $\rho(I) \subseteq I$, there exists $1 \leq i \leq m$ such that $\mathrm{lm}(h_i)$ is a prefix of $\mathrm{lm}(g)$. But then $\mathrm{lm}(g_i)$ is also a prefix of $\mathrm{lm}(g)$, and since $g \neq g_i$, this is a contradiction to the assertion of Proposition 5.3.4 that $\mathrm{lm}(f)$ is irreducible with respect to $\to_{\rho,\rho(I)\setminus\{f\}}$ for all $f \in \rho(I)$. □

The set $\rho(I)$ depends on the monomial order with respect to which the reduced Gröbner basis of I is computed. However, Proposition 5.3.6 implies that the finiteness of $\rho(I)$ is independent of the monomial order.

We will now investigate under which conditions the set $\rho(I)$ is finite. Clearly, if the reduced Gröbner basis G of I is infinite, then so must be $\rho(I)$, or equivalently, the finiteness of $\rho(I)$ implies the finiteness of G. The converse, however, need not hold. The finiteness of G alone is only sufficient to guarantee that also $\rho(I)$ is finite if I is the trivial ideal $I = \{0\}$. In this case, $G = \emptyset = \rho(I)$. We will see that, in all other cases, we need the additional requirement that also $\langle X\rangle_{\mathrm{irr,I}}$ is finite. To this end, we recall the definition of a zero-dimensional ideal.

Definition 5.3.7. *An ideal $I \trianglelefteq K\langle X\rangle$ is* zero-dimensional *if the quotient ring $K\langle X\rangle/I$ is finite dimensional as a K-vector space.*

The following result follows immediately from Theorem 2.4.45.

Corollary 5.3.8. *An ideal $I \trianglelefteq K\langle X\rangle$ is zero-dimensional if and only if $\langle X\rangle_{\mathrm{irr,I}}$ is finite.*

A nice property of zero-dimensional ideals is that they admit finite Gröbner bases with respect to any monomial order, see [Mor87, Prop. 5.1].

Lemma 5.3.9. *The reduced Gröbner basis of a zero-dimensional ideal is finite with respect to any monomial order.*

Moreover, being zero-dimensional is also a sufficient condition for an ideal to have a finite right Gröbner basis.

Lemma 5.3.10. *The set $\rho(I)$ is finite for any zero-dimensional ideal $I \trianglelefteq K\langle X\rangle$.*

Proof. It follows from Lemma 5.3.9 that the reduced Gröbner basis of I is finite. Then the finiteness of $\rho(I)$ follows directly from the representation (5.1) and the fact that $\langle X\rangle_{\mathrm{irr,I}}$ is finite by Corollary 5.3.8. □

Surprisingly, for a nontrivial ideal, the property of being zero-dimensional is also a necessary condition for it to have a finite right Gröbner basis. This fact is captured by the following lemma.

Lemma 5.3.11. *Let $I \trianglelefteq K\langle X\rangle$ be a nontrivial ideal. If I has a finite right Gröbner basis, then I is zero-dimensional.*

Proof. Let G be a finite right Gröbner basis of I and denote $N = \max\{|\mathrm{lm}(g)| \mid g \in G\}$. Let $w \in \langle X\rangle$ such that $|w| \geq N$ and choose $g \in G$ arbitrary. Then $wg \in I$, and therefore, since G is a right Gröbner basis, there exists $g' \in G$ such that $\mathrm{lm}(g')$ is a prefix of $\mathrm{lm}(wg)$. By the way w was chosen, $\mathrm{lm}(g')$ is already a prefix of w, showing that $w \notin \langle X\rangle_{\mathrm{irr,I}}$. Since w was arbitrary of length at least N, this shows that $\langle X\rangle_{\mathrm{irr,I}}$ can contain only finitely many elements. Hence, I is zero-dimensional by Corollary 5.3.8. □

Combining Proposition 5.3.6 with Lemma 5.3.10 and Lemma 5.3.11, we arrive at the following result.

Theorem 5.3.12. *For $I \trianglelefteq K\langle X\rangle$, the following are equivalent:*

1. *I is finitely generated as a right ideal;*

2. I has a finite right Gröbner basis;

3. the set $\rho(I)$ is finite;

4. I is zero-dimensional or the trivial ideal;

However, the condition of I being zero-dimensional is very strong and in practical applications rarely fulfilled. Typically, the reduced Gröbner basis of I is not even finite. In such cases, there is no chance to obtain a finite right generating set. Consequently, in practice, we have to content ourselves with finite approximations of $\rho(I)$ and can therefore only work with a right subideal $I_\rho \subsetneq I$. *A priori*, it is usually not clear how to choose such a finite approximation. In our practical applications, we have so far simply selected all elements in $\rho(I)$ up to a certain degree bound. Thus far, this has worked very well, allowing us to find the correct polynomials.

Moreover, when working with polynomials originating from operator statements, we can choose the subideal I_ρ so that it still contains all polynomials that are of interest to us and such that I_ρ is more likely to have a finite right generating set. This follows from the fact that, for proving operator statements, we are only interested in polynomials in I that correspond to actual operator identities, that is, in polynomials whose terms are compatible with the sorts of the used signature. We formalise this idea via the following definition, which is based on the notions introduced in Sections 4.1 and 4.3.

Definition 5.3.13. *Let $\Sigma = (O, C, F, \sigma)$ be a signature and let $X \subseteq \mathbf{Var} \cup (C \setminus \mathbf{Zero})$. A polynomial $f \in K\langle X\rangle$ is* compatible with the signature Σ *if there exists an arithmetic operator statement $\varphi = s \approx t \in \mathbf{Form}(\Sigma)$ such that $f = T_\varphi(s) - T_\varphi(t)$, using the translation function T_φ described in Section 4.3.1.*

Example 5.3.14. *Consider a signature $\Sigma = (O, C, F, \sigma)$ with $O = \{u, v\}$, $c \in C$, and σ such that*

$$\sigma(a) = (u, v), \qquad \sigma(b) = (v, u), \qquad \sigma(c) = (u, v),$$

where $a, b \in \mathbf{Var}$. Let K be any field. The polynomial $a + b \in K\langle a, b, c\rangle$ is not compatible with the signature Σ because no arithmetic operator statement can lead to this polynomial. The reason is that the variables a and b have different sorts, and thus, the expression $a + b$

is not a well-formed term in the signature Σ. An example of a compatible polynomial is $aba + c - a$, which arises, for example, as the translation of the arithmetic operator statement $a \cdot b \cdot a + c \approx a \in \mathbf{Form}(\Sigma)$.

If the objective is to find polynomials in I that are compatible with a given signature Σ, then $\rho(I)$ can be replaced by

$$\rho_\Sigma(I) = \{f \in \rho(I) \mid f \text{ compatible with } \Sigma\}\,.$$

The following corollary from [RRH21, Thm. 16] ensures under very mild assumptions on the generators of I that, when working with $\rho_\Sigma(I)$ instead of $\rho(I)$, we can still form all compatible polynomials in I.

Corollary 5.3.15. *Let Σ be a signature and let $f_1, \ldots, f_r, f \in K\langle X\rangle$ be compatible with Σ. Furthermore, let $I = (f_1, \ldots, f_r)$. Then,*

$$f \in I \iff f \in (\rho_\Sigma(I))_\rho\,.$$

Proof. Note that our notion of compatibility with a signature is equivalent to the notion of compatibility with a labelled quiver [RRH21, Def. 6]. With this, the result follows from [RRH21, Thm. 16]. □

Remark 5.3.16. *It has been observed that the noncommutative version of Buchberger's algorithm preserves compatibility of polynomials with a signature [HSW98; Che+20; SL20]. This means that, if all polynomials that are given as input to this algorithm are compatible with a signature, then so are all polynomials in the output. We note that this behaviour carries over to one-sided ideal intersections. More precisely, if all generators of two right ideals I_ρ and J_ρ are compatible with a signature, then so are all polynomials in a Gröbner basis of $I_\rho \cap J_\rho$ when Buchberger's algorithm is used in combination with Theorem 5.3.2. In case of two-sided ideal intersections, however, this is no longer true. A Gröbner basis of the intersection $I \cap J$ of two ideals I and J with compatible generators computed with Buchberger's algorithm might contain incompatible polynomials. The reason for this is the introduction of the commutator relations $tx_k - x_k t$ during the computation of $I \cap J$.*

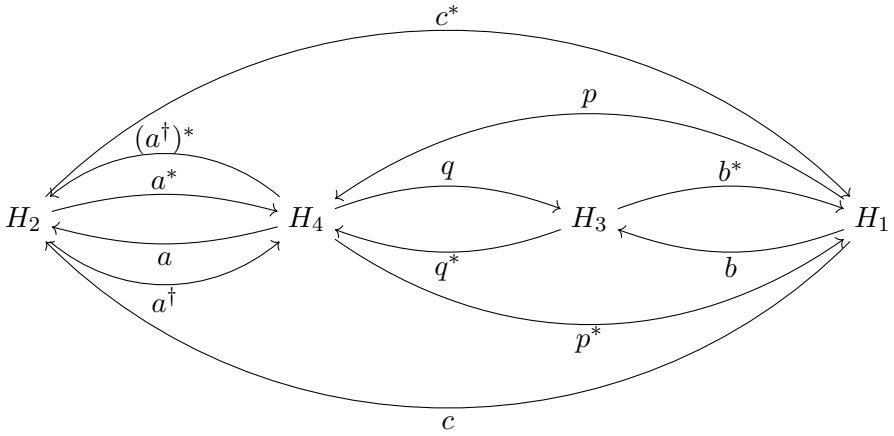


Figure 5.1: Sorts of the variables encoding Theorem 5.3.1.

Typically, the set $\rho_\Sigma(I)$ is a lot smaller than $\rho(I)$. To illustrate this point, we finish the proof of Theorem 5.3.1.

Example 5.3.17. *Recall that we had translated the assumptions of Theorem 5.3.1 into a set F of noncommutative polynomials. This set consists of the following elements:*

$$
\begin{array}{llll}
aa^\dagger a - a, & a^*(a^\dagger)^* a^* - a^*, & (a^\dagger)^* a^* - aa^\dagger, & \\
a^\dagger a a^\dagger - a^\dagger, & (a^\dagger)^* a^* (a^\dagger)^* - (a^\dagger)^*, & a^*(a^\dagger)^* - a^\dagger a, & \\
c - ap, & c^* - p^* a^*, & c^*(a^\dagger)^* - b^* q, & a^\dagger c - q^* b,
\end{array}
$$

where the first two lines correspond to the existence of the Moore-Penrose inverse $A^\dagger$ (plus the respective adjoint statements) and the last line corresponds to the range inclusions $\mathcal{R}(C) \subseteq \mathcal{R}(A)$ and $\mathcal{R}((A^\dagger C)^*) \subseteq \mathcal{R}(B^*)$ (plus the respective adjoint statements). The sorts of the involved symbols are depicted in Figure 5.1, using the symbols H_i to represent the spaces $\mathcal{H}_i$, $i = 1, \ldots, 4$.

The existence of a solution Y of $AYB = C$ corresponds to finding a compatible polynomial $ayb - c \in (F)$, where a, b, c are known but y is unknown. Using one-sided ideal intersection as discussed in this section, we can now find such a polynomial. To this end, we intersect $I = (F)$ with the right ideal $(a, c)_\rho$. Working over $K = \mathbb{Q}$ with a degree lexicographic

order, the reduced Gröbner basis of I consists of 17 polynomials. In this example, $\rho(I)$ as well as $\rho_\Sigma(I)$ seem to be infinite, where Σ is the signature used to encode the theorem. However, it turns out that it suffices to consider the finite approximation $\rho_\Sigma^{(5)}(I) \subseteq \rho_\Sigma(I)$ of compatible polynomials of degree at most 5 to find a suitable polynomial in the intersection. We note that $|\rho_\Sigma^{(5)}(I)| = 68$ while computing all elements in $\rho(I)$ up to degree 5 would yield 383 elements. Then the reduced Gröbner basis of $(\rho_\Sigma^{(5)}(I))_\rho \cap (a, c)_\rho$ contains the compatible polynomial $aq^*b - c$. Since the relevant ideal membership $aq^*b - c \in I$ also holds in $\mathbb{Z}\langle X, X^*\rangle$, this proves Theorem 5.3.1 and shows that Q^* is a solution of $AYB = C$.

5.3.2 Intersection with subalgebra

The second technique that we consider is the intersection of an ideal with a finitely generated subalgebra of $K\langle X\rangle$. To this end, we recall the following result from [Nor98], which is a combination of [Nor98, Thm. 3] and the remarks at the end of [Nor98, Sec. 4].

Theorem 5.3.18. *Let $I \trianglelefteq K\langle X\rangle$ be an ideal generated by $F \subseteq K\langle X\rangle$. Furthermore, let $Y = \{y_1, \ldots, y_m\}$ be a new set of indeterminates, and let $\bar{\varphi}\colon K\langle Y\rangle \to K\langle X\rangle/I$ be the K-algebra homomorphism defined by $\bar{\varphi}(y_i) = [h_i]$ for some $h_i \in K\langle X\rangle$, $i = 1, \ldots, m$. Consider the ideal generated by*

$$H = F \cup \{y_i - h_i \mid 1 \leq i \leq m\} \subseteq K\langle X, Y\rangle.$$

Then $\ker(\bar{\varphi}) = (H) \cap K\langle Y\rangle$.

Thus, by exploiting the elimination property of Gröbner bases (Theorem 2.4.43), we can obtain a Gröbner basis of the kernel of K-algebra homomorphisms. This result allows us to also study K-subalgebras, and in particular, to intersect ideals with such subalgebras. If $K\langle H\rangle$ is the K-subalgebra of $K\langle X\rangle$ generated by $H = \{h_1, \ldots, h_m\} \subseteq K\langle X\rangle$, that is,

$$K\langle H\rangle \coloneqq \{f(h_1, \ldots, h_m) \mid f \in K\langle y_1, \ldots, y_m\rangle\},$$

then $K\langle H\rangle$ can be considered as the image of the homomorphism $\varphi\colon K\langle Y\rangle \to K\langle X\rangle$ defined by $\varphi(y_i) = h_i$ for $i = 1, \ldots, m$. The following corollary tells us how to compute the intersection of $K\langle H\rangle$ with an ideal $I \trianglelefteq K\langle X\rangle$.

Corollary 5.3.19. *Let $H = \{h_1, \ldots, h_m\} \subseteq K\langle X\rangle$ and $I \trianglelefteq K\langle X\rangle$. Consider the K-algebra homomorphism*

$$\bar{\varphi}\colon K\langle Y\rangle \to K\langle X\rangle/I, \quad y_i \mapsto [h_i].$$

Then,

$$I \cap K\langle H\rangle = \{f(h_1, \ldots, f_m) \mid f \in \ker(\bar{\varphi})\}.$$

Proof. We denote $F = \{f(h_1, \ldots, f_m) \mid f \in \ker(\bar{\varphi})\}$.

To show the first inclusion $I \cap K\langle H\rangle \subseteq F$, let $g \in I \cap K\langle H\rangle$. Since $g \in K\langle H\rangle$, there exists $f \in K\langle Y\rangle$ such that $f(h_1, \ldots, h_m) = g$. Then $\bar{\varphi}(f) = [g] = [0]$, since $g \in I$. Thus, $f \in \ker(\bar{\varphi})$, and consequently, $g = f(h_1, \ldots, h_m) \in F$.

For the other inclusion $I \cap K\langle H\rangle \supseteq F$, let $g \in F$. By definition, there exists $f \in \ker(\bar{\varphi})$ such that $f(h_1, \ldots, h_m) = g$, showing that $g \in K\langle H\rangle$. To finish to proof, we note that $g \in I$ since $[g] = \bar{\varphi}(f) = [0]$. □

5.3.3 Homogeneous part of an ideal

The previous sections have shown that intersection techniques can provide useful tools for computing elements of special form in an ideal. However, not all properties of operators ultimately lead to statements about polynomials that can be solved by ideal intersections. One example of such a property is determining whether a given operator is positive. Recall that a linear operator P is called positive if there exists another linear operator Q such that P factors as $P = Q^*Q$. A concrete statement, where one has to determine the positivity of a given operator, is the following part of [AG10, Thm. 4.3].

Theorem 5.3.20. *Let $A\colon \mathcal{H}_3 \to \mathcal{H}_2$, $B\colon \mathcal{H}_1 \to \mathcal{H}_3$, $C\colon \mathcal{H}_1 \to \mathcal{H}_2$ be bounded linear operators on complex Hilbert spaces such that $\mathcal{R}(B) \subseteq \mathcal{R}(A^*)$. If there exists a bounded, positive linear operator $X\colon \mathcal{H}_3 \to \mathcal{H}_3$ such that $AXB = C$, then $B^*A^\dagger C$ is positive.*

Proving this statement using Theorem 5.1.7 leads to the following problem. Given the ideal (F) generated by the assumptions F of the theorem, find a polynomial of the form $p - q^*q \in (F)$, where p is known but the polynomial q, and therefore also q^*, are unknown.

Here, q^* denotes the image of q under the involutive antiautomorphism * that sends each variable x_i to an adjoint variable x_i^*, and each x_i^* back to x_i. Ideal intersections are not useful here since the second term q^*q is completely unknown. If q is a monomial, a polynomial of the desired form can be found by computing the *homogeneous part* of the ideal I.

In this section, we describe how the homogeneous part of an ideal can be computed. Our result is a generalisation of [Mil16], which describes how the set of all monomials contained in a commutative ideal can be computed. We note that a variant of the approach described in [Mil16] can also be found in [SST13, Alg. 4.4.2].

Recall from Example 2.2.36 that a matrix $A \in \mathbb{R}^{n\times m}$ with rows $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n \in \mathbb{R}^m$ defines a grading of the free algebra on $X = \{x_1, \ldots, x_n\}$ by the monoid $(\mathbb{R}^m, +)$ specified by the degree function $\deg_A$ mapping each word $w = x_{i_1} \ldots x_{i_d}$ to $\deg_A(w) = \sum_{j=1}^{d} \boldsymbol{a}_{i_j}$. Given such a grading of $K\langle X\rangle$ and an ideal $I \trianglelefteq K\langle X\rangle$, we can now ask what are the polynomials in I that are homogeneous with respect to $\deg_A$. More precisely, we want to consider the ideal generated by all these polynomials. Analogous to the commutative case (see, for example, [KR05, Tut. 50]), we call this ideal the *homogeneous part* of I with respect to $\deg_A$.

Definition 5.3.21. *Let $I \trianglelefteq K\langle X\rangle$ and $A \in \mathbb{R}^{n\times m}$. The* homogeneous part *of I with respect to $\deg_A$ is the ideal*

$$\hom_A(I) = (f \in I \mid f \text{ is homogeneous w.r.t. } \deg_A)\,.$$

The next lemma follows immediately from the definition above.

Lemma 5.3.22. *Let $I \trianglelefteq K\langle X\rangle$ and $A \in \mathbb{R}^{n\times m}$. Then $\hom_A(I)$ is the largest homogeneous ideal with respect to $\deg_A$ contained in I.*

In the following, we describe how a Gröbner basis of $\hom_A(I)$ can be computed. In our approach, we have to restrict ourselves to the case that A has only integer entries, that is, $A \in \mathbb{Z}^{n\times m}$. Note that this is equivalent to having $A \in \mathbb{Q}^{n\times m}$ as, for any nonzero $c \in \mathbb{Q}$, the matrix cA induces the same decomposition of $K\langle X\rangle$ as A.

For the following, some notation is needed. First, for two sets Y and Z, we denote by $[Y, Z] = \{yz - zy \mid y \in Y, z \in Z\}$ the set of commutator relations between Y and Z. Then, with two new sets of indeterminates $T = \{t_1, \ldots, t_m\}$ and $T^{-1} = \{t_1^{-1}, \ldots, t_m^{-1}\}$, we let $J \subseteq K\langle X, T, T^{-1}\rangle$ be the ideal generated by

$$[X \cup T \cup T^{-1}, T \cup T^{-1}] \cup \{1 - t_j t_j^{-1} \mid j = 1, \ldots, m\}.$$

Furthermore, we let $\mathcal{A} = K\langle X, T, T^{-1}\rangle / J$. The equivalence class of an element $f \in K\langle X, T, T^{-1}\rangle$ in $\mathcal{A}$ is $[f]$. Note that, in $\mathcal{A}$, the elements $[t_j]$ and $[t_j^{-1}]$ commute with each other and with everything else. Furthermore, they are also invertible.

Remark 5.3.23. *The quotient algebra $\mathcal{A}$ could be defined equivalently as the following quotient of a mixed algebra (Definition 3.1.2):*

$$\mathcal{A} = {K[T, T^{-1}]\langle X\rangle} \big/ {(1 - t_j t_j^{-1} \mid j = 1, \ldots, m)}.$$

We can think of the equivalence classes of monomials in the variables $T \cup T^{-1}$ as commutative monomials. Hence, there is a bijection τ between exponent vectors and suitable representatives of these equivalence classes. More precisely, for $\boldsymbol{a} = (a_1, \ldots, a_m) \in \mathbb{Z}^m$ we define

$$\tau(\boldsymbol{a}) = s_1^{|a_1|} \ldots s_m^{|a_m|} \in \langle T, T^{-1}\rangle, \quad \text{where } s_j = \begin{cases} t_j & \text{if } a_j \geq 0 \\ t_j^{-1} & \text{otherwise} \end{cases}.$$

Example 5.3.24. *For $\boldsymbol{a} = (3, -2, 0, 1) \in \mathbb{Z}^4$, we have $\tau(\boldsymbol{a}) = t_1^3 (t_2^{-1})^2 t_4$.*

The map τ turns a vector of integers into a monomial in $\langle T, T^{-1}\rangle$. The following lemma captures an important property of this map in $\mathcal{A}$. It follows from the fact that the equivalence classes $[t_j]$ commute with each other and with their designated inverses.

Lemma 5.3.25. *For $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}^m$, we have $[\tau(\boldsymbol{a} + \boldsymbol{b})] = [\tau(\boldsymbol{a})] \cdot [\tau(\boldsymbol{b})]$.*

For a matrix $A \in \mathbb{Z}^{n\times m}$ with rows $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n$, we can use the map τ to define the K-algebra homomorphism

$$\varphi_A \colon K\langle X\rangle \to K\langle X, T, T^{-1}\rangle, \quad x_i \mapsto x_i\tau(\boldsymbol{a}_i).$$

Furthermore, we let $\overline{\varphi}_A \colon K\langle X\rangle \to \mathcal{A}$ be the composition of φ_A with the canonical projection that sends each element in $K\langle X, T, T^{-1}\rangle$ to its equivalence class in $\mathcal{A}$. Applying $\overline{\varphi}_A$ to a polynomial f maps each monomial $w \in \operatorname{supp}(f)$ to $[ww_t]$, where $w_t = \tau(\deg_A(w)) \in \langle T, T^{-1}\rangle$ is a monomial in the variables $T \cup T^{-1}$ encoding $\deg_A(w)$. Finally, we need the *extension* of an ideal by $\overline{\varphi}_A$.

Definition 5.3.26. *Let $I \trianglelefteq K\langle X\rangle$ be an ideal with generating set F. The* extension *of I along the homomorphism $\overline{\varphi}_A$ is the ideal*

$$I^{\overline{\varphi}_A} = (\overline{\varphi}_A(f) \mid f \in F) \trianglelefteq \mathcal{A}.$$

The following result ensures that the extension is independent of the concrete generating set, and thus, well-defined.

Lemma 5.3.27. *For $F, G \subseteq K\langle X\rangle$ such that $(F) = (G)$, we have*

$$(\overline{\varphi}_A(f) \mid f \in F) = (\overline{\varphi}_A(g) \mid g \in G)\,.$$

Proof. Denote $I_F = (\overline{\varphi}_A(f) \mid f \in F)$ and $I_G = (\overline{\varphi}_A(g) \mid g \in G)$. We show $\overline{\varphi}_A(f) \in I_G$ for all $f \in F$, which implies that $I_F \subseteq I_G$. The other inclusion will follow symmetrically. Let $f \in F$. By assumption, f is a linear combination of the elements in G, that is, $f = \sum_{i=1}^{d} p_i g_i q_i$ with $p_i, q_i \in K\langle X\rangle$ and $g_i \in G$. With this, we get

$$\overline{\varphi}_A(f) = \sum_{i=1}^{d} \overline{\varphi}_A(p_i)\overline{\varphi}_A(g_i)\overline{\varphi}_A(q_i) \in I_G,$$

using the fact that $\overline{\varphi}_A$ is a K-algebra homomorphism. □

We can finally state the main result of this section.

Theorem 5.3.28. *For $I \trianglelefteq K\langle X\rangle$ and $A \in \mathbb{Z}^{n\times m}$, we have*

$$\hom_A(I) = I^{\overline{\varphi}_A} \cap K\langle X\rangle \coloneqq \bigcup_{[f]\in I^{\overline{\varphi}_A}} [f] \cap K\langle X\rangle.$$

Proof. We note that it follows from the definition of $I^{\overline{\varphi}_A}$ that $f \in I$ implies $\overline{\varphi}_A(f) \in I^{\overline{\varphi}_A}$. Now, to prove the inclusion $\hom_A(I) \subseteq I^{\overline{\varphi}_A} \cap K\langle X\rangle$, let $f \in \hom_A(I)$. Without loss of generality, we can assume that f is homogeneous. Let $\boldsymbol{a} = \deg_A(f) \in \mathbb{Z}^m$. Note that, since f is homogeneous, every term in f has degree $\boldsymbol{a}$. Due to this fact, in $\mathcal{A}$, we have $[\tau(\boldsymbol{a})f] = [\varphi_A(f)] = \overline{\varphi}_A(f) \in I^{\overline{\varphi}_A}$. To see that this implies that also $[f] \in I^{\overline{\varphi}_A}$, we use Lemma 5.3.25 and compute

$$[f] = [\tau(\mathbf{0})] \cdot [f] = [\tau(-\boldsymbol{a})] \cdot [\tau(\boldsymbol{a})] \cdot [f] = [\tau(-\boldsymbol{a})] \cdot [\tau(\boldsymbol{a})f] \in I^{\overline{\varphi}_A}.$$

For the other inclusion $I^{\overline{\varphi}_A} \cap K\langle X\rangle \subseteq \hom_A(I)$, we show that $I^{\overline{\varphi}_A} \cap K\langle X\rangle \subseteq I$ and that $I^{\overline{\varphi}_A} \cap K\langle X\rangle$ is a homogeneous ideal. Then the claim follows from Lemma 5.3.22. For the first part, let $f \in I^{\overline{\varphi}_A} \cap K\langle X\rangle$. Then f can be written as

$$f = \sum_{i=1}^{d} p_i \varphi_A(f_i) q_i + \sum_{i=1}^{e} u_i g_i v_i,$$

with $f_i \in I$, $g_i \in [X \cup T \cup T^{-1}, T \cup T^{-1}] \cup \{1 - t_j t_j^{-1} \mid j = 1, \dots, m\}$ and $p_i, q_i, u_i, v_i \in K\langle X, T, T^{-1}\rangle$. Note that the left-hand side of this identity does not depend on the indeterminates in T and T^{-1}. Hence, by setting $t_j = t_j^{-1} = 1$ for all $j = 1, \dots, m$, we obtain $f = \sum_{i=1}^{d} \tilde{p}_i f_i \tilde{q}_i$ with $\tilde{p}_i, \tilde{q}_i \in K\langle X\rangle$. This shows that $f \in I$.

Finally, to show that $I^{\overline{\varphi}_A} \cap K\langle X\rangle$ is a homogeneous ideal, we extend the $\mathbb{Z}^m$-grading of $K\langle X\rangle$ defined by A to a $\mathbb{Z}^m$-grading of $K\langle X, T, T^{-1}\rangle$ by setting

$$\deg_A(x_i) = \boldsymbol{a}_i, \quad \deg_A(t_j) = -\boldsymbol{e}_j, \quad \deg_A(t_j^{-1}) = \boldsymbol{e}_j,$$

where $\boldsymbol{a}_i$ denotes the ith row of A and $\boldsymbol{e}_j$ the jth unit vector of $\mathbb{Z}^m$ for $i = 1, \dots, n$, $j = 1, \dots, m$. The ideal J is homogeneous with respect to this grading. Consequently, this induces a well-defined grading on $\mathcal{A}$ with respect to which the ideal $I^{\overline{\varphi}_A}$ is homogeneous,

because $\overline{\varphi}_A(f)$ has degree 0 for every $f \in K\langle X\rangle$. To see that also the intersection $I^{\overline{\varphi}_A} \cap K\langle X\rangle$ is homogeneous, let $f \in I^{\overline{\varphi}_A} \cap K\langle X\rangle$. Write $f = h_1 + \ldots h_d$ with homogeneous components $h_i \in K\langle X\rangle$, $i = 1, \ldots, d$ and note that the homogeneous components of $[f]$ are $[h_1], \ldots, [h_d]$. Thus, since $I^{\overline{\varphi}_A}$ is homogeneous, we have $[h_i] \in I^{\overline{\varphi}_A}$ for $i = 1, \ldots, d$. As the h_i do not depend on T or T^{-1}, they are also contained in $I^{\overline{\varphi}_A} \cap K\langle X\rangle$. This shows that the intersection is also a homogeneous ideal. □

To turn Theorem 5.3.28 into an effective procedure, we can make use of the fact that there is a bijection between ideals in $K\langle X, T, T^{-1}\rangle/J$ and ideals in $K\langle X, T, T^{-1}\rangle$ containing J. Using this, we can obtain a Gröbner basis of $\hom_A(I)$ as described in Algorithm 9.

Algorithm 9: Gröbner basis enumeration of homogeneous part

Input: $f_1, \ldots, f_r \in K\langle X\rangle$ generating $I = (f_1, \ldots, f_r)$, a matrix $A \in \mathbb{Z}^{n\times m}$
Output (if the algorithm terminates): $G \subseteq I$ a Gröbner basis of $\hom_A(I)$

1 $T \leftarrow \{t_1, \ldots, t_m\}$, $T^{-1} \leftarrow \{t_1^{-1}, \ldots, t_m^{-1}\}$;
2 $J \leftarrow$ the ideal in $K\langle X, T, T^{-1}\rangle$ generated by

$$[X \cup T \cup T^{-1}, T \cup T^{-1}] \cup \{1 - t_j t_j^{-1} \mid j = 1, \ldots, m\};$$

3 $H \leftarrow (\varphi_A(f_1), \ldots, \varphi_A(f_r)) \subseteq K\langle X, T, T^{-1}\rangle$;
4 enumerate a Gröbner basis $g_0, g_1, \ldots$ of $H + J$ w.r.t. an elimination order for $T \cup T^{-1}$;
5 $G \leftarrow \{g_n \mid n \in \mathbb{N}, g_n \in K\langle X\rangle\}$;
6 **return** G;

Remark 5.3.29. *If $A \in \mathbb{N}^{n\times m}$, then the ideal J can be replaced by*

$$J' = \left([X \cup T, T] \cup \{1 - t_j t_j^{-1} \mid j = 1, \ldots, m\}\right).$$

Note that, in this case, we do not have any commutator relations involving the variables in T^{-1}. This is because, if $A \in \mathbb{N}^{n\times m}$, the generators of $I^{\overline{\varphi}_A}$ do not contain any indeterminates from T^{-1}. Thus, to cancel any indeterminates from T from these generators, it is enough to move them to the right, exploiting the fact that these variables still commute with each other and with all $x \in X$, and then cancel them using the relations $1 - t_j t_j^{-1}$, $j = 1, \ldots, m$.

We fix $X = \{x_1, \ldots, x_n\}$ and $X^* = \{x_1^*, \ldots, x_n^*\}$. To end this section, we discuss how computing the homogeneous part of an ideal $I \subseteq K\langle X, X^*\rangle$ allows to find elements of

the form $p - q^*q \in I$, where $p \in K\langle X, X^*\rangle$ is given and q is unknown. In case that q is a monomial, we can choose a grading of $K\langle X, X^*\rangle$ that makes elements of the form $p - q^*q$ homogeneous. Taking the matrix A such that $\deg_A(x_i) = \boldsymbol{e}_i$ and $\deg_A(x_i^*) = -\boldsymbol{e}_i$, for $i = 1, \ldots, n$, where $\boldsymbol{e}_i$ denotes the ith unit vector of $\mathbb{Z}^n$, yields $\deg_A(q^*q) = \mathbf{0}$ for all $q \in \langle X, X^*\rangle$. Now, if there exists an element of the form $p - q^*q \in I$, and if we introduce a new variable v with $\deg_A(v) = \mathbf{0}$ and set $I' = I + (v - p)$, then the homogeneous polynomial $v - q^*q$ of degree $\mathbf{0}$ lies in I'. By enumerating generators of $\hom_A(I')$ we can systematically search for this element. We note that we can increase our chances of finding a suitable element by computing a Gröbner basis of $\hom_A(I')$ with respect to an elimination order for $\{v\}$. Because then, provided that I contains an element of the form $p - q^*q$, this Gröbner basis must contain an element with leading monomial v. In the following, we apply this procedure to prove Theorem 5.3.20.

Example 5.3.30. *To prove Theorem 5.3.20 by a computation with polynomials, we have to translate all properties of the operators first into identities and then into noncommutative polynomials. For the assumptions, the existence of a positive solution X and the range inclusion translate into the identities $AY^*YB = C$ and $B = A^*Z$, respectively, with new operators Y, Z. Furthermore, the existence of the Moore-Penrose inverse $A^\dagger$ is encoded by the four defining identities of $A^\dagger$. The sorts of the symbols encoding the domains and codomains of the operators are depicted in Figure 5.2, using the symbols H_i to represent the spaces $\mathcal{H}_i$, $i = 1, 2, 3$.*

Translating all these identities and the respective adjoint statements into noncommutative polynomials, gives a set $F \subseteq \mathbb{Q}\langle X, X^*\rangle$ of 10 polynomials with integer coefficients in 12 indeterminates. Proving Theorem 5.3.20 boils down to finding an operator Q such that $B^*A^\dagger C = Q^*Q$, or in terms of polynomials, to finding a polynomial of the form $b^*a^\dagger c - q^*q \in (F)$, where $b^*, a^\dagger, c \in X \cup X^*$ are known but $q \in \mathbb{Q}\langle X, X^*\rangle$, and therefore also q^*, are unknown.

Following the procedure outlined above, we consider the ideal $I' = (F) + (v - b^*a^\dagger c) \subseteq \mathbb{Q}\langle X, X^*, v\rangle$ and enumerate a Gröbner basis of the homogeneous part $\hom_A(I')$ with respect to the degree matrix $A \in \mathbb{Z}^{13\times 6}$ such that $\deg_A(v) = \mathbf{0}$, $\deg_A(x_i) = \boldsymbol{e}_i$, $\deg_A(x_i^*) = -\boldsymbol{e}_i$ for all $x_i \in X$, $x_i^* \in X^*$. We do this computation with respect to an elimination order for $\{v\}$. To speed up the computation, we first compute the reduced Gröbner basis G of I' and then use this generating set as input to enumerate a Gröbner basis of $\hom_A(I')$. While

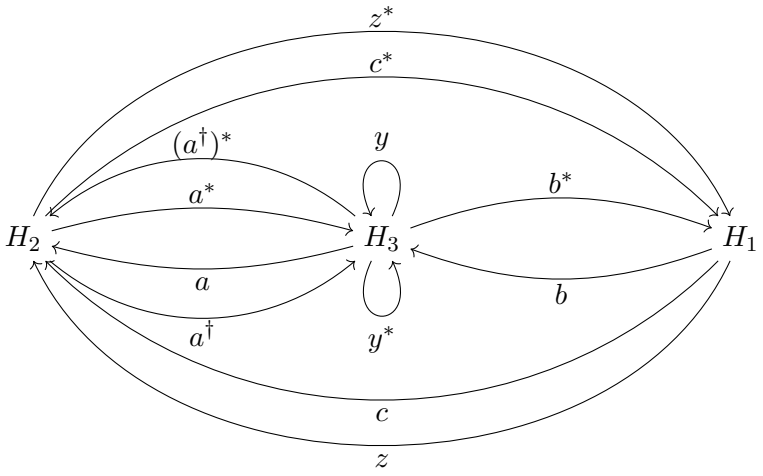


Figure 5.2: Sorts of the variables encoding Theorem 5.3.20.

$|G| = 25$*, the Gröbner basis of* $\hom_A(I')$ *seems to be infinite. However, after enumerating 273 generators of* $\hom_A(I')$*, one can see that* $v - b^*y^*yb = v - (yb)^*yb \in \hom_A(I')$*, which shows that* $b^*a^\dagger c - (yb)^*yb \in (F)$*. After verifying that this polynomial is compatible with the signature* Σ *and that the ideal membership also holds in* $\mathbb{Z}\langle X, X^*\rangle$*, this reveals that* $B^*A^\dagger C = (YB)^*YB$ *is indeed positive, and consequently proves Theorem 5.3.20.*

5.3.4 Monomial part of a right ideal

Other elements in an ideal that are often of special interest are monomials. The goal of this section is to effectively describe the *monomial part* of a (right) ideal I, that is, the (right) ideal generated by all monomials contained in I. More precisely, we show how a simple adaptation of the technique described in the previous section allows to compute all monomials in a given right ideal. We note that the following definition is analogous to the commutative case (see, for example, the discussion before [SST13, Alg. 4.4.2]).

Definition 5.3.31. *Let* $I \trianglelefteq K\langle X\rangle$ *be an ideal. The* monomial part *of* I *is the ideal* $\mathrm{mon}(I) = (m \in I \mid m \in \langle X\rangle)$*. Analogously, for a right ideal* $I_\rho \trianglelefteq_r K\langle X\rangle$*, the* monomial part *of* I_ρ *is the right ideal* $\mathrm{mon}(I_\rho) = (m \in I_\rho \mid m \in \langle X\rangle)_\rho$*.*

This is clearly a monomial (right) ideal as defined below.

Definition 5.3.32. *A (right) ideal $I \subseteq K\langle X\rangle$ is called a* monomial (right) ideal *if I has a (right) generating set consisting only of monomials.*

A classical characterisation of monomial ideals is given by the following result.

Lemma 5.3.33. *A (right) ideal $I \subseteq K\langle X\rangle$ is a monomial (right) ideal if and only if, with every element $f \in I$, also* $\operatorname{supp}(f) \subseteq I$.

The next result follows directly from the definition above.

Lemma 5.3.34. *For a (right) ideal $I \subseteq K\langle X\rangle$,* $\operatorname{mon}(I)$ *is the largest monomial (right) ideal contained in I.*

In the following, we describe how an adaptation of the technique from the previous section allows to compute a right Gröbner basis of $\operatorname{mon}(I_\rho)$ for a given right ideal $I_\rho \trianglelefteq_r K\langle X\rangle$. As for the homogeneous part of an ideal, we need some notation. First, we define the ideal

$$J = \Big([X,T] \cup \{1 - t_j t_j^{-1} \mid j = 1,\dots,n\}\Big) \trianglelefteq K\langle X, T, T^{-1}\rangle, \tag{5.2}$$

where $T = \{t_1,\dots,t_n\}$ and $T^{-1} = \{t_1^{-1},\dots,t_n^{-1}\}$ are new indeterminates. Then we consider the quotient algebra $\mathcal{B} = K\langle X, T, T^{-1}\rangle/J$. Recall that the equivalence class of an element $f \in K\langle X, T, T^{-1}\rangle$ in $\mathcal{B}$ is denoted by $[f]$. Note that, in $\mathcal{B}$, the $[t_j]$ commute with all $[x_i]$ but not with each other. The fact that the $[t_j]$ do not commute with each other is the main and crucial difference compared to the previous section. Furthermore, each $[t_j]$ can also be cancelled from the right.

Additionally, we define the K-algebra homomorphism

$$\varphi\colon K\langle X\rangle \to K\langle X, T, T^{-1}\rangle, \quad x_i \mapsto x_i t_i. \tag{5.3}$$

The map φ can be considered as a special case of the map φ_A defined in the previous section by setting A to be the $n \times n$ identity matrix. As before, $\overline{\varphi}$ denotes the composition of φ with the canonical projection onto $\mathcal{B}$. Applying $\overline{\varphi}$ to a polynomial f maps each

monomial $w \in \operatorname{supp}(f)$ to the equivalence class $[wt_w]$, where t_w is a copy of w but in the variables $t_1, \ldots, t_n$.

Next, for a right ideal $I_\rho \trianglelefteq_r K\langle X\rangle$ with right generating set F, we consider the extension of I_ρ along the homomorphism $\overline{\varphi}$, which we denote by

$$I_\rho^{\overline{\varphi}} = (\overline{\varphi}(f) \mid f \in F)_\rho \trianglelefteq_r \mathcal{B}.$$

Note that, as in the previous section, this definition is independent of the generating set F, and the proof is completely analogous. However, in contrast to before, $I_\rho^{\overline{\varphi}}$ is now a right ideal. The main result of this section is the following theorem.

Theorem 5.3.35. *Let $I_\rho \trianglelefteq_r K\langle X\rangle$ be a right ideal. Furthermore, let $I_\rho^{\overline{\varphi}}$ be the extension of I_ρ along $\overline{\varphi}$. Then,*

$$\operatorname{mon}(I_\rho) = I_\rho^{\overline{\varphi}} \cap K\langle X\rangle \coloneqq \bigcup_{[f] \in I_\rho^{\overline{\varphi}}} [f] \cap K\langle X\rangle.$$

Before we proceed to prove this theorem, we describe in Algorithm 10 how a right Gröbner basis of $\operatorname{mon}(I_\rho)$ can be obtained.

Algorithm 10: Gröbner basis enumeration of monomial part

Input: a right generating set $F \subseteq K\langle X\rangle$ of a right ideal I_ρ
Output (if the algorithm terminates): $G \subseteq I_\rho$ a right Gröbner basis of $\operatorname{mon}(I_\rho)$

1 $T \leftarrow \{t_1, \ldots, t_n\}$, $T^{-1} \leftarrow \{t_1^{-1}, \ldots, t_n^{-1}\}$;
2 $J \leftarrow$ the ideal in $K\langle X, T, T^{-1}, y\rangle$ generated by

$$[X, T] \cup \{1 - t_j t_j^{-1} \mid j = 1, \ldots, n\};$$

3 $H \leftarrow (y \cdot \varphi(f) \mid f \in F) \subseteq K\langle X, T, T^{-1}, y\rangle$;
4 enumerate a Gröbner basis $g_0, g_1, \ldots$ of $H + J$ w.r.t. an elimination order for $T \cup T^{-1}$;
5 $G' \leftarrow \{g \mid yg = g_n \text{ for some } n \in \mathbb{N}\}$;
6 $G \leftarrow G' \cap K\langle X\rangle$;
7 **return** G;

Remark 5.3.36. *Algorithm 10 requires the computation of a right Gröbner basis of the right ideal $I_\rho^{\overline{\varphi}}$ in the quotient algebra $\mathcal{B} = K\langle X, T, T^{-1}\rangle / J$. As described in [Hey01], this*

can be done effectively by introducing a new tag variable y and computing the Gröbner basis of a two-sided ideal containing J in the free algebra $K\langle X, T, T^{-1}, y\rangle$. This is what is done in lines 3 – 5 of Algorithm 10. We note that there are also other ways to compute a Gröbner basis of a one-sided ideal in a quotient algebra (see, for example, [Xiu12, Sec. 6.1.2]), however the approach described above has the advantage that it can be realised with a standard two-sided Gröbner basis implementation without any additional adaptations.

The proof of the commutative analogue of Theorem 5.3.35 in [Mil16] relies on the fact that a certain ideal is homogeneous with respect to a certain matrix grading and that an ideal can only be homogeneous with respect to this grading if it is a monomial ideal. Unfortunately, this argument does not immediately carry over to the case of noncommutative polynomials because no matter which matrix grading we choose, monomials that are permutations of each other will always have the same degree. For example, the right ideal $(x_1x_2 - x_2x_1)_\rho$ is homogeneous with respect to any matrix grading but it is clearly not a monomial ideal. In order to prove Theorem 5.3.35, we introduce the notion of a *separating pseudograding* for a subset $S \subseteq R$ of a ring R.

Definition 5.3.37. *Let R be a ring and let $(N, \cdot)$ be a monoid. Furthermore, let $S \subseteq R$. We call a decomposition*

$$R = \bigoplus_{\alpha \in N} S_\alpha,$$

of R into a direct sum of abelian groups a separating (right) N-pseudograding *for S if the following conditions hold:*

1. *$S_1 S_\alpha \subseteq S_\alpha$ for all $\alpha \in N$;*
2. *$|S \cap S_\alpha| \leq 1$ for all $\alpha \in N$;*

Intuitively speaking, a separating N-pseudograding for S allows us to separate the elements in S as they all have to lie in different subgroups S_α. As in the case of usual gradings, we omit the information about the monoid N in a separating N-pseudograding if N is clear from the context.

The following example shows that the multidegree yields a separating pseudograding for commutative monomials.

Example 5.3.38. *Let $R = K[X]$ be the usual polynomial ring in $X = \{x_1, \ldots, x_n\}$. Consider the monoid $(\mathbb{N}^n, +)$ and let $S = [X]$. For $\boldsymbol{a} = (a_1, \ldots, a_n) \in \mathbb{N}^n$, we denote $\boldsymbol{x}^{\boldsymbol{a}} = x_1^{a_1} \ldots x_n^{a_n} \in [X]$. The grading of R induced by the multidegree $\deg(\boldsymbol{x}^{\boldsymbol{a}}) = \boldsymbol{a}$ is a separating $\mathbb{N}^n$-pseudograding for S. This follows from the fact that, for all $\boldsymbol{a} = (a_1, \ldots, a_n) \in \mathbb{N}^n$, the set $S_{\boldsymbol{a}}$ is given by $S_{\boldsymbol{a}} = \{c\boldsymbol{x}^{\boldsymbol{a}} \mid c \in K \setminus \{0\}\}$.*

For the following definition and proposition, we fix a ring R and a monoid $(N, \cdot)$. Given a separating (N-)pseudograding for a set $S \subseteq R$, we can define *separable homogeneous* elements, or short *s-homogeneous* elements, and, based on this, *separable homogeneous* ideals, or short *s-homogeneous* ideals.

Definition 5.3.39. *Let $S \subseteq R$ and let $R = \bigoplus_{\alpha \in N} S_\alpha$ be a separating pseudograding for S. An element $r \in R$ is called* separable homogeneous*, or short* s-homogeneous*, if $r \in S_\alpha$ for some $\alpha \in N$. If $r \neq 0$, then we call α the* s-degree *of r, denoted by* $\operatorname{sdeg}(r) = \alpha$*. A (right) ideal $I \subseteq R$ is called* separable homogeneous*, or short* s-homogeneous*, if I is generated by s-homogeneous elements.*

The following proposition is the crucial observation about separating pseudogradings. It is a weaker version of the property that, with an element r, a homogeneous ideal also contains all homogeneous components of r. In the following, $1 \in N$ denotes the identity element of the monoid.

Proposition 5.3.40. *Let $S \subseteq R$ and $R = \bigoplus_{\alpha \in N} S_\alpha$ be a separating pseudograding for S. Furthermore, let $I_\rho \trianglelefteq_r R$ be an s-homogeneous right ideal generated by s-homogeneous elements of s-degree 1. If $r = c_1 s_1 + \cdots + c_d s_d \in I_\rho$ with $s_1, \ldots, s_d \in S$ pairwise different and $c_1, \ldots, c_d \in S_1$, then $c_1 s_1, \ldots, c_d s_d \in I_\rho$.*

Proof. Without loss of generality, we can assume that $c_i s_i \neq 0$ for all $i = 1, \ldots, d$. We note that it suffices to show that $c_1 s_1 \in I_\rho$, for then also $r - c_1 s_1 \in I_\rho$, and the statement follows by induction. We denote by $F \subseteq S_1$ a right generating set of I_ρ consisting of s-homogeneous elements of s-degree 1. Then, with $b_f \in R$ and $b_{f,\alpha} \in S_\alpha$ such that $b_f = \sum_{\alpha \in N} b_{f,\alpha}$, we can write r as

$$r = \sum_{f \in F} f b_f = \sum_{f \in F} f \sum_{\alpha \in N} b_{f,\alpha} = \sum_{f \in F} \sum_{\alpha \in N} f b_{f,\alpha}.$$

As all $f \in S_1$, it follows from the definition of a separating pseudograding that $fb_{f,\alpha} \in S_1 S_\alpha \subseteq S_\alpha$. Furthermore, by the definition of a separating pseudograding for S, there exist pairwise different $\alpha_1, \ldots, \alpha_d \in N$ such that $S \cap S_{\alpha_i} = \{s_i\}$ for all $i = 1, \ldots, d$. Then, also $c_i s_i \in S_1 S_{\alpha_i} \subseteq S_{\alpha_i}$ for all $i = 1, \ldots, d$. Since the subgroups S_{α_i} have trivial intersection with each other, the nonzero elements $c_2 s_2, \ldots, c_d s_d$ cannot lie in S_{α_1}. Therefore, by comparing s-degrees, one can see that $c_1 s_1 = \sum_{f \in F} fb_{f,\alpha_1} \in I_\rho$. □

Coming back to noncommutative polynomials, our next goal is to define a separating pseudograding for $\langle X \rangle$ in $K\langle X, T, T^{-1} \rangle$. To this end, we denote by $\mathcal{F}_n$ the free group of rank n generated by $\gamma_1, \ldots, \gamma_n$. This allows us to consider the two monoid homomorphisms

$$\begin{aligned} &\lambda\colon \langle X, T, T^{-1} \rangle \to \mathcal{F}_n, && x_i \mapsto \gamma_i, && t_i \mapsto 1, && t_i^{-1} \mapsto 1, \\ &\mu\colon \langle X, T, T^{-1} \rangle \to \mathcal{F}_n, && x_i \mapsto 1, && t_i \mapsto \gamma_i, && t_i^{-1} \mapsto \gamma_i^{-1}. \end{aligned}$$

Using λ and μ, we define the map $\operatorname{sdeg}\colon \langle X, T, T^{-1} \rangle \to \mathcal{F}_n$ that sends each $w \in \langle X, T, T^{-1} \rangle$ to $\operatorname{sdeg}(w) \coloneqq \mu(w)^{-1}\lambda(w) \in \mathcal{F}_n$.

Example 5.3.41. *Let $X = \{x_1, x_2\}$, $T = \{t_1, t_2\}$, and $T^{-1} = \{t_1^{-1}, t_2^{-1}\}$. For $w = x_1 t_1 x_2 t_2 \in \langle X, T, T^{-1} \rangle$, we have* $\operatorname{sdeg}(w) = (\gamma_1\gamma_2)^{-1}\gamma_1\gamma_2 = 1$. *Analogously, taking $w' = t_1 t_2^{-1} x_1 t_1^{-1} x_2 \in \langle X, T, T^{-1} \rangle$, yields* $\operatorname{sdeg}(w') = (\gamma_1\gamma_2^{-1}\gamma_1^{-1})^{-1}\gamma_1\gamma_2 = \gamma_1\gamma_2\gamma_2$.

The following lemma captures some important properties of the map sdeg.

Lemma 5.3.42. *Let $w, w' \in \langle X, T, T^{-1} \rangle$, $x_i \in X$, $t_j \in T$ and $t_j^{-1} \in T^{-1}$. The following hold:*

1. $\operatorname{sdeg}(w x_i t_j w') = \operatorname{sdeg}(w t_j x_i w')$;
2. $\operatorname{sdeg}(w t_j t_j^{-1} w') = \operatorname{sdeg}(w w')$;
3. $\operatorname{sdeg}(w) = 1 \implies \operatorname{sdeg}(w w') = \operatorname{sdeg}(w')$;

Proof. The properties 1 and 2 follow immediately from the definition. To prove the third property, we assume $\operatorname{sdeg}(w) = 1$. This yields

$$\begin{aligned} \operatorname{sdeg}(ww') &= \mu(ww')^{-1}\lambda(ww') = \mu(w')^{-1}\mu(w)^{-1}\lambda(w)\lambda(w') \\ &= \mu(w')^{-1}\operatorname{sdeg}(w)\lambda(w') = \mu(w')^{-1}\lambda(w') = \operatorname{sdeg}(w'). \end{aligned}$$

□

The map sdeg allows us to decompose $K\langle X, T, T^{-1}\rangle$ into a direct sum of the abelian subgroups

$$K\langle X, T, T^{-1}\rangle_\alpha = \left\{ \sum_{w \in \langle X, T, T^{-1}\rangle} c_w w \;\middle|\; c_w \in K,\ \mathrm{sdeg}(w) = \alpha \text{ if } c_w \neq 0 \right\},$$

for $\alpha \in \mathcal{F}_n$. In particular, the following proposition tells us that this decomposition forms a separating $\mathcal{F}_n$-pseudograding for $\langle X\rangle$, and hence, justifies the name sdeg.

Proposition 5.3.43. *The abelian subgroups $K\langle X, T, T^{-1}\rangle_\alpha$ with $\alpha \in \mathcal{F}_n$ as defined above form a separating $\mathcal{F}_n$-pseudograding for $\langle X\rangle$.*

Proof. It is clear that the subgroups $K\langle X, T, T^{-1}\rangle_\alpha$ decompose $K\langle X, T, T^{-1}\rangle$ into a direct sum. Furthermore, the first condition of Definition 5.3.37 follows immediately from the third part of Lemma 5.3.42. Finally, we note that sdeg restricted to $\langle X\rangle$ is injective, that is, $\mathrm{sdeg}(w) \neq \mathrm{sdeg}(w')$ for all $w \neq w'$, $w, w' \in \langle X\rangle$. This implies the second property of a separating $\mathcal{F}_n$-pseudograding. □

The generators of the ideal J from (5.2) induce an equivalence relation on the monoid $\langle X, T, T^{-1}\rangle$ under which the map sdeg is invariant, that is, if $w, w' \in \langle X, T, T^{-1}\rangle$ are such that $w - w' \in J$, then $\mathrm{sdeg}(w) = \mathrm{sdeg}(w')$. This follows from the first two properties in Lemma 5.3.42. Hence, we can extend sdeg to the quotient algebra $\mathcal{B}$ by setting $\mathrm{sdeg}([w]) = \mathrm{sdeg}(w)$ for all $w \in \langle X, T, T^{-1}\rangle$. Consequently, the separating $\mathcal{F}_n$-pseudograding for $\langle X\rangle$ in $K\langle X, T, T^{-1}\rangle$ can be extended in a straightforward way to give a separating $\mathcal{F}_n$-pseudograding for $\langle \overline{X}\rangle = \{[x] \mid x \in \langle X\rangle\}$ in $\mathcal{B}$.

Finally, we have all tools available to prove Theorem 5.3.35. We split this proof into two lemmas.

Lemma 5.3.44. *Let $I_\rho \trianglelefteq_r K\langle X\rangle$ be a right ideal. Furthermore, let $I_\rho^{\overline{\varphi}}$ be the extension of I_ρ along $\overline{\varphi}$. Then,*

$$\mathrm{mon}(I_\rho) \;\subseteq\; I_\rho^{\overline{\varphi}} \cap K\langle X\rangle \;\subseteq\; I_\rho.$$

Proof. It follows from the definition of $I_\rho^{\overline{\varphi}}$ that $f \in I_\rho$ implies $\overline{\varphi}(f) \in I_\rho^{\overline{\varphi}}$. Now, to prove the first inclusion $\mathrm{mon}(I_\rho) \subseteq I_\rho^{\overline{\varphi}} \cap K\langle X\rangle$, let $w = x_{i_1} \dots x_{i_d} \in \mathrm{mon}(I_\rho)$. Then we get $[wt_{i_1} \dots t_{i_d}] = \overline{\varphi}(w) \in I_\rho^{\overline{\varphi}}$, which implies

$$[w] = [w] \cdot [t_{i_1} \dots t_{i_d} t_{i_d}^{-1} \dots t_{i_1}^{-1}] = [wt_{i_1} \dots t_{i_d}] \cdot [t_{i_d}^{-1} \dots t_{i_1}^{-1}] \in I_\rho^{\overline{\varphi}},$$

and consequently $w \in I_\rho^{\overline{\varphi}} \cap K\langle X\rangle$.

For the second inclusion $I_\rho^{\overline{\varphi}} \cap K\langle X\rangle \subseteq I_\rho$, let $f \in I_\rho^{\overline{\varphi}} \cap K\langle X\rangle$. Then f can be written as

$$f = \sum_{i=1}^{d} \varphi(f_i) q_i + \sum_{i=1}^{e} g_i v_i$$

with $f_i \in I_\rho$, $g_i \in [X, T] \cup \{1 - t_j t_j^{-1} \mid j = 1, \dots, n\}$ and $q_i, v_i \in K\langle X, T, T^{-1}\rangle$. Note that the left-hand side of this identity does not depend on the indeterminates in T and T^{-1}. Hence, by setting $t_j = t_j^{-1} = 1$ for all $j = 1, \dots, n$, we obtain $f = \sum_{i=1}^{d} f_i \tilde{q}_i \in I_\rho$ with some $\tilde{q}_i \in K\langle X\rangle$. □

Lemma 5.3.45. *Let $I_\rho \trianglelefteq_r K\langle X\rangle$ be a right ideal. Furthermore, let $I_\rho^{\overline{\varphi}}$ be the extension of I_ρ along $\overline{\varphi}$. Then $I_\rho^{\overline{\varphi}} \cap K\langle X\rangle$ is a monomial right ideal.*

Proof. Let $f = c_1 w_1 + \dots + c_d w_d \in I_\rho^{\overline{\varphi}} \cap K\langle X\rangle$ with nonzero $c_1, \dots, c_d \in K$ and pairwise different $w_1, \dots, w_d \in \langle X\rangle$. To prove that $I_\rho^{\overline{\varphi}} \cap K\langle X\rangle$ is a monomial right ideal, we show that $w_1, \dots, w_d \in I_\rho^{\overline{\varphi}} \cap K\langle X\rangle$, which is equivalent to $[w_1], \dots, [w_d] \in I_\rho^{\overline{\varphi}}$. To show this, we fix the $\mathcal{F}_n$-pseudograding for $\langle \overline{X}\rangle$ in $\mathcal{B}$ induced by the map sdeg and note that the image $\overline{\varphi}(g)$ of any nonzero $g \in K\langle X\rangle$ under the map $\overline{\varphi}$ defined in (5.3) is an s-homogeneous polynomial of s-degree 1. In particular, this means that $I_\rho^{\overline{\varphi}}$ is an s-homogeneous right ideal in $\mathcal{B}$ generated by s-homogeneous elements of s-degree 1. Since $\mathrm{sdeg}([c_i]) = 1$ and $[w_i] \in \langle \overline{X}\rangle$ for all $i = 1, \dots, d$, Proposition 5.3.40 is applicable and yields that $[c_1 w_1], \dots, [c_d w_d] \in I_\rho^{\overline{\varphi}}$. But then clearly also $[w_1], \dots, [w_d] \in I_\rho^{\overline{\varphi}}$. □

The proof of Theorem 5.3.35 now follows from Lemmas 5.3.34, 5.3.44, and 5.3.45.

Remark 5.3.46. *We make some remarks on Theorem 5.3.35.*

1. *Theorem 5.3.35 can also be adapted to work with left ideals. In this case, the ideal J with respect to which the quotient is taken, has to be changed to*

$$J' = \left([X,T] \cup \{1 - t_j^{-1}t_j \mid j = 1,\dots,n\}\right).$$

   Then, in the quotient $K\langle X,T,T^{-1}\rangle/J'$ each $[t_j]$ can be cancelled from the left instead of from the right. To then prove the theorem for left ideals, Definition 5.3.37 and the map sdeg *have to be adapted accordingly as well.*

2. *Unfortunately, this approach does not generalise to two-sided ideals, as witnessed by the following example.*

Example 5.3.47. *In this example, we show that the approach described in this section does not generalise to two-sided ideals in a straightforward way. To this end, we consider the ideal $I = (x_1 - x_2) \trianglelefteq K\langle x_1, x_2\rangle$ over an arbitrary field K. It is not hard to see that* $\mathrm{mon}(I) = \{0\}$. *However, if we consider the two-sided ideal $I^{\overline{\varphi}} = (\overline{\varphi}(x_1 - x_2)) = ([x_1t_1 - x_2t_2]) \trianglelefteq \mathcal{B}$, then $x_1x_2 - x_2x_1 \in I^{\overline{\varphi}} \cap K\langle x_1, x_2\rangle$ as the following computation shows:*

$$\begin{aligned} &[x_1t_1 - x_2t_2]\cdot[x_2t_1^{-1}] - [x_2]\cdot[x_1t_1 - x_2t_2]\cdot[t_1^{-1}] \\ &= [x_1x_2] - [x_2x_2t_2t_1^{-1}] - [x_2x_1] + [x_2x_2t_2t_1^{-1}] \\ &= [x_1x_2 - x_2x_1] \in I^{\overline{\varphi}}. \end{aligned}$$

To compute a generating set of $\mathrm{mon}(I)$ in case of a two-sided ideal $I \trianglelefteq K\langle X\rangle$, we were so far only able to prove the following rather restrictive result.

Proposition 5.3.48. *Let X, Y be two disjoint sets of indeterminates and let $I \trianglelefteq K\langle X,Y\rangle$. Furthermore, let $G \subseteq I$ be a Gröbner basis of I such that* $\mathrm{lm}(g) \in \langle X\rangle$ *and* $\mathrm{tail}(g) \in K\langle Y\rangle$ *for all $g \in G$. Then $G \cap K\langle X\rangle$ is a Gröbner basis of* $\mathrm{mon}(I)$.

To prove this proposition, we recall the following result about Gröbner bases of monomial ideals, which follows immediately from Theorem 2.4.37.

Lemma 5.3.49. *Any set of monomials $M \subseteq \langle X\rangle$ is a Gröbner basis of the ideal (M).*

Proof of Proposition 5.3.48. Denote $M = G \cap K\langle X\rangle$. Due to Lemma 5.3.49, it suffices to show that $(M) = \mathrm{mon}(I)$. The inclusion $(M) \subseteq \mathrm{mon}(I)$ is clear. For the other inclusion $\mathrm{mon}(I) \subseteq (M)$, let $w \in \mathrm{mon}(I)$ be a monomial. Since G is a Gröbner basis of I, there exist $g_1, \ldots, g_k \in G$ such that

$$w \to_{g_1} w_1 \to_{g_2} \cdots \to_{g_k} w_k = 0,$$

with $w_1, \ldots, w_k \in K\langle X, Y\rangle$. In particular, we have $w_1 = -a\mathrm{tail}(g_1)b$ for some $a, b \in \langle X\rangle$. Since $\mathrm{lm}(g_2) \in \langle X\rangle$ but $\mathrm{tail}(g_1) \in K\langle Y\rangle$, the second reduction can only act on a or b but not on $\mathrm{tail}(g_1)$. Without loss of generality, assume that the reduction with g_2 acts on a, so that $w_2 = a'\mathrm{tail}(g_2)a''\mathrm{tail}(g_1)b$. Analogously, since $\mathrm{lm}(g_3) \in \langle X\rangle$, the reduction with g_3 cannot act on $\mathrm{tail}(g_1)$ or $\mathrm{tail}(g_2)$, and as this also holds for all other reductions, w_k must be of the form

$$w_k = (-1)^k a_1\mathrm{tail}(g_{i_1})a_2\mathrm{tail}(g_{i_2})\ldots a_k\mathrm{tail}(g_{i_k})a_{k+1}$$

for some $a_1, \ldots, a_{k+1} \in \langle X\rangle$ and $\{i_1, \ldots, i_k\} = \{1, \ldots, k\}$. Note that all parts in any w_j that lie in $K\langle X\rangle$ are subwords of w. Furthermore, since the reductions with all g_i can only act on these parts, we get that $\mathrm{lm}(g_i)$ divides w for all $i = 1, \ldots, k$. Hence, in particular, $\mathrm{lm}(g_k)$ divides w. To finish the proof, we show that g_k is in fact a monomial and therefore contained in M. To see this, we note that w_k can only be zero if one of the factors in the product above is zero, and since none of the a_j can be zero, we must have $\mathrm{tail}(g_i) = 0$ for some $1 \leq i \leq k$. In particular, we know that $w_{k-1} \neq 0$, and therefore $\mathrm{tail}(g_k) = 0$. This means that g_k is a monomial, and so $g_k \in M$, which implies that $w \in (M)$. □

It remains open how to compute a generating set of $\mathrm{mon}(I)$ for an arbitrary (finitely generated) two-sided ideals in $K\langle X\rangle$.

5.4 Short proofs of ideal membership

Based on the theory developed in Chapter 4, universal truth of operator statements can be reduced to the verification of ideal membership in a free algebra. The corresponding cofactor representations that certify these ideal memberships can be considered as a proof of the operator statement. In general, cofactor representations are not unique and different

representations can differ drastically in their complexity. We could observe empirically that representations computed by Gröbner bases are often significantly longer than necessary.

In this section, we discuss the problem of finding *sparsest* cofactor representations of an ideal element, that is, representations with a minimal number of terms. We focus on the situation of noncommutative polynomials, as we are particularly interested in computing short proofs of operator statements. However, all techniques also apply analogously to commutative polynomials.

Although ideal membership in the free algebra is only semi-decidable, we show in Section 5.4.1 that the problem of computing cofactor representations with the number of terms bounded by $N \in \mathbb{N}$ is decidable, yet NP-complete. This yields a first (impractical) algorithm for computing sparsest cofactor representations (Algorithm 11).

In Section 5.4.2, we then describe how to obtain a practical algorithm for computing sparse (not necessarily sparsest) representations by making two simplifications:

1. We restrict the search space to a finite dimensional subspace by only considering cofactor representations with terms smaller than a designated bound.

2. We use the sum of the absolute values of the coefficients, that is, the ℓ_1-norm, of a representation as a complexity measure.

With these simplifications, we translate the problem of finding sparse cofactor representations into solving a linear programming problem. This leads to Algorithm 13, which computes, starting from any given cofactor representation, a minimal one with respect to the conditions 1 and 2.

We also show that the second simplification does in fact impose no restriction for a class of ideals that appears frequently when translating operator statements. In particular, we prove that, under certain assumptions satisfied by most examples studied in practice, Algorithm 13 computes a sparsest representation among all representations satisfying condition 1. Finally, we demonstrate the effectiveness of Algorithm 13 on several examples coming from actual operator statements in Section 5.4.3.

In the following, K is a field and X is a finite set of indeterminates. For the rest of this section, we fix a family of polynomials $(f_1, \ldots, f_r) \in K\langle X\rangle^r$ generating an ideal I, and we let $\Sigma = (K\langle X\rangle \otimes_K K\langle X\rangle)^{(\mathcal{E})}$ be the free $K\langle X\rangle$-bimodule on the set $\mathcal{E} = \{\varepsilon_1, \ldots, \varepsilon_r\}$ (see

Definition 2.2.45). Also, recall from Section 3.1.3 that $\overline{\cdot}\colon \Sigma \to I$ is the $K\langle X\rangle$-bimodule homomorphism that sends each basis element ε_i to $\overline{\varepsilon}_i = f_i$.

5.4.1 Decidability and complexity

Any cofactor representation of an ideal member $f \in I$ can be identified with an element $\alpha \in \Sigma$ such that $\overline{\alpha} = f$. The *weight* of such a representation is given by the ℓ_0-"norm" $\|\alpha\|_0 \coloneqq |\operatorname{supp}(\alpha)|$. Then, with the set $R(f) \coloneqq \{\alpha \in \Sigma \mid \overline{\alpha} = f\}$ of *(cofactor) representations* of f, a *sparsest* (cofactor) representation of f corresponds to a minimal element with respect to $\|\cdot\|_0$ in $R(f)$. We denote the set of all such minimal elements by $R_0(f)$. If $f \notin I$, we set $R(f) = R_0(f) = \emptyset$.

Remark 5.4.1. *As ideal membership in the free algebra is only semi-decidable, we can also not decide whether $R(f) = \emptyset$ or not.*

Remark 5.4.2. *The function $\|\cdot\|_0$ is not a norm as it is not homogeneous, but one can associate to it a metric called* Hamming distance.

In the following, we study the decidability and complexity of computing cofactor representations of bounded weight. To this end, we assume that the coefficient field K is computable, in the sense that the basic arithmetic operations as well as equality testing are effective. This means, in particular, that linear systems can be solved effectively using, for example, Gaussian elimination. With this, we consider the following problem.

Problem 5.4.3 (Sparse cofactor representation)**.**
Input: $f, f_1, \ldots, f_r \in K\langle X\rangle$, $N \in \mathbb{N}$
Output: a cofactor representation $\alpha \in R(f)$ with $\|\alpha\|_0 \leq N$ if one exists, otherwise False*.*

We show that Problem 5.4.3 is decidable, and we give an algorithm reducing it to Problem 5.4.4 below of finding sparse solutions of a linear system, formally also known as the Min-RVLS (MINimum Relevant Variables in Linear System) problem. This not only yields an algorithm for computing sparsest cofactor representations if $f \in I$, but it, in principle, also provides a semi-decision procedure for ideal membership. We focus on the first application here.

Problem 5.4.4 (Sparse solution of linear system [Min-RVLS])**.**
Input: $A \in K^{m\times n}, \boldsymbol{b} \in K^m$, $N \in \{0,\dots,n\}$
Output: a vector $\boldsymbol{y} \in K^n$ with $A\boldsymbol{y} = \boldsymbol{b}$ and $\|\boldsymbol{y}\|_0 \leq N$ if one exists, otherwise False*.*

Problem 5.4.4 arises in many areas [CDS01; CT05; Don06]. It is clearly decidable, for instance by looping over all N-subsets of $\{1,\dots,n\}$ for possible sets of nonzero coefficients of solutions. Furthermore, in many cases, most notably for $K = \mathbb{Q}$, it is known to be NP-hard, with the corresponding decision problem being NP-complete [GJ79, Problem MP5].

In order to reduce the sparse cofactor representation problem to linear algebra, we need to constrain the solutions to a finite dimensional vector space, which requires to bound the degree of a solution. The degree of elements in $R(f)$ can be arbitrarily large, but we can bound the degree of *minimal* representations. A cofactor representation $\alpha = \sum_{i=1}^{d} c_i a_i \varepsilon_{j_i} b_i \in R(f)$ is *minimal* if no sub-sum is a syzygy, that is, $\sum_{i\in J} c_i a_i f_{j_i} b_i \neq 0$ for all nonempty subsets $J \subseteq \{1,\dots,d\}$. To obtain the degree bound, we recall the notion of *(polynomial) rewriting* introduced in [RRH21, Def. 2].

Definition 5.4.5. *Let $f, g \in K\langle X\rangle$ and $a, b \in \langle X\rangle$ such that* $\operatorname{supp}(f) \cap \operatorname{supp}(agb) \neq \emptyset$*. For every $c \in K$, we say that f* can be rewritten to $f + cagb \in K\langle X\rangle$ by g*.*

Furthermore, we say that f can be rewritten to h by $G \subseteq K\langle X\rangle$ *if there are $h_0,\dots,h_d \in K\langle X\rangle$, $h_d = f$, $h_0 = h$ and $g_1,\dots,g_d \in G$ such that h_k can be rewritten to h_{k-1} by g_k for all $k = 1,\dots,d$.*

Rewriting can be considered as a weaker form of polynomial reduction, not requiring that a polynomial gets "simplified" by a rewriting step. Nevertheless, f can be rewritten to zero by $\{f_1,\dots,f_r\}$ if and only if $f \in I$, see [RRH21, Lem. 4]. More importantly, we can show that any minimal representation of f can be obtained by rewriting f to zero by $\{f_1,\dots,f_r\}$ and logging the rewriting steps.

Lemma 5.4.6. *Let $f \in (f_1,\dots,f_r)$ and $\alpha = \sum_{i=1}^{d} c_i a_i \varepsilon_{j_i} b_i \in R(f)$ be a minimal representation of f. Furthermore, for $k = 0,\dots,d$, let $h_k = \sum_{i=1}^{k} c_i a_i f_{j_i} b_i$. In particular, $h_d = f$ and $h_0 = 0$. Then, possibly after reordering the summands of α, h_k can be rewritten to h_{k-1} by $\{f_1,\dots,f_r\}$ for all $k = 1,\dots,d$.*

Proof. We perform induction on the weight d of a minimal representation of f. For $d = 0$ there is nothing to prove. Assume now that $d > 0$ and that the result is proven for polynomials with a minimal representation of weight $d-1$. Because α is a minimal representation, f cannot be 0. Since $f = \overline{\alpha} = \sum_{i=1}^{d} c_i a_i f_{j_i} b_i$, the support of f is contained in the union of the supports of the $a_i f_{j_i} b_i$, and there exists $1 \leq k \leq d$ such that $\operatorname{supp}(f) \cap \operatorname{supp}(a_k f_{j_k} b_k) \neq \emptyset$. Possibly after reordering the summands of α, we can assume $k = d$. So f can be rewritten to $h_{d-1} = f - c_d a_d f_{j_d} b_d$ using $f_{j_d} \in \{f_1, \ldots, f_r\}$. Furthermore, $h_{d-1} = \sum_{i=1}^{d-1} c_i a_i f_{j_i} b_i$ has a minimal representation of weight $d-1$, because adding one term results in a minimal representation of weight d for f. So, by induction hypothesis, this representation of h_{d-1} is (up to reordering of the summands) a sequence of rewritings by $\{f_1, \ldots, f_r\}$. □

Another crucial property of rewriting is that we can bound the degree of the output in terms of the degree of the input and the *degree difference* of the rewriter. The *degree difference* $\operatorname{degdiff}(g)$ of a nonzero $g \in K\langle X \rangle$ is $\operatorname{degdiff}(g) = \deg(g) - \operatorname{degmin}(g)$, where $\operatorname{degmin}(g) = \min_{w \in \operatorname{supp}(g)} |w|$.

Lemma 5.4.7. *Let $f, g \in K\langle X \rangle$ and $c \in K$, $a, b \in \langle X \rangle$. If f can be rewritten to $h = f + cagb$ by g, then $\max\{\deg(h), \deg(agb)\} \leq \deg(f) + \operatorname{degdiff}(g)$.*

Proof. By definition, there exists a monomial w in $\operatorname{supp}(g)$ such that $awb \in \operatorname{supp}(f)$, so $|awb| \leq \deg(f)$, or equivalently $|a| + |b| \leq \deg(f) - |w|$. Since $w \in \operatorname{supp}(g)$, $|w| \geq \operatorname{degmin}(g)$, and all in all,

$$\begin{aligned} \deg(agb) &= |a| + |b| + \deg(g) \\ &\leq \deg(f) - \operatorname{degmin}(g) + \deg(g) = \deg(f) + \operatorname{degdiff}(g). \end{aligned}$$

As $\deg(h) \leq \max\{\deg(f), \deg(agb)\}$ and $\operatorname{degdiff}(g) \geq 0$, we conclude that also $\deg(h) \leq \deg(f) + \operatorname{degdiff}(g)$. □

Combining Lemma 5.4.6 and 5.4.7, we obtain a bound on the degree of minimal cofactor representations of f of bounded weight. Since any sparsest cofactor representation is, in particular, minimal, this also yields a bound on the degree of sparsest representations. In the following, for a nonzero $\alpha = \sum_{i=1}^{d} c_i a_i \varepsilon_{j_i} b_i \in \Sigma$ with nonzero $c_i \in K$ and pairwise different $a_i \varepsilon_{j_i} b_i$, we define its *(weighted) degree* to be $\deg(\alpha) = \max_i \deg(a_i f_{j_i} b_i)$.

Corollary 5.4.8. *Let $f \in (f_1, \dots, f_r)$, $N \in \mathbb{N}$ and $\alpha \in R(f)$ be a minimal representation of f. If $\|\alpha\|_0 \leq N$, then* $\deg(\alpha) \leq \deg(f) + N \max_i \operatorname{degdiff}(f_i)$.

Proof. Write α as $\alpha = \sum_{i=1}^{d} c_i a_i \varepsilon_{j_i} b_i$ with nonzero $c_i \in K$ and pairwise different $a_i \varepsilon_{j_i} b_i$. By definition, $\deg(\alpha) = \max_i \deg(a_i f_{j_i} b_i)$, and, according to Lemma 5.4.6, each $a_i f_{j_i} b_i$ is a rewriter in a rewriting sequence from f to 0. Thus, Lemma 5.4.7 shows inductively that $\deg(\alpha) \leq \deg(f) + \|\alpha\|_0 \max_i \operatorname{degdiff}(f_i)$, and the result follows since $\|\alpha\|_0 \leq N$. □

As a consequence, we can state an algorithm for computing a cofactor representation of weight bounded by $N \in \mathbb{N}$, reducing to the problem of finding a sparse solution of a linear system.

Algorithm 11: Sparse cofactor representation

Input: $f, f_1, \dots, f_r \in K\langle X \rangle$, $N \in \mathbb{N}$
Output: $\alpha \in R(f)$ with $\|\alpha\|_0 \leq N$ if one exists, otherwise False

1 $D \leftarrow \deg(f) + N \max_i \operatorname{degdiff}(f_i)$;
2 $L \leftarrow \{a f_i b \mid a, b \in \langle X \rangle, \deg(a f_i b) \leq D, i = 1, \dots, r\}$;
3 **return** a K-linear combination of elements of L equal to f with $\leq N$ nonzero summands if one exists, otherwise False;

Corollary 5.4.9. *Algorithm 11 terminates and is correct.*

Proof. The algorithm reduces the problem to that of finding sparse solutions of a linear system. This problem is decidable (recall that K is computable), so the algorithm terminates.

There exists a representation of f of weight $\leq N$ if and only if there exists a *minimal* representation of f of weight $\leq N$. By Corollary 5.4.8, such a minimal representation is given by a K-linear combination of elements of L. So the algorithm is correct. □

It is also possible to describe a reduction of Problem 5.4.4 to Problem 5.4.3, which allows us to characterise the complexity of the problem of finding sparse representations in terms of the complexity of Problem 5.4.4. For the practically most relevant case of $K = \mathbb{Q}$, we arrive at the following theorem.

Theorem 5.4.10. *The problem of, given $f, f_1, \ldots, f_r \in \mathbb{Q}\langle X\rangle$ and $N \in \mathbb{N}$ (in unary form), deciding whether there exists a cofactor representation of f of weight at most N, is* NP*-complete.*

Proof. Over $\mathbb{Q}$, the decision problem associated to Problem 5.4.4 is NP-complete [GJ79, Problem MP5]. Given an input $A, \boldsymbol{b}, N$ to that problem, introduce one variable x_i for each row of A, interpret each column of A as the polynomial $f_j = \sum_i A_{i,j} x_i$, and the right-hand side as the polynomial $f = \sum_i \boldsymbol{b}_i x_i$. There is a one-to-one correspondence between solutions with N nonzero entries to the linear system and cofactor representations of f with weight N. So the problem of finding a representation of weight at most N is also NP-hard.

Furthermore, if there exists a representation of weight $\leq N$, then there exists one with degree $\leq \deg(f) + N \max_i \operatorname{degdiff}(f_i)$, which makes it polynomial size in N and the size of the input polynomials. The validity of that representation can be verified in polynomial time. So the problem is NP, and therefore NP-complete. □

Remark 5.4.11. *The requirement that N be given in unary format is necessary because unlike Problem 5.4.4, the input of Problem 5.4.3 is not at least of size N. If N is given in binary format, the decision problem is still* NP*-hard but no longer* NP*, because the degree bound is not polynomial in* $\log(N)$*. Also note that Algorithm 11 does not provide a polynomial time reduction of Problem 5.4.3 to Problem 5.4.4, even as a function of N.*

We note that the last step of Algorithm 11 is infeasible for nontrivial examples. To illustrate this point, we consider the following simple statement about the Moore-Penrose inverse taken from [Hog13, Ch. 5.7 Fact 11].

Theorem 5.4.12. *Let A be an invertible matrix with inverse B. Then B is the Moore-Penrose inverse of A.*

Proof. Let $A^\dagger$ be the Moore-Penrose inverse of A. So, in particular, $AA^\dagger A = A$, and hence $B = BAB = BAA^\dagger AB = BAA^\dagger = A^\dagger$. □

Example 5.4.13. *Using Theorem 5.1.7, Theorem 5.4.12 can be encoded in terms of the ideal membership* $b - a^\dagger \in (F)$ *with*

$$F = \{ab - 1, ba - 1, aa^\dagger a - a, a^\dagger aa^\dagger - a^\dagger, (a^\dagger)^* a^* - aa^\dagger, a^*(a^\dagger)^* - a^\dagger a\}$$

in the algebra $\mathbb{Z}\langle a, a^*, a^\dagger, (a^\dagger)^*, b\rangle$.

The proof given above is then equivalent to the following cofactor representation of $b - a^\dagger$ *certifying the ideal membership:*

$$b - a^\dagger = a^\dagger(ab - 1) - b(ab - 1) - b(aa^\dagger a - a)b + (ba - 1)a^\dagger ab. \tag{5.4}$$

This cofactor representation consists of 4 *terms. To see if there exists a representation with* ≤ 3 *terms, we can call Algorithm 11 with* $N = 3$*. The set* L *contains polynomials of degree at most* $D = 7$*, and it consists of* 88 672 *elements. This is too large to test all 3-subsets exhaustively.*

Using the techniques of Section 5.4.2, we will see that a much smaller set of elements is sufficient, and by applying the results of that section we will be able to verify that (5.4) *is in fact a sparsest representation of* $b - a^\dagger$*. This shows that the proof given above is a shortest proof of the theorem.*

5.4.2 Computing sparse representations

In this section, we restrict ourselves to the case $K = \mathbb{Q}$. Furthermore, we note that, in the following, we will rely heavily on terminology, notation, and results from Sections 3.1 – 3.3.

We have seen in the previous section that computing sparsest cofactor representations is equivalent to the NP-hard problem of finding sparsest solutions of a linear system. Several methods have been proposed to obtain approximate solutions of the latter [CW92; MZ93; CDS01] by using other measures as proxies for the sparsity of a solution and by minimising over them. One of these methods, called *Basis Pursuit* [CDS01], uses the ℓ_1-norm as an approximation for the sparsity of a solution.

In the following, we follow the Basis Pursuit approach and use the ℓ_1-norm $\|\alpha\|_1 \coloneqq \sum_{i=1}^d |c_i|$ of $\alpha = \sum_{i=1}^d c_i a_i \varepsilon_{j_i} b_i$ as a surrogate complexity measure of a cofactor representation. The advantage of this approach is that an ℓ_1-minimal solution of a linear system over $\mathbb{Q}$ can be

found efficiently using linear programming. Additionally, we use the effective description of the syzygy module provided by signature-based Gröbner basis algorithms to reduce the size of the linear system that we have to consider.

Based on Corollary 5.4.8, it suffices to consider only cofactor representations up to a degree bound when computing minimal representations. Here, we, more generally, restrict to representations with signature less than a designated bound $\sigma \in M(\Sigma)$. To this end, we fix a monomial order $\preceq$ on $\langle X \rangle$ and the module order $\preceq_\Sigma$ on $M(\Sigma)$, satisfying:

- $\preceq$ and $\preceq_\Sigma$ are *compatible* in the sense that, for all $a, b \in \langle X \rangle$ and $i = 1, \dots, r$, we have
$$a \prec b \iff a\varepsilon_i \prec_\Sigma b\varepsilon_i \iff \varepsilon_i a \prec_\Sigma \varepsilon_i b;$$
- $\preceq_\Sigma$ is *fair*, meaning that the set $\{\mu' \in M(\Sigma) \mid \mu' \prec_\Sigma \mu\}$ is finite for all $\mu \in M(\Sigma)$;

In the following, we will denote both orders by the same symbol $\preceq$. As in Chapter 3, this shall cause no confusion as module elements will be denoted by Greek letters and polynomials by Roman letters.

The fairness of the module order ensures that we work in a finite dimensional vector space when restricting to representations with signature less than a designated bound $\sigma \in M(\Sigma)$. If the module order is also compatible with the degree, that is, if $\deg(\alpha) \leq \deg(\beta)$ implies $\alpha \preceq \beta$, this includes all cofactor representations of degree $< \deg(\sigma)$.

So, formally, we seek a minimal element with respect to $\|\cdot\|_1$ in the set

$$R(f, \sigma) \coloneqq \{\alpha \in R(f) \mid \operatorname{sig}(\alpha) \prec \sigma\}$$

of cofactor representations of f *up to signature* $\sigma \in M(\Sigma)$. We denote the set of all such ℓ_1-minimal elements by $R_1(f, \sigma)$. Analogously, we let $R_0(f, \sigma)$ be the set of all minimal elements with respect to $\|\cdot\|_0$ in $R(f, \sigma)$.

The results in this section rely on the fact that we have some $\alpha \in R(f, \sigma)$. However, in general, for σ too small, the set $R(f, \sigma)$ can be empty, even if $R(f) \neq \emptyset$. To resolve this issue, we assume that we have a cofactor representation $\alpha \in R(f)$ and that σ is chosen so that $\sigma \succ \operatorname{sig}(\alpha)$. Note that this assumption, in particular, implies that $f \in (f_1, \dots, f_r)$. Such α can be obtained, for example, by reducing f to zero using a (partial) (labelled)

Gröbner basis and keeping track of the reductions. With this in mind, we assume that $R(f, \sigma) \neq \emptyset$.

In the following, we describe Algorithm 13, which allows to compute an element in the set $R_1(f, \sigma)$. To this end, we denote by $I^{[\Sigma]}$ the labelled module generated by $f_1, \ldots, f_r$ and by H_σ a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ up to signature σ. We note that H_σ can be computed, for example, using Algorithm 2, or with Algorithm 3 and the reconstruction techniques discussed in Section 3.3.3. In particular, since the module order is assumed to be fair, this computation can be done in finite time, see also Remark 3.3.17.

The general idea of Algorithm 13 is still to reduce the problem of computing sparse cofactor representations to computing certain solutions of a linear system. However, instead of choosing all polynomials af_ib to form the linear system like Algorithm 11 does, we use the information provided by H_σ to trim this set. More precisely, we find a finite set of module monomials $B = \{\mu_1, \ldots, \mu_d\} \subseteq M(\Sigma)$ such that $R_i(f, \sigma)$, $i = 0, 1$, has nonempty intersection with the $\mathbb{Q}$-vector space generated by B and then only consider the polynomials $\{\overline{\mu}_1, \ldots, \overline{\mu}_d\}$ to form the linear system. Furthermore, we now no longer seek a sparsest solution of the resulting system but an ℓ_1-minimal solution, which can be found with linear programming.

It remains to discuss how to find a suitable basis B and how to translate the problem of finding ℓ_1-minimal solutions of a linear system into a linear programming problem.

Finding a suitable basis B

Algorithm 11 essentially uses the basis $B = \{a\varepsilon_i b \mid a, b \in \langle X \rangle, i = 1, \ldots, r, \mathrm{sig}(a\varepsilon_i b) \prec \sigma\}$, which leads to finite dimensional, yet infeasibly large, linear systems. Using a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ up to signature σ, we can drastically reduce the dimension of the search space. To describe how this can be done, we first extend the notion of rewriting to module elements.

Definition 5.4.14. *Let $\alpha, \gamma \in \Sigma$ and $a, b \in \langle X \rangle$ such that $\mathrm{supp}(\alpha) \cap \mathrm{supp}(a\gamma b) \neq \emptyset$. For every $c \in \mathbb{Q}$, we say that α* can be rewritten to $\alpha + ca\gamma b$ by γ.

Furthermore, we say that α can be rewritten to β by $H \subseteq \Sigma$ *if there are* $\beta_0, \dots, \beta_d \in \Sigma$, $\beta_d = \alpha$, $\beta_0 = \beta$ *and* $\gamma_1, \dots, \gamma_d \in H$ *such that* β_k *can be rewritten to* β_{k-1} *by* γ_k *for all* $k = 1, \dots, d$.

With this, we can state a module version of Lemma 5.4.6. We note that we state all results in this section for both the ℓ_0-"norm" and the ℓ_1-norm to emphasise that they hold for both complexity measures likewise and that the restriction to $\|\cdot\|_1$ only comes later for the linear programming.

Lemma 5.4.15. *Let* $i \in \{0, 1\}$. *Furthermore, let* $\alpha \in R(f, \sigma)$, $\alpha_i \in R_i(f, \sigma)$, *and let* H_σ *be a Gröbner basis of* $\mathrm{Syz}(I^{[\Sigma]})$ *up to signature* σ. *Then* α *can be rewritten to* α_i *by* H_σ. *In particular, this rewriting can be done so that the signature of every rewriter* $a_j\gamma_j b_j$ *is less than* σ.

To prove Lemma 5.4.15, we make use of the fact that the ℓ_0-"norm" and the ℓ_1-norm are linear for elements of disjoint support.

Lemma 5.4.16. *For* $\alpha, \beta \in \Sigma$ *with* $\mathrm{supp}(\alpha) \cap \mathrm{supp}(\beta) = \emptyset$, *we have* $\|\alpha + \beta\|_i = \|\alpha\|_i + \|\beta\|_i$ *for* $i = 0, 1$.

Proof of Lemma 5.4.15. The difference $\alpha - \alpha_i$ is a syzygy with signature $\prec \sigma$. Since H_σ is a Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ up to signature σ, there exist $d \in \mathbb{N}$ and $\gamma_j \in H_\sigma$, $c_j \in \mathbb{Q}$, $a_j, b_j \in \langle X \rangle$ such that $\alpha_i = \alpha - \sum_{j=1}^{d} c_j a_j \gamma_j b_j$ and $\mathrm{sig}(a_j\gamma_j b_j) \preceq \max\{\mathrm{sig}(\alpha), \mathrm{sig}(\alpha_i)\} \prec \sigma$ for all j. Now, we essentially follow the proof of Lemma 5.4.6 and perform induction on d.

The case $d = 0$ is clear. Assume now that $d > 0$ and that the result is proven for all pairs (α, α_i) such that $\alpha - \alpha_i$ has a representation with $d - 1$ terms. Let $\beta = \sum_{j=1}^{d} c_j a_j \gamma_j b_j$. If $\beta = 0$, we are done since $\alpha = \alpha_i$. So assume $\beta \neq 0$, which implies $\|\beta\|_i > 0$. Then we must have $\mathrm{supp}(\alpha) \cap \mathrm{supp}(\beta) \neq \emptyset$, as otherwise Lemma 5.4.16 would yield the contradiction $\|\alpha_i\|_i = \|\alpha - \beta\|_i = \|\alpha\|_i + \|\beta\|_i > \|\alpha\|_i \geq \|\alpha_i\|_i$, where the last inequality follow from the minimality of $\|\alpha_i\|_i$. Thus, we have $\mathrm{supp}(\alpha) \cap \mathrm{supp}(a_j\gamma_j b_j) \neq \emptyset$ for some $1 \leq j \leq d$. Without loss of generality, assume $j = d$. Hence, α can be rewritten to $\beta' = \alpha - c_d a_d \gamma_d b_d$ by $\gamma_d \in H_\sigma$. Note that $\mathrm{sig}(a_d\gamma_d b_d) \prec \sigma$. Since $\beta' - \alpha_i = \sum_{j=1}^{d-1} c_j a_j \gamma_j b_j$ has a representation with $d - 1$ terms, the induction hypothesis implies that β' can be rewritten to α_i by H_σ using only rewriters $a_j\gamma_j b_j$ with signature $\prec \sigma$. □

Lemma 5.4.15 says that any $\alpha \in R(f,\sigma)$ can be rewritten to each element in $R_i(f,\sigma)$, $i = 0, 1$, by H_σ using only rewriters with signature bounded by σ. Consequently, to find a suitable basis B, it suffices, starting from some α, to only choose those syzygies that can appear in such rewriting sequences. Finding these elements is a purely combinatorial problem that can be solved without having to perform any actual rewriting steps. This leads to Algorithm 12, in which we collect precisely all those relevant syzygies. Algorithm 12 can be considered as an adaptation of the symbolic preprocessing in the F4 algorithm [Fau99]. In the following, for $V \subseteq \Sigma$, let $\operatorname{supp}(V) = \bigcup_{\gamma \in V} \operatorname{supp}(\gamma)$. Furthermore, $\operatorname{span}_{\mathbb{Q}}(V)$ denotes the $\mathbb{Q}$-vector space spanned by the elements in V.

Algorithm 12: Finding relevant syzygies

Input: $\alpha \in R(f,\sigma)$, H_σ a Gröbner basis of $\operatorname{Syz}(I^{[\Sigma]})$ up to signature σ
Output: $V \subseteq \operatorname{Syz}(I^{[\Sigma]})$ such that $R_i(f,\sigma) \subseteq \alpha + \operatorname{span}_{\mathbb{Q}}(V)$ for $i = 0, 1$

$V \leftarrow \emptyset$;
$\texttt{todo} \leftarrow \operatorname{supp}(\alpha)$; $\texttt{done} \leftarrow \emptyset$;
while $\texttt{todo} \neq \emptyset$ **:**
    select $\mu \in \texttt{todo}$, remove it, and add it to done;
    $\texttt{new} \leftarrow \{a\gamma b \mid a, b \in \langle X \rangle, \gamma \in H_\sigma, \mu \in \operatorname{supp}(a\gamma b), \operatorname{sig}(a\gamma b) \prec \sigma\}$;
    $\texttt{todo} \leftarrow \texttt{todo} \cup (\operatorname{supp}(\texttt{new}) \setminus \texttt{done})$;
    $V \leftarrow V \cup \texttt{new}$;
return V;

Proposition 5.4.17. *Algorithm 12 terminates and is correct.*

Proof. The conditions on the elements in new ensure that only module monomials smaller than σ are inserted into todo. Furthermore, each monomial is processed at most once. Consequently, termination follows from the fact that there are only finitely many monomials smaller than σ (recall that $\preceq$ is fair). Correctness follows from Lemma 5.4.15. □

Using Algorithm 12, we can set $B = \operatorname{supp}(\alpha) \cup \operatorname{supp}(V)$ as a basis of the search space, where V is the output of the algorithm given α and H_σ as input. In many cases, this set is small enough to reasonably work with.

Detecting redundant syzygies

As an optional step, we can remove redundant elements from V before forming the basis B in order to obtain a smaller basis, and thus, a smaller linear program to solve. More precisely, since we only want to compute one element in $R_i(f,\sigma)$, $i = 0, 1$, we can remove syzygies as long as we can ensure that there remains at least one rewriting sequence from α to at least one element in $R_i(f,\sigma)$. We mention two basic techniques that turned out useful in practice.

The first technique allows to remove syzygies from V that consist mostly of terms that appear in no other element. Such syzygies cannot lead to simpler representations. To make this statement precise, for $W \subseteq V \subseteq \mathrm{Syz}(I^{[\Sigma]})$ and $\beta = \sum_j c_j\mu_j \in W$ with $c_j \in \mathbb{Q}$, $\mu_j \in M(\Sigma)$, we denote

$$\beta_U \coloneqq \sum_j c_j\mu_j \text{ with } j \text{ such that } \mu_j \notin \mathrm{supp}\big((V \cup \{\alpha\}) \setminus \{\beta\}\big),$$
$$\beta_V \coloneqq \sum_j c_j\mu_j \text{ with } j \text{ such that } \mu_j \in \mathrm{supp}\big((V \cup \{\alpha\}) \setminus W\big).$$

Intuitively, the element β_U contains all those terms of β that are unique to β and that appear in no other element of $V \cup \{\alpha\}$, and β_V contains those terms that appear in β as well as in elements outside of W.

Proposition 5.4.18. *Let $i \in \{0,1\}$. Furthermore, let $\alpha \in R(f,\sigma)$ and $V \subseteq \mathrm{Syz}(I^{[\Sigma]})$ such that*

$$(\alpha + \mathrm{span}_{\mathbb{Q}}(V)) \cap R_i(f,\sigma) \neq \emptyset.$$

If $W \subseteq V$ satisfies $\|\beta_V\|_i \leq \|\beta_U\|_i$ for all $\beta \in W$, then

$$(\alpha + \mathrm{span}_{\mathbb{Q}}(V \setminus W)) \cap R_i(f,\sigma) \neq \emptyset.$$

Proposition 5.4.18 provides a sufficient condition for a subset $W \subseteq V$ to be redundant. In order to prove this, we need the following two lemmas. The first one states that the required property of W extends to the whole linear span. To this end, we extend the definition of β_U and β_V to elements $\beta = \sum_j b_j\beta_j \in \mathrm{span}_{\mathbb{Q}}(W)$, where $b_j \in \mathbb{Q}$ and $\beta_j \in W$, by $\beta_U \coloneqq \sum_j b_j\beta_{j,U}$ and $\beta_V \coloneqq \sum_j b_j\beta_{j,V}$. Note that, in general, these definitions depend

on the representation of β in terms of the elements in W; different linear combinations of the same element can yield different definitions of β_U and β_V. Therefore, to obtain an unambiguous definition, we assume that, for every element $\beta \in \operatorname{span}_{\mathbb{Q}}(W)$, one particular representation in terms of W has been fixed, and this is the representation that we use to compute β_U and β_V. We note that, for all our arguments, the particular choice of the representation does not matter. It is only important that, for each β, the elements β_U and β_V are computed with respect to the same representation.

Lemma 5.4.19. *Let V, W be as in Proposition 5.4.18. If $\beta \in \operatorname{span}_{\mathbb{Q}}(W)$, then $\|\beta_V\|_i \leq \|\beta_U\|_i$.*

Proof. Write $\beta = \sum_j b_j \beta_j$ with nonzero $b_j \in \mathbb{Q}$ and $\beta_j \in W$. By assumption $\|\beta_{j,V}\|_i \leq \|\beta_{j,U}\|_i$ for all j. Furthermore, all $\beta_{j,U}$ have pairwise disjoint supports as they consist of the monomials that are unique to each β_j. So, Lemma 5.4.16 implies that $\|\cdot\|_i$ is linear on linear combinations of the $\beta_{j,U}$. Using this and the triangular inequality, we get with $c_j = 1$ if $i = 0$ and $c_j = |b_j|$ if $i = 1$:

$$\begin{aligned}\|\beta_V\|_i &\leq \sum_j \|b_j \beta_{j,V}\|_i = \sum_j c_j \|\beta_{j,V}\|_i \\ &\leq \sum_j c_j \|\beta_{j,U}\|_i = \sum_j \|b_j \beta_{j,U}\|_i = \|\beta_U\|_i.\end{aligned}$$ □

The second lemma provides a lower bound on the norm of sums $\gamma + \beta \in \alpha + \operatorname{span}_{\mathbb{Q}}(W)$.

Lemma 5.4.20. *Let α, V, W be as in Proposition 5.4.18. If $\beta \in \operatorname{span}_{\mathbb{Q}}(W)$ and $\gamma \in \alpha + \operatorname{span}_{\mathbb{Q}}(V \setminus W)$, then $\|\gamma + \beta\|_i \geq \|\gamma\|_i - \|\beta_V\|_i + \|\beta_U\|_i$.*

Proof. Let $\beta' = \beta - (\beta_U + \beta_V)$. By definition, β_U and β' have pairwise different supports. Furthermore, $\gamma + \beta_V$ does not share a monomial with β_U and β' as $\operatorname{supp}(\gamma + \beta_V) \subseteq \operatorname{supp}((V \cup \{\alpha\}) \setminus W)$ and all monomials of β that lie in this set are collected in β_V. Therefore, Lemma 5.4.16 and the inverse triangle inequality imply

$$\begin{aligned}\|\gamma + \beta\|_i &= \|\gamma + \beta_V\|_i + \|\beta_U\|_i + \|\beta'\|_i \\ &\geq \|\gamma + \beta_V\|_i + \|\beta_U\|_i \geq \|\gamma\|_i - \|\beta_V\|_i + \|\beta_U\|_i.\end{aligned}$$ □

Proof of Proposition 5.4.18. We claim that removing, if present, elements from W from a representation $\delta \in \alpha + \operatorname{span}_{\mathbb{Q}}(V)$ cannot increase the norm. This implies the assertion of the proposition. To prove our claim, write δ as $\delta = \gamma + \beta$ with $\gamma \in \alpha + \operatorname{span}_{\mathbb{Q}}(V \setminus W)$ and $\beta \in \operatorname{span}_{\mathbb{Q}}(W)$. Now, Lemma 5.4.19 and 5.4.20, show that $\|\delta\|_i = \|\gamma + \beta\|_i \geq \|\gamma\|_i - \|\beta_V\|_i + \|\beta_U\|_i \geq \|\gamma\|_i$. □

The redundancy test provided by Proposition 5.4.18 is computationally fairly cheap to check for a given set $W \subseteq V$. However, finding suitable candidates for W is not so trivial. In our implementation, we test all singletons $\{\beta\} \subseteq V$, as well as all subsets $\{\beta, \gamma\} \subseteq V$ satisfying

- $\operatorname{supp}(\beta) \cap \operatorname{supp}(\gamma) \neq \emptyset$, and
- $\frac{|\operatorname{supp}(\delta) \cap \operatorname{supp}(V \setminus \{\delta\})|}{|\operatorname{supp}(\delta)|} \geq \frac{1}{3}$ for $\delta \in \{\beta, \gamma\}$.

This empirically provided the best trade-off between efficiency in applying the criterion and the effect it had on pruning V.

The second method does not directly allow to detect redundant elements in V. Instead it can be considered as an auxiliary technique that can cause additional applications of Proposition 5.4.18. The idea is to replace elements in V by linear combinations so that the number of occurrences of certain monomials is reduced. In particular, by exploiting the fact that

$$\operatorname{span}_{\mathbb{Q}}(V \cup \{\alpha - \beta, \gamma + \beta\}) = \operatorname{span}_{\mathbb{Q}}(V \cup \{\alpha - \beta, \alpha + \gamma\}), \tag{5.5}$$

we can reduce the number of occurrences of β at the cost of increasing the occurrences of α.

In our implementation, we apply this technique to all binomial syzygies $\mu - \sigma \in V$, where $\mu, \sigma \in M(\Sigma)$. After removing all occurrences of σ, Proposition 5.4.18 allows to delete the binomial syzygy from V. Additionally, we apply (5.5) randomly to elements $\alpha - \beta$ where $\|\beta\|_i > c\|\alpha\|_i$ for fixed $c > 1$. Often, this process triggers further invocations of Proposition 5.4.18 to remove elements from V. Table 5.1 shows the efficiency of the two methods presented in this section.

Translation into linear program

Once we have obtained a reasonable basis of module monomials $B = \{\mu_1, \ldots, \mu_d\}$ such that the $\mathbb{Q}$-vector space generated by B has nonempty intersection with $R_i(f, \sigma)$ for $i = 0, 1$, we can set up a linear system $A\boldsymbol{y} = \boldsymbol{b}$, where A is the matrix of size $s \times d$, with $s = |\bigcup_j \operatorname{supp}(\overline{\mu}_j)|$, whose jth column contains the coefficients of $\overline{\mu}_j$, associating to each row of A a monomial $w \in \bigcup_j \operatorname{supp}(\overline{\mu}_j)$. Similarly, $\boldsymbol{b}$ is a vector of size s containing the coefficients of f. The matrix A bears resemblance to the matrices appearing in Gröbner basis computations such as the F4 algorithm, aside from two main differences. In Gröbner basis computations, polynomials are encoded as the rows of a matrix, and the columns have to be ordered with respect to a (polynomial) monomial order. In our approach, polynomials are encoded as the columns and the order of the rows is irrelevant.

Every solution $\boldsymbol{y}$ of $A\boldsymbol{y} = \boldsymbol{b}$ corresponds to a cofactor representation of f with support in $B = \{\mu_1, \ldots, \mu_d\} \subseteq M(\Sigma)$. To see this, we denote by $A_{i,j}$ the entries of A and by $\boldsymbol{b}_i$ and $\boldsymbol{y}_j$ the coordinates of $\boldsymbol{b}$ and $\boldsymbol{y}$ respectively. Furthermore, let $w_i \in \bigcup_j \operatorname{supp}(\overline{\mu}_j)$ be the monomial that is associated to the ith row of A and the ith coordinate of $\boldsymbol{b}$. Then, we can write $f = \sum_{i=1}^{s} \boldsymbol{b}_i w_i$ and $\overline{\mu}_j = \sum_{i=1}^{s} A_{i,j} w_i$, for $j = 1, \ldots, d$, and we see that

$$f = \sum_{i=1}^{s} \boldsymbol{b}_i w_i = \sum_{i=1}^{s} \left(\sum_{j=1}^{d} A_{i,j} \boldsymbol{y}_j \right) w_i = \sum_{j=1}^{d} \boldsymbol{y}_j \left(\sum_{i=1}^{s} A_{i,j} w_i \right) = \sum_{j=1}^{d} \boldsymbol{y}_j \overline{\mu}_j,$$

showing that $\sum_{j=1}^{d} \boldsymbol{y}_j \mu_j \in R(f, \sigma)$ is a cofactor representation of f.

Moreover, every ℓ_i-minimal solution of $A\boldsymbol{y} = \boldsymbol{b}$ corresponds to an element in $R_i(f, \sigma)$. As noted before, computing ℓ_0-minimal, that is, sparsest, solutions is NP-hard. Therefore, we restrict ourselves to the case $i = 1$ and consider the problem

$$(P_1): \quad \min_{\boldsymbol{y}} \|\boldsymbol{y}\|_1 \quad \text{subject to} \quad A\boldsymbol{y} = \boldsymbol{b},$$

where $\|\boldsymbol{x}\|_1 = \sum_j |\boldsymbol{x}_j|$. It is well-known that (P_1) can be recast as a linear program, see for example [CDS01, Sec. 3.1]. A linear program (in standard form) [Sch98] is an optimisation problem for $\boldsymbol{v} \in \mathbb{Q}^t$ of the form

$$(LP): \quad \min_{\boldsymbol{v}} \boldsymbol{c}^T \boldsymbol{v} \quad \text{subject to} \quad U\boldsymbol{v} = \boldsymbol{w}, \quad \boldsymbol{v} \geq 0,$$

where $\boldsymbol{v} \geq 0$ is to be understood component-wise. The problem (P_1) can be equivalently formulated as a linear program by setting

$$t = 2d, \quad \boldsymbol{c}^T = (1, \ldots, 1), \quad U = (A \mid -A), \quad \boldsymbol{v} = \begin{pmatrix} \boldsymbol{p} \\ \boldsymbol{q} \end{pmatrix}, \quad \boldsymbol{w} = \boldsymbol{b},$$

with vectors $\boldsymbol{p}, \boldsymbol{q} \in \mathbb{Q}^d$. This linear program can then be solved efficiently using the simplex algorithm [Dan51] or interior-point methods [PW00] and a solution $\boldsymbol{y}$ of (P_1) is given by $\boldsymbol{y} = \boldsymbol{p} - \boldsymbol{q}$.

Putting everything together

Finally, we combine the results of the previous sections to form Algorithm 13 for computing an element in $R_1(f, \sigma)$. In the algorithm, $I^{[\Sigma]}$ denotes the labelled module generated by $f_1, \ldots, f_r$.

Algorithm 13: ℓ_1-minimal cofactor representation

Input: $f_1, \ldots, f_r \in \mathbb{Q}\langle X \rangle$, $f \in (f_1, \ldots, f_r)$, $\sigma \in M(\Sigma)$, $\alpha \in R(f, \sigma)$
Output: an element in $R_1(f, \sigma)$

1 $H_\sigma \leftarrow$ Gröbner basis of $\mathrm{Syz}(I^{[\Sigma]})$ up to signature σ;
2 $V \leftarrow$ apply Algorithm 12 to α and H_σ;
3 $V \leftarrow$ prune V using the techniques from Section 5.4.2;
4 $\{\mu_1, \ldots, \mu_d\} \leftarrow \mathrm{supp}(V \cup \{\alpha\})$;
5 $A \leftarrow$ matrix with columns containing the coefficients of $\overline{\mu}_1, \ldots, \overline{\mu}_d$;
6 $\boldsymbol{b} \leftarrow$ vector containing the coefficients of f;
7 $\boldsymbol{v} \leftarrow$ solution of the linear program (LP) with

$$\boldsymbol{c}^T = (1, \ldots, 1), \qquad U = (A \mid -A), \qquad \boldsymbol{v} = \begin{pmatrix} \boldsymbol{p} \\ \boldsymbol{q} \end{pmatrix}, \qquad \boldsymbol{w} = \boldsymbol{b};$$

8 **return** $\sum_{i=1}^{d} (\boldsymbol{p}_i - \boldsymbol{q}_i)\mu_i$;

Theorem 5.4.21. *Algorithm 13 terminates and is correct.*

Proof. Termination follows from the fact that H_σ can be computed in finite time by Theorem 3.3.16, see also Remark 3.3.17 (recall that the module order is assumed to be fair), and from Proposition 5.4.17. Correctness follows from the discussions in this section. □

Remark 5.4.22. *Algorithm 13 weighs each monomial μ_i equally by a weight of* 1*. It is also possible to weigh the monomials differently by changing the vector $\boldsymbol{c}$ so that $\boldsymbol{c}_i$ encodes the weight of μ_i. This allows, for example, to weigh monomials by their degree, yielding representations that prefer monomials with small degree. In this case, the output of the algorithm is no longer guaranteed to be in $R_1(f, \sigma)$.*

Special case: totally unimodular matrices

In general, the output of Algorithm 13 need not be a sparsest representation of f up to signature σ, that is, it need not be an element in $R_0(f, \sigma)$. In this section, we discuss a special case when this is indeed true. To this end, we consider the linear system $A\boldsymbol{y} = \boldsymbol{b}$ constructed in Algorithm 13. We are interested in situations where the augmented matrix $(A \mid \boldsymbol{b})$ is *totally unimodular* as defined below.

Definition 5.4.23. *A matrix $T \in \{-1, 0, 1\}^{m \times n}$ is called* totally unimodular *if every square submatrix of T has determinant* 0 *or* ± 1*.*

Theorem 5.4.24. *Let A and $\boldsymbol{b}$ be as constructed in Algorithm 13. If the augmented matrix $(A \mid \boldsymbol{b})$ is totally unimodular, then the output of Algorithm 13 is an element in $R_0(f, \sigma)$.*

In order to prove the theorem, we take a closer look at the coefficients of the sparsest and ℓ_1-minimal solutions of $A\boldsymbol{y} = \boldsymbol{b}$. It is well-known that totally unimodular coefficient matrices and integer right-hand sides yield integer optima for linear programs [Sch98, Cor. 19.1a]. The following lemma extends this statement under slightly stricter assumptions.

Lemma 5.4.25. *Let the augmented matrix $(A \mid \boldsymbol{b})$ be totally unimodular. If $A\boldsymbol{y} = \boldsymbol{b}$ is solvable, then any sparsest or ℓ_1-minimal solution $\boldsymbol{y}$ satisfies $\boldsymbol{y} \in \{-1, 0, 1\}^d$.*

Proof. Since removing linearly dependent rows does not change the solution set of a solvable system, we can assume that A has full row rank $s = \operatorname{rank}(A)$.

Sparsest solution. The columns of A corresponding to the nonzero entries of a sparsest solution $\boldsymbol{y}$ have to be linearly independent (otherwise there would exist a sparser solution). We can extend them by further columns of A to obtain an invertible $s \times s$ matrix A'. Then

$A'\boldsymbol{y}' = \boldsymbol{b}$, where $\boldsymbol{y}'$ contains those coordinates of $\boldsymbol{y}$ that correspond to the columns of A that are in A'. By assumption $\det(A') = \pm 1$. Furthermore, the matrix A'_i, obtained by replacing the ith column of A' by $\boldsymbol{b}$, is – up to permutation of columns – a submatrix of $(A \mid \boldsymbol{b})$. Consequently, $\det(A'_i) \in \{-1, 0, 1\}$, and applying Cramer's rule shows $\boldsymbol{y}'_i = \frac{\det(A'_i)}{\det(A')} \in \{-1, 0, 1\}$. Then the result follows since any coordinate of $\boldsymbol{y}$ that does not appear in $\boldsymbol{y}'$ has to be zero.

ℓ_1-minimal solution. We consider the equivalent linear program (LP) and note that $\boldsymbol{y} \in \{-1, 0, 1\}^d$ if and only if $\boldsymbol{v} \in \{0, 1\}^{2d}$. If $\boldsymbol{v}$ is a solution of (LP), then it has to be a basic feasible solution. This means $\|\boldsymbol{v}\|_0 \leq s$ and that the columns of U that correspond to the nonzero coordinates of $\boldsymbol{v}$ can be extended to an invertible $s \times s$ submatrix U' of U. Since $(U \mid \boldsymbol{b}) = (A \mid -A \mid \boldsymbol{b})$ is totally unimodular, the same arguments as in the other case show that $\boldsymbol{v} \in \{-1, 0, 1\}^{2d}$, and the statement follows from the nonnegativity constraint of (LP). □

Using this lemma, we can now prove Theorem 5.4.24.

Proof of Theorem 5.4.24. By construction, the system $A\boldsymbol{y} = \boldsymbol{b}$ has a solution. For $i = 0, 1$, let α_i be the module element corresponding to an ℓ_i-minimal solution of the system. Note that, again by construction, $\alpha_i \in R_i(f, \sigma)$. By Lemma 5.4.25, α_i contains only nonzero coefficients ± 1, which implies that $\|\alpha_i\|_0 = \|\alpha_i\|_1$ for $i = 0, 1$, and the result follows. □

In most applications, all polynomials involved are of the form $p - q$ with $p, q \in \langle X \rangle \cup \{0\}$ encoding identities of operators of the form $P = Q$. Such polynomials are called *pure difference binomials*. The following corollary of Theorem 5.4.24 ensures that Algorithm 13 computes a sparsest representation up to signature σ provided that the input polynomials are pure difference binomials.

Corollary 5.4.26. *Let $f, f_1, \ldots, f_r \in \mathbb{Q}\langle X \rangle$ be pure difference binomials, $\sigma \in M(\Sigma)$ and $\alpha \in R(f, \sigma)$. Given these elements as input, the output of Algorithm 13 is an element in $R_0(f, \sigma)$.*

Proof. Let A, $\boldsymbol{b}$ be as constructed in Algorithm 13. By assumption on $f, f_1, \ldots, f_r$, each column of $(A \mid \boldsymbol{b})$ contains at most one entry $+1$ and at most one entry -1 with all other entries being 0. Each square submatrix U of $(A \mid \boldsymbol{b})$ either contains a zero column (then U

is singular), a column with one nonzero entry (then expansion of $\det(U)$ along this column yields inductively $\det(U) \in \{-1, 0, 1\}$), or each column of U contains exactly one entry $+1$ and one entry -1 (then $\mathbf{1}^T U = 0$ showing that U is singular). Thus, $(A \mid \boldsymbol{b})$ is totally unimodular and the result follows from Theorem 5.4.24. □

Example 5.4.13 (continuing from p. 238)**.** *We revisit Example 5.4.13. All polynomials that appear in this example are pure difference binomials. Hence, Corollary 5.4.26 implies that Algorithm 13 yields a sparsest cofactor representation up to the used signature bound σ. In particular, if a degree-compatible module order is used and if σ is chosen so that $\deg(\sigma) > 7$, then, by Corollary 5.4.8, the computed representation is a sparsest one (independent of any bound).*

Applying Algorithm 13 to Example 5.4.13, with the cofactor representation given in (5.4) *and a suitable signature bound σ, yields again* (5.4)*, showing that this is a sparsest cofactor representation. The basis used to form the linear system only consists of* 300 *elements, compared to the* 88 672 *that Algorithm 11 would need.*

To end this section, we note that, for pure difference binomials $f, f_1, \ldots, f_r$, computing sparsest representations can also be considered as a shortest path finding problem. In this case, the vertices of a (possibly infinite) graph are given by all pure difference binomials $p - q$ that lie in the ideal $(f_1, \ldots, f_r)$, and we draw an (undirected) edge from $p - q$ to $p' - q'$ if $p' - q' = (p - q) \pm a f_i b$ for some $1 \leq i \leq r$ and $a, b \in \langle X \rangle$. Then, any path of length N from f to zero yields a representation of f with at most N terms. In particular, shortest paths correspond to sparsest representations. Combining this path finding idea with the degree bound from Corollary 5.4.8 to cut off paths that cannot lead to sparsest representations, also yields a suitable approach for finding shortest representations for the special case of pure difference binomials.

5.4.3 Experiments

We have written a prototype implementation of Algorithm 13 for SageMath using our package `signature_gb` (see Section 6.3) for the signature-based computations and the IBM ILOG CPLEX optimisation studio [IBM23] for the linear programming. This implementation, together with the benchmark examples described below, is available at

`https://clemenshofstadler.com/software/`.

In Table 5.1, we compare the weight of cofactor representations computed by Algorithm 13 to those found by other approaches. In particular, we compare our algorithm to tracing standard Gröbner basis computations and reductions, and to tracing reductions done with a labelled Gröbner basis.

As benchmark examples, we recover short proofs of the following (recent) results in operator theory on the Moore-Penrose inverse:

- `SVD` encodes [Hog13, Ch. 5.7 Fact 4], which provides a formula for the Moore-Penrose inverse of a matrix in terms of the matrix's singular value decomposition.
- `ROL` encodes the implication (2) $\Rightarrow$ (1) in [KDC07, Thm. 3], which provides a sufficient condition for the identity $(AB)^\dagger = B^\dagger A^\dagger$ to hold, where $X^\dagger$ is the Moore-Penrose inverse of an element in a ring with involution.
- `ROL-`n encodes the implication (n) $\Rightarrow$ (1) in [DD10, Thm. 2.1]. This family provides several sufficient conditions for the identity $(AB)^\dagger = B^\dagger A^\dagger$ to hold, where $X^\dagger$ is the Moore-Penrose inverse of a bounded operator on Hilbert spaces.
- `Hartwig-`n encodes the implication (n) $\Rightarrow$ (1) in [Cve+21, Thm. 2.3]. This family provides several sufficient conditions for the identity $(ABC)^\dagger = C^\dagger B^\dagger A^\dagger$ to hold, where $X^\dagger$ is the Moore-Penrose inverse of an element in a ring with involution.
- `Ker` encodes part of [RP87, Thm. 1], which characterises the existence of Moore-Penrose inverses in additive categories with involution in terms of kernels of morphisms.
- `SMW` encodes [Den11, Thm. 2.1], which generalises the Sherman–Morrison–Woodbury formula in terms of the Moore-Penrose inverse.
- `Sum` encodes [Li08, Lem. 1], which provides a sufficient condition for the identity $(A+B)^\dagger = A^\dagger + B^\dagger$ to hold, where $X^\dagger$ is the Moore-Penrose inverse of an element in a C^*-algebra.

For all examples, $\preceq_{\text{deglex}}$ is used in combination with the degree-compatible order $\preceq_{\text{DoPoT}}$ for the signature-based computations.

The first columns of Table 5.1 contain information about the ideals that arise when translating the operator statements. In particular, we list the number of generators of each ideal and their maximal degree. Moreover, in the column for Algorithm 13, we provide

Example	#gens	deg	GB	SigGB	Algo. 13 (bound)	w/o pruning	w/ pruning	ratio $\neq 0$
SVD	32	3	49	39	25 (10)	127 k × 397 k	117 k × 326 k	0.82
ROL	28	5	80	39	30 (12)	22 k × 102 k	22 k × 55 k	0.54
ROL-2	28	5	31	21	15 (12)	24 k × 107 k	23 k × 59 k	0.55
ROL-3	28	5	39	44	31 (12)	19 k × 87 k	18 k × 46 k	0.52
ROL-4	28	5	91	46	33 (12)	68 k × 236 k	64 k × 136 k	0.57
ROL-5	28	5	27	30	22 (12)	33 k × 134 k	31 k × 79 k	0.59
ROL-6	28	5	37	39	30 (12)	22 k × 99 k	21 k × 54 k	0.54
ROL-7	40	9	78	23	17 (12)	18 k × 86 k	17 k × 45 k	0.52
ROL-8	44	7	1203	19	17 (12)	258 k × 965 k	242 k × 548 k	0.57
Hartwig-4	23	15	153	54	46 (18)	353 k × 1756 k	334 k × 1340 k	0.76
Hartwig-5	26	15	117	43	35 (17)	407 k × 1654 k	392 k × 1305 k	0.79
Hartwig-6	24	15	45	33	29 (17)	218 k × 967 k	215 k × 771 k	0.80
Ker	12	3	39	34	23 (12)	50 k × 142 k	50 k × 129 k	0.90
SMW	36	7	102	42	39 (12)	44 k × 114 k	42 k × 91 k	0.80
Sum	20	3	766	178	85 (9)	11 k × 18 k	10 k × 16 k	0.92

Table 5.1: Comparison of weights of cofactor representations computed by standard Gröbner bases (GB), by signature Gröbner bases (SigGB), and by Algorithm 13 (Algo. 13). Also, size comparison of the coefficient matrix A (rounded to thousands) in Algorithm 13 with and without applying the pruning techniques from Section 5.4.2.

information on the used signature bound. A value n in this column indicates that we consider only cofactor representations of degree $< n$. The degree bounds were chosen so that the computation would finish for the larger examples Hartwig-n and ROL-8 within about 90 minutes on a regular laptop and for the remaining smaller examples within a few minutes. We note that these degree bounds are strictly smaller than those that Corollary 5.4.8 yields, but the latter were computationally infeasible. Nevertheless, Table 5.1 shows that Algorithm 13 still allows to find sparser representations for all considered examples.

Apart from the last three (Ker, SMW, Sum), all benchmark examples only consist of pure difference binomials. For those, Corollary 5.4.26 implies that the representations computed by Algorithm 13 are the sparsest up to the respective degree bounds. For the remaining examples, the algorithm can still be used to find ℓ_1-minimal representations, which are heuristically also sparse, but without guarantee that they are the sparsest.

We also tested an adapted version of Algorithm 13 as described in Remark 5.4.22 that minimises the total number of symbols appearing in a cofactor representation. For most benchmark examples, the thereby computed representations have the same (minimal) weight as those found with the standard version of the algorithm, but the total number of

symbols decreases by up to 15%. Only for `ROL-3` does the weight increase by one, while the number of symbols decreases from 196 to 172.

In the last columns of Table 5.1, we compare the size of the matrix A constructed in Algorithm 13 with and without applying the pruning techniques discussed in Section 5.4.2. We also list the ratio between the number of nonzero entries in the pruned matrix and the number of nonzero entries in the original matrix. As the table shows, in some examples the size of the resulting linear system can be reduced drastically, cutting the number of nonzero entries almost in half.

6 Software

We have implemented all algorithms presented in this thesis in several software packages. This chapter is devoted to the presentation of these packages.

In Section 6.1, we present our SageMath package `operator_gb`, which allows to certify ideal membership in the free algebra based on the ability to compute noncommutative Gröbner bases and cofactor representations. It also provides dedicated methods that facilitate proving statements about linear operators, including several heuristics based on the methods discussed in Section 5.3 for finding polynomials of certain form in ideals. We give an overview on the functionality of the package in Section 6.1.1 and discuss implementation details in Section 6.1.2. While the former is published as the appendix of [BHR23], the latter is presented here for the first time. In Section 6.1.2, we also briefly discuss a noncommutative version of Faugère's F4 algorithm [Fau99] for computing Gröbner bases in the free algebra, using linear algebra to perform polynomial reductions. We refer to [Xiu12; Hof20] for a detailed description.

Section 6.2 is dedicated to the latest version of our Mathematica package `OperatorGB`, which was initiated in [Hof20] and provides similar functionality as the SageMath package. We only discuss the methods added in the latest version, consisting primarily of heuristics for finding polynomials of certain form and of a method to automate diagram chases in abelian categories (see also Section 7.3). For all basic commands as well as insights concerning data structures and design choices, we refer to [Hof20, Ch. 6].

Finally, in Section 6.3, we give an overview of our SageMath package `signature_gb` for signature and labelled Gröbner basis computations in the free algebra. Notably, in Section 6.3.2, we present how to combine signature-based techniques with linear algebra style reductions in the noncommutative setting, leading to a signature-based F4 algorithm for computing signature Gröbner bases in the free algebra. Our adaptations are completely

analogous to the case of commutative polynomials discussed in [AP10; Li+19] or [EF17, Sec. 13].

6.1 SageMath package `operator_gb`

In this section, we give an introduction to the functionality provided by the SageMath package `operator_gb`. We assume that the reader is already familiar with SageMath, and otherwise refer to [Sag20]. Furthermore, we also discuss implementation details.

At the time of writing, the package is still under development and not part of the official SageMath distribution. The current version, however, can be downloaded from

`https://github.com/ClemensHofstadler/operator_gb`

and installed as described on the webpage. The code can then be loaded into a SageMath session by the following command.

```
sage: from operator_gb import *
```

6.1.1 Functionality

For now, the package only offers functionality for computations over the coefficient domain $\mathbb{Q}$. In the future, we plan to extend the functionality to other (finite) fields and subsequently also to coefficient rings such as $\mathbb{Z}$.

Certifying operator statements

The basic use-case of the package is to compute proofs of operator statements by certifying ideal membership of noncommutative polynomials. To this end, the package provides the command `certify(assumptions, claim)`, which allows to certify whether a noncommutative polynomial `claim` lies in the ideal generated by a list of polynomials `assumptions`.

For example, to certify that $abc - d$ lies in the ideal generated by $ab - d$ and $c - 1$, proceed as follows.

```
sage: F.<a,b,c,d> = FreeAlgebra(QQ)
sage: assumptions = [a*b - d, c - 1]
sage: proof = certify(assumptions, a*b*c - d)
Computing a (partial) Groebner basis and reducing the claims...

Done! Ideal membership of all claims could be verified!
```

Remark 6.1.1. *Note that noncommutative polynomials are entered using the* `FreeAlgebra` *data structure provided by* SageMath.

The computed `proof` provides a cofactor representation of `claim` in terms of the elements in `assumptions`. More precisely, it is a list of tuples (a_i, j_i, b_i) with terms a_i, b_i in the free algebra and integers j_i such that

$$\texttt{claim} = \sum_{i=1}^{|\texttt{proof}|} a_i \cdot \texttt{assumptions}[j_i] \cdot b_i.$$

The package provides a `pretty_print_proof` command to visualise the proof in form of a string.

```
sage: proof

                                [(1,0,c), (d,1,1)]

sage: pretty_print_proof(proof, assumptions)

                        -d + a*b*c = (-d + a*b)*c + d*(-1 + c)
```

Remark 6.1.2. *The* `certify` *command also checks if the computed cofactor representation is valid over* $\mathbb{Z}$ *as well, that is, if all coefficients that appear are integers. If this is not the case, it produces a warning, but still continues the computation and returns the result.*

It is also possible to give `certify` a list of polynomials as `claim`. In this case, a cofactor representation of each element in `claim` is computed.

```
sage: claims = [a*b*c - d, a*b - c*d]
sage: proof = certify(assumptions, claims)
Computing a (partial) Groebner basis and reducing the claims...

Done! Ideal membership of all claims could be verified!
sage: pretty_print_proof(proof[0], assumptions)

                -d + a*b*c = (-d + a*b)*c + d*(-1 + c)

sage: pretty_print_proof(proof[1], assumptions)

                  a*b - c*d = (-d + a*b) - (-1 + c)*d
```

If ideal membership cannot be verified, `certify` returns `False`. This outcome can occur because of two reasons. Either `claim` is simply not contained in the ideal generated by `assumptions`, or `certify`, which is an iterative procedure, had not been run for enough iterations to verify the ideal membership. To avoid the latter situation, `certify` can be passed an optional argument `maxiter` to determine the maximal number of iterations it is run. By default, this value is set to 10.

```
sage: assumptions = [a*b*a - a*b]
sage: claim = a*b^20*a - a*b^20
sage: certify(assumptions, claim)
Computing a (partial) Groebner basis and reducing the claims...

Starting iteration 5...
Starting iteration 10...
Failed! Not all ideal memberships could be verified.

                                False
```

```
sage: proof = certify(assumptions, claim, maxiter=20)
Computing a (partial) Groebner basis and reducing the claims...

Starting iteration 5...
Starting iteration 10...
Starting iteration 15...
Done! Ideal membership of all claims could be verified!
```

Remark 6.1.3. *Since ideal membership in the free algebra is only semi-decidable, we cannot decide whether the number of iterations of* `certify` *was simply too low or whether* `claim` *is really not contained in the ideal.*

Useful auxiliary functions for treating operator statements

The package provides some auxiliary functions which help in constructing polynomials that commonly appear when treating operator statements.

- `pinv(a, b, a_adj, b_adj)`: generate the polynomials

  `a*b*a - a`, `b*a*b - b`, `b_adj*a_adj - a*b`, `a_adj*b_adj - b*a`

  encoding the four Penrose identities for `a` with Moore-Penrose inverse `b` and respective adjoints `a_adj` and `b_adj`.

- `adj(f)`: compute the adjoint $\texttt{f}^*$ of a polynomial `f`. Each variable `x` is replaced by `x_adj`. Note that all variables `x` and `x_adj` have to be defined as generators of the same `FreeAlgebra`.

- `add_adj(F)`: add to a list of polynomials `F` the corresponding adjoint elements.

Quivers and detecting typos

When encoding operator identities, the resulting polynomials can become quite intricate and it can easily happen that typos occur. To detect typos, it can help to syntactically check if entered polynomials correspond to correctly translated operator identities, respecting

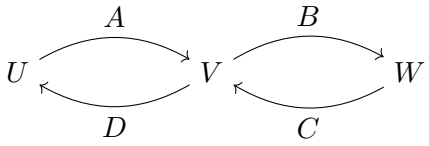


Figure 6.1: Quiver encoding domains and codomains of operators.

the restrictions imposed by the domains and codomains. To this end, the package allows to encode the domains and codomains in form of a directed labelled multigraph, called *(labelled) quiver*.

Computationally, a quiver is given by a list of triplets (u, v, a), where u and v can be any symbols that encode the domain U and the codomain V of the basic operator A and a is the indeterminate representing A. For example, a quiver encoding the situation of operators A, B, C, D on spaces U, V, W as in Figure 6.1, can be constructed as follows.

```
sage: F.<a,b,c,d> = FreeAlgebra(QQ)
sage: Q = Quiver([('U','V',a), ('V','W',b), ('W','V',c), ('V','U',d)])
sage: Q
```

```
Labelled quiver with 3 vertices in the labels {a, b, c, d}
```

One can easily check if a polynomial is compatible with the situation of operators encoded by a quiver.

```
sage: Q.is_compatible(a*b + c*d)
```

```
False
```

```
sage: Q.is_compatible(a*d + c*b)
```

```
True
```

A quiver can be handed as an optional argument to `certify`, which then checks all input polynomials for compatibility with the given quiver and raises an error if required.

```
sage: assumptions = [a*d, c*b]
# typo in the claim, c*b -> b*c
sage: claim = a*d - b*c
sage: certify(assumptions, claim, quiver=Q)
```

```
  ValueError:  The claim a*d - b*c is not compatible with the quiver
```

Gröbner basis computations

Behind the scenes, the `certify` command computes Gröbner bases in the free algebra. In this section, we present the methods of the package that allow to do such computations.

Ideals and monomial orders

The main data structure provided by the package is that of a (two-sided) ideal in the free algebra, called `NCIdeal`. Such an ideal can be constructed from any finite set of noncommutative polynomials.

```
sage: F.<x,y,z> = FreeAlgebra(QQ)
sage: gens = [x*y*z - x*y, y*z*x*y - y]
sage: NCIdeal(gens)
```

```
      NCIdeal (-x*y + x*y*z, -y + y*z*x*y) of Free Algebra on
      3 generators (x, y, z) over Rational Field with x < y < z
```

Attached to an `NCIdeal` also comes a monomial order with respect to which further computations are done. By default, this is a degree lexicographic order, where the indeterminates are sorted as in the parent `FreeAlgebra`. The order of the variables can be individualised by providing a list as an optional argument `order`. Furthermore, by providing a list of lists, block orders (also known as elimination orders) can be defined. The order within each block is still degree lexicographic and blocks are provided in ascending order.

```
sage: NCIdeal(gens, order=[y,x,z])

       NCIdeal (-x*y + x*y*z, -y + y*z*x*y) of Free Algebra on
       3 generators (x, y, z) over Rational Field with y < x < z

sage: NCIdeal(gens, order=[[y,x],[z]])

       NCIdeal (-x*y + x*y*z, -y + y*z*x*y) of Free Algebra on
       3 generators (x, y, z) over Rational Field with y < x « z
```

Gröbner bases and normal forms

For computing Gröbner bases, the class `NCIdeal` provides the method `groebner_basis` with the following optional arguments:

- `maxiter` (default: 10): Maximal number of iterations executed.
- `maxdeg` (default: ∞): Maximal degree of considered ambiguities.
- `trace_cofactors` (default: True): If cofactor representations of each Gröbner basis element in terms of the generators should be computed.
- `criterion` (default: True): If Gebauer-Möller criteria as described in [Xiu12, Sec. 4.2.2] should be used to detect redundant ambiguities.
- `reset` (default: True): If all internal data should be reset. If set to False, this allows to continue previous (partial) Gröbner basis computations.
- `verbose` (default: 0): 'Verbosity' value determining the amount of information about the computational progress that is printed.

In the following, we illustrate how to compute a Gröbner basis of the ideal $I = (xyx - xy, yxxy - y)$.

```
sage: F.<x,y> = FreeAlgebra(QQ)
sage: gens = [x*y*x - x*y, y*x*x*y - y]
sage: I = NCIdeal(gens)
sage: G = I.groebner_basis(); G
```

$$[-xy + xyx, -y + yx^2y, -y + yx, -xy + xy^2,$$
$$-xy + xy^2x, -y + y^2, -y + y^3]$$

We note that the polynomials output by the `groebner_basis` routine are not SageMath noncommutative polynomials but our own `NCPolynomial`s. They provide similar functionality as the native data structure (basic arithmetic, equality testing, coefficient/monomial extraction), but can additionally also store a cofactor representation. In particular, the elements output by the `groebner_basis` command all hold a cofactor representation with respect to the generators of the `NCIdeal`.

```
sage: f = G[2]
sage: pretty_print_proof(f.cofactors(), I.gens())
```

$$-y + yx = yx(-xy + xyx) + (-y + yx^2y) - (-y + yx^2y)x$$

Remark 6.1.4. *To convert an* `NCPolynomial` *back into* SageMath*'s native data structure, our class provides the method* `to_native`*. Conversely, to convert a* SageMath *noncommutative polynomial* `f` *into an* `NCPolynomial`*, one can use* `NCPolynomial(f)`*.*

The package also allows to interreduce a set of `NCPolynomial`s using the command `interreduce`.

```
sage: interreduce(G)
```

$$[-y + yx, -y + y^2]$$

To compute the normal form of an element `f` with respect to the generators of an `NCIdeal`, the class provides the method `reduced_form`. The output of this method

is an `NCPolynomial` `g` holding a cofactor representation of the difference `f-g` with respect to the generators of the `NCIdeal`. The method `reduced_form` accepts the same optional arguments as `groebner_basis`.

```
sage: f = I.reduced_form(y^2 - y); f
```

$$0$$

```
sage: pretty_print_proof(f.cofactors(), I.gens())
```

$$-y + y^2 = (-y + y*x^2*y) - y*x*(-x*y + x*y*x)*y - (-y + y*x^2*y)*y - y*x*(-x*y + x*y*x)*x*y + (-y + y*x^2*y)*x^2*y$$

```
sage: I.reduced_form(y^2)
```

$$y$$

Heuristics for finding polynomials of certain form

One of the main functionalities provided by the package are dedicated heuristics for systematically searching for polynomials of certain form in an `NCIdeal`. To this end, the class `NCIdeal` provides the method `find_equivalent_expression(f)`, which searches for elements of the form `f - g`, with arbitrary `g`, in an `NCIdeal`. It accepts the following optional arguments:

- All optional arguments that also `groebner_basis` accepts with the same effects.
- `order`: A monomial order with respect to which the computation is executed. The argument has to be provided like a custom order when defining an `NCIdeal`.
- `heuristic` (default: 'groebner'): Determines the heuristic used. Available are
  - 'naive': Try exhaustively all monomials `m` up to a degree bound and check if `f - m` is in the ideal.

  - 'groebner': Enumerate a Gröbner basis and search in the Gröbner basis for suitable elements containing `f`.
  - 'subalgebra': Intersect the two-sided ideal with a subalgebra to find suitable elements.
  - 'right-ideal'/'left-ideal': Intersect the two-sided ideal with a right/left ideal to find suitable elements.

- `prefix` (default: None): A term `p` providing the prefix of `g`, i.e., the heuristic looks for elements of the form `f - p*h` with arbitrary `h` (required for heuristic 'right-ideal').
- `suffix` (default: None): A term `s` providing the suffix of `g`, i.e., the heuristic looks for elements of the form `f - h*s` with arbitrary `h` (required for heuristic 'left-ideal').
- `degbound` (default: 5): Some heuristics only compute up to a fixed degree bound. This argument allows to change this degree bound.
- `quiver` (default: None): Use a quiver to restrict the search space only to polynomials that are compatible with this quiver.

```
sage: F.<a,b,c,d> = FreeAlgebra(QQ)
sage: gens = [a*b*a-a, b*a*b-b, a*b-c*d, b*a-d*c, c*d*c-c, d*c*d-d]
sage: I = NCIdeal(gens)
sage: I.find_equivalent_expression(a*b)

                              [-a*b + c*d]

sage: I.find_equivalent_expression(a*b, heuristic='naive', suffix=b)

                            [a*b - c*d*a*b]

sage: I.find_equivalent_expression(a*b, heuristic='right-ideal',
....: prefix=a*b)

                  [- a*b + a*b*c*d, - a*b + a*b*a*b]
```

Additionally, the class `NCIdeal` provides methods for applying cancellability.

- `I.apply_left_cancellability(a, b)`: Search for elements of the form `a*b*f` in `I` and return `b*f`.
- `I.apply_right_cancellability(a, b)`: Search for elements of the form `f*a*b` in `I` and return `f*a`.

Both methods can be given an optional argument `heuristic` to determine the used search heuristic. Available are 'subalgebra', 'one-sided', and 'two-sided' (default: 'subalgebra').

```
sage: I.apply_left_cancellability(c, a)
```

$$[-a + a*b*a, -a^2 + a*d*c*a]$$

```
# verify ideal membership to check correctness of result
sage: I.reduced_form(c*(-a^2 + a*d*c*a))
```

$$0$$

```
sage: I.apply_right_cancellability(a*b, c*d, heuristic='two-sided',
....: maxiter=5)
```

$$[-a*b + a*b*a*b, -a*b + c*d*a*b]$$

```
# verify ideal membership to check correctness of result
sage: I.reduced_form((-a*b + c*d*a*b)*c*d)
```

$$0$$

6.1.2 Implementation details

Noncommutative F4 algorithm

The package `operator_gb` implements a noncommutative version of Faugère's F4 algorithm [Fau99] for computing Gröbner bases in the free algebra. In contrast to Buchberger's algorithm, where only one S-polynomial is reduced at a time, the main idea of the F4 algorithm is to reduce several S-polynomials by a list of polynomials simultaneously. This is done by representing polynomials in terms of a matrix and computing a reduced row echelon form of this matrix. We refer to [Xiu12; Hof20] for further information and a detailed description of this algorithm in the noncommutative setting.

The most costly step of a Gröbner basis computation with the F4 algorithm is the matrix normal form computation. Thus, special elimination techniques were developed in the commutative setting to improve these computations by exploiting the special structure of the matrices that arise in Gröbner basis computations (sparse, rank deficient, almost block triangular). The most notable such technique is the Faugère-Lachartre elimination algorithm [FL10], which can also be applied in the noncommutative setting. The algorithm is illustrated in Figure 6.2.

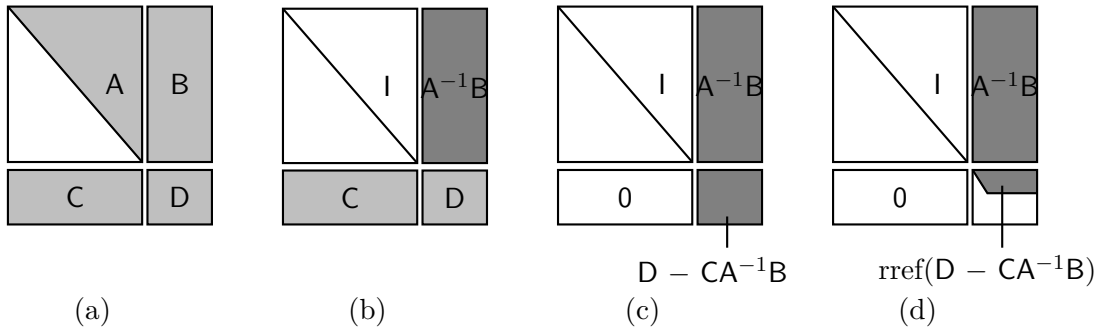


Figure 6.2: Faugère-Lachartre elimination.

The idea of the Faugère-Lachartre elimination is to permute all already visible pivots into an upper triangular submatrix A by swapping rows and columns. The rest of the matrix is divided into three blocks B, C, and D, as shown in (a). Since A is invertible, a reduction of these rows yields an identity and $A^{-1}B$, see (b). Then the identity part is used to zero out the submatrix C and D is updated accordingly, see (c). Finally, the lower right block

is reduced to reduced row echelon form (for example, using the same procedure recursively or by standard Gaussian elimination).

Typically, the submatrices A and C are very sparse, allowing for efficient linear algebra techniques to speed up the computations. Furthermore, the upper triangular structure of A allows to compute the product $A^{-1}B$ by backward substitution.

The package `operator_gb` implements the Faugère-Lachartre elimination as described above. Furthermore, it uses the noncommutative versions of the Gebauer-Möller criteria described in [Xiu12, Sec. 4.2.2] for detecting redundant ambiguities.

Monomials

The most fundamental objects that a Gröbner basis algorithm has to work with are monomials, in our setting, that is, words from the free monoid. In our implementation, monomials are represented by strings. This is a natural choice since strings are optimised for precisely the most crucial operations that are needed in Gröbner basis computations: multiplication of monomials (string concatenation) and divisibility tests (substring checking). We have also tried other data structures more similar to the commutative exponent vectors but their performance could not compare to strings.

When transforming the input into our data structure, we do not use the variable names provided by the user. Instead, each variable is first mapped to a single ASCII character. If more than 128 (the number of ASCII characters) indeterminates are needed, each variable is mapped to a unique combination of two ASCII characters. The use of more than $128^2 = 16\,384$ indeterminates is not supported (and probably also not needed). In this way, the internal representations are kept as compact as possible while still giving the user the freedom to enter arbitrary names for the indeterminates.

Divisibility tests of monomials occur in several places in Gröbner basis algorithms. The most apparent example is during polynomial reduction, when searching for a basis element whose leading monomial divides the term being reduced. Another example is the computation of inclusion ambiguities. In both cases, we have a varying monomial (*the needle*) of which we want to find a divisor among a fixed – typically large – set of divisor candidates (*the haystack*). In the examples above, the haystack is the set of leading monomials of the current partial Gröbner basis.

Testing divisibility of the needle by all elements in the haystack sequentially is linear in the size of the haystack and can become very expensive. Therefore, it makes sense to use dedicated algorithms for multi-pattern string matching. We use the Aho-Corasick algorithm [AC75] for this, in form of the open source implementation `pyahocorasick` [Muł22].

The Aho-Corasick algorithm first transforms the haystack into a finite-state machine that resembles a prefix tree with additional edges that eliminate the need for backtracking. By using this data structure, the algorithm can match the needle against all elements in the haystack simultaneously. This results in a search-complexity linear in the length of the strings and independent of the size of the haystack. Building the finite-state machine is still linear in the size of the haystack, however this cost is negligible since this operation has to be done very rarely compared to the number of searches – namely only when new elements are added to the Gröbner basis.

Besides making divisibility tests more efficient, the Aho-Corasick data structure can also be used to improve the computation of overlap ambiguities. Generating overlap ambiguities requires comparing all suffixes B of a new leading monomial AB against all prefixes of the already known leading monomials. The prefix tree of the Aho-Corasick algorithm allows to do this in an efficient way. By passing a suffix B through the prefix tree, all overlap ambiguities with overlap B can be computed simultaneously. To compute all overlap ambiguities between AB and the known leading monomials, we also have to compare all prefixes A of AB against all suffixes of the other elements. To do this efficiently as well, we keep – besides the prefix tree – also a suffix tree of the known leading monomials. Then the remaining overlap ambiguities can be computed by passing all prefixes A of AB through the suffix tree. This reduces the complexity of computing all overlap ambiguities from proportional in the size of the haystack to linear in the length of the strings.

Polynomials

The main algorithms of the package rely on linear algebra for polynomial reduction, which has the advantage that only very minimal polynomial arithmetic is required. The only operations that need to be available for polynomials are multiplication by a monomial and by a constant, and retrieving the leading monomial. In particular, polynomial addition is not required, as all polynomial reductions are performed implicitly during the matrix normal form computation. This allows to implement polynomials in a very simple fashion.

In our implementation, each polynomial consists of a list of coefficients and a list of monomials sorted so that the ith coefficient belongs to the ith monomial. Additionally, we keep a separate reference to the leading monomial. While being very simple, this data structure allows to efficiently do all relevant operations.

6.2 Mathematica package `OperatorGB`

The MATHEMATICA package `OperatorGB` essentially provides the same functionality as the SAGEMATH package. Most prominently, it also offers a `Certify` command for certifying ideal membership in the free algebra based on the computation of cofactor representations. We refer to [Hof20, Ch. 6] for further information on the `Certify` command and related functionality, as well as for remarks on the design choices made when implementing these methods. In the following, we describe some new methods that have been added to the package recently. The latest version of the package can be obtained from

`https://github.com/ClemensHofstadler/OperatorGB`.

The package can be loaded into MATHEMATICA by placing it in the current working directory and executing the following line.

```
In[1]:= << OperatorGB.m

        Package OperatorGB version 1.4.2
        Copyright 2019, Institute of Mathematics, University of Kassel
        by Clemens Hofstadler, clemens.hofstadler@mathematik.uni-kassel.de
```

6.2.1 Functionality

Heuristics for finding polynomials

Since version 1.3, the package also implements the heuristics described in Section 5.3 for finding elements of certain form in one- or two-sided ideals. In particular, the package provides the following methods.

- `Intersect[I, J, MaxIter -> 1]`: Enumerate a Gröbner basis of the intersection of the two-sided ideals `I` and `J`. The optional argument `MaxIter` determines the maximal number of iterations of the underlying Gröbner basis computation (default: 1).

```
In[2]:= SetUpRing[{x, y, z}];
        I = {x ** y};
        J = {y ** z};
        Intersect[I, J, MaxIter -> 3]

Out[2]= {x ** y ** z, y ** z ** x ** y, x ** y ** y ** z,
        -y ** z ** x ** x ** y,  -y ** z ** y ** x ** y,
        -y ** z ** z ** x ** y, x ** y ** y ** y ** z}
```

- `IntersectSubalgebra[I, S, MaxIter-> 5]`: Enumerate elements in the intersection of the two-sided ideal `I` with the subalgebra `S`. The optional argument `MaxIter` determines the maximal number of iterations of the underlying Gröbner basis computation (default: 5).

```
In[3]:= SetUpRing[{x, y}];
        I = {x ** x - 3 x, x ** y ** x};
        S = {x};
        IntersectSubalgebra[I, S, MaxIter -> 3]

Out[3]= {-3 x + x ** x}
```

- `IntersectRightIdeal[I, I_ρ, Q, MaxDeg -> 5]`: Enumerate a right Gröbner basis of the intersection of the two-sided ideal `I` with the right ideal $\mathtt{I}_\rho$. Since a generating set of this intersection is typically infinite, we restrict the output only to polynomials that are compatible with the quiver `Q`. The optional argument `MaxDeg` determines an upper bound on the lengths of the monomials w used in Proposition 5.3.4 to transform a (partial) two-sided Gröbner basis into a (partial) right Gröbner basis. Hence, this argument yields a stopping criterion (default: 5).

```
In[4]:= SetUpRing[{x, y, z}];
        I = {x ** y};
        I_ρ = {x, z};
```

```
(* We set Q to be the trivial quiver with only one vertex
over all variables that appear in our ideals. *)
Q = TrivialQuiver[{x, y, z}];
IntersectRightIdeal[I, I_ρ, Q, MaxDeg -> 2]
```

Out[4]=
```
{x ** y, x ** x ** y, z ** x ** y, x ** x ** x ** y,
x ** z ** x ** y, z ** x ** x ** y, z ** y ** x ** y,
z ** z ** x ** y}
```

As can be seen, the number of generators of this intersection grows quite fast. Restricting only to generators where the variable `z` cannot appear to the left of `x` or `y`, drastically reduces the number of generators.

In[5]:=
```
Q = {{x, 1, 1}, {y, 1, 1}, {z, 2, 1}};
IntersectRightIdeal[I, I_ρ, Q, MaxDeg -> 2]
```

Out[5]=
```
{x ** y, x ** x ** y, x ** x ** x ** y}
```

- `IntersectLeftIdeal[I, I_λ, Q, MaxDeg -> 5]`: Enumerate a left Gröbner basis of the intersection of the two-sided ideal `I` with the left ideal I_λ. The other arguments for this method serve the same purpose as in `IntersectRightIdeal`.

- `Hom[cofactors, I, MaxIter, A : I_n]`: Enumerate a Gröbner basis of the homogeneous part hom(`I`) of the ideal `I` with respect to the grading defined by the matrix `A`. The ith row of `A` specifies the degree of x_i, where the x_i are ordered as they appear in `SetUpRing`. By default, `A` is the identity matrix I_n. The argument `MaxIter` determines an upper bound on the number of iterations of the underlying Gröbner basis computation that is done during this procedure. Additionally, for each polynomial `f` in the output, a cofactor representation of `f` in terms of the generators of `I` is saved in the list `cofactors`.

In[6]:=
```
SetUpRing[{x, y, z}]
I = {x ** y - y, x + y ** x, z ** y ** x};
cofactors = {};
Hom[cofactors, I, 3]
```

Out[6]=
```
{z ** x, z ** y, -x ** y ** x + y ** x ** x,
-x ** y ** y + y ** x ** y}
```

Using the degree matrix

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

yields

```
In[7]:= cofactors = {};
        A = {{2, 0}, {-1, 0}, {0, 1}};
        Hom[cofactors, I, 3, A]

Out[7]= {-x ** y ** y ** x + y ** x ** y ** x, z ** y,
        -x ** y ** x + y ** x ** x, -x ** y ** y + y ** x ** y,
        -x + x ** y ** y ** x, -y + x ** y ** y ** y, z ** x}
```

- `Mon[cofactors, I, MaxIter, OneSided -> "right"]`: Enumerate a one-sided Gröbner basis of mon(`I`), the one-sided ideal generated by all monomials in the one-sided ideal `I`. The arguments `cofactors` and `MaxIter` serve the same purpose as in `Hom`. The optional value `OneSided` determines whether `I` (and consequently also the output mon(`I`)) is considered as a right ideal or as a left ideal (default: `"right"`).

```
In[8]:= I = {x ** x ** y - y ** x, x ** x + x ** y - x,
           x ** y ** x ** y - y ** x ** y, x ** y ** x};
        cofactors = {};
        Mon[cofactors, I, 5]

Out[8]= {x ** y ** x, y ** x ** y, x ** x ** y ** y,
        x ** x ** x ** y ** y}

In[9]:= (* Now consider I as left ideal *)
        cofactors = {};
        Mon[cofactors, I, 5, OneSided -> "left"]

Out[9]= {x ** y ** x}
```

Remark 6.2.1. *Thus far, the computation of cofactor representations is only supported for* `Hom` *and* `Mon`.

Some of the commands discussed above require the ideal to be one-sided. To this end, the package provides the methods `ToRightGB[I, d, X, Q]` and `ToLeftGB[I, d, X, Q]` that take a two-sided ideal `I`, a quiver `Q`, a set of variables `X`, and an integer `d` as input and enumerate a right, respectively left, Gröbner basis of `I`. Since these Gröbner bases are typically infinite, the variable `d` gives an upper bound on the lengths of the monomials w used in Proposition 5.3.4 to transform a (partial) two-sided Gröbner basis into a (partial) one-sided Gröbner basis. Additionally, we restrict ourselves to only computing generators that are compatible with the quiver `Q`. This can drastically speed up the computation. The set `X` determines which indeterminates may appear in the generators, that is, computations are done in $\mathbb{Q}\langle \mathtt{X} \rangle$.

We illustrate the usage of these commands to compute monomials in a two-sided ideal `I`. To this end, we first translate the two-sided generating set into a one-sided generating set. Since we do not want to restrict our computations in any way, we set `Q` to be the trivial quiver and `X` to be the set of all variables.

```
In[10]:= SetUpRing[{x, y}]
         I = {x ** x ** y - y ** x, x ** x + x ** y - x,
          x ** y ** x ** y - y ** x ** y, x ** y ** x}
         X = {x, y};
         Q = TrivialQuiver[X];
         rightGens = ToRightGB[I, 3, X, Q];
         cofactors = {};
         Mon[cofactors, rightGens, 4];

Out[10]= {y ** x, x ** y ** x}
```

Diagram chasing

The package also provides a simple interface for performing diagram chasing proofs in abelian categories. More precisely, it offers the method `DiagramChase` that takes as input an ideal `I` and an integer `n`. The ideal `I` is generated by polynomials describing some diagram for which a proof by diagram chasing shall be found. Additionally, the user can provide the optional arguments `ExactAt`, `Mono`, `Epi`, `Algorithm`, and `MaxIter`. `ExactAt` is a list of pairs of variables determining which symbols represent exact sequences of

morphisms in the diagram. Similarly, `Mono` and `Epi` are lists of variables that determine which morphisms in the diagram are monomorphisms and epimorphisms respectively. By default, `ExactAt`, `Mono` and `Epi` are all empty. The optional argument `Algorithm` determines which of the heuristics for finding polynomials discussed previously is used for the computations. The options here are `"one-sided"`, `"two-sided"`, and `"subalgebra"` (default: `"one-sided"`). The optional argument `MaxIter` determines how many iterations of the underlying Gröbner basis computations are executed (default: 3).

We refer to Section 7.3 for an illustration of this functionality.

6.3 SageMath package `signature_gb`

We give a brief overview of the SageMath package `signature_gb` for signature and labelled Gröbner basis computations in the free algebra. Since the data structures used are the same as in the case of the `operator_gb` package (see Section 6.1.2), we omit this information here. Instead, after presenting the functionality of the package, we describe how to combine the F4 algorithm with signature-based techniques, as this is the algorithm used in our implementation.

At the time of writing, the package is still under development, but a beta version can be obtained from

`https://github.com/ClemensHofstadler/signature_gb`

and installed as described on the webpage.

6.3.1 Functionality

The basic data structure provided by the package is that of a `LabelledModule`. It can be constructed from a finite list of SageMath noncommutative polynomials as follows. We note that, so far, only polynomials with rational coefficients are supported. Furthermore, variable names are restricted to single (lower- and uppercase) characters.

```
sage: from signature_gb import *
Package signature_gb version 0.1.0 (beta version)
by Clemens Hofstadler, clemens.hofstadler@mathematik.uni-kassel.de

sage: F.<a,b,c,d> = FreeAlgebra(QQ)
sage: gens = [a*b*a-a, b*a*b-b, a*b-c*d, b*a-d*c, c*d*c-c, d*c*d-d]
sage: M = LabelledModule(gens,[a,b,c,d]); M

        Labelled module generated by
         -a + a*b*a^[e_1], -b + b*a*b^[e_2], a*b - c*d^[e_3],
         b*a - d*c^[e_4], -c + c*d*c^[e_5], -d + d*c*d^[e_6]
        Monomial order:  a < b < c < d, Signature order:  dpot
```

The second argument provides the monomial order with respect to which the computations are executed. A list yields a degree lexicographic order. By default, the module order is degree-over-position-over-term (dpot). It can be changed to degree-over-term-over-position (dtop) as follows.

```
sage: LabelledModule(gens,[a,b,c,d],signature_order='dtop')

        Labelled module generated by
         -a + a*b*a^[e_1], -b + b*a*b^[e_2], a*b - c*d^[e_3],
         b*a - d*c^[e_4], -c + c*d*c^[e_5], -d + d*c*d^[e_6]
        Monomial order:  a < b < c < d, Signature order:  dtop
```

The main functionality provided by a `LabelledModule` is computing signature and labelled Gröbner bases. A signature Gröbner basis can be enumerated as follows.

```
sage: M = LabelledModule(gens,[a,b,c,d])
sage: G, H = M.signature_GB(100)
18 ambiguities in total (computation took 0.00104)
10 critical pairs were generated.
```

```
22 ambiguities in total (computation took 0.00012)
10 critical pairs were generated.
All critical pairs were reduced to 0.

sage: G

                    [poly:  -a + a*b*a, sig :  e1,
                    poly:  -b + b*a*b, sig :  e2,
                    poly:  -a*b + c*d, sig :  e3,
                    poly:  -b*a + d*c, sig :  e4,
                    poly:  -c + c*d*c, sig :  e5,
                    poly:  -d + d*c*d, sig :  e6,
                    poly:  -c + a*b*c, sig :  e3*c,
                    poly:  -d + d*a*b, sig :  d*e3,
                    poly:  -d + b*a*d, sig :  e4*d,
                    poly:  -c + c*b*a, sig :  c*e4]
```

Remark 6.3.1. *By default, the package provides some information on the computational progress. In particular, it lists the number of ambiguities and S-polynomials (in the algorithm called* `critical pairs`*) computed in every iteration.*

The commands above run 100 iterations of the signature-based algorithm and outputs a (partial) signature Gröbner basis `G` and a (partial) Gröbner basis `H` of the leading term module of the syzygy module. To compute a signature basis up to some fixed signature, a `sig_bound` can be provided in form of a positive integer N. Then the algorithm computes a signature basis up to degree N (if the number of iterations is chosen large enough).

```
sage: M = LabelledModule(gens,[a,b,c,d])
sage: G,H = M.signature_GB(100,sig_bound=3)
18 ambiguities in total (computation took 0.00018)
10 critical pairs were generated.
All critical pairs were reduced to 0.
```

To reconstruct a (partial) labelled Gröbner basis and a (partial) basis of the syzygy module, run the following commands in the given order.

```
sage: G,H = M.signature_GB(100)
sage: G2 = M.reconstruct_labelled_basis()
sage: H2 = M.reconstruct_syzygies()
sage: G2
```

```
[poly :  -a + a*b*a, label :  e1,
poly :  -b + b*a*b, label :  e2,
poly :  -a*b + c*d, label :  -e3,
poly :  -b*a + d*c, label :  -e4,
poly :  -c + c*d*c, label :  e5,
poly :  -d + d*c*d, label :  e6,
poly :  -c + a*b*c, label :  e5 + e3*c,
poly :  -d + d*a*b, label :  e6 + d*e3,
poly :  -d + b*a*d, label :  e6 + e4*d,
poly :  -c + c*b*a, label :  e5 + c*e4]
```

Once a labelled Gröbner basis is reconstructed, a `LabelledModule` also provides the possibility to test ideal membership of noncommutative polynomials. If ideal membership can be verified, it outputs a cofactor representation.

```
sage: F.<a,b,c,d,e> = FreeAlgebra(QQ)
sage: gens = [1-a*b,1-b*a,a*e*a-a, e*a*e-e, a*e-c*d, e*a-d*c,
....: c*d*c-c, d*c*d-d]
sage: M = LabelledModule(gens,[a,b,c,d,e])
sage: G,H = M.signature_GB(100)
sage: M.reconstruct_labelled_basis()
sage: M.membership_test(b - e)
Membership test SUCCESSFUL.
```

```
poly :  b - e, label :  b*e1 - e2*e - b*e3*b - b*a*e*e1
```

6.3.2 Signature-based F4 algorithm

The algorithms for computing signature and labelled Gröbner bases presented in Chapter 3 are all phrased in Buchberger style analogous to Algorithm 1. However, as noticed in the commutative case [AP10; Li+19], signature-based techniques can also be combined with the use of linear algebra for polynomial reduction, allowing to reduce S-polynomials in batches rather than individually, yielding a signature-based F4 algorithm.

The package `signature_gb` implements such a version of the F4 algorithm for computing signature Gröbner bases in the free algebra. In the following, we give a very brief description of this algorithm. We refer to [EF17, Sec. 13] for a more thorough explanation on how to include signature-based techniques into the F4 algorithm in the commutative case, and note that the adaptations to the noncommutative setting are straightforward.

Such an algorithm selects several ambiguities at once, forms the corresponding S-polynomials, and subsequently organises them, together with their regular reducers, in a matrix. This is done by writing the coefficients of f, for each selected signature polynomial $f^{(\sigma)}$, as the entries of a row in A. In this way, every column of A corresponds to a monomial. Additionally, the row corresponding to $f^{(\sigma)}$ is labelled by the signature σ. To ensure that a reduction of A corresponds to performing regular sig-reductions, the following two conditions have to be satisfied:

1. The columns of A are sorted in decreasing order with respect to the monomials they represent.

2. The rows of A are sorted in increasing order with respect to their signature label.

Then, performing one-sided Gaussian elimination on A, where a row can only be used to eliminate rows below it that have strictly larger signature, corresponds precisely to regular sig-reducing the selected signature polynomials. In the resulting matrix, each nonzero row has a unique pivot, but its appearance may, in fact, not be triangular. We call this normal form a *sig-echelon form* of A. Finally, the rows of this reduced matrix whose first entry has changed during this normal form computation are added as new basis elements or newly identified syzygy signatures, depending on whether the reduced row is zero or not.

Signature-based Faugère-Lachartre elimination

One crucial optimisation of the F4 algorithm is to use Faugère-Lachartre elimination for the matrix normal form computations. However, the first step of this technique requires to freely swap rows (see (a) in Figure 6.2). This is not allowed in signature-based computations. To exploit this technique also for the computation of signature Gröbner bases, one can divide the matrix into blocks and perform Faugère-Lachartre elimination block-wise. This procedure, which is also described in [Li+19], is illustrated in Figure 6.3.

First, as usual for signature-based computations, the rows of the matrix are sorted by increasing signature. Then they are split into two consecutive blocks, see (a). If several rows share the same signature, one has to ensure that the last row of the first block has a signature strictly smaller than the first row of the second block. Then one-sided reductions are performed on the first block to obtain a sig-echelon form, yielding (b). At this point, all polynomials relevant for the signature Gröbner basis are collected from Block 1. Next, the reduced first block is analysed and all visible pivots are swapped into an upper triangular submatrix A. The remaining matrix is divided into three blocks B, C, and D as in the usual Faugère-Lachartre elimination, see (c). Note that this rearrangement of the first block does not affect the sig-reductions that will be performed on the second block since all rows in Block 1 have strictly smaller signature than those in Block 2. Then the usual Faugère-Lachartre elimination is performed, see (d), where for the final reduction of the lower right block one-sided reductions have to be used (or this procedure is used recursively) to obtain a sig-echelon form.

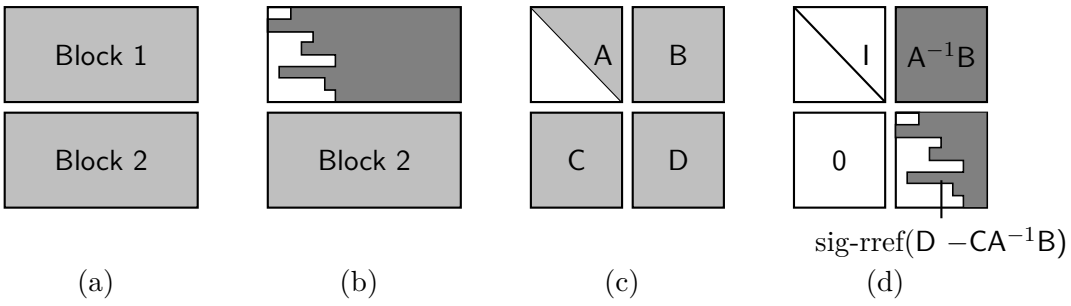


Figure 6.3: Signature-safe adaptation of Faugère-Lachartre elimination.

7 Applications

7.1 Moore-Penrose case study

Using our framework and the functionality provided by our software package `operator_gb`, we have successfully automated the proofs of a variety of theorems concerning the Moore-Penrose inverse, ranging from classical facts in the Handbook of Linear Algebra [Hog13, Sec. I.5.7] over important characterisations of the Moore-Penrose inverse $(AB)^\dagger$ of a product of linear operators A, B on Hilbert spaces [DD10] to very recent improvements [Cve+21] of a classical result by Hartwig that were found with the help of our software. We have assembled a Jupyter notebook containing the automated proofs of all statements, which is available at

`https://cocalc.com/georeg/Moore-Penrose-case-study/notebook`.

For our case study, we considered the first 25 facts in the section on the Moore-Penrose inverse in the Handbook of Linear Algebra [Hog13, Sec. I.5.7]. Among these 25 statements, we found that five cannot be treated within the framework, as they contain properties that cannot be expressed in terms of identities of operators, such as properties of the matrix entries or statements that require induction. Additionally, three statements can only be partially handled for the same reason. The remaining 17 statements, along with those parts of the three statements mentioned before that can be treated within the framework, can all be translated into polynomial computations and proven fully automatically with the help of our software. The corresponding polynomial computations take place in ideals generated by up to 70 polynomials in up to 18 indeterminates. The proof of each statement takes less than one second and the computed cofactor representations, certifying the required ideal memberships, consist of up to 226 terms.

As part of our case study, we also examined Theorems 2.2 – 2.4 in [DD10], which provide several necessary and sufficient conditions for the reverse order law $(AB)^\dagger = B^\dagger A^\dagger$ to hold, where A, B are bounded linear operators on Hilbert spaces with closed ranges. Our software can automatically prove all these statements in less than five seconds altogether, yielding algebraic proofs that consist of up to 279 terms. We note that, in contrast to the original proofs in [DD10], which rely on matrix forms of bounded linear operators that are induced by some decompositions of Hilbert spaces, our proofs do not require any structure on the underlying spaces except a certain cancellability assumption and the property that range inclusions can be translated into factorisations as discussed in Section 5.2.4. This implies that our proofs generalise the results from bounded operators on Hilbert spaces to morphisms in arbitrary preadditive semicategories meeting these assumptions.

Finally, our case study contains fully automated proofs of Theorem 2.3 and 2.4 in our joint work [Cve+21], which provide necessary and sufficient conditions for the triple reverse order law $(ABC)^\dagger = C^\dagger B^\dagger A^\dagger$ to hold, where A, B, C are elements in a ring with involution. These theorems can be considered as significant improvements of a classical result by Hartwig [Har86]. We discuss them in more detail in the succeeding section.

7.2 Improvements of Hartwig's triple reverse order law

For a classical group inverse, it is easy to show that $(ab)^{-1} = b^{-1}a^{-1}$ for all group elements a, b. In the setting of generalised inverses, this *reverse order law* need no longer hold unconditionally. The *reverse order law problem*, originally posed by Greville [Gre66], asks for necessary and sufficient conditions for the reverse order law to hold. Greville considered it in the case of the Moore-Penrose inverse of the product of two matrices. Namely, for given matrices A, B such that AB is defined, he showed

$$(AB)^\dagger = B^\dagger A^\dagger \iff \mathcal{R}(A^*AB) \subseteq \mathcal{R}(B) \text{ and } \mathcal{R}(BB^*A^*) \subseteq \mathcal{R}(A^*),$$

where $\mathcal{R}(M)$ denotes the range of a matrix M.

This was followed by further research on this subject branching in several directions:

- for products of more than two matrices;
- for different classes of generalised inverses;

- in different settings (operator algebras, C^*-algebras, rings, etc.);

We refer to [CW17, Ch. 2] for a textbook exposition summarising these results and for further references.

One of the first to be inspired by Greville's result was Hartwig [Har86], who studied the reverse order law for the Moore-Penrose inverse of a product of three matrices. He gave the following characterisation. In the following, a square matrix M is called *EP* if $\mathcal{R}(M) = \mathcal{R}(M^*)$.

Theorem 7.2.1. *Let A, B, C be complex matrices such that ABC is defined and let $P = A^\dagger ABCC^\dagger$, $Q = CC^\dagger B^\dagger A^\dagger A$. The following conditions are equivalent:*

(i) $(ABC)^\dagger = C^\dagger B^\dagger A^\dagger$;

(ii) $PQP = P$, $QPQ = Q$, *and both of* A^*APQ *and* $QPCC^*$ *are Hermitian;*

(iii) $PQP = P$, $QPQ = Q$, *and both of* A^*APQ *and* $QPCC^*$ *are EP;*

(iv) $PQP = P$, $\mathcal{R}(A^*AP) = \mathcal{R}(Q^*)$, *and* $\mathcal{R}(CC^*P^*) = \mathcal{R}(Q)$;

(v) $PQ = (PQ)^2$, $\mathcal{R}(A^*AP) = \mathcal{R}(Q^*)$, *and* $\mathcal{R}(CC^*P^*) = \mathcal{R}(Q)$;

Since then, this statement has been generalised to other settings such as algebras of bounded linear operators [DD14] or C^*-algebras [Mil18]. In both papers, results analogous to Hartwig's paper were obtained, but with the additional conditions of regularity of all three elements and their products. In our joint work [Cve+21], we present several significant improvements of Hartwig's triple reverse order law, using the framework developed in Chapter 4 and the software package `operator_gb`. The improvements include the following generalisations:

- We consider the problem in rings with involution, which is a more abstract setting than what has been considered in the literature so far.
- We relax conditions (iv) and (v) in the original result of Hartwig (Theorem 7.2.1), by replacing the respective equalities of ranges assumed in both of these conditions with appropriate inclusions of ranges.

- Compared to the results for algebras of operators and C^*-algebras, we significantly reduce the set of starting assumptions upon which these results are based by dropping certain regularity conditions.
- We also generalise the result by showing that $B^\dagger$ can be replaced by an arbitrary element $\widetilde{B}$ that need not be related to B in any way. In this way, the regularity assumption of the element B is dropped.

The main setting that we consider is that of a (noncommutative) ring R with a unit $1 \neq 0$ and an involution $a \mapsto a^*$ satisfying

$$(a^*)^* = a, \qquad (a+b)^* = a^* + b^*, \qquad (ab)^* = b^*a^*.$$

As in other settings, an element $b \in R$ is the unique *Moore-Penrose inverse* of an element $a \in R$ if it satisfies the Penrose identities

$$aba = a, \qquad bab = b, \qquad (ab)^* = ab, \qquad (ba)^* = ba.$$

We denote the Moore-Penrose inverse of a by $a^\dagger$. Furthermore, an element $a \in R$ is *right $*$-cancellable* if, for all $z \in R$, $zaa^* = 0$ implies $za = 0$, and *EP* if $aR = a^*R$.

We only list one of the improvements of Theorem 7.2.1 below and refer for the other results to [Cve+21].

Theorem 7.2.2 ([Cve+21, Thm. 2.3])**.** *Let $a, b, c \in R$ be such that a, c are Moore-Penrose invertible. Let $p = a^\dagger abcc^\dagger$ and $q = cc^\dagger \widetilde{b} a^\dagger a$, for $\widetilde{b} \in R$. The following conditions are equivalent:*

(i) abc is Moore-Penrose invertible and $(abc)^\dagger = c^\dagger \widetilde{b} a^\dagger$;

(iv) $pqp = p$, $a^*apR \supseteq q^*R$, and $cc^*p^*R \subseteq qR$;

(v) abc is right $*$-cancellable, $pq = (pq)^2$, $a^*apR \supseteq q^*R$, and $cc^*p^*R \subseteq qR$;

(vi) $qpq = q$, $a^*apR \supseteq q^*R$, and $cc^*p^*R \subseteq qR$;

In the following, we discuss different aspects and use cases of using the framework developed in Chapter 4 to prove Hartwig's original result and its improvements.

First, we focus on the implication $(v) \Rightarrow (i)$ in Theorem 7.2.1: if $PQ = (PQ)^2$, $\mathcal{R}(A^*AP) = \mathcal{R}(Q^*)$, and $\mathcal{R}(CC^*P^*) = \mathcal{R}(Q)$, then $(ABC)^\dagger = C^\dagger B^\dagger A^\dagger$, which can be proven using Theorem 5.1.7.

As discussed in Section 5.2.4, the four inclusions of ranges are equivalent to the following identities for some operators U_1, U_2, V_1, V_2:

$$A^*AP = Q^*V_1, \qquad A^*APV_2 = Q^*, \qquad CC^*P^* = QU_1, \qquad CC^*P^*U_2 = Q. \tag{7.1}$$

For each Moore-Penrose inverse $A^\dagger, B^\dagger, C^\dagger, (ABC)^\dagger$ we have the four defining identities, and finally, we also have the identity $PQ = (PQ)^2$ as an assumption.

Translating these identities into polynomials, we introduce an indeterminate for each basic operator. Moreover, for each indeterminate, we introduce another indeterminate representing the adjoint of the corresponding operator. In total, this amounts to 22 indeterminates. Similarly, each identity of operators is translated into two polynomials, one for the identity itself and one for its adjoint. Thereby, we obtain a set F of 34 noncommutative polynomials with integer coefficients representing the assumptions. The claim corresponds to the polynomial $f = m^\dagger - c^\dagger b^\dagger a^\dagger$, where $m^\dagger$ is the indeterminate introduced for $(ABC)^\dagger$.

Then we can use our software package `operator_gb` to show that f lies in the ideal generated by the polynomials of F. The cofactor representation certifying this ideal membership was computed in less than 5 seconds and has 937 terms. Since this cofactor representation only contains integer coefficients, the ideal membership also holds over the integers. Hence, by Theorem 5.1.7, this proves that $(ABC)^\dagger = C^\dagger B^\dagger A^\dagger$ holds under the given assumptions. In fact, the implication $(v) \Rightarrow (i)$ is proven in any preadditive semicategory in which it can be formulated.

By investigating the cofactor representations that are computed by our software package, one can check which assumptions of a theorem are really needed, and which are in fact redundant. For instance, it turns out that the first and last identity in (7.1) can be dropped. This corresponds to relaxing the range conditions in (v) to $\mathcal{R}(A^*AP) \supseteq \mathcal{R}(Q^*)$ and $\mathcal{R}(CC^*P^*) \subseteq \mathcal{R}(Q)$. Additionally, we could observe that the cofactor representation

of f contains no polynomial associated to any of the four defining equations of $B^\dagger$, showing that $B^\dagger$ can, in fact, be replaced by an arbitrary operator $\tilde{B}$ that does not have to be related to B in any way. These observations led to some of the generalisations in Theorem 7.2.2.

It is also possible to prove the implication $(i) \Rightarrow (v)$ using our framework, in particular Theorem 5.1.7, and software. To this end, first explicit expressions for U_1, U_2, V_1, V_2 in terms of the other basic operators have to be found. Using the heuristics implemented in `operator_gb`, we can find the following elements:

$$\begin{aligned} U_1 &= BCC^*B^*A^*(A^\dagger)^*, & U_2 &= (B^\dagger)^*(C^\dagger)^*C^\dagger B^\dagger A^\dagger A, \\ V_1 &= B^*A^*ABCC^\dagger, & V_2 &= B^\dagger A^\dagger (A^\dagger)^*(B^\dagger)^*(C^\dagger)^*C^*. \end{aligned} \tag{7.2}$$

Then, using the defining identities of $A^\dagger, B^\dagger, C^\dagger, (ABC)^\dagger$, the identity $(ABC)^\dagger = C^\dagger B^\dagger A^\dagger$, and the corresponding adjoint statements as assumptions, the software finds cofactor representations of the polynomial corresponding to $PQ = (PQ)^2$ as well as of the polynomials associated to the four identities in (7.1), where U_1, U_2, V_1, V_2 have been replaced by the expressions in (7.2). We note that these cofactor representations only contain polynomials with integer coefficients. Hence, based on Theorem 5.1.7, this proves the implication $(i) \Rightarrow (v)$.

Similarly to the equivalence discussed above, the software can also be used to prove all other parts of Theorem 7.2.1 as well as of the generalised Theorem 7.2.2. In the following, we explain how this can be done using the equivalence $(i) \Leftrightarrow (v)$ in Theorem 7.2.2.

For the implication $(v) \Rightarrow (i)$, we translate the assumptions $pq = (pq)^2$, $a^*apR \supseteq q^*R$, $cc^*p^*R \subseteq qR$ and their adjoint statements into polynomials. Note that, in order to translate the set inclusions, we can use factorisations analogous to (7.1). In contrast to the original statement of Hartwig, where the Moore-Penrose invertibility of ABC is already given, we now have to prove that $m = abc$ is Moore-Penrose invertible and that $m^\dagger = c^\dagger \tilde{b} a^\dagger$. Hence, the claim is that $\tilde{m} = c^\dagger \tilde{b} a^\dagger$ satisfies the four defining identities of $m^\dagger$. However, trying to show the ideal membership of the corresponding polynomials in the ideal generated by the polynomials representing the assumptions fails. This is because these polynomials do not contain any information about the right $*$-cancellability of m. To use this property, we have to find a polynomial in the ideal generated by the polynomials associated to our assumptions that corresponds to an identity to which this property is applicable. Using the heuristics of the package, we can find a polynomial corresponding to the identity $(1 - m\tilde{m})mm^* = 0$

in the ideal generated by the polynomials representing the assumptions. We can apply the right $*$-cancellability of m to $(1-m\tilde{m})mm^* = 0$ to obtain $(1-m\tilde{m})m = 0$. After including the polynomial associated to this new identity in the set of translated assumptions, the software manages to verify the ideal membership of all polynomials corresponding to the claimed identities fully automatically, and thereby proves the claimed statement.

The proof of $(i) \Rightarrow (v)$ of Theorem 7.2.2 using the software essentially proceeds along the same lines as the proof discussed above concerning the same implication in Hartwig's original theorem. The only difference is that now also the right $*$-cancellability of m has to be shown. To this end, we include the identity $zmm^* = 0$ in the assumptions and prove $zm = 0$ with an arbitrary ring element z. When translating these identities into polynomials, z has to be replaced by a new indeterminate that does not satisfy any additional relations. The software then proves the ideal membership of the polynomial associated to the claimed identity in the ideal generated by the polynomials representing the assumptions fully automatically.

Similarly to the implications discussed above, also all other implications of Theorem 7.2.1, Theorem 7.2.2, and all other results presented in [Cve+21] can be proven automatically using the framework and software. The relevant algebraic computations are available as part of the notebook `https://cocalc.com/georeg/Moore-Penrose-case-study/notebook`.

7.3 Diagram chases

Recall from Section 2.1.1 that abelian categories form a special class of preadditive categories where kernels and cokernels of morphisms exist and have certain nice properties. They are very important structures providing a natural setting for homological algebra and have many applications in pure category theory and algebraic geometry, see, for example, [Bor94, Ch. 1] or [Mac13, Ch. VIII] for textbook expositions and further details. In particular, many statements in homological algebra, for example, can be phrased as *diagram lemmas* in abelian categories. Commonly, these diagram lemmas are proven using a technique called *diagram chasing* and different approaches have been developed to automate such computations [Him20; Pos22].

In this section, we discuss how the framework introduced in this thesis allows to prove diagram lemmas by computations with noncommutative polynomials. We note that this

work is, thus far, still very much in its beginnings. For illustrative purposes, we consider the following classical example of a diagram lemma, which is known as *Four lemma* and part of the more general *Five lemma* [Mac13, Lem. VIII.4.4].

Lemma 7.3.1 (The Four Lemma)**.** *Let*

$$\begin{array}{ccccccc} A & \xrightarrow{f} & B & \xrightarrow{g} & C & \xrightarrow{h} & D \\ \downarrow{\scriptstyle \alpha} & & \downarrow{\scriptstyle \beta} & & \downarrow{\scriptstyle \gamma} & & \downarrow{\scriptstyle \delta} \\ A' & \xrightarrow[f']{} & B' & \xrightarrow[g']{} & C' & \xrightarrow[h']{} & D' \end{array}$$

be a commutative diagram with exact rows. If β, δ are monic and α is epi, then γ is also monic.

In order to express the assumptions and the claim of this statement in terms of the language of our framework, we use the following characterisations. We note that these characterisations are inspired by the "elementary rules for chasing diagrams" in terms of subobjects, see, for example, [Mac13, Thm. VIII.4.3].

Proposition 7.3.2. *For morphisms in an abelian category the following hold:*

1. *$f\colon U \to V$ is monic if and only if, for all $g\colon T \to U$, $f \circ g = 0$ implies $g = 0$;*
2. *$f\colon U \to V$ is epi if and only if, for all $g\colon V \to W$, $g \circ f = 0$ implies $g = 0$;*
3. *$f\colon U \to V$ is epi if and only if, for all $x\colon T \to V$, there exist $y\colon S \to U$ and $e\colon S \to T$ with e epi such that $f \circ y = x \circ e$;*
4. *a sequence $U \xrightarrow{f} V \xrightarrow{g} W$ is exact if and only if $g \circ f = 0$ and, for all $x\colon T \to V$ with $g \circ x = 0$, there exist $y\colon S \to U$ and $e\colon S \to T$ with e epi such that $f \circ y = x \circ e$;*

Proof. The characterisations 1 and 2 follow immediately from the definitions of monic and epi.

For 3, first assume that f is epi and let $x\colon T \to V$. Form the pullback of f and x, yielding the following diagram.

$$\begin{array}{ccc} S & \xrightarrow{f'} & T \\ {\scriptstyle x'}\downarrow & & \downarrow{\scriptstyle x} \\ U & \xrightarrow{f} & V \end{array}$$

Since f is epi, also f' is epi (see, for example, [Mac13, Prop. VIII.4.2]), which yields $f \circ y = x \circ e$ with $e = f'$ and $y = x'$ as required. Conversely, assume that, for every $x\colon T \to V$, there exist $y\colon S \to U$ and $e\colon S \to T$ with e epi such that $f \circ y = x \circ e$. Then, for the particular choice $x = 1_V$, we have $f \circ y = 1_V \circ e = e$, showing that $f \circ y$ is epi. But then also f itself is epi because, for arbitrary $g, h\colon V \to W$,

$$g \circ f = h \circ f \implies g \circ f \circ y = g \circ f \circ y \implies g = h.$$

Finally, for characterisation 4, let $f = \tilde{m} \circ \tilde{e}$ be the natural factorisation of f. Note that $\tilde{m} = \mathrm{im}(f)$ and that $\tilde{e}$ is epi. First, we observe that

$$g \circ f = g \circ \tilde{m} \circ \tilde{e} = 0 \iff g \circ \tilde{m} = 0 \iff \tilde{m} = \ker(g) \circ u \iff \mathrm{im}(f) \leqq \ker(g),$$

for some morphism u. This shows the first part of the equivalence.

For the second part, we first assume that $\ker(g) \leqq \mathrm{im}(f)$, which means that there is u such that $\ker(g) = \mathrm{im}(f) \circ u$. If we let x be such that $g \circ x = 0$, then x factors through $\ker(g) = \mathrm{im}(f) \circ u = \tilde{m} \circ u$, that is, $x = \tilde{m} \circ u \circ v$ for some v. With $\tilde{y} = u \circ v$, form the pullback on the left part of the diagram.

$$\begin{array}{ccccc} S & \xrightarrow{e} & T & = & T \\ {\scriptstyle y}\downarrow & & \downarrow{\scriptstyle \tilde{y}} & & \downarrow{\scriptstyle x} \\ U & \xrightarrow{\tilde{e}} & \bullet & \xrightarrow{\tilde{m}} & V \end{array}$$

Since $\tilde{e}$ is epi, also e is epi, which gives $f \circ y = \tilde{m} \circ \tilde{e} \circ y = x \circ e$ as required.

To finish the proof, assume that, for all x with $g \circ x = 0$, there exist y and e with e epi such that $f \circ y = x \circ e$. Applying this assumption to $x = \ker(g)$ yields $f \circ y = \ker(g) \circ e$. Furthermore, we have $\operatorname{coker}(f) \circ f = 0$, and hence, also

$$0 = \operatorname{coker}(f) \circ f \circ y = \operatorname{coker}(f) \circ \ker(g) \circ e,$$

which implies that $\operatorname{coker}(f) \circ \ker(g) = 0$ since e is epi. But this means that $\ker(g)$ factors through $\ker(\operatorname{coker}(f)) = \operatorname{im}(f)$, that is, $\ker(g) = \operatorname{im}(f) \circ u$ for some morphism u, showing that $\ker(g) \leqq \operatorname{im}(f)$. □

Using these characterisations, we can translate Lemma 7.3.1 into a formula in the language of our framework. For example, the exactness of $A \xrightarrow{f} B \xrightarrow{g} C$ can be translated into the operator statement

$$gf \approx 0 \wedge \forall x \exists y, e \forall h : gx \approx 0 \to (fy \approx xe \wedge (he \approx 0 \to h \approx 0)), \tag{7.3}$$

where we have omitted the sorts of the variables for better readability. In general, translating Lemma 7.3.1 yields an operator statement consisting of assumptions that are all either basic identities of the form $a_i \approx b_i$, like $gf \approx 0$, or slight generalisations of existential quasi-identities $Q_1 x_1 \dots Q_n x_n : \bigwedge_i \left(\bigwedge_j s_{i,j} \approx t_{i,j} \to p_i \approx q_i \right)$, as in

$$\forall x \exists y, e \forall h : (gx \approx 0 \to fy \approx xe) \wedge ((gx \approx 0 \wedge he \approx 0) \to h \approx 0),$$

which is equivalent to the second part of (7.3). The claimed property translates into the formula $\forall z : \gamma z \approx 0 \to z \approx 0$.

To prove Lemma 7.3.1, we can then proceed as follows. First, we form the polynomial ideal I generated by the polynomials $a_i - b_i$ corresponding to all assumptions that are plain identities $a_i \approx b_i$ as well as by the polynomial γz encoding the identity $\gamma z \approx 0$. Then we alternate between the following steps:

1. Search for instantiations of $s_{i,j} - t_{i,j}$ in I. If $s_{i,j} - t_{i,j} \in I$ for some i and all j, extend I by adding the corresponding instantiation of $p_i - q_i$ to it.

2. Try to verify that $z \in I$.

Once we can verify that indeed $z \in I$, this shows that $z = 0$, proving, by Proposition 7.3.2, that γ is indeed monic. In practice, this looks as follows.

Example 7.3.3. *We start by forming the ideal I generated by the following polynomials*

$$\begin{array}{llll} \beta f - f'\alpha, & \gamma g - g'\beta, & \delta h - h'\gamma, & \\ gf, & hg, & g'f', & h'g', \\ \gamma z & & & \end{array}$$

encoding the commutativity of the diagram, part of the exactness assumptions, as well as the assumption $\gamma z = 0$.

Then, using the techniques discussed in Section 5.3 for finding polynomials of certain form in an ideal, we search for elements in I to which we can apply the second part of the exactness assumptions or the properties of the monomorphisms β, δ and the epimorphism α. Doing this, we can find, for example, the element $\delta hz \in I$, which corresponds to the identity $\delta \circ h \circ z = 0$. Applying the fact that δ is monic, we can derive the new identity $h \circ z = 0$ and add the polynomial hz to I, enlarging the ideal in this way.

Now, again using the methods from Section 5.3, we search for elements in the enlarged ideal to which we can apply our assumptions. An easy find is now the new element hz to which we can apply the exactness of $B \xrightarrow{g} C \xrightarrow{h} D$, yielding the new identity $g \circ y = z \circ e$ with new morphism y and epimorphism e. On the polynomial level, this means we can add the element $gy - ze$ to the ideal I, and we obtain a new deduction rule $\forall h : he \in I \implies h \in I$, encoding the property $\forall h : h \circ e = 0 \implies h = 0$ of the epimorphism e.

Repeating the process, we can then find the polynomial $g'\beta y$, encoding $g' \circ \beta \circ y = 0$, to which we can apply the exactness of $A' \xrightarrow{f'} B' \xrightarrow{g'} C'$. This yields the new polynomial $f'y' - \beta y e'$ and a new deduction rule $\forall h : he' \in I \implies h \in I$.

The morphism corresponding to y' must have the same codomain as α because it appears to the right of f'. Therefore, we can apply the fact that α is epi and deduce the new identity $\alpha \circ y'' = y' \circ e''$ with new morphism y'' and epimorphism e''. Thus, we can add the corresponding the polynomial $\alpha y'' - y'e''$ to our ideal and we obtain a new deduction rule $\forall h : he'' \in I \implies h \in I$.

Then we can find the polynomial $\beta fy'' - \beta ye'e'' \in I$, to which we can apply the fact that β is monic and add the new element $fy'' - ye'e''$. With this new element, we can now find the polynomial $zee'e''$ in the ideal, and finally, after applying the deduction rules for e, e', e'', we arrive at the polynomial z, which finishes the proof.

The process outlined above is implemented in our MATHEMATICA package `OperatorGB` in form of a method called `DiagramChase`, see also Section 6.2.1 for additional information. Using this method, proving Lemma 7.3.1 reduces to the following commands.

First, we set up all basic assumptions.

```
In[11]:= commut = {β ** f - f' ** α, γ ** g -  g' ** β, δ ** h - h' ** γ};
         exact = {g ** f, h ** g, g' ** f', h' ** g'};
         monic = {γ ** z};
         assumptions = Join[commut, exact, monic];
         SetUpRing[{z}, {f, g, h, f', g', h'}, {α, β, γ, δ}]
```

Then we can start the diagram chase, giving as additional information which pairs of variables correspond to exact sequences and which variables encode mono- and epimorphisms. Furthermore, we specify that 3 iterations of the steps outlined above should be executed.

```
In[12]:= G = DiagramChase[assumptions, 3,
                ExactAt -> {{f, g}, {g, h}, {f', g'}, {g', h'}},
                Mono -> {β, δ},
                Epi -> {α}];
```

After about 18 seconds, the procedure terminates and returns a set `G` consisting of 30 elements, one of which is

```
Out[12]= z ** e70 ** e83 ** e150
```

where `e70`, `e83`, `e150` are the (random) names for the newly introduced epimorphisms e, e', e''. From this, we can deduce that also `z` itself lies in the ideal once we apply the deduction rules for `e70`, `e83`, `e150`, which finishes the proof. We note that, if we run `DiagramChase` for a few more iterations, then these deduction rules will also be applied automatically, and the set `G` will contain `z` itself.

Similarly, also the Five lemma can be proven fully automatically. Furthermore, by extending the approach to be able to treat kernels and cokernels, also statements like the *Snake lemma* [Mac13, Lem. VIII.4.5] can be treated. We have to note, however, that this does not work fully automatically yet.

Bibliography

[AC75] Alfred V. Aho and Margaret J. Corasick. "Efficient String Matching: An Aid to Bibliographic Search". In: *Communications of the ACM* 18 (1975), pp. 333–340.

[Ack54] Wilhelm Ackermann. *Solvable Cases of the Decision Problem*. North Holland Publishing Co., 1954.

[AG10] M. Laura Arias and M. Celeste Gonzalez. "Positive solutions to operator equations $AXB = C$". In: *Linear Algebra and its Applications* 433 (2010), pp. 1194–1202.

[AL94] William W. Adams and Philippe Loustaunau. *An Introduction to Gröbner Bases*. American Mathematical Society, Providence, 1994.

[AP10] Martin Albrecht and John Perry. *F4/5*. arXiv preprint. 2010. arXiv: `1006.4933`.

[AP11] Alberto Arri and John Perry. "The F5 criterion revised". In: *Journal of Symbolic Computation* 46 (2011), pp. 1017–1029.

[BB98] Miguel A. Borges and Mijail Borges. "Gröbner Bases Property on Elimination Ideal in the Noncommutative Case". In: *Gröbner Bases and Applications*. Cambridge University Press, 1998, pp. 323–337.

[Ber78] George M. Bergman. "The diamond lemma for ring theory". In: *Advances in Mathematics* 29 (1978), pp. 178–218.

[BGV13] José L. Bueso, José Gómez-Torrecillas, and Alain Verschoren. *Algorithmic Methods in Non-Commutative Algebra. Applications to Quantum Groups*. Springer Science & Business Media, 2013.

[BHL17] Jason Bell, Albert Heinle, and Viktor Levandovskyy. "On noncommutative finite factorization domains". In: *Transactions of the American Mathematical Society* 369 (2017), pp. 2675–2695.

[BHR23] Klara Bernauer, Clemens Hofstadler, and Georg Regensburger. "How to Automatise Proofs of Operator Statements: Moore-Penrose Inverse; A Case Study". In: *Computer Algebra in Scientific Computing*. 2023, pp. 39–68.

[BN98] Franz Baader and Tobias Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.

[Bok76] Leonid A. Bokut'. "Imbeddings into simple associative algebras". In: *Algebra i Logika* 15 (1976), pp. 117–142.

[Bor94] Francis Borceux. *Handbook of Categorical Algebra: Volume 2, Categories and Structures*. Cambridge University Press, 1994.

[Buc65] Bruno Buchberger. "Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal". PhD thesis. University of Innsbruck, Austria, 1965.

[Bus94] Samuel R. Buss. "On Herbrand's Theorem". In: *International Workshop on Logic and Computational Complexity*. 1994, pp. 195–209.

[BW93] Thomas Becker and Volker Weispfenning. *Gröbner Bases. A Computational Approach to Commutative Algebra*. Springer-Verlag, 1993.

[CDS01] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. "Atomic Decomposition by Basis Pursuit". In: *SIAM Review* 43 (2001), pp. 129–159.

[Che+20] Cyrille Chenavier, Clemens Hofstadler, Clemens G. Raab, and Georg Regensburger. "Compatible rewriting of noncommutative polynomials for proving operator identities". In: *Proceedings of ISSAC 2020*. 2020, pp. 83–90.

[CLO15] David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms. An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 2015.

[Clu+18] Thomas Cluzeau, Jamal Hossein Poor, Alban Quadrat, Clemens G. Raab, and Georg Regensburger. "Symbolic computation for integro-differential-time-delay operators with matrix coefficients". In: *IFAC-PapersOnLine* 51 (2018), pp. 153–158.

[CLV21] Cyrille Chenavier, Arthur Léonard, and Tristan Vaccon. *On the difficulty of computing non-commutative signature Gröbner bases*. Private communication. 2021.

[Coh03] Paul M. Cohn. *Basic Algebra: Groups, Rings and Fields*. Springer, 2003.

[Coh85] Paul M. Cohn. *Free Rings and Their Relations*. Academic Press, 1985.

[Con71] John H. Conway. *Regular algebra and finite machines*. Chapman and Hall, 1971.

[CT05] Emmanuel J. Candes and Terence Tao. "Decoding by Linear Programming". In: *IEEE Transactions on Information Theory* 51 (2005), pp. 4203–4215.

[Cve+21] Dragana S. Cvetković-Ilić, Clemens Hofstadler, Jamal Hossein Poor, Jovana Milošević, Clemens G. Raab, and Georg Regensburger. "Algebraic proof methods for identities of matrices and operators: improvements of Hartwig's triple reverse order law". In: *Applied Mathematics and Computation* 409, 126357 (2021).

[CW17] Dragana S. Cvetković-Ilić and Yimin Wei. *Algebraic Properties of Generalized Inverses*. Springer, 2017.

[CW92] Ronald R. Coifman and M. Victor Wickerhauser. "Entropy-Based Algorithms for Best Basis Selection". In: *IEEE Transactions on Information Theory* 38 (1992), pp. 713–718.

[Dan51] George B. Dantzig. "Maximization of a linear function of variables subject to linear inequalities". In: *Activity Analysis of Production and Allocation* 13 (1951), pp. 339–347.

[Dav58] Martin Davis. *Computability and Unsolvability*. McGraw-Hill Book Co., Inc., 1958.

[DD10] Dragan S. Djordjević and Nebojša Č. Dinčić. "Reverse order law for the Moore-Penrose inverse". In: *Journal of Mathematical Analysis and Applications* 361 (2010), pp. 252–261.

[DD14] Nebojša Č. Dinčić and Dragan S. Djordjević. "Hartwig's triple reverse order law revisited". In: *Linear and Multilinear Algebra* 62 (2014), pp. 918–924.

[Den11] Chun Yuan Deng. "A generalization of the Sherman-Morrison-Woodbury formula". In: *Applied Mathematics Letters* 24 (2011), pp. 1561–1564.

[Don06] David L. Donoho. "Compressed Sensing". In: *IEEE Transactions on Information Theory* 52 (2006), pp. 1289–1306.

[Dou66] Ronald G. Douglas. "On majorization, factorization, and range inclusion of operators on Hilbert space". In: *Proceedings of the American Mathematical Society* 17 (1966), pp. 413–415.

[DV01] Anatoli Degtyarev and Andrei Voronkov. "Equality Reasoning in Sequent-Based Calculi". In: *Handbook of Automated Reasoning*. Elsevier, 2001, pp. 611–706.

[Ede+23] Christian Eder, Pierre Lairez, Rafael Mohr, and Mohab Safey El Din. "A signature-based algorithm for computing the nondegenerate locus of a polynomial system". In: *Journal of Symbolic Computation* 119 (2023), pp. 1–21.

[Ede13] Christian Eder. "An analysis of inhomogeneous signature-based Gröbner basis computations". In: *Journal of Symbolic Computation* 59 (2013), pp. 21–35.

[EF17] Christian Eder and Jean-Charles Faugère. "A survey on signature-based algorithms for computing Gröbner bases". In: *Journal of Symbolic Computation* 80 (2017), pp. 719–784.

[EFT21] Heinz-Dieter Ebbinghaus, Jörg Flum, and Wolfgang Thomas. *Mathematical Logic*. Springer, 2021.

[EP11] Christian Eder and John Perry. "Signature-based algorithms to compute Gröbner bases". In: *Proceedings of ISSAC 2011*. 2011, pp. 99–106.

[EPP17] Christian Eder, Gerhard Pfister, and Adrian Popescu. "On signature-based Gröbner bases over Euclidean rings". In: *Proceedings of ISSAC 2017*. 2017, pp. 141–148.

[Fau02] Jean-Charles Faugère. "A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5)". In: *Proceedings of ISSAC 2002*. 2002, pp. 75–83.

[Fau99] Jean-Charles Faugère. "A new efficient algorithm for computing Gröbner bases (F_4)". In: *Journal of Pure and Applied Algebra* 139 (1999), pp. 61–88.

[FL10] Jean-Charles Faugère and Sylvain Lachartre. "Parallel Gaussian Elimination for Gröbner bases computations in finite fields". In: *Proceedings of the 4th International Workshop on Parallel and Symbolic Computation*. 2010, pp. 89–97.

[FV21] Maria Francis and Thibaut Verron. "On two signature variants of Buchberger's algorithm over principal ideal domains". In: *Proceedings of ISSAC 2021*. 2021, pp. 139–146.

[Gar05] W. Dale Garraway. "Sheaves for an involutive quantaloid". In: *Cahiers de Topologie et Géométrie Différentielle Catégoriques* 46 (2005), pp. 243–274.

[Gen35] Gerhard Gentzen. "Untersuchungen über das logische Schließen. I." In: *Mathematische Zeitschrift* 35 (1935), pp. 176–210.

[GGV10] Shuhong Gao, Yinhua Guan, and Frank Volny IV. "A new incremental algorithm for computing Gröbner bases". In: *Proceedings of ISSAC 2010*. 2010, pp. 13–19.

[GHM19] Yves Guiraud, Eric Hoffbeck, and Philippe Malbos. "Convergent presentations and polygraphic resolutions of associative algebras". In: *Mathematische Zeitschrift* 293 (2019), pp. 113–179.

[Gil60] Paul C. Gilmore. "A Proof Method for Quantification Theory: Its Justification and Realization". In: *IBM Journal of Research and Development* 4 (1960), pp. 28–35.

[Gio+91] Alessandro Giovini, Teo Mora, Gianfranco Niesi, Lorenzo Robbiano, and Carlo Traverso. "'One sugar cube, please' or selection strategies in the Buchberger algorithm". In: *Proceedings of ISSAC 1991*. 1991, pp. 49–54.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

[GM86] Rüdiger Gebauer and H. Michael Möller. "Buchberger's Algorithm and Staggered Linear Bases". In: *Proceedings of SYMSAC 1986*. 1986, pp. 218–221.

[Göd30] Kurt Gödel. "Die Vollständigkeit der Axiome des logischen Funktionenkalküls". In: *Monatshefte für Mathematik und Physik* 37 (1930), pp. 349–360.

[Gre00] Edward L. Green. "Multiplicative bases, Gröbner bases, and right Gröbner bases". In: *Journal of Symbolic Computation* 29 (2000), pp. 601–623.

[Gre66] Thomas N. E. Greville. "Note on the generalized inverse of a matrix product". In: *SIAM Review* 8 (1966), pp. 518–521.

[GVW16] Shuhong Gao, Frank Volny IV, and Mingsheng Wang. "A new framework for computing Gröbner bases". In: *Mathematics of Computation* 85 (2016), pp. 449–465.

[Har86] Robert E. Hartwig. “The reverse order law revisited”. In: *Linear Algebra and its Applications* 76 (1986), pp. 241–246.

[Her30] Jacques Herbrand. “Recherches sur la théorie de la démonstration”. PhD thesis. University of Paris, France, 1930.

[Hey01] Anne Heyworth. “One-Sided Noncommutative Gröbner Bases with Applications to Computing Green’s Relations”. In: *Journal of Algebra* 242 (2001), pp. 401–416.

[Hig52] Graham Higman. “Ordering by divisibility in abstract algebras”. In: *Proceedings of the London Mathematical Society* 3 (1952), pp. 326–336.

[Him20] Markus Himmel. “Diagram Chasing in Interactive Theorem Proving”. Bachelor’s thesis. Karlsruher Institut für Technologie (KIT), Germany, 2020.

[HJ20] Amir Hashemi and Masoumeh Javanbakht. “On the construction of staggered linear bases”. In: *Journal of Algebra and its Applications* 20, 2150132 (2020).

[HM23] Amir Hashemi and H. Michael Möller. “A new algorithm for computing staggered linear bases”. In: *Journal of Symbolic Computation* 117 (2023), pp. 1–14.

[Hof20] Clemens Hofstadler. “Certifying operator identities and ideal membership of noncommutative polynomials”. Master’s thesis. Johannes Kepler University Linz, Austria, 2020.

[Hog13] Leslie Hogben. *Handbook of Linear Algebra*. CRC press, 2013.

[HRR22a] Clemens Hofstadler, Clemens G. Raab, and Georg Regensburger. “Computing Elements of Certain Form in Ideals to Prove Properties of Operators”. In: *Mathematics in Computer Science* 16, 17 (2022).

[HRR22b] Clemens Hofstadler, Clemens G. Raab, and Georg Regensburger. *Universal truth of operator statements via ideal membership*. arXiv preprint. 2022. arXiv: `2212.11662`.

[HS99] J. William Helton and Mark Stankus. “Computer Assistance for ‘Discovering’ Formulas in System Engineering and Operator Theory”. In: *Journal of Functional Analysis* 161 (1999), pp. 289–363.

[HSW98] J. William Helton, Mark Stankus, and John J. Wavrik. “Computer simplification of formulas in linear systems theory”. In: *IEEE Transactions on Automatic Control* 43 (1998), pp. 302–314.

[Hum90] James E. Humphreys. *Reflection groups and Coxeter groups*. Cambridge University Press, 1990.

[HV22] Clemens Hofstadler and Thibaut Verron. "Signature Gröbner bases, bases of syzygies and cofactor reconstruction in the free algebra". In: *Journal of Symbolic Computation* 113 (2022), pp. 211–241.

[HV23a] Clemens Hofstadler and Thibaut Verron. *Short proofs of ideal membership*. arXiv preprint. 2023. arXiv: 2302.02832.

[HV23b] Clemens Hofstadler and Thibaut Verron. "Signature Gröbner Bases in Free Algebras over Rings". In: *Proceedings of ISSAC 2023*. 2023, pp. 298–306.

[HW94] J. William Helton and John J. Wavrik. "Rules for computer simplification of the formulas in operator model theory and linear systems". In: *Nonselfadjoint Operators and Related Topics*. 1994, pp. 325–354.

[IBM23] IBM. *ILOG CPLEX Optimization Studio*. Version 22.1. 2023. URL: https://www.ibm.com/products/ilog-cplex-optimization-studio/cplex-optimizer.

[KDC07] Jerry J. Koliha, Dragan Djordjević, and Dragana Cvetković. "Moore-Penrose inverse in rings with involution". In: *Linear Algebra and its Applications* 426 (2007), pp. 371–381.

[Kin14] Simon A. King. "A non-commutative F5 algorithm with an application to the computation of Loewy layers". In: *Journal of Symbolic Computation* 65 (2014), pp. 111–129.

[KN85] Deepak Kapur and Paliath Narendran. "A finite Thue system with decidable word problem and without equivalent finite canonical system". In: *Theoretical Computer Science* 35 (1985), pp. 337–344.

[KR05] Martin Kreuzer and Lorenzo Robbiano. *Computational Commutative Algebra 2*. Springer, 2005.

[KS16] Daniel Kroening and Ofer Strichman. *Decision Procedures. An Algorithmic Point of View*. Springer, 2016.

[Li+19] Ting Li, Yao Sun, Zhenyu Huang, Dingkang Wang, and Dongdai Lin. "Speeding Up the GVW Algorithm via a Substituting Method". In: *Journal of Systems Science and Complexity* 32 (2019), pp. 205–233.

[Li08] Yuan Li. "The Moore-Penrose inverses of products and differences of projections in a C^*-algebra". In: *Linear Algebra and its Applications* 428 (2008), pp. 1169–1177.

[LL09] Roberto La Scala and Viktor Levandovskyy. "Letterplace ideals and non-commutative Gröbner bases". In: *Journal of Symbolic Computation* 44 (2009), pp. 1374–1393.

[LMA23] Viktor Levandovskyy, Tobias Metzlaff, and Karim Abou Zeid. "Computing free non-commutative Gröbner bases over $\mathbb{Z}$ with Singular:Letterplace". In: *Journal of Symbolic Computation* 115 (2023), pp. 201–222.

[LS15] Douglas Lind and Klaus Schmidt. "A survey of algebraic actions of the discrete Heisenberg group". In: *Russian Mathematical Surveys* 70 (2015), pp. 657–714.

[LSA20] Viktor Levandovskyy, Hans Schönemann, and Karim Abou Zeid. "Letterplace – a Subsystem of Singular for Computations with Free Algebras via Letterplace Embedding". In: *Proceedings of ISSAC 2020*. 2020, pp. 305–311.

[Mac13] Saunders Mac Lane. *Categories for the Working Mathematician*. Springer Science & Business Media, 2013.

[Man93] María Manzano. "Introduction to Many-sorted Logic". In: *Many-sorted Logic and its Applications*. John Wiley & Sons, Inc., 1993, pp. 3–86.

[Mil16] Ezra Miller. *Finding all monomials in a polynomial ideal*. arXiv preprint. 2016. arXiv: `1605.08791`.

[Mil18] Jovana Milošević. "Hartwig's Triple Reverse Order Law in C*-algebras". In: *Filomat* 32 (2018), pp. 4229–4232.

[MMT92] H. Michael Möller, Teo Mora, and Carlo Traverso. "Gröbner Bases Computation Using Syzygies". In: *Proceedings of ISSAC 1992*. 1992, pp. 320–328.

[Moo20] Eliakim H. Moore. "On the reciprocal of the general algebraic matrix". In: *Bulletin of the American Mathematical Society* 26 (1920), pp. 294–295.

[Mor15] Ferdinando Mora. "De Nugis Groebnerialium 4: Zacharias, Spears, Möller". In: *Proceedings of ISSAC 2015*. 2015, pp. 283–290.

[Mor16] Teo Mora. *Solving Polynomial Equation Systems IV: Volume 4, Buchberger Theory and Beyond*. Cambridge University Press, 2016.

[Mor85] Ferdinando Mora. "Gröbner bases for non-commutative polynomial rings". In: *International Conference on Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes*. 1985, pp. 353–362.

[Mor87] Teo Mora. "Gröbner bases and the word problem". Preprint, University of Genova. 1987.

[Mor94] Teo Mora. "An introduction to commutative and noncommutative Gröbner bases". In: *Theoretical Computer Science* 134 (1994), pp. 131–173.

[Muł22] Wojciech Muła. *pyahocorasick*. Version 2.0.0. 2022. URL: `https://github.com/WojciechMula/pyahocorasick/`.

[MZ93] Stéphane G. Mallat and Zhifeng Zhang. "Matching Pursuits With Time-Frequency Dictionaries". In: *IEEE Transactions on Signal Processing* 41 (1993), pp. 3397–3415.

[MZ98] Alexander A. Mikhalev and Andrej A. Zolotykh. "Standard Gröbner-Shirshov bases of free algebras over rings. I. Free associative algebras". In: *International Journal of Algebra and Computation* 8 (1998), pp. 689–726.

[Nas63] Crispin St. J. A. Nash-Williams. "On well-quasi-ordering finite trees". In: *Mathematical Proceedings of the Cambridge Philosophical Society*. 1963, pp. 833–835.

[Nel+10] Timothy Nelson, Daniel J. Dougherty, Kathi Fisler, and Shriram Krishnamurthi. *On the finite model property in order-sorted logic*. Technical report. Worcester Polytechnic Institute. 2010.

[Nor01] Patrik Nordbeck. "Canonical bases for algebraic computations". PhD thesis. Lund University, Sweden, 2001.

[Nor98] Patrik Nordbeck. "On some Basic Applications of Gröbner bases in Non-commutative Polynomial Rings". In: *Gröbner Bases and Applications*. Cambridge University Press, 1998, pp. 463–472.

[Pen55] Roger Penrose. "A generalized inverse for matrices". In: *Mathematical Proceedings of the Cambridge Philosophical Society* 51 (1955), pp. 406–413.

[Pos22] Sebastian Posur. "On free abelian categories for theorem proving". In: *Journal of Pure and Applied Algebra* 226, 106994 (2022).

[PR81] Roland Puystjens and Donald W. Robinson. "The Moore-Penrose inverse of a morphism with factorization". In: *Linear Algebra and its Applications* 40 (1981), pp. 129–141.

[Pri96] F. Leon Pritchard. "The Ideal Membership Problem in Non-Commutative Polynomial Rings". In: *Journal of Symbolic Computation* 22 (1996), pp. 27–48.

[PW00] Florian A. Potra and Stephen J. Wright. "Interior-point methods". In: *Journal of Computational and Applied Mathematics* 124 (2000), pp. 281–302.

[RN10] Stuart J. Russell and Peter Norvig. *Artificial Intelligence. A Modern Approach*. Pearson Education, Inc., 2010.

[Row06] Louis H. Rowen. *Graduate Algebra: Commutative View*. American Mathematical Society, 2006.

[Row91] Louis H. Rowen. *Ring theory*. Student Edition. Academic Press, Inc., 1991.

[RP87] Donald W. Robinson and Roland Puystjens. "Generalized inverses of morphisms with kernels". In: *Linear Algebra and its Applications* 96 (1987), pp. 65–86.

[RRH21] Clemens G. Raab, Georg Regensburger, and Jamal Hossein Poor. "Formal proofs of operator identities by a single formal computation". In: *Journal of Pure and Applied Algebra* 225, 106564 (2021).

[RS02] Gerhard Rosenberger and Martin Scheer. "Classification of the finite generalized tetrahedron groups". In: *Contemporary Mathematics* 296 (2002), pp. 207–230.

[RS12] Bjarke Hammersholt Roune and Michael Stillman. "Practical Gröbner basis computation". In: *Proceedings of ISSAC 2012*. 2012, pp. 203–210. Full version including the appendix available at arXiv: 1206.6940.

[Sag20] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.1)*. 2020. URL: `https://www.sagemath.org`.

[Sch21] Leonard Schmitz. "Varieties over Module Homomorphisms and their Correspondence to free Algebras". Master's thesis. RWTH Aachen, Germany, 2021.

[Sch98] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, 1998.

[SL20] Leonard Schmitz and Viktor Levandovskyy. "Formally Verifying Proofs for Algebraic Identities of Matrices". In: *Intelligent Computer Mathematics*. 2020, pp. 222–236.

[SST13] Mutsumi Saito, Bernd Sturmfels, and Nobuki Takayama. *Gröbner Deformations of Hypergeometric Differential Equations*. Springer Science & Business Media, 2013.

[Sun+12] Yao Sun, Dingkang Wang, Xiaodong Ma, and Yang Zhang. "A Signature-Based Algorithm for Computing Gröbner Bases in Solvable Polynomial Algebras". In: *Proceedings of ISSAC 2012*. 2012, pp. 351–358.

[Til87] Bret Tilson. "Categories as algebra: An essential ingredient in the theory of monoids". In: *Journal of Pure and Applied Algebra* 48 (1987), pp. 83–198.

[Wal87] Christoph Walther. *A Many-sorted Calculus Based on Resolution and Paramodulation*. Morgan Kaufmann, 1987.

[Win96] Franz Winkler. *Polynomial Algorithms in Computer Algebra*. Springer Science & Business Media, 1996.

[Xiu12] Xingqiang Xiu. "Non-commutative Gröbner bases and applications". PhD thesis. University of Passau, Germany, 2012.