Synthesis of a Groebner Bases Algorithm by Lazy Thinking

Adrian Craciun,
Research Institute for Symbolic Computation − Linz, Austria
Institute e−Austria − Timisoara, Romania
acraciun@risc.uni−linz.ac.at, acraciun@ieat.ro

1 TheoremaPrivateDirectory!!!! − Instructions

2 System Initializations

3 The Problem of Groebner Bases

3.1 The Specification of the Groebner Bases Problem

In the following, consider F be a finite set of polynomials. The problem of Groebner bases has to do with finding an algorithm, GB, that satisfies the following specification (correctness theorem):

```
Theorem["Groebner bases specification", any[F], with[is-finite[F]]
    is-finite-Groebner-basis[F, GB[F]]]
```

where

$$\text{Definition}\Big[\text{"finite Groebner basis"},\ \text{any}[F, G],\ \text{with}[\text{is-finite}[F]],\quad \text{is-finite-Groebner-basis}[F, G] \Leftrightarrow \bigwedge \begin{cases} \text{is-finite}[G] \\ \text{is-Groebner-basis}[G] \\ \text{ideal}[F] = \text{ideal}[G] \end{cases}\Big]$$

3.2 Subproblem:: Second part of the Groebner Bases Specification

$$\text{Theorem}[\text{"Groebner Bases specification: is Groebner Base"},\ \text{any}[F],\ \text{is-Church-Rosser}[\longrightarrow_{CPC[F]}]]$$

3.3 Knowledge Relevant to the Subproblem at Hand [Including Algorithm Scheme]

3.3.1 General Properties of Polynomials

3.3.2 Properties Involving Polynomial Reduction

Proposition["polynomial reductions are noetherian", any[G],
is-Noetherian[$\longrightarrow_G$]]

Proposition["one reduction step:pp", any[is-pp[p], G, f],

$$(p \longrightarrow_G f) \Rightarrow \underset{g}{\exists} \left(\bigwedge \begin{cases} g \in G \\ \mathrm{lp}[g] \mid p \\ f = \mathrm{rd}[p, g] \end{cases} \right)]$$

Proposition["totally reduces modulo a set", any[f, g, G],
$(f \twoheadrightarrow_G g) \Leftrightarrow (g = \mathrm{trd}[f, G])$]

Proposition["common successor", any[f1, f2, G],

$$f1 \downarrow_G f2 \Leftrightarrow \underset{g}{\exists} \left(\bigwedge \begin{cases} f1 \twoheadrightarrow_G g \\ f2 \twoheadrightarrow_G g \end{cases} \right)]$$

Proposition["Church Rosser: Newman:pp", any[G], with[is-Noetherian[$\longrightarrow_G$]],

$$\text{is-Church-Rosser}[\longrightarrow_G] \Leftrightarrow \underset{p}{\forall} \underset{f1, f2}{\forall} \left(\left(\bigwedge \begin{cases} \text{is-pp}[p] \\ p \longrightarrow_G f1 \\ p \longrightarrow_G f2 \end{cases} \right) \Rightarrow f1 \downarrow_G f2 \right)]$$

3.3.3 Algorithm Scheme: CPC

CPC[F] = CPC[F, pairs[F]]
CPC[F, ⟨⟩] = F

CPC[F, ⟨⟨g1, g2⟩, $\overline{p}$⟩] =
where[f = **lc**[g1, g2], h1 = trd[rd[f, g1], F], h2 = trd[rd[f, g2], F],

$$\begin{cases} \mathrm{CPC}[F, \langle \overline{p} \rangle] & \Leftarrow \; h1 = h2 \\ \mathrm{CPC}\left[F \frown \mathbf{df}[h1, h2], \langle \overline{p} \rangle \asymp \left\langle \langle F_k, \mathbf{df}[h1, h2] \rangle \underset{k=1,\ldots,|F|}{\mid} \right\rangle\right] & \Leftarrow \; \text{otherwise} \end{cases}]$$

3.3.4 Algorithm Scheme: CPC [processed]

```
Proposition["processed CPC scheme: variant", any[g1, g2, F],
  ((g1 ∈ CPC[F] ∧ g2 ∈ CPC[F]) ⇒ ⋁
        { trd[rd[lc[g1, g2], g1], CPC[F]] = trd[rd[lc[g1, g2], g2], CPC[F]]
        { df[trd[rd[lc[g1, g2], g1], CPC[F]], trd[rd[lc[g1, g2], g2], CPC[F]]] ∈ CPC[F]  )
]
```

3.3.5 Knowledge on Diamonds

3.3.6 Reduction in Steps

3.3.7 Collecting the Knowledge: Theories

4 Lazy Thinking Semiautomated

4.1 Lazy Thinking Semiautomated: Step 1

4.1.1 The Proof Attempt

```
Prove[Theorem["Groebner Bases specification: is Groebner Base"], using → Theory["pre GB1"],
      by → BasicProver,
      ProverOptions →
        {GRWTarget → {"goal", "kb"}, DisableMatchExist → True, UseSkolemFunctions → False,
          RWInsideQuantifiers → True, DeleteGroundKBFacts → False, UseEqualitiesFirst → False,
          RWExistentialGoal → True(*<--- Very Important Option to be set .... wont work without it.... *),
          AllowIntroduceQuantifiers → True, ModusPonensUnknownSymbols → {lc, df}}] // Last // Timing
```

4.1.2 Generate Conjecture

```
FailureAnalyser[$TmaProofObject]
```

$\{\{$•lf[23, trd[rd[p_0, g_0], CPC[F_0]] = trd[rd[p_0, g_1], CPC[F_0]], •finfo[]],
•asml[•lf[12.1, $g_1 \in$ CPC[F_0], •finfo[]], •lf[12.2, lp[g_1] | p_0, •finfo[]], •lf[12.3, $f2_0$ = rd[p_0, g_1], •finfo[]],
•lf[13.1, trd[rd[lc[g_0, g_1], g_0], CPC[F_0]] = trd[rd[lc[g_0, g_1], g_1], CPC[F_0]], •finfo[]],
•lf[16, $\forall_{a,q}$ (trd[rd[$a * q *$ lc[g_0, g_1], g_0], CPC[F_0]] = trd[rd[$a * q *$ lc[g_0, g_1], g_1], CPC[F_0]]), •finfo[]],
•lf[17, $\forall_{a,q}$ ($a * q *$ trd[rd[lc[g_0, g_1], g_0], CPC[F_0]] = $a * q *$ trd[rd[lc[g_0, g_1], g_1], CPC[F_0]]), •finfo[]],
•lf[2.1, is-Noetherian[$\longrightarrow_{\mathrm{CPC}[F_0]}$], •finfo[]], •lf[24, $\forall_{a,q}$ ($a * q *$ rd[p_0, g_1] = $a * q *$ rd[p_0, g_1]), •finfo[]],
•lf[25, $\forall_{a,q}$ ($a * q * f2_0$ = $a * q *$ rd[p_0, g_1]), •finfo[]], •lf[4.1, is-pp[p_0], •finfo[]],
•lf[4.2, $p_0 \longrightarrow_{\mathrm{CPC}[F_0]} f1_0$, •finfo[]], •lf[4.3, $p_0 \longrightarrow_{\mathrm{CPC}[F_0]} f2_0$, •finfo[]],
•lf[8.1, $g_0 \in$ CPC[F_0], •finfo[]], •lf[8.2, lp[g_0] | p_0, •finfo[]],
•lf[8.3, $f1_0$ = rd[p_0, g_0], •finfo[]], •lf[9, $\forall_{a,q}$ ($a * q * f1_0$ = $a * q *$ rd[p_0, g_0]), •finfo[]]]$\}\}$

```
GenerateConjectures[$TmaProofObject, {}, {lc, df}, {}]
```

•lma["conjecture$293", •range[], True, •flist[
•lf["conjecture$293.1", $\forall_{g5,g6,p3}$ ((lp[$g5$] | $p3$) $\wedge$ (lp[$g6$] | $p3$) $\wedge$ is-pp[$p3$] $\Rightarrow \exists_{a,q}$ ($p3 = a * q *$ lc[$g5$, $g6$]))]]]

4.1.3 Generate Conjecture No Quantification

4.2 Lazy Thinking Semiautomated: Step 2

4.3 Lazy Thinking Semiautomated: Step 3

5 Lazy Thinking Automated