

Symbolic Computation: Current Trends

Bruno Buchberger
RISC, Linz, Austria

Talk at Max Planck Institute for Physics, München, January 31,
2006

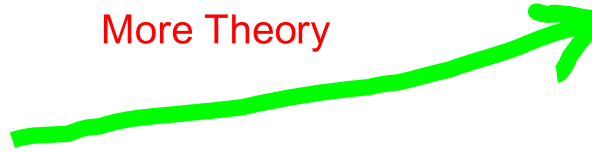
Copyright Bruno Buchberger 2006.

Copyright Note: This file may be copied, stored and distributed under the following conditions:

- the file is kept complete and unchanged including this copyright note
- an e-message is sent to bruno.buchberger@jku.at
- if material from this talk is used in publications etc., this talk must be cited appropriately.

The Evolution of "Computing"

More Theory



Numerics

algorithms
in

approximate
finitary representations

of infinite
mathematical structures

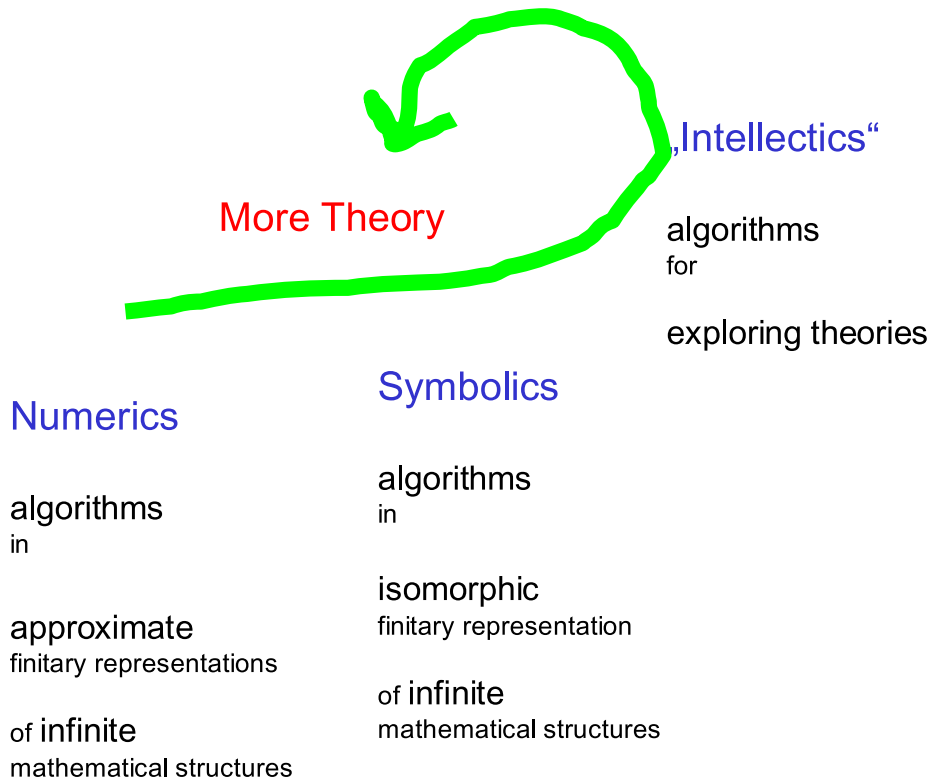
Symbolics

algorithms
in

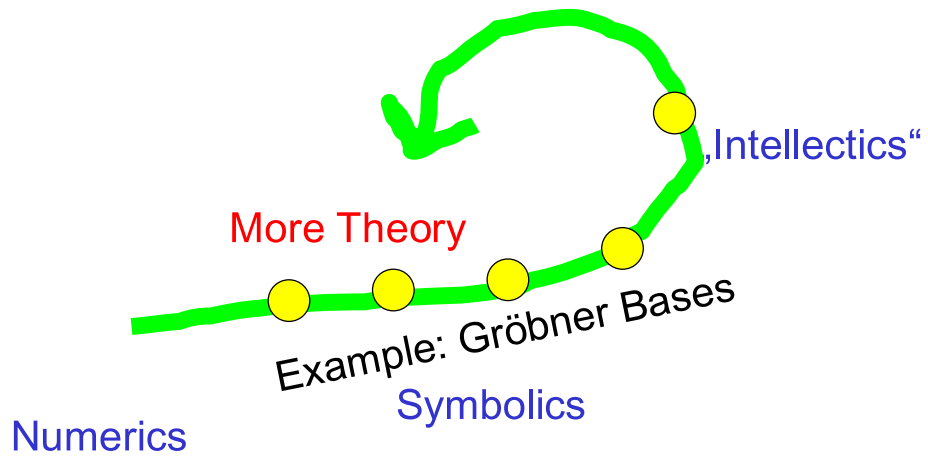
isomorphic
finitary representation

of infinite
mathematical structures

The Evolution of "Computing"



In This Talk



Current Math Systems

More Interaction Numerics / Symbolics

More Symbolics

More Intellectics

Appendix

Current Math Systems

More Interaction Numerics / Symbolics

More Symbolics

More Intellectics

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

- Systems like *Mathematica*, Maple, Derive, Matlab, ... FORM, Singular, Cocoa, ...
- An enormous potential for science (physics, ...) and engineering.
- Help!

Example:

```
DSolve[{y''[x] == a y'[x] + y[x], y[0] == 1, y'[0] == 0}, y, x]
```

```
{ {y -> Function[{x},
  
$$\frac{a e^{\frac{1}{2} (a - \sqrt{4+a^2}) x} + \sqrt{4+a^2} e^{\frac{1}{2} (a - \sqrt{4+a^2}) x} - a e^{\frac{1}{2} (a + \sqrt{4+a^2}) x} + \sqrt{4+a^2} e^{\frac{1}{2} (a + \sqrt{4+a^2}) x}}{2 \sqrt{4+a^2}}$$

  ]} }
```

⏪ ⏩ ⏴ ⏵

9 of 66

Example:

```
Solve[{1/s + 1/t == 1/F, 1/(d+s) + 1/(t-e) == 1/F, c == e F / (f (t-e)), M == t/s}, {d, e, s, t}]
```

```
{ {d -> - c f F (1+M) / (M (c f - F M)), e -> c f F (1+M) / (c f + F), s -> F (1+M) / M, t -> F (1+M)} }
```

⏪ ⏩ ⏴ ⏵

10 of 66

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](#):

- [What](#) are Gröbner bases?
- [How](#) can Gröbner bases be computed?
- [Why](#) are Gröbner bases important? (Dozens of fundamental problems can be reduced to Gröbner bases construction!)

⏪ ⏩ ⏴ ⏵

11 of 66

The Linear Combination of Polynomials

$$f_1 = -2y + xy$$

$$f_2 = -x^2 + y^2$$

$$-2y + xy$$

$$-x^2 + y^2$$

Leading power products: w.r.t. an ordering of the power products (e.g. lexicographically, by total degree or ...)

Consider now the following linear combination of f_1 and f_2 :

$$g = (y) f_1 + (-x + 2) f_2$$

$$y(-2y + xy) + (2 - x)(-x^2 + y^2)$$

$$g = (y) f_1 + (-x + 2) f_2 \quad // \text{Expand}$$

$$-2x^2 + x^3$$

Observation: The leading power product x^3 of g is

neither a multiple of the leading power product xy of f_1

nor a multiple of the leading power product y^2 of f_2 .



12 of 66

Definition of Groebner Bases

A set F of **polynomials** is called a **Groebner basis** (w.r.t. the chosen ordering of power products) iff the above phenomenon cannot happen, i.e.

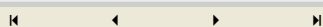
for all $f_1, \dots, f_m \in F$ and all (**infinitely many**) polynomials h_1, \dots, h_m ,

the leading power product of $h_1 f_1 + \dots + h_m f_m$

is a multiple of the leading power product of

at least one of the polynomials in F .

Counterexample: The Set $F = \{f_1, f_2\}$ of the Above Example is **not** a Groebner basis.



13 of 66

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

F is a Gröbner basis $\iff \forall_{f_1, f_2 \in F} \text{remainder}[F, \text{S-polynomial}[f_1, f_2]] = 0$.

$$\text{s-polynomial}[-2y + xy, -x^2 + y^2] = y(-2y + xy) - x(-x^2 + y^2)$$

$$x^3 - 2y^2$$

Proof: Nontrivial. Combinatorial.

The theorem **reduces** an **infinite** check to a **finite** check: Recall definition of "F is a Gröbner basis":

for all $f_1, \dots, f_m \in F$ and all (**infinitely many**) polynomials h_1, \dots, h_m ,

the leading power product of $h_1 f_1 + \dots + h_m f_m$

is a multiple of the leading power product of at least one of the polynomials in F.

The power of the Gröbner bases method is contained in this theorem and its proof.

⏪ ⏩

14 of 66

The Problem of **Constructing** Gröbner Bases

Given F , find G such that G is a Gröbner basis

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

⏪ ⏩

15 of 66

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

Recall the main theorem:

F is a Gröbner basis $\iff \forall_{f_1, f_2 \in F} \text{remainder}[F, \text{S-polynomial}[f_1, f_2]] = 0$.

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

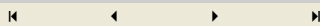
Start with $G := F$.

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

$h := \text{remainder}[G, \text{S-polynomial}[f_1, f_2]]$

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

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



16 of 66

Termination of the Algorithm

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



17 of 66

Example of Application: Solve Systems

```
f1 = x y - 2 y z - z;
f2 = y2 - x2 z + x z;
f3 = z2 - y2 x + x;
F = {f1, f2, f3};
```

```
{time, G} = GroebnerBasis[F] // Timing
```

```
{0.01 Second,
{-z - 4 z3 + 17 z4 - 3 z5 + 45 z6 - 60 z7 + 29 z8 - 124 z9 + 48 z10 - 64 z11 + 64 z12,
-22001 z + 14361 y z + 16681 z2 + 26380 z3 + 226657 z4 + 11085 z5 -
90346 z6 - 472018 z7 - 520424 z8 - 139296 z9 - 150784 z10 + 490368 z11,
43083 y2 - 11821 z + 267025 z2 - 583085 z3 + 663460 z4 - 2288350 z5 +
2466820 z6 - 3008257 z7 + 4611948 z8 - 2592304 z9 + 2672704 z10 - 1686848 z11,
43083 x - 118717 z + 69484 z2 + 402334 z3 + 409939 z4 + 1202033 z5 -
2475608 z6 + 354746 z7 - 6049080 z8 + 2269472 z9 - 3106688 z10 + 3442816 z11}}
```

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

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

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

```
{1.11022 × 10-15 + 5.55112 × 10-16 i,
(-523.519 - 4967.65 i) - (4757.86 + 8428.97 i) y,
(-7846.9 - 8372.06 i) + 43083 y2, (-16311.7 + 16611. i) + 43083 x}
```

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

```
{{y -> -0.473535 - 0.205184 i}}
```

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

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

```
{{x -> 0.378611 - 0.385558 i}}
```

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

```
{-3.21965 × 10-15 - 3.45557 × 10-15 i,
4.02456 × 10-15 - 8.04912 × 10-16 i, 5.07927 × 10-15 + 1.83187 × 10-15 i}
```

Example of Application: Invariant Theory

A Question: Can

$$h = x_1^7 x_2 - x_1 x_2^7$$

$$x_1^7 x_2 - x_1 x_2^7$$

be expressed as a polynomial in

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

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

?

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

19 of 66

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. Second,
  {-i1^2 i2 + 4 i2^2 + i3^2, i2 - i1 x1^2 + x1^4, -i1^2 i3 x1 + 2 i2 i3 x1 + i1 i3 x1^3 - i1^2 i2 x2 + 4 i2^2 x2,
   i1^2 x1 - 2 i2 x1 - i1 x1^3 + i3 x2, -i1 i3 + 2 i3 x1^2 - i1^2 x1 x2 + 4 i2 x1 x2,
   -i3 x1 - 2 i2 x2 + i1 x1^2 x2, -i3 - i1 x1 x2 + 2 x1^3 x2, -i1 + x1^2 + x2^2}}
```

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

```
{{0, i3 + 1/2 i1 x1 x2 + x1^3 x2, 0, 3 i1 x2 / 4 - 1/2 x1^2 x2 + x2^3 / 2, i1 - x1^2 / 2 + 3 x2^2 / 4,
  3 i1 x1 / 2 + x1 x2^2, x2^4 / 2, -1/4 i1^2 x1 x2 - 1/2 i1 x1 x2^3 - x1 x2^5}, i1^2 i3 - i2 i3}
```

Theorem (Sweedler, Sturmfels et al. 1988): h can be represented in terms of l iff remainder of h w.r.t. "Groebner basis of l 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
```

```
x1^7 x2 - x1 x2^7
```

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

```
{{0, -1/2 i1 x1 + i1 x2 + x1^2 x2, 0, 3 i1 / 4 - x1^2 / 2 + x2^2 / 2,
  -x1 / 4 + 3 x2 / 4, 3 i1 / 4 + x1 x2, x2^3 / 2, -1/4 i1^2 x1 - 1/2 i1 x1 x2^2 - x1 x2^4},
  -i1^3 x1 + 2 i1 i2 x1 + 1/2 i1 i3 x1 + i1^2 x1^3 - i2 x1^3 + 1/2 i3 x1^3 + 1/2 i1 i2 x2}
```

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

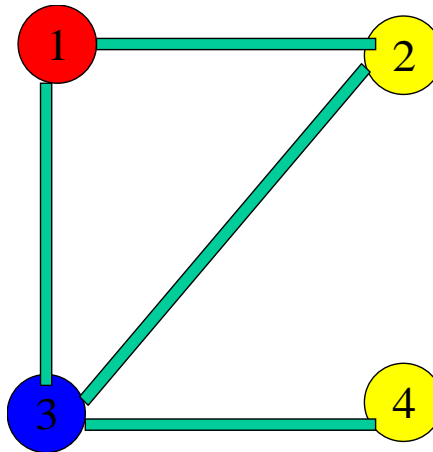
20 of 66

Application: Graph Coloring

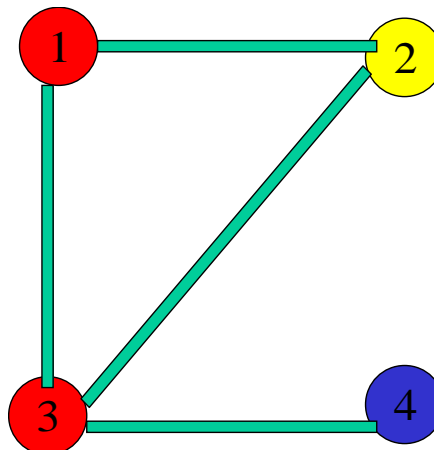
The Problem:

Find all admissible colorings in k colors of a graph with n vertices and edges E :

An admissible coloring in 3 colors of a graph with 4 vertices and edges $\{1,2\}, \{1,3\}, \{2,3\}, \{3,4\}$:



Not an admissible coloring in 3 colors of the same graph:



The Translation into a Groebner Bases Problem

Theorem: The possible colorings of the above graph correspond 1-1 to the common solutions of the following set of polynomials:

```
{ -1 + x13, ... at vertex 1 color is a 3 - ary root of 1
  -1 + x23, ... at vertex 2 color is a 3 - ary root of 1
  -1 + x33,
  -1 + x43,
  x12 + x1 x2 + x22, ... the colors at 1 and 2 must be different,
  x12 + x1 x3 + x32,
  x22 + x2 x3 + x32,
  x32 + x3 x4 + x42}
```

⏪

⏩

⏪

⏩

22 of 66

Solution by Groebner Bases

Compute a Groebner basis of this polynomial set and compute all solutions.

```
GB = GroebnerBasis[{-1 + x13, -1 + x23, -1 + x33, -1 + x43,
  x12 + x1 x2 + x22, x12 + x1 x3 + x32, x22 + x2 x3 + x32, x32 + x3 x4 + x42},
  {x4, x3, x2, x1}]
```

```
{-1 + x13, x12 + x1 x2 + x22, x1 + x2 + x3, x1 x2 - x1 x4 - x2 x4 + x42}
```

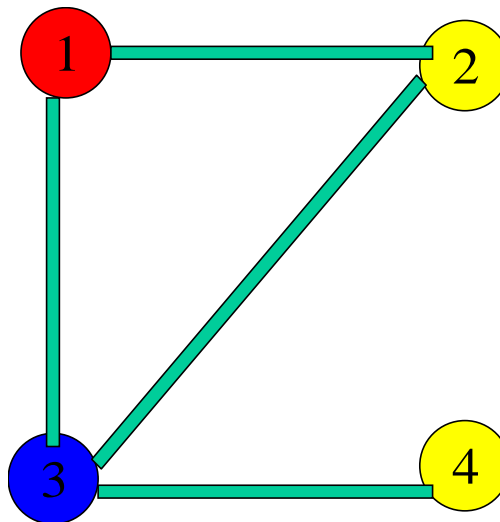
```
Solve[{-1 + x13 == 0, -1 + x23 == 0, -1 + x33 == 0, -1 + x43 == 0,
  x12 + x1 x2 + x22 == 0, x12 + x1 x3 + x32 == 0, x22 + x2 x3 + x32 == 0, x32 + x3 x4 + x42 == 0},
  {x4, x3, x2, x1}]
```

```
{{x4 → 1, x1 → 1, x2 → -1 + (-1)1/3, x3 → -(-1)1/3},
 {x4 → 1, x1 → 1, x2 → -1 - (-1)2/3, x3 → (-1)2/3},
 {x4 → 1, x2 → 1, x1 → -1 + (-1)1/3, x3 → -(-1)1/3},
 {x4 → 1, x2 → 1, x1 → -1 - (-1)2/3, x3 → (-1)2/3},
 {x4 → -(-1)1/3, x2 → -1 + (-1)1/3, x1 → -(-1)1/3, x3 → 1},
 {x4 → -(-1)1/3, x2 → -1 - (-1)2/3, x1 → (-1)2/3, x3 → 1},
 {x4 → (-1)2/3, x2 → -1 + (-1)1/3, x1 → -(-1)1/3, x3 → 1},
 {x4 → (-1)2/3, x2 → -1 - (-1)2/3, x1 → (-1)2/3, x3 → 1},
 {x4 → -1 + (-1)1/3, x1 → 1, x2 → -1 + (-1)1/3, x3 → -(-1)1/3},
 {x4 → -1 + (-1)1/3, x2 → 1, x1 → -1 + (-1)1/3, x3 → -(-1)1/3},
 {x4 → -1 - (-1)2/3, x1 → 1, x2 → -1 - (-1)2/3, x3 → (-1)2/3},
 {x4 → -1 - (-1)2/3, x2 → 1, x1 → -1 - (-1)2/3, x3 → (-1)2/3}}
```

Slightly re-organized output:

```
{x1 → 1, x2 → -(-1)1/3, x3 → -1 + (-1)1/3, x4 → 1},
{x1 → 1, x2 → -(-1)1/3, x3 → -1 + (-1)1/3, x4 → -(-1)1/3},
{x1 → 1, x2 → (-1)2/3, x3 → -1 - (-1)2/3, x4 → 1},
{x1 → 1, x2 → (-1)2/3, x3 → -1 - (-1)2/3, x4 → (-1)2/3},
{x1 → -(-1)1/3, x2 → 1, x3 → -1 + (-1)1/3, x4 → 1},
{x1 → -(-1)1/3, x2 → 1, x3 → -1 + (-1)1/3, x4 → -(-1)1/3},
{x1 → -(-1)1/3, x2 → -1 + (-1)1/3, x3 → 1, x4 → -(-1)1/3},
{x1 → -(-1)1/3, x2 → -1 + (-1)1/3, x3 → 1, x4 → -1 + (-1)1/3},
{x1 → (-1)2/3, x2 → 1, x3 → -1 - (-1)2/3, x4 → 1},
{x1 → (-1)2/3, x2 → 1, x3 → -1 - (-1)2/3, x4 → (-1)2/3},
{x1 → (-1)2/3, x2 → -1 - (-1)2/3, x3 → 1, x4 → (-1)2/3},
{x1 → (-1)2/3, x2 → -1 - (-1)2/3, x3 → 1, x4 → -1 - (-1)2/3}
```

For example, $\{x_1 \rightarrow 1, x_2 \rightarrow -(-1)^{1/3}, x_3 \rightarrow -1 + (-1)^{1/3}, x_4 \rightarrow -(-1)^{1/3}\}$ corresponds to



Application: Integer Optimization

Example (B. Sturmfels):

What is the minimum number of coins (e.g. p Pennies, n Nickels, d Dimes, q Quarters) for composing a given value, e.g. 117?

Reduction to Gröbner Bases Problem (C. Traverso et al. 1986):

Code the integer **values** p, n, d, q as **exponents** of power products!

Code the **goal function** as the (generalized) **degree** of the power products!

Code the **exchange rules** of the coins (the relations between the quantities) as **polynomials** consisting of power products:

$$F = \{P^5 - N, P^{10} - D, P^{25} - Q\}$$

$$\{-N + P^5, -D + P^{10}, P^{25} - Q\}$$

Now compute the Gröbner basis of F (w.r.t. degree ordering):

$$G = \text{GroebnerBasis}[F, \text{MonomialOrder} \rightarrow \text{DegreeLexicographic}]$$

$$\{-D + N^2, D^3 - NQ, D^2N - Q, -N + P^5\}$$

Now you can be sure that, starting with any admissible solution (e.g. $(p=17, n=10, d=5, q=0)$), by reduction modulo G , you will end up with a minimal solution:

$$\text{PolynomialReduce}[P^{17}N^{10}D^5, G, \text{MonomialOrder} \rightarrow \text{DegreeLexicographic}]$$

$$\{ \{D^9 P^{17} + D^8 N^2 P^{17} + D^7 N^4 P^{17} + D^6 N^6 P^{17} + D^5 N^8 P^{17} + D^4 P^{17} Q^2 + P^7 Q^4, \\ D^7 P^{17} + D^4 N P^{17} Q + D^2 P^{17} Q^2, P^{17} Q^3, D P^2 Q^4 + N P^7 Q^4 + P^{12} Q^4\}, DN P^2 Q^4 \}$$

Answer: take 4 quarters, 1 dime, 1 nickel, 2 pennies.

More Applications

Gröbner Bases 98 Conference:

B. B., F. Winkler. *Gröbner Bases: Theory and Applications*. Cambridge University Press, 1998. 560 pages.



This book contains tutorials and original papers.

This book contains also:

B. B. *Introduction to Gröbner Bases*, pp. 3-31.

B. B. *An Algorithmic Criterion for the Solvability of Systems of Algebraic Equations*, pp. 540-560.
(English translation of the original paper from 1970, in which Gröbner bases were introduced.)

A continuation of this book is the special issue of the JSC on Gröbner bases edited by Q.N. Tran and F. Winkler, 2000.

Current Math Systems

More Interaction Numerics / Symbolics

More Symbolics

More Intellectics

Example: The Numerics of Gröbner Bases

In both directions (H. Stetter et al. 1987 - 2006):

- Start from Gröbner bases and compute solutions (reduction to an eigenvalue problem).
- Numerically, compute (a numerical variant) of Gröbner bases.

Current Math Systems

More Interaction Numerics / Symbolics

More Symbolics

More Intellectics



28 of 66

Example: Computation on Operators

computation on (finitary representations of) **numbers**: e.g. computation on algebraic numbers



computation on (finitary representations of) **functions** (on numbers): e.g. symbolic integration



computation on (finitary representations of) **operators** (on functions): e.g. symbolic generation of Green's functions for boundary-value problems

Project SFB 1322 (B.B. and H. Engl, RICAM), PhD thesis and postdoc work of M. Rosenkranz:

M.Rosenkranz, B.B, H.W.Engl. Solving Linear BVPs via Non-commutative Gröbner Bases. *Applicable Analysis*, 82(7), July 2003, pp. 655–675.

M.Rosenkranz. A New Symbolic Method for Solving Linear Two-Point BVPs on the Operator Level. Journal of Symbolic Computation, 39, February 2005, pp.171–199.

Basic Idea and Procedure

Given a two-point BVP (e.g. beam equation):

$$T u = f$$

$$B_1 u = \dots = B_n u = 0$$

■ Example: Beam Deflection

Problem:

$$-u''''(x) + \frac{(\sigma(x)/EI) u(x)}{f(x)} = 0$$

$$u(0) = 0 \qquad u''(0) = 0$$

$$u(1) = 0 \qquad u''(1) = 0$$

Assuming regularity, every linear two-point BVP admits to write the solution as:

$$u(x) = \int_0^1 g(x, \xi) f(\xi) d\xi$$

g Green's function $[a, b]^2 \rightarrow \mathbb{C}$
 f Given forcing function $[a, b] \rightarrow \mathbb{C}$
 u Desired solution function $[a, b] \rightarrow \mathbb{C}$

Traditional solution method (see Kamke): Matrix inversion based on a fundamental system.

In the beam example:

$$g(x, \xi) = \begin{cases} \frac{1}{3} x \xi - \frac{1}{6} \xi^3 - \frac{1}{2} x^2 \xi + \frac{1}{6} x \xi^3 + \frac{1}{6} x^3 \xi & \text{if } 0 \leq \xi \leq x \leq 1 \\ \frac{1}{3} x \xi - \frac{1}{2} x \xi^2 - \frac{1}{6} x^3 + \frac{1}{6} x \xi^3 + \frac{1}{6} x^3 \xi & \text{if } 0 \leq x \leq \xi \leq 1 \end{cases}$$

■

We want to find its Green's **operator G** in the sense of

$$T G = 1 \quad (\text{i.e. } T(G f) = f)$$

$$B_1 G = \dots = B_n G = 0 \quad (\text{i.e. } B_1(G f) = 0)$$

We do the following:

- Compute the solution space N of the homogeneous equation $Tu = 0$.

- Determine a projector P onto N such that $M = (1 - P) C^\infty[a, b]$ fulfills the boundary conditions.
- Find the right inverse T^\diamond of T (a variant of Moore-Penrose inverse).
- Build up $G = (1 - P) T^\diamond$ as the crude Green's operator.
- Reduce G with respect to the [Green's system](#) (a non-commutative Gröbner basis by the main theorem; 233 S-polys needed!) for obtaining a standard representation.
- (Optionally, extract Green's function g from standard representation of G).

⏪ ⏩

30 of 66

The Green's System

```
System["1. Equalities for Isolating Differential Operators", any[f],
      DA = 1                                "DA"
      DB = -1                               "DB"
      -----
      D [f] = [f] D + [f ' ]              "DM"
      -----
      DL = 0                                "DL"
      DR = 0                                "DR"
]
```

```
System["2. Equalities for Isolating Boundary Operators", any[f],
      LA = 0                                "LA"
      RA = A + B                            "RA"
      LB = A + B                            "LB"
      RB = 0                                "RB"
      -----
      L [f] = f^← L                         "LM"
      R [f] = f^→ R                         "RM"
      -----
      LL = L                                "LL"
      LR = R                                "LR"
      RL = L                                "RL"
      RR = R                                "RR"
]
```

```
System["Equalities for Algebraic Simplification", any[f, g],
      [f] [g] = [f g]                      "MM"
]
```

System["3. Equalities for Contracting Integration Operators", any[f],

$$A [f] A = [\int^* f] A - A [\int^* f] \quad \text{"AMA"}$$

$$A [f] B = [\int^* f] B + A [\int^* f] \quad \text{"AMB"}$$

$$B [f] A = [\int_* f] A + B [\int_* f] \quad \text{"BMA"}$$

$$B [f] B = [\int_* f] B - B [\int_* f] \quad \text{"BMB"}$$

$$A A = [\int^* 1] A - A [\int^* 1] \quad \text{"AA"}$$

$$A B = [\int^* 1] B + A [\int^* 1] \quad \text{"AB"}$$

$$B A = [\int_* 1] A + B [\int_* 1] \quad \text{"BA"}$$

$$B B = [\int_* 1] B - B [\int_* 1] \quad \text{"BB"}$$

]

System["4. Equalities for Absorbing Integration Operators", any[f],

$$A [f] D = -f^{\leftarrow} L + [f] - A [f'] \quad \text{"AMD"}$$

$$B [f] D = f^{\rightarrow} R - [f] - B [f'] \quad \text{"BMD"}$$

$$A D = -L + 1 \quad \text{"AD"}$$

$$B D = R - 1 \quad \text{"BD"}$$

$$A [f] L = [\int^* f] L \quad \text{"AML"}$$

$$B [f] L = [\int_* f] L \quad \text{"BML"}$$

$$A [f] R = [\int^* f] R \quad \text{"AMR"}$$

$$B [f] R = [\int_* f] R \quad \text{"BMR"}$$

$$A L = [\int^* 1] L \quad \text{"AL"}$$

$$B L = [\int_* 1] L \quad \text{"BL"}$$

$$A R = [\int^* 1] R \quad \text{"AR"}$$

$$B R = [\int_* 1] R \quad \text{"BR"}$$

]

Current Math Systems

More Interaction Numerics / Symbolics

More Symbolics

More Intellectics

Automated (Dis-) Proving in Geometry

Reduction of the Problem to Gröbner bases computation:

Geo Theorem \longrightarrow (by coordinatization)

$\forall_{x,y,\dots} (\text{poly1}(x,y,\dots)=0 \wedge \dots \Rightarrow \text{poly}(x,y,\dots)=0) \longrightarrow$

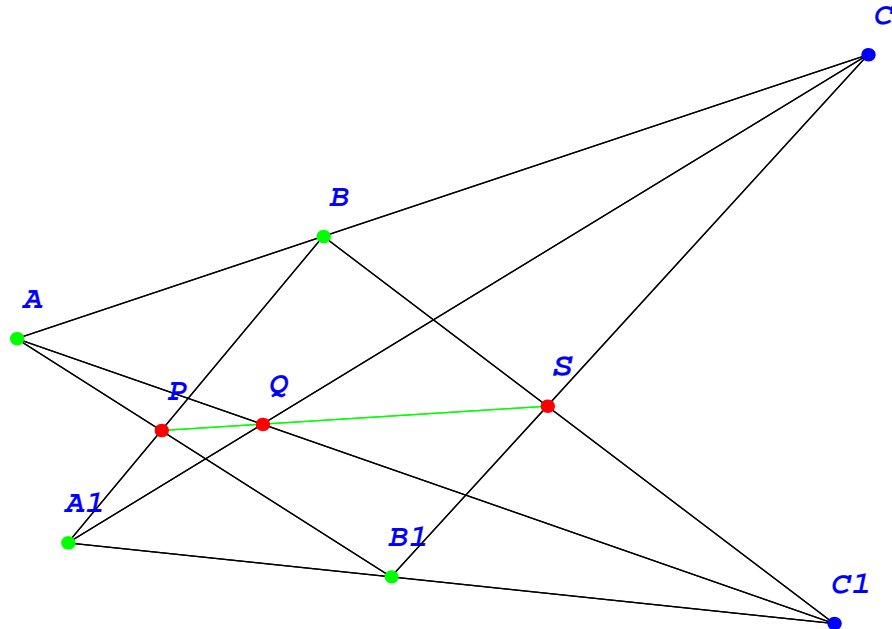
$\neg \exists_{x,y,\dots} (\text{poly1}(x,y,\dots)=0 \wedge \dots \wedge \text{poly}(x,y,\dots) \neq 0) \longrightarrow$

$\neg \exists_{x,y,\dots,a} (\text{poly1}(x,y,\dots)=0 \wedge \dots \wedge a \cdot \text{poly}(x,y,\dots) - 1 = 0)$

The latter question [can be decided by the Gröbner basis method!](#)

Example: Pappus Theorem

- What does the theorem say geometrically?



- Textbook formulation:

Let A, B, C and A_1, B_1, C_1 be on two lines and $P = AB_1 \cap A_1B$, $Q = AC_1 \cap A_1C$, $S = BC_1 \cap B_1C$. 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:

$$(1) \quad \forall_{x_1, x_2, x_3, x_4, x_5, x_6, x_7} (u_3 u_4 + -u_2 u_5 + -u_3 u_7 + u_5 u_7 + u_2 x_1 + -u_4 x_1 = 0 \wedge \\ u_5 x_2 + -u_4 x_3 = 0 \wedge -u_1 u_2 + u_1 x_2 + -u_3 x_2 + u_2 x_3 = 0 \wedge \\ x_1 x_4 + -u_7 x_5 = 0 \wedge -u_2 u_6 + -u_3 x_4 + u_6 x_4 + u_2 x_5 = 0 \wedge \\ u_1 u_7 + -u_1 x_6 + x_1 x_6 + -u_7 x_7 = 0 \wedge -u_4 u_6 + -u_5 x_6 + u_6 x_6 + u_4 x_7 = 0 \Rightarrow \\ x_3 x_4 + -x_2 x_5 + -x_3 x_6 + x_5 x_6 + x_2 x_7 + -x_4 x_7 = 0)$$

This problem is equivalent to:

$$(2) \quad \neg \left(\exists_{x_1, x_2, x_3, x_4, x_5, x_6, x_7} (u_3 u_4 + -u_2 u_5 + -u_3 u_7 + u_5 u_7 + u_2 x_1 + -u_4 x_1 = 0 \wedge \\ u_5 x_2 + -u_4 x_3 = 0 \wedge -u_1 u_2 + u_1 x_2 + -u_3 x_2 + u_2 x_3 = 0 \wedge \\ x_1 x_4 + -u_7 x_5 = 0 \wedge -u_2 u_6 + -u_3 x_4 + u_6 x_4 + u_2 x_5 = 0 \wedge \\ u_1 u_7 + -u_1 x_6 + x_1 x_6 + -u_7 x_7 = 0 \wedge -u_4 u_6 + -u_5 x_6 + u_6 x_6 + u_4 x_7 = 0 \wedge \\ x_3 x_4 + -x_2 x_5 + -x_3 x_6 + x_5 x_6 + x_2 x_7 + -x_4 x_7 \neq 0) \right)$$

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

$$(3) \quad \neg \left(\exists_{x_1, x_2, x_3, x_4, x_5, x_6, x_7, v_0} (u_3 u_4 + -u_2 u_5 + -u_3 u_7 + u_5 u_7 + u_2 x_1 + -u_4 x_1 = 0 \wedge \\ u_5 x_2 + -u_4 x_3 = 0 \wedge -u_1 u_2 + u_1 x_2 + -u_3 x_2 + u_2 x_3 = 0 \wedge \\ x_1 x_4 + -u_7 x_5 = 0 \wedge -u_2 u_6 + -u_3 x_4 + u_6 x_4 + u_2 x_5 = 0 \wedge \\ u_1 u_7 + -u_1 x_6 + x_1 x_6 + -u_7 x_7 = 0 \wedge -u_4 u_6 + -u_5 x_6 + u_6 x_6 + u_4 x_7 = 0 \wedge \\ 1 + -v_0 (x_3 x_4 + -x_2 x_5 + -x_3 x_6 + x_5 x_6 + x_2 x_7 + -x_4 x_7) = 0) \right)$$

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.

⏪ ⏩

34 of 66

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

■ Initialize Theorema

```
Needs["Theorema`"];
```

```
Needs["Theorema`Provers`UserProvers`PCS`"]
```

```
Needs["Theorema`Provers`UserProvers`VariousPC`"]
```

```
Union::normal : Nonatomic expression expected at
position 2 in {HoldPattern := ¬, HoldPattern := ∧, HoldPattern := ∨} U
$TmaSTKnowledgeBase. Mehr...
```

```
Union::normal : Nonatomic expression expected at
position 2 in {HoldPattern := ¬, HoldPattern := ∧, HoldPattern := ∨} U
$TmaSTKnowledgeBase. Mehr...
```

```
SetGlobals[TraceLevel → 0, DebugLevel → 0,
SearchDepth → 100, Prover → PCS, FormatMetas → "Subscripted"]
```

```
SetOptions[Prove, transformBy → ProofSimplifier,
TransformerOptions → {branches → Proved}];
```

```
$RecursionLimit = 1024;
```

```
$TmaSolveLevel = 3;
```

```
$NDDebug = False;
```

```
Off[General::spell]
```

```
Off[General::spell1]
```

```
SetGlobals[LabelStyle → "Numeric", Prover → NNEqIndProver];
Needs["Theorema`Provers`Cascade`Cascade`"];
Needs["Theorema`Provers`Cascade`ConjectureGenerator`"]
```

```
Needs["Theorema`Provers`UserProvers`GroebnerBasesProver`"];
$TmaProofPresentation = 100;
```

```
Theorema`Provers`PCS`Solver`Private`$TmaSolveLevel = 3
```

```
3
```

```
SetOptions[Prove, transformBy → ProofSimplifier,
  TransformerOptions → {branches → Proved}];
$RecursionLimit = 1024; Off[General::spell]; Off[General::spell1];
$TmaQRTarget = {"kb", "goal"};
```

■ Example

```
Definition["limit:", any[f, a],
  limit[f, a] ⇔  $\forall_{\epsilon > 0} \exists_N \forall_{n \geq N} |f[n] - a| < \epsilon$ 
```

```
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!



35 of 66

Algorithm-Supported Mathematical Theory Exploration

A new world-wide movement (approx. 20 research groups, e.g. Mizar, Isabelle, Omega, NuPrL, Coq, etc.)

Our Theorema Group is a (founding) member of this network.

Goals:

- invent (**axioms**, **definitions** for) new concepts (operations: predicates, functions) (e.g. **limit**)
- invent and prove **properties** of notions
- invent **problems** about notions
- invent methods (**algorithms**) for problems and prove their correctness
- compute (**apply** algorithms to data)
- organize, **store**, and retrieve knowledge



36 of 66

Example: Automated Synthesis of the Gröbner Bases Algorithm (BB 2005)

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

by this new algorithm synthesis method,

the key idea of the main theorem (the notion of S-polynomial) is automatically generated and verified.



37 of 66

The Algorithm Invention ("Synthesis") Problem

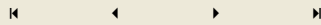
Given a problem specification P (in predicate logic), find an algorithm A such that

$$\forall_x P[x, A[x]].$$

Examples of specifications P :

```
P[x, y] ⇔ is-greater[x, y]
P[x, y] ⇔ is-sorted-version[x, y]
P[x, y] ⇔ has-derivative[x, y]
P[x, y] ⇔ are-factors-of[x, y]
P[x, y] ⇔ is-Gröbner-basis[x, y]
....
```

A general algorithm S for "all" P cannot exist but ...



38 of 66

Literature

There is a rich literature on algorithm synthesis methods, see survey

[Basin et al. 2004] D. Basin, Y. Deville, P. Flener, A. Hamfelt, J. F. Nilsson. Synthesis of Programs in Computational Logic. In: M. Bruynooghe, K. K. Lau (eds.), Program Development in Computational Logic, Lecture Notes in Computer Science, Vol. 3049, Springer, 2004, pp. 30-65.

My method is in the class of "scheme-based" methods. Closest (but essentially different):

[Lau et al. 1999] K. K. Lau, M. Ornaghi, S. Tärnlund. Steadfast logic programs. Journal of Logic Programming, 38/3, 1999, pp. 259-294.

And the work of A. Bundy and his group (U of Edinburgh) on the automated invention of induction schemes.

b



39 of 66

Algorithm Synthesis by "Lazy Thinking" (BB 2002)

"Lazy Thinking" Method for Algorithm Synthesis =

My Advice to "Humans" (or "Computers") How to Invent Algorithms.

Given: A problem P. Find: An algorithm A for P.

- ♣ Learn how to **prove**.
- ♣ Completely understand the problem P. ("**Specification**" of the problem.)
- ♣ Collect (discover, prove) "complete" **knowledge** on the auxiliary notion appearing in the problem P.
- ♣ Consider known fundamental ideas of how to structure algorithms in terms of subalgorithms ("**algorithm schemes A**").
Try one scheme A after the other.
- ♣ For the chosen scheme A, try to prove $\forall_x P[x, A[x]]$: From the **failing proof construct specifications** for the subalgorithms B occurring in A.

How Far Can We Go With the Method ?

Can we automatically synthesize algorithms for **non-trivial problems**? What is "non-trivial"?

Example of a non-trivial problem (?): construction of Gröbner bases.

"Non-trivial": The **invention** of the notion of **S-polynomial** and the characterization of Gröbner-bases by finitely many S-polynomial checks.

With the "Lazy Thinking" method, it is possible to invent the essential idea of the B.B.'s Gröbner bases algorithm fully automatically: See [BB 2005].

The Problem of Constructing Gröbner Bases

Find algorithm `Gb` such that

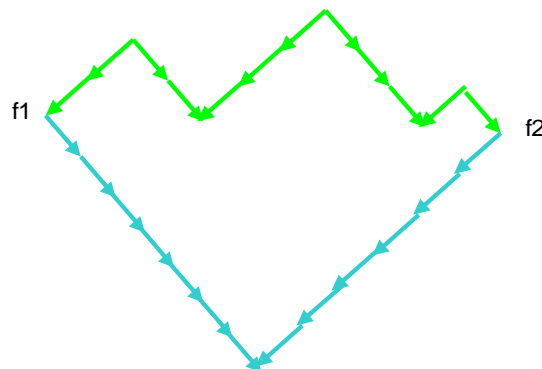
$$\forall \text{ is-finite}[F] \quad \left(\begin{array}{l} \text{is-finite}[\text{Gb}[F]] \\ \text{is-Gröbner-basis}[\text{Gb}[F]] \\ \text{ideal}[F] = \text{ideal}[\text{Gb}[F]]. \end{array} \right)$$

$$\text{is-Gröbner-basis}[G] \Leftrightarrow \text{is-confluent}[\rightarrow_G].$$

\rightarrow_G ... a division step.

Confluence of Division \rightarrow_G

$$\text{is-confluent}[\rightarrow] : \Leftrightarrow \forall_{f_1, f_2} (f_1 \leftrightarrow^* f_2 \Rightarrow f_1 \downarrow^* f_2)$$



Knowledge on the Concepts Involved

$$h1 \rightarrow_G h2 \Rightarrow p . h1 \rightarrow_G p . h2$$

etc.

44 of 66

Algorithm Scheme "Critical Pair / Completion"

$$\begin{aligned}
 A[F] &= A[F, \text{pairs}[F]] \\
 A[F, \langle \rangle] &= F \\
 A[F, \langle \langle g1, g2 \rangle, \bar{p} \rangle] &= \\
 &\text{where } [f = \text{lc}[g1, g2], h1 = \text{trd}[\text{rd}[f, g1], F], h2 = \text{trd}[\text{rd}[f, g2], F], \\
 &\left\{ \begin{array}{l} A[F, \langle \bar{p} \rangle] \qquad \qquad \qquad \leftarrow h1 = h2 \\ A[F - \text{df}[h1, h2], \langle \bar{p} \rangle \times \langle \langle F_k, \text{df}[h1, h2] \rangle_{k=1, \dots, |F|} \rangle] \leftarrow \text{otherwise} \end{array} \right.]
 \end{aligned}$$

This scheme can be tried in any domain, in which we have a reduction operation rd that depends on sets F of objects and a Noetherian relation $>$ which interacts with rd in the following natural way:

$$\forall_{f, g} (f > \text{rd}[f, g]).$$

45 of 66

The Essential Problem

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

$$\forall_F \left(\begin{array}{l} \text{is-finite}[A[F]] \\ \text{is-Gröbner-basis}[A[F]] \\ \text{ideal}[F] = \text{ideal}[A[F]]. \end{array} \right)$$

using the above scheme for A 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.

⏪ ⏩

46 of 66

Now Start the (Automated) Correctness Proof

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

⏪ ⏩

47 of 66

Details

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

$$\forall_{g1, g2 \in G} \left(\text{where} [f = \text{lc}[g1, g2], h1 = \text{trd}[\text{rd}[f, g1], F], \right. \\ \left. h2 = \text{trd}[\text{rd}[f, g2], F], \bigvee \{ h1 = h2 \right. \\ \left. \text{df}[h1, h2] \in G \} \right).$$

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.

⏪ ⏩

48 of 66

Using Available Knowledge

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

$$\text{is-Church-Rosser}[\rightarrow_G] \Leftrightarrow \forall_p \forall_{f_1, f_2} \left(\left(\begin{array}{l} p \rightarrow f_1 \\ p \rightarrow f_2 \end{array} \right) \Rightarrow f_1 \downarrow^* f_2 \right).$$

Newman's lemma (1942):

$$\text{is-Church-Rosser}[\rightarrow] \Leftrightarrow \forall_{f, f_1, f_2} \left(\left(\begin{array}{l} f \rightarrow f_1 \\ f \rightarrow f_2 \end{array} \right) \Rightarrow f_1 \downarrow^* f_2 \right).$$

Definition of "f1 and f2 have a common successor":

$$f_1 \downarrow^* f_2 \Leftrightarrow \exists_g \left(\begin{array}{l} f_1 \rightarrow^* g \\ f_2 \rightarrow^* g \end{array} \right)$$

The (Automated) Proof Attempt

Let now the power product p and the polynomials f_1, f_2 be arbitrary but fixed and assume

$$\begin{cases} p \rightarrow_G f_1 \\ p \rightarrow_G f_2. \end{cases}$$

We have to find a polynomial g such that

$$\begin{cases} f_1 \rightarrow_G^* g, \\ f_2 \rightarrow_G^* g. \end{cases}$$

From the assumption we know that there exist polynomials g_1 and g_2 in G such that

$$\begin{cases} lp[g_1] \mid p, \\ f_1 = rd[p, g_1], \\ lp[g_2] \mid p, \\ f_2 = rd[p, g_2]. \end{cases}$$

From the final situation in the algorithm scheme we know that for these g_1 and g_2

$$\bigvee \begin{cases} h_1 = h_2 \\ df[h_1, h_2] \in G, \end{cases}$$

where

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

⏪ ⏩ ⏴ ⏵

50 of 66

Case h1=h2

```
lc[g1, g2] →g1 rd[lc[g1, g2], g1] →G* trd[rd[lc[g1, g2], g1], G] =
trd[rd[lc[g1, g2], g2], G] ←G* rd[lc[g1, g2], g2] ←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,

```
∀a,q ( a q lc[g1, g2] →g1 a q rd[lc[g1, g2], g1] →G* a q trd[rd[lc[g1, g2], g1], G] =
a q trd[rd[lc[g1, g2], g2], G] ←G* a q rd[lc[g1, g2], g2] ←g2 a q lc[g1, g2] ).
```

Now we are stuck in the proof.

⏪ ⏩ ⏴ ⏵

51 of 66

Now Use the Specification Generation Algorithm

Using the above specification generation rule, we see that we could proceed successfully with the proof if $lc[g1,g2]$ satisfied the following requirement

```
∀p,g1,g2 ( ( ( { lp[g1] | p } ) ⇒ ( ∃a,q ( p = a q lc[g1, g2] ) ) ) ), (lc requirement)
```

With such an lc , we then would have

```
p →g1 rd[p, g1] = a q rd[lc[g1, g2], g1] →G* a q trd[rd[lc[g1, g2], g1], G] =
a q trd[rd[lc[g1, g2], g2], G] ←G* a q rd[lc[g1, g2], g2] = rd[p, g2] ←g2 p
```

and, hence,

```
f1 →G* a q trd[rd[lc[g1, g2], g1], G],
```

$$f2 \rightarrow_g^* a \text{ trd}[\text{rd}[\text{lc}[g1, g2], g1], G],$$

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

« ‹ › »

52 of 66

Summarize the (Automatically Generated) Specifications of the Subalgorithm lc

Using the above specification generation rule, we see that we could proceed successfully with the proof if $lc[g1, g2]$ satisfied the following requirement

$$\forall_{p, g1, g2} \left(\left(\left\{ \begin{array}{l} lp[g1] \mid p \\ lp[g2] \mid p \end{array} \right\} \right) \Rightarrow (lc[g1, g2] \mid p) \right),$$

and the requirements:

$$\begin{array}{l} lp[g1] \mid lc[g1, g2], \\ lp[g2] \mid lc[g1, g2]. \end{array}$$

Now this problem can be attacked independently of any Gröbner bases theory, ideal theory etc.

« ‹ › »

53 of 66

A Suitable lc

$$lc_p[g1, g2] = lcm[lp[g1], lp[g2]]$$

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!

« ‹ › »

54 of 66

Case $h1 \neq h2$

In this case, $df[h1, h2] \in G$:

In this part of the proof we are basically stuck right at the beginning.

We can try to reduce this case to the first case, which would generate the following requirement

$$\forall_{h1, h2} (h1 \downarrow_{\{df[h1, h2]\}} * h2) \text{ (df requirement) .}$$

⏪ ⏩ ⏴ ⏵

55 of 66

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 satisfies (df requirement), namely

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

because, in fact,

$$\forall_{f, g} (f \downarrow_{\{f-g\}} * g) .$$

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

⏪ ⏩ ⏴ ⏵

56 of 66

Conclusions

Intellectics:

= algorithm-supported mathematical theory exploration

= mathematical knowledge management

= "(Anti)bourbakism of the 21st century"

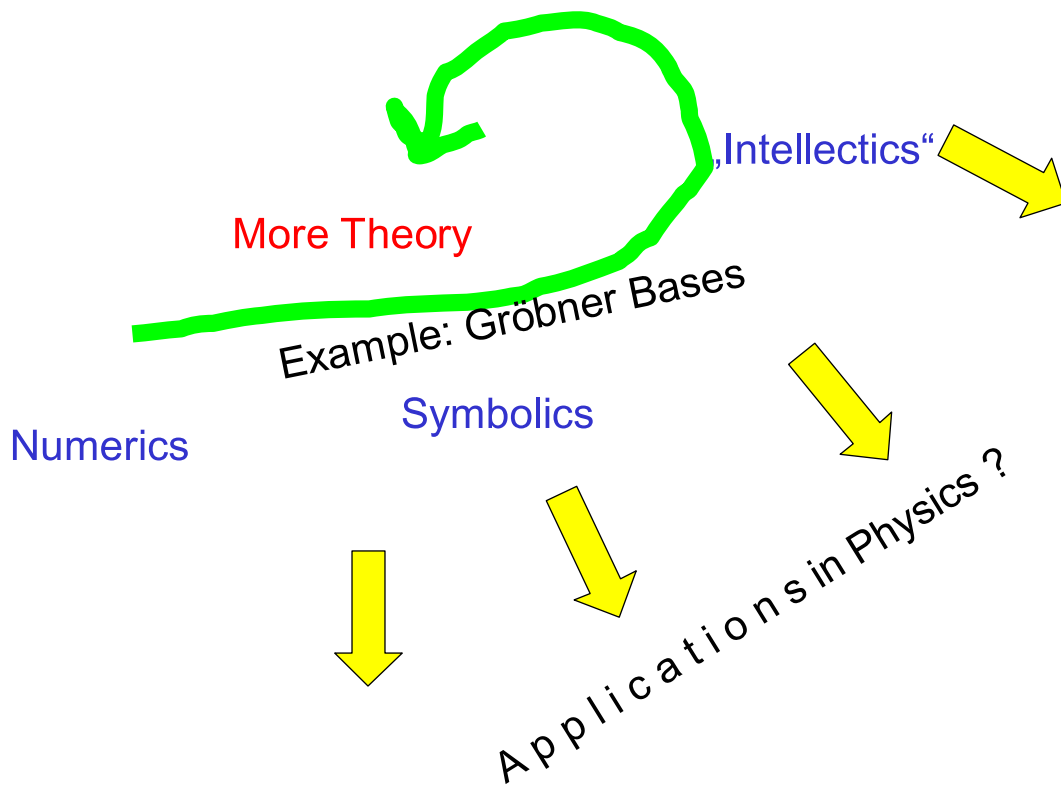
Will drastically change the way

- how we do research in math,
- how we teach math,
- how we apply math,
- how we store and retrieve math knowledge.

⏪ ⏩ ⏴ ⏵

57 of 66

For Physics ?



Special Semester on Gröbner Bases, Feb - July 2006

At RICAM and RISC, Linz, Austria, see

www.ricam.oeaw.ac.at/srs/groeb/

Kick-off: Mo, February 6, 9 h with a Tutorial on Groebner Bases by B.B.

Registration: goto "[expression of interest](#)" form: visiting researcher, postdoc, and doc fellowships available.



◀ ◁ ▷ ▶

59 of 66

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.

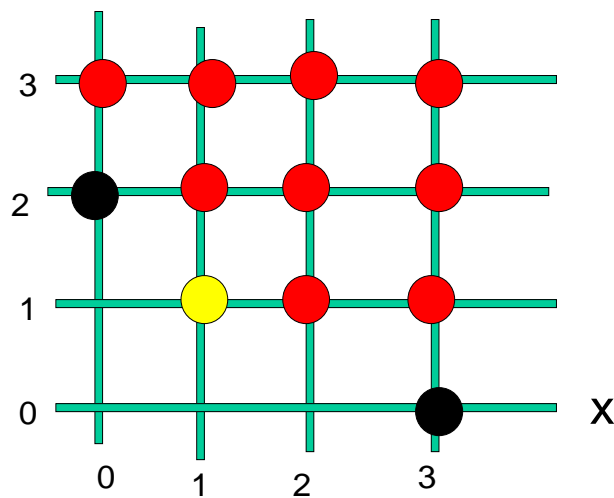
A huge literature.



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".



61 of 66

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 available on the [TI-98](#) (implemented in Derive).



62 of 66

Textbooks on Gröbner Bases

T. Kreuzer, L. Robbiano: *Algorithmic Commutative Algebra I*. Springer, Heidelberg, 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.

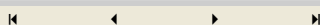
T.Becker, V.Weispfenning. *Gröbner Bases: A Computational Approach to Commutative Algebra*. Springer, New York, 1993.

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.](#)



63 of 66

Gröbner Bases on the Web

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

⏪ ⏩

64 of 66

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).
- ... numerous [applications](#),

⏪ ⏩

65 of 66

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-New York, 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-New York, 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.