Gröbner Bases and Macaulay Matrices in Isabelle/HOL

Alexander Maletzky*

RISC, Johannes Kepler University Linz, Austria,
alexander.maletzky@risc.jku.at

Abstract

We present the formalization of computing Gröbner bases by row reducing Macaulay matrices, in the Isabelle/HOL proof assistant. More precisely, we formalized that after row reducing a sufficiently large matrix constructed from an initial set of polynomials one can read off a Gröbner basis of that set from the resulting reduced row echelon form. In doing so we closely followed the recent thesis by Manuela Wiesinger-Widi. To the best of our knowledge, this formalization is the first formalization of its kind in any proof assistant.

1 Introduction

As is well known, Gröbner bases [2] can be computed by critical-pair/completion algorithms that take an initial set of polynomials as input and repeatedly add new elements to it until the resulting set is a Gröbner basis. Although said algorithms terminate in all instances, it is not known a-priori how many new elements must be added, i. e. how many iterations of the main loop must be carried out. An alternative approach to computing Gröbner bases proceeds by converting the initial set of polynomials into a big matrix, the so-called *Macaulay matrix*, then transforming this matrix into reduced row echelon form by standard techniques known from linear algebra, and finally reading off a Gröbner basis from the resulting row echelon form. Thus, the iterative nature of critical-pair/completion algorithms is replaced by an n-step approach, where n depends on the input but is fixed a-priori, to computing Gröbner bases. Manuela Wiesinger-Widi proves in [13] that said method works indeed correctly, under the provision that the Macaulay matrix constructed at the beginning is sufficiently large, i. e. contains sufficiently many *shifts* of the original polynomials. In [13] she also provides upper bounds on the dimensions of the Macaulay matrices, both in the general case of arbitrary input and in the special case where the input consists of two binomials.

This report presents the formalization of computing Gröbner bases via Macaulay matrices in the open-source proof assistant Isabelle/HOL [10, 12], closely following [13]. More precisely, we formalized Macaulay matrices (i. e. matrices

*The research was funded by the Austrian Science Fund (FWF): P 29498-N31

constructed from sets of multivariate polynomials) and then proved that the reduced row echelon form of such a matrix can be translated back into a set of polynomials which happens to be a Gröbner basis—at least if the Macaulay matrix is large enough, as pointed out earlier. We did not formalize, as of yet, concrete upper bounds on the dimensions of the matrices that are necessary for turning the whole approach into an executable algorithm; this is still ongoing work. However, we discovered, as a side result of our ongoing formalization effort, that the general bounds presented in [13] can be improved by roughly a factor of 2. Details can be found in Section 2.4.

The formalization is freely available on GitHub [8] for the current development version of Isabelle[1] and the Archive of Formal Proofs[2]. It builds upon an existing formalization of Faugére's F_4 algorithm [5, 7], described in [9], which in turn depends on a formalization of matrices, row reduction and reduced row echelon forms [11]; both are contained in the Archive of Formal Proofs. To the best of our knowledge, the computation of Gröbner bases via Macaulay matrices has never been formalized in any proof assistant before.

The report is divided into two main parts: Section 2 reviews the theoretical background of Gröbner bases, Macaulay matrices and reduced row echelon forms; this part is included only for the exposition to be self-contained, but a more detailed presentation of these concepts can also be found in [13]. Section 3, then, presents the actual formalization of the theory in Isabelle/HOL.

2 Theoretical Background

2.1 Preliminaries

Let in the remainder K always be a field and $X = \{x_1, \ldots, x_n\}$ a set of n indeterminates. $[X]$ denotes the commutative monoid of *power-products* in X, i. e. the set of all terms of the form $x_1^{\alpha_1} \cdot \ldots \cdot x_n^{\alpha_n}$ for $\alpha_i \in \mathbb{N}$ $(1 \leq i \leq n)$ endowed with the usual multiplication of such terms, and $K[X]$ denotes the polynomial ring in X over K, i. e. all K-linear combinations of power-products in $[X]$ with the usual addition and multiplication. A polynomial of the form $c \cdot t$, for $c \in K\backslash\{0\}$ and $t \in [X]$, is called a *monomial*.

Furthermore, we fix an *admissible order relation* $\preceq$ on $[X]$, which is a linear ordering such that $1 = x_1^0 \ldots x_n^0$ is the least element wrt. $\preceq$ and such that $s \preceq t$ implies $s \cdot u \preceq t \cdot u$ for all $s, t, u \in [X]$. For all $p \in K[X]$ and $t \in [X]$, $\mathrm{C}(p, t)$ denotes the coefficient of t in p (which may be 0), and $\mathrm{supp}(p)$ denotes the *support* of p, i. e. the finite set $\{t \in [X] \mid \mathrm{C}(p, t) \neq 0\}$. If $p \neq 0$, $\mathrm{lp}(p)$ denotes the *leading power-product* of p, which is the largest (wrt. $\preceq$) power-product in $\mathrm{supp}(p)$. Finally, $\mathrm{lc}(p)$ denotes the *leading coefficient* of p, defined as $\mathrm{lc}(p) := \mathrm{C}(p, \mathrm{lp}(p))$.

2.2 Gröbner Bases and Ideals

We briefly recall the basic properties of Gröbner bases and ideals, to make this report as self-contained as possible. Readers not familiar with the theory are referred to any standard textbook on the subject, as for instance [3].

[1] `http://isabelle.in.tum.de/repos/isabelle`

[2] `http://devel.isa-afp.org/`

Let $F \subseteq K[X]$. Then the ideal generated by F, written $\langle F \rangle$, is the uniquely smallest set satisfying (i) $F \subseteq \langle F \rangle$, (ii) $\langle F \rangle$ is closed under addition, and (iii) $\langle F \rangle$ is closed under multiplication by arbitrary polynomials (i. e. $p \in \langle F \rangle \wedge q \in K[X] \Rightarrow q \cdot p \in \langle F \rangle$).

A set $G \subseteq K[X]$ is a *Gröbner basis* if, and only if, for every $p \in \langle G \rangle \backslash \{0\}$ there is $g \in G \backslash \{0\}$ such that $\mathrm{lp}(g) \mid \mathrm{lp}(p)$. Although several equivalent characterizations of Gröbner bases exist in the literature, this is the one we are going to use throughout the paper.

It is not difficult to prove that every ideal of $K[X]$ admits a finite Gröbner basis. It is much more challenging, though, to decide whether a given set is a Gröbner basis, let alone to explicitly construct one given an arbitrary finite generating set F of the ideal in question. Fortunately, Buchberger in [2] proved an alternative characterization of Gröbner bases that can be effectively decided and which, furthermore, can be transformed into a critical-pair/completion algorithm for computing a Gröbner basis G from F, with the additional property $\langle G \rangle = \langle F \rangle$. The details of this well-known algorithm are not so important for the present report, but interested readers may find them in literally every textbook on Gröbner bases, like [3].

So, the problem solved by Buchberger's algorithm [2] and by Wiesinger-Widi's approach [13] (whose formalization in Isabelle/HOL we present in this report) is as follows:

Problem 1. Let $F \subseteq K[X]$ be finite. Find a finite set $G \subseteq K[X]$ such that G is a Gröbner basis and $\langle G \rangle = \langle F \rangle$.

We omit any explanations why Problem 1 is interesting in the first place, and why Gröbner bases are important; instead, we again refer to any of the many expositions about Gröbner bases in the literature. What we do mention, however, is the import fact that Gröbner bases are in general not unique; indeed, an ideal may even have infinitely many Gröbner bases. Luckily one can impose stronger conditions on generating sets of ideals that lead to uniqueness. For instance, a set $F \subseteq K[X]$ is called *reduced* if (i) $\mathrm{lc}(f) = 1$ for all $f \in F$, and (ii) for all $f, g \in F \backslash \{0\}$ and $t \in \mathrm{supp}(g)$ with $f \neq g$ we have $\neg \mathrm{lp}(f) \mid t$. Reduced Gröbner bases are unique for every ideal, at least up to the implicitly fixed admissible order relation $\preceq$:

Theorem 1. *Let $F \subseteq K[X]$. Then there exists a unique reduced Gröbner basis G with $\langle G \rangle = \langle F \rangle$; moreover, G is finite.*

Proof. See, for instance, Theorem 5 in Chapter 2, § 7, of [3]. □

2.3 Macaulay Matrices and Reduced Row Echelon Forms

Let F be a finite list of polynomials and $T \subset [X]$ finite; let m be the length of F and $\ell = |T|$. The *Macaulay matrix* $\mathrm{Mac}(F, T)$ of F wrt. T is the matrix $A \in K^{m \times \ell}$ such that $A_{i,j} = \mathrm{C}(F_i, \hat{T}_j)$, where $\hat{T}$ is the list of elements of T sorted descending wrt. $\preceq$. In other words, the (i, j)-th entry of $\mathrm{Mac}(F, T)$ is the coefficient of the j-th largest power-product in T in the i-th polynomial in F; of course, such entries could well be 0.

As an abbreviation we also introduce $\mathrm{Mac}(F)$ to denote $\mathrm{Mac}\left(F, \bigcup_{i=1}^{m} \mathrm{supp}(F_i)\right)$, where T is fixed as the set of all power-products appearing in at least one polynomial in F with non-zero coefficient. That means, $\mathrm{Mac}(F)$ does not contain 0-columns, although it may still contain 0-rows.

Let now $A \in K^{m\times\ell}$ be an arbitrary matrix and T again an ℓ-element set of power-products. The right-inverse of function Mac, denoted $\mathrm{Mac}^{-1}(A, T)$, gives the unique list F' of polynomials such that $\mathrm{Mac}(F', T) = A$ and such that $\mathrm{supp}(F'_i) \subseteq T$ for all $i \le m$. Thus, we always have $\mathrm{Mac}(\mathrm{Mac}^{-1}(A, T), T) = A$, but not necessarily $\mathrm{Mac}^{-1}(\mathrm{Mac}(F, T), T) = F$; the latter equality only holds if $\mathrm{supp}(F_i) \subseteq T$ for all $i \le m$.

Example 1. Let $F := [x_2^3 - 5x_1^2x_2 - 2, -4x_2^3 + 2x_2^2 + x_1^2x_2, 2x_2^3 - x_2^2 - x_1 + 4]$ be a 3-element list in $\mathbb{Q}[x_1, x_2]$, let $T := \bigcup_{i=1}^3 \mathrm{supp}(F_i) = \{x_2^3, x_2^2, x_1^2x_2, x_1, 1\}$, and let $\preceq$ be the purely lexicographic order relation with $x_1 \prec x_2$. Then

$$\mathrm{Mac}(F,T) = \begin{array}{c} \\ F_1 \\ F_2 \\ F_3 \end{array} \begin{array}{c} \begin{array}{ccccc} x_2^3 & x_2^2 & x_1^2x_2 & x_1 & 1 \end{array} \\ \begin{pmatrix} 1 & 0 & -5 & 0 & -2 \\ -4 & 2 & 1 & 0 & 0 \\ 2 & -1 & 0 & -1 & 4 \end{pmatrix} \end{array}.$$

As before, let $A \in K^{m\times\ell}$ be some matrix. If $1 \le i \le m$, A_i denotes the i-th row of A. If $A_i \ne 0$, then $\mathrm{pivot}(A, i)$ is defined to be the smallest index $1 \le j \le \ell$ such that $A_{i,j} \ne 0$. We say that A is in *reduced row echelon form* (rref) if, and only if, for all non-zero rows A_i, $A_{i,\mathrm{pivot}(A,i)} = 1$ and $A_{i',\mathrm{pivot}(A,i)} = 0$ for all $1 \le i' \le m$ with $i' \ne i$ (i. e. the pivot element in the i-th row is the only non-zero element in the respective column). Please note that usually one imposes stronger conditions on rrefs, as for instance the rows being sorted wrt. their pivot columns, but we do not need them here.

As is well-known, every matrix can be brought into a reduced row echelon form by means of *elementary row operations*, which are swapping rows, multiplying one row by a non-zero scalar factor and adding one row to another. We omit the (quite obvious) details here, but point out that the rref of a matrix in our setting is unique only up to the order of rows. Finally, we define the function $\mathrm{rref}(A)$ to return *some* rref of the matrix A.

Example 2. Consider $\mathrm{Mac}(F, T)$ from Example 1. A rref of this matrix is

$$\begin{pmatrix} 1 & 0 & 0 & -10 & 38 \\ 0 & 1 & 0 & -19 & 72 \\ 0 & 0 & 1 & -2 & 8 \end{pmatrix}.$$

We conclude this section by an important observation concerning the *row space* $\mathrm{rspace}(A)$ of a matrix $A \in K^{m\times\ell}$, i. e. the vector-subspace of K^ℓ spanned by the rows of A:

Theorem 2. *Let $A \in K^{m\times\ell}$. Then* $\mathrm{rspace}(\mathrm{rref}(A)) = \mathrm{rspace}(A)$.

Proof. Obvious, since no elementary row operation changes the row space. □

2.4 Computing Gröbner Bases by Macaulay Matrices

As one further preliminary we need the notion of a *shift* of a polynomial: a shift of $p \in K[X]$ is simply a multiple of p by a power-product $t \in [X]$, i. e. $t \cdot p$. For $T \subseteq [X]$ and $F \subseteq K[X]$, $T \cdot F$ is defined as $T \cdot F := \{t \cdot f \mid t \in T, f \in F\}$.

The informal algorithm for computing Gröbner bases by Macaulay matrices proceeds as follows:

Algorithm 1 (Computing Gröbner bases by Macaulay matrices)**.**
Input: $F \subseteq K[X]$ finite
Output: Gröbner basis $G \subseteq K[X]$, $\langle G\rangle = \langle F\rangle$

1. Consider some shifts of F, i.e. a finite subset of $[X] \cdot F$, and collect the elements in a list F' (the order of the elements in the list is irrelevant).
2. Set $T := \bigcup_{f\in F'} \operatorname{supp}(f)$.
3. Compute $A := \operatorname{rref}(\operatorname{Mac}(F', T))$.
4. Set $G := \operatorname{Mac}^{-1}(A, T)$. If sufficiently many shifts have been considered in Step 1, then G is a Gröbner basis of F.

The intuition why Algorithm 1 is indeed correct will hopefully become clear in the proof sketch of Theorem 3 below.

But anyway, there is still a problem: G is only a Gröbner basis if sufficiently many shifts are considered in Step 1 of the algorithm—but what exactly does *sufficient* mean? So, the key question that must be answered before being able to effectively compute Gröbner bases by Macaulay matrices is

> Which shifts of the polynomials in F are necessary such that the result obtained from Algorithm 1 is indeed a Gröbner basis?

The answer to this question is given in [13]:

Theorem 3 (Theorem 2.3.3 in [13])**.** *Let H be a Gröbner basis of $F = \{f_1, \ldots, f_m\}$ and let $S \subseteq [X]\cdot F$ finite be such that for all $h \in H$ there exist $q_1, \ldots, q_m \in K[X]$ with $h = \sum_{i=1}^{m} q_i\, f_i$ and $\operatorname{supp}(q_i) \cdot \{f_i\} \subseteq S$. Then S is a suitable set of shifts of F for computing a Gröbner basis of F (not necessarily H) by Algorithm 1.*

Proof sketch. Let F', T, A and G be as in Algorithm 1. As one can easily verify, the choice of S ensures that for all $h \in H$ we have

$$\operatorname{Mac}(\{h\}, T) \in \operatorname{rspace}(\operatorname{Mac}(F', T)) = \operatorname{rspace}(A)$$

where the $(1 \times |T|)$-dimensional matrix $\operatorname{Mac}(\{h\}, T)$ is identified with a row vector in $K^{|T|}$. Hence, every h can be written as a *linear* combination of the elements in G, and since for $g_1, g_2 \in G$ with $g_1 \neq g_2$ we also have $\operatorname{lp}(g_1) \neq \operatorname{lp}(g_2)$ (follows readily from the fact that A is a rref), we can infer that $\operatorname{lp}(H) \subseteq \operatorname{lp}(G)$. Furthermore, $G \subseteq \langle F'\rangle = \langle H\rangle$ by construction, and so G is a Gröbner basis of $\langle F'\rangle$, because H is. □

Theorem 3 is a first step towards the final solution, but at first glance it does not really help, because in order to apply it we must already know a Gröbner basis of F. After a closer look, however, one realizes that not a Gröbner basis G itself must be known, but only the *cofactors* q_i of the elements of G; and not even that, because it clearly suffices if only a finite superset of these cofactors is known, characterized by, say, a certain degree bound. And fortunately, such degree bounds exist; let for this $\operatorname{Dube}(a, b) := 2\left(\frac{a^2}{2} + a\right)^{2^{b-1}}$:

Theorem 4 (Theorem 8.2 in [4])**.** *Let* $F = \{f_1, \ldots, f_m\} \subseteq K[X]$ *be a set of* homogeneous[3] *polynomials and set* $d := \max_{i=1}^m(\deg(f_i))$*. Then, for every admissible ordering* $\preceq$ *and for all* g *in the reduced Gröbner basis of* F *we have* $\deg(g) \leq \mathrm{Dube}(d, n-1)$*, where* n *is the number of indeterminates in* X*.*

From this one can easily infer:

Theorem 5 (Corollary 5.4 in [1])**.** *Let* $F = \{f_1, \ldots, f_m\} \subseteq K[X]$ *be an arbitrary set of polynomials and set* $d := \max_{i=1}^m(\deg(f_i))$*. Then, for every admissible ordering* $\preceq$*, there exists a Gröbner basis* G *of* F *such that for every* $g \in G$ *there exist* $q_1, \ldots, q_m \in K[X]$ *with* $g = \sum_{i=1}^m q_i\, f_i$ *and* $\deg(q_i\, f_i) \leq \mathrm{Dube}(d, n)$ *for all* $1 \leq i \leq m$*.*

Together, Theorems 3 and 5 provide an effective answer to the problem of the shifts:

Theorem 6. *Let* $F = \{f_1, \ldots, f_m\}$*,* $d := \max_{i=1}^m(\deg(f_i))$ *and in Step 1 of Algorithm 1 consider the shifts* $S := \bigcup_{i=1}^m \{t \cdot f_i \mid \deg(t \cdot f_i) \leq \mathrm{Dube}(d, n)\}$ *of* F*. Then the result* G *returned by the algorithm is a Gröbner basis of* F*.*

This concludes the description of the algorithm for computing Gröbner bases via Macaulay matrices. A comparison with [13] reveals that the degree-bound we present here, namely $\mathrm{Dube}(d, n)$, is much smaller than the bound presented in the cited work. This is because the author of [13] was unaware of [1] and therefore added the general cofactor bound by Hermann [6] to Dubé's Gröbner basis bound in order to ensure that the degrees of the representations of the Gröbner basis elements stay below the given bound. In short, the overall bound obtained in [13] is $\mathrm{Dube}(d, n) + \sum_{j=0}^{n-1}(md)^{2^j}$, where m is the number of the input polynomials. Theorem 5 illustrates that the second summand is superfluous.

Remark 1. The statement of Theorem 2.3.6 in [13], in particular the part about reduced Gröbner bases, is not quite correct: Algorithm 1, together with the degree-bound presented in [13], does not always return a reduced Gröbner basis. This is because the Gröbner basis bound $\mathrm{Dube}(d, n)$ is only valid for *some* Gröbner basis, but not necessarily for the *reduced* Gröbner basis of the ideal in question. (For making assertions about reduced Gröbner bases, the input must be homogeneous; cf. Theorem 4.)

2.5 Comparison to Faugère's F_4 Algorithm

The algorithm presented in Section 2.4 bears several similarities with J.-C. Faugère's F_4 algorithm for computing Gröbner bases [5]. There, one also constructs Macaulay matrices of sets of polynomials, computes their rref, and extracts new polynomials from the rref. The main difference between F_4 and Algorithm 1 is that the former algorithm performs the Macaulay-computations many times, namely once in each iteration of the usual critical-pair/completion algorithm; the algorithm presented here, on the other hand, only constructs one single Macaulay matrix. Typically, the matrices constructed in F_4 are much smaller than the one matrix constructed in Algorithm 1, because the bound

[3] A polynomial is called *homogeneous* if all power-products in its support have the same degree.

Theorem 5 gives is in most cases way too large. This implies that F_4 usually outperforms Algorithm 1 on concrete examples, unless better bounds than Theorem 5 can be proved for the concrete input sets in question.

Note that besides Algorithm 1 we also formalized Faugère's F_4 algorithm in Isabelle/HOL [9].

3 Formalization in Isabelle/HOL

We formalized big portions of Section 2.4 in Isabelle/HOL. More precisely, we formalized Algorithm 1 and proved Theorem 6 where, however, the concrete degree bound $\mathrm{Dube}(d, n)$ is replaced by a universally quantified variable which is only assumed to be *some* feasible degree bound for the given input. This not only makes the resulting theorems more general (and thus enables us to instantiate the theorems by better bounds in special cases of the input set F), but it in particular saved us from proving Theorem 4, which is left as future work. In this section, we outline the general structure of our formalization, summarize important definitions, intermediate lemmas and final theorems, and also highlight further interesting features and technicalities.

The starting point of the formalization are the existing formalizations of multivariate polynomials and Gröbner bases [7] as well as that of matrices and Jordan normal forms [11] in Isabelle/HOL. Indeed, [7] already contains everything presented in Sections 2.1 and 2.2 of this report, in particular the definition of Gröbner bases and the formal statement and proof of Theorem 1. In addition, it also contains the definitions of Macaulay matrices and reduced row echelon forms (rrefs), which are needed in Faugère's F_4 algorithm. Although we will not present in this report how Gröbner bases are formalized in [7] (referring the interested reader to [9] instead), we do recall the formalization of Macaulay matrices for making the report as self-contained as possible.

The formalization is structured into two Isabelle theories:

- 'Macaulay-Matrix', which is in fact part of [7], defines Macaulay matrices of lists of polynomials and proves certain properties about rrefs of such Macaulay matrices. It is described in Section 3.1.
- 'Groebner-Macaulay' proves, by building upon results from 'Macaulay-Matrix', that the rref of a sufficiently large Macaulay matrix indeed gives a Gröbner basis of the input polynomials. It is described in Section 3.2

In the remainder of this section, we present these two theories in detail. It must be noted, however, that we slightly simplify the presentation in this report compared to the actual formalization, in order to keep things as simple and comprehensible as possible. Section 3.3 summarizes the most important differences.

3.1 Theory 'Macaulay-Matrix'

We begin with the formal definitions of Macaulay matrices in Isabelle/HOL right away, assuming familiarity with Isabelle's syntax:

```
definition poly_to_row :: "α list ⇒ (α ⇒₀ β) ⇒ β vec"
  where "poly_to_row ts p = vec_of_list (map (lookup p) ts)"
```

```
definition polys_to_mat :: "α list ⇒ (α ⇒₀ β) list ⇒ β mat"
  where "polys_to_mat ts ps = mat_of_rows (length ts) (map (poly_to_row ts) ps)"

definition Macaulay_mat :: "(α ⇒₀ β) list ⇒ β mat"
  where "Macaulay_mat ps = polys_to_mat (Keys_to_list ps) ps"
```

Several comments on these definitions are in place:

- First of all, the type $\alpha \Rightarrow_0 \beta$ is the type of *polynomial mappings* from α to β, which are all functions from α to β with finite support, i. e. where only finitely many arguments are mapped to values different from 0.

- Throughout the formalization, the type variable α is assumed to be the type of the power-products, and β the type of the coefficients of polynomials[4]. Hence, following [9], we abstract from any concrete view on power-products as objects of type, say, $\chi \Rightarrow_0$ `nat`, mapping indeterminates of type χ to their exponents. Instead, the type of power-products must only form a multiplicative cancellative commutative monoid – at least for the moment; only later, when introducing degree bounds in theory 'Groebner-Macaulay', the abstract view on power-products is replaced by the more concrete $\chi \Rightarrow_0$ `nat`.

- Apart from being a cancellative commutative monoid, α must also be ordered by an admissible order relation $\preceq$. In the formalization, this is achieved through a *locale*. So, all definitions, algorithm, theorems, etc. are automatically parameterized over the admissible ordering $\preceq$ implicitly fixed in the context of the locale. See [9] for more information about how this works in practice.

- In the definition of `poly_to_row`, `lookup` is the coefficient-lookup function for polynomial mappings. So, `lookup`(p, t)[5] corresponds to $\mathrm{C}(p, t)$ defined in Section 2.1.

- Constants `vec_of_list` and `mat_of_rows` are functions for constructing vectors and matrices from lists and rows, respectively. They are defined in [11]. Matrices of type β `mat` are represented as infinite mappings of type `nat` $\Rightarrow$ `nat` $\Rightarrow \beta$ with explicit dimensions attached to them; so, they are isomorphic to triples (r, c, m), where r is the number of rows, c is the number of columns, and m is the actual infinite mapping. Vectors of type β `vec` are represented analogously.

- Constant `Keys_to_list`, finally, returns the sorted (descending wrt. $\preceq$) list of all power-products occurring in its argument-list with non-zero coefficient. It is defined in terms of `keys_to_list` (with lower-case "k"), which does the same for a single polynomial.

Dually to `poly_to_row`, `polys_to_mat` and `Macaulay_mat` we then define operations for transforming vectors and matrices into polynomials and lists of polynomials, respectively:

[4] β is always tacitly assumed to belong at least to sort `zero`.

[5] We take the liberty to use standard mathematical notation instead of Isabelle notation in informal text.

```
definition list_to_fun :: "α list ⇒ β list ⇒ α ⇒ β"
  where "list_to_fun ts cs t = (case map_of (zip ts cs) t of Some c ⇒ c | None ⇒ 0)"

definition list_to_poly :: "α list ⇒ β list ⇒ (α ⇒₀ β)"
  where "list_to_poly ts cs = Abs_poly_mapping (list_to_fun ts cs)"

definition row_to_poly :: "α list ⇒ β vec ⇒ (α ⇒₀ β)"
  where "row_to_poly ts r = list_to_poly ts (list_of_vec r)"

definition mat_to_polys :: "α list ⇒ β mat ⇒ (α ⇒₀ β) list"
  where "mat_to_polys ts A = map (row_to_poly ts) (rows A)"
```

Here, `Abs_poly_mapping` converts a function of type $\alpha \Rightarrow \beta$ into a polynomial mapping of type $\alpha \Rightarrow_0 \beta$, `list_of_vec` converts a vector into a list, and `rows` returns the list of rows of its argument-matrix.

Having now defined all operations for converting between lists of polynomials and matrices, we prove many simple lemmas about these operations, for instance what the dimensions and the (i, j)-th entry of `polys_to_mat`(ts, ps) are, and that `polys_to_mat` and `mat_to_polys` are indeed inverses of each other:

```
lemma dim_row_polys_to_mat: "dim_row (polys_to_mat ts ps) = length ps"

lemma dim_col_polys_to_mat: "dim_col (polys_to_mat ts ps) = length ts"

lemma polys_to_mat_index:
  assumes "i < length ps" and "j < length ts"
  shows "(polys_to_mat ts ps) $$ (i, j) = lookup (ps ! i) (ts ! j)"

lemma polys_to_mat_to_polys:
  assumes "Keys (set ps) ⊆ set ts"
  shows "mat_to_polys ts (polys_to_mat ts ps) = ps"

lemma mat_to_polys_to_mat:
  assumes "distinct ts" and "length ts = dim_col A"
  shows "(polys_to_mat ts (mat_to_polys ts A)) = A"
```

Here, `dim_row` and `dim_col` return the number of rows and columns, respectively, of the given matrix, `$$` is the infix operator for the 0-based access of the entries of a matrix (analogous to `!` for lists and `$` for vectors), and `Keys` gives the set of all power-products occurring in its argument-set with non-zero coefficients.

Next comes the definition of the reduced row echelon form of matrices. Fortunately, this concept has already been formalized in [11], so we can simply reuse the definition there and only slightly adapt it to fit our needs:

```
definition row_echelon :: "β mat ⇒ β::field mat"
  where "row_echelon A = fst (gauss_jordan A (1ₘ (dim_row A)))"
```

As can be seen, in [11] the corresponding function is called gauss_jordan and not only returns $\mathrm{rref}(A)$ of the given matrix A, but also the invertible matrix P such that $\mathrm{rref}(A) = P \cdot A$. Since we do not need P, we simply discard it by projecting the result of gauss_jordan onto its first component, which is precisely $\mathrm{rref}(A)$.

The definition of rref in [11] uses so-called *pivot functions*:

```
definition pivot_fun :: "β mat ⇒ (nat ⇒ nat) ⇒ nat ⇒ bool"
  where "pivot_fun A f nc = (let nr = dim_row A in
                              (∀i<nr. f i ≤ nc ∧
                                (f i < nc ⟶ A $$ (i, f i) = 1 ∧
                                            (∀i'<nr. i' ≠ i ⟶ A $$ (i',f i) = 0)) ∧
                                (∀j<f i. A $$ (i, j) = 0) ∧
```

```
                              (Suc i < nr ⟶ f (Suc i) > f i ∨ f (Suc i) = nc)))"
```

Hence, a matrix A is in rref if, and only if, there exists some pivot function f for A.

Next we transfer theorems proved about gauss_jordan in [11] to their counterparts about row_echelon:

```
lemma dim_row_echelon [simp]:
  shows "dim_row (row_echelon A) = dim_row A"
    and "dim_col (row_echelon A) = dim_col A"

lemma row_space_row_echelon [simp]: "row_space (row_echelon A) = row_space A"

lemma row_echelon_pivot_fun:
  obtains f where "pivot_fun (row_echelon A) f (dim_col (row_echelon A))"
```

In particular note that constant row_space gives the row space of its argument, and that Lemma *row-space-row-echelon* hence corresponds to Theorem 2 in Section 2.3. Lemma *row-echelon-pivot-fun* states that row_echelon indeed returns rrefs.

The fact that row_echelon does not change the row space also allows us to conclude that the ideal generated by the polynomials extracted from the rref of a Macaulay matrix is the same as the ideal generated by the initial set of polynomials:

```
lemma ideal_row_echelon:
  assumes "Keys (set ps) ⊆ set ts" and "distinct ts"
  shows "ideal (set (mat_to_polys ts (row_echelon (polys_to_mat ts ps)))) =
         ideal (set ps)"
```

Besides ideals, we also need linear hulls of sets polynomials: The *linear hull* of a set $B \subseteq K[X]$ is the set of all finite linear combinations of elements in B. Hence, it is like an ideal, but the cofactors may only be constants from K. In the formalization, the linear hull generated by a set B is denoted by phull(B). The first simple lemma we can prove about the relationship between phull and ideal is that the former always gives a subset of the latter:

```
lemma phull_subset_ideal: "phull B ⊆ ideal B"
```

In addition to that, we also prove a lemma analogous to *ideal-row-echelon*:

```
lemma phull_row_echelon:
  assumes "Keys (set ps) ⊆ set ts" and "distinct ts"
  shows "phull (set (mat_to_polys ts (row_echelon (polys_to_mat ts ps)))) = phull (set ps)"
```

The importance of phull for our formalization will become clear later.

Finally, the last notion we introduce in this theory is Macaulay_list. It is defined formally as

```
definition Macaulay_list :: "(α ⇒₀ β) list ⇒ (α ⇒₀ β) list"
  where "Macaulay_list ps = filter (λp. p ≠ 0)
                  (mat_to_polys (Keys_to_list ps) (row_echelon (Macaulay_mat ps)))"
```

Macaulay_list(ps) first constructs the Macaulay matrix of the list ps, then transforms it into rref, then converts the result back into a list of polynomials, and eventually removes all occurrences of 0 from that list. The crucial properties of Macaulay_list, which will feature a prominent role later on, are as follows:

```
lemma phull_Macaulay_list: "phull (set (Macaulay_list ps)) = phull (set ps)"
```

```
lemma ideal_Macaulay_list: "ideal (set (Macaulay_list ps)) = ideal (set ps)"

lemma Macaulay_list_is_monic_set: "is_monic_set (set (Macaulay_list ps))"

lemma Macaulay_list_distinct_lp:
  assumes "p ∈ set (Macaulay_list ps)" and "q ∈ set (Macaulay_list ps)" and "p ≠ q"
  shows "lp p ≠ lp q"

lemma Macaulay_list_lp:
  assumes "p ∈ phull (set ps)" and "p ≠ 0"
  obtains g where "g ∈ set (Macaulay_list ps)" and "g ≠ 0" and "lp p = lp g"
```

The first two lemmas, *phull-Macaulay-list* and *ideal-Macaulay-list*, are mere corollaries of *phull-row-echelon* and *ideal-row-echelon*, respectively. The third lemma, *Macaulay-list-is-monic-set*, expresses that every polynomial in the resulting list is monic, i. e. has leading coefficient 1; this follows readily from the fact that row_echelon computes rrefs, and that in rrefs the pivot element of each row is 1. The last two lemmas state properties of the leading power-products of the polynomials in Macaulay_list(ps): *Macaulay-list-distinct-lp* expresses that different polynomials have different leading power-products, and *Macaulay-list-lp* expresses that for every non-zero element in the linear hull generated by the polynomials in ps, there exists a polynomial in Macaulay_list(ps) with the same leading power-product. Especially the last lemma will be of utmost importance for formally proving the main theorems about Gröbner bases and Macaulay matrices presented informally in Section 2.4 and formally in the upcoming Section 3.2.

We conclude this section with some words on the effective computability of the various functions introduced in the preceding paragraphs. Thanks to the underlying development [11], row_echelon is effectively computable, and so are mat_to_polys and polys_to_mat (and hence also Macaulay_mat and Macaulay_list) for concrete representations of multivariate polynomials, e g. by associative lists. There is still one drawback, though: in [11], only a dense representation of matrices by immutable arrays is formalized, rendering the computation of rrefs of big (but sparse) matrices practically impossible. A more sophisticated representation of the kind of sparse matrices arising in this theory, e. g. a hybrid dense-sparse representation (dense for rows, sparse for columns) by immutable arrays and associative lists, would be highly desirable and is possible future work.

3.2 Theory 'Groebner-Macaulay'

As an immediate consequence of Lemma *Macaulay-list-lp* we can observe that Macaulay_list(ps) always returns a Gröbner basis if there is *some* Gröbner basis of ps contained in phull(ps):

```
lemma Macaulay_list_is_GB:
  assumes "is_Groebner_basis G" and "ideal (set ps) = ideal G" and "G ⊆ phull (set ps)"
  shows "is_Groebner_basis (set (Macaulay_list ps))"
```

Before we can utilize this lemma to formulate and prove Theorem 6 we have to formalize the concept of *degree-bounds* for computing Gröbner bases, in order to get rid of the Gröbner basis G in the assumptions of the lemma above. To that end, as briefly mentioned at the beginning of Section 3.1, we now replace the abstract view of power-products of type α by the more concrete one as mappings of type $\chi \Rightarrow_0$ nat. So, here and henceforth χ plays the role of the type of indeterminates.

We begin by defining constants is_GB_cofactor_bound and is_hom_GB_bound as follows:

```
definition is_GB_cofactor_bound::"((χ ⇒₀ nat) ⇒₀ β) set ⇒ nat ⇒ bool"
  where "is_GB_cofactor_bound F b ⟷
              (∃G. is_Groebner_basis G ∧ ideal G = ideal F ∧
                (⋃g∈G. indets g) ⊆ (⋃f∈F. indets f) ∧
                (∀g∈G. ∃q. g = (∑f∈F. q f * f) ∧ (∀f∈F. poly_deg (q f * f) ≤ b)))"

definition is_hom_GB_bound :: "((χ ⇒₀ nat) ⇒₀ β) set ⇒ nat ⇒ bool"
  where "is_hom_GB_bound F b ⟷
              ((∀f∈F. homogeneous f) ⟶ (∀g∈reduced_GB G. poly_deg g ≤ b))"
```

As can probably be guessed from its name, is_GB_cofactor_bound(F, b) expresses that there exists some Gröbner basis G of F such that every polynomial $g \in G$ can be written as $g = \sum_{f \in F} q_f\, f$, where the products $q_f\, f$ satisfy the additional requirement that their degree be not greater than the given bound b. Theorem 5 shows that is_GB_cofactor_bound$(F, \mathrm{Dube}(d, n))$ always holds, where d and n depend on F and are as in that theorem. Instead of formalizing and proving only this known bound we opted to generalize Theorem 6 by allowing the cofactor bound for the given set F to be arbitrary, as long as it *is* a valid cofactor bound; this is ensured by adding an assumption of the form is_GB_cofactor_bound(F, b) to the formal statement of the theorem, as can be seen below. The reason for said approach is simple: although the Dubé-bound holds for *all* sets F, it can be drastically improved if additional constraints are imposed on the input sets F. For instance, in [13] Wiesinger-Widi derives much smaller bounds if F consists only of two binomials, and our setting allows us to easily obtain variants of Theorem 6 incorporating said better bounds for this special case. Note that the third conjunct in the definition of is_GB_cofactor_bound is technical and merely expresses that all indeterminates appearing in the Gröbner basis G must also appear in F; in fact, indets is an auxiliary function returning precisely the set of indeterminates appearing in its argument.

is_hom_GB_bound(F, b) expresses that b is a bound for the degrees of the elements in the reduced Gröbner basis of F, provided that F consists of homogeneous polynomials. Theorem 4 shows that is_hom_GB_bound$(F, \mathrm{Dube}(d, n-1))$ always holds. Although as of yet we have not formalized Theorem 4, we did formally prove a variant of Theorem 5: If is_hom_GB_bound(F^*, b) is true for some b, where F^* denotes the *homogenization* of F wrt. a fresh indeterminate, then b also satisfies is_GB_cofactor_bound(F, b). The formal statement of this theorem contains some odd technicalities that are difficult to explain on the spot, and so we decided to omit it here. Actually, is_hom_GB_bound only appears in this theorem, but it will not play any role later on.

From now on, we fix a finite set X of indeterminates by setting up a local theory context:

```
context
  fixes X :: "χ set"
  assumes fin_X: "finite X"
begin
```

X plays the role of the set of indeterminates in which all polynomials in the remainder are supposed to be. Hence, card(X) corresponds to the number n in

Section 2. The explicit finiteness assumption is necessary because χ, the type of indeterminates, could in principle be infinite.[6]

We will additionally use the following auxiliary concepts whose formal definitions we omit here:

- $\mathsf{P}[X]$ is the set of all polynomials of type $(\chi \Rightarrow_0 \mathsf{nat}) \Rightarrow_0 \beta$ in which only indeterminates in X appear (with non-zero exponents in power-products with non-zero coefficients). In short: $p \in \mathsf{P}[X] \longleftrightarrow \mathsf{indets}(p) \subseteq X$.

- $\mathsf{deg\_le\_sect}(X, d)$, for a natural number d, is the set of all power-products in X whose degree is less than or equal to d.

deg_le_sect enables us to define function deg_shifts:

```
definition deg_shifts :: "nat ⇒ ((χ ⇒0 nat) ⇒0 β) list ⇒ ((χ ⇒0 nat) ⇒0 β) list"
  where "deg_shifts d fs = concat (map (λf. (map (λt. monom_mult 1 t f)
                              (pps_to_list (deg_le_sect X (d - poly_deg f))))) fs)"
```

Without going into the details of its formal definition, $\mathsf{deg\_shifts}(d, \mathit{fs})$ returns a list of all shifts of the polynomials in list *fs* by power-products in X up to degree d.[7]

If d is instantiated by a valid degree-bound for the cofactors of a Gröbner basis of $\mathsf{set}(\mathit{fs})$, $\mathsf{deg\_shifts}(d, \mathit{fs})$ constructs a list of all shifts of the polynomials in the list *fs* that are needed to compute a Gröbner basis by virtue of Theorem 6. So, we finally obtain the main theorem we set out to prove:

```
theorem thm_2_3_6:
  assumes "set fs ⊆ P[X]" and "is_GB_cofactor_bound (set fs) b"
  shows "is_Groebner_basis (set (Macaulay_list (deg_shifts b fs)))"
```

Keep in mind that the theorem is stated in the local theory context set up earlier, which in particular means that X is still a *finite* set of indeterminates. The only difference between *thm-2-3-6* and Theorem 6, its informal counterpart, is the absence of the concrete Dubé-bound in the former, as announced above. Instead, the theorem holds true for all valid bounds for the given list *fs*. Formalizing such bounds is ongoing work; see Section 4.

Since all functions Macaulay_list depends upon are effectively executable (see end of Section 3.1), we now have a formally verified, executable algorithm for computing Gröbner bases by computing the rref of Macaulay matrices – at least once the missing bounds are formalized.

3.3 Differences to the Actual Formalization

In order to simplify the presentation, a couple of things have been changed here compared to the actual formalization in Isabelle/HOL. Below, we briefly summarize the two main differences. Readers not intending to look at the Isabelle-sources of the formalization may safely skip this part.

First, power-products are written *additively* rather than multiplicatively in the formalization: 0 takes the role of 1, + that of $\cdot$, etc. This deviation from common mathematical practice has technical reasons and no further implications for the rest of the formalization.

[6] This comes in handy when indeterminates shall be represented by natural numbers.

[7] Function pps_to_list turns a finite set of power-products into a sorted list; keys_to_list, which is mentioned at the beginning of Section 3.1, is defined in terms of it.

Second, big parts of the Isabelle-theories deal with *modules* and *submodules*, generalizing the traditional setting of polynomial rings and ideals. In a nutshell, this means that instead of scalar polynomials we consider vectors of polynomials represented as polynomials mappings from *terms* to coefficients. A term can be thought of as the product of a power-product and a canonical basis vector of the module, i. e. a vector whose components are all zero, except one, which is a power-product. However, just as power-products are abstracted from by using the type variable α (as described at the beginning of Section 3.1), terms are also abstracted from in the formalization; in fact, terms are represented by the type variable τ that only has to satisfy certain abstract properties (encoded in a locale). More information on the formalization of terms, vectors of polynomials, modules and submodules in Isabelle/HOL can be found in [9]. As a side result of this approach, constant ideal actually reads as pmdl (standing for 'polynomial (sub)module') in the sources. Only just before defining is_GB_cofactor_bound and is_hom_GB_bound, when the abstract view of power-products (or, more precisely, terms) is replaced by a concrete one, the module-setting is left for the traditional one of scalar polynomials and ideals.

4 Conclusion and Future Work

We presented the formalization of a method for computing Gröbner bases by transforming certain Macaulay matrices into reduced row echelon form, following [13]. This formalization is distributed across two theories in Isabelle/HOL, called 'Macaulay-Matrix' and 'Groebner-Macaulay'; the former theory consists of roughly 1200 lines of code, and the latter of roughly 500 lines, summing up to a total of 1700 lines of code. These numbers are relatively small because we could heavily build upon existing formalizations of Gröbner bases and matrices.

The method of computing Gröbner bases via Macaulay matrices depends on degree-bounds for cofactors of Gröbner bases of finite sets of polynomials, and since the dimensions of the Macaulay matrices in turn depend on these bounds, the method is only feasible if tight bounds are known; otherwise, the matrices quickly become far too big to be handled efficiently by any computer implementation. Unfortunately, all general degree-bounds, like Dubé's (Theorem 4), are double-exponential in the number n of indeterminates, meaning that for generic input systems F the method presented here is only applicable if n is small (up to around 5). For larger n we have to restrict ourselves to sets F that belong to classes of polynomial systems for which significantly smaller bounds are known. One such class contains all sets consisting of precisely two binomials; good bounds for this class are derived by Wiesinger-Widi in [13].

Although the formalization presented in this paper covers the method for computing Gröbner bases by Macaulay matrices, concrete degree-bounds are still missing. Our current goal is to formalize Dubé's general bound, and later try to do the same with Wiesinger-Widi's bounds for two binomials. The proofs given by Wiesinger-Widi in [13] are fairly technical, essentially reducing the original problem to a combinatorial problem over the lattice $\mathbb{N}^n$, and thus render their formalization in a proof assistant a challenging task.

References

[1] Matthias Aschenbrenner and Anton Leykin. Degree Bounds for Gröbner Bases in Algebras of Solvable Type. *Journal of Pure and Applied Algebra*, 213:1578–1605, 2009.

[2] Bruno Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenrings nach einem nulldimensionalen Polynomideal*. PhD thesis, Mathematisches Institut, Universität Innsbruck, Austria, 1965. English translation in Journal of Symbolic Computation 41(3-4):475–511, 2006.

[3] David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms*. Undergraduate Texts in Mathematics. Springer, fourth edition, 2015.

[4] Thomas W. Dubé. The Structure of Polynomial Ideals and Gröbner Bases. *SIAM Journal on Computing*, 19(4):750–773, 1990.

[5] Jean-Charles Faugère. A New Efficient Algorithm for Computing Gröbner Bases (F_4). *Journal of Pure and Applied Algebra*, 139(1):61–88, 1999.

[6] Grete Hermann. Die Frage der endlich vielen Schritte in der Theorie der Polynomideale. *Mathematische Annalen*, 95:736–788, 1926. English translation in ACM SIGSAM Bull. 32(3):8–30, 1998.

[7] Fabian Immler and Alexander Maletzky. Gröbner Bases Theory. *Archive of Formal Proofs*, 2016. Formal proof development.

[8] Alexander Maletzky. Gröbner Bases and Macaulay Matrices, 2018. `https://github.com/amaletzk/Isabelle-Groebner/tree/modules`.

[9] Alexander Maletzky and Fabian Immler. Gröbner Bases of Modules and Faugère's F_4 Algorithm in Isabelle/HOL. In Florian Rabe, William Farmer, Grant Passmore, and Abdou Youssef, editors, *Intelligent Computer Mathematics (Proceedings of CICM 2018, Hagenberg, Austria, August 13-17)*, volume 11006 of *Lecture Notes in Computer Science*, pages 178–193. Springer, 2018.

[10] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL—A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.

[11] René Thiemann and Akihisa Yamada. Matrices, Jordan Normal Forms, and Spectral Radius Theory. *Archive of Formal Proofs*, 2015. Formal proof development.

[12] Makarius Wenzel. *The Isabelle/Isar Reference Manual*, 2018. Part of the Isabelle documentation, `https://isabelle.in.tum.de/dist/Isabelle2017/doc/isar-ref.pdf`.

[13] Manuela Wiesinger-Widi. *Gröbner Bases and Generalized Sylvester Matrices*. PhD thesis, Research Institute for Symbolic Computation, Johannes Kepler University Linz, Austria, 2015. `http://epub.jku.at/obvulihs/content/titleinfo/776913`.