Tutorial for the DPGB Library for Discussing Parametric Polynomial Systems

Montserrat Manubens, Antonio Montes[*]

Departament de Matemàtica Aplicada II.
Universitat Politècnica de Catalunya.
C/ Jordi Girona 1-3, 08034-Barcelona, Spain.
{montserrat.manubens, antonio.montes}@upc.edu
Tel.: +34 934137695, +34 934137704
http://www-ma2.upc.edu/∼montes

Abstract

This is a tutorial describing the use of the DPGB software library, developed in the CAS system *Maple*, for discussing polynomial systems with parameters. This kind of problems appear very often in engineering as well as in applied mathematics. Starting with a set of polynomial equations with rational coefficients containing parameters, the main routine of the library DISPGB builds up a discussion tree containing all the necessary information about the solutions depending on the values of the parameters.

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

1 Introduction

DPGB (Discussing Parametric Gröbner Bases) is a software library developed for the CAS (Computer Algebra System) *Maple* for discussing polynomial

[*]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

systems with parameters. This kind of problem is very common in engineering (robotics, electrical networks (Mo95; Mo98), chemical models, modelling, etc.) as well as in applied mathematics (finding the dimension and size of parametric varieties, automatic theorem proving, quantifier elimination, deformation of commutative algebras, etc).

The theoretical background of the current algorithms in use today is Comprehensive Gröbner Bases (CGB) introduced by V. Weispfenning (We02). Based on these ideas, in 2000 A. Montes began to develop a more efficient algorithm DISPGB. Its basic idea is to build up a dichotomic tree using an almost canonical description of the specifications. The first public Release 1.4 is described in (Mo02).

Inspired by (Mo02), Weispfenning (We02; We03) defined a discriminant ideal that was used to obtain a canonical description of the discussion tree. Notwithstanding its canonicity, this new algorithm (CCGB) has a higher degree of complexity than DISPGB, and to date has not been implemented.

Based on the idea of discriminant ideals, DISPGB has been developed to rebuild the initial DISPGB tree to obtain a more compact one. Release 2.4 is described in (MaMo05a; MaMo05b), and uses only one discriminant to rebuild the tree. This tutorial refers to the new improved Release 3.0. which is available on the web[1]. It uses iteratively the discriminant ideals to obtain a very compact discussion. The rewritten tree, even if it is not completely canonical, is very compact. This last Release contains options to stop the rebuilding of the tree at the stage of Releases 1, 2 or at any desired level.

We provide a practical description of how to use DPGB for non specialists, and assume no specific knowledge on the theory involved. For a general comprehension of the Computer Algebra theory involved, we refer to (BeWe93; CoLiSh) as well as to the above cited references.

In Section 2, the basic concepts involved are roughly described in order to obtain a sufficient perspective on how to use the software. In Section 3, a very simple linear system with parameters is discussed using the software to show the practical interest and simplicity of using DPGB. Section 4 can be left out on a first reading. It describes how the compact discussion tree obtained in Release 3 is built from the initially computed tree, and how the initial tree can be accessed. Other facilities for a deeper understanding of the software are also discussed. In Section 5, a more interesting problem, namely the inverse kinematic problem for a simple robot is studied, and the geometrical interpretation of the discussion is analyzed. Section 6 shows how to obtain a Comprehensive Gröbner Basis (CGB) for a given parametrical system. CGB's are of more theoretical than practical interest. Nevertheless, the

[1]http://www-ma2.upc.edu/∼montes/

software is able to construct one and to test if a given basis is a CGB. Section 7 is devoted to another practical control problem: a tensegrity structure. Finally, some benchmarks for the software are presented in Section 8.

2 Description of basic concepts

In this Section we describe the basic concepts used in the software. More specific details will be given in the examples.

Let S be a set of polynomial equations in the variables $\overline{x} = x_1, \ldots, x_n$ and the parameters $\overline{a} = a_1, \ldots, a_m$ with rational coefficients. From the system $S = \{f_1(\overline{x}, \overline{a}) = 0, \ldots, f_s(\overline{x}, \overline{a}) = 0\}$ take the set of polynomials $F = \{f_1(\overline{x}, \overline{a}), \ldots, f_s(\overline{x}, \overline{a})\}$. F is said to be a basis of the *ideal* $I = \langle F \rangle$ which consists of all the polynomial consequences of the equations S, namely all the polynomials of the form $f = \sum_{i=1}^{s} h_i(\overline{x}, \overline{a})\, f_i(\overline{x}, \overline{a})$, where the h_i are arbitrary polynomials. There are infinitely many bases generating an ideal. *Gröbner bases* are the best bases for obtaining the solutions in an algebraic form. For a more precise and formal definition see Chapter 2 of (CoLiSh).

Substituting the values of the parameters by specific numbers is a *specialization* of the system. The solutions of the system depend on the specialization. Nevertheless, specializing a Gröbner basis relative to variables and parameters does not, in general, provide a specialized Gröbner basis. The algorithm `dispgb` provides *specifications* Σ for the classes of specializations of the system having the same behavior, and the corresponding specialized Gröbner bases B giving the algebraic description of the solutions. These specifications Σ of the parameters are determined by the pair (N, W), where N is the *null-condition ideal* and W is a *set of non-null conditions*.

In order to obtain the discussion of S we need to choose a monomial order $\succ_{\overline{x}}$ in the variables and another $\succ_{\overline{a}}$ in the parameters. Usually one chooses lexicographical orders for variables and parameters. In *Maple* they are denoted `xord := plex(`$x_1, \ldots, x_n$`)` and `pord := plex(`$a_1, \ldots, a_m$`)`.

Then we assign the result of the call to the basic routine `dispgb` of the library to a variable T. The call is

```
> T:=dispgb(F,xord,pord):
```

The routine `dispgb` builds up a binary dichotomic tree T containing all the necessary information about the discussion. The inner vertices of T correspond to dichotomic decisions about certain polynomials in the parameters. These polynomials are supposed null in the right branch and not null in the left branch. In the first example we will specify how to interpret these decisions.

Vertices in the tree are labelled as a sequence of 0 and 1. For example, a vertex v that is located at distance 3 from the top vertex for which two consecutive null decisions and a non-null decision over the parameters have been taken is labelled [0,0,1]. The decision taken to branch, the actual specification $\Sigma = (N, W)$ and the specialized base B are stored at the vertex in the tree structure T.

All the data stored in the tree T can be directly accessed. Nevertheless it is not convenient to print out the whole content of the table T. There exist two auxiliary output routines `tplot` and `finalcases` which select the relevant data of T in an ordered way.

The routine `tplot` plots a graphical output of the discussion tree. At each branching vertex v, the branching condition is shown (in red). `tplot` also prints the leading power products of the fundamental polynomials of the specialized systems (Gröbner bases) at the corresponding terminal vertices (in black). This provides a first simple scheme of the discussion.

Terminal vertices represent the specifications of the initial system in which the different types of solutions are grouped. `finalcases` outputs a list of lists containing the most relevant algebraic data stored at each terminal vertex. For a detailed description of its output see Section 3.

It must be noted that the discussion is done over the complex field $\mathbb{C}$. In most practical cases we are interested only in the real solutions. The restriction to $\mathbb{R}$ must be studied particularly for each case.

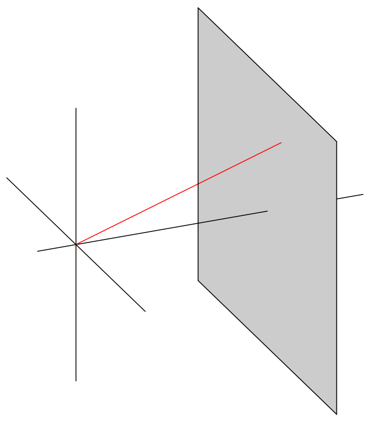
First, the `dpgb` package must be read to start using the library:

```
>  read(‘<path>/Dpgb30.mpl‘):
>  with(dpgb);
>  libname:=libname,‘<path>‘:
```

The last call includes the `maple.hdb` help library of `dpgb` –that must be located at the corresponding help path– to the name `libname` that includes all the help calls that are available by *Maple* in the session and will allow to use help calls on the routines of `dpgb` in the form `?dpgb`, or the name of a concrete routine.

3 A ray tracing example

Aas a first example let us consider the following linear system which describes the impact of a simple ray tracing model to the plane $x = 1$ with the virtual viewpoint located at the origin.



Calling (v_x, v_y, v_z) the direction of the ray, and (x, y, z) the impact point of the ray on the plane $x = 1$, the system is:

$$\begin{cases} x & = & 1 \\ v_y\, x - v_x\, y & = & 0 \\ v_z\, x - v_x\, z & = & 0 \end{cases}$$

This example is very simple and easy to compute by hand. Nevertheless it is instructive to compare with the automatic computation.

Consider the corresponding polynomial basis

$$F := [x - 1,\ v_y\, x - v_x\, y,\ v_z\, x - v_x\, z].$$

Now define monomial termorders for the variables and the parameters:

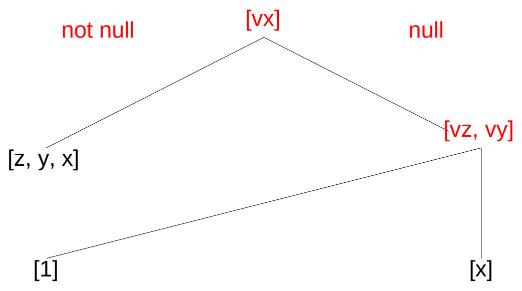
```
> xord:=plex(x,y,z):  pord:=plex(vx,vy,vz):
```

Then use the routine `dispgb` to build up the discussion tree.

```
> T:=dispgb(F,xord,pord):
```

The `tplot` routine provides a graph showing the essential features of the discussion:

```
> tplot(T);
```



This scheme makes the discussion easy to understand. At the vertices the *discriminant ideals* are shown (in red). Discriminant ideals separate the generic case (in the given specialization) from the more specialized cases. A more detailed description of discriminant ideals is given in Section 4.

The top discriminant ideal is $\Delta_{[\ ]} = [v_x]$, meaning that, at vertex [ ], the decision $v_x = 0$ is taken on the right *null* branch and $v_x \neq 0$ on the left *non-null* branch. It is clear that this is a dichotomic branching. The discussion proceeds by computing the new discriminant ideal $\Delta_{[1]} = [v_z, v_y]$. *All* polynomials in $\Delta_{[1]}$ are considered null on the right null branch and *at least one* of them non-null on the left non-null branch. This is again a dichotomic decision. In this example, the discriminant $\Delta_{[\ ]} = [v_x]$ at the top level is principal (it consists of a single polynomial), but this is no longer so at vertex [1], where it consists of two polynomials.

At the terminal vertices, explicit proper non-overlapping specifications are obtained for which the solutions are of the same kind. At them, only the set of leading power products (lpp) of the polynomials in the specialized Gröbner basis of the solutions are shown, as they provide the basic features of the solutions. As it can be observed in the picture, the discussion provided by Release 3 is totally compact. To obtain a deeper comprehension on how the algorithm works, we comment in Section 4 how this discussion is obtained from the initial one.

To display the algebraic content of the tree we call the routine `finalcases`. This routine outputs a list of a lists containing the most important algebraic objects for each terminal case. The call is:

```
>  fc:=finalcases(T);
```

The result can be visualized as:

$$\begin{array}{l}
Case = [\,[1],\, [-v_z + v_x\, z,\, -v_y + v_x\, y,\, x-1],\, [\ ],\, \{v_x\},\, [z,\, y,\, x]\,] \\
Case = [\,[0,1],\, [1],\, [v_x],\, \{[v_y,\, v_z]\},\, [1]\,] \\
Case = [\,[0,0],\, [x-1],\, [v_z,\, v_y,\, v_x],\, \{\ \},\, [x]\,]
\end{array}$$

There are 3 cases in this example, respectively labelled [1], $[0,1]$ and $[0,0]$.

For each terminal vertex v the corresponding list contains the following objects:

$$[\langle\text{label of } v\rangle,\ \langle\text{basis}\rangle, \langle\text{null conditions}\rangle, \langle\text{non-null conditions}\rangle,\ \langle\text{lpp}\rangle\rangle]$$

There exist some options to get more information from the tree (namely the history on the assumptions or the decomposition of each case). Let us now explain the 5 default objects provided by `finalcases`:

1. The label, denoted with zeroes and ones, records the decisions leading up to the current node.

2. The reduced Gröbner basis of the specialized system. For example, in the generic case [1] it is $B = [-v_z + v_x\, z,\ -v_y + v_x\, y,\ x - 1]$.

3, 4. The specification summarizing the complete specialization of the case in the semi-canonical form $N,\ W$, where N is the reduced Gröbner basis of the null condition ideal given as a list, and W is a set of non-null conditions. W is presented in the form

$$\{c_1, \dots, c_t, [c_{t+1}, \dots, c_{t+s}]\}.$$

This must be understood as meaning that all the polynomials c_i for $1 \leq i \leq t$ are non-null, and at least one of the polynomials c_i for $1 + t \leq i \leq t + s$ is non-null. Notice that either t or s or both can be 0.

5. The set of lpp of the current specialized Gröbner basis. This set characterizes the degrees of freedom of the solution. For example, in the generic case [1] it is $[x, y, z]$ which corresponds to a consistent system with unique solution (Cramer). In the case $[0, 1]$ the lpp set is $[1]$, an inconsistent system (i.e. without solution). And in the case $[0, 0]$ its lpp set is $[x]$. This represents a solution with two degrees of freedom.

It is easy to give a geometrical interpretation of the discussion using the above outlined data: The generic case is obtained by the assumption $v_x \neq 0$. This is the unique condition ensuring that the ray intersects the plane, and the solution is easily obtained from the basis, namely: $z = v_z/v_x$, $y = v_y/v_x$, $x = 1$, as expected. Case $[0, 1]$ corresponds to $v_x = 0$ and at least one of v_y or v_z not null. This represents a ray parallel to the plane with no intersection whose direction is defined by a non null vector. Finally, case $[0, 0]$ is a degenerated case corresponding to a ray defined by a null vector. The solution is now the entire plane $x = 1$.

4 From Release 1 to Release 3

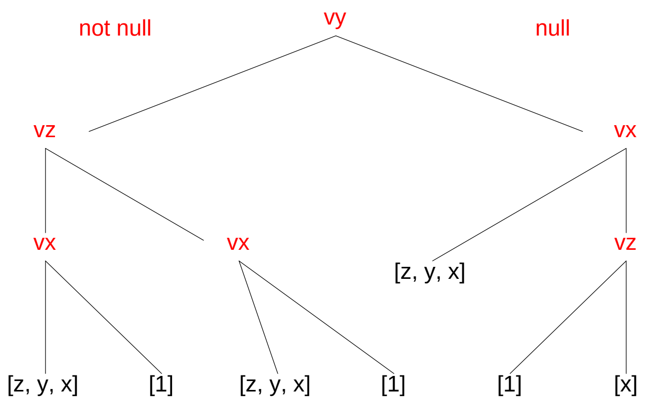
We recommend to read this Section to readers interested in a deeper knowledge of the algorithm or in obtaining faster results.

The use of *discriminant ideals* was introduced by Weispfenning (We03) and a different kind of discriminant ideal was implemented for the first time in Release 2 of `dpgb` (MaMo05a; MaMo05b) and used in the whole tree in the current Release 3. At the top level, the discriminant ideal provides the most general condition ensuring that whenever some polynomial in it is non-null, the system specializes to the generic case. It is called *discriminant* ideal because for all special cases all the polynomials in it are zero, and whenever some of them is non-null then the specialization is generic.

The initial tree built up by `dispgb` does not use discriminant ideals at all but uses simple polynomials to branch at each step. Release 2 first computes the discriminant ideal from the data in the initial tree, and outputs a rewritten tree containing the discriminant ideal at the root node. In Release 3 the rebuilding technique is iterated at each level and there all the decisions are based on discriminant ideals resulting a more compact discussion tree.

The different rebuilt forms of the tree can be obtained with the option `rebuild =` n. We give here the results for the preceding example. The first tree is:

```
>  T0:=dispgb(F,xord,pord,rebuild=0);
>  tplot(T0);
```

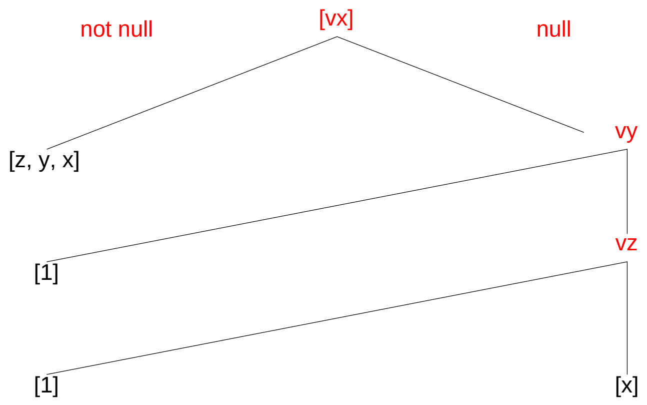


```
>  finalcases(T0);
```

$$Case = [\,[1,1,1],\ [-v_z + v_x\,z,\ -v_y + v_x\,y,\ x-1],\ [\,],\{v_y,\ v_x,\ v_z\},\ [z,\ y,\ x]\,]$$
$$Case = [\,[1,1,0],\ [1],\ [v_x],\ \{v_y,\ v_z\},\ [1]\,]$$
$$Case = [\,[1,0,1],\ [z,\ -v_y + v_x\,y,\ x-1],\ [v_z],\ \{v_y,\ v_x\},\ [z,\ y,\ x]\,]$$
$$Case = [\,[1,0,0],\ [-v_y],\ [v_z,\ v_x],\ \{v_y\},\ [1]\,]$$
$$Case = [\,[0,1],\ [-v_z + v_x\,z,\ y,\ x-1],\ [v_y],\ \{v_x\},\ [z,\ y,\ x]\,]$$
$$Case = [\,[0,0,1],\ [-v_z],\ [v_y,\ v_x],\ \{v_z\},\ [1]\,]$$
$$Case = [\,[0,0,0],\ [x-1],\ [v_z,\ v_y,\ v_x],\ \{\},\ [x]\,]$$

The first rewritten tree can be accessed with the call

```
>  T1:=dispgb(F,xord,pord,rebuild=1):  tplot(T1);
```



```
>  fc1:=finalcases(T1):
```

$$Case = [\,[1],\ [-v_z + v_x\, z,\ -v_y + v_x\, y,\ x-1],\ [\,],\ \{v_x\},\ [z,\ y,\ x]\,]$$
$$Case = [\,[0,1],\ [1],\ [v_x],\ \{v_y\},\ [1]\,]$$
$$Case = [\,[0,0,1],\ [-v_z],\ [v_y,\ v_x],\ \{v_z\},\ [1]\,]$$
$$Case = [\,[0,0,0],\ [x-1],\ [v_z,\ v_y,\ v_x],\ \{\},\ [x]\,]$$

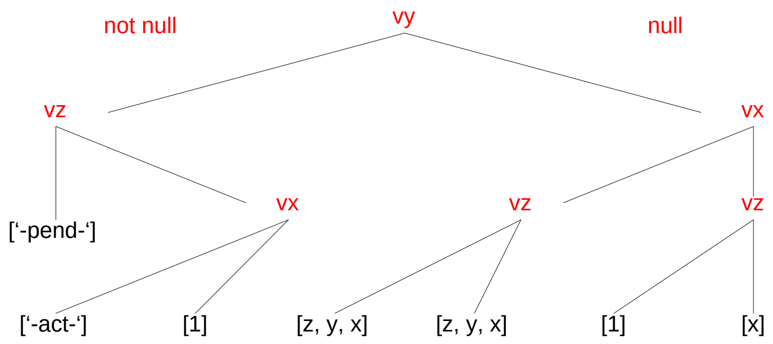
Observe that using `rebuild=1` option, the root discriminant ideal $[v_x]$ is already used and the tree is more compact than without rebuilding but less than the complete rebuilding.

We can also stop the computation of the tree during the execution of the algorithm. In this case, the already computed parts of the tree can be accessed by a global variable called __UU. Let us see what happens if we stop the computation before it terminates:

```
>  dispgb(S,xord,pord,rebuild=0):

Warning, computation interrupted

>  tplot(__UU);
```



The computation of the tree is carried out in pre-order and, by default, beginning by the null branch. As shown in the picture, the computation was stopped when the vertex $[1,0,1]$ was being evaluated and vertex $[1]$ was not yet started.

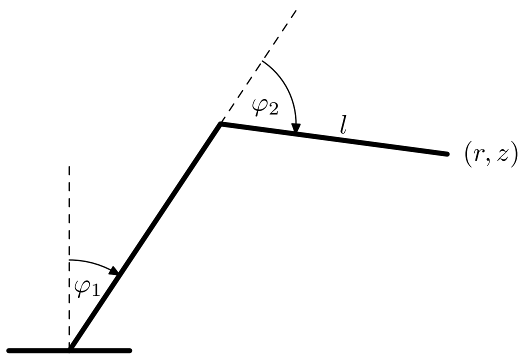


Figure 1: A two arms robot.

5 Inverse kinematic problem of a simple robot

We consider now a more interesting problem for which the system of equations is not linear in the variables, namely the simple two arms robot of Figure 1.

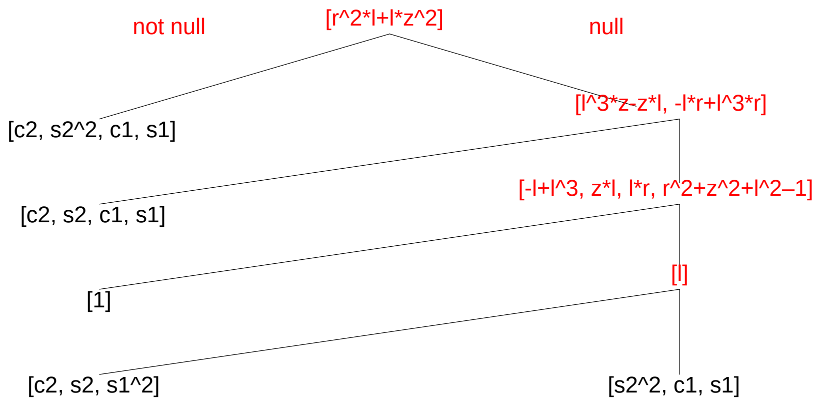
The inverse kinematic problem consists of finding the angles φ_1, φ_2 of the robot arms to reach a given position (r, z). This is of practical interest for commanding the robot. Thus we take the angles as variables and the position as parameters. We also add l as a parameter, which models that the second arm has variable length.

The system of equations describing this problem is:

$$\begin{cases} r = \cos\varphi_1 + l\ \cos(\varphi_2 - \varphi_1) \\ z = \sin\varphi_1 + l\ \sin(\varphi_2 - \varphi_1) \end{cases}$$

A typical technique for transforming this trigonometric system into a polynomial one consists of setting $c_i = \cos\varphi_i$, $s_i = \sin\varphi_i$, for $i = 1, 2$ and adding the two equations $\sin^2\varphi_i + \cos^2\varphi_i = 1$ for $i = 1, 2$. We obtain an equivalent polynomial system F in the variables c_1, s_1, c_2, s_2 to which we can apply the algorithm:

```
> F:=[r-c1-l*c1*c2+l*s1*s2,z-s1-l*c1*s2-l*s1*c2,
> c1^2+s1^2-1,c2^2+s2^2-1]:
> xord:=plex(s1,c1,s2,c2):  pord:=plex(r,z,l):
> T:=dispgb(F,xord,pord):
> tplot(T);
```



```
> finalcases(T);
```

$$Case = [\,[1], [l^2 + 2\,l\,c_2 + 1 - z^2 - r^2, 4\,l^2\,s_2^2 - 2\,l^2 + l^4 - 2\,z^2\,l^2 \\ -2\,l^2\,r^2 + 1 - 2\,z^2 - 2\,r^2 + z^4 + 2\,r^2\,z^2 + r^4, \\ l^2\,r - r\,z^2 - r^3 - r - 2\,z\,l\,s_2 + (2\,r^2 + 2\,z^2)\,c_1, \\ l^2\,z - z^3 - z\,r^2 - z + (2\,r^2 + 2\,z^2)\,s_1 + 2\,l\,s_2\,r], \\ [\,], \{l,\ r^2 + z^2\}, [c_2,\ s_2^2,\ c_1,\ s_1]\,]$$

$$Case = [\,[0,1],\ [1 + l^2 + 2\,l\,c_2,\ r - l^2\,r + 2\,z\,l\,s_2, 1 - 4\,z^2 - 2\,l^2 + l^4 + (4\,l^2\,r - 4\,r)\,c_1, \\ l^4 - 2\,l^2 + 1 + 4\,z^2 + (4\,l^2\,z - 4\,z)\,s_1],\ [r^2 + z^2], \\ \{l,\ l - 1,\ l + 1,\ r, z\},\ [c_2,\ s_2,\ c_1,\ s_1]\,]$$

$$Case = [\,[0,0,1],\ [1],\ [z\,l\,(l-1)\,(l+1),\ l\,r\,(l-1)\,(l+1),\ l\,(r^2 + z^2)], \\ \{[l\,(l-1)\,(l+1),\ z\,l,\ l\,r,\ r^2 + z^2 + l^2 - 1]\},\ [1]\,]$$

$$Case = [\,[0,0,0,1],\ [l\,c_2 + 1,\ s_2,\ c_1^2 + s_1^2 - 1],\ [(l-1)\,(l+1),\ z,\ r], \{l\}, \\ [c_2,\ s_2,\ s_1^2]\,]$$

$$Case = [\,[0,0,0,0],\ [c_2^2 + s_2^2 - 1,\ -r + c_1,\ -z + s_1],\ [l,\ -1 + z^2 + r^2],\ \{\ \}, \\ [s_2^2,\ c_1,\ s_1]\,].$$

It is now easy to make a geometrical interpretation of the solutions.

The generic case [1] corresponds to $l \neq 0$ and the end of the arm not at the origin: $r^2 + z^2 \neq 0$. In this case, two solutions, corresponding to the two symmetrical positions of the arms relative to the direction (r, z) are obtained which can be computed by finding the variables from the basis:

$$\left\{ \begin{array}{lcl} c_2 & = & \frac{1}{2\,l}(r^2 + z^2 - l^2 - 1) \\ s_2 & = & \pm\frac{1}{2\,l}\sqrt{((r^2 + z^2 - (l-1)^2)\,((l+1)^2 - (r^2 + z^2))} \\ c_1 & = & \frac{1}{2(r^2+z^2)}\,(2\,z\,l\,s_2 + r\,(r^2 + z^2 - l^2 + 1)) \\ s_1 & = & \frac{1}{2(r^2+z^2)}\,(-2\,r\,l\,s_2 + z\,(r^2 + z^2 - l^2 + 1)) \end{array} \right.$$

The two solutions correspond to the two signs of the angle φ_2. Then the angle φ_1 is uniquely determined by fixing the sign of s_2.

It must be noticed that these solutions always exist in the complex field, but in practice we are only interested in real ones. So, for real solutions, we must restrict to values of $-1 \leq c_2 \leq 1$ in the formula. Using the formula for $\cos\varphi_2$ this implies $(l-1)^2 \leq r^2+z^2 \leq (l+1)^2$. This restriction has an evident geometrical interpretation, namely the end of the arm must lie between the circles of radius $l-1$ and $l+1$. This also ensures real values for the two angles.

Consider now the rest of special cases.

The case $[0,1]$ corresponds to the specification $r^2+z^2=0$, and $l, l-1, l+1, r, z \neq 0$. This has no real solutions in the parameters. Its solutions are of the form $z = i\,r$ over $\mathbb{C}$. So we need not consider $[0,1]$ for the real case.

The case $[0,0,1]$ summarizes all the inconsistent cases ($B=[1]$). Nevertheless the compact form obtained with the complete rebuild of the tree contains three cases, that become apparent with the option `rebuild = 2`, and can also be retrieved from the complete rebuild tree using the `splitcase` routine. Applying it to the case $[0,0,1]$ the specification splits into the following ones:

```
>  splitcase([0,0,1],T);
```

$$[\,[\,[z,r],\ \{l,l-1,l+1\}],\ [[(l-1)(l+1),r^2+z^2],\ \{l,z,r\}],\ [[l],\ \{z^2+r^2-1\}]\,]$$

Let us give a geometrical interpretation of all these special sub-cases. The first one corresponds to the origin $z=r=0$ with $l \neq 1$ and $l \neq -1$, which is obviously inconsistent. The second sub-case corresponds, as in case $[0,1]$, to $r^2+z^2=0$ and $r, z \neq 0$, but now also $l=1$ or $l=-1$. This sub-case is not only complex ($z = i\,r$) but also inconsistent because of the specification of l. The third sub-case corresponds to a length 0 of the second arm, which implies $r^2+z^2=1$. However, the specification is just $r^2+z^2-1 \neq 0$ so this is also obviously inconsistent.

The case $[0,0,0,1]$ corresponds to the origin $r=z=0$ and $l=1$ or $l=-1$ (and $l \neq 0$ which is a redundant specification). Now clearly the angle φ_1 is free, and the angle $\varphi_2=\pi$ for $l=1$ and $\varphi_2=0$ for $l=-1$ (this second option has no proper geometrical interpretation as lengths are considered positive). This is exactly the result provided by the automatic discussion.

The case $[0,0,0,0]$ represents the degenerated case with arm length $l=0$ and the end-point on the circle $z^2+r^2=1$. In this case, the angle φ_2 is free and the angle φ_1 is determined by its sine and cosine from $\cos\varphi_1 = z$, $\sin\varphi_1 = r$.

6 Comprehensive Gröbner Bases

A *Comprehensive Gröbner Basis* (CGB) of an ideal is a basis specializing to a Gröbner basis for every specialization of the parameters, i.e. when the parameters are replaced, the result is always a basis that contains the fundamental polynomials from which the algebraic content of the solutions can be directly obtained. It must be noticed that the specialized bases will not be reduced and can contain redundant polynomials.

A first candidate for a CGB is the Gröbner basis computed wrt the product order of variables and parameters. It is well known (Gi87) that it does not always specialize to a Gröbner basis. Weispfennning (We92) gave the first algorithm to obtain a CGB and more recently he has given an algorithm to compute a Canonical Comprehensive Gröbner Basis (We03) (CCGB). This algorithm has high complexity and has not been implemented yet. Nevertheless, `dispgb` is more efficient and can also be used to compute a CGB, which is however not necessarily canonical.

CGB has a theoretical interest but, in practice, we are interested in the discussion of the solutions rather than in a CGB. Anyway, its computation is currently feasible with the `dpgb` library although it is very expensive in computing time. The call to `cgb` makes an internal call to `dispgb` with the option `rebuild=0` to obtain the discussion tree. The remaining algorithm uses the product order combining variables and parameters what is much more time consuming than working separately wrt variables and parameters. First it computes the Gröbner basis wrt the product order. Then it detects the cases not specializing to Gröbner bases and finally it builds up pre-images of the corresponding polynomials.

Let us consider the same example of the ideal defined in Section 5:

$$F = [r - c_1 - l\,c_1\,c_2 + l\,s_1\,s_2, z - s_1 - l\,c_1\,s_2 - l\,s_1\,c_2, c_1^2 + s_1^2 - 1, c_2^2 + s_2^2 - 1]$$

Using the `dpgb` library

```
>  G:=cgb(F,xord,pord,'numpols');
```

the result is:

$$\begin{aligned} G := [&c_1^2 + s_1^2 - 1,\ c_2^2 + s_2^2 - 1,\ 4\,s_1\,z^3 + 4\,c_1^2\,r^2 + 4\,c_1^2\,z^2 + 4\,z\,s_1\,r^2 - 2\,l\,c_2\,r^2 \\ &-2\,l\,c_2\,z^2 + 4\,l^2\,c_1\,r - l^2\,r^2 - l^2\,z^2 + l^4 - 4\,c_1\,r - r^2 - 5\,z^2 - 2\,l^2 + 1, \\ &2\,c_1\,r^2 + 2\,c_1\,z^2 - 2\,z\,l\,s_2 - r + l^2\,r - r\,z^2 - r^3, \\ &l^2 + 2\,l\,c_2 + 1 - z^2 - r^2, 2\,s_1\,z + 2\,c_1\,r - r^2 - z^2 + l^2 - 1, \\ &s_2\,s_1 + c_2\,c_1 + l\,c_1 - s_2\,z - c_2\,r, s_1\,r - z\,c_1 + l\,s_2, \\ &4\,s_2\,z\,c_1 + 4\,c_2\,c_1\,r + 4\,l\,c_1\,r - 4\,s_2\,z\,r - 2\,c_2\,r^2 + 2\,c_2\,z^2 - 2\,c_2 - l\,r^2 - l\,z^2 \\ &+l^3 - 3\,l,\ -2\,c_2\,z\,c_1 - 2\,z\,l\,c_1 + s_2\,z^2 + 2\,z\,c_2\,r + 2\,s_2\,r\,c_1 - s_2\,r^2 + l^2\,s_2 - s_2, \\ &2\,l^2\,s_1 - 2\,s_1 - 4\,l\,c_1\,s_2 + 2\,l\,s_2\,r - r^2\,z - z^3 + l^2\,z + 3\,z, \\ &c_2\,s_1 + s_1\,l - s_2\,c_1 + s_2\,r - c_2\,z] \end{aligned}$$

The optional variable `'numpols'` takes the value 1, meaning that only 1 single polynomial has been added to the Gröbner basis wrt variables and parameters to obtain a CGB.

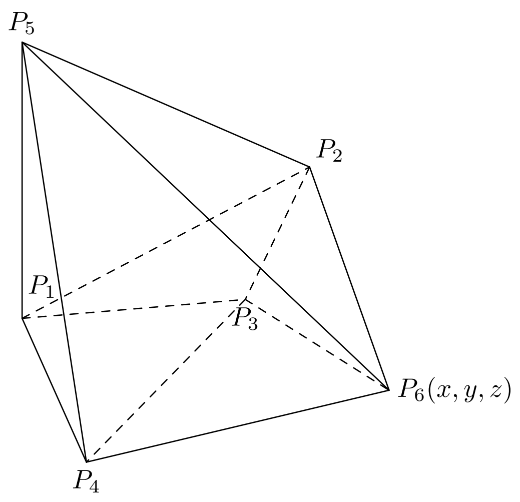


Figure 2: Tensegrity framework: oblique triangular prism.

To understand the concept of CGB we may explore what happens if we specialize G to $l = 0$, $r = 1$, $z = 1$, which corresponds to the case [0,0,1] and has no solutions, as we know from the discussion in Section 5.

```
> G0:=subs(l=0,r=1,z=1,G);
```

$$\begin{aligned}G0 := [&c_1^2 + s_1^2 - 1,\ c_2^2 + s_2^2 - 1,\ 8\,s_1 + 8\,c_1^2 - 4\,c_1 - 5, 4\,c_1 - 3,\ -1,\\ &2\,s_1 + 2\,c_1 - 3,\ s_2\,s_1 + c_2\,c_1 - s_2 - c_2,\ s_1 - c_1,\\ &4\,s_2\,c_1 + 4\,c_2\,c_1 - 4\,s_2 - 2\,c_2,\ -2\,c_2\,c_1 - s_2 + 2\,c_2 + 2\,s_2\,c_1,\\ &-2\,s_1 + 1,\ c_2\,s_1 - s_2\,c_1 + s_2 - c_2]\end{aligned}$$

We can observe that G_0 contains the constant polynomial -1. This implies that the system is not consistent ($-1 \neq 0$), as we wanted to show. If we directly specialize F the result would not be explicit.

7 A tensegrity problem

Consider now the following tensegrity problem (GuOr04): given the five points

$$P_1(0,0,0),\ P_2(1,1,1),\ P_3(0,1,0),\ P_4(1,0,0),\ P_5(0,0,1)$$

determine a sixth one $P_6(x,y,z)$ such that the framework (Figure 2) with vertices $\{P_1, \ldots P_6\}$ admits non-null scalar stresses w_{ij} assigned to the edges forming a complete graph minus the edges $\{P_1\,P_6,\ P_2\,P_4,\ P_3\,P_5\}$ (non-null self-stress problem). The equilibrium condition must be fulfilled at every vertex i:

$$\sum_{j\,:\,ij\text{ edge}} w_{ij}(P_i - P_j) = 0$$

The system of equations corresponding to this particular example is:

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

S is a linear homogeneous system in the variables w_{ij} with the parameters x, y, z. The generic solution is clearly inconsistent as the system is over-determined. It is easy to eliminate variables using the *generalized Gaussian elimination* (`gge` routine) implemented in the library. The `gge` routine performs iterated pseudo-divisions of the given polynomials wrt the product order and obtains a new simpler basis of the initial ideal. Set

$$\text{xpord} := \text{plex}(w_{12}, w_{13}, w_{14}, w_{15}, w_{23}, w_{25}, w_{34}, w_{45}, w_{26}, w_{36}, w_{46}, w_{56}, x, y, z),$$

and call

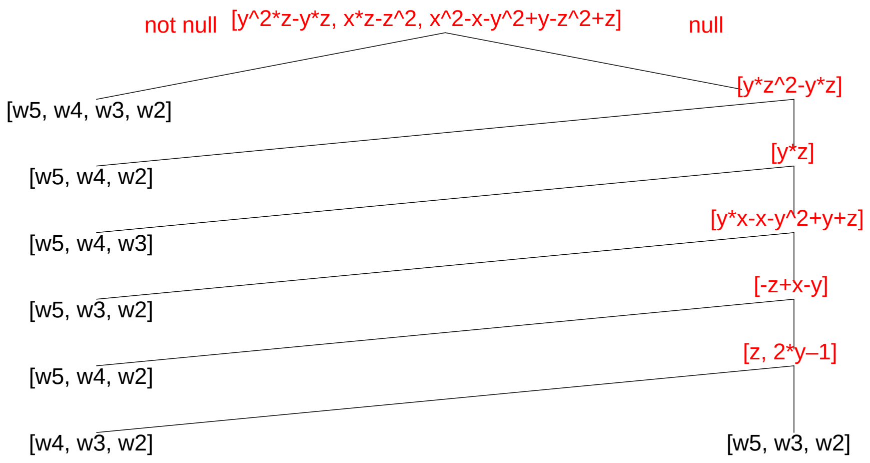
```
>  B:=gge(S,xpord);
```

The output `gge` basis can be particularly expressed as $B = B_1 \cup B_2$, where B_2 is the elimination ideal wrt the variables $w_{26} = w_2$, $w_{36} = w_3$, $w_{46} = w_4$, $w_{56} = w_5$ (i.e. the ideal containing all the polynomials involving only these variables) and B_1 expresses the remaining variables linearly in terms of w_2, w_3, w_4, w_5:

$$\begin{aligned}B_1 = [&w_{45}+zw_5, w_{34}+yw_4, w_{25}+w_5y, w_{23}+zw_3, w_{15}-2zw_5+w_5,\\ &w_{14}-2zw_5+w_5, w_{13}-2zw_5+w_5, w_{12}+2zw_5-w_5]\\ B_2 = [&-zw_5+w_5x-w_5y, -zw_5+w_4z, w_4x+yw_4-w_4-zw_5+w_5,\\ &w_3y-w_3+yw_4-2zw_5+w_5, xw_3-yw_4-zw_3,\\ &w_2z-w_2+zw_3+2zw_5-w_5, w_2y-w_2+w_5y+2zw_5-w_5,\\ &w_2x-w_2+zw_3+w_5y+2zw_5-w_5].\end{aligned}$$

So, the discussion is reduced to the system B_2. Calling now `dispgb` for B_2 the discussion becomes:

```
>  T:=dispgb(B2,plex(w2,w3,w4,w5),plex(x,y,z)):
>  tplot(T);
```



```
> finalcases(T);
```

$Case = [\,[1],\, [w_5,\, w_4,\, w_3,\, w_2],\, [\,],\, \{[y\,z\,(y-1),\, z\,(x-z),$
$\quad x^2 - x - y^2 + y - z^2 + z]\}, [w_5,\, w_4,\, w_3,\, w_2]\,]$
$Case = [\,[0,\, 1],\, [w_5,\, w_4,\, (z-1)\,w_2 + z\,w_3],\, [y-1,\, x-z],\, \{y,\, z-1,\, z\},$
$\quad [w_5,\, w_4,\, w_2]\,]$
$Case = [\,[0,\, 0,\, 1],\, [w_5,\, w_4,\, w_3],\, [z-1,\, y-1,\, x-1],\, \{y,\, z\},\, [w_5,\, w_4,\, w_3]\,]$
$Case = [\,[0,\, 0,\, 0,\, 1],\, [w_5,\, (y-1)\,w_3 + y\,w_4,\, w_2],\, [y\,z,\, z\,(x-z),$
$\quad x^2 - x - y^2 + y - z^2 + z],\, \{y\,x - x - y^2 + y + z\},\, [w_5,\, w_3,\, w_2]\,]$
$Case = [\,[0,\, 0,\, 0,\, 0,\, 1],\, [w_5,\, w_4,\, w_2],\, [y\,z,\, z\,(x-z),\, y\,x - x - y^2 + y + z,$
$\quad x^2 - x - y^2 + y - z^2 + z],\, \{-z + x - y\},\, [w_5,\, w_4,\, w_2]\,]$
$Case = [\,[0,\, 0,\, 0,\, 0,\, 0,\, 1],\, [w_4 - w_5,\, w_3 + (-1 + 2\,z)\,w_5,\, w_2 + (-2\,z + 1)\,w_5]\,]$
$[y\,z,\, -z + x - y],\, \{[2\,y - 1,\, z]\},\, [w_4,\, w_3,\, w_2]\,]$
$Case = [\,[0,\, 0,\, 0,\, 0,\, 0,\, 0],\, [w_5,\, w_3 - w_4,\, w_2],\, [z,\, 2\,y - 1,\, 2\,x - 1],\, \{\,\},$
$\quad [w_5,\, w_3,\, w_2]\,]$

The generic case corresponds to the trivial 0 solution for the stresses w_{ij} and is just the non interesting case. The discriminant is

$$\Delta_{[\,]} := [y\,z\,(y-1),\; z\,(x-z), x^2 - x - y^2 + y - z^2 + z]$$

and the system has no trivial solutions only when all the polynomials in $\Delta_{[\,]}$ are zero.

Table 1 summarizes the real points corresponding to the specifications of the null self-stress cases fulfilling all the solutions of the discriminant ideal $\Delta_{[\,]}$. Figure 3 shows the geometric locus of these points that corresponds to four straight lines lying on the hyperboloid $x^2 - x - y^2 + y - z^2 + z = 0$:

Case	Points	Exceptions
$[0,1]$	$\{(t,1,t) : t \in \mathbb{R}\}$	$(1,1,1),\ (0,1,0)$
$[0,0,1]$	$(1,1,1)$	
$[0,0,0,1]$	$\{(1-t,t,0) : t \in \mathbb{R}\}$	$(0,1,0),\ (\frac{1}{2},\frac{1}{2},0)$
$[0,0,0,0,1]$	$(0,1,0)$	
$[0,0,0,0,0,1]$	$\{(t,t,0) : t \in \mathbb{R}\}$ $\{(t,0,t) : t \in \mathbb{R}\}$	$(\frac{1}{2},\frac{1}{2},0)$
$[0,0,0,0,0,0]$	$(\frac{1}{2},\frac{1}{2},0)$	

Table 1: Non-null self-stress solutions

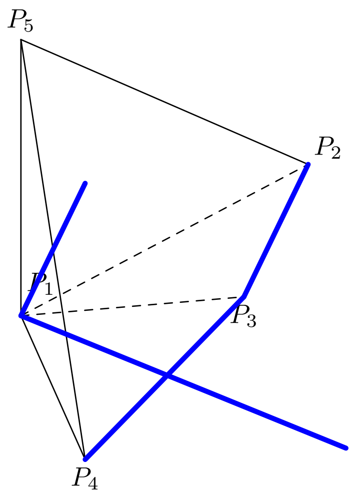


Figure 3: Graphic of the non-null self-stress solutions

8 Benchmarks

We tested the current implementation, Release 3.0 in *M*aple 8, of `dispgb` using a 2 GHz Pentium 4 at 512 MB to a set of several examples taken from the literature. Table 2 summarizes computing times of `dispgb` algorithm for a subset of these examples without rebuilding the tree, with 1 rebuild and totally rebuilt. The bases of the different examples are detailed below:

- S11. $[(d_4d_3R + r_2^2 - d_4d_3r_2^2 + d_4^2d_3^2 - d_4d_3^3 - d_4^3d_3 + d_4d_3 + Z - R)t^4$
  $+ (-2r_2d_4R + 2r_2d_4^3 + 2r_2d_4d_3^2 - 4r_2d_3d_4^2 + 2r_2^3d_4 + 2r_2d_4)t^3$
  $- (2r_2^2 - 2R + 4d_4^2r_2^2 + 4d_4^2 + 2Z - 2d_4^2d_3^2)t^2$
  $+ (-2r_2d_4R + 2r_2d_4d_3^2 + 2r_2d_4 + 2r_2d_4^3 + 4r_2d_3d_4^2 + 2r_2^3d_4)t$
  $+ r_2^2 + d_4^3d_3 - d_4d_3R + d_4d_3r_2^2 + Z - R - d_4d_3 + d_4^2d_3^2 + d_4d_3^3]$;
- S12. $[a - l_3c_3 - l_2c_1, b - l_3s_3 - l_2s_1, c_1^2 + s_1^2 - 1, c_3^2 + s_3^2 - 1]$;
- S14. $[t^3 - cut^2 - uv^2 - uw^2, t^3 - cvt^2 - vu^2 - vw^2, t^3 - cwt^2 - wu^2 - wv^2]$;
- S15. $[a + ds_1, b - dc_1, l_2c_2 + l_3c_3 - d, l_2s_2 + l_3s_3 - c, s_1^2 + c_1^2 - 1, s_2^2 + c_2^2 - 1,$
  $s_3^2 + c_3^2 - 1]$;

We tested several other problems and in some of them only partial results have been reached. We present two significant examples:

- S17. $[axt^2 + bytz - x(x^2 + cy^2 + dz^2), ayt^2 + bzxt - x(y^2 + cz^2 + dx^2),$
  $azt^2 + bxyt - x(z^2 + cx^2 + dy^2)]$
- S18. $[(3x^2 + 9v^2 - 3v - 3x)t_1^2t_2^2 + (3v^2 - 3v + 6vx - 3x + 3x^2)t_2^2$
  $+ (3v + 3v^2 + 3x^2 - 3x - 6vx)t_1^2 - 24v^2t_1t_2 + 9v^2 - 3x + 3x^2 + 3v,$
  $(3x^2 + 9v^2 - 3v - 3x)t_2^2t_3^2 + (3v + 3v^2 + 3x^2 - 3x - 6vx)t_2^2$
  $+ (3v^2 - 3v + 6vx - 3x + 3x^2)t_3^2 - 24v^2t_2t_3 + 9v^2 - 3x + 3x^2 + 3v,$
  $(3x^2 + 9v^2 - 3v - 3x)t_3^2t_1^2 + (3v^2 - 3v + 6vx - 3x + 3x^2)t_1^2$
  $+ (3v + 3v^2 + 3x^2 - 3x - 6vx)t_3^2 - 24v^2t_3t_1 + 9v^2 - 3x + 3x^2 + 3v]$

For S17 (GoTrZa00), `dispgb` gets bogged down after computing 35 terminal vertices in 1375 sec. It is unable to finish the whole discussion tree, and so no rebuilding using the discriminant ideal can be achieved. The label of the 35th vertex is $[1, 1, 0, 1, 0, 0]$, thus all vertices beginning with $[1, 0, \ldots]$ have been already determined (the tree is built up in pre-order beginning with the 0 vertices).

S18 corresponds to the benzene molecule studied in (EmMo99). The situation reached by `dispgb` is similar to S17, getting bogged down after 45 seconds when the 9th vertex labelled $[1, 1, 0, 0]$ has been computed.

Identification	CPU time rebuild=0	CPU time rebuild=1	CPU time complete
S10. Section 5 Two arms robot	6.8	8.7	19.8
S11. (Co04) Coste robot	9.6	13.9	62.7
S12. (Ry00) Rychlik robot	5.2	9.7	75.4
S14. (GoTrZa00) Hilbert functions	8.1	8.2	8.6
S15. (GoRe93; De99) Romin robot	12.4	24.1	371.2
S16. Section 7, (GuOr04) Tensegrity	12.8	29.1	63.2

Table 2: Benchmarks (in sec.)

As one can observe in Table 2, rebuilding the tree is time consuming. In many practical cases, the best discussion appears to be setting the option `rebuild=1` to the `dispgb` call, which rebuilds only the generic case. We would recommend to use in general this option as the quickest practical discussion.

9 Acknowledgements

We would like to thank Professors Pelegrí Viader and Julien Pfeifle for their 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.

[Co04] M. Coste. Classifying serial manipulators: Computer Algebra and geometric insight. Plenary talk. (Personal communication). *Proceedings* of EACA-2004, 323–323, 2004.

[CoLiSh] D. Cox, J. Little, D. O'Shea. Ideals, Varieties and Algorithms. Springer, New-York, 2nd edition, 1996.

[De99] S. Dellière. Triangularisation de systèmes constructibles. Application à l'évaluation dynamique. Thèse Doctorale, Université de Limoges. Limoges, 1995.

[Du95] D. Duval. Évaluation dynamique et clôture algébrique en Axiom. *Journal of Pure and Applied Algebra*, **99**:267–295, 1995.

[EmMo99] I. Z. Emiris, B. Mourrain. Computer Algebra Methods for Studying and Computing Molecular Conformations. *Algorithmica* **25**: 372-402, 1999.

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

[GoRe93] M.J. González-López, T. Recio. The ROMIN inverse geometric model and the dynamic evaluation method. In: Computer Algebra in Industry. Ed. A.M. Cohen, Wiley & Sons, 117–141, 1991.

[GoTrZa00] M.J. González-López, L. González-Vega, C. Traverso, A. Zanoni. Gröbner Bases Specialization through Hilbert Functions: The Homogeneous Case. *SIGSAM BULL*, Issue 131, **34**:1, 1-8, 2000.

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

[MaMo05a] M. Manubens, A. Montes. Improving DISPGB Algorithm Using the Discriminant Ideal. *Proceedings* of Algorithmic Algebra and Logic 2005, 159–166, 2005.

[MaMo05b] M. Manubens, A. Montes. Improving DISPGB Algorithm Using the Discriminant Ideal. To be published in *Jour. Symb. Comp.*

[Mo95] A. Montes. Solving the Load Flow Problem Using the 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.

[Ry00] M. Rychlik. Complexity and Applications of Parametric Algorithms of Computational Algebraic Geometry. In: Dynamics of Algorithms. Ed. R. del la Llave, L. Petzold, and J. Lorenz. IMA Volumes in Mathematics and its Applications, Springer-Verlag **118**:1–29, 2000.

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

[We02] V. Weispfenning. Canonical Comprehensive Gröbner Bases. *Proceedings* of ISSAC 2002. ACM-Press, 270–276, 2002.

[We03] V. Weispfenning. Canonical Comprehensive Gröbner Bases. *Jour. Symb. Comp.* **36**:669–683, 2003.