formalization of mathematics

Freek Wiedijk
Radboud University Nijmegen

TYPES Summer School 2005
Göteborg, Sweden
2005 08 23, 11:10

intro

the best of two worlds

formalization of mathematics is like:

- **computer programming**

  concrete, explicit

  a formalization is much like a computer program

- **doing mathematics**

  abstract, non-trivial

  a formalization is much like a mathematical textbook

you will like it only if you like **both** programming and mathematics
but in that case you will like it very very much!

table of contents: the two parts of this talk

first hour: an overview of

the current state of the art in formalization of mathematics

in the reader: QED manifesto

second hour: an overview of

Mizar, the most ‘mathematical’ proof assistant

in the reader: Mizar tutorial

first hour:
state of the art in formalization of mathematics

four ways to do mathematics in the computer

- **numerical mathematics**, experimentation, visualisation

  numbers: computer $\longrightarrow$ human

- **computer algebra**

  formulas: computer $\longrightarrow$ human

- **automated theorem provers**

  proofs: computer $\longrightarrow$ human

- **proof assistants**

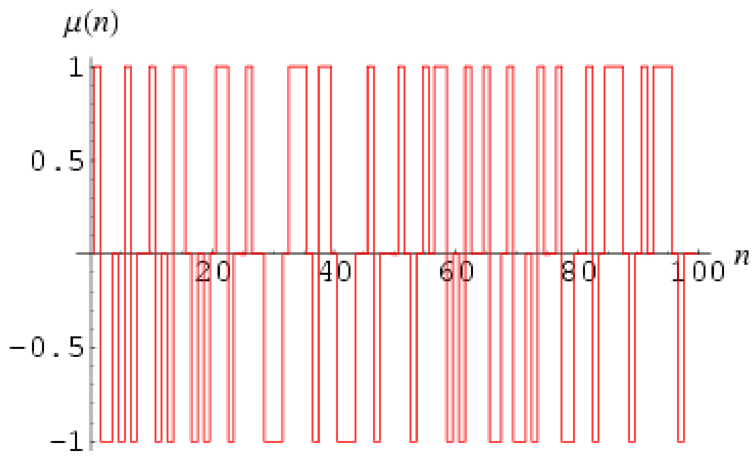
  proofs: human $\longrightarrow$ computer

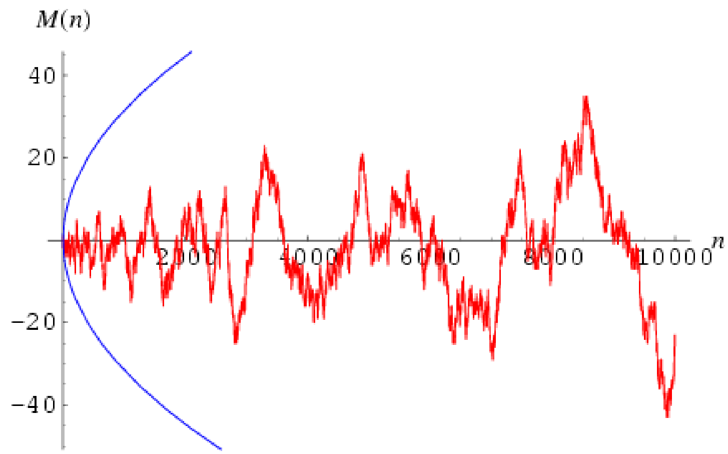
numerical mathematics: Merten's conjecture

Möbius function:

$$\mu(n) = \begin{cases} 0 & \text{when } n \text{ has duplicate prime factors} \\ 1 & \text{when } n \text{ has an even number of different prime factors} \\ -1 & \text{when } n \text{ has an odd number of different prime factors} \end{cases}$$

Mertens, 1897: $$\Big|\sum_{k=1}^{n} \mu(n)\Big| < \sqrt{n} \quad ?$$





Merten's conjecture (continued)

Odlyzko & te Riele, 1985: Mertens conjecture is false!

50 uur computer time

first n where it fails has tens of digits

indirect proof!

2000 zeroes of the Riemann zeta function to 100 decimals precision

14.134725141734693790457251983562470270784257115699243175685567460149963429809256764949010393171561012 7...
21.022039638771554992628479593896902777334340524902781754629520403587598586068890799713658514180151419 5...
25.010857580145688763213790992562821818659549672557996672496542006745092098441644277840238224558062440 7...
30.424876125859513210311897530584091320181560023715440180962146036993329389333277920290584293902089110 6...
32.935061587739189690662368964074903488812715603517039009280003440784815608630551005938848496135348724 5...
37.586178158825671257217763480705332821405597350830793218333001113622149089618537264730329104945823803 4...
40.918719012147495187398126914633254395726165962777279536161303667253280528720071282996003719889546875 5...
43.327073280914999519496122165406805782645668371836871446878893685521088322305053626456349371063190933 5...
48.005150881167159727942472749427516041686844001144425117775312519814090216416308281330335372305400997 7...
49.773832477672302181916784678563724057723178299676662100781955750433511611151573927873270750740093130 0...
52.970321477714460644147296608880990063825017888821224779900748140317564950304188054137587827094399298 8...
56.446247697063394804367759476706127552782264471716631845450969843958475280274505666903011314274852387 4...
59.347044002602353079653648674992219031098772806466669698122451754746800152699629811838102487074633548 4...
60.831778524609809844259901824524003802910090451219178257101348824808493667294920538430841670394343356 5...
65.112544048081606660875054253183705029348149295166722405966501086675343232668685384416774784438659471 4...
67.079810529494173714478828896522216770107144951745558874196669551694901218956196983530293975085833034 3...
69.546401711173979252926857526554738443012474209602510157324539999663387672274910419533344933178340356 3...
72.067157674481907582522107969826168390480906621456697086683306151488407372399608348363525330412174532 9...

computer algebra: symbolic integration of $\displaystyle\int_0^\infty \frac{e^{-(x-1)^2}}{\sqrt{x}}\,dx$

```
> int(exp(-(x-t)^2)/sqrt(x), x=0..infinity);
```

$$\frac{1}{2}\,\frac{e^{-t^2}\left(-\frac{3(t^2)^{\frac{1}{4}}\pi^{\frac{1}{2}}2^{\frac{1}{2}}e^{\frac{t^2}{2}}K_{\frac{3}{4}}(\frac{t^2}{2})}{t^2}+(t^2)^{\frac{1}{4}}\pi^{\frac{1}{2}}2^{\frac{1}{2}}e^{\frac{t^2}{2}}K_{\frac{7}{4}}(\frac{t^2}{2})\right)}{\pi^{\frac{1}{2}}}$$

```
> subs(t=1,%);
```

$$\frac{1}{2}\,\frac{e^{-1}\left(-3\pi^{\frac{1}{2}}2^{\frac{1}{2}}e^{\frac{1}{2}}K_{\frac{3}{4}}(\frac{1}{2})+\pi^{\frac{1}{2}}2^{\frac{1}{2}}e^{\frac{1}{2}}K_{\frac{7}{4}}(\frac{1}{2})\right)}{\pi^{\frac{1}{2}}}$$

```
> evalf(%);
```

$$0.4118623312$$

```
> evalf(int(exp(-(x-1)^2)/sqrt(x), x=0..infinity));
```

$$1.973732150$$

automated theorem proving: Robbins' conjecture

computers

. . . can in the near future play chess better than a human

. . . can in the near future do mathematics better than a human?

Robbins, 1933: is every Robbins algebra a Boolean algebra?

EQP, 1996: **yes!**

eight days of computer time

one of the very few proofs that has first been found by a computer

not very conceptual: just searches through very many possibilities

interesting research, but currently not relevant for mathematics

the QED manifesto

let's formalize all of mathematics!

QED manifesto, 1994:

> QED is the very tentative title of a project to build a computer system that effectively represents all important mathematical knowledge and techniques.

pamphlet by anonymous group, led by Bob Boyer
utopian vision

proposed many times
never got very far (yet)

the two kinds of computer proof

- **correctness of computer software and hardware**

  (serious branch of computer science: ‘formal methods’)

  statements: big

  proofs: shallow

  computer does the main part of the proof

- **correctness of mathematical theorems**

  (slow and thorough style of doing mathematics, still in its infancy)

  statements: small

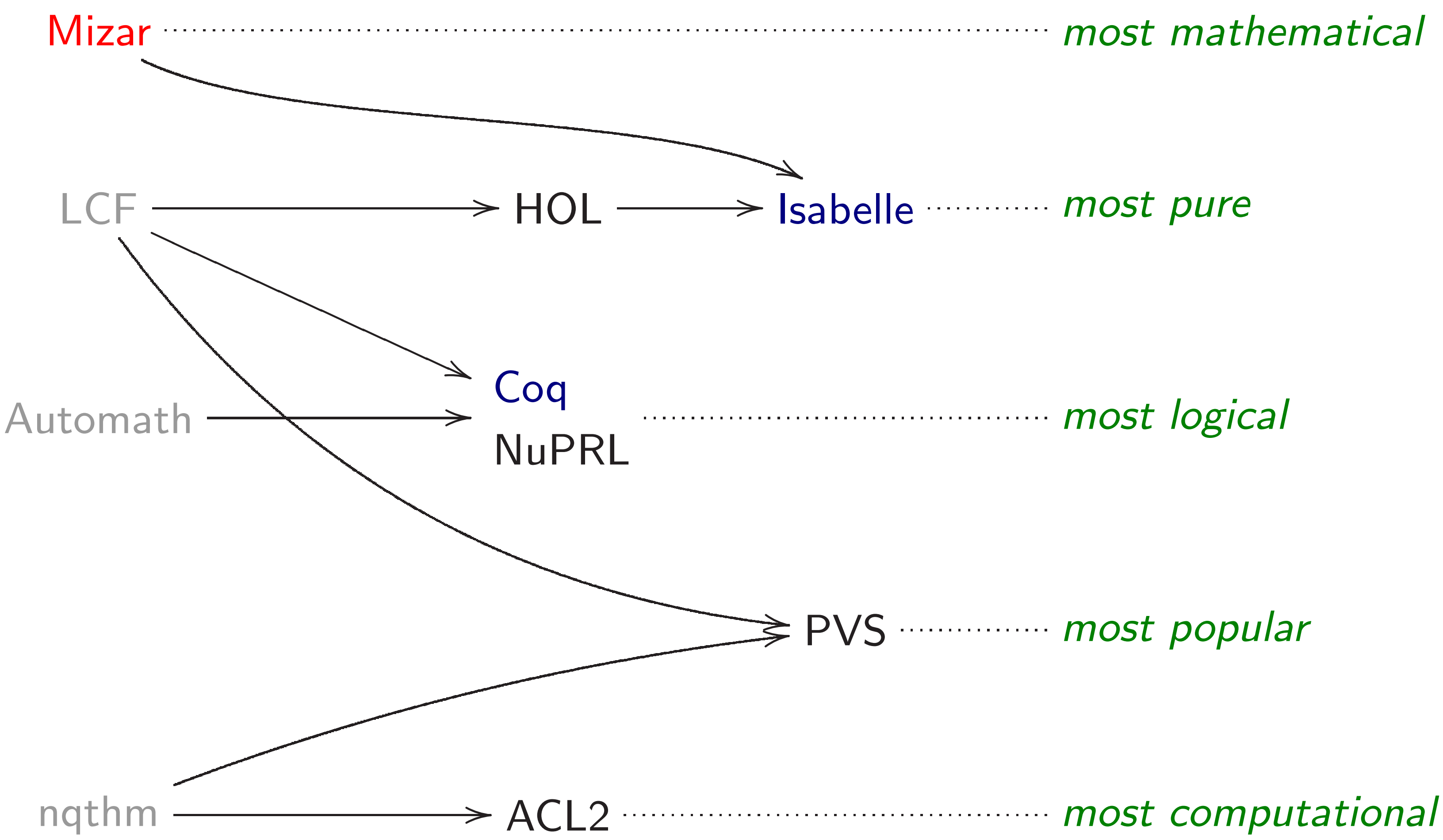
  proofs: deep

  human does the main part of the proof

four prehistorical systems

1968	**Automath**
	Netherlands, de Bruijn
1971	**nqthm**
	US, Boyer & Moore
1972	**LCF**
	UK, Milner
1973	**Mizar**
	Poland, Trybulec

seven current systems for mathematics



a ‘top 100’ of mathematical theorems

1. The Irrationality of the Square Root of 2 $\leftarrow$ *all systems*
2. Fundamental Theorem of Algebra $\leftarrow$ Mizar, HOL, Coq
3. The Denumerability of the Rational Numbers $\leftarrow$ Mizar, HOL, Isabelle
4. Pythagorean Theorem $\leftarrow$ Mizar, HOL, Coq
5. Prime Number Theorem $\leftarrow$ Isabelle
6. Gödel’s Incompleteness Theorem $\leftarrow$ HOL, Coq, nqthm
7. Law of Quadratic Reciprocity $\leftarrow$ Isabelle, nqthm
8. The Impossibility of Trisecting the Angle and Doubling the Cube $\leftarrow$ HOL
9. *The Area of a Circle*
10. Euler’s Generalization of Fermat’s Little Theorem $\leftarrow$ Mizar, HOL, Isabelle

. . . . . .

63% formalized

`http://www.cs.ru.nl/~freek/100/`

(advertisement) the seventeen provers of the world

LNAI 3600

one theorem

seventeen formalisations + explanations about the systems

HOL, Mizar, PVS, Coq, Otter, Isabelle, Agda, ACL2, PhoX, IMPS, Metamath, Theorema, Lego, NuPRL, Ωmega, B method, Minlog

`http://www.cs.ru.nl/~freek/comparison/`

state of the art: recent big formalizations

Prime Number Theorem

Bob Solovay's challenge:

> I suspect that fully formalizing the **usual** proof of the prime number theorem [. . .] is beyond the current capacities of the [formalization] community. Say within the next ten years.

Jeremy Avigad e.a.:

```
"pi(x) == real(card(y. y<=x & y:prime))"
"(%x. pi x * ln (real x) / (real x)) ----> 1"
```

1 megabyte = 30,000 lines = 42 files of Isabelle/HOL

the **elementary** proof by Selberg from 1948

Four Color Theorem

Georges Gonthier:

```
(m : map) (simple_map m) -> (map_colorable (4) m)
```

2.5 megabytes = 60,000 lines = 132 files of Coq 7.3.1

streamlined proof by Robertson, Sanders, Seymour & Thomas from 1996

- contains interesting mathematics as well
  'planar hypermaps'
- very interesting 'own' proof language on top of Coq

  ```
  Move=> x' p'; Elim: p' x' => [|y' p' Hrec] x' //=; Rewrite: ~Hrec.
  By Congr andb; Congr orb; Rewrite: /eqdf (monic2F_eqd (f_finv (Inode g'))).
  ```

- heavily relies on **reflection**
  'this formalization really needs Coq'

Jordan Curve Theorem

Tom Hales:

```
`!C. simple_closed_curve top2 C ==>
   (?A B.  top2 A /\ top2 B /\
     connected top2 A /\ connected top2 B /\
   ~(A = EMPTY) /\ ~(B = EMPTY) /\
    (A INTER B = EMPTY) /\ (A INTER C = EMPTY) /\
        (B INTER C = EMPTY) /\
       (A UNION B UNION C = euclid 2))`
```

2.1 megabytes = 75,000 lines = 15 files of HOL Light

proof through the Kuratowski characterization of planarity

- ‘warming up exercise’ for the Flyspeck project
- beat the Mizar project at formalizing this first
- also uses an ‘own’ proof style

state of the art: current big projects

the continuous lattices formalization

formalize a complete ‘advanced’ mathematics textbook

A Compendium of Continuous Lattices

by Gierz, Hofmann, Keimel, Lawson, Mislove & Scott

> [...] For if not, then $V \subseteq \bigcup\{L \setminus \downarrow v : v \in V\}$; and by quasicompactness and the fact that the $L \setminus \downarrow v$ form a directed family, there would be a $v \in V$ with $V \subseteq L \setminus \downarrow v$, notably $v \notin V$, which is impossible. [...]

project led by Grzegorz Bancerek

about 70% formalized

4.4 megabytes = 127,000 lines = 58 files of Mizar

the Flyspeck project

Kepler in strena sue de nive sexangula, 1661:

is the way one customarily stacks oranges the most efficient way to stack spheres?

Tom Hales, 1998: **yes !**

proof: depends on computer checking
3 gigabytes programs & data, couple of months of computer time

referees say to be 99% certain that everything is correct

FlysPecK project
'Formal Proof of Kepler'

so why did the qed project not take off?

reason one: differences between systems

---

foundations differ very much

set theory $\longleftrightarrow$ type theory $\longleftrightarrow$ higher order logic $\longleftrightarrow$ PRA

classical $\longleftrightarrow$ constructive

extensional $\longleftrightarrow$ intensional

impredicative $\longleftrightarrow$ predicative

choice $\longleftrightarrow$ only countable choice $\longleftrightarrow$ no choice

two utopias simultaneously?

- **formalization of mathematics**
- **doing mathematics in weak logics**

(advertisement) a questionnaire about intuitionism

`http://www.intuitionism.org/`

ten questions about intuitionism

currently: seventeen sets of answers by various people

3. Do you agree that there are only three infinite cardinalities?
7. Do you agree that for any two statements the first implies the second or the second implies the first?

putting systems together

OMDoc

XML standard for encoding of mathematical documents
developed by Michael Kohlhase

can be used both for natural language documents and for formalizations
modularized language architecture

supports both OpenMath and Content MathML encoding of formulas

does not really address semantical differences between systems

Logosphere

converting between the foundations of various systems

project led by Carsten Schürmann

formalize foundations of each system in the Twelf logical framework

translate all formalizations into Twelf

use Twelf to relate those formalizations

systems that are currently supported:

- **first order resolution provers**
- **HOL**
- **NuPRL**
- **PVS**

reason two: why mathematicians are not interested (yet)

the cost is too high...

$$\text{de Bruijn factor} = \frac{\text{size of formalization}}{\text{size of normal text}}$$

question: is this a constant?

experimental: around **4**

$$\text{de Bruijn factor in time} = \frac{\text{time to formalize}}{\text{time to understand}}$$

much larger than **4**

formalizing one textbook page $\approx$ 1 man·week = 40 man·hours

. . . and the gain is too little

l'art pour l'art

Paul Libbrecht in Saarbrücken: 'mental masturbation'

it's not **impossibly** expensive
formalizing all of undergraduate mathematics $\approx$ 140 man·years
the price of about **one** Hollywood movie

but: after formalization we just have a big incomprehensible file
we don't have a good argument yet for spending that money

certainty that it's fully correct?

is that important enough to pay for 140 man·years?

and it does not look like mathematics

most systems: 'proof' = list of tactics = unreadable computer code

even in Mizar and Isar: **still** looks like code

even formulas: too much 'decoding' needed to understand what it says

```
Variable J : interval.      Hypothesis pJ : proper J.
Variable F, G : PartIR.     Hypothesis derG : Derivative J pJ G F.
Let G_inc := Derivative_imp_inc _ _ _ _ derG.

Theorem Barrow : forall a b (H : Continuous_I (Min_leEq_Max a b) F) Ha Hb,
 let Ha' := G_inc a Ha in let Hb' := G_inc b Hb in
 Integral H [=] G b Hb'[-]G a Ha'.
```

$$G' = F \;\Rightarrow\; \int_a^b F(x)\,dx = G(b) - G(a)$$

so what is needed most to promote formalization of mathematics?

- **decision procedures**

  very important, main strength of PVS

- in particular: **computer algebra**

  Macsyma, Maple, Mathematica

  (really: **computer calculus**)

  high school mathematics should be trivial!

$$x = i/n\,,\quad n = m + 1 \quad\vdash\quad n!\cdot x = i\cdot m!$$

$$\frac{k}{n} \geq 0 \quad\vdash\quad \left|\frac{n-k}{n} - 1\right| = \frac{k}{n}$$

$$n \geq 2\,,\quad x = \frac{1}{n+1} \quad\vdash\quad \frac{x}{1-x} < 1$$

second hour:
a tour of Mizar, a proof assistant for mathematics

why is Mizar interesting?

- **a system for mathematicians**
- **the proof language**

  only other system with similar language: Isabelle/Isar
- **many other interesting ideas**
  - type system

    soft typing

    ‘attributes’

    multiple inheritance between structure types
  - expression syntax

    type directed overloading

    bracket-like operators

    arbitrary ASCII strings for operators

example formalizations

example: Coq version

```
Definition ge (n m : nat) : Prop :=
  exists x : nat, n = m + x.
Infix ">=" := ge : nat_scope.

Lemma ge_trans :
  forall n m p : nat, n >= m -> m >= p -> n >= p.
Proof.
 unfold ge. intros n m p H H0.
 elim H. clear H. intros x H1.
 elim H0. clear H0. intros x0 H2.
 exists (x0 + x).
 rewrite plus_assoc. rewrite <- H2. auto.
Qed.
```

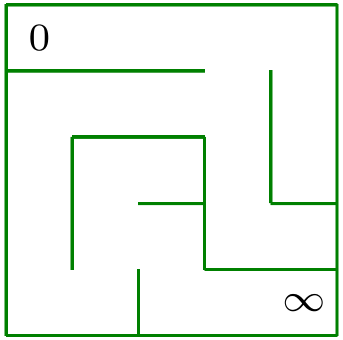
example: Mizar version

```
reserve n,m,p,x,x0 for natural number;

definition let n,m;
 pred n >= m means :ge: ex x st n = m + x;
end;

theorem ge_trans: n >= m & m >= p implies n >= p
proof
 assume that H: n >= m and H0: m >= p;
 consider x such that H1: n = m + x by H,ge;
 consider x0 such that H2: m = p + x0 by H0,ge;
 n = p + (x + x0) by H1,H2;
 hence n >= p by ge;
end;
```

procedural versus declarative



- **procedural**

  E E S E N E S S S W W W S E E E

  HOL, Isabelle, Coq, NuPRL, PVS

- **declarative**

  (0,0) (1,0) (2,0) (3,0) (3,1) (2,1) (1,1) (0,1) (0,2) (0,3) (0,4) (1,4) (1,3) (2,3) (2,4) (3,4) (4,4)

  Mizar, Isabelle

another small example

> *If every poor person has a rich father,*
> *then there is a rich person with a rich grandfather.*

```
 assume that
A1: for x st x is poor holds father(x) is rich and
A2: not ex x st x is rich & father(father(x)) is rich;
 consider p being person;
 now let x;
  x is poor or father(father(x)) is poor by A2;
  hence father(x) is rich by A1;
 end;
 then father(p) is rich & father(father(father(p))) is rich;
 hence contradiction by A2;
```

Theorem. *There are irrational numbers x and y such that x^y is rational.*

Proof. We have the following calculation

$$\left(\sqrt{2}^{\sqrt{2}}\right)^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\cdot\sqrt{2}} = \sqrt{2}^{2} = 2$$

which is rational. Furthermore Pythagoras showed that $\sqrt{2}$ is irrational. Now there are two cases:

- Either $\sqrt{2}^{\sqrt{2}}$ is rational. Then take $x = y = \sqrt{2}$.
- Or $\sqrt{2}^{\sqrt{2}}$ is irrational. In that case take $x = \sqrt{2}^{\sqrt{2}}$ and $y = \sqrt{2}$. And by the above calculation then $x^y = 2$, which is rational. $\square$

lemmas used in the proof

`AXIOMS:22`	$x \leq y \wedge y \leq z \Rightarrow x \leq z$
`INT_2:44`	2 is prime
`IRRAT_1:1`	p is prime $\Rightarrow \sqrt{p} \notin \mathbb{Q}$
`POWER:38`	$a > 0 \Rightarrow (a^b)^c = a^{bc}$
`SQUARE_1:def 3`	$x^2 = x \cdot x$
`SQUARE_1:def 4`	$0 \leq a \Rightarrow (x = \sqrt{a} \Leftrightarrow 0 \leq x \wedge x^2 = a)$
`SQUARE_1:84`	$1 < \sqrt{2}$
`POWER:53`	`‘a to_power 2 = a^2’`

DEMO

```
reserve x,y for real number;

theorem ex x,y st x is irrational & y is irrational &
  x to_power y is rational
proof
 set r = sqrt 2;
C: r > 0 by SQUARE_1:84,AXIOMS:22;
B1: r is irrational by INT_2:44,IRRAT_1:1;
B2: (r to_power r) to_power r
   = r to_power (r * r) by C,POWER:38
  .= r to_power r^2 by SQUARE_1:def 3
  .= r to_power 2 by SQUARE_1:def 4
  .= r^2 by POWER:53
  .= 2 by SQUARE_1:def 4;
 per cases;
 suppose
A1: r to_power r is rational;
  take x = r, y = r;
  thus thesis by A1,B1;
 end;
 suppose
A2: r to_power r is irrational;
  take x = r to_power r, y = r;
  thus thesis by A2,B1,B2;
 end;
end;
```

example of how Mizar is like English

Hardy & Wright, An Introduction to the Theory of Numbers

Theorem 43 (Pythagoras' theorem). $\sqrt{2}$ is irrational.

The traditional proof ascribed to Pythagoras runs as follows. If $\sqrt{2}$ is rational, then the equation

$$a^2 = 2b^2 \qquad (4.3.1)$$

is soluble in integers a, b with $(a, b) = 1$. Hence a^2 is even, and therefore a is even. If $a = 2c$, then $4c^2 = 2b^2$, $2c^2 = b^2$, and b is also even, contrary to the hypothesis that $(a, b) = 1$. $\square$

Mizar language approximation of this text

```
theorem Th43: sqrt 2 is irrational
proof
  assume sqrt 2 is rational;
  consider a,b such that
4_3_1: a^2 = 2*b^2 and
    a,b are_relative_prime;
  a^2 is even;
  a is even;
  consider c such that a = 2*c;
  4*c^2 = 2*b^2;
  2*c^2 = b^2;
  b is even;
  thus contradiction;
end;
```

full Mizar

```
theorem Th43: sqrt 2 is irrational
proof
  assume sqrt 2 is rational;
  then consider a, b such that
A1: b <> 0 and
A2: sqrt 2 = a/b and
A3: a, b are_relative_prime by Def1;
A4: b^2 <> 0 by A1, SQUARE_1:73;
  2 = (a/b)^2 by A2, SQUARE_1:def 4
    .= a^2/b^2 by SQUARE_1:69;
  then
4_3_1: a^2 = 2 * b^2 by A4, REAL_1:43;
  a^2 is even by 4_3_1, ABIAN:def 1;
  then
A5: a is even by PYTHTRIP:2;
:: continue in next column
  then consider c such that
A6: a = 2 * c by ABIAN:def 1;
A7: 4 * c^2 =(2 * 2) * c^2
      .= 2^2 * c^2 by SQUARE_1:def 3
      .= 2 * b^2 by A6, 4_3_1, SQUARE_1:68;
  2 * (2 * c^2) = (2 * 2) * c^2 by AXIOMS:16
      .= 2 * b^2 by A7;
  then 2 * c^2 = b^2 by REAL_1:9;
  then b^2 is even by ABIAN:def 1;
  then b is even by PYTHTRIP:2;
  then 2 divides a & 2 divides b by A5, Def2;
  then
A8: 2 divides a gcd b by INT_2:33;
  a gcd b = 1 by A3, INT_2:def 4;
  hence contradiction by A8, INT_2:17;
end;
```

some explanations about Mizar

the proof language

---

forward reasoning

$\langle$statement$\rangle$ **by** $\langle$references$\rangle$
$\langle$statement$\rangle$ **proof** $\langle$steps$\rangle$ **end**

natural deduction

thus $\langle$statement$\rangle$	$\longrightarrow$	closes the proof
assume $\langle$statement$\rangle$	$\longrightarrow$	$\longrightarrow$-introduction
let $\langle$variable$\rangle$	$\longrightarrow$	$\forall$-introduction
thus $\langle$statement$\rangle$	$\longrightarrow$	$\wedge$-introduction
consider $\langle$variable$\rangle$ **such that** $\langle$statement$\rangle$	$\longrightarrow$	$\exists$-elimination
take $\langle$term$\rangle$	$\longrightarrow$	$\exists$-introduction
per cases; suppose $\langle$statement$\rangle$**;** ...	$\longrightarrow$	$\vee$-elimination

‘semantics’ ?

Mizar is just first order predicate logic + set theory

Mizar proofs are just Fitch-style natural deduction

but:

- Mizar variables have types. . .

  . . . and these types are quite powerful!

- Mizar has ‘second-order theorems’ called schemes

- Mizar defines function symbols using something like Church’s ι operator (‘unique choice’)

Tarski-Grothendieck set theory

`TARSKI:def 3`	$X \subseteq Y \Leftrightarrow (\forall x.\, x \in X \Rightarrow x \in Y)$
`TARSKI:def 5`	$\langle x, y\rangle = \{\{x, y\}, \{x\}\}$
`TARSKI:def 6`	$X \sim Y \Leftrightarrow \exists Z.\, (\forall x.\, x \in X. \Rightarrow \exists y.\, y \in Y \wedge \langle x, y\rangle \in Z) \wedge$ $(\forall y.\, y \in Y. \Rightarrow \exists x.\, x \in X \wedge \langle x, y\rangle \in Z) \wedge$ $(\forall x \forall y \forall z \forall u.\, \langle x, y\rangle \in Z \wedge \langle z, u\rangle \in Z \Rightarrow (x = z \Leftrightarrow y = u))$

`TARSKI:def 1`	$x \in \{y\} \Leftrightarrow x = y$
`TARSKI:def 2`	$x \in \{y, z\} \Leftrightarrow x = y \vee x = z$
`TARSKI:def 4`	$x \in \bigcup X \Leftrightarrow \exists Y.\, x \in Y \wedge Y \in X$
`TARSKI:2`	$(\forall x.\, x \in X \Leftrightarrow x \in Y) \Rightarrow X = Y$
`TARSKI:7`	$x \in X \Rightarrow \exists Y.\, Y \in X \wedge \neg \exists x.\, x \in X \wedge x \in Y$
`TARSKI:sch 1`	$(\forall x\, \forall y\, \forall z.\, P[x, y] \wedge P[x, z] \Rightarrow y = z) \Rightarrow$ $(\exists X.\, \forall x.\, x \in X \Leftrightarrow \exists y.\, y \in A \wedge P[y, x])$
`TARSKI:9`	$\exists M.\, N \in M \wedge (\forall X \forall Y.\, X \in M \wedge Y \subseteq X \Rightarrow Y \in M) \wedge$ $(\forall X.\, X \in M \Rightarrow \exists Z.\, Z \in M \wedge \forall Y.\, Y \subseteq X \Rightarrow Y \in Z) \wedge$ $(\forall X.\, X \subseteq M \Rightarrow X \sim M \vee X \in M)$

types!

Mizar is based on set theory but it is a typed system

Mizar types are soft types:

$$M : N(t_1, \ldots, t_n)$$

should really be read as a **predicate**

$$N(t_1, \ldots, t_n, M)$$

This means that:

- one Mizar term can have many different types at the same time
- a Mizar typing can be used as a logical formula!

```
let x be Real;    ⟷    assume not x is Nat;
```

types! (continued)

think of Mizar types as predicates that the system keeps track of for you

Mizar types are used for three things:

- **type based overloading**

`x + y`	sum of two numbers
`X + Y`	adding the elements of two sets
`X + y`	mixing these two things
`v + w`	sum of two elements of a vector space
`I + J`	sum of two ideals in a ring
`x + y`	'join' of two elements of a lattice
`p + i`	adding an offset to a pointer

- **inferring implicit arguments**
- **automatic inference of propositions**

types! (continued)

- Mizar has dependent types
  (much like in all the other dependent type systems)
- Mizar has a subtype relation
  every type except the type '`set`' has a supertype
- Mizar has 'type modifiers' called attributes
  a type can be prefixed with one or more adjectives
  an adjective is either an attribute or the negation of an attribute
  (behaves like intersection types)

`non empty` `finite` `Subset of NAT`

notation

any ASCII string can be used for a Mizar operator

```
  func ].a,b.] -> Subset of REAL means
:: MEASURE5:def 3
   for x being R_eal holds
    x in it iff (a <' x & x <=' b & x in REAL);

  pred a,b are_convergent<=1_wrt R means
:: REWRITE1:def 9
   ex c being set st ([a,c] in R or a = c) & ([b,c] in R or b = c);
```

Mizar in the world

Mizar Mathematical Library

the biggest library of formalized mathematics

49,588	lemmas
1,820,879	lines of 'code'
64	megabytes
165	'authors'
912	'articles'

Mizar, the program

- implemented in Delphi Pascal/Free Pascal
- source not freely available, but

write Mizar 'article'

$\downarrow$

become member of Association of Mizar Users

$\downarrow$

get source

- no small proof checking 'kernel'
  correctness of Mizar check depends on correctness of whole program
- users can not automate proofs inside the system

Mizar Mathematical Library

```
theorem :: RUSUB_2:35
for V being RealUnitarySpace, W being Subspace of V,
 L being Linear_Compl of W holds
  V is_the_direct_sum_of L,W & V is_the_direct_sum_of W,L;
```

Formalized Mathematics

(35) Let V be a real unitary space, W be a subspace of V, and L be a linear complement of W. Then V is the direct sum of L and W and the direct sum of W and L.

some reasons to prefer Mizar over Isar

- the set theory of Mizar is much more powerful and expressive than the HOL logic of Isabelle/HOL
- Mizar is much more able to talk about abstract mathematics, and in particular about algebraic structures, with nice notation
- dependent types are way cool

some reasons to prefer Isar over Mizar

- Isabelle gives you an interactive system
- Isabelle allows you to mix declarative and procedural proof
- Isabelle has much more possibilities of automation
- Isabelle allows you to define binders

is Mizar a difficult system?

no, not difficult at all!

Mizar is about as complex as the Pascal programming language
(proof assistants tend to resemble their implementation language)

reasons that people sometimes think Mizar is a complex language

- lack of proper documentation
- natural language-like syntax

extro

gazing into the crystal ball

Henk's futuristic QED questions

- will proof assistants ever become common among mathematicians?
- if so: when will this happen?
  - the most optimistic answer: it already is here!
  - the experienced user's answer: fifty years

but what do **you** expect?