Symbolic Computation: A Personal View on the Future of Mathematics

Bruno Buchberger
Research Institute for Symbolic Computation
Johannes Kepler University, Linz, Austria

Symbolic Computation, roughly, is

computer algebra plus automated reasoning.

Symbolic Computation, more profoundly, is

computation (algorithms)
with expressions that have (infinite) semantics.

Integral[Sin[3 x^2,x]] = Sqrt[pi/6] FresnelS[Sqrt[6/pi] x]

Some Views (Metamathics)

An Example

Conclusions

Some Views

Mathematics is software (with and without computers).

Mathematics is the art of ex-planation.

Hence, of highest relevance to the woman on the street.

Mathematics is in the eye of the hurricane of innovation.

Hence, of highest relevance to society.

Mathematics is metamathics.

This is of high relevance to the future working mathematicians.

Proving and computing should be done in one system.

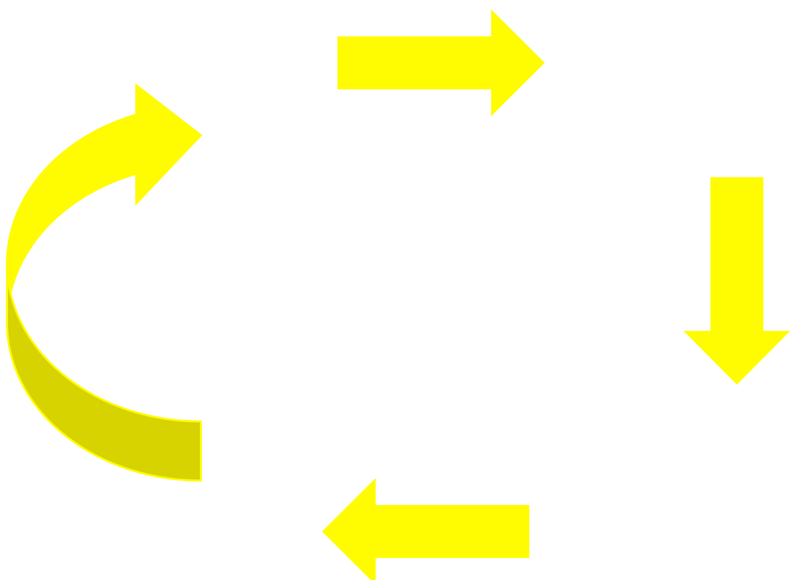
Predicate logic is a nice frame for metamathics.

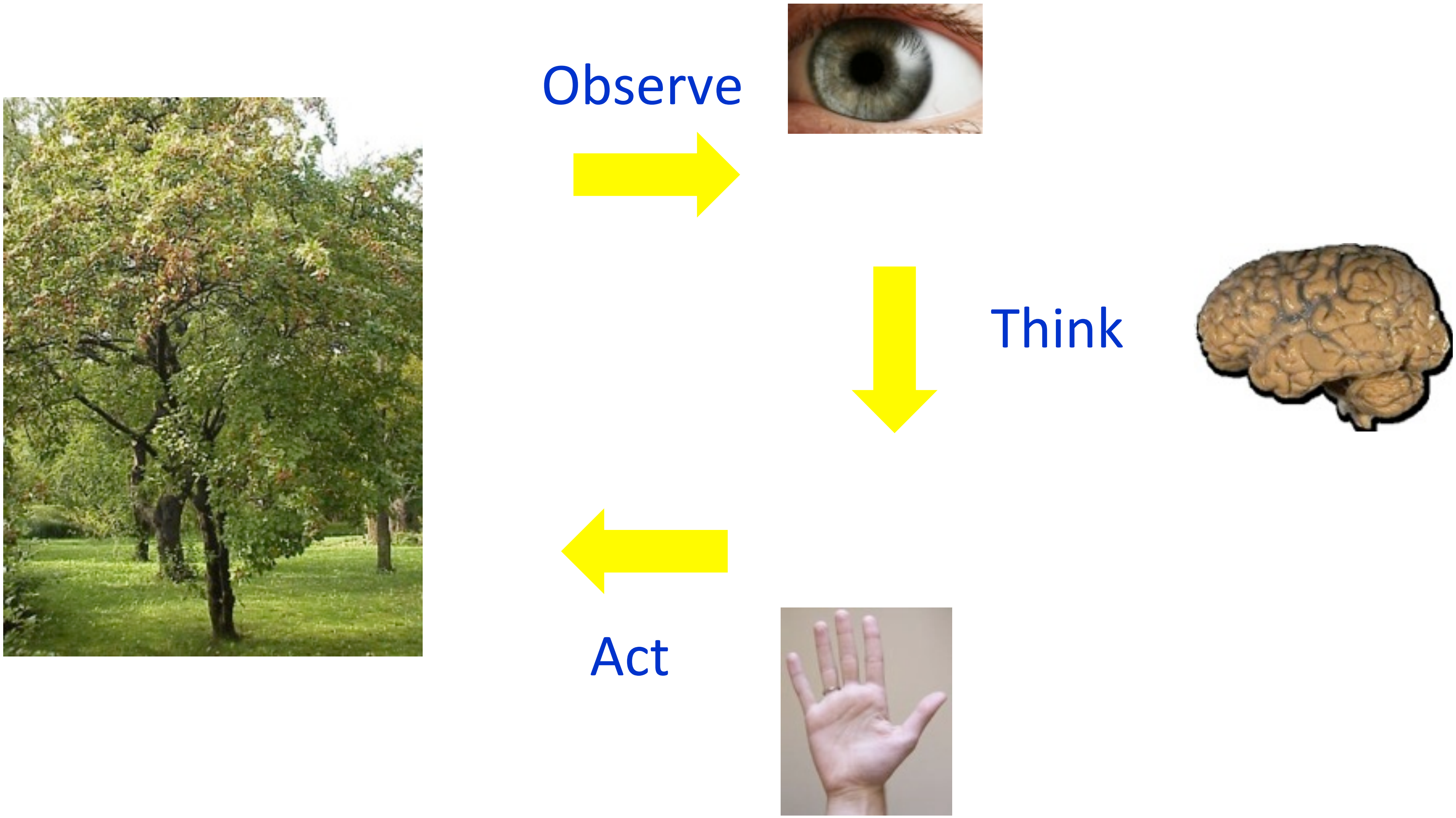
It also contains a nice programming language.

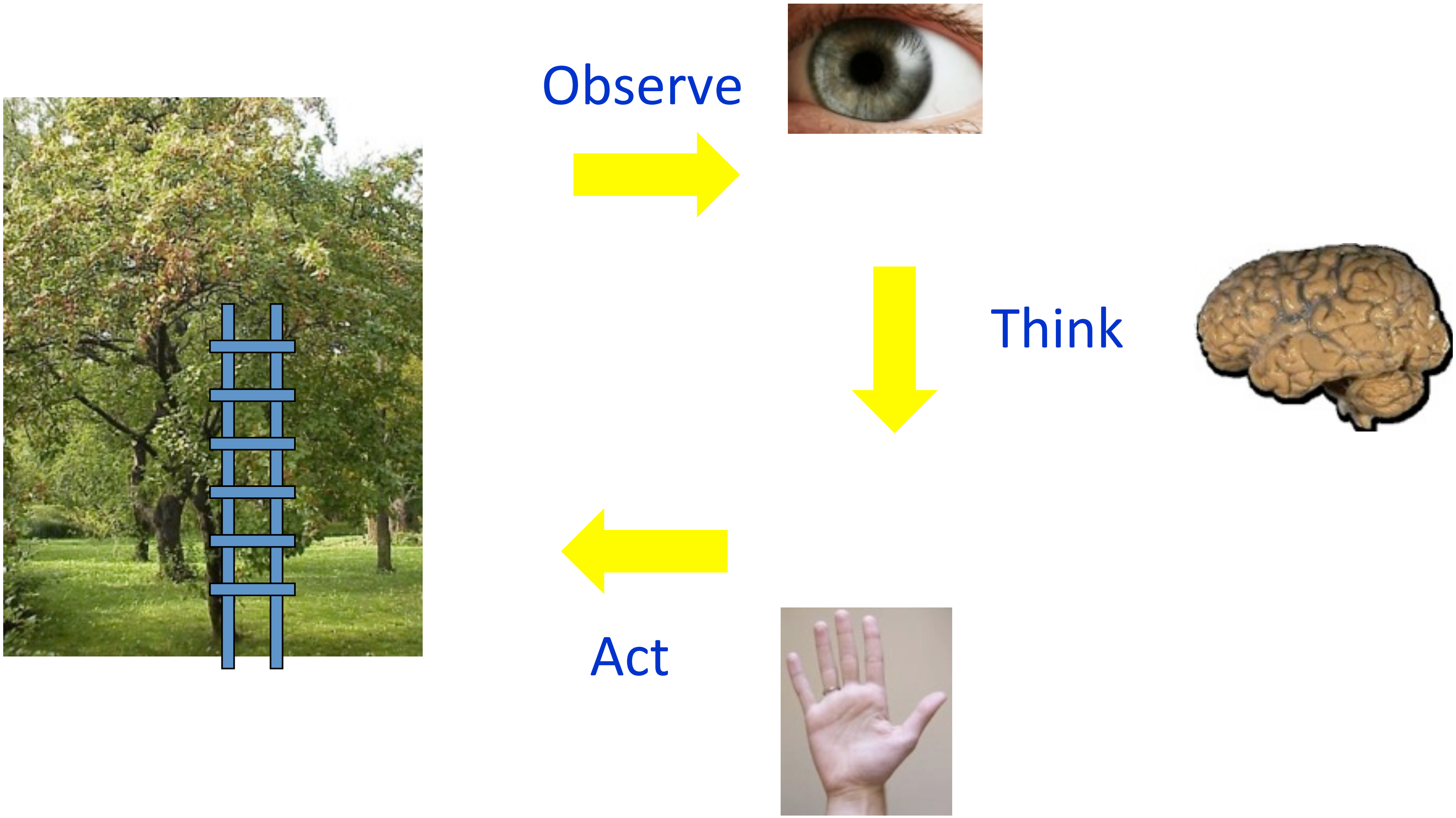
(The Theorema language is predicate logic with sequence variables.)

Metamathics

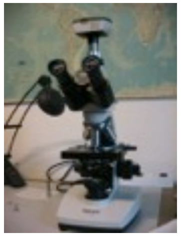
Start from (before) Adam and Eve …



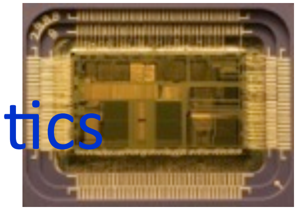

Observe
Think
Act


Observe
Think
Act

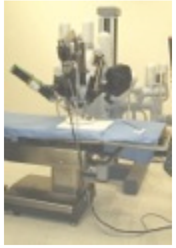
Science



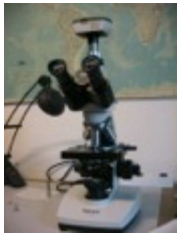
Mathematics



Technology

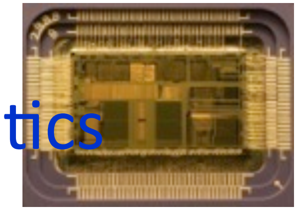


Science

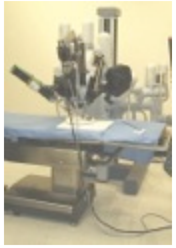


Intersubjective
Verifiability

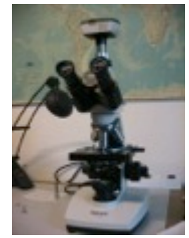
Mathematics



Technology

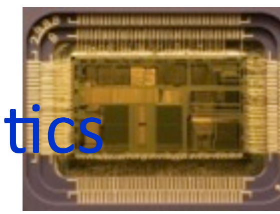


Science



Anonymous
Peer
Reviewing

Mathematics



Technology

Science

Mathematics

Technology

"Of course", a good scientist (mathematician, engineer, teacher, student, …, university, school, …) tries to live all three aspects equally well.

Science

Mathematics

Technology

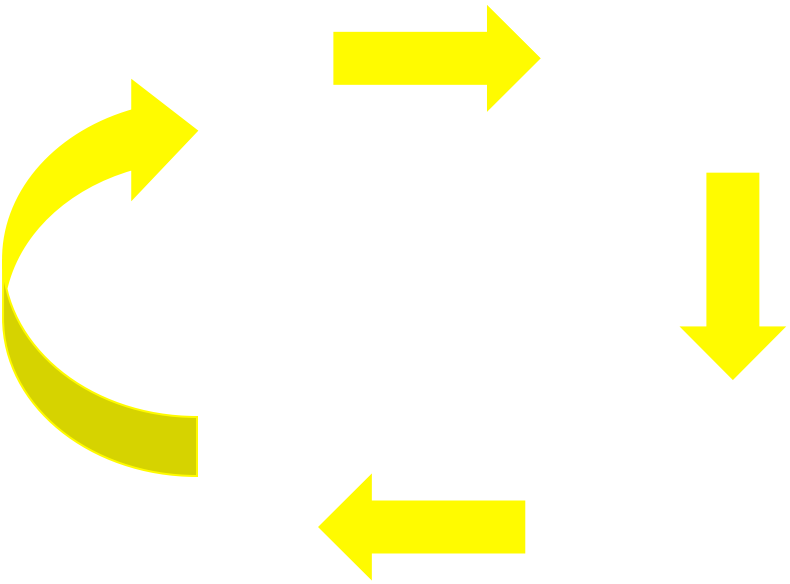
However, methodologically,
the three arrows are different!

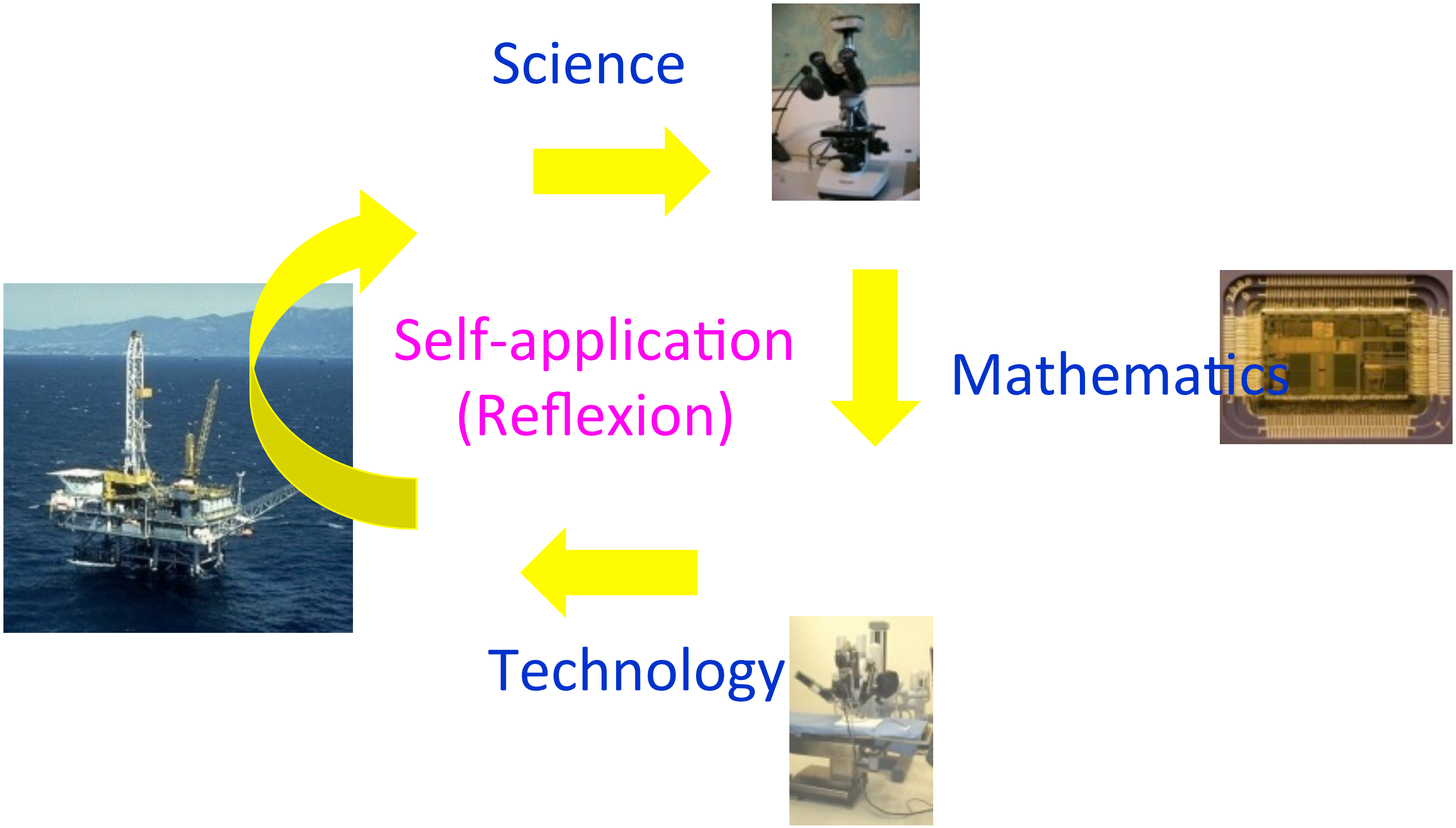
Science

Thus, now, mathematics, essentially, is
the art of gaining knowledge
and solving problems
by thinking (reasoning, ...)

Mathematics

Technology




Science
Mathematics
Technology
Self-application
(Reflexion)

Self-application:

the intelligence of nature,

the nature of intelligence.

By self-application,
the science / techno / eco spiral
gains speed, sophistication, …

The history of science
is the history of

how much these arrows became explicit.

If one understands this

then one can look (forward) to the future.

Metamathics:

mathematics applied to mathematics

Metamathics:

doing mathematics
on two floors at the same time.

Metamathics: Working on two floors at the same time.

Metamathics is not (only) meta-mathematics.

Practically, metamathics comes in two flavors.

Flavor "Lift Knowledge to Inference":

Stage n of exploring a theory:

- Using the current inference system, derive knowledge on the current notions.
  (E.g.: " lim(f+g) = lim(f) + lim(g) ".)

- Turn this knowledge into new inference techniques.
  (E.g.: " $\lim_{n} (S+T) = \lim_{n}(S) + \lim (T)$.)

Stage n+1 of exploring a theory:

- Using … new inference techniques …

Flavor “Develop Knowledge and Inference in Parallel”:

Stage n of exploring a theory:

- Using the current inference system try to derive new knowledge on the current notions.

- At the same time, experiment with designing new inference techniques for facilitating the derivation of new knowledge.

Predicate logic is practical for both floors of metamathics.

There is a subtle problem with lifting: How can one safely (automatically) lift proven knowledge to correct inference rules?

My view: metamathics will be the mathematics of 21^{st} century.

(?)

Ancient mathematics: see = observe

Modern mathematics:
distinction between seeing (observing) and seeing (proving).

20^{th} century math: logic and computer.

- The Theorema Project reflects my view of mathematics (“from 17 to 71”)
- As a project, initiated and conceived Theorema around 1995
- A cooperative project of the Theorema Group (T Jebelean, T Kutsia, W Windsteiger, …)
- Many similar projects (Calculemus, MKM, …, see also QED)
- At International Congress of Math 2014, Seoul: Global Math Digital Library task force
- Vladimir Voevodsky’s talk 2013

www.risc.jku.at/research/theorema/software/

An Example of Metamathics: Gröbner Bases

In the flavor of: „Knowledge and Inference in Parallel“:

- „first floor“: Solve Gröbner‘s 1964 problem (Gordan 1899)
- „second floor“ (1995 - ...): Lazy Thinking

How would I proceed, in the spirit of metamathics, if Gröbner gave me his problem today (2015)?

First Floor: Gröbner Bases

Definition of Gröbner bases: … (or something similar) …
(F is a GB iff division w.r.t. F is unique.)

Auxiliary knowledge on polynomials …, e.g. congruence modulo ideals …

The Main Idea of Algortihmic Gröbner Bases Theory (BB 1965):

Theorem: F is a Gröbner basis iff

all "S-polys", w.r.t. F can be reduced to 0.

S-poly(f,g) := u . f - v . g,

where u, v are such that

u . L(f) = v . L(g) = LCM (L(f), L(g)).

In 1964, it took me 1 ½ years to get the idea of S-polys, the theorem (and algorithm) and then to prove the theorem.

Algorithm: For obtaining Gröbner bases G for F,

iterate the formation of S-polys

until all of them reduce (“divide”) to 0.

Hence, the question: could one (automatically) obtain the idea for S-polys, the theorem, its proof, and the algorithm by working on the "second" floor.

Second Floor: Lazy Thinking

- Let’s assume we already implemented an induction prover for polynomials (and similar inductive structures).

- Let’s also assume that we studied the literature for available “algorithmic ideas” (I call them “algorithm schemes” here).

- (Among the algorithm schemes there could be one, which I call “critical pair / completion” scheme. Before 1964, cum grano salis, this was the flavor e.g. in Todd – Coxeter’s algorithm.)

- Here is now an idea for working on the “second floor”: Lazy Thinking.

The Lazy Thinking Method for Algorithm Synthesis:

Given: A problem specification P.

1st Step: Try out (one of finitely many) algorithm schemes A.

Example of an algorithm scheme: “divide and conquer”:

A(x) := S(x) if x is basic
M (A (L(x)), A (R(x)) if x is not basic.

Another example of an algorithm scheme:
critical pair / completion (see demo).

A(x) := S(x) if x is basic
… CP(.. X ..) …

2nd Step: Try to prove (automatically): for all x, P(x, A(x)).

The proof will probably fail because nothing is known about the sub-algorithms S, M, L, R, …

3rd Step (the kernel of lazy thinking): From the failing proof, try to derive (automatically) specifications for the subalgorithms in the scheme (e.g. S, M, L, R, …) that will make the proof work.

4th Step: Iterate Lazy Thinking until sub- … -sub-algorithm specifications are found for which algorithms are known.

When this Lazy Thinking procedure is applied to the specification P of the Gröbner bases problem (using the “critical pair / completion algorithm scheme”) then, in a couple of minutes, one obtains the main idea of Gröbner bases theory, i.e.

- the central notion of S-polynomial,
- the main theorem on S-polys and Gröbner bases,
- and the algorithm for computing Gröbner bases.

See demo.

Another Example of Metamathics: Complexity for Bivariate Gröbner Bases

See paper and talk by A. Maletzky and BB at International Conference on Mathematical Software (August 2014).

Flavor: „Lift Knowledge to Inference".

Conclusions

- Mathematics will always be in the center of the innovation spiral.
- Ultimately, mathematics strives towards becoming algorithms (… software). In other words, mathematics strives towards trivializing itself.
- This is the miracle and power of mathematics.

Conclusions

- 21st century mathematics: metamathics.
- The level of sophistication in metamathics has no upper bound. (Thank you, Gödel!)

Conclusions

- World mathematics will evolve into a global metamathics journal.

Conclusions

- Education in mathematics and computer science: Acquire the ability to do metamathics.

Anticlusion

- Forget "metamathics".
- Just say "mathematics" (but think also meta).

Contact:

Bruno Buchberger
Tel.: +43 (0) 664 4211646
bruno.buchberger@risc.jku.at

Professor of Computer Mathematics (1974 -)
Research Institute for Symbolic Computation (RISC)
Johannes Kepler University, Linz, Austria, Europe

Founder and Head (1989-2013)
of Softwarepark Hagenberg