Calculemus & Issac

Communicating mathematics over the web

Henk Barendregt
Nijmegen University

Arjeh Cohen
Eindhoven University

The current situation

There are systems for

- Computing
  numerical and symbolic
- Proving
  fully automated (Otter)—interactive (COQ)—human constructed (Automath)
- Visualising, Editing, ...

We focus on

CAS: Computer Algebra Systems, for symbolic computations

PA: Proof Assistants, from fully automated to human constructed

Shortcomings of established systems

They are good at computations and/or proofs
but most of them have some of the following drawbacks.

- not impeccable:

  sloppy about side-conditions

  contain bugs

Non impeccability examples

- side conditions:

$$\int x^a \mathrm{d}x = \frac{1}{a+1}\, x^{a+1} \qquad\qquad (a \neq -1).$$

- bugs:

`int(f,x=0..1)` = `int(simplify(f),x=0..1)`

with $f(x) = 1 - 1/(2 + e^{2\pi i x})$:

Maple V release 4: $1 = 0.$

Mathematica: $\frac{1}{2} = \frac{1}{2}.$

Shortcomings of established systems

They are good at computations and/or proofs
but most of them have some of the following drawbacks.

- not impeccable:
  sloppy about side-conditions
  contain bugs
- limitations:
  availability (license, sources)
  computational/proving capabilities for specific tasks
- special purpose:
  CAS are short on semantics
  PA lack computational power

CLAIM: In an appropriate open setup we can do much better.

Integrating Computer Algebra Systems & Proof Assistants

- Proving often needs computing
- Computer algebra can benefit from formal side conditions
- Proofs warrant reliability

We can represent mathematics

not just $\sqrt{2}$ or $\int_0^\pi e^{-\sin x}\,dx$

but even an entire Hilbert space.

Abstract history of mathematics

activity	#years
Counting	6000
Computing	3000
Defining and proving	2500
Defining, proving & computing	1250
Modern mathematics	200

Only since 19-th century proving and computing in informal mathematics have been fully integrated.

Needed: common foundation

- Expressive
- Deductive power
- Reliable
- Efficient

Expressivity

- Axiomatic method (Aristotle)
  objects and properties
  objects: atomic (primitive) or composed (defined)
  properties: axioms or proved

Expressivity

- Axiomatic method (Aristotle)
  objects and properties
  objects: atomic (primitive) or composed (defined)
  properties: axioms or proved
- Equational logic (Leibniz)

Expressivity

- Axiomatic method (Aristotle)
  objects and properties
  objects: atomic (primitive) or composed (defined)
  properties: axioms or proved
- Equational logic (Leibniz)
- Predicate logic (Frege)

Expressivity

- Axiomatic method (Aristotle)
  objects and properties
  objects: atomic (primitive) or composed (defined)
  properties: axioms or proved
- Equational logic (Leibniz)
- Predicate logic (Frege)
- Intuitionistic logic (Heyting)
  can express effective dependence

Intuitionism

I can make a machine that says “Yes” within 5 seconds,
if the Riemann hypothesis is correct
and says “No” within 5 seconds if it is not.

Intuitionism

I can make a machine that says “Yes” within 5 seconds,
if the Riemann hypothesis is correct
and says “No” within 5 seconds if it is not.

One can have

$$\vdash \exists n\ An$$

but there is no witness, i.e.

$$\nvdash A0,\ \nvdash A1,\ \nvdash A2, \ldots$$

Brouwer showed that both unsatisfactory phenomena are caused by the law of excluded middle

$$A \vee \neg A$$

Heyting formalised intuitionistic logic. Gödel showed that intuitionistic logic is stronger than classical by defining

$$\exists^{o} x\ Ax \ \Leftrightarrow\ \neg\forall x\ \neg Ax \tag{1}$$

$$A \vee^{o} B \ \Leftrightarrow\ \neg(\neg A\ \&\ \neg B) \tag{2}$$

Computability expressed

If in intuitionism

$$\vdash \forall x \exists y\ Axy$$

then the y can be found effectively from the x.

Expressivity

- Axiomatic method (Aristotle)
  objects and properties
  objects: atomic (primitive) or composed (defined)
  properties: axioms or proved
- Equational logic (Leibniz)
- Predicate logic (Frege)
- Intuitionistic logic (Heyting)
  can express effective dependence
- Type theory (Church)
  multi-sorted
  λ-terms: explicit calculus of executable functions
  formal proofs
  inductive types, e.g. `nat::= zero | succ nat`

From informal to formal mathematics

- Informal mathematics

  In situation Γ one has P.
  Proof. ... QED

- In logic

  $\Gamma \vdash P$

- In type theory

  $\Gamma \vdash a : P$

The last statement is decidable

$$\Gamma \vdash a : P \Leftrightarrow \mathtt{type}_{\Gamma}(a) = P$$

Deductive power

- First-order logic (Frege)
  $I \subseteq R$ is an ideal
- Second-order logic (Hilbert-Bernays)
  R is a noetherian ring
- Higher-order logic (Church)

  There are natural type theories corresponding to these three logics

- Settheory (Zermelo-Fraenkel: ZFC)
  every chain of ideals is contained in a maximal one
  Can be formulated in first-order logic with special axioms

Reliability

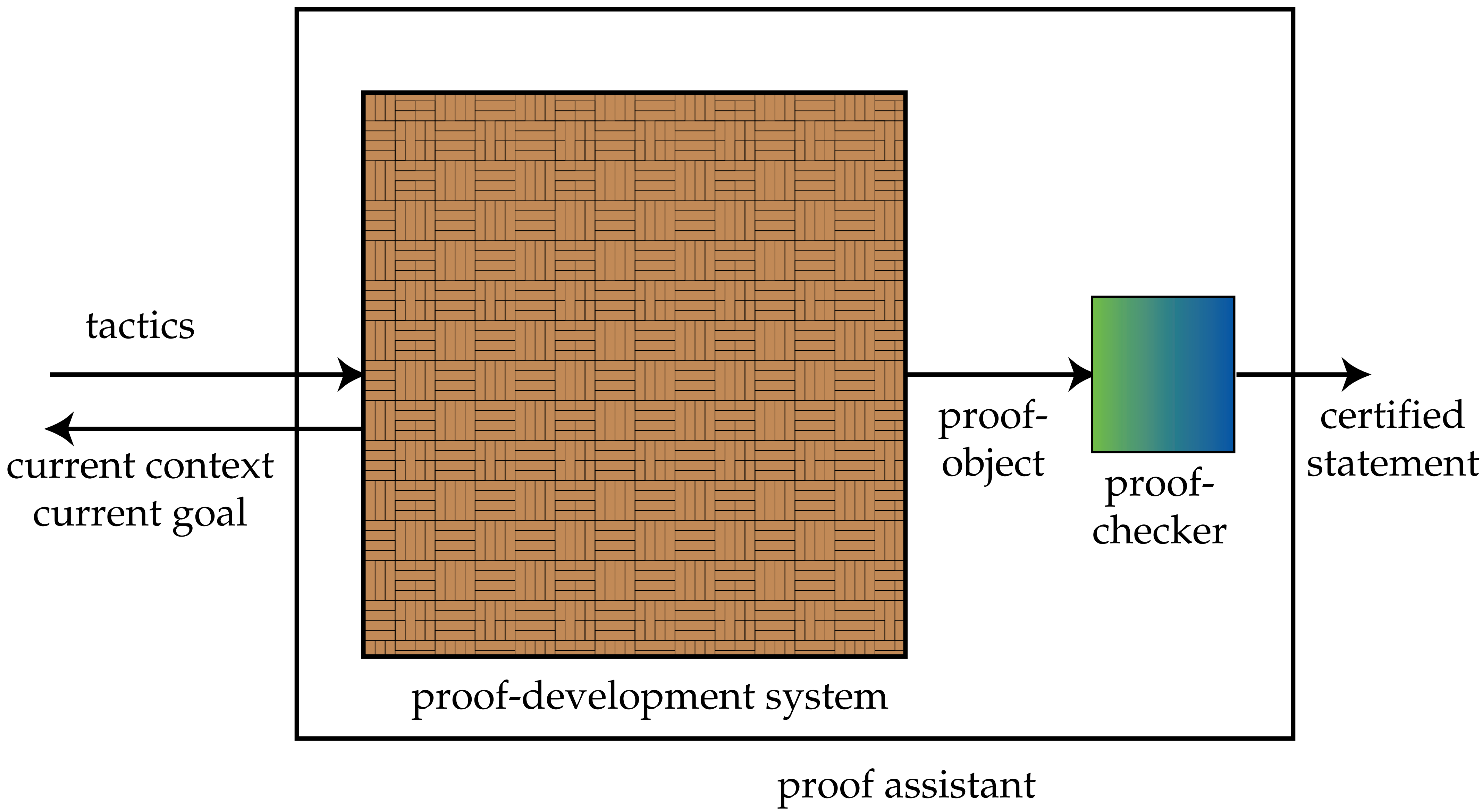
Proof-objects: fully formalised proofs such that
a small program can verify them
the de Bruijn criterion

Proof-objects give enable us to give the predicate

evidently impeccable proof-assistent

the highest degree of reliability

Proof-objects have been considered first by Whitehead & Russell. They have been refined by de Bruijn.


tactics
current context
current goal
proof-object
proof-checker
certified statement
proof-development system
proof assistant

Interaction between computing and proving

Computation problem $\mathbf{9^9 + 8^8 = 8^8 + 9^9}$?

Prove $\forall n, m \in \mathbb{N}\ n + m = m + n$

Provability problem. Let $P_n = (..((\text{true}\!\to \underbrace{p)\!\to\! p) \ldots \!\to\! p}_{n \times p}).$

$$\vdash \mathbf{P_{2000}} \ ?$$

Better first prove

$$\begin{aligned} P_{2k} &\equiv \text{true} \\ P_{2k+1} &\equiv p \end{aligned}$$

and show by **computing** that 2000 is even.

Moral for efficiency

If $F(a) = b$, then

$$\vdash \underline{F}\ \underline{a} = \underline{b}$$

should not be difficult to show.

Poincaré Principle

$$\vdash \underline{F}\ \underline{a} = \underline{F(a)}$$

is an axiom for some class of functions.

Proof by computing

Define $f : \mathbb{R} \to \mathbb{R}$

$$f(x) = \sin x + \sin 2x + \sin 4x + \sin 6x.$$

Then f is continuous.

This can be proved as follows

$L ::= \mathtt{x} \mid L \texttt{ + } L \mid \mathtt{sin}\ L$
$[\![\ ]\!]{:}L \to (\mathbb{R} \to \mathbb{R})$

$\vdash \forall e{:}L\ [\![e]\!]$ is continuous

$[\![$‘`sin x + sin 2x + sin 4x + sin 6x`’$]\!] =$
$\quad x \mapsto \sin x + \sin 2x + \sin 4x + \sin 6x$

Implementing the Poincaré Principle

$$\frac{\Gamma \vdash a : P(t)}{\Gamma \vdash a : P(t')} \quad \text{if } t =_R t'.$$

R arbitrary TRS

$$\boxed{\begin{array}{rcl} p & \rightarrow_{\text{simpl}} & p^{\text{simpl}} \\ 7+5 & \rightarrow_{\text{sum}} & 12 \end{array}}$$

Implementing the Poincaré Principle

$$\frac{\Gamma \vdash a : P(t)}{\Gamma \vdash a : P(t')} \quad \text{if } t =_R t'.$$

R arbitrary TRS versus

δ definition unfolding
β $(\lambda x.M)N = M[x := N]$
ι primitive recursion on data types

$$\boxed{\begin{array}{rl} p \rightarrow_{\text{simpl}} & p^{\text{simpl}} \\ 7+5 \rightarrow_{\text{sum}} & 12 \end{array}} \quad \text{versus} \quad \boxed{\begin{array}{rl} F_{\text{simpl}}\, p \rightarrow_{\delta\beta\iota} & p^{\text{simpl}} \\ F_{\text{sum}}\, 7\ 5 \rightarrow_{\delta\beta\iota} & 12 \end{array}}$$

Implementing the Poincaré Principle

$$\frac{\Gamma \vdash a : P(t)}{\Gamma \vdash a : P(t')} \qquad \text{if } t =_R t'.$$

R arbitrary TRS versus

δ definition unfolding

β $(\lambda x.M)N = M[x := N]$

ι primitive recursion on data types

$$\boxed{\begin{array}{rl} p \to_{\text{simpl}} & p^{\text{simpl}} \\ 7+5 \ \to_{\text{sum}} & 12 \end{array}} \quad \text{versus} \quad \boxed{\begin{array}{rl} F_{\text{simpl}}\, p \to_{\delta\beta\iota} & p^{\text{simpl}} \\ F_{\text{sum}}\ 7\ 5 \to_{\delta\beta\iota} & 12 \end{array}}$$

$S(F)$ specification

$\Gamma \vdash S(F)$ validity

Basic statements

P is provable from axioms Γ by proof a using R as rewriting and L-logic

$$\Gamma \vdash_R^L a : P$$

This is equivalent to

$$\mathbf{type}_\Gamma^{\mathbf{L}}(\mathbf{a}) =_{\mathbf{R}} \mathbf{P}$$

which is decidable.

Summary

There is a family of logical systems (Type Theory) such that

- They contain proof-objects
- They can reason and compute at variable strengths
- Their main assertion $\Gamma \vdash^L_R a : P$ is 'easily' checkable

This family is a good candidate for communicating mathematics over the web.

A proof in Coq
Every antisymmetric binary relation is irreflexive.

```
Require Export intuitionistic.
(* Export deals with pretty printing and parsing;
   intuitionistic has some basic tautologies.*)

Section Tarski.

Variables X:Set; R:X->X->Prop.

(* Let X be a set. Let R be a binary predicate on X. *)

Definition antisymmetrical : Prop
                           := (x,y:X)((R x y)->~(R y x)).

Definition irreflexive : Prop
                       := (x:X)~(R x x).
```

```
Lemma tarski : antisymmetrical -> irreflexive.
(*Proof*)
Show.
Intro a. Unfold irreflexive.
Intro x. Intro b. Apply (a x x b).
Assumption.
Defined.(* QED *)
Print tarski.

(* tarski =
[a:antisymmetrical; x:X; b:(R x x)](a x x b b)
     : antisymmetrical->irreflexive
*)

End Tarski.
```

```
(*
[Cooking #antisymmetrical.cci]
[Cooking #irreflexive.cci]
[Cooking #tarski.cci]
Constant antisymmetrical : (X:Set)(X->X->Prop)->Prop
Constant irreflexive : (X:Set)(X->X->Prop)->Prop
Constant tarski
  : (X:Set; R:(X->X->Prop))(antisymmetrical X R)->(irreflexive X R)
*)

Print tarski.

(*
tarski =
[X:Set; R:(X->X->Prop); a:(antisymmetrical X R); x:X; b:(R x x)]
 (a x x b b)
     : (X:Set; R:(X->X->Prop))(antisymmetrical X R)->(irreflexive X R)
*)
```

General mathematical assertion

will look like

$$\Gamma \vdash a \colon P$$

where,

- Γ is the context,
- P is a statement, and
- a is a proof of P.

In our ideal setup, such assertions can be communicated by individual agents on the Web.

Why better

- evidently impeccable:
  - By a simple procedure
    (type inference & confluent R)
    one checks that a has type P.
- general purpose:
  - By varying Γ, we can cater for any logic, any kind of (incomplete) theorem, etc.
  - The various individual CASs and PAs will each add to the computational and proof construction power.

Example of a formal assertion

$$\Gamma \vdash a : P$$

$\Gamma =$ Peano arithmetic.

$P = \gcd(4,6) = 2$.

$a =$ [sketchy] a (long formal) sequence of statements made up of axioms in Γ and

Lemma. $\gcd(m,n) = d$ *iff there are integers* x *and* y *with* $xm + yn = d$ *and* $d|m$ *and* $d|n$,

with an application to $m = 4$ and $n = 6$ and $d = 2$, with $x = -1$ and $y = 1$.

Hierarchy of algorithms wrt ‘convincibility’

algorithm	examples
1. proven (to be correct)	Euclid, Buchberger
2. checkable	factorization, antiderivative, LLL
3. witnesses can be supplied	primality, matrix conjugacy, Jacobson radical
4. traceable	Todd Coxeter
5. oracle	heuristic searches, nonexistence of PG(2,10)

(Convincibility represents a dimension different from complexity.)

The convincibility hierarchy, once more

Given an algorithm f and a specification S of f
(so that $S(x, y)$ holds whenever $f(x) = y$),
we can describe the algorithm properties formally as follows.

algorithm	formula	
1. proven	$\Gamma \vdash \forall x\ S(x, f(x))$	autarkic
2. checkable	$\forall x, y\ \left[f(x) = y \Rightarrow\ \Gamma \vdash a(\underline{x}, \underline{y}) : S(\underline{x}, \underline{y})\right]$	skeptic
3. witnesses	$\forall x, y\ \left[f(x) = y \Rightarrow \exists \underline{z}\ \Gamma \vdash a(\underline{x}, \underline{y}, \underline{z}) : S(\underline{x}, \underline{y})\right]$	skeptic
4. traceable	as for witness case, but long $\underline{z}$	skeptic
5. oracle	(no evidence)	believing

Primality witnesses

Lemma (Pocklington). *Let $n > 1$ be a natural number for which there exist primes $q_1, \ldots, q_t$ and a positive integer g such that*

1. $q_1 \cdots q_t$ *divides* $n - 1$,
2. $g^{n-1} = 1 \mod n$,
3. $\gcd(g^{(n-1)/q_i} - 1, n) = 1$ *for* $i = 1, \ldots, t$, *and*
4. $q_1 \cdots q_t \geq \sqrt{n}$.

Then n is a prime.

CAS used to provide the witnesses g, $q_1, \ldots, q_t$ and (recursively) witnesses for primality of the latter.

See presentation by Olga Caprotti & Martijn Oostdijk.

Gröbner basis witness program

$\Gamma :=$ $R = \mathbb{Q}[x, y]$, reduction ordering on R with $x > y$, etc.

$I =$ $(x^2y - 1, xy^2 - 1)R$ (in Γ).

$P :=$ $\{x - y, y^3 - 1\}$ is a Gröbner basis of I.

$M =$ the witness: a 2×2 matrix over R such that

$$\begin{pmatrix} x - y \\ y^3 - 1 \end{pmatrix} = M \begin{pmatrix} x^2y - 1 \\ xy^2 - 1 \end{pmatrix}.$$

$a =$ a proof of P, can be constructed from M using 'trivial' (normal form) computations only.

Gröbner basis proven

Poincaré Principle at work.

$$
\begin{aligned}
P :=&\quad \{x-y, y^3-1\} \text{ is a Gröbner basis of } I.\\
P =&\quad \mathrm{GB}(I) = \{x-y, y^3-1\}.
\end{aligned}
$$

GB = Buchberger's algorithm. Is proven to be correct (Théry, Pottier).

P' := the statement P with GB(I) replaced by the result of the GB computation.

Poincaré Principle: check

$P' = \quad \{x-y, y^3-1\} = \{x-y, y^3-1\}$, which is obviously true.

Conclude:

$$\Gamma \vdash GB(I) = \{x-y, y^3-1\}.$$

Antiderivative

f is a function , F an antiderivative found.

- Check $F' = f$.
- Complication: Need proven differentiation algorithm.

Conjugacy of matrices

Given two square matrices x_1, x_2 (over a field). Are they conjugate?

- Either $\exists g$, invertible matrix, such that $gx_1g^{-1} = x_2$, or
- they have distinct Jordan blocks (show by conjugation to Jordan normal form).

Conclusion:
Witness program suffices.

Factorization of integers

- Check $x = x_1 x_2 \cdots x_n$
- Complication: also need to check that the x_i are prime,
  so witnesses are needed.

Conclusion:
For full factorization algorithm needs witness program for primes.

Similarly for polynomial factorization.

Galois group of a polynomial

Query: What is the Galois group of

$$F := x^{12} + \frac{100}{3}x^{11} + 450x^{10} + 3100x^9 + 10975x^8 + 15000x^7$$
$$-14500x^6 - 49000x^5 + 15375x^4 + 57500x^3 + 7500x + 625?$$

- Matzat
- Obvious: M_{12} (Mathieu group) or A_{12} (Alternating group).
- S := 6-set resolvent polynomial, degree(S)=$\binom{12}{6} = 924$.
- S factors into $T \cdot U$, with degree(T) $= 132$.

(Van Hoeij, 55 Ks and 174 Mb on SUN sparc Ultra-4)

- F and 5 -set resolvent of F are irreducible, so Galois group of F is M_{12}.
- Galois group up to isomorphism determination 'algorithm' could work with witnesses if enough permutation group knowledge is given.

- Galois group algorithm can be reduced to GB, so proven (far from practical situation).

Todd Coxeter coset enumeration.

$$G := \text{free group on } a, b, c, d \text{ modulo the relators } a^2, b^2, c^2, d^2, (ab)^3, (dc)^3, (ca)^3, (cb)^4, (cbab)^4, (abc)^7, (ad)^2, (bd)^2.$$

$$H := \langle a, b, c \rangle$$

- The Todd Coxeter algorithm enumerates cosets of G wrt subgroup $H = \langle a, b, c \rangle$.
- e.g., H, Hd, $Hda = Had = Hd$, $\ldots$
- GAP finds exactly 15360 cosets.
- Application: provides a 128-cover of a known graph on 120 vertices (both locally a known graph on 28 vertices).
- All identifications of cosets can be certified by relations found during the coset enumeration

Conclusion:
Semi-algorithm (termination may fail); but if successful, then traceable.

The group G has a normal subgroup $N \cong 2^{1+6}$ with quotient $O^+(6,2) \cong L(4,2) \cong Alt_8$. The group is an amalgam of the subgroups

$$\begin{aligned}
\langle w, u, b\rangle &\cong L(3,2), \\
\langle w, u, a\rangle &\cong L(2,2) \times 2, \\
\langle a, b\rangle &\cong Sym_3,
\end{aligned}$$

over

$$\begin{aligned}
\langle w, u, b\rangle \cap \langle w, u, a\rangle \langle w, u\rangle &\cong L(2,2), \\
\langle w, u, a\rangle \cap \langle a, b\rangle = \langle a\rangle &\cong 2, \\
\langle a, b\rangle \cap \langle w, u, b\rangle = \langle b\rangle &\cong 2.
\end{aligned}$$

Radical determination.

- R a finite-dimensional algebra (given by multiplication table).
- S radical of R, is the biggest solvable ideal of R.
- S can be determined algorithmically (suitable field).
- Better than oracle?

Lemma. *S is the radical of R iff*

1. S is solvable, and

2. the radical of R/S is trivial.

Conclusion:
Radical determination has (complicated) witnesses,
but is ‘cross checkable’ by the lemma.

The same algorithm applied to a simpler situation does the verification.

Conclusion on Algorithms

- CAS algorithms can be geared to serve PA in a hierarchically sorted variety of ways.
- There is a trade off between convincibility and efficiency.
- In the oracle case, often have cross check facilities (no meaning for PA's).
- There is a role for heuristic algorithms, semi-algorithms, can be incorporated (oracles, witnesses).
- For interaction *query* facilities would be welcome.
- Apart from a query facility, interfaces with individual packages need to come about (OpenMath can be the carrier).

Queries

can be incorporated in a formal system:

- $$\Gamma \vdash f(x) =?$$
  asking for a computation,
  e.g., with Γ = Peano arithmetic, and f = gcd.

- $$\Gamma \vdash P?$$
  asking for the truth (provability) of P,
  e.g., with
  - Γ the theory of Euclidean geometry and
  - P = 'For each triangle, there is a point all of whose cevalines have the same length.'

- $$\Gamma \vdash ? : P$$
  for conjecturing P in the context Γ.

How realistic is the proposed setup?

- Partial or gappy proof
  can be expressed by means of axioms thrown into Γ as 'hypotheses'.
  For "Try induction on ..." for a proof of "..." see interactive PA.
- The proposed language is essentially the language of COQ.
- OpenMath can support it.
  Γ to be stored in Content Dictionaries.
  OMDoc is a first attempt at defining a structured mathematical document.
- Communication (agent based) technology is on the way.
- Editors are a concern (Math General instead of Proof General).
  Also, to lesser extent, rendering of mathematics.

Conclusion

For Web related mathematical activities, a lot of dull work, searches, computations, proof completions and proof verifications can be left to machines.

The setup of type theory has the following advantages over the current situation:

- It does not rely on a single server, package, program.
- It enables communication across a network with equal partners.
- It can lead to a mathematical world which
  - allows for most (common) axiom systems,
  - knows democratic distributed 'collaboration',
  - has a warranty against corruption.
- Both humans and computers can participate.