Formal Mathematics:

A Key to the Future

Bruno Buchberger
RISC, Austria

Talk at "Engineering and Life Sciences"
June 26-30, 2006, Avignon, France

Dedicated to the 60th Birthday of Gautam Dasgupta

Purpose

- Give the flavor of a certain area in algorithmic mathematics:

  "formal math", "automated reasoning", "intellectics", "mathematical knowledge management", "computer-supported mathematical theory exploration", ...

- Does this have applications for life sciences?

Application to Life Sciences

- ????

- A recent project with "Genomica" company: imitate the process of how biologists guess the mapping between the presence of certain genes and biologic behavior. (⟶ "Algorithm synthesis")

- Processes in life sciences cannot be simulated without simulating the "evolution" of the processes.

  Example: simulation of a cell. (⟶ "Self-reference" is an important issue.)

4 of 46

Conference Announcement: "Algebraic Biology 2007", July 2-4, 2007

www.risc.uni-linz.ac.at/about/conferences/ab2007/

We need biologists who present problems !

Symbolic methods tutorial week before AB 2007 !

5 of 46

Focus of This Talk

- Formal mathematics: Why a key? Answer: Increase the efficiency of the mathematical research process.
- Mathematical research process: invention and verification.
- In this talk only one example: automated invention of a (non-trivial) algorithm and automated proof, namely algorithm for constructing Groebner bases.

6 of 46

Contents of the Talk

Current Math Systems

Groebner Bases

Groebner Bases Applied for Automated Reasoning

Automated Reasoning About Groebner Bases

8 of 46

All Current Algorithmics (Numerics, Symbolics,...) is Available in Systems

- Systems like *Mathematica*, Maple, Derive, Mathlab, ... FORM, Singular, Cocoa, ...
- An enormous potential for science (physics, ...) and engineering.
- Help! and additional Packages
- ⟶ The other math talks at this conference

9 of 46

Remark:

There is lots of new and deep mathematics behind the (numeric, discrete, graphic, algebraic, and symbolic) algorithms of the current math systems.

In this talk only one example: Gröbner bases theory:

- Why are Gröbner bases important? (Dozens of fundamental problems in pure and applied math can be reduced to Gröbner bases constructions! Examples: non-linear equation solving, diophantine equs with poly coefficients, presentations of polys as polys of polys, decomposition of varieties, canonical simplification modulo poly relations, ...)
- What are Gröbner bases?
- How can Gröbner bases be computed?

10 of 46

Current Math Systems

Groebner Bases

Groebner Bases Applied for Automated Reasoning

Automated Reasoning About Groebner Bases

11 of 46

Example of a Groebner Basis (BB 1965, ...)

B.B. An Algorithmic Criterion for the Solvability of Systems of Algebraic Equations, aequationes mathematicae, 1970. (English translation in: B.B., F.Winkler. Gröbner Bases: Theory and Applications. Cambridge University Press, 1998, pp. 540-560.

A system of polynomials (not a Groebner basis):

```
f1 = x y - 2 y z - z;
f2 = y^2 - x^2 z + x z;
f3 = z^2 - y^2 x + x;

F = {f1, f2, f3};
```

A ("the") corresponding Groebner basis:

```
G = GroebnerBasis[F]
```

$$\{-z - 4 z^3 + 17 z^4 - 3 z^5 + 45 z^6 - 60 z^7 + 29 z^8 - 124 z^9 + 48 z^{10} - 64 z^{11} + 64 z^{12},$$
$$-22001 z + 14361 y z + 16681 z^2 + 26380 z^3 + 226657 z^4 + 11085 z^5 - 90346 z^6 - 472018 z^7 - 520424 z^8 - 139296 z^9 - 150784 z^{10} + 490368 z^{11},$$
$$43083 y^2 - 11821 z + 267025 z^2 - 583085 z^3 + 663460 z^4 - 2288350 z^5 + 2466820 z^6 - 3008257 z^7 + 4611948 z^8 - 2592304 z^9 + 2672704 z^{10} - 1686848 z^{11},$$
$$43083 x - 118717 z + 69484 z^2 + 402334 z^3 + 409939 z^4 + 1202033 z^5 - 2475608 z^6 + 354746 z^7 - 6049080 z^8 + 2269472 z^9 - 3106688 z^{10} + 3442816 z^{11}\}$$

What important property of Groebner bases can we observe here?

```
zsol = NSolve[G[[1]] == 0, z]
```

$$\{\{z \to -0.331304 - 0.586934\, i\}, \{z \to -0.331304 + 0.586934\, i\}, \{z \to -0.296413 - 0.705329\, i\}, \{z \to -0.296413 + 0.705329\, i\}, \{z \to -0.163124 - 0.37694\, i\}, \{z \to -0.163124 + 0.37694\, i\}, \{z \to 0.\}, \{z \to 0.0248919 - 0.89178\, i\}, \{z \to 0.0248919 + 0.89178\, i\}, \{z \to 0.468852\}, \{z \to 0.670231\}, \{z \to 1.39282\}\}$$

```
Gsubnum = G /. zsol[[1]]
```

$$\{1.11022 \times 10^{-15} + 5.55112 \times 10^{-16}\, i, (-523.519 - 4967.65\, i) - (4757.86 + 8428.97\, i)\, y, (-7846.9 - 8372.06\, i) + 43083\, y^2, (-16311.7 + 16611.\, i) + 43083\, x\}$$

```
ysol = NSolve[ Gsubnum[[2]] == 0, y]
```

$$\{\{y \to -0.473535 - 0.205184\, i\}\}$$

Theorem (Roider, Kalkbrener et al. 1990): It suffices to consider the poly in y with lowest degree.

```
xsol = NSolve[ Gsubnum[[4]] == 0, x]
```

$$\{\{x \to 0.378611 - 0.385558\, i\}\}$$

```
F /. zsol[[1]] /. ysol[[1]] /. xsol[[1]]
```

$\{-3.21965\times10^{-15}-3.45557\times10^{-15}\,i,\ 4.02456\times10^{-15}-8.04912\times10^{-16}\,i,\ 5.07927\times10^{-15}+1.83187\times10^{-15}\,i\}$

12 of 46

Another Example of Application of Groebner Bases: Invariant Theory

A Question: Can

```
h = x1^7 x2 - x1 x2^7
```

$x_1^7 x_2 - x_1 x_2^7$

be expressed as a polynomial in

```
F = {x1^2 + x2^2, x1^2 x2^2, x1^3 x2 - x1 x2^3}
```

$\{x_1^2+x_2^2,\ x_1^2 x_2^2,\ x_1^3 x_2 - x_1 x_2^3\}$

?

Note: These polynomials are fundamental invariants for the group $\mathbb{Z}_4$.

13 of 46

Reduction to Groebner Bases Computation

```
{time, GB} = GroebnerBasis[
   {-i1 + x1^2 + x2^2, -i2 + x1^2 x2^2, -i3 + x1^3 x2 - x1 x2^3}, {x2, x1, i3, i2, i1}] // Timing
```

$\{0.\ \text{Second},$
$\{-i_1^2 i_2 + 4 i_2^2 + i_3^2,\ i_2 - i_1 x_1^2 + x_1^4,\ -i_1^2 i_3 x_1 + 2 i_2 i_3 x_1 + i_1 i_3 x_1^3 - i_1^2 i_2 x_2 + 4 i_2^2 x_2,$
$i_1^2 x_1 - 2 i_2 x_1 - i_1 x_1^3 + i_3 x_2,\ -i_1 i_3 + 2 i_3 x_1^2 - i_1^2 x_1 x_2 + 4 i_2 x_1 x_2,$
$-i_3 x_1 - 2 i_2 x_2 + i_1 x_1^2 x_2,\ -i_3 - i_1 x_1 x_2 + 2 x_1^3 x_2,\ -i_1 + x_1^2 + x_2^2\}\}$

```
PolynomialReduce[x1^7 x2 - x1 x2^7, GB,
  {x2, x1, i3, i2, i1}, MonomialOrder → Lexicographic]
```

```
i1^2 i3 - i2 i3
```

Theorem (Sweedler, Sturmfels et al. 1988): *h* can be represented in terms of *I* iff remainder of *h* w.r.t. "Groebner basis of *I* with slack variables" is a polynomial in the slack variables (which gives the representation).

```
i1^2 i3 - i2 i3 /. {i1 → x1^2 + x2^2, i2 → x1^2 x2^2, i3 → x1^3 x2 - x1 x2^3} // Expand
```

$x_1^7 x_2 - x_1 x_2^7$

```
R = PolynomialReduce[x1^6 x2 - x1 x2^6, GB,
  {x2, x1, i3, i2, i1}, MonomialOrder → Lexicographic]
```

$$-i_1^3 x_1 + 2\, i_1 i_2 x_1 + \frac{1}{2} i_1 i_3 x_1 + i_1^2 x_1^3 - i_2 x_1^3 + \frac{1}{2} i_3 x_1^3 + \frac{1}{2} i_1 i_2 x_2$$

$x_1^6 x_2 - x_1 x_2^6$ can not be expressed by the fundamental invariants in I.

14 of 46

Another Example of the Application of Groebner Bases: Computation on Differential Operators

See talk by M. Rosenkranz.

15 of 46

More Applications

> 1000 papers and > 10 textbooks on Groebner bases.

Applications in: Algebraic Geometry, Cryptography, Coding Theory, Integer Optimization, Algebraic Combinatorics, Combinatorial and Special Function Identities, Symbolic Summation, Symbolic Analysis (in particular, Differential Equations), Geometry Theorema Proving, Control Theory, etc.

Gröbner Bases 2006 Special Semester at RICAM and RISC (Feb - June 2006): 10 Proceedings volumes will be issued.

16 of 46

The Problem of *Constructing* Gröbner Bases

Definition: F is a Gröbner basis iff

polynomial reduction ("division") w.r.t. F is unique.

Problem: Given *F*, find *G* such that *G* is a Gröbner basis

and F and G generate the same set of linear combinations.

Why is this problem fundamental?

Many problems that are difficult for general F are easy for Gröbner bases G.

Hence, many difficult problems can be solved by (easy) reductions to the problem of constructing Gröbner bases, for wich these problems are easy.

The "Main Theorem" of Gröbner Bases Theory (BB 1965):

F is a Gröbner basis $\Longleftrightarrow \underset{f_1, f_2 \in F}{\forall}$ reduction[*F*, S–polynomial[f_1, f_2]] = 0.

```
S-polynomial[-2 y + x y, -x^2 + y^2] = y (-2 y + x y) - x (-x^2 + y^2)
```

$x^3 - 2 y^2$

Main intuition: least common multiple of the leading power products play the important role.

Proof: Nontrivial. Combinatorial.

The power of the Gröbner bases method is contained in the invention of the notion of S-polynomial and the theorem, and the proof of this theorem.

An Algorithm for *Constructing* Gröbner Bases (BB 1965)

Recall the main theorem:

$$F \text{ is a Gröbner basis} \iff \underset{f_1, f_2 \in F}{\forall} \text{reduction}[F, \text{S–polynomial}[f_1, f_2]] = 0.$$

Based on the main theorem, the problem can be solved by the following algorithm:

Start with G:= F.

For any pair of polynomials $f_1, f_2 \in G$:

h := remainder[G, S–polynomial[f_1, f_2]]

If $h = 0$, consider the next pair.

If $h \neq 0$, add h to G and iterate.

Termination of the Algorithm

Termination: by Dickson's Lemma (Dickson 1913, BB 1970).

Current Math Systems

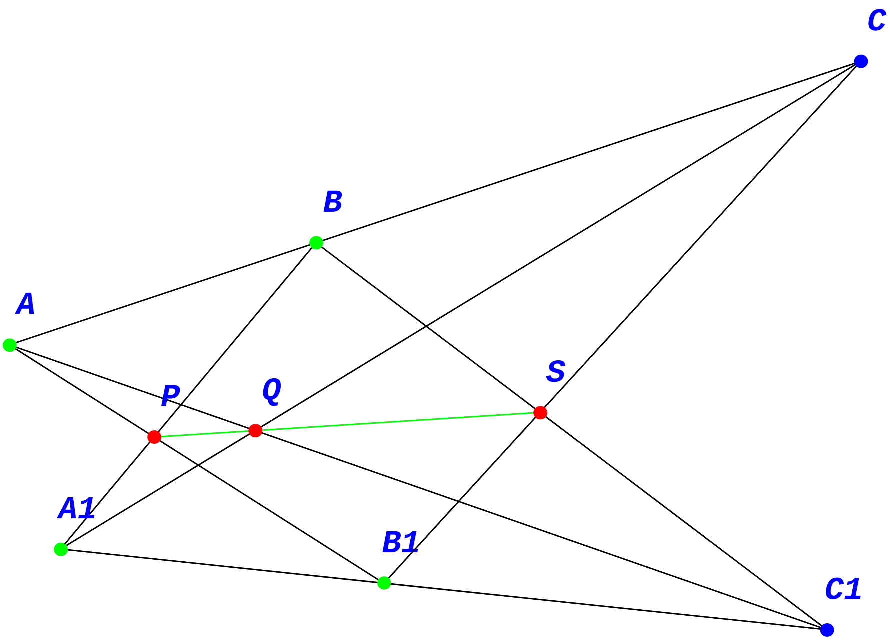
Groebner Bases

Groebner Bases Applied for Automated Reasoning

Automated Reasoning About Groebner Bases

Example: Pappus Theorem

- What does the theorem say geometrically?



- Textbook formulation:

Let A,B, C and A1,B1, C1 be on two lines and P = AB1 ⋂ A1B, Q = AC1 ⋂ A1C, S = BC1 ⋂ B1C. Then P, Q, and S are collinear.

- Input to the system:

```
Proposition["Pappus", any[A, B, A1, B1, C, C1, P, Q, S],
 point[A, B, A1, B1] ∧ pon[C, line[A, B]] ∧ pon[C1, line[A1, B1]] ∧
   inter[P, line[A, B1], line[A1, B]] ∧ inter[Q, line[A, C1], line[A1, C]] ∧
   inter[S, line[B, C1], line[B1, C]] ⇒ collinear[P, Q, S]]
```

- Input to the system:

```
Prove[Proposition["Pappus"], by → GeometryProver,
 ProverOptions → {Method -> "GroebnerProver", Refutation → True}]
```

```
- ProofObject -
```

- Notebook generated automatically by the proving algorithm based on Groebner basis algorithm:

Prove:

(Proposition (Pappus))

$$\forall_{A,B,A1,B1,C,C1,P,Q,S} (\text{point}[A, B, A1, B1] \wedge \text{pon}[C, \text{line}[A, B]] \wedge \text{pon}[C1, \text{line}[A1, B1]] \wedge \text{inter}[P, \text{line}[A, B1], \text{line}[A1, B]] \wedge \text{inter}[Q, \text{line}[A, C1], \text{line}[A1, C]] \wedge \text{inter}[S, \text{line}[B, C1], \text{line}[B1, C]] \Rightarrow \text{collinear}[P, Q, S])$$

with no assumptions.

To prove the above statement we shall use the Gröbner basis method. First we have to transform the problem into algebraic form.

Algebraic Form:

To transform the geometric problem into algebraic form we have to chose first an orthogonal coordinate system.

Let's have the origin in point A, and points $\{B, C\}$ on the two axes.

Using this coordinate system we have the following points:

$$\{\{A, 0, 0\}, \{B, 0, u_1\}, \{A1, u_2, u_3\}, \{B1, u_4, u_5\}, \{C, 0, u_6\}, \{C1, u_7, x_1\}, \{P, x_2, x_3\}, \{Q, x_4, x_5\}, \{S, x_6, x_7\}\}$$

The algebraic form of the assertion is:

(1)

$$\forall_{x_1,x_2,x_3,x_4,x_5,x_6,x_7} (u_3 u_4 + -u_2 u_5 + -u_3 u_7 + u_5 u_7 + u_2 x_1 + -u_4 x_1 == 0 \wedge u_5 x_2 + -u_4 x_3 == 0 \wedge -u_1 u_2 + u_1 x_2 + -u_3 x_2 + u_2 x_3 == 0 \wedge x_1 x_4 + -u_7 x_5 == 0 \wedge -u_2 u_6 + -u_3 x_4 + u_6 x_4 + u_2 x_5 == 0 \wedge u_1 u_7 + -u_1 x_6 + x_1 x_6 + -u_7 x_7 == 0 \wedge -u_4 u_6 + -u_5 x_6 + u_6 x_6 + u_4 x_7 == 0 \Rightarrow x_3 x_4 + -x_2 x_5 + -x_3 x_6 + x_5 x_6 + x_2 x_7 + -x_4 x_7 == 0)$$

This problem is equivalent to:

(2)

$$\neg \left(\underset{x_1,x_2,x_3,x_4,x_5,x_6,x_7}{\exists} (u_3\, u_4 + -u_2\, u_5 + -u_3\, u_7 + u_5\, u_7 + u_2\, x_1 + -u_4\, x_1 == 0 \wedge u_5\, x_2 + -u_4\, x_3 == 0 \wedge -u_1\, u_2 + u_1\, x_2 + -u_3\, x_2 + u_2\, x_3 == 0 \wedge x_1\, x_4 + -u_7\, x_5 == 0 \wedge -u_2\, u_6 + -u_3\, x_4 + u_6\, x_4 + u_2\, x_5 == 0 \wedge u_1\, u_7 + -u_1\, x_6 + x_1\, x_6 + -u_7\, x_7 == 0 \wedge -u_4\, u_6 + -u_5\, x_6 + u_6\, x_6 + u_4\, x_7 == 0 \wedge x_3\, x_4 + -x_2\, x_5 + -x_3\, x_6 + x_5\, x_6 + x_2\, x_7 + -x_4\, x_7 \neq 0)\right)$$

To remove the last inequality, we use the Rabinowitsch trick: Let v_0 be a new variable. Then the problem becomes:

(3)

$$\neg \left(\underset{x_1,x_2,x_3,x_4,x_5,x_6,x_7,v_0}{\exists} (u_3\, u_4 + -u_2\, u_5 + -u_3\, u_7 + u_5\, u_7 + u_2\, x_1 + -u_4\, x_1 == 0 \wedge u_5\, x_2 + -u_4\, x_3 == 0 \wedge -u_1\, u_2 + u_1\, x_2 + -u_3\, x_2 + u_2\, x_3 == 0 \wedge x_1\, x_4 + -u_7\, x_5 == 0 \wedge -u_2\, u_6 + -u_3\, x_4 + u_6\, x_4 + u_2\, x_5 == 0 \wedge u_1\, u_7 + -u_1\, x_6 + x_1\, x_6 + -u_7\, x_7 == 0 \wedge -u_4\, u_6 + -u_5\, x_6 + u_6\, x_6 + u_4\, x_7 == 0 \wedge 1 + -v_0\, (x_3\, x_4 + -x_2\, x_5 + -x_3\, x_6 + x_5\, x_6 + x_2\, x_7 + -x_4\, x_7) == 0)\right)$$

This statement is true iff the corresponding Gröbner basis is $\{1\}$.

The Gröbner bases is $\{1\}$.

Hence, the statement and the original assertion is true.

Statistics:

Time needed to compute the Gröbner bases: 0.42 Seconds.

Current Math Systems

Groebner Bases

Groebner Bases Applied for Automated Reasoning

Automated Reasoning About Groebner Bases

Predicate Logic Proving: Automated Proofs of Theorems in Analysis (The "PCS" Prover: BB 2001)

Initialize Theorema

Example

```
Definition["limit:", any[f, a],
    limit[f, a] ⟺ ∀_{ϵ, ϵ>0} ∃_N ∀_{n, n≥N} |f[n] − a| < ϵ]
```

```
Proposition["limit of sum", any[f, a, g, b],
    (limit[f, a] ∧ limit[g, b])  ⇒  limit[f + g, a + b]]
```

```
Definition["+:", any[f, g, x],
    (f + g)[x] = f[x] + g[x]]
```

```
Lemma["|+|", any[x, y, a, b, δ, ϵ],
    (|(x + y) − (a + b)| < (δ + ϵ))  ⟸  (|x − a| < δ ∧ |y − b| < ϵ)]
```

```
Lemma["max", any[m, M1, M2],
    m ≥ max[M1, M2]  ⇒  (m ≥ M1 ∧ m ≥ M2)]
```

```
Theory["limit",
    Definition["limit:"]
    Definition["+:"]
    Lemma["|+|"]
    Lemma["max"]                ]
```

```
Prove[Proposition["limit of sum"], using → Theory["limit"], by → PCS]
```

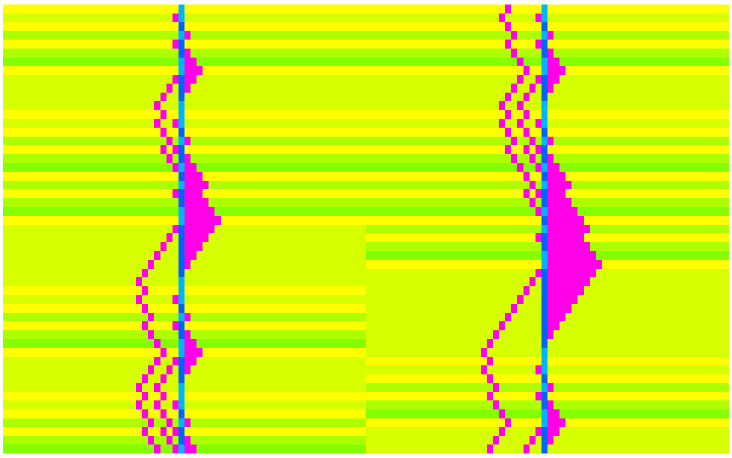
```
- ProofObject -
```

Proof contains interesting algorithmic and didactic information!

Induction Prover

Example: Inductive Proofs about Behavior of Turing Machines (in a Project with J. Hertel 2001)

```
Show-Raster[Compute[TM⋱[TM0, ic,
    0++++++ * (0++++++ * 0++++++)],
  using → ⟨Definition["TM0:"], Theory["⊢TM:"]⟩], 1, 33, 50]
```



- Graphics -

Prove:

(Proposition (TM0 left run)) $\underset{k,n,l}{\forall}\left(\mathrm{TM}{\hookleftarrow}[\mathrm{TM0}, \langle 2, \langle \square[l], 1, \blacksquare[\square[1, n+k]]\rangle\rangle, k] =, \langle 2, \langle \square[l, \square[0, k]], 1, \blacksquare[\square[1, n]]\rangle\rangle\right)$

under the assumptions:

(Definition (TM0:): 1) $\underset{l,r}{\forall}\left(\mathrm{rc}[\mathrm{TM0}, \langle 1, \langle l, 0, r\rangle\rangle] := \langle 1, \mathrm{R}, 2\rangle\right)$,

(Definition (TM0:): 2) $\underset{l,r}{\forall}\left(\mathrm{rc}[\mathrm{TM0}, \langle 1, \langle l, 1, r\rangle\rangle] := \langle 1, \mathrm{L}, 3\rangle\right)$,

(Definition (TM0:): 3) $\underset{l,r}{\forall}\left(\mathrm{rc}[\mathrm{TM0}, \langle 2, \langle l, 0, r\rangle\rangle] := \langle 0, \mathrm{L}, 1\rangle\right)$,

(Definition (TM0:): 4) $\underset{l,r}{\forall}\left(\mathrm{rc}[\mathrm{TM0}, \langle 2, \langle l, 1, r\rangle\rangle] := \langle 0, \mathrm{R}, 2\rangle\right)$,

(Definition (TM0:): 5) $\underset{l,r}{\forall}\left(\mathrm{rc}[\mathrm{TM0}, \langle 3, \langle l, 0, r\rangle\rangle] := \langle 1, \mathrm{R}, 1\rangle\right)$,

(Definition (TM0:): 6) $\underset{l,r}{\forall}\left(\mathrm{rc}[\mathrm{TM0}, \langle 3, \langle l, 1, r\rangle\rangle] := \langle 1, \mathrm{L}, 4\rangle\right)$, (Definition (TM0:): 7) $\underset{l,r}{\forall}$ (rc[TM0, ⟨4, ⟨l, 0, r⟩⟩] := ⟨1, L, 1⟩),

(Definition (tu⊢)) $\underset{x,\bar{x}}{\forall}$ (x ⌣ ⟨x̄⟩ := ⟨x, x̄⟩),

(Proposition (nu⊢): 1) $\underset{x}{\forall}$ ($x + 0 := x$),

(Proposition (nu⊢): 2) $\underset{x,y}{\forall}$ ($x + y^+ := x^+ + y$),

(Proposition (nu⊢): 3) $\underset{y}{\forall}$ ($0 + y := y$),

(Definition (ta⊢): 1) $\underset{s,\bar{t},\bar{u}}{\forall}$ (□[s, □[t̄], ū] := □[s, t̄, ū]),

(Definition (ta⊢): 2) $\underset{s,\bar{t},\bar{u}}{\forall}$ (■[s, ■[t̄], ū] := ■[s, t̄, ū]),

(Definition (ta⊢): 3) $\underset{a,m,n,s,\bar{u}}{\forall}$ (□[s, ▫[a, m], ▫[a, n], ū] := □[s, ▫[a, m + n], ū]),

(Definition (ta⊢): 4) $\underset{a,m,n,s,\bar{u}}{\forall}$ (■[s, ▫[a, m], ▫[a, n], ū] := ■[s, ▫[a, m + n], ū]),

(Definition (ta⊢): 5) $\underset{a,s,\bar{u}}{\forall}$ (□[s, ▫[a, 0], ū] := □[s, ū]),

(Definition (ta⊢): 6) $\underset{a,s,\bar{u}}{\forall}$ (■[s, ▫[a, 0], ū] := ■[s, ū]),

(Definition (ta⊢): 7) $\underset{n,\bar{u}}{\forall}$ (□[▫[0, n], ū] := □[ū]),

(Definition (ta⊢): 8) $\underset{n,\bar{u}}{\forall}$ (■[ū, ▫[0, n]] := ■[ū]),

(Definition (ta⊢): 9) ic := ⟨1, ⟨□[], 0, ■[]⟩⟩,

(Definition (r↵:): 1) $\underset{u,v,z,l,r,s,t,n}{\forall}$ (r↵[⟨u, L, s⟩, ⟨t, ⟨□[l, ▫[v, n^+]], z, ■[r]⟩⟩] :=,
⟨s, ⟨□[l, ▫[v, n]], v, ■[▫[u, 0^+], r]⟩⟩)

(Definition (r↵:): 2)
$\underset{u,z,r,s,t}{\forall}$ (r↵[⟨u, L, s⟩, ⟨t, ⟨□[], z, ■[r]⟩⟩] := ⟨s, ⟨□[], 0, ■[▫[u, 0^+], r]⟩⟩),

(Definition (r↵:): 3) $\underset{u,v,z,l,r,s,t,n}{\forall}$ (r↵[⟨u, R, s⟩, ⟨t, ⟨□[l], z, ■[▫[v, n^+], r]⟩⟩] :=,
⟨s, ⟨□[l, ▫[u, 0^+]], v, ■[▫[v, n], r]⟩⟩)

(Definition (r↵:): 4)
$\underset{u,z,l,s,t}{\forall}$ (r↵[⟨u, R, s⟩, ⟨t, ⟨□[l], z, ■[]⟩⟩] := ⟨s, ⟨□[l, ▫[u, 0^+]], 0, ■[]⟩⟩),

(Definition (TM↵:): 1) $\underset{P,c}{\forall}$ (TM↵[P, c] := r↵[rc[P, c], c]),

(Definition (TM↵:): 2) $\underset{P,c}{\forall}$ (TM↵[P, c, 0] := c),

(Definition (TM↵:): 3) $\underset{P,c,s}{\forall}$ (TM↵[P, c, s^+] := TM↵[P, TM↵[P, c, s]]),

(Definition (TM↵:): 4) $\underset{P,c}{\forall}$ (TM⋱[P, c, 0] := ⟨c⟩),

(Definition (TM↵:): 5) $\underset{P,c,s}{\forall}$ (TM⋱[P, c, s^+] := c ⌣ TM⋱[P, TM↵[P, c], s]).

As there are several methods which can be applied, we have different choices to proceed with the proof.

Alternative proof 1: failed

The proof of (Proposition (TM0 left run)) fails.(The Simplifier was unable to transform the proof situation.)

Alternative proof 2: proved

We prove (Proposition (TM0 left run)) by induction on *k*.

Induction Base:

(1)

$\underset{n,l}{\forall}\left(\mathrm{TM}{\hookleftarrow}[\mathrm{TM0}, \langle 2, \langle \square[l], 1, \blacksquare[\square[1, n+0]]\rangle\rangle, 0] = \langle 2, \langle \square[l, \square[0, 0]], 1, \blacksquare[\square[1, n]]\rangle\rangle\right).$

As there are several methods which can be applied, we have different choices to proceed with the proof.

Alternative proof 1: proved

We take in (1) all variables arbitrary but fixed and prove:

(4) $\mathrm{TM}{\hookleftarrow}[\mathrm{TM0}, \langle 2, \langle \square[\overline{l_1}], 1, \blacksquare[\square[1, n_1+0]]\rangle\rangle, 0] = \langle 2, \langle \square[\overline{l_1}, \square[0, 0]], 1, \blacksquare[\square[1, n_1]]\rangle\rangle$

.

A proof by simplification of (4) works.

Simplification of the lhs term:

$\mathrm{TM}{\hookleftarrow}[\mathrm{TM0}, \langle 2, \langle \square[\overline{l_1}], 1, \blacksquare[\square[1, n_1+0]]\rangle\rangle, 0]$ =by (Proposition (nu⊢): 1)

$\mathrm{TM}{\hookleftarrow}[\mathrm{TM0}, \langle 2, \langle \square[\overline{l_1}], 1, \blacksquare[\square[1, n_1]]\rangle\rangle, 0]$ =by (Definition (TM↵:): 2)

$\langle 2, \langle \square[\overline{l_1}], 1, \blacksquare[\square[1, n_1]]\rangle\rangle\rfloor$

Simplification of the rhs term:

$\langle 2, \langle \square[\overline{l_1}, \square[0, 0]], 1, \blacksquare[\square[1, n_1]]\rangle\rangle$ =by (Definition (ta⊢): 5)

$\langle 2, \langle \square[\overline{l_1}], 1, \blacksquare[\square[1, n_1]]\rangle\rangle\rfloor$

Alternative proof 2: pending

Pending proof of (1).

Induction Step:

Induction Hypothesis:

(2) $\underset{n,l}{\forall}\left(\mathrm{TM}{\hookleftarrow}[\mathrm{TM0}, \langle 2, \langle \square[l], 1, \blacksquare[\square[1, n+k_1]]\rangle\rangle, k_1] = \langle 2, \langle \square[l, \square[0, k_1]], 1, \blacksquare[\square[1, n]]\rangle\rangle\right)$

Induction Conclusion:

(3) $\underset{n,l}{\forall}\left(\mathrm{TM}{\hookleftarrow}[\mathrm{TM0}, \langle 2, \langle \square[l], 1, \blacksquare[\square[1, n+k_1^{+}]]\rangle\rangle, k_1^{+}] = . \langle 2, \langle \square[l, \square[0, k_1^{+}]], 1, \blacksquare[\square[1, n]]\rangle\rangle\right)$

As there are several methods which can be applied, we have different choices to proceed with the proof.

Alternative proof 1: proved

We take in (3) all variables arbitrary but fixed and prove:

(5) TM↵[TM0, 〈2, 〈□[$\overline{l_2}$], 1, ■[□[1, $n_2 + k_1^+$]]〉〉, k_1^+] =.
〈2, 〈□[$\overline{l_2}$, □[0, k_1^+]], 1, ■[□[1, n_2]]〉〉

A proof by simplification of (5) works.

Simplification of the lhs term:

TM↵[TM0, 〈2, 〈□[$\overline{l_2}$], 1, ■[□[1, $n_2 + k_1^+$]]〉〉, k_1^+] =by (Proposition (nu⊢): 2)

TM↵[TM0, 〈2, 〈□[$\overline{l_2}$], 1, ■[□[1, $n_2^+ + k_1$]]〉〉, k_1^+] =by (Definition (TM↵:): 3)

TM↵[TM0, TM↵[TM0, 〈2, 〈□[$\overline{l_2}$], 1, ■[□[1, $n_2^+ + k_1$]]〉〉, k_1]] =by (2)

TM↵[TM0, 〈2, 〈□[$\overline{l_2}$, □[0, k_1]], 1, ■[□[1, n_2^+]]〉〉] =by (Definition (TM↵:): 1)

r↵[rc[TM0, 〈2, 〈□[$\overline{l_2}$, □[0, k_1]], 1, ■[□[1, n_2^+]]〉〉],
〈2, 〈□[$\overline{l_2}$, □[0, k_1]], 1, ■[□[1, n_2^+]]〉〉] =by (Definition (TM0:): 4)

r↵[〈0, R, 2〉, 〈2, 〈□[$\overline{l_2}$, □[0, k_1]], 1, ■[□[1, n_2^+]]〉〉] =by (Definition (r↵:): 3)

〈2, 〈□[$\overline{l_2}$, □[0, k_1], □[0, 0^+]], 1, ■[□[1, n_2]]〉〉 =by (Definition (ta⊢): 3)

〈2, 〈□[$\overline{l_2}$, □[0, $k_1 + 0^+$]], 1, ■[□[1, n_2]]〉〉 =by (Proposition (nu⊢): 2)

〈2, 〈□[$\overline{l_2}$, □[0, $k_1^+ + 0$]], 1, ■[□[1, n_2]]〉〉 =by (Proposition (nu⊢): 1)

〈2, 〈□[$\overline{l_2}$, □[0, k_1^+]], 1, ■[□[1, n_2]]〉〉⌋

Simplification of the rhs term:

〈2, 〈□[$\overline{l_2}$, □[0, k_1^+]], 1, ■[□[1, n_2]]〉〉⌋

Alternative proof 2: pending

Pending proof of (3).

25 of 46

Automated Synthesis of the Gröbner Bases Algorithm by the "Lazy Thinking Method" (BB 2002 ...)

Starting from a formal (predicate logic) specification of the problem,

$$\underset{\text{is-finite}[F]}{\forall} \begin{pmatrix} \textbf{is-finite}[\,\textbf{Gb}[F]\,] \\ \textbf{is-Gröbner-basis}[\,\textbf{Gb}[F]] \\ \textbf{ideal}[F] = \textbf{ideal}[\,\textbf{Gb}[F]]. \end{pmatrix}$$

and a list of possible "algorithm schemes", e.g.

```
A[F] = A[F, pairs[F]]
A[F, ⟨⟩] = F

A[F, ⟨⟨g1, g2⟩, p̄⟩] =
 where[f = lc[g1, g2], h1 = trd[rd[f, g1], F], h2 = trd[rd[f, g2], F],
   { A[F, ⟨p̄⟩]                                              ⇐ h1 = h2
   { A[F ⌢ df[h1, h2], ⟨p̄⟩ ≍ ⟨⟨F_k, df[h1, h2]⟩ | k=1,…,|F| ⟩]  ⇐ otherwise  ]
```

by this new algorithm synthesis method, the key idea of the main theorem (the notion of S-polynomial) is automatically generated and verified:

```
lc[g1, g2] = lcm[lp[g1], lp[g2]],
```

```
df[h1, h2] = h1 - h2.
```

26 of 46

The Algorithm Synthesis Method ("Lazy Thinking") is Based on

- the use of "formula schemes" (re-use high-level mathematical knowledge),
- automated theorem proving,
- learning from failing proofs.

27 of 46

The Essential Problem

The problem of synthesizing a Gröbner bases algorithm can now be also stated by asking whether, starting with the proof of

$$\underset{F}{\forall}\begin{pmatrix}\texttt{is-finite[A[F]]}\\ \texttt{is-Gröbner-basis[A[F]]}\\ \texttt{ideal[F] = ideal[A[F]]}\end{pmatrix},$$

we can *automatically* produce the idea that

```
lc[g1, g2] = lcm[lp[g1], lp[g2]]
```

and

```
df[h1, h2] = h1 - h2
```

and prove that the idea is correct.

Now Start the (Automated) Correctness Proof

With current theorem proving technology, in the *Theorema* system (and other provers), the proof attempt could be done automatically. (Ongoing PhD thesis of A. Craciun.)

Details

It should be clear that, if the algorithm terminates, the final result is a finite set (of polynomials) G that has the property

$$\underset{g1,g2\in G}{\forall}\left(\mathrm{where}\left[f = \mathrm{lc}[g1, g2],\ h1 = \mathrm{trd}[\mathrm{rd}[f, g1], F],\ h2 = \mathrm{trd}[\mathrm{rd}[f, g2], F],\ \bigvee\begin{cases} h1 = h2 \\ \mathrm{df}[h1, h2] \in G\end{cases}\right]\right).$$

We now try to prove that, if G has this property, then

```
is-finite[G],
ideal[F] = ideal[G],
is-Gröbner-basis[G],
   i.e. is-Church-Rosser[ →G ].
```

Here, we only deal with the third, most important, property.

Using Available Knowledge

Using Newman's lemma and some elementary properties it can be shown that it is sufficient to prove

$$\text{is-Church-Rosser}[\rightarrow_G] \Leftrightarrow \underset{p}{\forall}\ \underset{f1,f2}{\forall} \left(\left(\begin{cases} p \rightarrow f1 \\ p \rightarrow f2 \end{cases}\right) \Rightarrow f1 \downarrow^* f2\right).$$

31 of 46

The (Automated) Proof Attempt

Let now the power product p and the polynomials f1, f2 be arbitary but fixed and assume

$$\begin{cases} p \rightarrow_G f1 \\ p \rightarrow_G f2. \end{cases}$$

We have to find a polyonomial g such that

$$f1 \rightarrow_G^* g,$$
$$f2 \rightarrow_G^* g.$$

From the assumption we know that there exist polynomials g1 and g2 in G such that

$$lp[g1] \mid p,$$
$$f1 = rd[p, g1],$$
$$lp[g2] \mid p,$$
$$f2 = rd[p, g2].$$

From the final situation in the algorithm scheme we know that for these g1 and g2

$$\bigvee \begin{cases} h1 = h2 \\ df[h1, h2] \in G, \end{cases}$$

where

$$h1 := trd[f1', G],\ f1' := rd[lc[g1, g2], g1],$$
$$h2 := trd[f2', G],\ f2' := rd[lc[g1, g2], g2].$$

32 of 46

Case h1=h2

$$lc[g1, g2] \rightarrow_{g1} rd[lc[g1, g2], g1] \rightarrow_G^* trd[rd[lc[g1, g2], g1], G] =$$
$$trd[rd[lc[g1, g2], g2], G] \leftarrow_G^* rd[lc[g1, g2], g2] \leftarrow_{g2} lc[g1, g2].$$

(Note that here we used the requirements rd[lc[g1,g2],g1]<lc[g1,g2] and rd[lc[g1,g2],g2]<lc[g1,g2].)

Hence, by elementary properties of polynomial reduction,

$$\underset{a,q}{\forall}\left(a\,q\,\mathrm{lc}[g1, g2] \to_{g1} a\,q\,\mathrm{rd}[\mathrm{lc}[g1, g2], g1] \to_{G}^{*} a\,q\,\mathrm{trd}[\mathrm{rd}[\mathrm{lc}[g1, g2], g1], G] = a\,q\,\mathrm{trd}[\mathrm{rd}[\mathrm{lc}[g1, g2], g2], G] \leftarrow_{G}^{*} a\,q\,\mathrm{rd}[\mathrm{lc}[g1, g2], g2] \leftarrow_{g2} a\,q\,\mathrm{lc}[g1, g2]\right).$$

Now we are stuck in the proof.

33 of 46

Now Use a Specification Generation Algorithm

Specification generation rule (rough sketch; the intelligence is in the details of this rule!): Collect the temporary assumptions and temporary goals, write a "⟹" in between and generalize from constant terms to variables. (The details are a little tricky.)

In the case of the proof at hand, we see that we could proceed successfully with the proof if lc[g1,g2] satisfied the following requirement

$$\underset{p,g1,g2}{\forall}\left(\left(\begin{cases}\mathrm{lp}[g1] \mid p \\ \mathrm{lp}[g2] \mid p\end{cases}\right) \Rightarrow \left(\underset{a,q}{\exists}\left(p = a\,q\,\mathrm{lc}[g1, g2]\right)\right)\right), \quad (\textbf{lc requirement})$$

With such an lc, we then would have

$$p \to_{g1} \mathrm{rd}[p, g1] = a\,q\,\mathrm{rd}[\mathrm{lc}[g1, g2], g1] \to_{G}^{*} a\,q\,\mathrm{trd}[\mathrm{rd}[\mathrm{lc}[g1, g2], g1], G] = a\,q\,\mathrm{trd}[\mathrm{rd}[\mathrm{lc}[g1, g2], g2], G] \leftarrow_{G}^{*} a\,q\,\mathrm{rd}[\mathrm{lc}[g1, g2], g2] = \mathrm{rd}[p, g2] \leftarrow_{g2} p$$

and, hence,

$$f1 \to_{G}^{*} a\,q\,\mathrm{trd}[\mathrm{rd}[\mathrm{lc}[g1, g2], g1], G],$$

$$f2 \to_{G}^{*} a\,q\,\mathrm{trd}[\mathrm{rd}[\mathrm{lc}[g1, g2], g1], G],$$

i.e. we would have found a suitable g.

34 of 46

Summarize the (Automatically Generated) Specifications of the Subalgorithm lc

(lc requirement), which also could be written in the form:

$$\underset{p,g1,g2}{\forall}\left(\left(\begin{cases} \mathrm{lp}[g1] \mid p \\ \mathrm{lp}[g2] \mid p \end{cases}\right) \Rightarrow (\mathrm{lc}[g1, g2] \mid p)\right),$$

and

```
lp[g1] | lc[g1, g2],
lp[g2] | lc[g1, g2],
```

wich is a consequence of

```
rd[lc[g1, g2], g1] < lc[g1, g2],
rd[lc[g1, g2], g2] < lc[g1, g2].
```

Summarize Again

For synthesizing an algorithm for the Gröbner bases problem it suffices to find an lc satisfying

$$\underset{p,g1,g2}{\forall}\left(\left(\begin{cases} \mathrm{lp}[g1] \mid p \\ \mathrm{lp}[g2] \mid p \end{cases}\right) \Rightarrow (\mathrm{lc}[g1, g2] \mid p)\right),$$

and

```
lp[g1] | lc[g1, g2],
lp[g2] | lc[g1, g2].
```

This problem can be solved by any high-school student (or university professor)! No knowledge on Gröbner bases theory necessary!

A Suitable lc

```
lcp[g1, g2] = lcm[lp[g1], lp[g2]]
```

is a suitable function that satisfies the above requirements.

Eureka! The crucial function lc (the "critical pair" function) in the critical pair / completion algorithm scheme has been synthesized automatically!

36 of 46

Case h1≠h2

In this case, df[h1,h2]∈G:

In this part of the proof (wich is much easier) we are basically stuck right at the beginning. By the requirement generation algorithm we obtain the following requirement for df:

$$\underset{h1,h2}{\forall} \left(h1 \downarrow_{\{df[h1,h2]\}}{}^{*}\, h2\right) \quad (\text{df requirement}).$$

37 of 46

Looking to the Knowledge Base for a Suitable df

(Looking to the knowledge base of elementary properties of polynomial reduction, it is now easy to find a function df that satifies (df requirement), namely

```
df[h1, h2] = h1 - h2,
```

because, in fact,

$$\underset{f,g}{\forall} \left(f \downarrow_{\{f-g\}}{}^{*}\, g\right).$$

Eureka! The function df (the "completion" function) in the critical pair / completion algorithm scheme has been "automatically" synthesized!)

Conclusion

We illustrated the automated synthesis of a non-trivial algorithm.

- Non trivial: ~ 1960 a conjecture was made that (Groebner bases) related problems are algorithmically unsolvable.
- An algorithm was found 1965 by human (young BB) mathematical exploration.
- A human (old BB) systematic algorithm invention method was able to synthesize, 2005, an algorithm automatically.

Implications for increase in efficiency of the mathematical exploration process (with implications on all sciences).

Possible implications on life sciences:

- synthesizing "algorithms" between structure and behavior.
- understanding the phenomenon of self-reference in evolution.
- ...

Appendix: More Details on Gröbner Bases and References

How Difficult is the Construction of Gröbner Bases?

Very Easy

The structure of the algorithm is easy. The operations needed in the algorithm are elementary. "Every high-school student can execute the algorithm." (See palm-top TI-98.)

Very Difficult

The inherent complexity of the problems that can be solved by the GB method (e.g. graph colorings) is "exponential". Hence, the worst-case complexity of the GB algorithm *must* be high.

Sometimes Easy

Mathematically interesting examples often have a lot of "structure" and, in concrete examples, GB computations can be reasonably, even surprisingly, fast.

Enormous Potential for Improvement

More *mathematical* theorems can lead to drastic speed-up:

- The use of "criteria" for eliminating the consideration of certain S-polynomials.
- *p*-adic approaches and floating point approaches.
- The "Gröbner Walk" approach.
- The "linear algebra" approach. (Generalized Sylvester matrices.)
- The "numerics" approach.

Tuning of the algorithm:

- Heuristics, strategies for choosing orderings, selecting S-polynomials etc.
- Good implementation techniques.

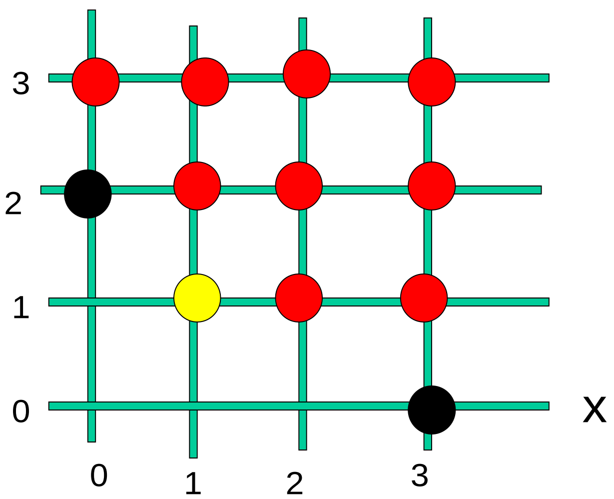
A huge literature.

40 of 46

Why "Gröbner" Bases?

Professor Wolfgang Gröbner (1899-1980) was my PhD thesis supervisor.

He gave me the problem of finding "the uncovered points if the black points are given".



In my thesis (1965) and journal publication (1970) I introduced:

* the concept of Gröbner bases and reduced Gröbner bases

* the S-polynomials

* the main theorem with proof

* the algorithm with termination and correctness proof

* the uniqueness of Gröbner bases

* first applications (computing in residue rings, Hilbert function, algebraic systems)

* the technique of base-change w.r.t. to different orderings

* a complete computer implementation

* first complexity considerations.

However, in the thesis, I did not use the name "Gröbner bases". I introduced this name only in 1976, for honoring Gröbner, when people started to become interested in my work.

My later contributions:

* the technique of criteria for eliminating unnecessary reductions

* an abstract characterization of "Gröbner bases rings".

41 of 46

Gröbner Bases on Your Desk and in Your Palm

GB implementations are contained in all the current math software systems like *Mathematica* (see demo), Maple, Magma, Macsyma, Axiom, Derive, Reduce, Mupad, ...

Software systems specialized on Gröbner bases: RISA-ASIR (M. Noro, K. Yokoyama), CoCoA, Macaulay, Singular, ...

Gröbner bases are now availabe on the TI-98 (implemented in Derive).

42 of 46

Textbooks on Gröbner Bases

T. Kreuzer, L. Robbiano: *Algorithmic Commutative Algebra I.* Springer, Heidelber, 2000: Contains a list of all other, approx. 10, textbooks on GB.

W.W.Adams, P. Loustenau. *Introduction to* Gröbner *Bases*. Graduate Studies in Mathematics: Amer. Math. Soc., Providence, R.I., 1994.

T.Becker, V.Weispfenning. Gröbner *Bases: A Computational Approach to Commutative Algebra.* Springer, New York, 1993.

D.Cox, J.Little, D.O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra.* Springer, New York, 1992.

....

M. Maruyama. Gröbner Bases and Applications. 2002.

M. Noro, K. Yokoyama. Computational Fundamentals of Gröbner Bases. University of Tokyo Press, 2003.

43 of 46

Gröbner Bases on the Web

Search. E.g. in the Research Index you obtain ~ 3000 citations.

44 of 46

Original Publications on Gröbner Bases

Approximately 600 papers appeared meanwhile on Gröbner bases.

J of Symbolic Computation, in particular, special issues.

ISSAC Conferences.

Mega Conferences.

ACA Conferences.

...

The essential additional original ideas in the literature:

- Gröbner bases can be constructed w.r.t. arbitrary "admissible" orderings (W. Trinks 1978)
- Gröbner bases w.r.t. to "lexical" orderings have the elimination property (W. Trinks 1978)
- Gröbner bases can be used for computing syzygies and the S-polys generate the module of syzygies (G. Zacharias 1978)
- A given F, w.r.t. the *infinitely* many admissible orderings, has only *finitely* many Gröbner bases and, hence, we can construct a "universal" Gröbner bases for F (L. Robbiano, V. Weispfenning, T. Schwarz 1988)
- Starting from a Gröbner bases for F for ordering O_1 one can "walk", by changing the basis only slightly, to a basis for a "nearby" ordering O_2 and so on ... until one arrives at a Gröbner bases for a desired ordering O_k (Kalkbrener, Mall 1995, Nam 2000).

- Use arbitrary linear algebra algorithms for the reduction (remaindering) process: (Faugère 1997).
- ... numerours applications,

Research Topics

- the inner structure of Groebner bases: generalized Sylvester matrices
- the numerics of GB computations
- axiomatic characterization of Groebner rings
- generalizations (e.g. non-commutative poly-rings)
- speeding up the computation
- Groebner bases for particular classes of ideals (avoid computation)
- the study of admissible orderings
- applications (problem reductions, e.g. functional analysis, BV problems, **Rosenkranz 2003**)

References

On Gröbner Bases

[Buchberger 1970]

B. Buchberger. Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems (An Algorithmical Criterion for the Solvability of Algebraic Systems of Equations). Aequationes mathematicae 4/3, 1970, pp. 374-383. (English translation in: [Buchberger, Winkler 1998], pp. 535 -545.) Published version of the PhD Thesis of B. Buchberger, University of Innsbruck, Austria, 1965.

[Buchberger 1998]

B. Buchberger. Introduction to Gröbner Bases. In: [Buchberger, Winkler 1998], pp.3-31.

[Buchberger, Winkler, 1998]

B. Buchberger, F. Winkler (eds.). Gröbner Bases and Applications, Proceedings of the International Conference "33 Years of Gröbner Bases", 1998, RISC, Austria, London Mathematical Society Lecture Note Series, Vol. 251, Cambridge University Press, 1998.

[Becker, Weispfenning 1993]

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

■ On Mathematical Knowledge Management

B. Buchberger, G. Gonnet, M. Hazewinkel (eds.)

Mathematical Knowledge Management.

Special Issue of Annals of Mathematics and Artificial Intelligence, Vol. 38, No. 1-3, May 2003, Kluwer Academic Publisher, 232 pages.

A.Asperti, B. Buchberger, J.H.Davenport (eds.)

Mathematical Knowledge Management.

Proceedings of the Second International Conference on Mathematical Knowledge Management (MKM 2003), Bertinoro, Italy, Feb.16-18, 2003, Lecture Notes in Computer Science, Vol. 2594, Springer, Berlin-Heidelberg-NewYork, 2003, 223 pages.

A.Asperti, G.Bancerek, A.Trybulec (eds.).

Proceedings of the Third International Conference on Mathematical Knowledge Management, MKM 2004,

Bialowieza, Poland, September 19-21, 2004, Lecture Notes in Computer Science, Vol. 3119, Springer, Berlin-Heidelberg-NewYork, 2004

■ On Theorema

[Buchberger et al. 2000]

B. Buchberger, C. Dupre, T. Jebelean, F. Kriftner, K. Nakagawa, D. Vasaru, W. Windsteiger. The Theorema Project: A Progress Report. In: M. Kerber and M. Kohlhase (eds.), Symbolic Computation and Automated Reasoning (Proceedings of CALCULEMUS 2000, Symposium on the Integration of Symbolic Computation and Mechanized Reasoning, August 6-7, 2000, St. Andrews, Scotland), A.K. Peters, Natick, Massachusetts, ISBN 1-56881-145-4, pp. 98-113.

■ On Theory Exploration and Algorithm Synthesis

[Buchberger 2000]

B. Buchberger. Theory Exploration with *Theorema*.

Analele Universitatii Din Timisoara, Ser. Matematica-Informatica, Vol. XXXVIII, Fasc.2, 2000, (Proceedings of SYNASC 2000, 2nd International Workshop on Symbolic and Numeric Algorithms in Scientific Computing, Oct. 4-6, 2000, Timisoara, Rumania, T. Jebelean, V. Negru, A. Popovici eds.), ISSN 1124-970X, pp. 9-32.

[Buchberger 2003]

B. Buchberger. Algorithm Invention and Verification by Lazy Thinking.

In: D. Petcu, V. Negru, D. Zaharie, T. Jebelean (eds), Proceedings of SYNASC 2003 (Symbolic and

Numeric Algorithms for Scientific Computing, Timisoara, Romania, October 1–4, 2003), Mirton Publishing, ISBN 973–661–104–3, pp. 2–26.

[Buchberger, Craciun 2003]

B. Buchberger, A. Craciun. Algorithm Synthesis by Lazy Thinking: Examples and Implementation in Theorema. in: Fairouz Kamareddine (ed.), Proc. of the Mathematical Knowledge Management Workshop, Edinburgh, Nov. 25, 2003, Electronic Notes on Theoretical Computer Science, volume dedicated to the MKM 03 Symposium, Elsevier, ISBN 044451290X, to appear.

[Buchberger 2004]

B. Buchberger.

Towards the Automated Synthesis of a Gröbner Bases Algorithm.

RACSAM (Review of the Royal Spanish Academy of Science), Vol. 98/1, to appear, 10 pages.