formal proof with the computer

Freek Wiedijk
Radboud University Nijmegen

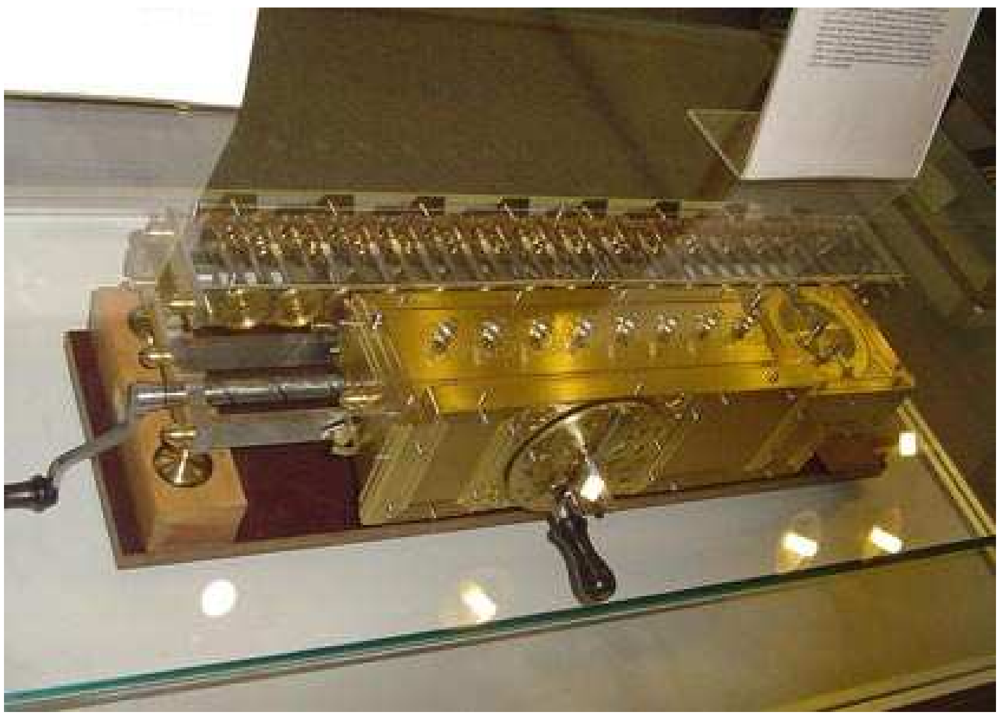
University of Groningen
2010 03 17, 16:15

a very brief history of formal proof

calculus ratiocinator

Gottfried Leibniz, 1646–1716

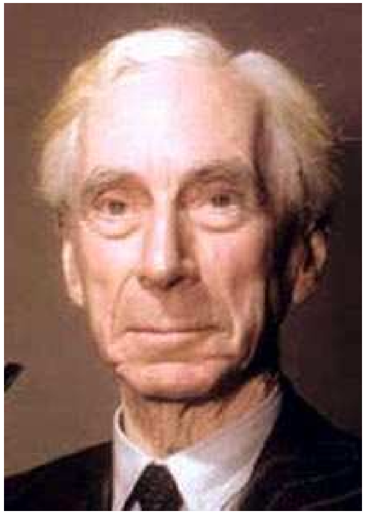
replace reasoning with calculation: **Calculemus!**



Alfred North Whitehead & Bertrand Russell

logicism =
'mathematics can be seen as part of logic'

Principia Mathematica, 1910–1913
'arithmetic on the real numbers'



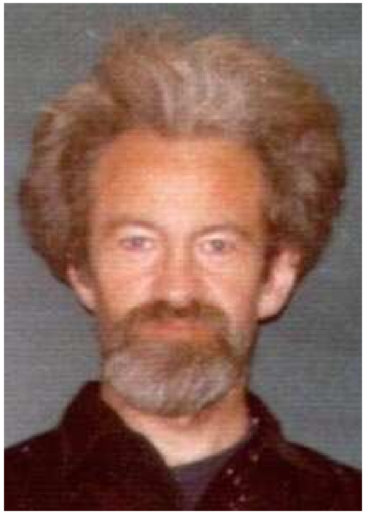
'$1 + 1 = 2$' only proved on page 360

Automath

N.G. de Bruijn, 1968:

proof assistant
= interactive theorem prover
= proof checker

Bert Jutting, 1979:
translation of **Grundlagen** by Edmund Landau



143 pages in German
'arithmetic on the real numbers'

full night of computer time in 1979
less than half a second in 2010

what formal proof looks like

between mathematics and computer science

mathematical journal article or textbook
(mathematical vernacular)

$\updownarrow^{?}$

formalization

$\|$

source code of a computer program

first example: a poem by Marjolein Kool

Een bolleboos riep laatst met zwier
gewapend met een vel A-vijf:
Er is geen allergrootst getal,
dat is wat ik bewijzen ga.
Stel, dat ik u nu zou bedriegen
en hier een potje stond te jokken,
dan ik zou zonder overdrijven
het grootste kunnen op gaan noemen.
Maar ben ik klaar, roept u gemeen:
‘Vermeerder dat getal met twee!’
En zien we zeker en gewis
dat dit toch niet het grootste was.
En gaan we zo nog door een poos,
dan merkt u: dit is onbegrensd.
En daarmee heb ik q.e.d.
Ik ben hier diep gelukkig door.
‘Zo gaan’, zei hij voor hij bezwijmde,
‘bewijzen uit het ongedichte’.

```
theorem
  not ex n st for m holds n >= m
proof

  assume not thesis;
  then consider n such that
A1: for m holds n >= m;

  set n’ = n + 2;

  n’ > n by XREAL_1:31;

  then not for m holds n >= m;

  hence contradiction by A1;

end;
```

second example: an IMO puzzle

Problem. For all x and y

$$f(x+y)+f(x-y)=2f(x)g(y)$$

$|f(x)| \leq 1$ for all x, and f is not identically zero.
Prove that $|g(x)| \leq 1$ for all x.

Solution. Let k be the least upper bound for $|f(x)|$.
Suppose $|g(y)| > 1$ for some y. Then

$$2k \geq |f(x+y)| + |f(x-y)| \geq |f(x+y)+f(x-y)| = 2|f(x)||g(y)|$$

This implies that $|f(x)| \leq k/|g(y)|$.
In other words $k/|g(y)|$ is an upper bound for $|f(x)|$.
But this is less than k. Contradiction. □

```
let IMO = prove
 (`!f g. (!x y. f(x + y) + f(x - y) = &2 * f(x) * g(y)) /\
         ~(!x. f(x) = &0) /\
         (!x. abs(f(x)) <= &1)
         ==> !x. abs(g(x)) <= &1`,
  let LL = REAL_ARITH `&1 < k ==> &0 < k` in
  REPEAT STRIP_TAC THEN SPEC_TAC(`x:real`,`y:real`) THEN
  ABBREV_TAC `k = sup (IMAGE (\x. abs(f(x))) (:real))` THEN
  MP_TAC(SPEC `IMAGE (\x. abs(f(x))) (:real)` SUP) THEN
  ASM_SIMP_TAC[FORALL_IN_IMAGE; EXISTS_IN_IMAGE; IN_UNIV] THEN
  ANTS_TAC THENL [ASM SET_TAC[]; STRIP_TAC] THEN
  SIMP_TAC[GSYM REAL_NOT_LT; GSYM NOT_EXISTS_THM] THEN STRIP_TAC THEN
  FIRST_X_ASSUM(MP_TAC o SPEC `k / abs(g(y:real))`) THEN
  SIMP_TAC[NOT_IMP; NOT_FORALL_THM] THEN CONJ_TAC THENL
   [ASM_MESON_TAC[REAL_LE_RDIV_EQ; REAL_ABS_MUL; LL;
      REAL_ARITH `u + v = &2 * z /\ abs u <= k /\ abs v <= k ==> abs z <= k`];
    ASM_MESON_TAC[REAL_NOT_LE; REAL_LT_LDIV_EQ; REAL_LT_LMUL; REAL_MUL_RID; LL;
      REAL_ARITH `~(z = &0) /\ abs z <= k ==> &0 < k`]]);;
```

```
Theorem imo1972:
 forall f g: R -> R,
 (forall x y: R,  f (x + y) + f (x - y) = 2 * f x * g y) ->
 ~ (forall x: R, f x = 0) ->
 Rfbound f 1 -> Rfbound g 1.
intros f g Eq Nz Rf1 y.
case (Rle_or_lt (Rabs (g y)) 1); intros H; auto.
case (Rflub_def f _ Rf1); clear Rf1.
intros k [Kb Kl].
assert (Kp: 0 < k); [idtac | clear Nz].
 case (Rle_or_lt k 0); auto; intros H1.
 case Nz; intros x.
 assert (HH: forall x, Rabs x = 0 -> x = 0);
  [intros x1; ffourier | idtac].
 apply HH; clear HH.
 apply Rle_antisym; [idtac | apply Rabs_pos].
 assert (HH := Kb x); ffourier.
```

etcetera

```
theorem
  (for x,y holds f.(x+y)+f.(x-y)=2*f.x*g.y) &
  (ex x st f.x<>0) & (for x holds abs(f.x)<=1)
  implies for x holds abs(g.x)<=1
  proof
    assume that
A1: for x,y holds f.(x+y)+f.(x-y)=2*f.x*g.y;
    given z such that
A2: f.z<>0;
    assume
A3: for x being Element of REAL holds abs(f.x)<=1;
    let y such that
A4: abs(g.y) > 1;
    set X = rng abs f, k = upper_bound X, D = abs(g.y);
A5: abs(g.y) > 0 by A4,XREAL_1:2;
A6: X is bounded_above
    proof
```

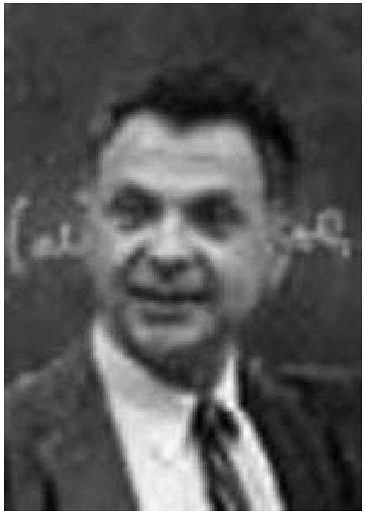
etcetera

. . . and in Isabelle

```
theorem IMO:
   assumes "ALL (x::real) y. f(x + y) + f(x - y) = (2::real) * f x * g y"
   and "~ (ALL x. f(x) = 0)"
   and "ALL x. abs(f x) <= 1"
   shows "ALL y. abs(g y) <= 1"
proof (clarify, rule leI, clarify)
   obtain k where "isLub UNIV z. EX x. abs(f x) = z k"
     by (subgoal_tac "EX k. ?P k", force, insert prems,
         auto intro!: reals_complete isUbI setleI)
   hence "ALL x. abs(f x) <= k"
     by (intro allI, rule isLubD2, auto)
   fix y
   assume "abs(g y) > 1"
   have "ALL x. abs(f x) <= k / abs(g y)"
   proof
     fix x
     have "2 * abs(g y) * abs(f x) = abs(f(x + y) + f(x - y))"
```

etcetera

formal proof versus artificial intelligence



$$a \vee b = b \vee a$$

$$a \vee (b \vee c) = (a \vee b) \vee c$$

$$\neg(\neg(a \vee b) \vee \neg(a \vee \neg b)) = a$$

Herbert Robbins, 1933: always a Boolean algebra?

William McCune + EQP, 1996: 34 line proof

eight days of computer time

formal proof versus computer algebra

```
> 2*infinity-infinity, 2*x-x;
```

$$\text{undefined}, x$$

```
> subs(x=infinity, 2*x-x);
```

$$\text{infinity}$$

```
> int(1/(1-x),x) = int(simplify(1/(1-x)),x);
```

$$-\ln(1-x) = -\ln(-1+x)$$

```
> evalf(subs(x=-1, %));
```

$$-0.6931471806 = -0.6931471806 - 3.141592654i$$

the de Bruijn criterion

all software has bugs . . .

why trust a proof checker?

> [. . .] This is one of the reasons for keeping AUTOMATH as primitive as possible. [. . .]

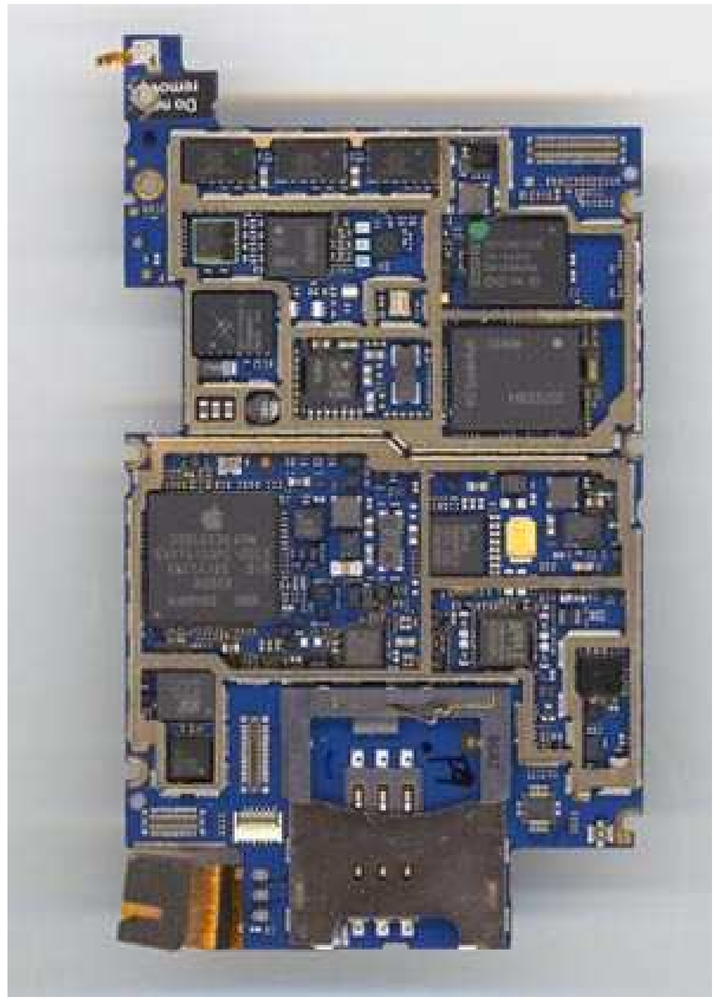
- small independent checker(s)
  Ivy system for Otter/Prover9

- small proof checking **kernel** inside the system
  LCF architecture

  Robin Milner, 1972

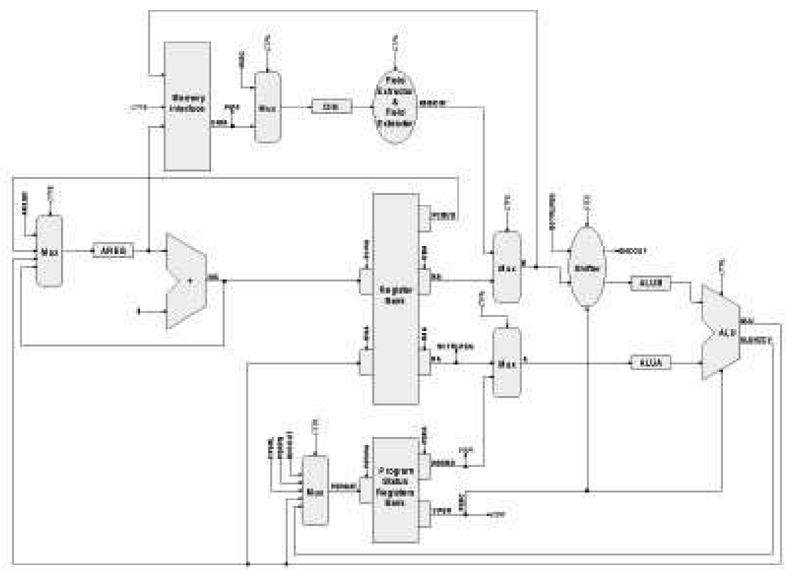
state of the art in formal proof

the ARM microprocessor

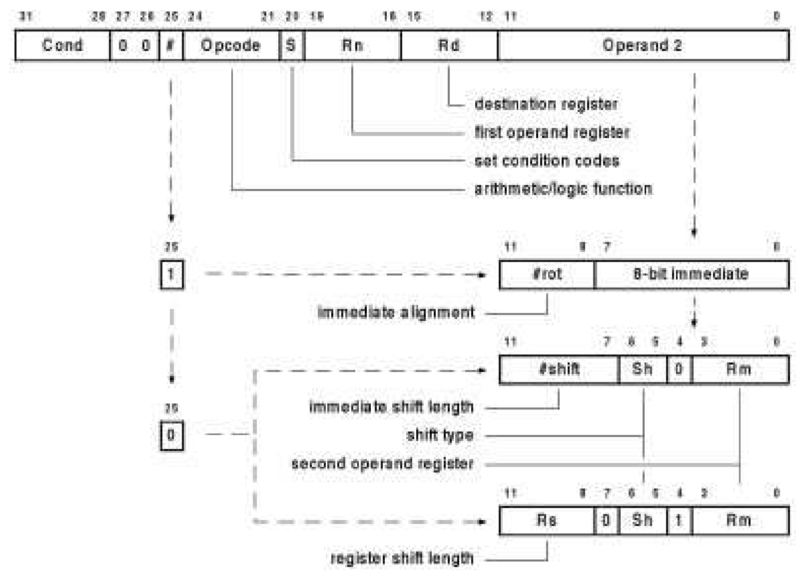


. . . proved correct in HOL4 by Anthony Fox

University of Cambridge, 2002

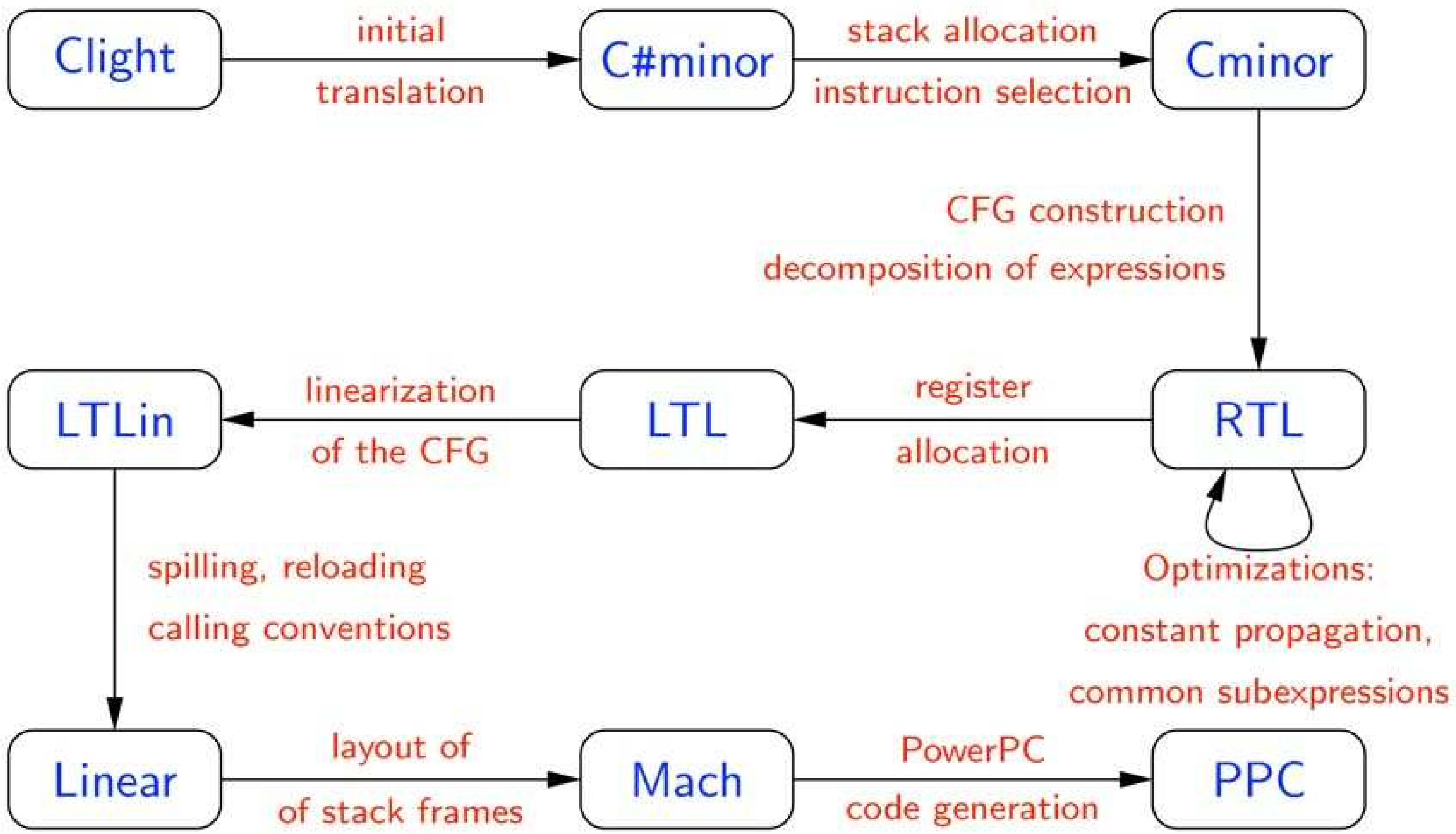


ARM6 architecture



ARMv3 instruction set

an optimizing compiler



... proved correct in Coq by Xavier Leroy

INRIA, France, 2006

optimizing compiler

... *programmed* in Coq's functional language

... compiling *from* C

... compiling *to* machine code

```
Theorem transf_c_program_correct:
  forall p tp beh,
  transf_c_program p = OK tp ->
  not_wrong beh ->
  Csem.exec_program p beh ->
  Asm.exec_program tp beh.
```

microkernel
≈ hypervisor

seL4
implementation of L4 kernel
programmed in C

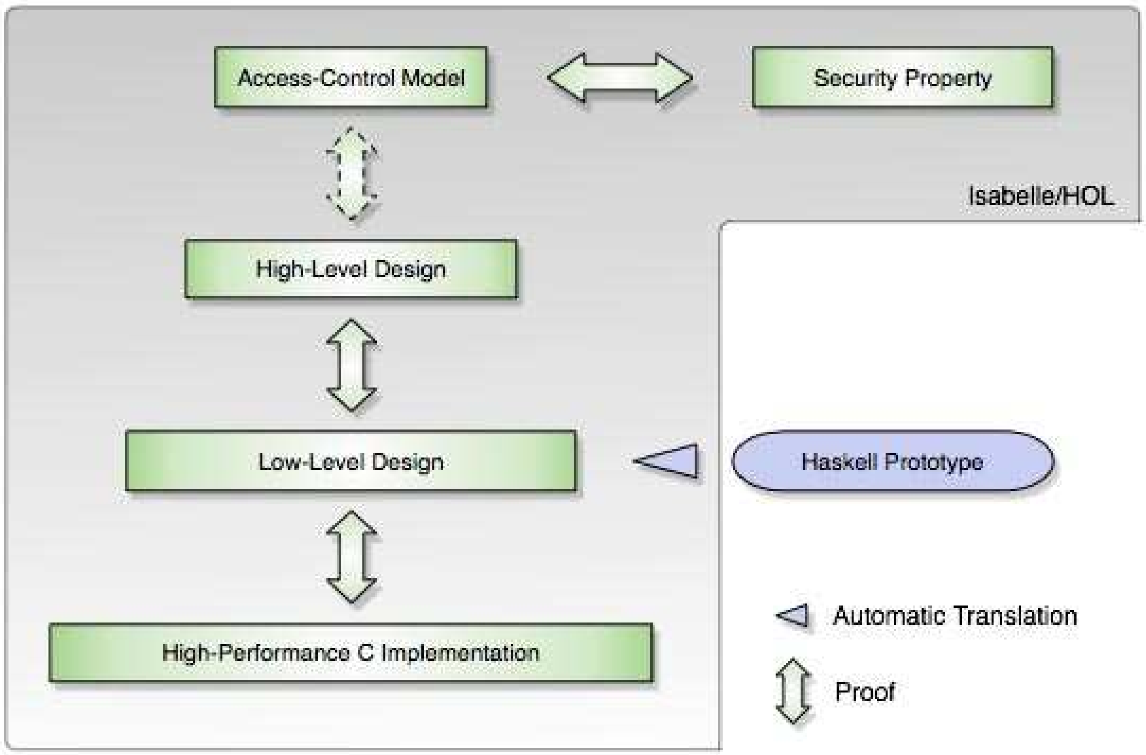
8,700 lines of C + 600 lines of ARM assembly
2 person-years for the implementation

200,000 lines of Isabelle
20 person-years for the correctness proof

160 bugs before verification
0 bugs after verification (?)

. . . proved correct in Isabelle by Gerwin Klein

NICTA, Australia, 2009



the prime number theorem

Jacques Hadamard, Charles Jean de la Vallée-Poussin, 1896

$$\lim_{n\to\infty} \frac{\pi(n)}{\left(\frac{n}{\ln n}\right)} = 1$$

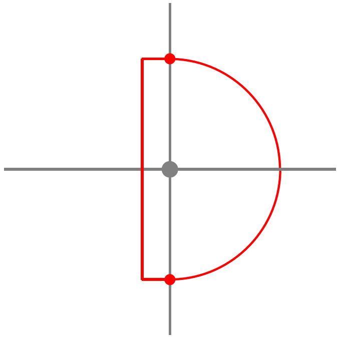
number of primes $\leq$ given number

$$\begin{aligned}
\pi(1000000000000) &= 37607912018 \\
\frac{1000000000000}{\ln(1000000000000)} &= 36191206825.27\ldots \\
\int_2^{1000000000000} \frac{dx}{\ln(x)} &= 37607950279.75\ldots
\end{aligned}$$

... proved correct in HOL Light by John Harrison

Intel Oregon research center, 2008

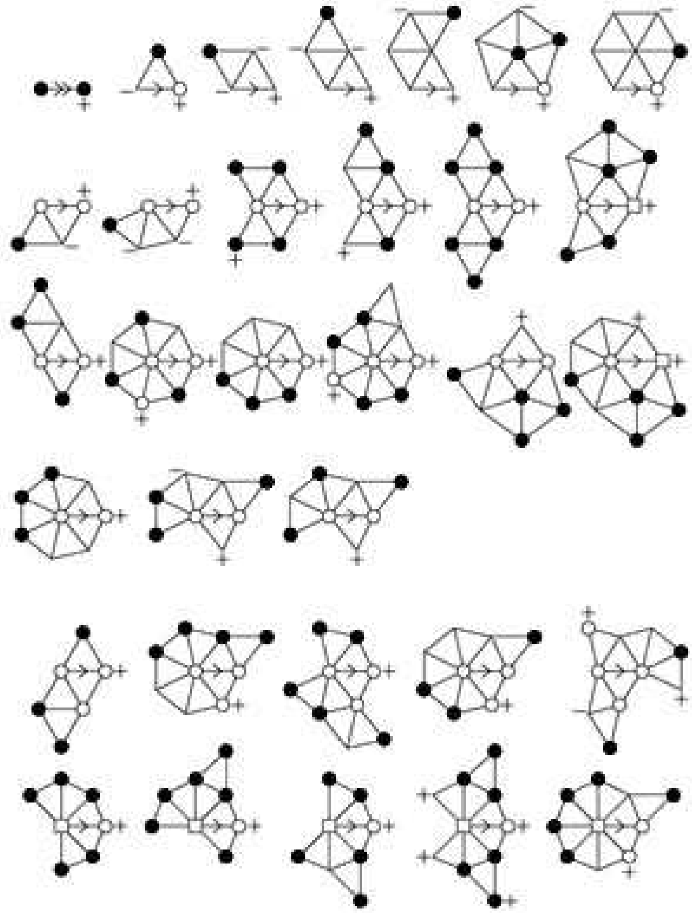
$$2\pi i F(w) = \int_{\Gamma} F(z+w) N^z \left(\frac{1}{z} + \frac{z}{R^2} \right) dz$$



etcetera

```
  ALL_TAC] THEN
SUBGOAL_THEN
 `((\z. f(w + z) * Cx(&N) cpow z * (Cx(&1) / z + z / Cx(R) pow 2))
  has_path_integral (Cx(&2) * Cx pi * ii * f(w))) (A ++ B)`
ASSUME_TAC THENL
 [MP_TAC(ISPECL
```

etcetera



... proved correct in Coq by Georges Gonthier

```
Lemma unavoidability : reducibility ->
  forall g, ~ minimal_counter_example g.
Proof.
move=> Hred g Hg; case: (posz_dscore Hg) => x Hx.
have Hgx: valid_hub x by split.
have := (Hg : pentagonal g) x;
  rewrite 7!leq_eqVlt leqNgt.
rewrite exclude5 ?exclude6 ?exclude7 ?exclude8
  ?exclude9 ?exclude10 ?exclude11 //.
case/idP; apply: (@dscore_cap1 g 5) => {x n Hn Hx Hgx}// y.
pose x := inv_face2 y; pose n := arity x.
have ->: y = face (face x) by rewrite /x /inv_face2 !Enode.
rewrite (dbound1_eq (DruleFork (DruleForkValues n))) // leqz_nat.
case Hn: (negb (Pr58 n)); first by rewrite source_drules_range //.
have Hrp := no_fit_the_redpart Hred Hg.
apply: (check_dbound1P (Hrp the_quiz_tree) _
  (exact_fitp_pcons_ Hg x)) => //.
rewrite -/n; move: n Hn; do 9 case=> //.
Qed.
```

the Feit-Thompson project of Georges Gonthier

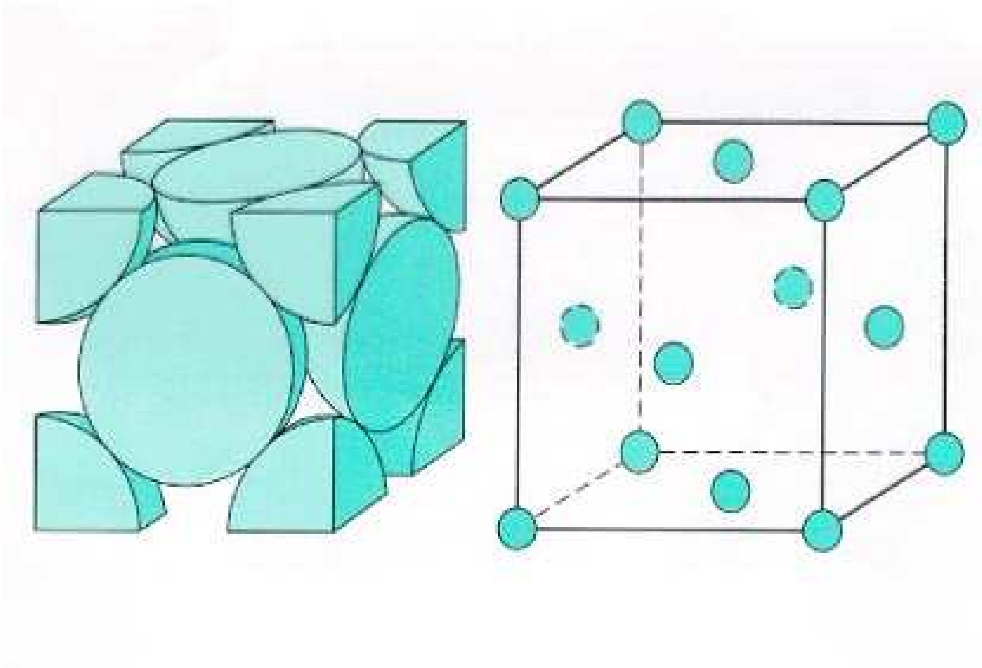
Feit-Thompson theorem

every finite group of odd order is solvable

'It takes a professional group theorist about a year of hard work to *understand* the proof completely.'

— Wikipedia

the Flyspeck project of Tom Hales



Kepler conjecture, 1661: is this the densest sphere packing possible?
Tom Hales, 1998: yes!

3 gigabytes of data, couple of months of computer time
referees 99% certain that everything is correct

why current systems for formal proof are not perfect yet

all of the undergraduate curriculum?

---

de Bruijn factor

= ratio in size between (gzipped) formal and textbook mathematics

$$\approx 4$$

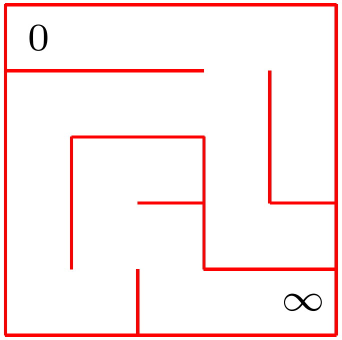
de Bruijn factor in time

$$\approx 2\tfrac{1}{2}\,\frac{\text{person-year}}{\text{megabyte}} \approx 1\,\frac{\text{week}}{\text{textbook page}}$$

formalizing *all* basic mathematics: the **Drake equation**

$$12 \cdot 400 \cdot 3 \cdot 4 \cdot \frac{2\frac{1}{2}}{1024} \approx 140 \text{ person-years} \approx 15 \cdot 10^6 \text{ €} \approx 1 \text{ Hollywood movie}$$

procedural versus declarative proofs



- **procedural**

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

  HOL4, HOL Light, Coq, Isabelle old style, PVS, B method

- **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 new style, ACL2

incompatible foundations

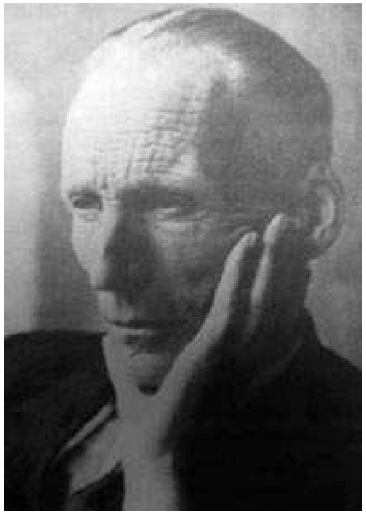
- **set theory**
  Mizar, Isabelle/ZF, B method
  often untyped

- **higher order logic**
  HOL4, HOL Light, Isabelle/HOL, PVS
  weak version of set theory ($V_{\omega+\omega}$), typed

- **primitive recursive arithmetic**
  ACL2
  very weak foundation (no $\exists$), untyped, computational

- **type theory**
  Coq
  as strong as set theory, typed, computational, intuitionistic

halting problem



$\forall M \in$ Turing machines

$\big((M \text{ terminates}) \vee \neg(M \text{ terminates})\big)$

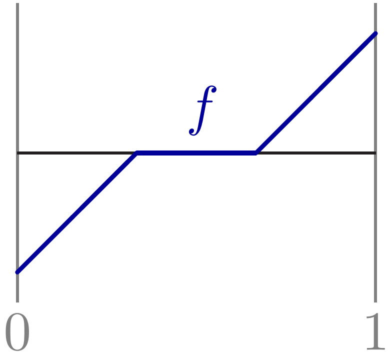
is unprovable

intermediate value theorem

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

$\big(f(0) < 0 \wedge f(1) > 0 \Rightarrow \exists x \in (0,1)\ (f(x) = 0)\big)$

is unprovable



undefinedness

$$\frac{1}{0} = 0 \textbf{ ?}$$

HOL Light . . . . provable

Coq . . . . not provable, negation not provable

IMPS . . . . negation provable

PVS . . . . not a correct formula

$$\forall x \in \mathbb{R} \left(x \neq 0 \Rightarrow \frac{1}{x} \neq 0\right) \textbf{ ?}$$

$$\forall x \in \mathbb{R} \left(x = 0 \lor \frac{1}{x} \neq 0\right) \textbf{ ?}$$

$$\forall x \in \mathbb{R} \left(\frac{1}{x} \neq 0 \lor x = 0\right) \textbf{ ?}$$

formal libraries

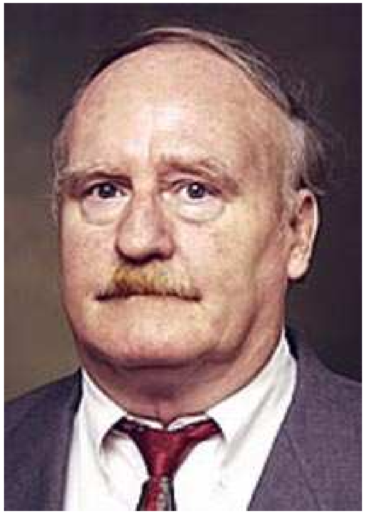
- *beautifully integrated, but made by an isolated genius*
  - John Harrison's HOL Light library
  - Georges Gonthier's Ssreflect library
- *made by a whole community, but not well integrated*
  - Mizar's MML
  - Coq's contribs
  - Isabelle's AFP

Nijmegen's MathWiki project just started
1 postdoc + 1 PhD student

formalizations + 'Proof General on the web' + 'Wikipedia for math'
Coq + Mizar + Isabelle + . . .

QED manifesto

anonymous 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. [...]

The QED project will be a major scientific undertaking requiring the cooperation and effort of hundreds of deep mathematical minds, considerable ingenuity by many computer scientists, and broad support and leadership from research agencies.

formal proof as extreme mathematics

Coq proofs are developed interactively using a number of tactics as elementary proof steps. The sequence of tactics used constitutes the proof script. Building such scripts is surprisingly addictive in a videogame kind of way, but reading and reusing them when specifications change is difficult.

— Xavier Leroy, *On proving in Coq*

everyday reasoning : mathematics = mathematics : formal proof

formal proof = mathematics^2 / informal reasoning