Comprehensive Gröbner bases. Improving DISPGB algorithm using the discriminant ideal.

Montserrat Manubens, Antonio Montes

MA2–IR–04–00015

Comprehensive Gröbner bases. Improving `DISPGB` algorithm using the discriminant ideal. [*]

Montserrat Manubens, Antonio Montes
Departament de Matemàtica Aplicada 2,
Universitat Politècnica de Catalunya, Spain.

Abstract

In 1992, V. Weispfenning proved the existence of Comprehensive Gröbner Bases (CGB) and gave an algorithm to compute one of them. The algorithm was not very efficient and not canonical. Using his suggestions, A. Montes obtained in 2002 a more efficient algorithm (`DISPGB`) for Discussing Parametric Gröbner Bases. Inspired in the philosophy of `DISPGB`, V. Weispfenning defined, in 2002, how to obtain a Canonical Comprehensive Gröbner Basis (CCGB) for polynomials over a ring of polynomials on a set of parameters, and provided a constructive method.

In this paper we use Weispfenning's CCGB ideas to make substantial improvements on the `DISPGB` algorithm. It now includes rewriting of the discussion tree using the *discriminant ideal* and provides a compact and effective discussion. We also describe the new algorithms in the `DPGB` library containing the improved `DISPGB` as well as new routines to check whether a given basis is a CGB or not, and to obtain a CGB.

Keywords: discriminant ideal, comprehensive Gröbner bases, parametric polynomial system.
MSC: 68W30, 13P10, 13F10.

1 Introduction

Let $R = k[\overline{a}]$ be the polynomial ring in the parameters $\overline{a} = a_1, \dots, a_m$ over the field k, and $S = R[\overline{x}]$ the polynomial ring over R in the set of variables $\overline{x} = x_1, \dots, x_n$. Let $\succ_{\overline{x}}$ be a monomial order wrt the variables $\overline{x}$, $\succ_{\overline{a}}$ a monomial order wrt the parameters $\overline{a}$ and $\succ_{\overline{xa}}$ the product order. In 1992, Professor Volker Weispfenning [We92] proved the existence of a Comprehensive Gröbner Basis CGB wrt $\succ_{\overline{x}}$ for any ideal $I \subset S$ such that for every specialization $\sigma_{\overline{a}} : R \to K'$ of the parameters, $\sigma_{\overline{a}}(\text{CGB})$ is a Gröbner basis of the specialized ideal $\sigma_{\overline{a}}(I) \cdot K'$, denoted in the following $\sigma_{\overline{a}}(I)$. He also provided an algorithm to compute it. In fact the proof in [We92] applies to more general integral domains R. There are two known implementations of this algorithm [Pe94],[Sc91].

In 1995, A. Montes [Mo95], one of the authors, used classical Gröbner basis theory to study the load-flow problem in electrical networks [Mo98]. V. Weispfenning recommended

[*]Work partially supported by the Ministerio de Ciencia y Tecnología under project BFM2003-00368, and by the Generalitat de Catalunya under project 2001 SGR 00224

him to use the Comprensive Gröbner Basis algorithm [We92][Pe94] for this problem. This was the origin of a fruitful dialog. Its use in the load-flow problem provided interesting information over the parameters, but was rather complicated and not very efficient. Moreover, it was not canonical and V. Weispfenning was interested in obtaining a canonical description of parametric Gröbner bases, i.e. not algorithm depending. Working on the subject, Montes [Mo02] provided a more efficient new algorithm (`DISPGB`) to Discuss Parametric Gröbner Bases, but it continued being not canonical. `DISPGB` produces a set of non-faithful, canonically reduced Gröbner bases (Gröbner system) in a dichotomic tree discussion about the vanishing or not of some polynomials in R, for the different specifications of specializations, each of which are semi-canonically determined. The ideas in `DISPGB` were used by V. Weispfenning [We02] to prove the existence of a Canonical Comprehensive Gröbner Basis (CCGB), as well as to give a method to obtain it. The main idea for building up the canonical tree is the obtention of an ideal $J \subset R$, structurally associated to the ideal $I \subset S$ and the order $\succ_{\overline{x}}$, which clearly separates the essential specializations not included in the generic case. Let us denote it *Weispfenning's discriminant ideal* of $(I, \succ_{\overline{x}})$. In the new Weispfenning's algorithm, J can be defined and must be computed at the beginning of the discussion using a method that is relatively time consuming.

The discriminant ideal was one of the lacks of the old `DISPGB`, and an insufficient alternative algorithm `GENCASE` was provided. In this paper, following Weispfenning, we obtain a *discriminant ideal* denoted as N, which can be determined from the data obtained after building the `DISPGB` tree using a less time consuming algorithm and, moreover, we prove that $J \subset N$. We conjecture that $J = N$. We have verified it in more than twenty different examples, and no counter-example has been found. The ideal N allows to rewrite the tree, getting a strictly better discussion.

We also prove that for a large set of parametric polynomial systems (at least for all prime ideals I) the discriminant ideal is principal, and in this case we have a unique *discriminant polynomial* to distinguish the generic case from the essential specializations. All the theoretical results commented above are detailed in section 2.

In section 3, we describe the improvements introduced in the algorithms. We made a complete revision of the ancient release, simplifying the algorithm and highly increasing its speed. New routines `CANSPEC` and `PNORMALFORM` are given that perform semi-canonical specifications of specializations and reductions of polynomials. The algorithm is completely rewritten and the flow control simplified. Further reductions of the tree, eliminating similar brother terminal vertices, are performed using the algorithm `COMPACTVER`.

Following P. Gianni [Gi87], we are interested in knowing whether some basis of I is a comprehensive Gröbner basis or not. In particular the question arises for the reduced Gröbner basis of I wrt the product order $\succ_{\overline{xa}}$. We give, in section 4, a simple algorithm `ISCGB` that uses `DISPGB` tree to answer that question. We also give an algorithm `PREIMAGE` to compute a faithful pre-image of the non-faithful specialized polynomials in the reduced bases. This allows to construct a CGB that can be chosen minimal in some sense, but not canonical. It will be interesting to compare our CGB with Weispfenning's CCGB when implemented.

Finally, in section 5, we have selected two illustrative examples that show well the power of the algorithm.

The new `DISPGB` [1] algorithm is efficient and provides a compact discussion of parametric systems of polynomial equations. An incipient version of it was presented in [MaMo04].

2 Generic case, discriminant ideal and special cases

Set $K = k(\overline{a})$ be the quotient field of R. Consider $G = \mathrm{gb}(I \cdot K, \succ_{\overline{x}})$, the reduced Gröbner basis of $I \cdot K$ wrt $\succ_{\overline{x}}$. As K is a field, it can be computed trough the ordinary Buchberger algorithm. The polynomials in G have leading coefficient 1. With this normalization g can have denominators in R. Let $d_g \in R$ be the least common multiple of the denominators of g. To obtain a polynomial in S corresponding to g, it suffices to multiply g by d_g. Following Weispfenning [We02], for each $g \in G$ we can obtain a *minimal lifting*, $a_g g$, of g such that $a_g \in R$, $a_g g \in I$ and a_g is minimal wrt $\succ_{\overline{a}}$. Doing this for all $g \in G$ we obtain G', a minimal lifting of G that Weispfenning calls the *generic Gröbner basis* of $(I, \succ_{\overline{x}})$. Of course, $d_g \mid a_g$. We will use a sub-lifting of G with the polynomials $G'' = \{d_g g \ : \ g \in G\} \subset S$, and this will be our *generic case basis*, because it is simpler to compute, and corresponds to our standard form of reducing polynomials, as it will be seen in section 3.

A specialization σ for which the set of lpp (leading power products) of the reduced Gröbner basis of $\sigma(I)$ is not equal to the set of $\mathrm{lpp}(G, \succ \overline{x})$ is a *singular specialization.*

`DISPGB` builds up a binary dichotomic tree $T(I, \succ_{\overline{x}}, \succ_{\overline{a}})$, branching at the vertices deciding about some $p \in R$ whether it is null or not. Each vertex $v \in T$ contains the pair (G_v, Σ_v). $\Sigma_v = (N_v, W_v)$ is the semi-canonical specification of the specializations in v, where N_v is the radical ideal of the current null-conditions assumed (from which all factors of polynomials in W_v are dropped), and W_v is the set of irreducible polynomials (conveniently normalized and reduced by N_v) of the current non-null conditions assumed. Considering W_v^* the multiplicative closed set generated by W_v, then $G_v \subset (W_v^*)^{-1}\,(K[\overline{x}]/N_v)$ is the reduced form of the basis of $\sigma(I)$ for the specification of the specialization $\sigma \in \Sigma_v$. At a terminal vertex, the basis G_v is the reduced Gröbner basis of $\sigma(I)$, up to normalization, for all specializations $\sigma \in \Sigma_v$.

V. Weispfenning [We02] introduces the following ideal associated to each $g \in G$:

$$J_g = \{a \in R \ : \ ag \in I\} = d_g \cdot (I : d_g g) \bigcap R$$

the second formula being computable with ordinary Gröbner basis techniques. Then the radical of the intersection $J = \sqrt{\bigcap_{g \in G} J_g}$ is used to separate the generic case in his algorithm. We shall call J *Weispfenning's discriminant ideal.* A specialization σ is said to be *essential* (for $I, \succ_{\overline{x}}$), if for some $g \in G$ is $J_g \subseteq \ker(\sigma)$.

V. Weispfenning proves the following two theorems:

W1: $J = \bigcap \{\ker(\sigma) \ : \ \sigma \text{ is essential }\}$.

W2: Let σ be an inessential specialization. Then

(i) $\sigma(G)$ is defined for every $g \in G$ and $\mathrm{lpp}(\sigma(g), \succ_{\overline{x}}) = \mathrm{lpp}(g, \succ_{\overline{x}})$.

(ii) $\sigma(G)$ is the reduced Gröbner basis of the ideal $\sigma(I)$.

[1] Release 2.3 of the library `DPGB`, actually implemented in *Maple* and available at the site http://www-ma2.upc.edu/∼montes/

On the other side, in the DISPGB tree $T(I, \succ_{\overline{x}}, \succ_{\overline{a}})$ specializations are grouped into disjoint final cases i by the specification Σ_i, and for all specializations in Σ_i the reduced Gröbner bases have the same set of lpp wrt $\succ_{\overline{x}}$.

Numerate the terminal vertices using $1 \leq i \leq k$. Final cases for which $\mathrm{lpp}(G_i, \succ_{\overline{x}}) \neq \mathrm{lpp}(G, \succ_{\overline{x}})$ are said to be *singular cases*. Denote by A the set of indexes of the singular cases:

$$A = \{1 \leq i \leq k \ : \ \mathrm{lpp}(G_i, \succ_{\overline{x}}) \neq \mathrm{lpp}(G, \succ_{\overline{x}})\}.$$

The tree, being dichotomic, provides a partition of $(K')^m$ into disjoint specifications, and thus

$$(K')^m = \bigcup_{i=1}^{k} \left(\mathbb{V}(N_i) \setminus \bigcup_{w \in W_i} \mathbb{V}(w) \right) = U_s \bigcup U_g,$$

where U_s is the set of points $\overline{a} \in (K')^m$ corresponding to singular specifications, i.e.

$$U_s(I, \succ_{\overline{x}}) = \{\overline{a} \in (K')^m \ : \ \sigma_{\overline{a}} \text{ is singular } \} = \bigcup_{i \in A} \left(\mathbb{V}(N_i) \setminus \bigcup_{w \in W_i} \mathbb{V}(w) \right).$$

Let us call $N(I, \succ_{\overline{x}}) = \mathbb{I}(U_s)$ the *discriminant ideal*. We have

Theorem 2.1. *The discriminant ideal* $N(I, \succ_{\overline{x}}) = \mathbb{I}(U_s)$ *is* $N = \bigcap_{i \in A} N_i$.

This theorem allows to compute N from the output of BUILDTREE, i.e. the first tree construction in DISPGB. (See section 3).

Proof. We prove both inclusions:

- $\subseteq$: For all $f \in N = \mathbb{I}(U_s)$ and $\overline{a} \in U_s$ is $f(\overline{a}) = 0$. Thus $\sigma_{\overline{a}}(f) = 0$ for all $\overline{a} \in U_s$. Taking now $\overline{a}$ such that $\sigma_{\overline{a}} \in \Sigma_i$ this imply that $f \in N_i$. As this can be done for all $i \in A$, it follows that $N \subseteq \bigcap_{i \in A} N_i$.
- $\supseteq$: For all $f \in \bigcap_{i \in A} N_i$ and all $\overline{a} \in U_s$ it exists $i \in A$ such that $\sigma_{\overline{a}} \in \Sigma_i$ and of course $f \in N_i$. Thus $\sigma_{\overline{a}}(f) = 0$, i.e. $f(\overline{a}) = 0$ for all $\overline{a} \in U_s$. Thus $f \in \mathbb{I}(U_s) = N$.

□

Before proving the next theorem we need the following

Lemma 2.2. *Any singular specialization* (*defined here*) *is essential* (*defined by V. Weispfenning*).

Proof. Let $\sigma_{\overline{a}}$ be a singular specialization. If it were not essential, the reduced Gröbner basis of $\sigma(I)$ would be the generic basis G by Weispfenning theorem (W2), and this is a contradiction with the definition of singular specialization. Thus $\sigma_{\overline{a}}$ is essential. □

Theorem 2.3. $J \subseteq N$.

Proof. By Weispfenning's theorem (W1), if $f \in J$ then, for all essential $\sigma_{\overline{a}}$, $f \in \ker(\sigma_{\overline{a}})$, and thus $f(\overline{a}) = 0$. So, by lemma 2.2, for all singular $\sigma_{\overline{a}}$, $f(\overline{a}) = 0$. This implies that for all $i \in A$ and $\sigma_{\overline{a}} \in \Sigma_i$ is $f(\overline{a}) = 0$, and thus $f \in \sqrt{N_i} = N_i$. Finally, by proposition 2.1, $f \in N$. □

Conjecture 2.4. *We formulate two forms*

(*i*) (*Strong conjecture*). *All essential specializations are singular.*

(*ii*) (*Weak conjecture*). $J \supseteq N$.

Proposition 2.5. *The strong formulation of conjecture 2.4 implies the weak formulation.*

Proof. If $f \in N$ then, for all $i \in A$, $f \in N_i$. Thus, for all singular specialization $\sigma_{\overline{a}}$, $f(\overline{a}) = 0$ and, if the strong form of the conjecture is true, then also for all $\sigma_{\overline{a}}$ essential, $f(\overline{a}) = 0$, and thus $f \in \ker(\sigma_{\overline{a}})$. Thus, by Weispfenning's theorem (W1), $f \in J$. □

In any case, by definition, N is discriminant, i.e. for any $\overline{a} \notin \mathbb{V}(N)$ the Gröbner basis of $\sigma_{\overline{a}}(I)$ is generic, whether any singular specification is in $\mathbb{V}(N)$. Thus, what we called minimal singular variety in [Mo02] is described by N. If the strong formulation of the conjecture is true, then every specialization σ for which $N \subset \ker(\sigma)$ is not only essential but also singular, and so the corresponding set of lpp of its reduced Gröbner basis cannot be generic.

We have tested our conjecture for more than twenty examples and we have not found any counter-example of any of the two formulations. Nevertheless the weak formulation is the most interesting one, and a failure of the strong formulation does not invalidate necessarily the weak formulation.

In most cases Weispfenning's discriminant ideal J is principal, as stated in the following

Theorem 2.6. *If $I \subset S$ is a prime ideal and the generic Gröbner basis G wrt $\succ_{\overline{x}}$ is not* [1]*, then the discriminant ideal $J(I, \succ_{\overline{x}})$ is principal, and is generated by the radical of the* lcm *of all the denominators of the polynomials in G.*

Proof. Let $g \in G$. We have $J_g = d_g \cdot (I : d_g g) \bigcap R$. If $h \in J_g$ then $d_g \mid h$ as $d_g g$ has no common factor with d_g. Thus $(h/d_g) \cdot (d_g g) \in I$. By hypothesis $d_g g \neq 1$ and I is prime. As $h/d_g \in R$, we have $h/d_g \notin I$. Thus, necessarily $d_g g \in I$ and $d_g \in J_g$. As $d_g \mid h$ for all $h \in J_g$ it follows that $J_g = \langle d_g \rangle$ is principal. As $J = \sqrt{\bigcap_{g \in G} J_g}$ is the intersection of principal ideals, the proposition follows. □

Not only prime ideals have principal discriminant ideals as the following example shows. Take

$$I = \langle ax_1 + x_2 + x_3 + b, x - 1 + ax_2 + x_3 + b, x_1 + x_2 + ax_3 + b \rangle.$$

Computing the Gröbner basis of I wrt $\mathrm{lex}(x_1, x_2, x_3, a, b)$ one sees that

$$I = \langle (a+2)x_3 + b, x_2 - x_3, x_1 + x_2 + ax_3 + b \rangle \cap \langle a - 1, x_1 + x_2 + ax_3 + b \rangle$$

and I is not prime. The generic Gröbner basis wrt $\mathrm{lex}(x_1, x_2, x_3)$ is in this case $G = [x_3 + b/(a+2), x_2 + b/(a+2), x_1 + b/(a+2)]$. Thus $d_g = a+2$ for each $g \in G$. It is easy to compute J for this example and one obtains $J = \langle (a+2)(a-1) \rangle$, which is still principal, even if I is

Routines of the old algorithm	Routines of the new algorithm	Improvements	Obsolete routines
`DISPGB` `BRANCH`	`DISPGB` `BUILDTREE` `DISCRIMINANTIDEAL` `REBUILDTREE` `COMPACTVERT`	The action of building the tree of the old `DISPGB` is done by the new `BUILDTREE`. The actual `DISPGB` includes also rebuilding of the tree (`REBUILDTREE`) and its compactatification (`COMPACTVERT`).	`GENCASE`
`BRANCH` `NEWVERTEX`	`BUILDTREE`	Better flow control, no incompatible branching.	`BRANCH`
`NEWCOND`	`CONDTOBRANCH`	More robust, ensures that no incompatible branches will start.	`NEWCOND`
`CANSPEC`	`CANSPEC`	Uses radical ideal. More robust.	
-	`PNORMALFORM`	Standard polynomial reduction wrt Σ.	
`CONDPGB`	`CONDPGB`	Uses `CONDTOBRANCH` and Weispfenning's standard pair selection.	
-	`DISCRIMINANTIDEAL`	Determines the discriminant ideal N.	
-	`REBUILDTREE`	Rebuilds the tree beginning the discussion with N.	`GENCASE` (external)
-	`COMPACTVERT`	Eliminates brother terminal vertices with the same lpp sets.	

Table 1:

not prime and has a prime component with generic Gröbner basis [1]. It would be interesting to characterize which ideals I have principal discriminant and which do not. But it is now clear that in the most interesting cases one has principal discriminants. This gives new insight into our concept of singular variety used in the algorithm [Mo02], to understand the parallelism and differences between the new Weispfenning's [We02] algorithm and `DISPGB`, and allows to improve our old algorithm.

Under that perspective, we have completely revised the old algorithm [Mo02] and obtained a much more efficient and compact discussion. An intermediate version was presented in [MaMo04]. We shall describe now the improvements introduced in the new `DPGB` library, and refer to [Mo02], that describes the old `DPGB`, for all unexplained details.

3 Improved DISPGB algorithm

In this section, we describe the improvements introduced in `DISPGB` algorithm. Table 1 summarizes the basic differences between old [Mo02] and new algorithms.

```
$T \leftarrow$ DISPGB$(B, \succ_{\overline{x}}, \succ_{\overline{a}})$
Input:
$B \subseteq R[\overline{a}][\overline{x}]$ : basis of $I$,
$\succ_{\overline{x}}, \succ_{\overline{a}}$ : termorders wrt the variables $\overline{x}$ and the parameters $\overline{a}$ respectively.
Output:
$T$: table with binary tree structure, containing $(G_v, \Sigma_v)$ at vertex $v$
BEGIN
$T := \phi$,  global variable
$v := [\,]$  # (label of the current vertex)
$\Sigma := ([\,], \phi)$  # (current specification)
BUILDTREE$(v, B, \Sigma)$ # (recursive, stores the computations in $T$)
$N :=$ DISCRIMINANTIDEAL$(T)$
COMPACTVERT$(T)$ # (compacts $T$)
REBUILDTREE$(T, N)$ # (rebuilds $T$)
COMPACTVERT$(T)$ # (compacts $T$)
END
```

Table 2:

First, we have improved the construction of the discussion tree $T(I, \succ_{\overline{x}}, \succ_{\overline{a}})$ in order to have a simpler flow control and to make it faster by avoiding unnecessary and useless time-consuming computations. This part of the algorithm is now `BUILDTREE`. In the old algorithm this was done by the recursive routine `BRANCH` and was the unique action of `DISPGB`. As we explain later, it has been strongly reformed.

Then `DISCRIMINANTIDEAL` computes the discriminant ideal $N = \bigcap_{i \in A} N_i$, which, as shown in section 2, can be determined from `BUILDTREE` output.

Then `DISPGB` calls `REBUILDTREE`. This algorithm builds a new tree setting the discriminant ideal N at the top vertex and the generic case at the first non-null vertex denoted [1]. At the first null-vertex the conditions N are added and the old tree is rebuilt under it, recomputing the specifications and eliminating incompatible branches. The result is a drastic reduction of branches in the new tree. In the old `DPGB` library, this work was partially done by the external algorithm `GENCASE`, that now has become useless.

To further compact the tree, a new algorithm `COMPACTVERT` is used. This summarizes brother terminal vertices with the same set of lpp into their father vertex. It is called before and after `REBUILDTREE`. Table 2 shows `DISPGB` algorithm.

3.1 Building up the discussion tree: BUILDTREE.

We have simplified the flow control from the ancient `DISPGB`, and dropped useless operations. Now, all the hard work of the discussion is done by the recursive algorithm `BUILDTREE` which replaces the old `BRANCH` and makes `NEWVERTEX` useless. The discussion obtained is equivalent to that given by the old `DISPGB`, but now is more compact.

It computes the discussion tree faster than the old one, because now it assembles the discussion over the coefficients of the current basis in one single algorithm, avoiding unnecessary branchings and useless computations.

```
BUILDTREE($v, B, \Sigma$)
Input:
$v$, the label of the current vertex,
$B \subseteq R[\bar{a}][\bar{x}]$, the current basis,
$\Sigma = (N, W)$ the current reduced specification.
Output: No output, but the data are stored in the global tree variable $T$.
BEGIN
$c_f$ := false
$(c_b, c_d, G, \Sigma_0, \Sigma_1)$:=CONDTOBRANCH($B, \Sigma$)
IF $c_d$ THEN  # ($c_d$ is true if all lc($g$), $g \in G$ are decided non-null, false otherwise)
    $(c_b, c_f, G, \Sigma_0, \Sigma_1)$:=CONDPGB($G, \Sigma$)
END IF
$T_v := (G, \Sigma)$  # (Store data in the global tree variable $T$)
IF $c_f$ THEN  # ($c_f$ is true if the new vertex is terminal, false otherwise)
    RETURN()
ELSE
    IF $c_b$ THEN  # ($c_b$ is true if null and non-null conditions are both compatibles)
        BUILDTREE($(v, 0), G, \Sigma_0$)
        BUILDTREE($(v, 1), G, \Sigma_1$)
    ELSE
        BUILDTREE($v, G, \Sigma_1$)  # (and BUILDTREE continues in the same vertex) [a]
    END IF
END IF
END
```

[a] In this case, if CONDPGB has already started, then the list of known S-polynomials reducing to 0 can be kept.

Table 3:

Given a set of polynomials B generating the current ideal, BUILDTREE is a recursive algorithm that takes the current basis B_v at the vertex v, specialized wrt the current reduced specification $\Sigma_v = (N_v, W_v)$, and builds a binary tree T, containing the discussion under the vertex v, and stores the data at the vertices of T. It substitutes the old BRANCH and NEWVERTEX. See table 3.

Theorem 16 in [Mo02] still applies to the reformed BUILDTREE, thus we can assert the correctness and finiteness of the algorithm.

Next we comment the most important algorithms used by BUILDTREE.

The algorithm CONDTOBRANCH substitutes the old NEWCOND and is more robust. It is used each time that BUILDTREE is recursively called and also inside CONDPGB, applying it to each new S-polynomial not reducing to zero. This avoids stopping Buchberger algorithm and saves incompatible branches.

Each time we need to know whether a given polynomial $f \in R$, for example the lc

```
$(c_b, c_d, G, \Sigma_0, \Sigma_1) \leftarrow$ **CONDTOBRANCH**$(B, \Sigma)$
Input:
$B \subseteq R[\bar{a}][\bar{x}]$, the current basis
$\Sigma = (N, W)$ a reduced specification.
Output:
$G$ is $B$ reduced wrt $\Sigma$,
$\Sigma_1$ is the reduced specification for the not null branch
$\Sigma_0$ is the reduced specification for the null branch
$c_b$ is true whenever $\Sigma_0$ exists, and false otherwise.
$c_d$ is true if all $g \in G$ have lc$(g)$ decided to not null, and false otherwise.
BEGIN
$G :=$ PNORMALFORM$(B, \Sigma)$
IF there is $g \in G$ with $l_c = \mathrm{lc}(g)$ not yet decided to not null wrt $\Sigma$ THEN
$c_d :=$ false
$(t, \Sigma_1) :=$ CANSPEC$(N_\Sigma, W_\Sigma \bigcup \{l_c\})$
$(t, \Sigma_0) :=$ CANSPEC$(\langle N_\Sigma, l_c \rangle, W_\Sigma)$
IF $t$ THEN $c_b :=$ true ELSE $c_b :=$ false ENDIF
ELSE
$c_d :=$ true
ENDIF
END
```

Table 4:

(leading coefficient) of a new S-polynomial is zero or not for the given specification, we will reduce it by $\Sigma = (N, W)$ using PNORMALFORM and then test if the remainder is or not compatible with choosing it null and non-null for the given specification using CANSPEC. The whole task is done by CONDTOBRANCH. See table 4.

At a given point BUILDTREE needs to use a Buchberger-like algorithm taking into account the specification and intending to determine a specializing Gröbner basis. This is CONDPGB (Conditional Parametric Gröbner Basis). The basic improvements on CONDPGB in the new version are: the call to CONDTOBRANCH instead of the old NEWCOND, and improving Buchberger algorithm by considering Weispfenning's normal strategy of pair selection [BeWe93]. We do not detail these improvements.

CANSPEC has also been modified. At each vertex v of the tree there is stored a pair (G_v, Σ_v), where $\Sigma_v = (N_v, W_v)$ is a specification of specializations. This means that for all $\sigma \in \Sigma_v$, $\sigma(N_v) = 0$ and $\sigma(w) \neq 0$, $\forall w \in W_v$. From the geometric point of view, a given $\Sigma = (N, W)$ describes the set of points of the affine space $\mathbb{V}(N) \setminus (\bigcup_{w \in W} \mathbb{V}(w)) \subseteq (K')^m$.

By proposition 5 in [Mo02], one can see that $\Sigma = (N, W)$ and $\Sigma' = (\sqrt{N}, W)$ describe equivalent specialization sets. And, by proposition 7, the same happens with $\widetilde{\Sigma} = (\widetilde{N}, \widetilde{W})$, where $\widetilde{N}$ has no factor laying in W and is radical, and $\widetilde{W}$ is the set of the irreducible factors of W with multiplicity one reduced modulus $\widetilde{N}$. So we choose the following representant of the specification describing equivalent specialization sets:

Definition 3.1. We will call $\Sigma = (N, W)$ a *reduced specification of specializations* if it is a specification such that

(i) $\langle N \rangle$ is a radical ideal, and $N = \mathrm{gb}(\langle N \rangle, \succ_{\overline{a}})$,

(ii) there is no factor of polynomials of $\langle N \rangle$ laying in W,

(iii) W is a set of distinct irreducible polynomials not laying in $\langle N \rangle$,

(iv) $\overline{W}^N = W$.

We note that the set W is not uniquely determined. There are infinitely many polynomials that cannot be null for the given specification. For example, suppose that $N = [a^2 - 1]$. Then, obviously, any polynomial in a with no root 1 nor -1, is not null. Perhaps previous decisions have taken $a \neq 0$, and the current reduced specification is $W = \{a\}, N = [a^2 - 1]$. The condition $a \neq 0$ is compatible with N, but in this case is redundant. We can also add to W other polynomials. Thus there is no unique reduced specification, but our choice is convenient. The task of obtaining reduced specifications is done by the reformed `CANSPEC`. See table 5.

Proposition 3.2. *Given any specification of specializations* $\Sigma = (N, W)$, *if* `CANSPEC` (Σ) *returns t = true, then it computes* $\widetilde{\Sigma}$ *a reduced specification of* Σ, *and moreover it does this in finitely many steps. Else it returns t = false.*

Proof. At the end of each step N_a is a radical ideal, W_a is a set of irreducible polynomials with multiplicity one reduced wrt N_a, so $\overline{W_a}^{N_a} = W_a$. And finally, as N_b is built by dropping from N_a all those factors lying in W_a, when the algorithm stops N_b is still radical. As at each completed step (N_b, W_b) satisfies the conditions of definition 3.1, if the algorithm returns `true`, then $\widetilde{\Sigma}$ is a reduced specification of specializations.

Let us now see that this is done in finitely many steps. The algorithm starts with $N_0 = N$. At next step it computes N_1, and then N_2, etc... These satisfy $N_0 \subseteq N_1 \subseteq N_2 \subseteq \cdots$. By the ACC, the process stabilizes. Only a finite number of factors can exist, so the action of dropping factors is also finite. □

The second necessary task is to reduce a given polynomial in S wrt Σ. This is done now in a standard form by `PNORMALFORM`. As now N is radical, to eliminate the coefficients reducing to zero for the given specification, it suffices to compute the remainder of the division by N. The non vanishing coefficients will also be reduced wrt N. Then we also eliminate all the factors that are in W in order to simplify further the polynomials. See table 6.

Nevertheless, the reduction using `PNORMALFORM` does not guarantee that all the coefficients of the reduced polynomial are not null for any specialization $\sigma \in \Sigma$. To decide if some coefficient can become 0 we need to apply `CONDTOBRANCH` and see if adding the new coefficient to the null conditions is compatible with Σ.

```
$(t, \widetilde{\Sigma}) \leftarrow$ **CANSPEC**$(\Sigma)$
Input: $\Sigma = (N, W)$ a specification not necessarily reduced.
Output:
$t$: a boolean valued variable.
$\widetilde{\Sigma}$: is a reduced specification if $t =$ true and $\phi$ otherwise (in this case
incompatible conditions have been found).
BEGIN
$N_a := N,\ N_b := \sqrt{N}$
$W_a := W,\ W_b :=$ the irreducible factors of $W$ without multiplicity and reduced wrt $N_a$;
IF $\prod_{q \in W_b} q = 0$ THEN RETURN(false,$\phi$) ENDIF
WHILE $(N_a \neq N_b$ AND $W_a \neq W_b)$ DO
$N_a := \phi$
FOR $p \in N_b$ DO
$p :=$ drop from $p$ all irreducible factors laying in $W_b$
IF $p = 1$ THEN RETURN(false,$\phi$) ENDIF
Add $p$ into $N_a$
END FOR
$W_a := W_b$
$N_b := \sqrt{N_a}$
$W_b :=$ the irreducible factors of $W_a$ without multiplicity and reduced wrt $N_b$
IF $\prod_{q \in W_b} q = 0$ THEN RETURN(false,$\phi$) ENDIF
END WHILE
$\widetilde{\Sigma} := (N_a, W_a)$
RETURN(true, $\widetilde{\Sigma}$)
END
```

Table 5:

```
$\tilde{f} \leftarrow$ **PNORMALFORM**$(f, \Sigma)$
Input: $f \in R[\bar{x}]$ a polynomial, $\Sigma = (N, W)$ a reduced specification,
Output: $f$ reduced versus $\Sigma$
BEGIN
$\widetilde{f} :=$ the product of the factors of $\overline{f}^{N}$ not laying in $W$, conveniently normalized
END
```

Table 6:

Another difficulty in PNORMALFORM is the following: suppose we have two polynomials f, g in I reducing to proportional polynomials $\sigma(f)$ and $\sigma(g)$ for every particular specialization $\sigma \in \Sigma$. (In this case we say that $\sigma(f)$ and $\sigma(g)$ are equivalent). It can happen that the reduced forms of f and g wrt Σ computed by PNORMALFORM, f_Σ and g_Σ, are not identical even if they are equivalent. For example, consider $\Sigma = (N = [a^2 - 1], W = \{a\})$. It can happen that the reductions of f and g become $ax - 1$ and $x - a$. They are not identical, but they are equivalent, because $a = 1/a$, taking into account N. It suffices to normalize with $\mathrm{lc}(p) = 1$ to obtain the same expression. PNORMALFORM is not able to reduce them to the same polynomial. Nevertheless we have the following

Proposition 3.3. *Given two polynomials $f, g \in S$ and a reduced specification Σ, consider the reduced forms f_Σ, g_Σ by* PNORMALFORM*. Then, for every $\sigma \in \Sigma$, $f_\Sigma \sim g_\Sigma$ wrt Σ iff*

(i) $\mathrm{lpp}(f_\Sigma) = \mathrm{lpp}(g_\Sigma)$ *and*

(ii) PNORMALFORM *applied to* $\mathrm{lc}(g_\Sigma)f_\Sigma - \mathrm{lc}(f_\Sigma)g_\Sigma = 0$.

Proof. If (i) holds, then $\mathrm{lc}(g_\Sigma)f_\Sigma$ and $\mathrm{lc}(f_\Sigma)g_\Sigma$ are equally normalized. If both expressions are now identical for every $\sigma \in \Sigma$ then their difference is in N and thus PNORMALFORM reduces it to zero. Obviously if one of both hypothesis fail, the reduced expressions are not equivalent wrt Σ. □

Thus PNORMALFORM does not obtain a canonical reduction of f wrt Σ, but it can canonically recognize two equivalent reduced expressions.

3.2 Reduction of brother final cases with the same lpp

In many practical computations and after applying these algorithms to many cases, we have observed that some discussion trees have pairs of terminal vertices hung from the same father vertex with same lpp set of their basis. As we are only interested in those bases having different lpp set, then each of these brother pairs, $\{v_0, v_1\}$, can be summarized in one single terminal vertex compacting them into their father v, eliminating the distinction of the latter condition decided in v.

Regarding this construction, we can define a partial order relation between two trees if one can be transformed into the other this way.

Definition 3.4. Let S and T be two binary trees. We will say that $S > T$ if

(i) T is a subtree of S with same root and same intermediate vertices, and

(ii) for each terminal vertex $v \in T$ there is in S either the same vertex $v \in S$ such that $(G_{v_T}, \Sigma_{v_T}) = (G_{v_S}, \Sigma_{v_S})$, or a subtree $\overline{S} \subset S$ pending from vertex $v \in S$ with all its terminal vertices $u \in \overline{S}$ with $\mathrm{lpp}(G_{u_{\overline{S}}}) = \mathrm{lpp}(G_{v_T})$.

So now, given a discussion binary tree T, we may find the minimum tree $\widetilde{T}$ within the set of all trees which can be compared with T regarding this relation. This is done by a recursive algorithm called COMPACTVERT.

Let us just note that the minimum tree will not have any brother terminal vertices with same lpp sets of their bases.

3.3 Rewriting the tree with the discriminant ideal

The tree T built by `BUILDTREE` can be rebuilt using the discriminant ideal N (see section 2). By theorem W2, if given $\sigma_{\overline{a}}$ there exists some $\delta \in N$ for which $\sigma_{\overline{a}}(\delta) \neq 0$, then $\sigma_{\overline{a}}(I)$ corresponds to the generic case. Thus, placing N as the top vertex denoted [ ] in the new tree T', for its non-null son vertex we will have $T'_{[1]} = (G_{[1]}, \Sigma_{[1]})$, where $G_{[1]}$ is the generic basis and $\Sigma_{[1]}$ is a union of specifications from T corresponding to

$$\Sigma_{[1]} = \{\sigma \ : \ \exists\, \delta \in N \text{ such that } \sigma(\delta) \neq 0\}.$$

No other intermediate vertices hang on this side of the top vertex. If the strong formulation of conjecture 2.4 holds, then no generic cases will hang from the first null-vertex.

The tree under the top vertex in the side of the null son, for which the choice is $\sigma(N) = 0$, will be slightly modified from the original T. The terminal vertices corresponding to singular cases that hang under it will not be modified, as for all of them, by construction, the condition is verified by the corresponding specifications.

Thus we can rebuild the tree using a recursive algorithm `REBUILDTREE` which goes through the old tree T and rewrites the new one T'. At each vertex v, it tests whether the condition N is already included in N_v. If it is the case, then it copies the whole subtree under it. Else it adds N to the null ideal N_v and calls `CANSPEC` to check whether the new condition is compatible or not. If the condition is compatible, then the basis will be reduced using `PNORMALFORM` and continue the algorithm. If it is not, then the recursion stops.

This algorithm produces a better new tree with possible less terminal cases (only generic type cases can be dropped). The reconstruction of the tree is very little time consuming.

3.4 New generalized Gaussian elimination `GGE`

We add here a short description of the improvements on the generalized Gaussian elimination algorithm `GGE`.

We realized, by analyzing the procedure of the old `GGE` (see [Mo02]), that there were some special cases for which we could know the result of the divisions at each step and thus we could skip them. These improvements reduced its computation time to the half.

But now, even though it is more efficient and faster, it has become less useful because the new improvements in `DISPGB`, detailed above, make, in general, `DISPGB` work faster without using `GGE`.

So, now the use of `GGE` within the execution of `DISPGB` is just optional (not used by default). However, it can be very useful for other applications like in the tensegrity problem shown in section 5 to eliminate some variables and simplify a given basis.

4 Comprehensive Gröbner basis

In [We02] the main goal is to obtain a comprehensive Gröbner basis, i.e. a basis of the ideal $I \subset R[\bar{x}]$ such that for every specialization of the parameters it specializes to a Gröbner basis of the specialized ideal.

With this aim, we have built an algorithm to test whether a given basis G is a comprehensive Gröbner basis for I. It is called `ISCGB`. It uses `PNORMALFORM` algorithm to specialize

```
$\tilde{B} \leftarrow$ CGB$(B,T)$
Input:
$B = \mathrm{gb}(I, \succ_{\overline{xa}})$
$F = \{(G_i, \Sigma_i) \ : \ 1 \leq i \leq k\}$ obtained from DISPGB
Output: $\tilde{B}$ a CGB of $I$
BEGIN
$\tilde{B} = B$
$\tilde{F}$ = SELECT cases from $F$ for which ISCGB$(B, \succ_{\overline{x}})$ is not a CGB.
WHILE $\tilde{F}$ is non empty DO
TAKE the first case $(G_1, \Sigma_1) \in \tilde{F}$
$\tilde{B} = \tilde{B} \bigcup \{$PREIMAGE$(g, \Sigma_1, B) \ : \ g \in G_1\}$
$\tilde{F}$ = SELECT cases from $\tilde{F}$ for which ISCGB$(\tilde{B}, \succ_{\overline{x}})$ is not a CGB.
END DO
END
```

Table 7:

G for every terminal case in the discussion tree and tests whether the lc of the specialized polynomials are non-zero wrt Σ. Then it checks if $\mathrm{lpp}(\sigma(G))$ includes the set of lpp of the reduced Gröbner basis wrt Σ for every terminal case. If so for every final case, then `ISCGB` returns `true` and else `false`.

The algorithm also informs, for a given basis $B \subset S$ of I, for which cases it is not a CGB. Thus we can compute pre-images of the polynomials in the cases for which B does not specialize to a Gröbner basis and add them to the original basis in order to obtain a comprehensive Gröbner basis.

Consider the case (G_v, Σ_v) and $g \in G_v$. To simplify notations we do not consider the subindex v. Let $H_g = \{f_1, \ldots, f_r\}$ be a basis of the ideal $I_g = I \bigcap \langle g, N \rangle$, whose polynomials are of the form $qg + n$, where $q \in S$ and $n \in \langle N \rangle$. I_g contains all the polynomials in I that can specialize to g (for those whith $\sigma(q)$ a non-null element of R for Σ). Let $f_i' = \overline{f_i}^N$. Obviously $H_g' = \{f_1', \ldots, f_r'\}$ is a basis of $\sigma(I_g)$. Using Gröbner bases techniques we can express $g \in \sigma(I_g)$ in the form $g = \sum_i \alpha_i f_i'$ where the α_i's are reduced wrt N, as we are in I_g/N. Then $h = \sum_i \alpha_i f_i$ specializes to g, and is a pre-image of g in I. This is used to build an algorithm `PREIMAGE` that computes a pre-image of g.

Combining `ISCGB` and `PREIMAGE`, we compute a CGB using the following algorithm. Let $B = \mathrm{gb}(I, \succ_{\overline{xa}})$ which is a tentative CGB[FoGiTr00],[Ka97], and $F = \{(G_i, \Sigma_i) \ : \ 1 \leq i \leq k\}$ the set of final cases of the tree discussion built up by `DISPGB`. We take them as input for the algorithm. See table 7.

This algorithm computes a CGB for the initial ideal I, although the construction is not canonical. We must comment that the construction of a CGB is much more time consuming than building the `DISPGB` tree. The reason is that for this purpose we are working wrt the product order $\succ_{\overline{xa}}$ instead of working wrt $\succ_{\overline{x}}$ and $\succ_{\overline{a}}$ separately.

5 Examples

We have widely tested the implementation on a high number of examples from the literature. We have selected here two simple significative examples. The first one is the classical robot arm, that has a very nice geometrical interpretation, and the second one is the study of a tensegrity problem described by a linear system with the trivial null solution in the generic case and has a non principal discriminant ideal.

5.1 Simple robot

The following system represents a simple robot arm (compare with the same example in [Mo02]):

$$\begin{aligned} B = [&s_1^2 + c_1^2 - 1, s_2^2 + c_2^2 - 1, l\,(s_1\,s_2 - c_1\,c_2) - c_1 + r,\\ &l\,(s_1\,c_2 + c_1\,s_2) + s_1 - z] \end{aligned}$$

Using the orders lex(s_1, c_1, s_2, c_2) and lex(r, z, l), respectively for variables and parameters, `DISPGB` produces the following outputs: The discriminant ideal is principal

$$N = J = [l\,(z^2 + r^2)].$$

The set of final cases expressed in the form $T_i = (G_i, (N_i, W_i))$ is:

$$\begin{aligned}
T_{[0,0,0]} &= ([s_2^2 + c_2^2 - 1, c_1 - r, s_1 - z],\ \ ([l, r^2 + z^2 - 1], \{\ \})),\\
T_{[0,0,1]} &= ([1],\ ([l], \{r^2 + z^2 - 1\})),\\
T_{[0,1,0,0]} &= ([l\,c_2 + 1, s_2, s_1^2 + c_1^2 - 1],\ \ ([l^2 - 1, z, r], \{l\})),\\
T_{[0,1,0,1]} &= ([1],\ \ ([l^2 - 1, r^2 + z^2], \{z, l\})),\\
T_{[0,1,1,0]} &= ([1],\ \ ([z, r], \{l + 1, l, l - 1\})),\\
T_{[0,1,1,1]} &= ([2\,l\,c_2 + l^2 + 1, 4\,(l^2 - 1)\,r\,c_1 + 2\,z\,l\,s_2 - (l^2 - 1)\,r,\\
&\quad (l^2 - 1)^2 - 4\,z^2, 4\,(l^2 - 1)\,z\,s_1 + (l^2 - 1)^2 + 4\,z^2],\\
&\quad ([z^2 + r^2], \{z, l + 1, r, l, l - 1\})),
\end{aligned}$$

$$\begin{aligned}
T_{[1]} = (&[2\,l\,c_2 + l^2 + 1 - z^2 - r^2, 4\,l^2\,s_2^2 + (l^2 - 1)^2\\
&-2\,(l^2 + 1)\,(r^2 + z^2) + (z^2 + r^2)^2,\\
&2\,(r^2 + z^2)\,c_1 - 2\,z\,l\,s_2 - r\,(r^2 + z^2 - l^2 + 1),\\
&2\,(r^2 + z^2)\,s_1 + 2\,l\,r\,s_2 + z\,(l^2 - r^2 - z^2)],\ \ ([\ ], \{l\,(r^2 + z^2)\})).
\end{aligned}$$

The generic case $T_{[1]}$ gives the usual formula for the robot and, for this problem, is the most interesting solution. It is characterized by the discriminant ideal N. The singular cases have simple geometrical interpretation, and give information about the degenerated cases.

A graphic plot of the tree is also provided in the library. At the intermediate vertices, the deciding polynomials are visualized, and at the terminal vertices the lpp sets of the reduced Gröbner bases are shown. See figure 1.

Now we apply `ISCGB` to $GB = \text{gb}(B, \text{lex}(s_1, c_1, s_2, c_2, r, z, l)$ wrt the obtained tree. The result is `false`, and a list of specializations for all the final cases is provided.

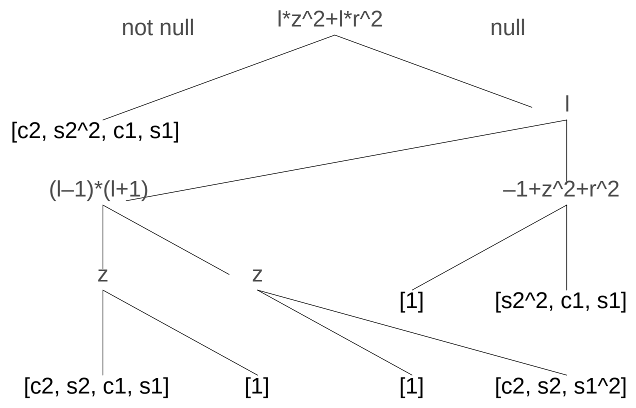

Figure 1: `DISPGB`'s graphic output for the robot arm.

$[[0,0,0], \{s_1, s_2c_1, s_2s_1, c_1, c_2s_1, s_1^2, s_2^2\}, \{s_1, c_1, s_2^2\}, \text{true}],$
$[[0,0,1], \{1, s_1, s_2c_1, s_2s_1, c_1, c_2s_1, s_1^2, s_2^2\}, \{1\}, \text{true}],$
$[[0,1,0,0], \{s_2, s_2c_1, s_2s_1, c_2s_1, c_2, s_1^2, s_2^2\}, \{s_2, c_2, s_1^2\}, \text{true}],$
$[[0,1,0,1], \{s_1, s_2, s_2c_1, s_2s_1, c_2s_1, c_2, s_1^2, s_2^2\}, \{1\}, \text{false}],$
$[[0,1,1,0], \{1, s_1, s_2, s_2s_1, c_2s_1, c_2, s_1^2, s_2^2\}, \{1\}, \text{true}],$
$[[0,1,1,1], \{s_1, s_2, s_2c_1, s_2s_1, c_2s_1, c_2, s_1^2, s_2^2\}, \{s_1, s_2, c_1, c_2\}, \text{false}],$
$[[1], \{s_1, s_2c_1, s_2s_1, c_1, c_2s_1, c_2, s_1^2, s_2^2\}, \{s_1, c_1, c_2, s_2^2\}, \text{true}]]$

There are only two cases for which GB is not a CGB. We now reconstruct the basis calling `CGB`. Only the last polynomial has been added to GB to obtain a CGB.

$$\begin{aligned}
CGB = [&2lc_2 + l^2 + 1 - z^2 - r^2,\ c_2^2 + s_2^2 - 1,\ 2(z^2+r^2)c_1 - 2zls_2 \\
&+r(l^2 - 1 - z^2 - r^2),\ 4zs_2c_1 - 4zrs_2 + 4rc_2c_1 + 4lrc_1 \\
&+2(z^2 - r^2 - 1)c_2 - l(z^2 + r^2 - l^2 + 3),\ 2rc_1s_2 - 2zc_1c_2 - 2zlc_1 \\
&+(-r^2 + z^2 - 1 + l^2)s_2 + 2zrc_2,\ 2(l^2-1)s_1 - 4lc_1s_2 + 2ls_2r \\
&-z(r^2 + z^2 - l^2 - 3),\ 2s_1z + 2c_1r - r^2 - z^2 + l^2 - 1, \\
&rs_1 - zc_1 + ls_2,\ s_1c_2 + ls_1 - c_1s_2 + rs_2 - zc_2,\ s_1s_2 + c_1c_2 \\
&+lc_1 - zs_2 - rc_2,\ c_1^2 + s_1^2 - 1,\ 4(r^2+z^2)c_1^2 - 4r(1 + z^2 + r^2 - l^2)c_1 \\
&+(r^2 + z^2 - l^2 + 1)^2 - 4z^2].
\end{aligned}$$

5.2 Tensegrity problem

We study here a problem formulated by M. de Guzmán and D. Orden in [GuOr04].

Suppose we are given the five points $P_1(0,0,0)$, $P_2(1,1,1)$, $P_3(0,1,0)$, $P_4(1,0,0)$, $P_5(0,0,1)$ and a sixth one $P_6(x,y,z)$ with unknown coordinates, to be determined under the conditions that the framework with vertices $\{P_1,\ldots,P_6\}$ and edges $\binom{\{P_1,\ldots,P_6\}}{2} \setminus \{P_1P_6, P_2P_4, P_3P_5\}$ stays in general position and admits a non-null self-stress.

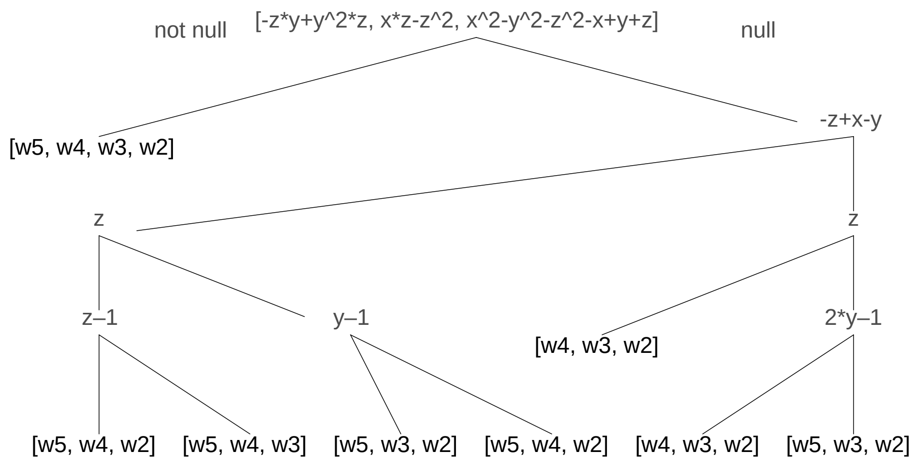

Figure 2: `DISPGB` graphic output for the tensegrity problem.

The system describing this problem is the following:

$$\begin{aligned}B = [&w_{12}+w_{14}, w_{12}+w_{13}, w_{12}+w_{15}, w_{12}+w_{23}+w_{25}-w_{26}x+w_{26},\\ &w_{12}+w_{25}-w_{26}y+w_{26}, w_{12}+w_{23}-w_{26}z+w_{26}, w_{23}+w_{34}+xw_{36},\\ &w_{13}+w_{34}-w_{36}y+w_{36}, w_{23}+zw_{36}, w_{14}+w_{34}+w_{45}-w_{46}x+w_{46},\\ &w_{34}+yw_{46}, w_{45}+zw_{56}, w_{15}+w_{45}-zw_{56}+w_{56},\\ &-w_{26}+w_{26}x+xw_{36}-w_{46}+w_{46}x+w_{56}x,\\ &-w_{26}+w_{26}y-w_{36}+w_{36}y+yw_{46}+w_{56}y,\\ &-w_{26}+w_{26}z+zw_{36}+w_{46}z-w_{56}+zw_{56}]\end{aligned}$$

In order to simplify the system, we compute the generalized Gaussian elimination of B wrt $\text{lex}(w_{12}, w_{13}, w_{14}, w_{15}, w_{23}, w_{25}, w_{34}, w_{45}, w_{26}, w_{36}, w_{46}, w_{56}, x, y, z)$.

Let us denote $w_{26} = w_2, w_{36} = w_3, w_{46} = w_4, w_{56} = w_5$. The `GGE` basis is $B' = B'_1 \cup B'_2$, where:

$$\begin{aligned}B'_1 = [&w_{45}+zw_5, w_{34}+yw_4, w_{25}+w_5y, w_{23}+zw_3, w_{15}-2zw_5+w_5,\\ &w_{14}-2zw_5+w_5, w_{13}-2zw_5+w_5, w_{12}+2zw_5-w_5]\\ B'_2 = [&-zw_5+w_5x-w_5y, -zw_5+w_4z, w_4x+yw_4-w_4-zw_5+w_5,\\ &w_3y-w_3+yw_4-2zw_5+w_5, xw_3-yw_4-zw_3,\\ &w_2z-w_2+zw_3+2zw_5-w_5, w_2y-w_2+w_5y+2zw_5-w_5,\\ &w_2x-w_2+zw_3+w_5y+2zw_5-w_5]\end{aligned}$$

B'_1 allows to express the variables $w_{12}, w_{13}, w_{14}, w_{15}, w_{23}, w_{25}, w_{34}, w_{45}$ in terms of w_2, w_3, w_4, w_5 and the parameters x, y, z. So, we only need to discuss B'_2.

Then, using the orders $\text{lex}(w_2, w_3, w_4, w_5)$ and $\text{lex}(x, y, z)$, respectively for variables and parameters, `DISPGB` produces the following outputs: The discriminant ideal is not principal

$$N = J = [y^2z-yz, zx-z^2, x^2-y^2-z^2-x+y+z].$$

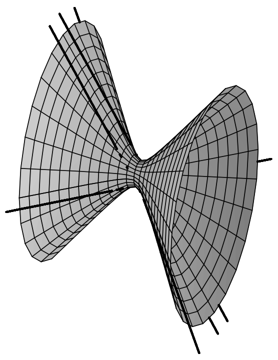

Figure 3: Location of the sixth point for non null self-stress.

The set of final cases (see figure 2) expressed in the form $T_i = (G_i, (N_i, W_i))$ is:

$$\begin{array}{lcl}
T_{[0,0,0,0]} & = & ([w_5, w_3 - w_4, w_2], ([z, 2y-1, 2x-1], \{\ \})), \\
T_{[0,0,0,1]} & = & ([w_5 + 2yw_4 - w_4, 2w_3y - w_3 + w_5, w_2 + w_5], \\
 & & ([z, x-y], \{2y-1\})), \\
T_{[0,0,1]} & = & ([-w_5 + w_4, w_3 + 2zw_5 - w_5, w_2 - 2zw_5 + w_5], \\
 & & ([y, x-z], \{z\})), \\
T_{[0,1,0,0]} & = & ([w_5, w_4, w_2], ([z, y-1, x], \{\ \})), \\
T_{[0,1,0,1]} & = & ([w_5, yw_4 + w_3y - w_3, w_2], ([z, y-1+x], \{2y-1, y-1\}), \\
T_{[0,1,1,0]} & = & ([w_5, w_4, w_3], ([z-1, y-1, x-1], \{\ \})), \\
T_{[0,1,1,1]} & = & ([w_5, w_4, w_2z - w_2 + zw_3], ([y-1, x-z], \{z, z-1\}), \\
T_{[1]} & = & ([w_5, w_4, w_3, w_2], ([\ ], \\
 & & \{[y^2z - yz, zx - z^2, x^2 - y^2 - z^2 - x + y + z]\}))
\end{array}$$

The generic solution is trivial ($w_5 = w_4 = w_3 = w_2 = 0$). In this problem, the interesting non trivial solutions are given by the conditions over the parameters described by the discriminant ideal:

$$\mathbb{V}(N) = \mathbb{V}(z, x-y) \bigcup \mathbb{V}(y, x-z) \bigcup \mathbb{V}(z, x+y-1) \bigcup \mathbb{V}(y-1, x-z)$$

These are 4 straight lines included in the hyperboloid $x^2 - y^2 - z^2 - x + y + z = 0$ illustrated in figure 3.

For this problem the Gröbner basis wrt variables and parameters is already a comprehensive Gröbner basis.

6 Acknowledgements

We want to thank Professor Volker Weispfenning for his useful suggestions and ideas and for encouraging us in undertaking the research on comprehensive Gröbner bases, as well as for his hospitality and kindness on the occasions in which we met.

We would also like to thank Professor Pelegrí Viader for his many helpful comments and his insightful perusal of our first draft.

References

[BeWe93] T. Becker, V. Weispfenning. Gröbner Bases: A Computational Approach to Commutative Algebra. Springer, New-York, 1993.

[Be94] T. Becker. On Gröbner bases under specialization. *Appl. Algebra Engrg. Comm. Comput.*, **5**:1–8, 1994.

[FoGiTr00] E. Fortuna, P. Gianni and B. Trager. Degree reduction under specialization. *Jour. Pure and Applied Algebra*, **164**(1-2):153–164. 2001. Proceedings MEGA 2000.

[Gi87] P. Gianni. Properties of Gröbner bases under specializations. In: Davenport, J.H. (ed.), *EUROCAL'87.* Springer LCNS **378**:293–297. 1987.

[GuOr04] M. de Guzmán, D. Orden. Finding tensegrity structures: geometric and symbolic aproaches. *Proceedings of EACA-2004*, 167–172. 2004.

[Ka97] M. Kalkbrenner. On the stability of Gröbner bases under specializations. *Jour. Symb. Comp.*, **24**(1):51–58. 1997.

[MaMo04] M. Manubens, A. Montes. Improving DPGB algorithm for parametric Gröbner basis. *Proceedings of EACA-2004*, 207–211. 2004.

[Mo95] A. Montes. Solving the load flow problem using Gröbner bases. *SIGSAM Bull.*, **29**:1–13, 1995.

[Mo98] A. Montes. Algebraic solution of the load-flow problem for a 4-nodes electrical network. *Math. and Comp. in Simul.* **45**:163–174, 1998.

[Mo02] A. Montes. New algorithm for discussing Gröbner bases with parameters. *Jour. Symb. Comp.*, **33**(1-2):183–208, 2002.

[Pe94] M. Pesh. Computing Comprehesive Gröbner Bases using MAS. User Manual, Sept. 1994.

[Sc91] E. Schönfeld. Parametrische Gröbnerbasen im Computeralgebrasystem ALDES/SAC-2. Dipl. thesis, Universität Passau, Germany, May 1991.

[We92] V. Weispfenning. Comprehensive Gröbner Bases. *Jour. Symb. Comp.* **14**:1–29, 1992.

[We02] V. Weispfenning. Canonical Comprehensive Gröbner bases. *Proc. ISSAC 2002.* ACM-Press, 270–276, 2002.