

A Crash Course in Gröbner Bases

Bruno Buchberger

Special Semester on Gröbner Bases, RICAM / RISC, Linz, Austria
February 6, 2006

Copyright Bruno Buchberger 2006

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

- The file is kept unchanged and complete including this copyright note.
- A message is sent to bruno.buchberger@jku.at.
- If the material is used, this talk should be cited appropriately.

Gröbner Bases: What and How?

Applications of Gröbner Bases

Discussion

Gröbner Bases: What and How?

Applications of Gröbner Bases

Discussion

Motivation

- Dozens of (difficult) problems turned out to be reducible to the construction of Gröbner bases. (~ 600 papers, 10 textbooks, entry 13P10 in AMS index).
- This is based on the fact that Gröbner bases have many nice properties (e.g. canonicity property, elimination property, syzygy problem).
- For the construction of Gröbner bases we have an algorithm.
- A "beautiful" theory: The notion of Gröbner bases and the algorithm is easy to explain, but correctness is based on a non-trivial theory.

- A very **active research** area: more efficient algorithms based on more theory, and more applications.

The **Special Semester 2006** addresses a few topical areas of Gröbner bases research (theory and applications) in detail, see the various Workshops scheduled.

Earlier, in 1998, we organized a **Special Conference** on Gröbner bases at RISC, see



This talk is based on the paper BB, "Introduction to Gröbner Bases", pp. 3-31, in this book.

The presentation in the paper is more formal.

The Linear Combination of Polynomials

$$\begin{aligned} f_1 &= -2y + xy \\ f_2 &= -x^2 + y^2 \end{aligned}$$

$$-2y + xy$$

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

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

(There are infinitely many "admissible" orderings for Gröbner bases theory that can be characterized by two easy axioms.)

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 .

Definition of Groebner Bases (BB 1965, Gordon 1899)

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. iff

for all $f_1, \dots, f_m \in F$ and all 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.

Example of a Groebner Basis

The following set G (results from F by adding $-2x^2 + x^3$) is a Groebner basis:

$$G = \{-2x^2 + x^3, -2y + xy, -x^2 + y^2\}$$

$$\{-2x^2 + x^3, -2y + xy, -x^2 + y^2\}$$

For example,

$$(1 + 3y)(-2x^2 + x^3) + (8x + 3xy)(-2y + xy) + (2 - x - y^2)(-x^2 + y^2) \quad // \text{Expand}$$

$$-4x^2 + 2x^3 - 16xy + 2x^2y + 3x^3y + 2y^2 - 7xy^2 + 4x^2y^2 - y^4$$

$$(1) (-2x^2 + x^3) + (8x)(-2y + xy) + (y)(-x^2 + y^2) // \text{Expand}$$

$$-2x^2 + x^3 - 16xy + 7x^2y + y^3$$

Why is it **difficult to check** whether a given F is a Groebner basis?

How can we check whether a given F is a Groebner basis?

How can we get an "equivalent" Groebner basis G for a given F (which may not be a Groebner basis)?

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

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

Proof: Nontrivial. Combinatorial. Some details later.

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 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.

S-Polynomials

$$f_1 = -2y + xy$$

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

$$-2y + xy$$

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

$$S\text{-polynomial}[f_1, f_2] = y f_1 - x f_2$$

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

```
s-polynomial[f1, f2] = y f1 - x f2 // Expand
```

```
| x3 - 2 y2
```

The Algorithm 'remainder'

Roughly, remainder[F, g] results from replacing power products in g by a lower products using the polynomials in F until no more replacements are possible.

Example:

Consider, again,

```
f1 = -2 x2 + x3;
f2 = -2 y + x y;
f3 = -x2 + y2;
```

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

and

```
g = x y - 3 x y2;
```

A "reduction" ("division") step on g w.r.t. F:

```
g1 = g + (3 x) f3
```

```
| x y - 3 x y2 + 3 x (-x2 + y2)
```

```
g1 = g + (3 x) f3 // Expand
```

```
| -3 x3 + x y
```

A next division step w.r.t. F:

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

```
| {-2 x2 + x3, -2 y + x y, -x2 + y2}
```

```
g2 = g1 + (-1) f2 // Expand
```

```
| -3 x3 + 2 y
```

A next division step w.r.t. F:

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

```
{-2 x2 + x3, -2 y + x y, -x2 + y2}
```

```
g3 = g2 + (3) f1 // Expand
```

```
-6 x2 + 2 y
```

This is the remainder of the division of g w.r.t. F because ...

Remainder Algorithms are Available in all Math Systems

```
PolynomialReduce[g, F, {y, x}]
```

```
{{0, 1 - 3 y, -6}, -6 x2 + 2 y}
```

Note: the remaindering algorithm can be extended to a "remaindering with co-factors":

```
g - 0 (-2 x2 + x3) - (1 - 3 y) (-2 y + x y) - (-6) (-x2 + y2) // Expand
```

```
-6 x2 + 2 y
```

Now We Can *Check* Groebnerianity

Let's again look to the above example:

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

```
{-2 x2 + x3, -2 y + x y, -x2 + y2}
```

```
PolynomialReduce[f1 y - f2 x2, F, {y, x}]
```

```
{{0, 0, 0}, 0}
```

```
PolynomialReduce[f1 y2 - f3 x3, F, {y, x}]
```

```
{{4 + 2 x + x2, -4 y - 2 x y, -8}, 0}
```

```
PolynomialReduce[f2 y - f3 x, F, {y, x}]
```

```
{{1, 0, -2}, 0}
```

The Problem of *Constructing* Gröbner Bases

Given F , find G s.t. $\text{Ideal}(F) = \text{Ideal}(G)$ and G is a Gröbner basis.

($\text{Ideal}(F)$:= the set of all linear combinations $h_1 f_1 + \dots + h_m f_m$

with $f_1, \dots, f_m \in F$ and h_1, \dots, h_m arbitrary polynomials.)

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

Recall the main theorem:

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

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.

The algorithm allows [many refinements and variants](#) which, however, are all based on the notion of [S-polynomial](#) and variants of the main theorem.

Correctness and Termination of the Algorithm

Correctness: Easy as soon as we know the main theorem.

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

A sequence p_1, p_2, \dots of power products with the property that, for all $i < j$, p_i does not divide p_j , must be finite.

Specializations

The [Gröbner bases algorithm](#),

for linear polynomials, specializes to [Gauss' algorithm](#), and

for univariate polynomials, specializes to [Euclid's algorithm](#).

Example

Let's again look at

$$\begin{aligned} f_1 &= -2y + xy \\ f_2 &= -x^2 + y^2 \end{aligned}$$

$$-2y + xy$$

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

$$F = \{f_1, f_2\}$$

$$\{-2y + xy, -x^2 + y^2\}$$

F is not a Groebner basis.

The S-polynomial of f_1, f_2 :

```
S-polynomial[f1, f2] = y f1 - x f2 // Expand
```

```
x3 - 2 y2
```

Its remainder w.r.t. F is:

```
-2 x2 + x3.
```

All the other S-polynomials have remainder 0. Hence, we arrived at a Groebner basis.

The Groebner basis algorithm is now available in all math software systems, e.g. in *Mathematica*:

```
G = GroebnerBasis[F, {y, x}]
```

```
{-2 x2 + x3, -2 y + x y, -x2 + y2}
```

Reduced Gröbner Bases

A set F of [polynomials](#) is called a [reduced Gröbner basis](#) (w.r.t. the chosen ordering of power products) iff

F is a Gröbner bases and,

for all $f \in F$,

$\text{remainder}[F - \{f\}, f] = f$ and

f is monic.

Algorithm for obtaining a reduced Gröbner basis: Compute a Gröbner basis and then "auto-reduce" the basis.

Extended Gröbner Basis Algorithm

Keeps track of how the polynomials in the Gröbner basis G can be linearly combined from the polynomials in F.

Gröbner Bases: What and How?

Applications of Gröbner Bases

Discussion

Applications are Based on Three Main Properties of Gröbner Bases

Canonicity Property

Elimination Property

Syzygy Property

Canonicity

Remaindering modulo a Gröbner basis F is a "canonical simplifier" for congruence modulo F :

$$f \equiv_F g \iff \text{remainder}[F, f] = \text{remainder}[F, g]$$

$$f \equiv_F \text{remainder}[F, f]$$

"Second order" canonicity: "Reduced Gröbner basis" is a "canonical simplifier" for "have same congruence":

$$\text{Ideal}[F] = \text{Ideal}[G] \iff \text{reduced-Gröbner-basis}[F] = \text{reduced-Gröbner-basis}[G]$$

$$\text{Ideal}[F] = \text{Ideal}[\text{reduced-Gröbner-basis}[F]].$$

Elimination Ideals

Let $<$ be the lexicographic ordering defined by $x_1 < x_2 < \dots < x_n$. If F is a Gröbner basis w.r.t. $<$, then

If F is a Gröbner basis w.r.t. $<$ then, for all $i \leq n$,

$$\text{Ideal}[F] \cap K[x_1, \dots, x_i] = \text{Ideal}[F \cap K[x_1, \dots, x_i]]$$

The "elimination ideals" of an ideal can be easily computed if we have a Gröbner basis for the ideal.

Syzygy Property (Linear Syzygies)

Given a tuple $\langle f_1, \dots, f_m \rangle$ of polynomials. How can we obtain a finite basis for the set of all possible polynomial solutions ("syzygies") $\langle h_1, \dots, h_m \rangle$ of the linear diophantine equation

$$h_1 \cdot f_1 + \dots + h_m \cdot f_m = 0 \quad ?$$

In the case that $F := \{ f_1, \dots, f_m \}$ is a Gröbner basis the following set of tuples is a finite basis for the infinite set of all syzygies:

consider all pairs f_i, f_j :

$$m := \text{LCM}[\text{LPP}[f_i], \text{LPP}[f_j], \quad u_i := m / \text{LPP}[f_i], \quad u_j := m / \text{LPP}[f_j]$$

$\langle H_1, \dots, H_m \rangle :=$ the cofactors obtained by remaindering

S-polynomial[f_i, f_j] modulo F

$$\langle h_1, \dots, h_i, \dots, h_j, \dots, h_m \rangle := \langle -H_1, \dots, u_i - H_i, \dots, -u_j - H_j, \dots, -H_m \rangle$$

Summarizing: the **S-polynomials** give a handle for obtaining a finite basis for the set of all **syzygies**!

(The **inhomogeneous** equation

$$h_1 \cdot f_1 + \dots + h_m \cdot f_m = g$$

can be solved by finding one solution of the inhomogeneous equation and adding the solutions of the homogeneous equations.)

(In case $\{ f_1, \dots, f_m \}$ is not a Gröbner basis, transform to Gröbner basis by the **extended Gröbner basis algorithm**, solve, and transform solutions back.)

(The case of **several linear diophantine equations** with polynomial coefficients can be reduced to the case of one equation. Alternatively, the entire Gröbner bases approach can be formulated for polynomial "**modules**" instead of polynomial rings.)

Application: Solving Polynomial Systems

Is based on the [elimination property](#) of Gröbner bases (w.r.t. lexicographic orderings).

A Simple System of Equations

$$\begin{aligned} f_1 &= -2y + xy \\ f_2 &= -x^2 + y^2 \end{aligned}$$

$$-2y + xy$$

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

Find x, y such that

$$\begin{aligned} -2y + xy &= 0 \\ -x^2 + y^2 &= 0 \end{aligned}$$

We compute

```
G = GroebnerBasis[F, {y, x}]
```

```
{-2 x^2 + x^3, -2 y + x y, -x^2 + y^2}
```

```
Solve[-2 x^2 + x^3 == 0, x]
```

```
{{x -> 0}, {x -> 0}, {x -> 2}}
```

```
{G[[2]], G[[3]]} /. {x -> 2}
```

```
{0, -4 + y^2}
```

```
Solve[-4 + y^2 == 0, y]
```

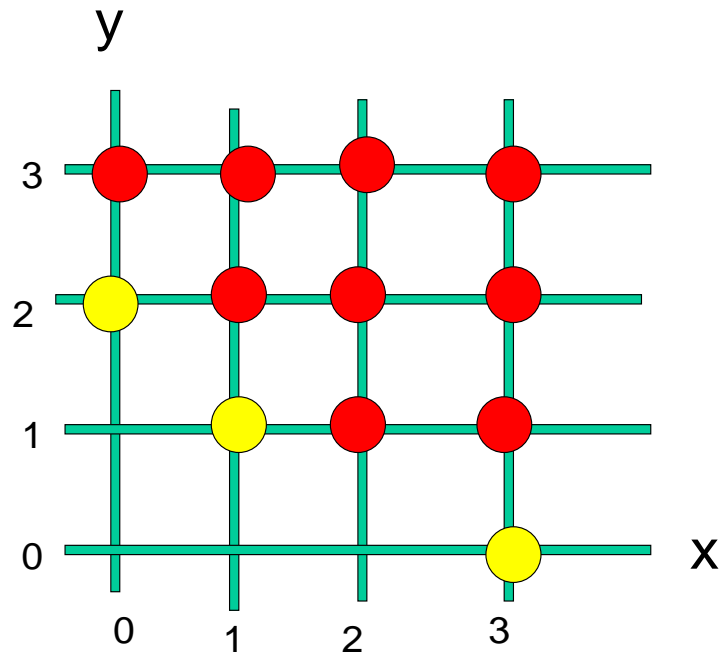
```
{{y -> -2}, {y -> 2}}
```

All this is already implemented in the *Mathematica* general Solve function:

```
Solve[{f1 == 0, f2 == 0}, {x, y}]
```

```
{{x -> 0, y -> 0}, {x -> 0, y -> 0}, {x -> 2, y -> -2}, {x -> 2, y -> 2}}
```

Theorem (BB): Solvability and the number of solutions can be predicted from the form of the Groebner basis.



A More Complicated System of Equations

```
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. 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}}
```

```
zsolexact1 = Solve[G[[1]] == 0, z]
```

```
{ {z -> 0},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 1]},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 2]},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 3]},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 4]},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 5]},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 6]},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 7]},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 8]},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 9]},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 10]},
  {z -> Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 11]} }
```

One can compute with these "abstract roots" like with "square roots". For example

```
G0 = G /. zsolexact1[[2]] // Simplify
```

```
{Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 1] + 4 Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 1]^3 - 17 Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 1]^4 + 3 Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 1]^5 - 45 Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 1]^6 + 60 Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 1]^7 - 29 Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 1]^8 + 124 Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 1]^9 - 48 Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 - 124 #1^8 + 48 #1^9 - 64 #1^10 + 64 #1^11 &, 1]^10 + 64 Root[-1 - 4 #1^2 + 17 #1^3 - 3 #1^4 + 45 #1^5 - 60 #1^6 + 29 #1^7 -
```

$$\begin{aligned}
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^{11} - \\
64 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^{12}, \\
-22001 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1] + \\
14361 y \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1] + \\
16681 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^2 + \\
26380 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^3 + \\
226657 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^4 + \\
11085 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^5 - \\
90346 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^6 - \\
472018 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^7 - \\
520424 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^8 - \\
139296 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^9 - \\
150784 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^{10} + \\
490368 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^{11}, \\
43083 y^2 - 11821 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + \\
& 29 \#1^7 - 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1] + \\
267025 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^2 - \\
583085 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^3 + \\
663460 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^4 - \\
2288350 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^5 + \\
2466820 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^6 - \\
3008257 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^7 + \\
4611948 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^8 - \\
2592304 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^9 + \\
2672704 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 - \\
& 124 \#1^8 + 48 \#1^9 - 64 \#1^{10} + 64 \#1^{11} \&, 1]^{10} - \\
1686848 \text{Root}[-1 - 4 \#1^2 + 17 \#1^3 - 3 \#1^4 + 45 \#1^5 - 60 \#1^6 + 29 \#1^7 -
\end{aligned}$$

```

124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 11,
43083 x - 118717 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 +
29 #17 - 124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] +
69484 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 + 29 #17 -
124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 2 +
402334 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 + 29 #17 -
124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 3 +
409939 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 + 29 #17 -
124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 4 +
1202033 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 + 29 #17 -
124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 5 -
2475608 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 + 29 #17 -
124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 6 +
354746 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 + 29 #17 -
124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 7 -
6049080 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 + 29 #17 -
124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 8 +
2269472 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 + 29 #17 -
124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 9 -
3106688 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 + 29 #17 -
124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 10 +
3442816 Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 - 60 #16 + 29 #17 -
124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] 11}

```

```

G0 /. {Root[-1 - 4 #12 + 17 #13 - 3 #14 + 45 #15 -
60 #16 + 29 #17 - 124 #18 + 48 #19 - 64 #110 + 64 #111 &, 1] -> α}

```

```

{α + 4 α3 - 17 α4 + 3 α5 - 45 α6 + 60 α7 - 29 α8 + 124 α9 - 48 α10 + 64 α11 - 64 α12,
-22001 α + 14361 y α + 16681 α2 + 26380 α3 + 226657 α4 + 11085 α5 -
90346 α6 - 472018 α7 - 520424 α8 - 139296 α9 - 150784 α10 + 490368 α11,
43083 y2 - 11821 α + 267025 α2 - 583085 α3 + 663460 α4 - 2288350 α5 +
2466820 α6 - 3008257 α7 + 4611948 α8 - 2592304 α9 + 2672704 α10 - 1686848 α11,
43083 x - 118717 α + 69484 α2 + 402334 α3 + 409939 α4 + 1202033 α5 -
2475608 α6 + 354746 α7 - 6049080 α8 + 2269472 α9 - 3106688 α10 + 3442816 α11}

```

No more details in this talk!

Alternatively, compute numerically:

```

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]]
```

```
{1.33227 × 10-15 + 9.71445 × 10-17 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}
```

```
PolynomialGCD[Gsubnum[[2]], Gsubnum[[3]]]
```

```
1
```

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

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

```
ysol = NSolve[ Gsubnum[[3]] == 0, y]
```

```
{{y → -0.473535 - 0.205184 i}, {y → 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 → 0.378611 - 0.385558 i}}
```

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

```
{-1.88738 × 10-15 + 2.35922 × 10-16 i,  
1.02696 × 10-15 + 1.52656 × 10-16 i, -1.94289 × 10-15 - 1.11022 × 10-16 i}
```

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 \mathbb{Z}_4 , i.e. a set of generators for the ring

$$\{f \in \mathbb{C}[x_1, x_2] \mid f(x_1, x_2) = f(-x_2, x_1)\},$$

i.e.

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

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

and all invariants can be expressed as polynomials in these invariants.

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, 3/4 i1 x2 - 1/2 x1^2 x2 + x2^3/2, i1 - x1^2/2 + 3 x2^2/4,
 3/2 i1 x1 + 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]
```

$$\left\{ \left\{ 0, \frac{i_1 x_1}{2} - i_1 x_2 - x_1^2 x_2, 0, \frac{3 i_1}{4} - \frac{x_1^2}{2} + \frac{x_2^2}{2}, \right. \right. \\ \left. \left. - \frac{x_1}{4} + \frac{3 x_2}{4}, \frac{3 i_1}{4} + x_1 x_2, \frac{x_2^3}{2}, -\frac{1}{4} i_1^2 x_1 - \frac{1}{2} i_1 x_1 x_2^2 - x_1 x_2^4 \right\}, \right. \\ \left. -i_1^3 x_1 + 2 i_1 i_2 x_1 + \frac{1}{2} i_1 i_3 x_1 + i_1^2 x_1^3 - i_2 x_1^3 + \frac{1}{2} i_3 x_1^3 + \frac{1}{2} i_1 i_2 x_2 \right\}$$

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

```
x1^6 x2 - x1 x2^6 /. {x1 -> -x2, x2 -> x1}
```

$$x_1^6 x_2 + x_1 x_2^6$$

Application: Automated (Dis-) Proving in Geometry

Reduction of the Problem to Gröbner bases computation:

Geo Theorem \longrightarrow (by coordinatization)

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

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

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

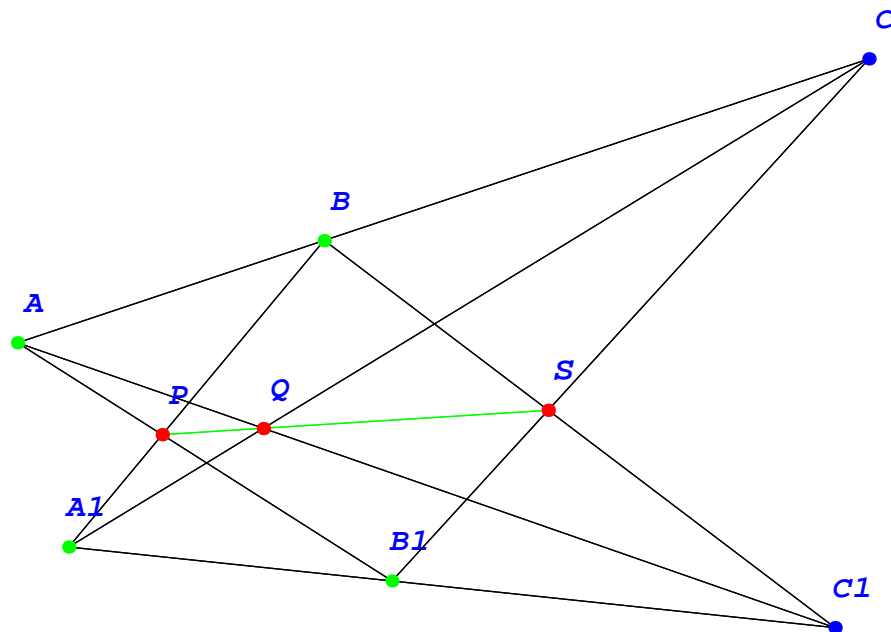
The latter question [can be decided by the Gröbner basis method!](#)

The method is implemented in the [Theorema System](#):

B. B., C. Dupre, T. Jebelean, F. Kriftner, K. Nakagawa, D. Vasaru, W. Windsteiger. The Theorema Project: A Progress Report. In: 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), M. Kerber and M. Kohlhasse (eds.), A.K. Peters, Natick, Massachusetts, ISBN 1-56881-145-4, pp. 98-113.

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}]
```

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

Prove:

(Proposition (Pappus))

$$\forall_{A, B, A_1, B_1, C, C_1, P, Q, S} (\text{point}[A, B, A_1, B_1] \wedge \text{pon}[C, \text{line}[A, B]] \wedge \text{pon}[C_1, \text{line}[A_1, B_1]] \wedge \text{inter}[P, \text{line}[A, B_1], \text{line}[A_1, B]] \wedge \text{inter}[Q, \text{line}[A, C_1], \text{line}[A_1, C]] \wedge \text{inter}[S, \text{line}[B, C_1], \text{line}[B_1, C]] \Rightarrow \text{collinear}[P, Q, S])$$

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 choose 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.

Example: Butterfly Theorem

- Textbook formulation:

A, B, C and D are four points on circle (O) . E is the intersection of AC and BD . Through E draw a line perpendicular to OE , meeting AD at F and BC at G . Show that $\overline{FE} = \overline{GE}$.

- Remark:

Using the default coordinate system the algebraic methods are very slow.

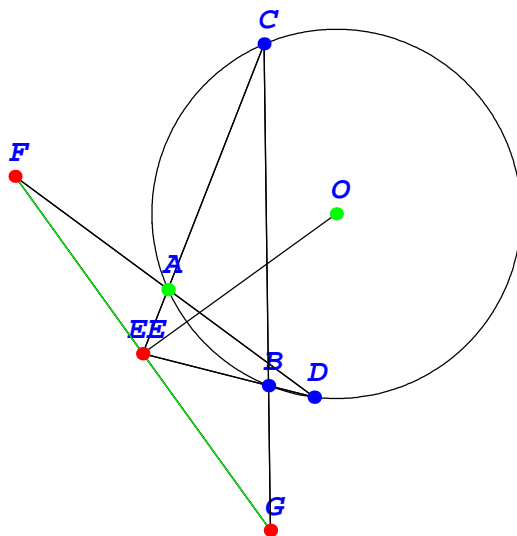
- Input to the system:

```
Proposition["Butterfly", any[O, A, B, C, D, EE, F, G], point[O, A] ^
  pon[C, circle[O, A]] ^
  pon[B, circle[O, A]] ^
  pon[D, circle[O, A]] ^
  inter[EE, line[A, C], line[B, D]] ^
  inter[F, line[D, A], tline[EE, EE, O]] ^ inter[G, line[B, C], line[EE, F]]
  => distequal[EE, F, G, EE]
```

- Input to the system:

```
Simplify[Proposition["Butterfly"], by -> GraphicSimplifier]
```

- Output generated automatically by the system



$|EEF| \cong |GEE|$ for this configuration of the points

- End of output

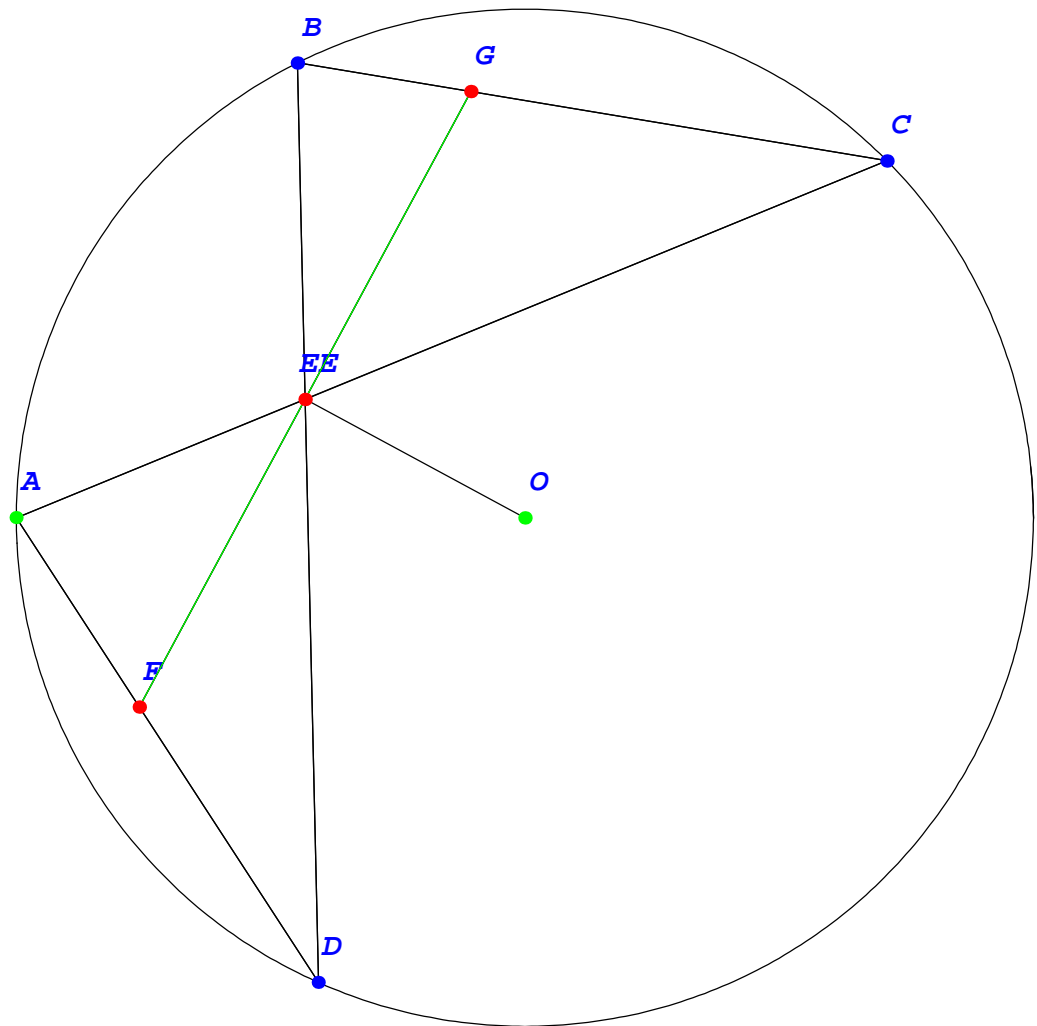
- Input to the system:

```
KnowledgeBase["Butterfly_g", any[O, A, B, C, D],
  {{O, {100, 100}}, {A, {0, 100}}, {B, {55, 190}}, {C, {171, 170}}, {D, {60, 10}}}]
```

- Input to the system:

```
Simplify[Proposition["Butterfly"],
  by -> GraphicSimplifier, using -> KnowledgeBase["Butterfly_g"]]
```

- Output generated automatically by the system



$|EEF| \cong |GEE|$ for this configuration of the points

- End of output
- Input to the system:

```
Prove[Proposition["Butterfly"], by → GeometryProver,
  ProverOptions → {Method → "GroebnerProver", Refutation → False}]
```

- Inserted notebook generated automatically by the system

Prove:

(Proposition (Butterfly))

$$\forall_{O,A,B,C,D,EE,F,G} (\text{point}[O, A] \wedge \text{pon}[C, \text{circle}[O, A]] \wedge \text{pon}[B, \text{circle}[O, A]] \wedge \text{pon}[D, \text{circle}[O, A]] \wedge \text{inter}[EE, \text{line}[A, C], \text{line}[B, D]] \wedge \text{inter}[F, \text{line}[D, A], \text{tline}[EE, EE, O]] \wedge \text{inter}[G, \text{line}[B, C], \text{line}[EE, F]] \Rightarrow \text{distequal}[EE, F, G, EE])$$

with no assumptions.

To prove the above statement we shall use the Gröbner bases 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 EE , and points $\{O\}$ and $\{F, G\}$ on the two axes.

Using this coordinate system we have the following points:

$$\{\{EE, 0, 0\}, \{O, u_1, 0\}, \{A, u_2, u_3\}, \{B, u_4, x_1\}, \\ \{F, 0, x_2\}, \{G, 0, x_3\}, \{C, x_4, x_5\}, \{D, x_6, x_7\}\}$$

The algebraic form of the given construction is:

(1)

$$\forall v_1, v_2, x_1, x_2, x_3, x_4, x_5, x_6, x_7 \quad ((-1) + u_4 v_2 + -v_2 x_6 == 0 \wedge \\ (-1) + u_2 v_1 + -v_1 x_4 == 0 \wedge 2 u_1 u_2 + -u_2^2 + -u_3^2 + -2 u_1 x_4 + x_4^2 + x_5^2 == 0 \wedge \\ 2 u_1 u_2 + -u_2^2 + -u_3^2 + -2 u_1 u_4 + u_4^2 + x_1^2 == 0 \wedge \\ 2 u_1 u_2 + -u_2^2 + -u_3^2 + -2 u_1 x_6 + x_6^2 + x_7^2 == 0 \wedge u_3 x_4 + -u_2 x_5 == 0 \wedge x_1 x_6 + -u_4 x_7 == 0 \wedge \\ -u_2 x_2 + -u_3 x_6 + x_2 x_6 + u_2 x_7 == 0 \wedge u_4 x_3 + x_1 x_4 + -x_3 x_4 + -u_4 x_5 == 0 \Rightarrow x_2^2 + -x_3^2 == 0)$$

We have to compute the Gröbner bases of the hypothesis polynomials (GB).

The polynomials of the Gröbner bases are:

$$\{-u_2^2 + -u_3^2 + (-2 u_1 u_2^2 + 2 u_2^3 + 2 u_2 u_3^2) v_1, \\ -2 u_1 u_2 + u_2^2 + u_3^2 + 2 u_1 u_4 + (4 u_1 u_2 u_4 + -2 u_2^2 u_4 + -2 u_3^2 u_4 + -2 u_1 u_4^2) v_2, \\ -2 u_1 u_2 u_3 u_4 + u_2^2 u_3 u_4 + u_3^3 u_4 + (2 u_1 u_2^2 + -u_2^3 + -u_2 u_3^2) x_1 + \\ (2 u_1 u_2^2 + -u_2^3 + -u_2 u_3^2 + -u_2^2 u_4 + -u_3^2 u_4) x_2, 2 u_1 u_2 u_3 u_4 + -u_2^2 u_3 u_4 + -u_3^3 u_4 + \\ (-2 u_1 u_2^2 + u_2^3 + u_2 u_3^2) x_1 + (2 u_1 u_2^2 + -u_2^3 + -u_2 u_3^2 + -u_2^2 u_4 + -u_3^2 u_4) x_3, \\ -2 u_1 u_2^2 + u_2^3 + u_2 u_3^2 + (u_2^2 + u_3^2) x_4, 2 u_1 u_2 u_3 + -u_2^2 u_3 + -u_3^3 + (-u_2^2 + -u_3^2) x_5, \\ 2 u_1 u_2 u_4 + -u_2^2 u_4 + -u_3^2 u_4 + (2 u_1 u_2 + -u_2^2 + -u_3^2 + -2 u_1 u_4) x_6, \\ (-2 u_1 u_2 + u_2^2 + u_3^2) x_1 + (-2 u_1 u_2 + u_2^2 + u_3^2 + 2 u_1 u_4) x_7, \\ 2 u_1 u_2 + -u_2^2 + -u_3^2 + -2 u_1 u_4 + u_4^2 + x_1^2\}$$

We compute the normal form of the conclusion polynomial modulo GB.

The normal form is: 0

As the obtained normal form equals to 0 the statement is generically true.

Statistics:

The length of the Gröbner bases is 9

Time needed to compute the Gröbner bases: 0.27 Seconds.

Time needed to compute the normal form of the conclusion polynomial modulo GB: 0.32 Seconds.

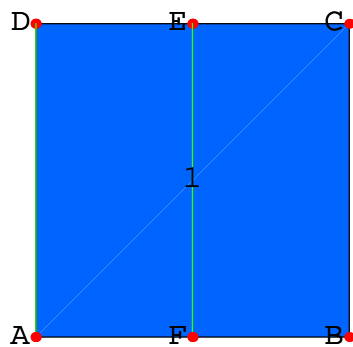
□

- End of inserted notebook

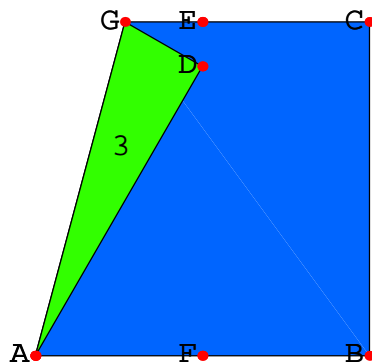
Example: The Correctness of Origami Constructions (jointly with T. Ida)

Example of an Origami Construction Problem: Starting from a square A, B, C, D , find a sequence of origami steps such that, finally, we arrive at an equilateral triangle.

A first solution (does not yield maximum edge length; other solution and more complicated examples see T. Ida, BB, J. Robu et al. 2003 and 2004):



Then we fix the point A and fold so that point D will lie on line EF . (This is a legal origami operation.)



Now we can do the analogous step with corner C , fixing B and bringing C onto the current position of D .

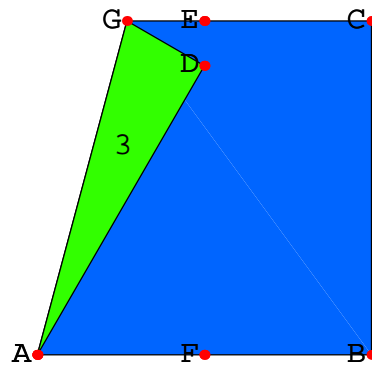
An Origami Proof Problem: Prove that, for all squares $ABCD$, $\overline{GD} = 2 \overline{ED}$.

The Translation into a Prove Problem on Equalities:

First, note that $\overline{AB} = \overline{BC} = \overline{CD} = \overline{DA}$, since we start from a square. Hence, whenever the length of one of these four edges occurs, we replace it by \overline{AB} .

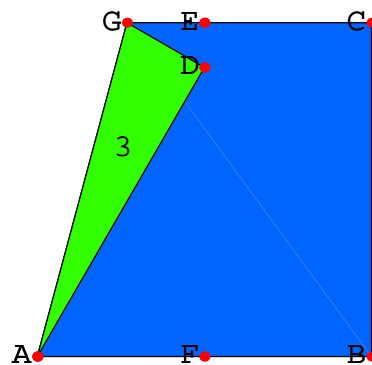
Now observe that

$$\overline{DF}^2 = \overline{AD}^2 - \overline{AF}^2 = \overline{AB}^2 - (\overline{AB} / 2)^2 = 3 / 4 \overline{AB}^2.$$



and

$$\overline{GD}^2 = \overline{GE}^2 + \overline{ED}^2 = (\overline{DC} / 2 - \overline{GD})^2 + (\overline{EF} - \overline{DF})^2 = (\overline{AB} / 2 - \overline{GD})^2 + (\overline{AB} - \overline{DF})^2.$$



We want to decide whether, under these assumptions,

$$\overline{GD} = 2 \overline{ED} = 2(\overline{EF} - \overline{DF}) = 2(\overline{AB} - \overline{DF}).$$

For abbreviation, let's write

$$a = \overline{AB}, \quad b = \overline{GD}, \quad f = \overline{DF}.$$

Then, what we want to prove is that

$$\forall_{a \neq 0, f, b} \left(\left\{ \begin{array}{l} f^2 = 3/4 a^2 \\ b^2 = (a/2 - b)^2 + (a - f)^2 \end{array} \right\} \Rightarrow (b = 2(a - f)) \right)$$

The Transformation to a Groebner Basis Construction Problem:

This is equivalent to

$$\exists_{a, f, b} \left(\left\{ \begin{array}{l} a \neq 0 \\ f^2 = 3/4 a^2 \\ b^2 = (a/2 - b)^2 + (a - f)^2 \\ b \neq 2(a - f) \end{array} \right\} \right),$$

which is equivalent to

$$\exists_{a, f, b, \xi, \eta} \left(\left\{ \begin{array}{l} a \eta = 1 \\ f^2 = 3/4 a^2 \\ b^2 = (a/2 - b)^2 + (a - f)^2 \\ (b - 2(a - f)) \xi = 1 \end{array} \right\} \right),$$

This question can be decided by computing the (reduced) Gröbner basis

```
J = GroebnerBasis[
  {a η - 1, f^2 - 3/4 a^2, b^2 - (a/2 - b)^2 - (a - f)^2, (b - 2(a - f)) ξ - 1}, {ξ, η, b, f, a}]
{1}
```

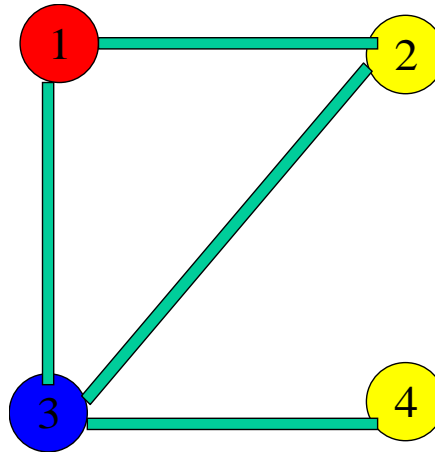
and to check whether or not this Gröbner basis is equal to {1}. Since this is the case, we know that the restricted version of the theorem is true.

Application: Graph Coloring

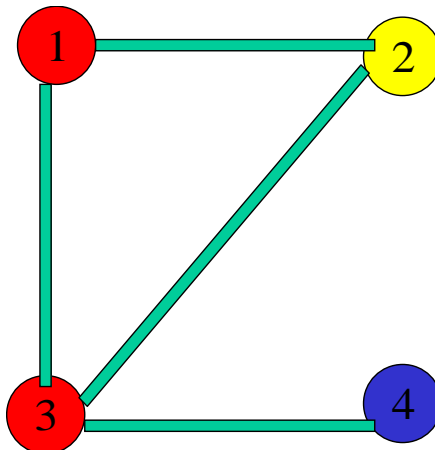
The Problem:

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:

$$(x_1^2 + x_1 x_2 + x_2^2) (x_1 - x_2) \quad // \text{Expand}$$

$$x_1^3 - x_2^3$$

```
{ -1 + x1^3, ... at vertex 1 color is a 3 - ary root of 1
  -1 + x2^3, ... at vertex 2 color is a 3 - ary root of 1
  -1 + x3^3,
  -1 + x4^3,
  x1^2 + x1 x2 + x2^2, ... the colors at 1 and 2 must be different,
  x1^2 + x1 x3 + x3^2,
  x2^2 + x2 x3 + x3^2,
  x3^2 + x3 x4 + x4^2 }
```

Solution by Groebner Bases

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

```
GB = GroebnerBasis[{-1 + x1^3, -1 + x2^3, -1 + x3^3, -1 + x4^3,
  x1^2 + x1 x2 + x2^2, x1^2 + x1 x3 + x3^2, x2^2 + x2 x3 + x3^2, x3^2 + x3 x4 + x4^2},
  {x4, x3, x2, x1}]
```

```
{ -1 + x1^3, x1^2 + x1 x2 + x2^2, -x1 - x2 - x3, -x1 x2 + x1 x4 + x2 x4 - x4^2 }
```

```
Solve[{-1 + x1^3 == 0, -1 + x2^3 == 0, -1 + x3^3 == 0, -1 + x4^3 == 0,
  x1^2 + x1 x2 + x2^2 == 0, x1^2 + x1 x3 + x3^2 == 0, x2^2 + x2 x3 + x3^2 == 0, x3^2 + x3 x4 + x4^2 == 0},
  {x4, x3, x2, x1}]
```

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

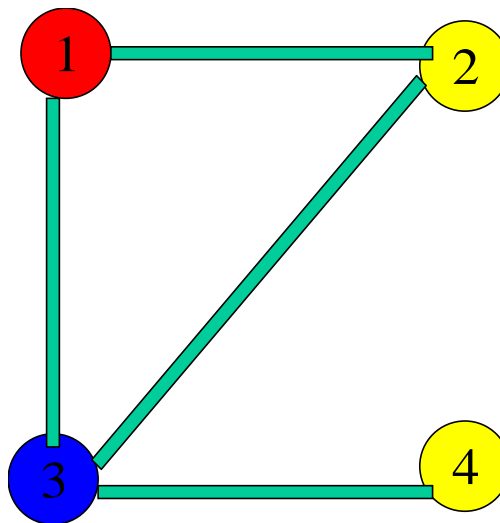
Slightly re-organized output:

```

{ {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 = {P5 - N, P10 - D, P25 - Q}
```

```
{-N + P5, -D + P10, P25 - Q}
```

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

```
G = GroebnerBasis[F, MonomialOrder → DegreeLexicographic]
```

```
{-D + N2, -D3 + N Q, D2 N - Q, -N + P5}
```

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:](#)

```
PolynomialReduce[P17 N10 D5, G, , MonomialOrder → DegreeLexicographic]
```

```
{ {D9 P17 + D8 N2 P17 + D7 N4 P17 + D6 N6 P17 + D5 N8 P17 + D4 P17 Q2 + P7 Q4,  
-D7 P17 - D4 N P17 Q - D2 P17 Q2, P17 Q3, D P2 Q4 + N P7 Q4 + P12 Q4}, D N P2 Q4}
```

[Answer:](#) take 4 quarters, 1 dime, 1 nickel, 2 pennies.

Application: Symbolic Solution of Boundary Value Problems (Differential Equations)

See work by Markus Rosenkranz, PhD thesis 2003 and RISC / RICAM Project "Scientific Computing".

Will be presented in Workshop D2.

Other Applications

Algebraic Geometry

Coding Theory

Cryptography

Invariant Theory

Integer Optimization

Statistics

Symbolic Integration

Symbolic Summation

Systems Theory

Gröbner Bases: What and How?

Applications of Gröbner Bases

Discussion

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.
- The "numerics" approach.

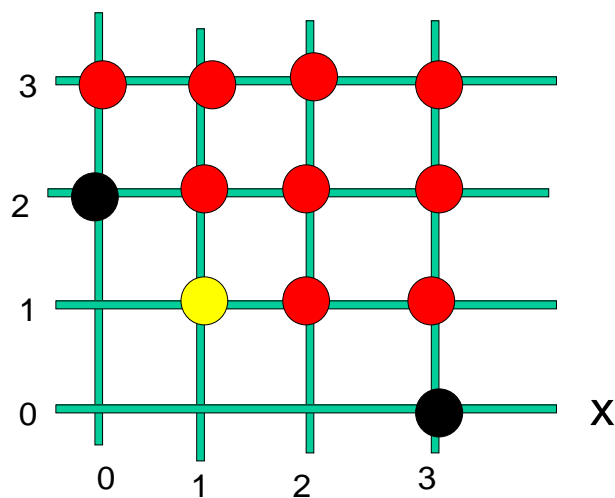
Tuning of the algorithm:

- 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, will be reprinted in English in the JSC 2006) and journal publication (1970, re-printed in BB, F. Winkler, 1998) 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.

Also [various details of the algorithm](#) are discussed, which later have been forgotten:

- * store intermediate polynomials
- * presentation of power products by numbers
- * reduction is linear algebra

(In the early days of computing, as a tendency, one tried to save memory and compensate by time.)

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", see Workshop C "Formal Gröbner Bases Theory".

More Info on Gröbner Bases?

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.

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).

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.](#)

[See papers data base and books in the secretary's office.](#)

Gröbner Bases on the Web

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

Original Publications on Gröbner Bases

[See papers data base.](#)

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).
- The [numerics](#) of Gröbner bases computation.
- ... numerous [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
- new applications

Appendix: Sketch of the Proof of the Main Theorem

Details see BB, Introduction to Gröbner Bases, in BB, FW 1998, pp. 1 - 31.

Equivalent definition of Gröbner bases:

F is a Gröbner basis $\iff \rightarrow_F$ has the Church-Rosser property.

$f \rightarrow_F g \dots f$ reduces to g in one remaindering step using divisors from F .

\rightarrow is Church-Rosser $\iff \forall_{g_1, g_2} (g_1 \leftrightarrow^* g_2 \implies g_1 \downarrow^* g_2)$

Main Theorem:

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

Proof: " \implies ": Easy.

For the direction " \impliedby " one can use the [Newman Lemma](#) (Newman 1942). (For the version of the algorithm with criteria one needs the generalized Newman lemma by BB.) For Noetherian \rightarrow :

\rightarrow is Church-Rosser $\iff \forall_{g_1, g_2, h} (g_1 \leftarrow h \rightarrow g_2 \implies g_1 \downarrow^* g_2)$

The proof of this lemma uses Noetherian induction. By using Newman's lemma in the proof of the main theorem, one takes induction out of the proof and is left with the specific technicalities of polynomial reduction.

Hence, we have to consider, for arbitrary polynomials g_1, g_2, h , the situation that

$$g_1 \leftarrow_F h \rightarrow_F g_2$$

and we have to show that we can always find a polynomial p such that

$$g_1 \rightarrow_F^* p \leftarrow_F^* g_2.$$

By the assumption, there exist polynomials f_1 and f_2 in F such that h reduces w.r.t. f_1 and f_2 . Let t_1 and t_2 be the power products in h on which these reductions work.

$$h = \dots + \square t_1 + \dots + \square t_1 + \dots$$

$$\begin{array}{cc} -u_1 f_1 & -u_2 f_2 \\ \text{yields } g_1 & \text{yields } g_2 \end{array}$$

Cases $t_1 < t_2$ and $t_2 < t_1$ easy (but not trivial!): by "semi-compatibility" of polynomial reduction.

Cases $t := t_1 = t_2$:

$$h = \dots + \square t + \dots$$

$$\begin{array}{cc} -u_1 f_1 & -u_2 f_2 \end{array}$$

In this case t is a multiple of the LCM m of $\text{LPP}[f_1]$ and $\text{LPP}[f_2]$: $t = v \cdot m$.

Since, by assumption of the theorem, the S-polynomial of f_1 and f_2 can be reduced to 0, the reduction of m in the two essentially different ways (starting once by using f_1 and once by using f_2) has a common successor.

Hence, by "stability" of polynomial reduction, by multiplication of all the steps by v , g_1 and g_2 have a common successor.

My Recent Research Interest: Automated Theory Exploration

For example: How can one **invent** (and verify) notions like "S-polynomial", theorems like the main theorem, and algorithms like the Gröbner bases algorithm **automatically**, i.e. by algorithms that work on formulae.

For example, algorithm synthesis:

Given the specification P of a problem.

Find an algorithm A such that $\forall_F P[F, A[F]]$.

I succeeded to come up with a method which, for many P , yields A automatically. In particular, with this method, starting from the specification of the Gröbner bases construction problem:

Given: F .

Find: G such that

G is finite

G is a Gröbner basis

$$\text{Ideal}[F] = \text{Ideal}[G],$$

one arrives automatically at the notion of S-polynomials and the above Gröbner bases algorithm based on the notion of S-polynomials.

For details see the recent publication

[B. Buchberger](#)

[Towards the Automated Synthesis of a Gröbner Bases Algorithm](#)

[RACSAM, 98/1 \(Rev. Acad. Cienc., Spanish Royal Academy of Science\), 98/1, pp. 65-75, 2005.](#)

and the [Workshop C "Formal Gröbner Bases Theory"](#), March 6-10, in the course of this special semester,