Computer-Understandable Mathematics

Josef Urban

Czech Technical University

Outline

What Is Formal (Computer-Understandable) Mathematics?

Automated Theorem Proving

Examples of Formal Proof

What Has Been Formalized?

Foundations and Other Issues

Flyspeck

Who Am I To Tell You?

- Original a student of math interested in automation of reasoning
- Wanted to learn math reasoning from large math libraries
- Wrote some formalizations
- Involved with several formal systems/projects
- Today mostly working on AI and automated reasoning over large libraries
- By no means an expert on every system I will talk about! (nobody is)

What Is Formal (Computer-Undertandable) Mathematics

- Conceptually very simple:
- Write all your axioms and theorems so that computer understands them
- Write all your inference rules so that computer understands them
- Use the computer to check that your proofs follow the rules
- But in practice, it turns out not to be so simple

OK, So Where Are The Hard Parts?

- Precise computer encoding of the mathematical language
  - How do you exactly encode a graph, a category, real numbers, $\mathbb{R}^n$, division, differentiation, computation
  - Lots of representation issues
  - Fluent switching between different representations
- Precise computer understanding of the mathematical proofs
  - “the following reasoning holds up to a set of measure zero”
  - “use the method introduced in the above pararaph”
  - “subdivide and jiggle the triangulation so that ...”
  - “the rest is a standard diagonalization argument”

Further Issues

- What foundations? (Set theory, higher-order logic, type theory, ...)
- What input syntax?
- What automation methods?
- What search methods?
- What presentation methods?

Digression: Automated Theorem Proving

Propositional – SATisfiability solving

- DPLL- Davis–Putnam–Logemann–Loveland algorithm
- choosing a literal
- assigning a truth value to it
- simplifying the formula
- recursively check if the simplified formula is satisfiable
- unit propagation
- Pure literal elimination
- clause learning
- basis of many more-involved algorithms, hardware checking, model checking, etc.
- systems: Minisat, Glucose, ...

Satisfiability Modulo Theories – SMT

- add theories like arithmetics, bit-arrays, etc.
- works like SAT, but simplifies the theory literals whenever possible
- very useful for software and hardware verification
- today also limited treatment of quantifiers (first-order logic):
- instantiate first-order terms by guessing their instances
- often incomplete for first-order logic
- systems: Z3, CVC4, Alt-Ergo, ...

First Order – Automated Theorem Proving (ATP)

- try to infer conjecture C from axioms Ax: $Ax \vdash C$
- most classical methods proceed by refutation: $Ax \wedge \neg C \vdash \bot$
- $Ax \wedge \neg C$ are turned into *clauses*: universally quantified disjunctions of atomic formulas and their negations
- *skolemization* is used to remove existential quantifiers
- strongest methods: resolution (generalized modus ponens) on clauses:
- $\neg man(X) \vee mortal(X), man(socrates) \vdash mortal(socrates)$
- resolution/superposition (equational) provers generate inferences, looking for the contradiction (empty clause)
- main problem: combinatorial explosion
- systems: Vampire, E, SPASS, Prover9, leanCoP, Waldmeister

Using First Order Automated Theorem Proving (ATP)

- 1996: Bill McCune proof of Robbins conjecture (Robbins algebras are Boolean algebras)
- Robbins conjecture unsolved for 50 years by mathematicians like Tarski
- ATP has currently very limited use for proving new conjectures
- mainly in very specialized algebraic domains: Veroff, Kinyon and Prover9
- however ATP has become very useful in Interactive Theorem Proving

Interactive Theorem Proving – Formal Verification

- verify complicated mathematical proofs
- verify complicated hardware and software designs
- operating systems, compilers, protocols, etc.
- very secure proof-checking kernel implementation
- enhanced by more advanced tactics for various types of goals (e.g., arithmetical solvers)
- recently a lot of progress and large finished projects – Flyspeck

End of Digression

Irrationality of 2 (informal text)

tiny proof from Hardy & Wright:

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

$$a^2 = 2b^2 \tag{4.3.1}$$

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 $(a, b) = 1$. □

Irrationality of 2 (Formal Proof Sketch)

exactly the same text in Mizar syntax:

```
theorem Th43: :: Pythagoras' theorem
  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;
```

Irrationality of 2 (checkable formalization)

full Mizar formalization (for details, see: http://mizar.cs.ualberta.ca/~mptp/mml5.29.1227/html/irrat_1.html)

```
theorem Th43: :: Pythagoras' theorem
  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;
  then a^2 is even by ABIAN:def 1;
  then
A5: a is even by PYTHTRIP:2;
  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;
```

Irrationality of 2 (checkable formalization)

full Mizar formalization (for details, see: `http://mizar.cs.ualberta.ca/~mptp/mml5.29.1227/html/irrat_1.html`)

```
theorem Th43: :: Pythagoras' theorem
  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;
  then a^2 is even by ABIAN:def 1;
  then
A5: a is even by PYTHTRIP:2;
  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;
```

Irrationality of 2 in HOL Light

```
let SQRT_2_IRRATIONAL = prove
 (`~rational(sqrt(&2))`,
  SIMP_TAC[rational; real_abs; SQRT_POS_LE; REAL_POS] THEN
  REWRITE_TAC[NOT_EXISTS_THM] THEN REPEAT GEN_TAC THEN
  DISCH_THEN(CONJUNCTS_THEN2 ASSUME_TAC MP_TAC) THEN
  SUBGOAL_THEN `~((&p / &q) pow 2 = sqrt(&2) pow 2)`
    (fun th -> MESON_TAC[th]) THEN
  SIMP_TAC[SQRT_POW_2; REAL_POS; REAL_POW_DIV] THEN
  ASM_SIMP_TAC[REAL_EQ_LDIV_EQ; REAL_OF_NUM_LT; REAL_POW_LT;
               ARITH_RULE `0 < q <=> ~(q = 0)`] THEN
  ASM_MESON_TAC[NSQRT_2; REAL_OF_NUM_POW; REAL_OF_NUM_MUL; REAL_OF_NUM_EQ]);;
```

Irrationality of 2 in Isabelle/HOL

```
theorem sqrt2_not_rational:
  "sqrt (real 2) ∉ ℚ"
proof
  assume "sqrt (real 2) ∈ ℚ"
  then obtain m n :: nat where
    n_nonzero: "n ≠ 0" and sqrt_rat: "¦sqrt (real 2)¦ = real m / real n"
    and lowest_terms: "gcd m n = 1" ..
  from n_nonzero and sqrt_rat have "real m = ¦sqrt (real 2)¦ * real n" by simp
  then have "real (m²) = (sqrt (real 2))² * real (n²)"
    by (auto simp add: power2_eq_square)
  also have "(sqrt (real 2))² = real 2" by simp
  also have "... * real (m²) = real (2 * n²)" by simp
  finally have eq: "m² = 2 * n²" ..
  hence "2 dvd m²" ..
  with two_is_prime have dvd_m: "2 dvd m" by (rule prime_dvd_power_two)
  then obtain k where "m = 2 * k" ..
  with eq have "2 * n² = 2² * k²" by (auto simp add: power2_eq_square mult_ac)
  hence "n² = 2 * k²" by simp
  hence "2 dvd n²" ..
  with two_is_prime have "2 dvd n" by (rule prime_dvd_power_two)
  with dvd_m have "2 dvd gcd m n" by (rule gcd_greatest)
  with lowest_terms have "2 dvd 1" by simp
  thus False by arith
qed
```

Irrationality of 2 in Coq

```
Theorem irrational_sqrt_2: irrational (sqrt 2%nat).
intros p q H H0; case H.
apply (main_thm (Zabs_nat p)).
replace (Div2.double (q * q)) with (2 * (q * q));
 [idtac | unfold Div2.double; ring].
case (eq_nat_dec (Zabs_nat p * Zabs_nat p) (2 * (q * q))); auto; intros H1.
case (not_nm_INR _ _ H1); (repeat rewrite mult_INR).
rewrite <- (sqrt_def (INR 2)); auto with real.
rewrite H0; auto with real.
assert (q <> 0%R :> R); auto with real.
field; auto with real; case p; simpl; intros; ring.
Qed.
```

Irrationality of 2 in Metamath

```
${
    $d x y $.
    $( The square root of 2 is irrational. $)
    sqr2irr $p |- ( sqr ` 2 ) e/ QQ $=
      ( vx vy c2 csqr cfv cq wnel wcel wn cv cdiv co wceq cn wrex cz cexp
      cmulc sqr2irrlem3 sqr2irrlem5 bi2rexa mtbir cc0 clt wbr wa wi wb nngt0t
      adantr cr ax0re ltmuldivt mp3an1 nnret zret syl2an mpd ancoms 2re 2pos
      sqrgt0i breq2 mpbii syl5bir cc nncnt mulzer2t syl breq1d adantl sylibd
      exp r19.23adv anc2li elnnz syl6ibr impac r19.22i2 mto elq df-nel mpbir )
      CDEZFGWDFHZIWEWDAJZBJZKLZMZBNOZAPOZWKWJANOZWLWFCQLCWGCQLRLMZBNOANOABSWIWM
      ABNNWFWGTUAUBWJWJAPNWFPHZWJWFNHZWNWJWNUCWFUDUEZUFWOWNWJWPWNWIWPBNWNWGNHZW
      IWPUGWNWQUFZWIUCWGRLZWFUDUEZWPWRWTUCWHUDUEZWIWQWNWTXAUHZWQWNUFUCWGUDUEZXB
      WQXCWNWGUIUJWGUKHZWFUKHZXCXBUGZWQWNUCUKHXDXEXFULUCWGWFUMUNWGUOWFUPUQURUSW
      IUCWDUDUEXACUTVAVBWDWHUCUDVCVDVEWQWTWPUHWNWQWSUCWFUDWQWGVFHWSUCMWGVGWGVHV
      IVJVKVLVMVNVOWFVPVQVRVSVTABWDWAUBWDFWBWC $.
      $( [8-Jan-02] $)
  $}
```

Irrationality of 2 in Metamath Proof Explorer

sqr2irr - Metamath Proof Explorer - Chromium

sqr2irr - Metamat ×

us.metamath.org/mpegif/sqr2irr.html

Proof of Theorem sqr2irr

Step	Hyp	Ref	Expression
1		sqr2irrlem3 10838	5 $\vdash \neg \exists x \in \mathbb{N} \exists y \in \mathbb{N} (x\uparrow 2) = (2 \cdot (y\uparrow 2))$
2		sqr2irrlem5 10840	6 $\vdash ((x \in \mathbb{N} \land y \in \mathbb{N}) \to ((\surd\text{‘}2) = (x / y) \leftrightarrow (x\uparrow 2) = (2 \cdot (y\uparrow 2))))$
3	2	2rexbiia 2329	5 $\vdash (\exists x \in \mathbb{N} \exists y \in \mathbb{N} (\surd\text{‘}2) = (x / y) \leftrightarrow \exists x \in \mathbb{N} \exists y \in \mathbb{N} (x\uparrow 2) = (2 \cdot (y\uparrow 2)))$
4	1, 3	mtbir 288	4 $\vdash \neg \exists x \in \mathbb{N} \exists y \in \mathbb{N} (\surd\text{‘}2) = (x / y)$
5		2re 8838	12 $\vdash 2 \in \mathbb{R}$
6		2pos 8849	12 $\vdash 0 < 2$
7	5, 6	sqrgt0ii 10213	11 $\vdash 0 < (\surd\text{‘}2)$
8		breq2 3595	11 $\vdash ((\surd\text{‘}2) = (x / y) \to (0 < (\surd\text{‘}2) \leftrightarrow 0 < (x / y)))$
9	7, 8	mpbii 200	10 $\vdash ((\surd\text{‘}2) = (x / y) \to 0 < (x / y))$
10		zre 9029	12 $\vdash (x \in \mathbb{Z} \to x \in \mathbb{R})$
11	10	adantr 444	11 $\vdash ((x \in \mathbb{Z} \land y \in \mathbb{N}) \to x \in \mathbb{R})$
12		nnre 8788	12 $\vdash (y \in \mathbb{N} \to y \in \mathbb{R})$
13	12	adantl 445	11 $\vdash ((x \in \mathbb{Z} \land y \in \mathbb{N}) \to y \in \mathbb{R})$
14		nngt0 8807	12 $\vdash (y \in \mathbb{N} \to 0 < y)$
15	14	adantl 445	11 $\vdash ((x \in \mathbb{Z} \land y \in \mathbb{N}) \to 0 < y)$
16		gt0div 8683	11 $\vdash ((x \in \mathbb{R} \land y \in \mathbb{R} \land 0 < y) \to (0 < x \leftrightarrow 0 < (x / y)))$
17	11, 13, 15, 16	syl3anc 1145	10 $\vdash ((x \in \mathbb{Z} \land y \in \mathbb{N}) \to (0 < x \leftrightarrow 0 < (x / y)))$
18	9, 17	syl5ibr 210	9 $\vdash ((x \in \mathbb{Z} \land y \in \mathbb{N}) \to ((\surd\text{‘}2) = (x / y) \to 0 < x))$
19		simpl 436	9 $\vdash ((x \in \mathbb{Z} \land y \in \mathbb{N}) \to x \in \mathbb{Z})$
20	18, 19	jctild 522	8 $\vdash ((x \in \mathbb{Z} \land y \in \mathbb{N}) \to ((\surd\text{‘}2) = (x / y) \to (x \in \mathbb{Z} \land 0 < x)))$
21		elnnz 9035	8 $\vdash (x \in \mathbb{N} \leftrightarrow (x \in \mathbb{Z} \land 0 < x))$
22	20, 21	syl6ibr 216	7 $\vdash ((x \in \mathbb{Z} \land y \in \mathbb{N}) \to ((\surd\text{‘}2) = (x / y) \to x \in \mathbb{N}))$
23	22	rexlimdva 2414	6 $\vdash (x \in \mathbb{Z} \to (\exists y \in \mathbb{N} (\surd\text{‘}2) = (x / y) \to x \in \mathbb{N}))$
24	23	impac 598	5 $\vdash ((x \in \mathbb{Z} \land \exists y \in \mathbb{N} (\surd\text{‘}2) = (x / y)) \to (x \in \mathbb{N} \land \exists y \in \mathbb{N} (\surd\text{‘}2) = (x / y)))$
25	24	reximi2 2396	4 $\vdash (\exists x \in \mathbb{Z} \exists y \in \mathbb{N} (\surd\text{‘}2) = (x / y) \to \exists x \in \mathbb{N} \exists y \in \mathbb{N} (\surd\text{‘}2) = (x / y))$
26	4, 25	mto 165	3 $\vdash \neg \exists x \in \mathbb{Z} \exists y \in \mathbb{N} (\surd\text{‘}2) = (x / y)$
27		elq 9308	3 $\vdash ((\surd\text{‘}2) \in \mathbb{Q} \leftrightarrow \exists x \in \mathbb{Z} \exists y \in \mathbb{N} (\surd\text{‘}2) = (x / y))$
28	26, 27	mtbir 288	2 $\vdash \neg (\surd\text{‘}2) \in \mathbb{Q}$
29		df-nel 2210	2 $\vdash ((\surd\text{‘}2) \notin \mathbb{Q} \leftrightarrow \neg (\surd\text{‘}2) \in \mathbb{Q})$
30	28, 29	mpbir 198	1 $\vdash (\surd\text{‘}2) \notin \mathbb{Q}$

Colors of variables: wff set class

What Has Been Formalized?

top 100 of interesting theorems/proofs
(Paul & Jack Abad, 1999, tracked by Freek Wiedijk)

1. $\sqrt{2} \notin \mathbb{Q}$
2. fundamental theorem of algebra
3. $|\mathbb{Q}| = \aleph_0$
4. 

$\Rightarrow a^2 + b^2 = c^2$
5. $\pi(x) \sim \frac{x}{\ln x}$
6. Gödel's incompleteness theorem
7. $\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}$
8. impossibility of trisecting the angle and doubling the cube

$\vdots$

32. four color theorem
33. Fermat's last theorem

$\vdots$

99. Buffon needle problem
100. Descartes rule of signs

What Has Been Formalized?

top 100 of interesting theorems/proofs
(Paul & Jack Abad, 1999, tracked by Freek Wiedijk)

1. $\sqrt{2} \notin \mathbb{Q}$
2. fundamental theorem of algebra
3. $|\mathbb{Q}| = \aleph_0$
4. 

$\Rightarrow a^2 + b^2 = c^2$
5. $\pi(x) \sim \frac{x}{\ln x}$
6. Gödel's incompleteness theorem
7. $\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}$
8. impossibility of trisecting the angle and doubling the cube

⋮

32. four color theorem
33. Fermat's last theorem

⋮

99. Buffon needle problem
100. Descartes rule of signs

all together	88%
HOL Light	86%
Mizar	57%
Isabelle	52%
Coq	49%
ProofPower	42%
Metamath	24%
ACL2	18%
PVS	16%

What Has Been Formalized?

top 100 of interesting theorems/proofs
(Paul & Jack Abad, 1999, tracked by Freek Wiedijk)

1. $\sqrt{2} \notin \mathbb{Q}$
2. fundamental theorem of algebra
3. $|\mathbb{Q}| = \aleph_0$
4. 

$\Rightarrow a^2 + b^2 = c^2$
5. $\pi(x) \sim \frac{x}{\ln x}$
6. Gödel's incompleteness theorem
7. $\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}$
8. impossibility of trisecting the angle and doubling the cube

$\vdots$

32. four color theorem
33. Fermat's last theorem

$\vdots$

99. Buffon needle problem
100. Descartes rule of signs

all together	88%
HOL Light	86%
Mizar	57%
Isabelle	52%
Coq	49%
ProofPower	42%
Metamath	24%
ACL2	18%
PVS	16%

What Has Been Formalized?

top 100 of interesting theorems/proofs
(Paul & Jack Abad, 1999, tracked by Freek Wiedijk)

1. $\sqrt{2} \notin \mathbb{Q}$
2. fundamental theorem of algebra
3. $|\mathbb{Q}| = \aleph_0$
4. 

$\Rightarrow a^2 + b^2 = c^2$
5. $\pi(x) \sim \frac{x}{\ln x}$
6. Gödel's incompleteness theorem
7. $\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}$
8. impossibility of trisecting the angle and doubling the cube

$\vdots$

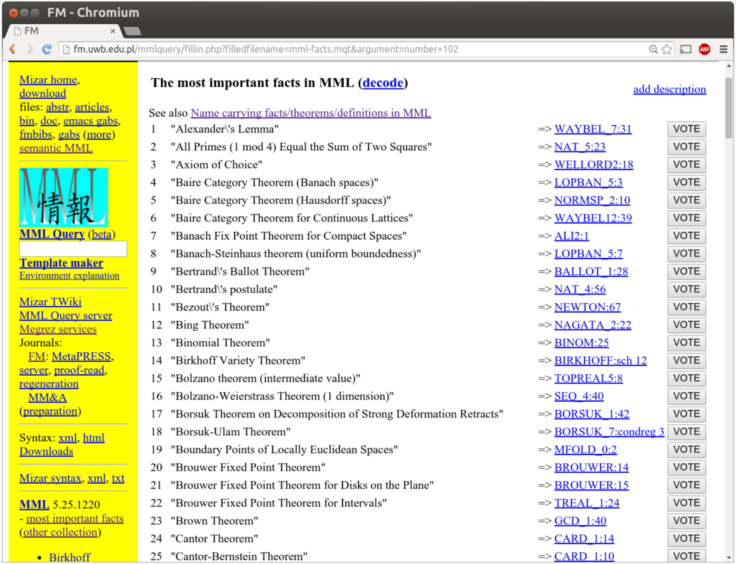
32. four color theorem
33. Fermat's last theorem

$\vdots$

99. Buffon needle problem
100. Descartes rule of signs

all together	88%
HOL Light	86%
Mizar	57%
Isabelle	52%
Coq	49%
ProofPower	42%
Metamath	24%
ACL2	18%
PVS	16%

Named Theorems in the Mizar Library



FM - Chromium

FM

fm.uwb.edu.pl/mmlquery/fillin.php?filledfilename=mml-facts.mqt&argument=number+102

Mizar home, download
files: abstr, articles, bin, doc, emacs gabs, fmbibs, gabs (more)
semantic MML

MML Query (beta)

Template maker
Environment explanation

Mizar TWiki
MML Query server
Megrez services
Journals:
FM: MetaPRESS, server, proof-read, regeneration
MM&A (preparation)

Syntax: xml, html
Downloads

Mizar syntax, xml, txt

MML 5.25.1220
- most important facts (other collection)

- Birkhoff

The most important facts in MML (decode)

add description

See also Name carrying facts/theorems/definitions in MML

1	"Alexander\'s Lemma"	=> WAYBEL_7:31	VOTE
2	"All Primes (1 mod 4) Equal the Sum of Two Squares"	=> NAT_5:23	VOTE
3	"Axiom of Choice"	=> WELLORD2:18	VOTE
4	"Baire Category Theorem (Banach spaces)"	=> LOPBAN_5:3	VOTE
5	"Baire Category Theorem (Hausdorff spaces)"	=> NORMSP_2:10	VOTE
6	"Baire Category Theorem for Continuous Lattices"	=> WAYBEL12:39	VOTE
7	"Banach Fix Point Theorem for Compact Spaces"	=> ALI2:1	VOTE
8	"Banach-Steinhaus theorem (uniform boundedness)"	=> LOPBAN_5:7	VOTE
9	"Bertrand\'s Ballot Theorem"	=> BALLOT_1:28	VOTE
10	"Bertrand\'s postulate"	=> NAT_4:56	VOTE
11	"Bezout\'s Theorem"	=> NEWTON:67	VOTE
12	"Bing Theorem"	=> NAGATA_2:22	VOTE
13	"Binomial Theorem"	=> BINOM:25	VOTE
14	"Birkhoff Variety Theorem"	=> BIRKHOFF:sch 12	VOTE
15	"Bolzano theorem (intermediate value)"	=> TOPREAL5:8	VOTE
16	"Bolzano-Weierstrass Theorem (1 dimension)"	=> SEQ_4:40	VOTE
17	"Borsuk Theorem on Decomposition of Strong Deformation Retracts"	=> BORSUK_1:42	VOTE
18	"Borsuk-Ulam Theorem"	=> BORSUK_7:condreg 3	VOTE
19	"Boundary Points of Locally Euclidean Spaces"	=> MFOLD_0:2	VOTE
20	"Brouwer Fixed Point Theorem"	=> BROUWER:14	VOTE
21	"Brouwer Fixed Point Theorem for Disks on the Plane"	=> BROUWER:15	VOTE
22	"Brouwer Fixed Point Theorem for Intervals"	=> TREAL_1:24	VOTE
23	"Brown Theorem"	=> GCD_1:40	VOTE
24	"Cantor Theorem"	=> CARD_1:14	VOTE
25	"Cantor-Bernstein Theorem"	=> CARD_1:10	VOTE

Big Formalizations

- Kepler Conjecture (Hales et all, 2014, HOL Light, Isabelle)
- Feit-Thompson (odd-order) theorem
  - Two graduate books
  - Gonthier et all, 2012, Coq
- Compendium of Continuous Lattices (CCL)
  - 60% of the book formalized in Mizar
  - Bancerek, Trybulec et all, 2003
- The Four Color Theorem (Gonthier and Werner, 2005, Coq)

Mid-size Formalizations

- Gödel's First Incompleteness Theorem — Natarajan Shankar (NQTHM), Russell O'Connor (Coq)
- Brouwer Fixed Point Theorem — Karol Pak (Mizar), John Harrison (HOL Light)
- Jordan Curve Theorem — Tom Hales (HOL Light), Artur Kornilowicz et al. (Mizar)
- Prime Number Theorem — Jeremy Avigad et al (Isabelle/HOL), John Harrison (HOL Light)
- Gödel's Second incompleteness Theorem — Larry Paulson (Isabelle/HOL)
- Central Limit Theorem – Jeremy Avigad (Isabelle/HOL)

Large Software Verifications

- seL4 – operating system microkernel
  - Gerwin Klein and his group at NICTA, Isabelle/HOL
- CompCert – a formaly verified C compiler
  - Xavier Leroy and his group at INRIA, Coq
- EURO-MILS – verified virtualization platform
  - ongoing 6M EUR FP7 project, Isabelle
- CakeML – verified implementation of ML
  - Magnus Myreen, HOL4

Substantial Libraries

- Mizar – Topology, Continuous lattices
- HOL Light – Analysis and topology in Euclidean space
- Coq – Finite Algebra (Mathematical Components)
- Isabelle/HOL – Probability and Measure Theory

Central Limit Theorem in Isabelle/HOL

The Top 100 Theorems in Isabelle - Chromium

The Top 100 Theo ×

www.cse.unsw.edu.au/~kleing/top100/#47

```
theorem (in prob_space) central_limit_theorem:
  fixes
    X :: "nat ⇒ 'a ⇒ real" and
    μ :: "real measure" and
    σ :: real and
    S :: "nat ⇒ 'a ⇒ real"
  assumes
    X_indep: "indep_vars (λi. borel) X UNIV" and
    X_integrable: "⋀n. integrable M (X n)" and
    X_mean_0: "⋀n. expectation (X n) = 0" and
    σ_pos: "σ > 0" and
    X_square_integrable: "⋀n. integrable M (λx. (X n x)²)" and
    X_variance: "⋀n. variance (X n) = σ²" and
    X_distrib: "⋀n. distr M borel (X n) = μ"
  defines
    "S n ≡ λx. ∑i<n. X i x"
  shows
    "weak_conv_m (λn. distr M borel (λx. S n x / sqrt (n * σ²)))
        (density lborel std_normal_density)"
```

Sylow's Theorems in Mizar

```
theorem :: GROUP_10:12
  for G being finite Group, p being prime (natural number)
  holds ex P being Subgroup of G st P is_Sylow_p-subgroup_of_prime p;

theorem :: GROUP_10:14
  for G being finite Group, p being prime (natural number) holds
    (for H being Subgroup of G st H is_p-group_of_prime p holds
      ex P being Subgroup of G st
      P is_Sylow_p-subgroup_of_prime p & H is Subgroup of P) &
    (for P1,P2 being Subgroup of G
      st P1 is_Sylow_p-subgroup_of_prime p & P2 is_Sylow_p-subgroup_of_prime p
      holds P1,P2 are_conjugated);

theorem :: GROUP_10:15
  for G being finite Group, p being prime (natural number) holds
    card the_sylow_p-subgroups_of_prime(p,G) mod p = 1 &
    card the_sylow_p-subgroups_of_prime(p,G) divides ord G;
```

Gödel Theorems in Isabelle

```
theorem Goedel_I:
  assumes "¬ {} ⊢ Fls"
  obtains δ where
      "{} ⊢ δ IFF Neg (PfP ⌈δ⌉)"
      "¬ {} ⊢ δ"
      "¬ {} ⊢ Neg δ"
      "eval_fm e δ"
      "ground_fm δ"

theorem Goedel_II:
  assumes "¬ {} ⊢ Fls"
    shows "¬ {} ⊢ Neg (PfP ⌈Fls⌉)"
```

http://afp.sourceforge.net/entries/Incompleteness.shtml

Prime Number Theorem in HOL Light

```
|- ((\n. &(CARD {p | prime p /\ p <= n}) / (&n / log(&n)))
      ---> &1) sequentially
```

Foundational Wars - Set Theory

- Mizar, MetaMath, Isabelle/ZF
- ZFC
- Tarski-Grothendieck (added inaccessible cardinals)
- strong choice
- issues:
  - how to add a type system
  - how to handle higher-order reasoning
  - how to compute

Foundational Wars - Higher-order logic (HOL)

- HOL4, HOL Light, Isabelle/HOL, ProofPower, HOL Zero
- based on polymorphic simply-typed lambda calculus
- but quickly added extensionality and choice (classical)
- weaker than set theory - canonical model is $V_{\omega+\omega} \setminus \{0\}$
- *HOL universe*: U is a set of non-empty sets, such that
    - U is closed under non-empty subsets, finite products and powersets
    - an infinite set $I \in U$ exists
    - a choice function *ch* over U exists (i.e., $\forall X \in U : ch(X) \in X$)
    - gurantees also function spaces ($I \to I$)
- Isabelle adds typeclasses, ad-hoc overloading
- issues:
    - can be too weak
    - not so well known foundations as ZFC
    - the type system does not have dependent types (e.g. matrix over a ring)
    - how to compute

Foundational Wars - Type theory

- Coq, Agda, NuPrl, HoTT
- constructive type theory
- Curry-Howard isomorphism:
  - formulas as types
  - proofs as terms
- proofs are in your universe of discourse!
- two proofs of the same formula might not be equal!
- what does it mean?
- excluded middle avoided, classical math not supported so much
- computation is a big topic
- very rich type system
- lots of research issues for constructivists
- non-experts typically don't have a good idea about the semantics of it all
- *'they have been calling it baroque, but it's almost rococo'* (A. Trybulec)

Foundational Wars - Logical Frameworks

- LF, Twelf, MMT, Isabelle?, Metamath?
- Try to cater for everybody
- Let users encode their logic and inference rules (deep embedding)
- issues:
  - None of them really used
  - maintenance – the embedded systems evolve fast
  - efficiency: Isabelle/Pure ended up enriching its kernel to fit HOL
  - efficiency: things like computation
  - probably needs a lot of investment to benefit multiple foundations
  - more ad-hoc translations between systems are often cheaper to develop

Implementation

- Most systems written in ML (OCAML or SML)
- Sometimes Lisp, Pascal, C++
- LCF approach (Milner): small inference kernel
- isolated by an abstract ML datatype "theorem"
- this means that only a small number of allowed inferences can result in a "theorem"
- Every more complicated procedure has to produce the kernel inferences, to get a "theorem"
- HOL Light - about 400 lines for the whole kernel
- Coq - about 20000 lines

HOL Light kernel - terms and types

```
module Hol : Hol_kernel = struct

  type hol_type = Tyvar of string
                | Tyapp of string *  hol_type list

  type term = Var of string * hol_type
            | Const of string * hol_type
            | Comb of term * term
            | Abs of term * term

  type thm = Sequent of (term list * term)
```

best for math or best for computer science?

- **math proof**

  statements: small, easy to get right
  proofs: intricate, interesting

  top systems: Coq/HOL Light/Mizar

- **cs proof**

  statements: large, easy to get wrong
  proofs: straightforward, mostly boring

  top systems: Coq/Isabelle/HOL4/ACL2/PVS

best for math or best for computer science?

- **math proof**

  statements: small, easy to get right
  proofs: intricate, interesting

  top systems: Coq/HOL Light/Mizar

- **cs proof**

  statements: large, easy to get wrong
  proofs: straightforward, mostly boring

  top systems: Coq/Isabelle/HOL4/ACL2/PVS

one system to rule them all?
should formal proofs be readable?

best for math or best for computer science?

- **math proof**

  statements: small, easy to get right
  proofs: intricate, interesting

  top systems: Coq/HOL Light/Mizar

- **cs proof**

  statements: large, easy to get wrong
  proofs: straightforward, mostly boring

  top systems: Coq/Isabelle/HOL4/ACL2/PVS

one system to rule them all?
should formal proofs be readable?

so . . . which systems?

two worlds:

- Coq
- the HOLs

so . . . which systems?

two worlds, four systems:

- Coq
- the HOLs
  - Isabelle/HOL
  - HOL Light
  - HOL4

so . . . which systems?

two worlds, four systems:

- Coq
- the HOLs
  - Isabelle/HOL
  - HOL Light
  - HOL4

why not also Mizar . . . ?

proofs are ‘too manual’, not enough automation

so . . . which systems?

two worlds, four systems:

- Coq
- the HOLs
  - Isabelle/HOL
  - HOL Light
  - HOL4

why not also Mizar . . . ?

proofs are ‘too manual’, not enough automation
but: source of great ideas

- better foundations (set theory)
- better type system (soft typing)

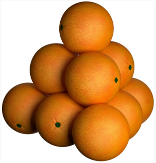
Feit-Thompson in Coq (Georges Gonthier)

- Announcement: `http://www.msr-inria.fr/news/feit-thomson-proved-in-coq/`
- Graph of Coq formalizations: `http://ssr2.msr-inria.inria.fr/~jenkins/current/index.html`
- Final result: `http://ssr2.msr-inria.inria.fr/~jenkins/current/Ssreflect.PFsection14.html#Feit_Thompson`
- Correspondence to the books: `http://ssr2.msr-inria.inria.fr/~jenkins/current/progress.html`

Example: The Flyspeck project

- Kepler conjecture (1611): The most compact way of stacking balls of the same size in space is a pyramid.

$$V = \frac{\pi}{\sqrt{18}} \approx 74\%$$



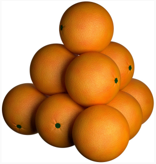
- Proved by Hales in 1998, 300-page proof + computations
- Big: Annals of Mathematics gave up reviewing after 4 years
- But referees of the Annals of Mathematics claim they cannot verify the programs

$$\frac{\begin{matrix} -x_1x_3 - x_2x_4 + x_1x_5 + x_3x_6 - x_5x_6 + \\ +x_2(-x_2 + x_1 + x_3 - x_4 + x_5 + x_6) \end{matrix}}{\sqrt{4x_2 \begin{pmatrix} x_2x_4(-x_2 + x_1 + x_3 - x_4 + x_5 + x_6) + \\ +x_1x_5(x_2 - x_1 + x_3 + x_4 - x_5 + x_6) + \\ +x_3x_6(x_2 + x_1 - x_3 + x_4 + x_5 - x_6) - \\ -x_1x_3x_4 - x_2x_3x_5 - x_2x_1x_6 - x_4x_5x_6 \end{pmatrix}}} < \tan(\frac{\pi}{2} - 0.74)$$

Example: The Flyspeck project

- Kepler conjecture (1611): The most compact way of stacking balls of the same size in space is a pyramid.

$$V = \frac{\pi}{\sqrt{18}} \approx 74\%$$



- Formal proof finished in 2014
- 20000 lemmas in geometry, analysis, graph theory
- All of it at `https://code.google.com/p/flyspeck/`
- All of it computer-understandable and verified in HOL Light:
- `polyhedron s /\ c face_of s ==> polyhedron c`
- However, this took 20 – 30 person-years!

Kepler conjecture formally

```
|- packing V <=>
     (!u v. u IN V /\ v IN V /\ dist(u,v) < &2 ==> u = v)

|- the_kepler_conjecture <=>
     (!V. packing V
       ==> (?c. !r. &1 <= r
           ==> &(CARD(V INTER ball(vec 0,r))) <=
               pi * r pow 3 / sqrt(&18) + c * r pow 2))
```

Kepler conjecture informally

In words, we define the Kepler conjecture to be the following claim: for every packing V, there exists a real number c such that for every real number $r \geq 1$, the number of elements of V contained in an open spherical container of radius r centered at the origin is at most

$$\frac{\pi\, r^3}{\sqrt{18}} + c\, r^2.$$

An analysis of the proof shows that there exists a small computable constant c that works uniformly for all packings V, but we only formalize the weaker statement that allows c to depend on V. The restriction $r \geq 1$, which bounds r away from 0, is needed because there can be arbitrarily small containers whose intersection with V is nonempty.

Parts of Flyspeck

- combination of traditional mathematical argument and three separate bodies of computer calculations.
- nearly a thousand nonlinear inequalities.
- The combinatorial structure of each possible counterexample to the Kepler conjecture is encoded as a plane graph satisfying a number of restrictive conditions. Any graph satisfying these conditions is said to be *tame*.
- A list of all tame plane graphs up to isomorphism has been generated by an exhaustive computer search. The formal statement that every tame plane graph is isomorphic to one of these cases. This was part was done in Isabelle and imported into HOL Light.
- a large collection of linear programs.

Kepler conjecture formally

URL: `https://code.google.com/p/flyspeck/source/browse/trunk/text_formalization/general/the_kepler_conjecture.hl?spec=svn3759&r=3701#69`

```
|-  the_nonlinear_inequalities /\
    import_tame_classification
    ==> the_kepler_conjecture

|- g in PlaneGraphs /\ tame g ==> fgraph g in Archive

(every tame plane graph is isomorphic to a graph
 appearing in the archive)
```

Kepler conjecture formally

- the_nonlinear_inequalities := conjunction of several hundred nonlinear inequalities.
- The domains of these partitioned $\rightarrow$ 23,000 inequalities
- 5k-9k CPU hours on Microsoft Azure cloud and independently on our big server in Nijmegen

Independent verification of Flyspeck

- Mark Adams: HOL Zero system
- more secure than HOL Light, indepedently implemented
- an fast exporter of the HOL Light verifications based on kernel modfications
- verification of every HOL Light kernel step inside HOL Zero
- so far only for the text part (the other parts are much slower)

Flyspeck: What Remains?

- Join the indepent parts more tightly
- Either export the HOL Light parts completely to Isabelle/HOL
- Or implement very fast computation in HOL Light ...
- ... and re-do the Isabelle part in HOL Light
- safe parallelized computing inside HOL Light
- to ensure that the nonlinear parts are merged safely

Flyspeck: Informal and Formal

- The flyspeck book (Dense Sphere Packings):
- http://www.cambridge.org/us/academic/subjects/mathematics/geometry-and-topology/dense-sphere-packings-blueprint-formal-proof
- You can get the source of the book at:
- https://code.google.com/p/flyspeck/source/browse/trunk/#trunk%2Fkepler_tex
- Demo of the informal/formal Wiki at mws.cs.ru.nl/agora_flyspeck/flyspeck/fly_demo

Aligned Formal and Informal Math - Flyspeck

Informal Formal

Definition of [fan, blade] DSKAGVP (fan) [fan ↔ FAN]

Let (V, E) be a pair consisting of a set $V \subset \mathbb{R}^3$ and a set E of unordered pairs of distinct elements of V. The pair is said to be a *fan* if the following properties hold.

1. (CARDINALITY) V is finite and nonempty. [cardinality ↔ fan1]
2. (ORIGIN) $\mathbf{0} \notin V$. [origin ↔ fan2]
3. (NONPARALLEL) If $\{\mathbf{v}, \mathbf{w}\} \in E$, then $\mathbf{v}$ and $\mathbf{w}$ are not parallel. [nonparallel ↔ fan6]
4. (INTERSECTION) For all $\varepsilon, \varepsilon' \in E \cup \{\{\mathbf{v}\} \,:\, \mathbf{v} \in V\}$, [intersection ↔ fan7]

$$C(\varepsilon) \cap C(\varepsilon') = C(\varepsilon \cap \varepsilon').$$

When $\varepsilon \in E$, call $C^0(\varepsilon)$ or $C(\varepsilon)$ a *blade* of the fan.

basic properties

The rest of the chapter develops the properties of fans. We begin with a completely trivial consequence of the definition.

Informal Formal

Lemma [] CTVTAQA (subset-fan)

If (V, E) is a fan, then for every $E' \subset E$, (V, E') is also a fan.

Proof

This proof is elementary.

Informal Formal

Lemma [fan cyclic] XOHLED

[$E(v)$ ↔ set_of_edge] Let (V, E) be a fan. For each $\mathbf{v} \in V$, the set

$$E(\mathbf{v}) = \{\mathbf{w} \in V \,:\, \{\mathbf{v}, \mathbf{w}\} \in E\}$$

is cyclic with respect to $(\mathbf{0}, \mathbf{v})$.

Proof

If $\mathbf{w} \in E(\mathbf{v})$, then $\mathbf{v}$ and $\mathbf{w}$ are not parallel. Also, if $\mathbf{w} \neq \mathbf{w}' \in E(\mathbf{v})$, then

Informal Formal

```
#DSKAGVP?

let FAN=new_definition`FAN(x,V,E) <=> ((UNIONS E) SUBSET V) /\ graph(E) /\ fan1(x,V,E) /\ fan2(x,V,
fan6(x,V,E)/\ fan7(x,V,E)`;;
```


basic properties

The rest of the chapter develops the properties of fans. We begin with a completely trivial consequence of the definition.

Informal Formal

```
let CTVTAQA=prove(`!(x:real^3) (V:real^3->bool) (E:(real^3->bool)->bool) (E1:(real^3->bool)->bool)
FAN(x,V,E) /\ E1 SUBSET E
==>
FAN(x,V,E1)`,

REPEAT GEN_TAC
THEN REWRITE_TAC[FAN;fan1;fan2;fan6;fan7;graph]
THEN ASM_SET_TAC[]);;
```

Informal Formal

```
let XOHLED=prove(`!(x:real^3) (V:real^3->bool) (E:(real^3->bool)->bool) (v:real^3).
FAN(x,V,E) /\ v IN V
==> cyclic_set (set_of_edge v V E) x v`,

MESON_TAC[CYCLIC_SET_EDGE_FAN]);;
```

Flyspeck: Informal and Formal Used to Learn Formal Parsing

- Demo of the probabilistic/semantic parser trained on informal/formal Flyspeck pairs:
- `http://colo12-c703.uibk.ac.at/hh/parse.html`
- The linguistic/semantic methods explained in `http://dx.doi.org/10.1007/978-3-319-22102-1_15`
- Compare with Wolfram Alpha:
- `https://www.wolframalpha.com/input/?i=sin+0+*+x+%3D+cos+pi+%2F+2`

Online parsing system trained on Flyspeck informal/formal pairs

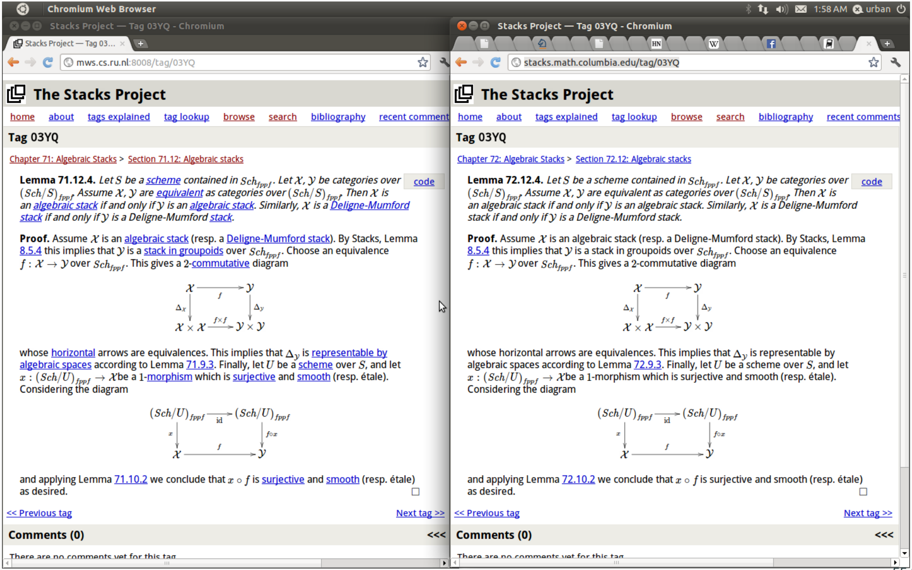
- "`sin ( 0 * x ) = cos pi / 2`"
- produces 16 parses, we order them by their probability resulting from training on related informal/formal pairs
- 11 of the 16 parses get type-checked by HOL Light as follows
- with all but three being proved by HOL(y)Hammer

```
sin (&0 * A0) = cos (pi / &2) where A0:real
sin (&0 * A0) = cos pi / &2 where A0:real
sin (&0 * &A0) = cos (pi / &2) where A0:num
sin (&0 * &A0) = cos pi / &2 where A0:num
sin (&(0 * A0)) = cos (pi / &2) where A0:num
sin (&(0 * A0)) = cos pi / &2 where A0:num
csin (Cx (&0 * A0)) = ccos (Cx (pi / &2)) where A0:real
csin (Cx (&0) * A0) = ccos (Cx (pi / &2)) where A0:real^2
Cx (sin (&0 * A0)) = ccos (Cx (pi / &2)) where A0:real
csin (Cx (&0 * A0)) = Cx (cos (pi / &2)) where A0:real
csin (Cx (&0) * A0) = Cx (cos (pi / &2)) where A0:real^2
```

The Stacks project: a first version of an automated concept linker

- The Stacks project: a large growing open-source book on algebraic stacks (about 5k pages in PDF)
- `http://stacks.math.columbia.edu/`
- The definition of algebraic stack: `http://stacks.math.columbia.edu/tag/03YQ`
- Our experimental auto-linking version of the web presentation:
- `http://mws.cs.ru.nl:8008/tag/03YQ`
- Can we link it with reasonably high success rate?
- Can we turn it into formal-math code (written in Mizar/Isabelle/HOL/Coq)?
- Can we verify and/or prove automatically a nontrivial fraction of the lemmas?

The Stacks project: a first version of an automated concept linker



ProofWiki vs Mizar Example - The ProofWiki version

ProofWiki is an informal-but-very-controlled proof corpus, sample proof:
`https://proofwiki.org/wiki/Zero_Element_is_Unique`
== Theorem ==
Let $(S, \circ)$ be an [[Definition:Algebraic Structure|algebraic structure]] that has a [[Definition:Zero Element|zero element]] $z \in S$. Then z is unique.
== Proof ==
Suppose z_1 and z_2 are both zeroes of $(S, \circ)$.
Then by the definition of [[Definition:Zero Element|zero element]]:
$z_2 \circ z_1 = z_1$ by dint of z_1 being a zero;
$z_2 \circ z_1 = z_2$ by dint of z_2 being a zero.
So $z_1 = z_2 \circ z_1 = z_2$.
So $z_1 = z_2$ and there is only one zero after all.
{{qed}}
// *NB: Informal proofs are buggy!*

ProofWiki vs Mizar Example - The Mizar version

Existing Mizar theorem – slightly different from ProofWiki:

```
Th9:  e1 is_a_left_unity_wrt o & e2 is_a_right_unity_wrt o
implies e1 = e2
proof
  assume that A1:  e1 is_a_left_unity_wrt o and
  A2:  e2 is_a_right_unity_wrt o;
  thus e1 = o.(e1,e2) by A2,Def6 .= e2 by A1,Def5;
end;
```

Mizar equivalent of the ProofWiki theorem – all steps proved automatically:

```
z1 is_a_unity_wrt o & z2 is_a_unity_wrt o implies z1 = z2
proof
  assume that A1:  z1 is_a_unity_wrt o and
  A2:  z2 is_a_unity_wrt o;
  A3:  o.(z2,z1) = z1 by Th3,A2; ::[ATP]
  A4:  o.(z2,z1) = z2 by Def 6,Def 7,A1,A3; ::[ATP]
  hence z1 = z2 by Th9,A1,Def 7,A2; ::[ATP]
end;
```