16.14 CGB: Computing Comprehensive Gröbner Bases

Authors: Andreas Dolzmann, Thomas Sturm, and Winfried Neun

16.14.1 Introduction

Consider the ideal basis $F = \{ax, x + y\}$. Treating a as a parameter, the calling sequence

```
torder({x,y},lex)$
groebner{a*x,x+y};

  {x,y}
```

yields $\{x, y\}$ as reduced Gröbner basis. This is, however, not correct under the specialization $a = 0$. The reduced Gröbner basis would then be $\{x+y\}$. Taking these results together, we obtain $C = \{x + y, ax, ay\}$, which is correct wrt. *all* specializations for a including zero specializations. We call this set C a *comprehensive Gröbner basis* (CGB).

The notion of a CGB and a corresponding algorithm has been introduced bei Weispfenning [**?**]. This algorithm works by performing case distinctions wrt. parametric coefficient polynomials in order to find out what the head monomials are under all possible specializations. It does thus not only determine a CGB, but even classifies the contained polynomials wrt. the specializations they are relevant for. If we keep the Gröbner bases for all cases separate and associate information on the respective specializations with them, we obtain a *Gröbner system*. For our example, the Gröbner system is the following;

$$\left[\begin{array}{c|c} a \neq 0 & \{x + y, ax, ay\} \\ a = 0 & \{x + y\} \end{array}\right].$$

A CGB is obtained as the union of the single Gröbner bases in a Gröbner system. It has also been shown that, on the other hand, a Gröbner system can easily be reconstructed from a given CGB [**?**].

The CGB package provides functions for computing both CGB's and Gröbner systems, and for turning Gröbner systems into CGB's.

16.14.2 Using the REDLOG Package

For managing the conditions occurring with the CGB computations, the CGB package uses the package REDLOG implementing first-order formulas, [**?**, **?**], which is also part of the REDUCE distribution.

16.14.3 Term Ordering Mode

The CGB package uses the settings made with the function `torder` of the GROEBNER package. This includes in particular the choice of the main variables. All variables not mentioned in the variable list argument of `torder` are parameters. The only term ordering modes recognized by CGB are `lex` and `revgradlex`.

16.14.4 CGB: Comprehensive Gröbner Basis

The function `cgb` expects a list F of expressions. It returns a CGB of F wrt. the current `torder` setting.

Example

```
torder({x,y},lex)$
cgb{a*x+y,x+b*y};

{x + b*y,a*x + y,(a*b - 1)*y}

ws;

{b*y + x,

 a*x + y,

 y*(a*b - 1)}
```

Note that the basis returned by the `cgb` call has not undergone the standard evaluation process: The returned polynomials are ordered wrt. the chosen term order. Reevaluation changes this as can be seen with the output of `ws`.

16.14.5 GSYS: Gröbner System

The function `gsys` follows the same calling conventions as `cgb`. It returns the complete Gröbner system represented as a nested list

$$\big\{\big\{c_1,\{g_{11},\ldots,g_{1n_1}\}\big\},\ldots,\big\{c_m,\{g_{m1},\ldots,g_{1n_m}\}\big\}\big\}.$$

The c_i are conditions in the parameters represented as quantifier-free REDLOG formulas. Each choice of parameters will obey at least one of the c_i. Whenever a

choice of parameters obeys some c_i, the corresponding $\{g_{i1}, \ldots, g_{in_i}\}$ is a Gröbner basis for this choice.

Example

```
torder({x,y},lex)$
gsys {a*x+y,x+b*y};

{{a*b - 1 <> 0 and a <> 0,

  {a*x + y,x + b*y,(a*b - 1)*y}},

 {a <> 0 and a*b - 1 = 0,

  {a*x + y,x + b*y}},

 {a = 0,{a*x + y,x + b*y}}}
```

As with the function `cgb`, the contained polynomials remain unevaluated.

Computing a Gröbner system is not harder than computing a CGB. In fact, `cgb` also computes a Gröbner system and then turns it into a CGB.

Switch CGBGEN: Only the Generic Case

If the switch `cgbgen` is turned on, both `gsys` and `cgb` will assume all parametric coefficients to be non-zero ignoring the other cases. For `cgb` this means that the result equals—up to auto-reduction—that of `groebner`. A call to `gsys` will return this result as a single case including the assumptions made during the computation:

Example

```
torder({x,y},lex)$
on cgbgen;
gsys{a*x+y,x+b*y};

{{a*b - 1 <> 0 and a <> 0,

  {a*x + y,x + b*y,(a*b - 1)*y}}}

off cgbgen;
```

16.14.6 GSYS2CGB: Gröbner System to CGB

The call `gsys2cgb` turns a given Gröbner system into a CGB by constructing the union of the Gröbner bases of the single cases.

Example

```
torder({x,y},lex)$
gsys{a*x+y,x+b*y}$
gsys2cgb ws;

{x + b*y,a*x + y,(a*b - 1)*y}
```

16.14.7 Switch CGBREAL: Computing over the Real Numbers

All computations considered so far have taken place over the complex numbers, more precisely, over algebraically closed fields. Over the real numbers, certain branches of the CGB computation can become inconsitent though they are not inconsistent over the complex numbers. Consider, e.g., a condition $a^2 + 1 = 0$.

When turning on the switch `cgbreal`, all simplifications of conditions are performed over the real numbers. The methods used for this are described in [**?**].

Example

```
torder({x,y},lex)$
off cgbreal;
gsys {a*x+y,x-a*y};

   2
{{a  + 1 <> 0 and a <> 0,

                      2
  {a*x + y,x - a*y,(a  + 1)*y}},

              2
 {a <> 0 and a  + 1 = 0,{a*x + y,x - a*y}},

 {a = 0,{a*x + y,x - a*y}}}

on cgbreal;
gsys({a*x+y,x-a*y});
```

```
{{a <> 0,

                          2
  {a*x + y,x - a*y,(a  + 1)*y}},

 {a = 0,{a*x + y,x - a*y}}}
```

16.14.8 Switches

cgbreal Compute over the real numbers. See Section 16.14.7 for details.

cgbgs Gröbner simplification of the condition. The switch `cgbgs` can be turned on for applying advanced algebraic simplification techniques to the conditions. This will, in general, slow down the computation, but lead to a simpler Gröbner system.

cgbstat Statistics of the CGB run. The switch `cgbstat` toggles the creation and output of statistical information on the CGB run. The statistical information is printed at the end of the run.

cgbfullred Full reduction. By default, the CGB functions perform full reductions in contrast to pure top reductions. By turning off the switch `cgbfullred`, reduction can be restricted to top reductions.